NEAR EAST UNIVERSITY

Faculty Of Engineering

Department Of Computer Engineering

Stock Control Program with Delphi For Computer Company

Graduating Project
COM 400

Student: Seniha Direl

Supervisor : Asst.Prof.Dr. Elbrus Immanov

Nicosia 2007

ACKNOWLADGEMENTS

"Firstly, I would like to thank to my supervisor Mr Elbrus Immanov for his invaluable advise and recommendation for finishing my project properly also,for his support ...

Second,I thank my family for their endless tolerance and support during the preparation of this project also my friend Kerime Durmaz for their patience .

Althougt, I encountered many problems in writing program. In that times, My friend, who helped me realising applied Delphi programming for completing my project isYusuf Alak . My sincere thanks to him and others my friends

I thank all the staff of the faculty of engineering for giving facilities to practise, teaching and solving problem in my comlete undergraduation program

ABSTRACT

The aim of this Project is to record the stock device for any Computer Technic service Company . The program was prepared by using Delphi 6 programming and using BDE(Borland Databade Engine) . Delphi is a programming language . Delphi and BDE works together .

This project consist of many different pages but most of them depended each other Initially, SIGN IN form comes to screen. Afterwards the Exchange foreign money calculator comes to screen.them we will see the main form This form contains 11 main menu and 24 submenu . and the Accounting Informations part.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

It is a programme that saves the following information :the list of the computers that come to the technical services of the computer firm for service ,the tools that are used during service period ,the service time,their cost ,the work that has done and things like that.It also saves the information and the payment of the fixed customers,the stock information which firm they are bought and their prices.It shows the cash balance of firm and also back up the data at the any time whenever wanted and saves it is a Yedek file.

The project consist of the introduction,3 chapters ,conclusion ,Appendix A and Appendix B.

Chapter one describes the introduction for delphi and explain environment of delphi.

Chapter two describes the database that use delphi programming language

Chapter three explain the computer sale and technical service program with all form .

Appendix A is included the source code of programme.

Appendix B is included the database table that use the programme

CHAPTER 1

1.BASIC CONCEPT OF DELPHI

1.1.Introduction to Delphi

Although I am not the most experienced or knowledgeable person on the forums I thought it was time to write a good introductory article for Delphi

1.2.What is Delphi?

Delphi is a Rapid Application Development (RAD) environment. It allows you to drag and drop components on to a blank canvas to create a program. Delphi will also allow you to use write console based DOS like programs.

Delphi is based around the Pascal language but is more developed object orientated derivative. Unlike Visual Basic, Delphi uses punctuation in its basic syntax to make the program easily readable and to help the compiler sort the code. Although Delphi code is not case sensitive there is a generally accepted way of writing Delphi code. The main reason for this is so that any programmer can read your code and easily understand what you are doing, because they write their code like you write yours.

For the purposes of this series I will be using Delphi 6. Delphi 6 provides all the tools you need to develop, test and deploy Windows applications, including a large number of so-called reusable components.
Borland Delphi, provides a cross platform solution when used with Borland Kylix - Borland's RAD tool for the Linux platform.

1.2.1.Delphi Compliers

There are two types complier for Delphi

- **Turbo Delphi** : Free industrial strength Delphi RAD (Rapid Application Development) environment and compiler for Windows. It comes with 200+ components and its own Visual Component Framework.

- **Turbo Delphi for .NET**: Free industrial strength Delphi application development environment and compiler for the Microsoft .NET platform.

1.2.2. What kind of programming can you do with Delphi?

The simple answer is "more or less anything". Because the code is compiled, it runs quickly, and is therefore suitable for writing more or less any program that you would consider a candidate for the Windows operating system.

You probably won't be using it to write embedded systems for washing machines, toasters or fuel injection systems, but for more or less anything else, it can be used (and the chances are that probably someone somewhere has!)

Some projects to which Delphi is suited:

- Simple, single user database applications
- Intermediate multi-user database applications
- Large scale multi-tier, multi-user database applications
- Internet applications
- Graphics Applications
- Multimedia Applications
- Image processing/Image recognition
- Data analysis
- System tools
- Communications tools using the Internet, Telephone or LAN
- Web based applications

This is not intended to be an exhaustive list, more an indication of the depth and breadth of Delphi's applicability. Because it is possible to access any and all of the Windows API, and because if all else fails, Delphi will allow you to drop a few lines of assembler code directly into your ordinary Pascal instructions, it is possible to do more or less anything. Delphi can also be used to write Dynamically Linked Libraries (DLLs) and can call out to DLLs written in other programming languages without difficulty.

Because Delphi is based on the concept of self contained Components (elements of code that can be dropped directly on to a form in your application, and exist in object form, performing their function until they are no longer required), it is possible to build applications very rapidly. Because Delphi has been available for quite some time, the number of pre-written components has been increasing to the point that now there is a component to do more or less anything you can imagine. The job of the programmer has become one of gluing together appropriate components with code that operates them as required.

1.2.3.History Of Delphi

Delphi was one of the first of what came to be known as "RAD" tools, for Rapid Application Development, when released in 1995 for the 16-bit Windows 3.1 . Delphi 2, released a year later, supported 32-bit Windows environments, and a C++ variant, C++ Builder , followed a few years after.

The chief architect behind Delphi, and its predecessor Turbo Pascal , was Anders Hejlsberg until he was headhunted in 1996 by Microsoft , where he worked on Visual J++ and subsequently became the chief designer of C Sharp programming language|C# and a key participant in the creation of the Microsoft .NET Framework.

In 2001 a Linux version known as Kylix programming tool|Kylix became available. However, due to low quality and subsequent lack of interest, Kylix was abandoned after version 3.

Support for Linux and Windows cross platform development (through Kylix and the CLX component library) was added in 2002 with the release of Delphi 6.

Delphi 8, released December 2003, was a .NET -only release that allowed developers to compile Delphi Object Pascal code into .NET Microsoft Intermediate Language|MSIL . It was also significant in that it changed its IDE for the first time, from the multiple-floating-window-on-desktop style IDE to a look and feel similar to Microsoft's Visual Studio.NET.

Although Borland fulfilled one of the biggest requests from developers (.NET support), it was criticized both for making it available too late, when a lot of former Delphi developers had already moved to C#, and for focusing so much on backward compatibility that it was not very easy to write new code in Delphi. Delphi 8 also lacked significant high-level features of the c sharp|C# language, as well as many of the more appealing features of Microsoft's Visual Studio IDE. (There were also concerns about the future of Delphi Win32 development. Because Delphi 8 did not support Win32, Delphi 7.1 was included in the Delphi 8 package.)

The next version, Delphi 2005 (Delphi 9), included the Win32 and .NET development in a single IDE, reiterating Borland's commitment to Win32 developers. Delphi 2005 includes design-time manipulation of live data from a database. It also includes an improved IDE and added a "for ... in" statement (like C#'s foreach) to the language. However, it was criticized by some for its bugs; both Delphi 8 and Delphi 2005 had stability problems when shipped, which were only partially resolved in service packs.

In late 2005 , Delphi 2006 was released and federated development of C# and Delphi.NET, Delphi Win32 and C++ into a single IDE. It was much more stable than Delphi 8 or Delphi 2005 when shipped, and improved even more after the service packs and several hotfixes.

On February 8 , 2006 , Borland announced that it was looking for a buyer for its IDE and database line of products, which include Delphi, to concentrate on its Application Lifecycle Management|ALM line. The news met with voluble optimism from the remaining Delphi users.

On September 6 , 2006, The Developer Tools Group (the working name of the not yet spun off company) of Borland Software Corporation released single language versions

of Borland Developer Studio, bringing back the popular "Turbo" moniker. The Turbo product set includes Turbo Delphi for Win32, Turbo Delphi for .NET, Turbo C++, and Turbo C#. Each version is available in two editions: "Explorer"—a free downloadable version—and "Professional"—a relatively cheap (US$399) version which opens access to thousands of third-party components. Unlike earlier "Personal" editions of Delphi, new "Explorer" editions can be used for commercial development.

On November 14 , 2006, Borland announced the cancellation of the sale of its Development tools; instead of that it would spin them off into an independent company named "CodeGear"

1.2.4.Advantages&Disadvantages Delphi

==Advantages==

Delphi exhibits the following advantages:

- Rapid Application Development (RAD)
- Based on a well-designed language - high-level and strongly typed, with low-level escapes for experts
- A large community on Usenet and the World Wide Web (e.g. news://newsgroups.borland.com and Borland's web access to Delphi)
- Can compile to a single executable, simplifying distribution and reducing DLL versioning issues
- Many VCL and third-party components (usually available with full source code) and tools (documentation, debug tools, etc.)
- Quick optimizing compiler and ability to use assembler code
- Multiple platform native code from the same source code
- High level of source compatibility between versions
- Cross Kylix - a third-party toolkit which allows you to compile native Kylix/Linux applications from inside the Windows Delphi IDE, hence easily enabling dual-platform development and deployment

- Cross FBC - a sister project to CrossKylix, which enables you to cross-compile your Windows Delphi applications to multi-platform targets - supported by the Free Pascal compiler - without ever leaving the Delphi IDE
- Class helpers to bridge functionality available natively in the Delphi RTL, but not available in a new platform supported by Delphi
- The language's object orientation features only class- and interface-based Polymorphism in object-oriented programming|polymorphism

Disadvantages

- Limited cross-platform capability for Delphi itself. Compatibles provide more architecture/OS combinations
- Access to platform and third party libraries require header files to be translated to Pascal. This creates delays and introduces the possibilities of errors in translation.
- There are fewer published books on Delphi than on other popular programming languages such as C++ and C#
- A reluctance to break any code has lead to some convoluted language design choices, and orthogonality and predictability have suffered

1.3. Delphi 6 Editions

There are 3 editions in Delphi 6 :

- **Delphi Personal** - makes learning to develop non-commercial Windows applications fast and fun. Delphi 6 Personal makes learning Windows development easy with drag-and-drop visual programming.
- **Delphi Professional** - adds the tools necessary to create applications with the latest Windows® ME/2000 look-and-feel. Dramatically enhance functionality with minimal code using the power and flexibility of SOAP and XML to easily integrate Web Services into client-side applications.

- **Delphi Enterprise** - includes additional tools, extensive options for Internet. Delphi 6 makes next-generation e-business development with Web Services a snap.

This Program will concentrate on the Enterprise edition..

1.3.1. Delphi 6 Archite

Delphi 6 Architect is designed for professional enterprise developers who need to adapt quickly to changing business rules and manage sophisticated applications that synchronize with multiple database schemas. Delphi 2006 Architect includes an advanced ECO III framework that allows developers to rapidly deploy scalable external facing Web applications with executable state diagrams, object-relational mapping, and transparent persistence.

Delphi 6 Architect includes all of the capabilities of the Enterprise edition, and includes the complete ECO III framework, including new support for ECO State Machines powered by State Chart visual diagrams, and simultaneous persistence to multiple and mixed database servers.

- State Chart Diagrams
- Executable ECO State Machines
- Multi- and Mixed- ECO database support

1.3.2.Installation Delphi 6

To install Delphi 6 Enterprise, run INSTALL.EXE (default location C:\Program Files\Borland Delphi) and follow the installation instructions.

We are prompted to select a product to install, you only have one choice "Delphi 6":

Figure 1.1 The Select Page For Start Installation

While the setup runs, you'll need to enter your serial number and the authorization key (the two you got from inside a Cd rom driver).

Figure 1.2 Serial Number And Authorization Screen

Later, the License Agreement screen will popup:

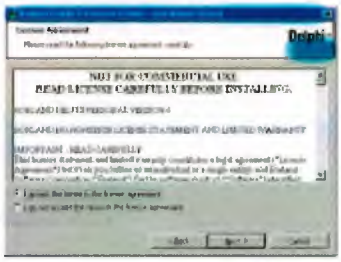


Figure 1.3 Lisanse Agreement Screen

After that, you have to pick the Setup Type, choose Typical. This way Delphi 6 Enterprise will be installed with the most common options. The next screen prompts you to choose the Destination folder.

Figure 1.4. SetUp Type and Destination Folder Screen

At the end of the installation process, the set-up program will create a sub menu in the Programs section of the Start menu, leading to the main Delphi 6 Enterprise program plus some additional tools.

1.5. Start Menu Screen

Figure 1.5. Start Menu

For a faster access to Delphi, create a shortcut on the Windows Desktop.

1.4. A Tour Of The Environment

This chapter explains how to start Delphi and gives you a quick tour of the main parts and tools of the Integrated Development Environment(IDE)

1.4.1. Running Delphi For The First Time

You can start Delphi in a similar way to most other Windows applications:

- Choose Programs | Borland Delphi 6 | Delphi 6 from the Windows Start menu
- Choose Run from the Windows Start menu and type Delphi32
- Double-click Delphi32.exe in the $(DELPHI)\Bin folder. Where $(DELPHI) if a folder where Delphi was installed. The default is C:\Program Files\Borland\Delphi6.
- Double-click the Delphi icon on the Desktop (if you've created a shortcut)

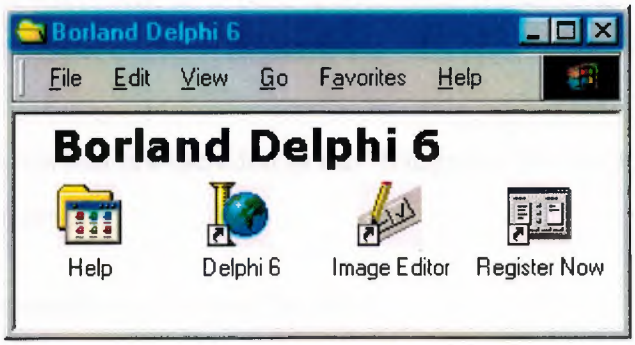


Figure 1.6. Borland Delphi 6 Folder

1.4.2. The Delphi IDE

As explained before, one of the ways to start Delphi is to choose Programs | Borland Delphi 6 | Delphi 6 from the Windows Start menu.

When Delphi starts (it could even take one full minute to start - depending on your hardware performance) you are presented with the IDE: the user interface where you can design, compile and debug your Delphi projects.

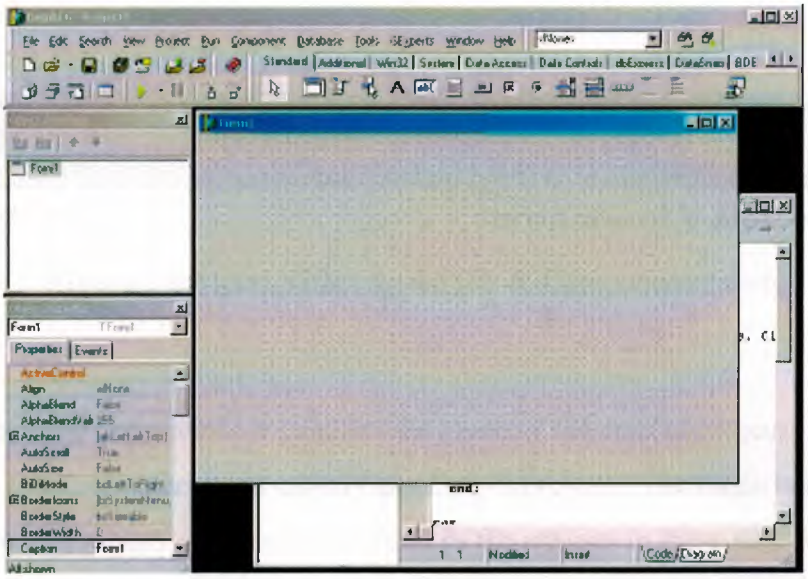


Figure 1.7.IDE

Like most other development tools (and unlike other Windows applications), Delphi IDE comprises a number of separate windows.

Some of the facilities that are included in the "Integrated Development Environment" (IDE) are listed below:

- A syntax sensitive program file editor
- A rapid optimising compiler
- Built in debugging /tracing facilities
- A visual interface developer
- Syntax sensitive help files
- Database creation and editing tools
- Image/Icon/Cursor creation / editing tools
- Version Control CASE tools

1.4.3. The Menus & Toolbar

The main window, positioned on the top of the screen, contains the main menu, toolbar and Component palette.

Figure 1.8. Menu ,Title , Speed Bar & Component Palette

The title bar of the main window contains the name of the current project (you'll see in some of the future chapters what exactly is a Delphi project). The menu bar includes a dozen drop-down menus - we'll explain many of the options in these menus later through this course. The toolbar provides a number of shortcuts to most frequently used operations and commands - such as running a project, or adding a new form to a project. To find out what particular button does, point your mouse "over" the button and wait for the tooltip. As you can see from the tooltip (for example, point to [Toggle Form/Unit]), many toolbuttons have keyboard shortcuts ([F12]).

The menus and toolbars are freely customizable. I suggest you to leave the default arrangement while working through the chapters of this course.

1.4.4. The Component Palette

You are probably familiar with the fact that any window in a standard Windows application contains a number of different (visible or not to the end user) objects, like: buttons, text boxes, radio buttons, check boxes etc. In Delphi programming terminology such objects are called controls (or components).Components are the building blocks of every Delphi application. To place a component on a window you drag it from the component palette. Each component has specific attributes that enable you to control your application at design and run time.

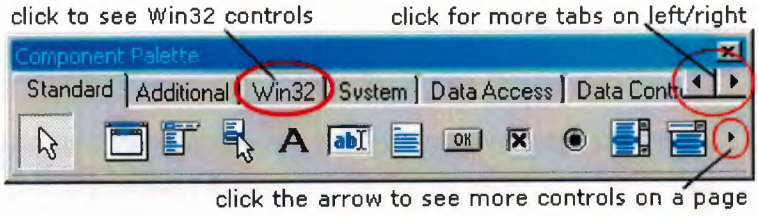


Figure 1.9. Component Palatte

Depending on the version of Delphi (assumed Delphi 6 Personal through this course), you start with more than 85 components at your disposal - you can even add more components later (those that you create or from a third party component vendor).

The components on the Component Palette are grouped according to the function they perform. Each page tab in the Component palette displays a group of icons representing the components you can use to design your application interface. For example, the Standard and Additional pages include controls such as an edit box, a button or a scroll box.

To see all components on a particular page (for example on the Win32 page) you simply click the tab name on the top of the palette. If a component palette lists more components that can be displayed on a page an arrow will appear on a far right side of the page allowing you to click it to scroll right. If a component palette has more tabs (pages) that can be displayed, more tabs can be displayed by clicking on the arrow buttons on the right-hand side.

1.4.5. The Code Editor

Each time you start Delphi, a new project is created that consists of one *empty* window. A typical Delphi application, in most cases, will contain more than one window - those windows are referred to as forms.

In our case this form has a name, it is called Form1. This form can be renamed, resized and moved, it has a caption and the three standard minimize, maximize and close buttons. As you can see a Delphi form is a regular Windows window

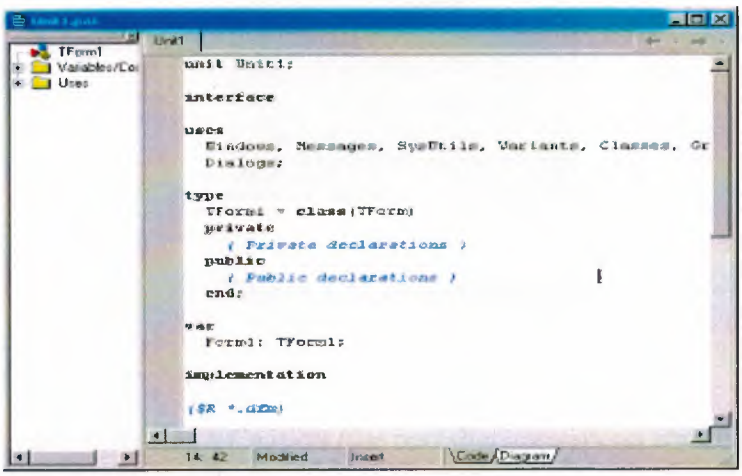


Fig.1.10. Code Editor Window

If the Form1 is the active window and you press [F12], the Code Editor window will be placed on top. As you design user interface of your application, Delphi automatically generates the underlying Object Pascal code. More lines will be added to this window as you add your own code that drives your application. This window displays code for the current form (Form1); the text is stored in a (so-called) unit - Unit1. You can open multiple files in the Code Editor. Each file opens on a new page of the Code editor, and each page is represented by a tab at the top of the window.

1.4.6. The Object Inspector

Each component and each form, has a set of properties – such as color, size, position, caption – that can be modified in the Delphi IDE or in your code, and a collection of events – such as a mouse click, keypress, or component activation – for which you can specify some additional behavior. The Object Inspector displays the properties and events (note the two tabs) for the selected component and allows you to change the property value or select the response to some event.

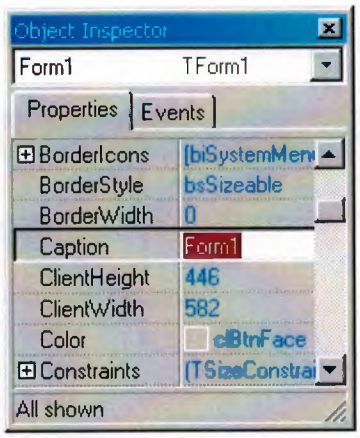


Figure 1.11. Object Inspector

For example, each form has a Caption (the text that appears on it's title bar). To change the caption of Form1 first activate the form by clicking on it. In the Object Inspector find the property Caption (in the left column), note that it has the 'Form1' value (in the right column). To change the caption of the form simply type the new text value, like 'My Form' (without the single quotes). When you press [Enter] the caption of the form will change to My Form.

Note that some properties can be changed more simply, the position of the form on the screen can be set by entering the value for the Left and Top properties - or the form can be simply dragged to the desired location.

1.4.7. The Object TreeView

Above the Object Inspector you should see the Object TreeView window. For the moment it's display is pretty simple. As you add components to the form, you'll see that it displays a component's parent-child relationships in a tree diagram. One of the great features of the Object TreeView is the ability to drag and drop components in order to change a component container without losing connections with other components.

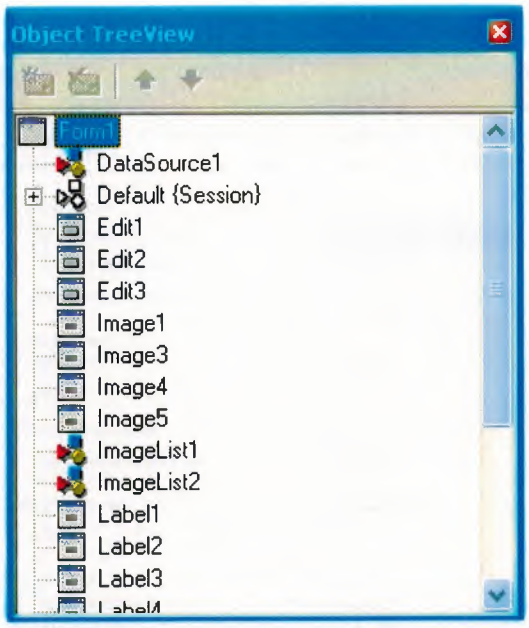


Figure 1.12. Object Tree View

The Object TreeView, Object Inspector and the Form Designer (the Form1 window) work cooperatively. If you have an object on a form (we have not placed any yet) and click it, its properties and events are displayed in the Object Inspector and the component becomes focussed in the Object TreeView.

1.4.8.Class Completion

Class Completion generates skeleton code for classes. Place the cursor anywhere within a class declaration; then press `Ctrl+Shift+C`, or right-click and select Complete Class at Cursor. Delphi automatically adds private **read** and **write** specifiers to the declarations for any properties that require them, then creates skeleton code for all the class's methods. You can also use Class Completion to fill in class declarations for methods you've already implemented.

To configure Class Completion, choose Tools|Environment Options and click the Explorer tab.

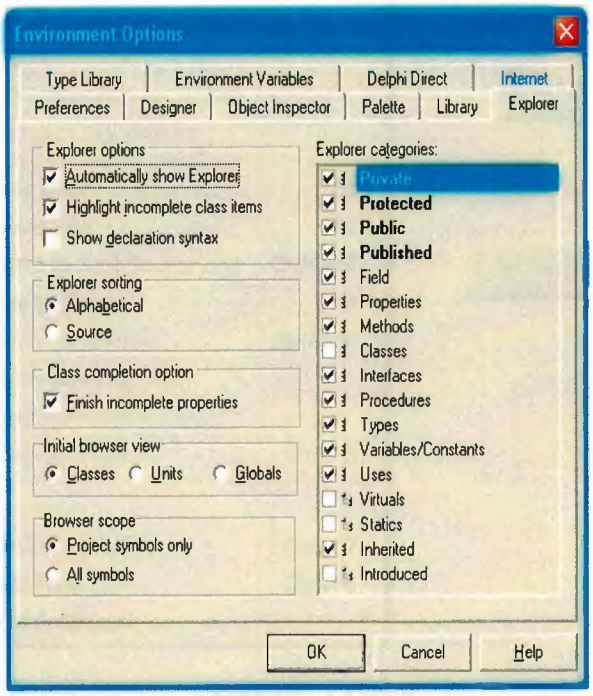


Fig.1.13.Class

1.4.9.Debugging applications

The IDE includes an integrated debugger that helps you locate and fix errors in your code. The debugger lets you control program execution, watch variables, and modify data values while your application is running. You can step through your code line by line, examining the state of the program at each breakpoint.

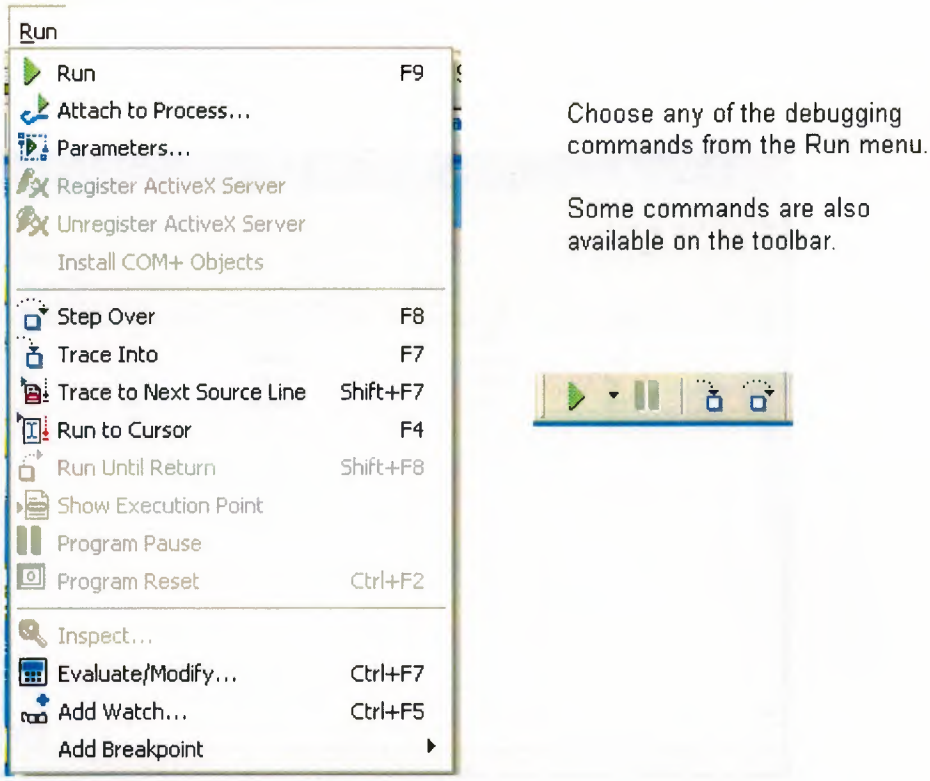


Figure1.**14.**Run

To use the debugger, you must compile your program with debug information. Choose Project|Options, select the Compiler page, and check Debug Information. Then you can begin a debugging session by running the program from the IDE. To set debugger options, choose Tools|Debugger Options.

Many debugging windows are available, including Breakpoints, Call Stack, Watches, Local Variables, Threads, Modules, CPU, and Event Log. Display them by choosing View|Debug Windows. To learn how to combine debugging windows for more convenient use, see "Docking tool windows".

1.4.10.Exploring databases

The SQL Explorer (or Database Explorer in some editions of Delphi) lets you work directly with a remote database server during application development. For example, you can create, delete, or restructure tables, and you can import constraints while you are developing a database application.

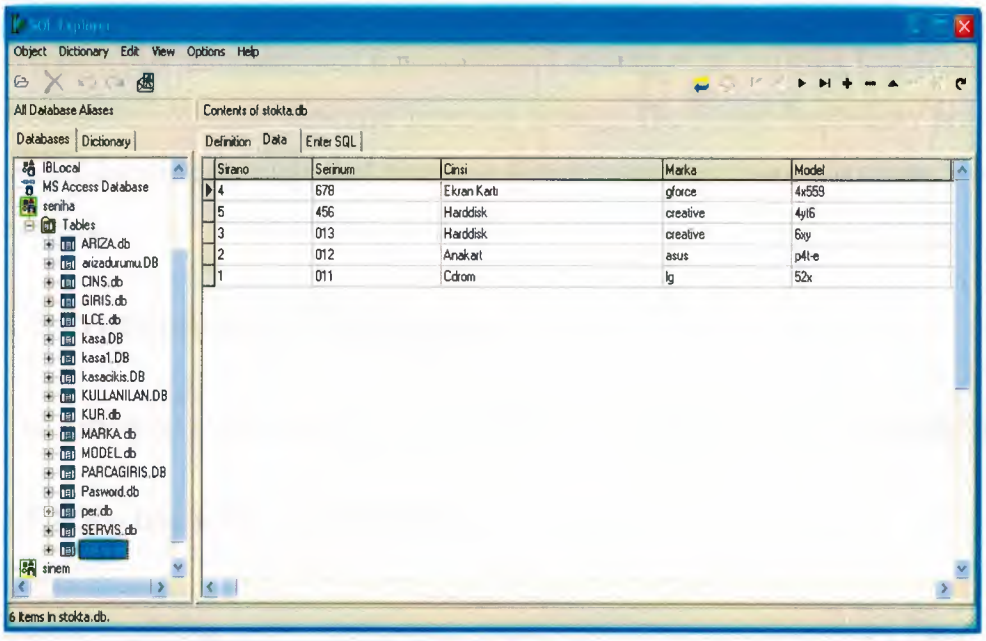


Figure 1.15.SQL Explorer

1.4.11.Templates and the Object Repository

The Object Repository contains forms, dialog boxes, data modules, wizards, DLLs, sample applications, and other items that can simplify development. Choose File|New to display the New Items dialog when you begin a project. Check the Repository to see if it contains an object that resembles one you want to create.

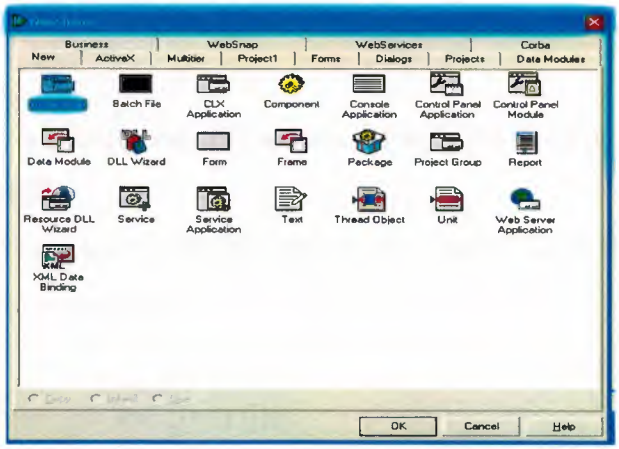


Figure 1.16. New Item

You can add your own objects to the Repository to facilitate reusing them and sharing them with other developers. Reusing objects lets you build families of applications with common user interfaces and functionality; building on an existing foundation also reduces development time and improves quality. The Object Repository provides a central location for tools that members of a development team can access over a network.

1.5.Programming With Delphi

The following section provide an overwiew of software development with Delphi.

1.5.1.Starting a New Application

Before beginning a new application, create a folder to hold the source files.

1. Create a folder called Seniha in the Projects directory off the main Delphi directory.
2. Open a new project.

Each application is represented by a project . When you start Delphi, it opens a blank project by default. If another project is already open, choose File|New Application to create a new project.

When you open a new project, Delphi automatically creates the following files.

- Project1.DPR : a source-code file associated with the project. This is called a project file.
- Unit1.PAS : a source-code file associated with the main project form. This is called a unit file.
- Unit1.DFM : a resource file that stores information about the main project form. This is called a form file.

Each form has its own unit and form files.

3. Choose File|Save All to save your files to disk. When the Save dialog appears, navigate to your Seniha folder and save each file using its default name.

Later on, you can save your work at any time by choosing File|Save All.

When you save your project, Delphi creates additional files in your project directory. You don't need to worry about them but don't delete them.

When you open a new project, Delphi displays the project's main form, named Form1 by default. You'll create the user interface and other parts of your application by placing components on this form.

Figure 1.17. Form Screen

The default form has maximize , minimize buttons and a close button , and a control menu

Next to the form, you'll see the Object Inspector, which you can use to set property values for the form and components you place on it.

The drop-down list at the top of the Object Inspector shows the current selected object.when an object is sellected the Object Inspector show its properties.

1.5.1.1. Setting Property Values

When you use the Object Inspector to set properties, Delphi maintains your source code for you. The values you set in the Object Inspector are called *design-time* settings.

For Example ; Set the background color of Form1 to Aqua.

Find the form's Color property in the Object Inspector and click the drop-down list displayed to the right of the property. Choose clAqua from the list.

1.5.2. Adding objects to the form

The Component palette represents components by icons grouped onto tabbed pages. Add a component to a form by selecting the component on the palette, then clicking on the form where you want to place it. You can also double-click a component to place it in the middle of the form.

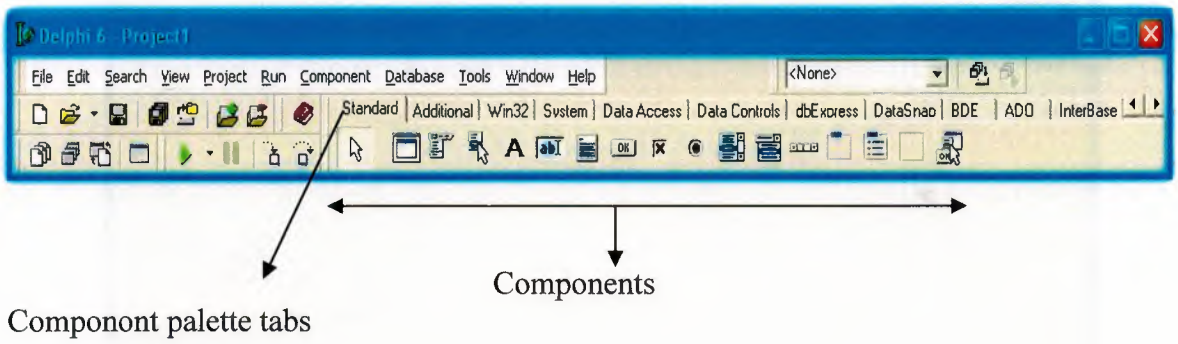


Figure 1.18.Standart Button

1.5.3.Add a Table and a StatusBar to the form:

Drop a Table component onto the form.

Click the BDE tab on the Component palette. To find the *Table* component, point at an icon on the palette for a moment; Delphi displays a Help hint showing the name of the component.

Fig.1.19.BDE Component palette

When you find the Table component, click it once to select it, then click on the form to place the component. The Table component is nonvisual, so it doesn't matter where you put it. Delphi names the object Table1 by default. (When you point to the component on the form, Delphi displays its name--Table1--and the type of object it is--TTable.)

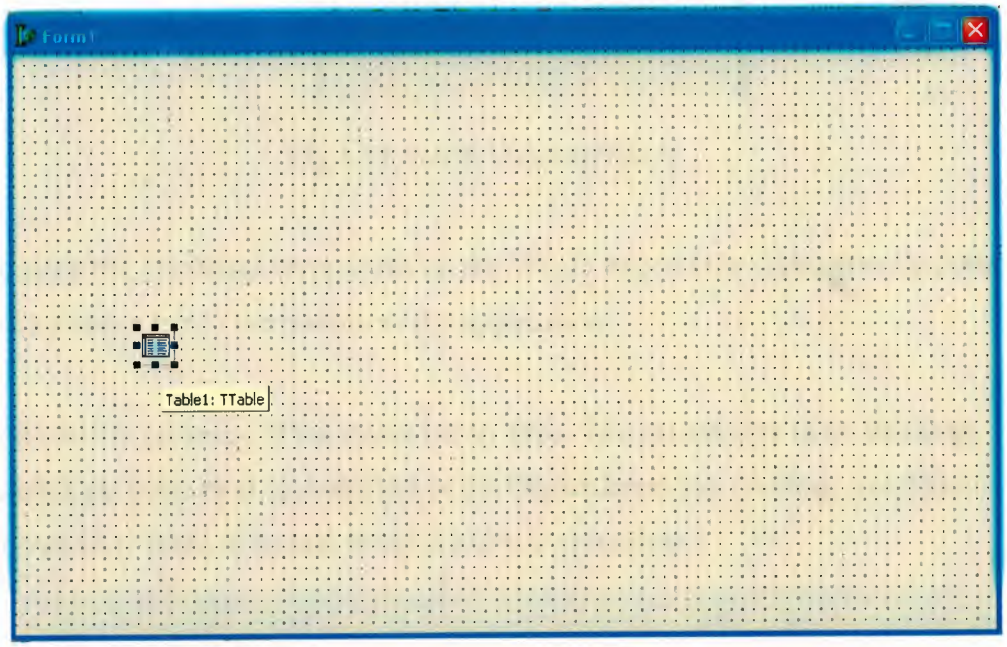


Figure 1.20. Table In The Form

Each Delphi component is a class; placing a component on a form creates an instance of that class. Once the component is on the form, Delphi generates the code necessary to construct an instance object when your application is running.

Set the DatabaseName property of Table1 to DBDEMOS. (DBDEMOS is an alias to the sample database that you're going to use.)

Select Table1 on the form, then choose the DatabaseName property in the Object Inspector. Select DBDEMOS from the drop-down list.

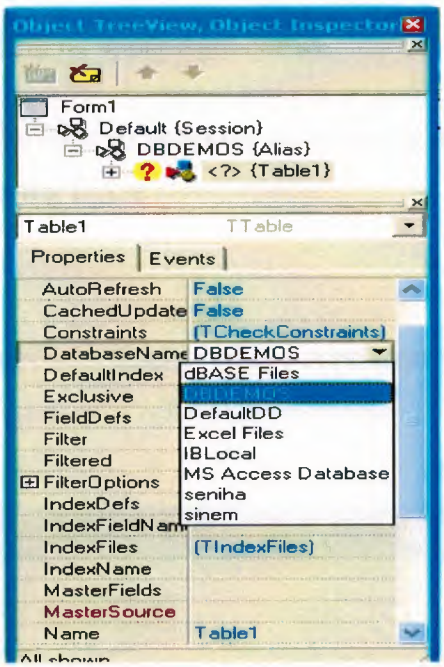


Fig.1.21.Select DatabaseName

Double-click the StatusBar component on the Win32 page of the Component palette. This adds a status bar to the bottom of the application.

Set the AutoHint property of the status bar to True. The easiest way to do this is to double-click on False next to AutoHint in the Object Inspector. (Setting AutoHint to True allows Help hints to appear in the status bar at runtime.)

1.5.4. Connecting to a Database

The next step is to add database controls and a DataSource to your form.

1. From the Data Access page of the Component palette, drop a DataSource component onto the form. The DataSource component is nonvisual, so it doesn't matter where you put it on the form. Set its DataSet property to Table1.

2. From the Data Controls page, choose the DBGrid component and drop it onto your form. Position it in the lower left corner of the form above the status bar, then expand it by dragging its upper right corner.

If necessary, you can enlarge the form by dragging its lower right corner. Your form should now resemble the following figure :

The Data Control page on Component palette holds components that let you view database tables.

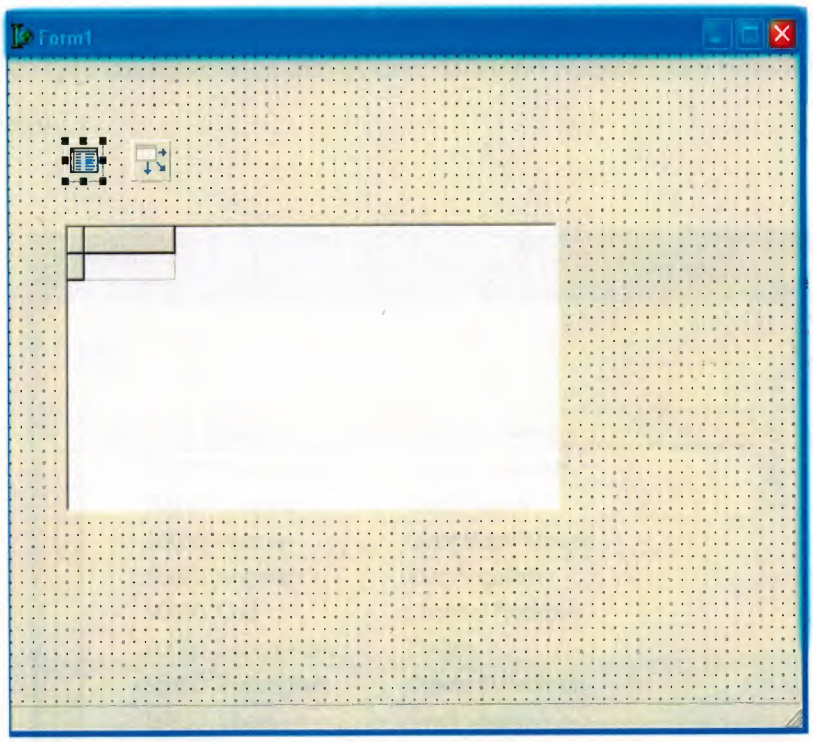


Figure 1.22.DBGrid In The Form

3. Set DBGrid properties to align the grid with the form. Double-click Anchors in the Object Inspector to display akLeft, akTop, akRight, and akBottom; set them all to True.

4. Set the DataSource property of DBGrid to DataSource1 (the default name of the DataSource component you just added to the form).

Now you can finish setting up the *Table1* object you placed on the form earlier.

5. Select the Table1 object on the form, then set its TableName property to BIOLIFE.DB. (Name is still Table1.) Next, set the Active property to True.

When you set Active to True, the grid fills with data from the BIOLIFE.DB database table. If the grid doesn't display data, make sure you've correctly set the properties of all the objects on the form, as explained in the instructions above. (Also verify that you copied the sample database files into your ...\Borland Shared\Data directory when you installed Delphi.)

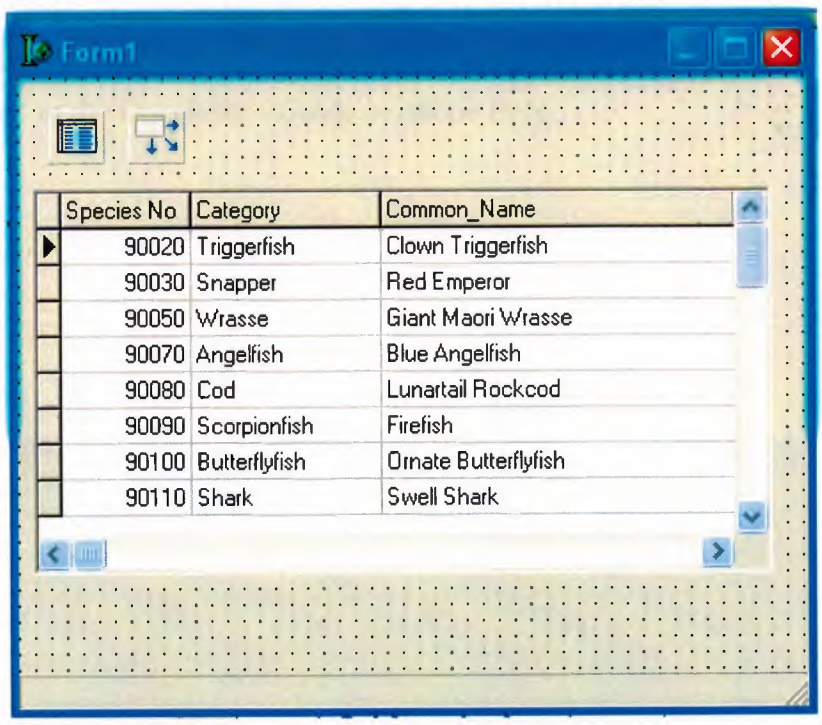
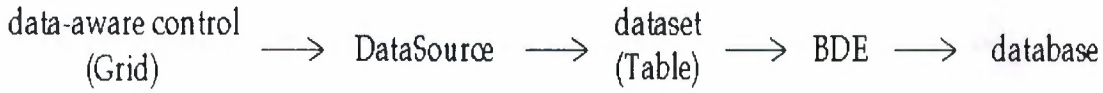


Figure 1.23. Show Table

The DBGrid control displays data at design time, while you are working in the IDE. This allows you to verify that you've connected to the database correctly. You cannot, however, edit the data at design time; to edit the data in the table, you'll have to run the application.

6. Press F9 to compile and run the project. (You can also run the project by clicking the Run button on the Debug toolbar, or by choosing Run from the Run menu.)
7. In connecting our application to a database, we've used three components and several levels of indirection. A data-aware control (in this case, a DBGrid) points to a DataSource object, which in turn points to a dataset object (in this case, a Table). Finally, the dataset (Table1) points to an actual database table (BIOLIFE), which is accessed through the BDE alias DBDEMOS. (BDE aliases are configured through the BDE Administrator.)



This architecture may seem complicated at first, but in the long run it simplifies development and maintenance. For more information, see "Developing database applications" in the Developer's Guide or online Help.

CHAPTER 2

Delphi enables you to create robust database applications quickly and easily. Delphi database applications can work directly with desktop databases like Paradox, dBASE,the Local InterBase Server, and ODBC data sources. The Delphi Client/Server edition also works with remote database servers such as Oracle, Sybase, Microsoft SQL Server, Informix, InterBase, and ODBC data sources. Delphi client applications can be scaled easily between mission critical network-based client/server databases, and local databases on a single machine.

This chapter introduces Delphi's database tools, including the Data Access and Data Controls component pages, the Fields Editor, the Database Desktop, and the Database Forms Expert.

2.1.Database Application in Delphi

A Delphi database application is built using Delphi database development tools, Delphi data-access components, and data-aware GUI components. A database application uses Delphi components to communicate with the Borland Database Engine (BDE), which in turn communicates with databases. The following figure illustrates the relationship of Delphi tools and Delphi database applications to the BDE and data sources:

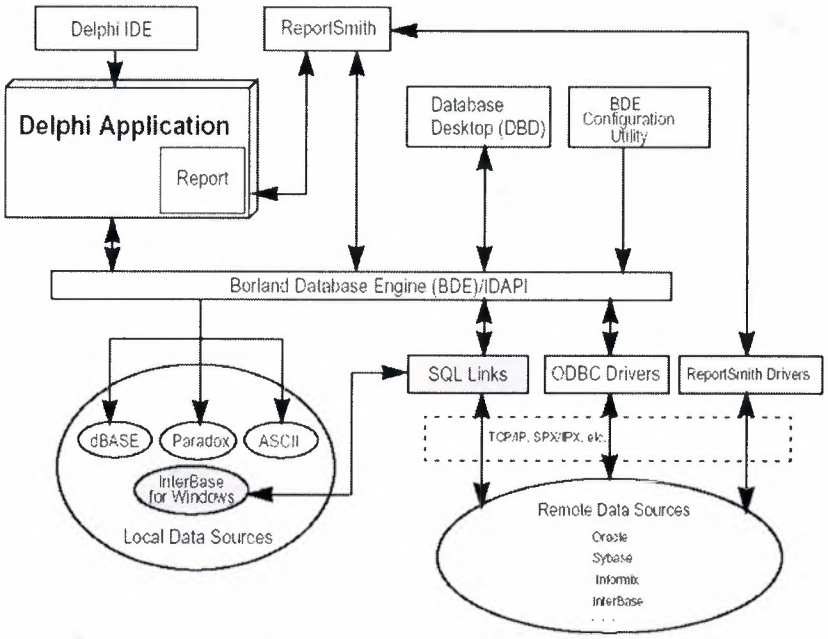


Figure 2.1. Delphi Database Architecture

The following table summarizes Delphi's database features:

TOOL	PURPOSE
Report Smith	Create,view,and print report
Borland Database Engine(BDE)	Access data form file based Paradox and dBASE tables,and from local InterBase Sever database
BDE Configuration Utility	Create and manage database connection aliases used by the BDE
Local InterBase Server	Provides a single-user,multi-instance desktop SQL server for building and testing Delphi applications,before scaling them up to a production database,such as oracle,sybase,informix,or Interbase on a remote server
InterBase SQL Link	Native driver that connect Delphi applications to the local interbase server.

Table.1.1.Database Features Summary

2.1.1. Database application development cycle

The goal of database application development is to build a product which meets end users' long-term needs. While this goal may seem obvious, it is important not to lose sight of it throughout the complexities and often conflicting demands of the development process. To create a successful application it is critical to define the end users' needs in detail early in the development process.

The three primary stages of database application development are

- Design and prototyping
- Implementation
- Deployment and maintenance

There are database and application tasks in each of these phases. Depending on the size and scope of the development project, the database and application tasks may be performed by different individuals or by the same individual. Often, one team or individual will be responsible for the database tasks of the project, and another team or individual will be responsible for the application tasks.

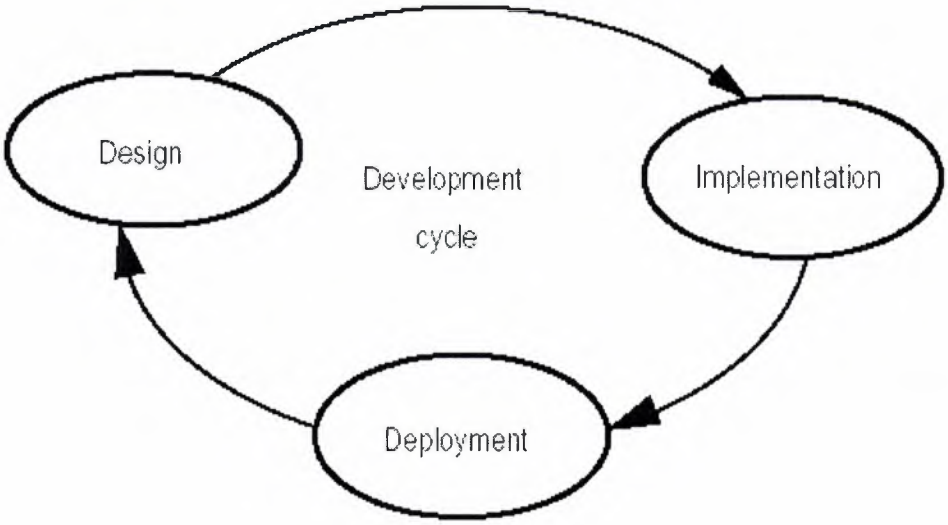


Figure 2.2 Deployment Cycle

2.1.1.1. Design phase

The design phase begins with requirements definition. In consultation with knowledgeable end users, define the functional specifications for the database and applications. Determine which aspects of the functional requirements will be implemented in the database design, and which aspects will be implemented in the applications.

For client/server applications, often certain functions can be performed either by the server or by the application; for example, a complex mathematical transform function could be performed either by the client application or by a stored procedure on the server. The hardware deployment configuration will generally determine whether such functions are best performed on the server or client. For example, if the client platforms are expected to be low-end desktop PCs, and the server platform is expected to be a high-end workstation, then it will probably be best to run computation-intensive functions on the server. If the hardware configuration changes, then it is possible to move the function between client and server in a later iteration.

2.1.1.2. Implementation phase

In the implementation phase, you use Delphi to build and test the application conceived in the design phase. During the implementation phase, you should use a duplicate data source, that is, a data source that has the same essential structure as the production database, but with a small subset of representative data. It is not recommended to develop an application against a production database, since the untested application may corrupt the data or otherwise interfere with normal database activities.

If your application will ultimately be deployed to use a desktop data source, make copies of the required tables with the Database Desktop, and populate them with representative "dummy" data.

If the application will ultimately be deployed to use a remote data source (an SQL server), then you can take two approaches during the implementation phase:

- Develop and test the application against a non-production database on the Local InterBase Server.
- Develop and test the application against a non-production database on the server.

The first approach has the advantage that is isolated on the development platform(s), and so will not interfere with other server activities. It will not consume server resources or increase network traffic. Its primary disadvantage is that only standard SQL server features can be used and tested during this phase, if you are using a server other than InterBase for the deployed application.

The second approach enables you to surface all server-specific features, but will consume network and server resources during testing. This approach can be dangerous, since it is conceivable that a programmer error could cause a server to crash during testing.

2.1.1.3. Deployment phase

In the deployment phase, the client/server application is put to the acid test: it is handed over to end users. To ensure that the application's basic functionality is error-free, deploy a prototype application before attempting to deploy a production application.

Since the ultimate judges of an application's efficacy are its users, developers must be prepared to incorporate changes to applications arising from their suggestions, changing business needs, and for general enhancement (for example, for usability). Sometimes application changes may require changes to the database, and conversely, changes to the database may require application changes. For this reason, application developers and database developers should work together closely during this phase. As features and enhancements are incorporated into the application, the application moves iteratively closer to completion.

2.1.2. Deploying an application

Deploying an application means giving it to the end users, and providing the necessary software they need to use the application in a production environment. Non-database applications require only an .EXE file to run—Delphi applications do not require a run time interpreter or DLL.

Typically, when deploying a database application, you will create a package that includes all the files that end users need to run the application and access data sources. These files include

- The application .EXE file and .DLL files (if any)
- Required ancillary files (for example, a README file or .HLP files for online help)
- BDE support for database access (desktop or server)
- ReportSmith Runtime for running and printing reports
- If the application uses VBX controls, include each VBX along with BIVBX11.DLL

If you are distributing the files on disks, you will generally want to compress them with a standard file compression utility, and provide the utility on the disk. You may also want to build a simple installation application to install the files for your users. For complex applications, you may want to use one of the many commercially-available installation programs.

2.1.2.1. Deploying BDE support

When you deploy a database application, you must ensure that the client platform has the correct version of the BDE installed. Delphi includes Redistributable BDE, with its own installation utility, that can you can redistribute with your applications. When you deploy an application, simply include a copy of the Redistributable BDE disk.

The Delphi license agreement requires you to make all the files in Redistributable BDE available to your application users. This requirement enables users to install the new version of the BDE for Delphi without interfering with existing Paradox and dBASE applications. You can advise your users to save disk space and install only the drivers required to run your application, but you must still distribute all the files in the Redistributable BDE.

FILE NAME	DESCRIPTION
IDAPI01.DLL	BDE API DLL
IDBAT01.DLL	BDE Batch utilies DLL
IDQRY01.DLL	BDE Query DLL
IDASCII01.DLL	BDE ASCII Driver DLL
IDPDX01.DLL	BDE Paradox Driver DLL
IDDBAS01.DLL	BDE dBASE Driver DLL
IDR10009.DLL	BDE Resources DLL
ILD01.DLL	Language Driver DLL
IDODBC01.DLL	BDE ODBC Socket DLL
ODBC.NEW	Microsoft ODBC Driver Manager DLL, version 2.0
ODBCINST.	NEW Microsoft ODBC Driver installation DLL, version 2.0
TUTILITY.DLL	BDE Tutility DLL
BDECFG.EXE	BDE Configuration Utility
BDECFG.HLP	BDE Configuration Utility Help
IDAPI.CFG	BDE (IDAPI) Configuration File

Table 2.1. Redistributable Borland Database Engine files

2.2. Using data access components and tools

This section describes how to use key Delphi features and tools when building database applications, including:

- The TSession component.
- Dataset components (TTable and TQuery), their properties, and their methods.
- TDataSource components, their properties, and their methods.
- TField objects, their properties, and their methods.
- The Fields Editor to instantiate and control TField objects.
- TReport and TBatchMove components.

2.2.1. Database components hierarchy

The Delphi database component hierarchy is important to show the properties, methods, and events inherited by components from their ancestors. The most important database components are

- *TSession*, a global component created automatically at run time. It is not visible on forms either at design time or run time.

- *TDatabase*, component that provides an additional level of control over server logins,transaction control, and other database features. It appears on the Data Access component page.

- *TDataSet* and its descendents, TTable and TQuery, collectively referred to as dataset components. TTable and TQuery components appear on the Data Access component page.

- *TDataSource*, a conduit between dataset components and data-aware components. It appears on the Data Access component page.

- *TFields*, components corresponding to database columns, created either dynamically by Delphi at run time or at design time with the Fields Editor. Data controls use them to access data from a database. In addition, you can define

calculated fields whose values are calculated based on the values of one or more database columns.

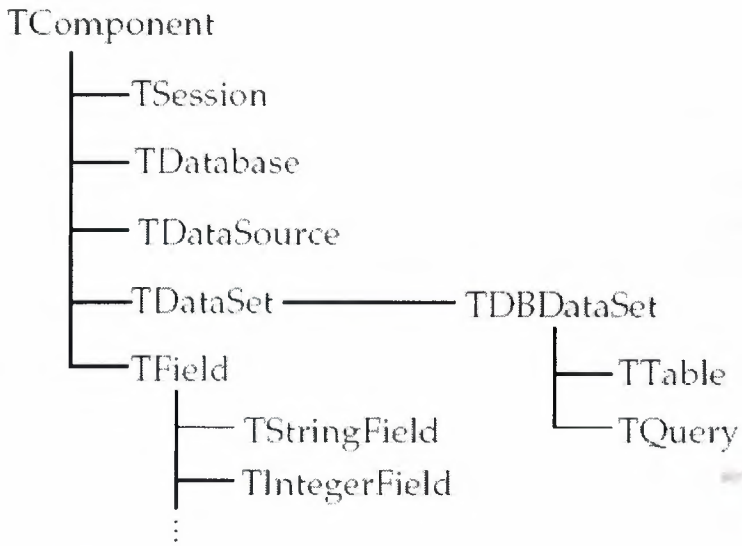


Fig 2.3. Delphi Data Access components hierarchy

2.3.Using Data Controls

To display and edit data from a database, use the components on the Data Controls page of the Component palette. Data controls include components such as TDBGrid for displaying and editing all specified records and fields in a table, and TDBNavigator for navigating among records, deleting records, and posting records when they change.

Figure 2.4. Data Controls Component palette

The following table summarizes the data controls in order from left to right as they appear on the Component palette:

DATA CONTROL	DESCRIPTION
TDBGrid	Displays information from a data source in a spreadsheet-like grid. Columns in the grid can be specified at design time using the Fields Editor or at run time (dynamically bound).
TDBNavigator	Provides buttons for navigating through data obtained from a data source. At design time, you can choose to include one or more buttons to navigate through records, update records, post records, and refresh the data from the data source
TDBText	Displays data from a specific column in the current data record.
TDBEdit	Uses an edit box to display and edit data from a specific column in the current data record.
TDBMemo	Displays memo-type database data. Memo fields can contain multiple lines of text or can contain BLOB (binary large object) data.
TDBImage	Displays graphic images and BLOB data from a specific column in the current data record.
TDBListBox	Displays a list of items from which a user can update a specific column in the current data record.
TDBComboBox	Combines a TDBEdit control with an attached list. The application user can update a specific column in the current data record by typing a value or by choosing a value from the drop-down list.

Table 2.2. Data controls

2.4. Using SQL in applications

SQL (Structured Query Language) is an industry-standard language for database operations. Delphi enables your application to use SQL syntax directly through the

TQuery component. Delphi applications can use SQL to access data from:

- Paradox or dBASE tables, using local SQL. The allowable syntax is a sub-set of ANSIstandard SQL and includes basic SELECT, INSERT, UPDATE, and DELETE statements. For more information on local SQL syntax, see Appendix C, "Using local SQL."

- Databases on the Local InterBase Server. Any statement in InterBase SQL is allowed.For information on syntax and limitations, see the InterBase Language Reference.

- Databases on remote database servers (Delphi Client/Server only). You must have installed the appropriate SQL Link. Any standard statement in the server's SQL is allowed. For information on SQL syntax and limitations, see your server documentation.

Delphi also supports heterogeneous queries against more than one server or table type (for example, data from an Oracle table and a Paradox table).

2.4.1. Using TQuery

TQuery is a dataset component, and shares many characteristics with TTable, "Using data access components and tools." In addition, Tquery enables Delphi applications to issue SQL statements to a database engine (either the BDE or a server SQL engine). The SQL statements can be either static or dynamic, that is, they can be set at design time or include parameters that vary at run time.

2.4.1.1. When to use Tquery

For simple database operations, TTable is often sufficient and provides portable database access through the BDE. However, TQuery provides additional capabilities that TTable does not. Use TQuery for:

- Multi-table queries (joins).
- Complex queries that require sub-SELECTs.
- Operations that require explicit SQL syntax.

TTable does not use SQL syntax; TQuery uses SQL, which provides powerful relational capabilities but may increase an application's overall complexity. Also, use of nonstandard (server-specific) SQL syntax may decrease an application's portability among servers.

2.4.1.2. How to use Tquery

To access a database, set the DatabaseName property to a defined BDE alias, a directory path for desktop database files, or a file name for a server database. If the application has a TDatabase component, DatabaseName can also be set to a local alias that it defines.

To issue SQL statements with a TQuery component:

- Assign the TQuery component's SQL property the text of the SQL statement. You can do this:

1. At design time, by editing the TQuery's SQL property in the Object Inspector,choosing the SQL property, and entering the SQL statements in the String List Editor dialog box. With Delphi Client/Server, you can also use the Visual Query Builder to construct SQL syntax.
2. At run time, by closing any current query with Close, clearing the SQL property with Clear, and then specifying the SQL text with the Add method.

- Execute the statement with the TQuery component's Open or ExecSQL method. Use Open for SELECT statements. Use ExecSQL for all other SQL statements. The differences between Open and ExecSQL are discussed in a subsequent section.
- To use a dynamic SQL statement, use the Prepare method, provide parameters and then call Open or ExecSQL. Prepare is not required, but will improve performance for dynamic queries executed multiple times.

The following diagram illustrates the lifetime of a TQuery component and the methods used to work with it:

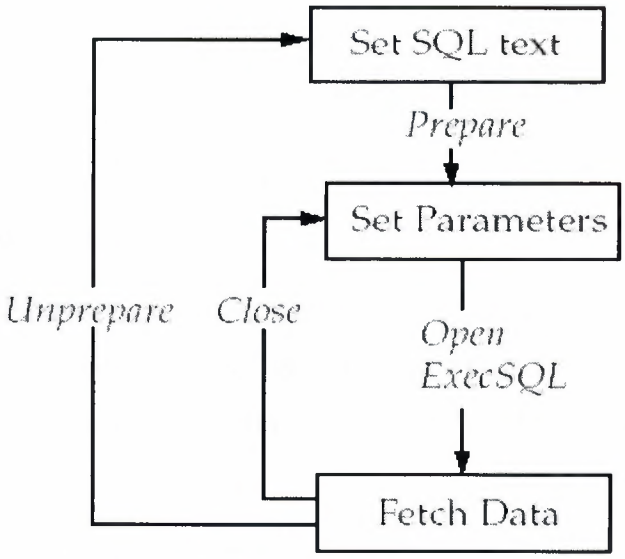


Figure 2.5. TQuery methods and flow

2.4.2. The SQL property

The SQL property contains the text of the SQL statement to be executed by a Query component. This property is of type TStrings, which means that it is a series of strings in a list. The list acts very much as if it were an array, but it is actually a special class with unique capabilities. For more information on TStrings, see the online VCL reference.

A Query component can execute two kinds of SQL statements:

- Static SQL statements
- Dynamic SQL statements

A static SQL statement is fixed at design time and does not contain any parameters or variables. For example, this statement is a static SQL statement:

SELECT * FROM CUSTOMER WHERE CUST_NO = 1234

A dynamic SQL statement, also called a parameterized statement, includes parameters for column or table names. For example, this is a dynamic SQL statement:

SELECT * FROM CUSTOMER WHERE CUST_NO = :Number

The variable Number, indicated by the leading colon, is a parameter which must be provided at run time and may vary each time the statement is executed.

2.5.Using Database Desktop

This appendix describes Database Desktop and provides a synopsis of Database Desktop features.

2.5.1.What is Database Desktop?

Database Desktop provides an easy way to create, restructure, and query tables to help you develop database applications with Delphi. You can use Database Desktop either as a standalone application on a single computer running Windows or as a multiuser application on a network.

2.5.2 Starting Database Desktop

To start Database Desktop, double-click the Database Desktop icon in the Delphi program group or choose File|Run in the Program Manager and run DBD.EXE.

CHAPTER 3

COMPUTER SALE AND TECHNICAL SERVICE PROGRAM

Figure 3.1.Sıgn In

This form obtain that o reach main form .if enter correct "user ıd" and correct "Password" then press LOGIN or enter button on the keyboard ,open the main form of program.if enter wrong "password "or "user ıd ",program shows the mistake message .

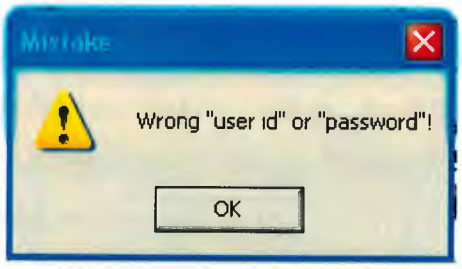


Figure 3.2. Mistake Message

If press cancel button .The program shutting down.

After enter correct "user ıd" and "password.we see the calculator for foreign money exchange.

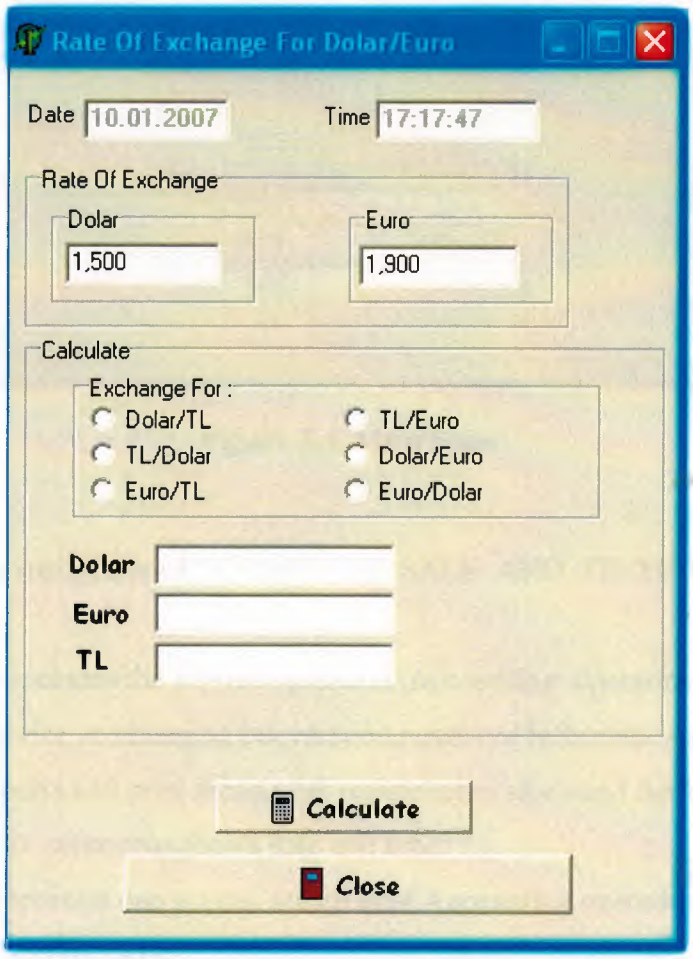


Figure 3.3 Exchange Money

This Form shows the date and time every entering .we enter rate of exchange for dolar and euro then sellect the which type want to change , enter the money press the calculate button program calculates the exchange.If press the close button or Esc on the keyboard exit the form and see the main form.

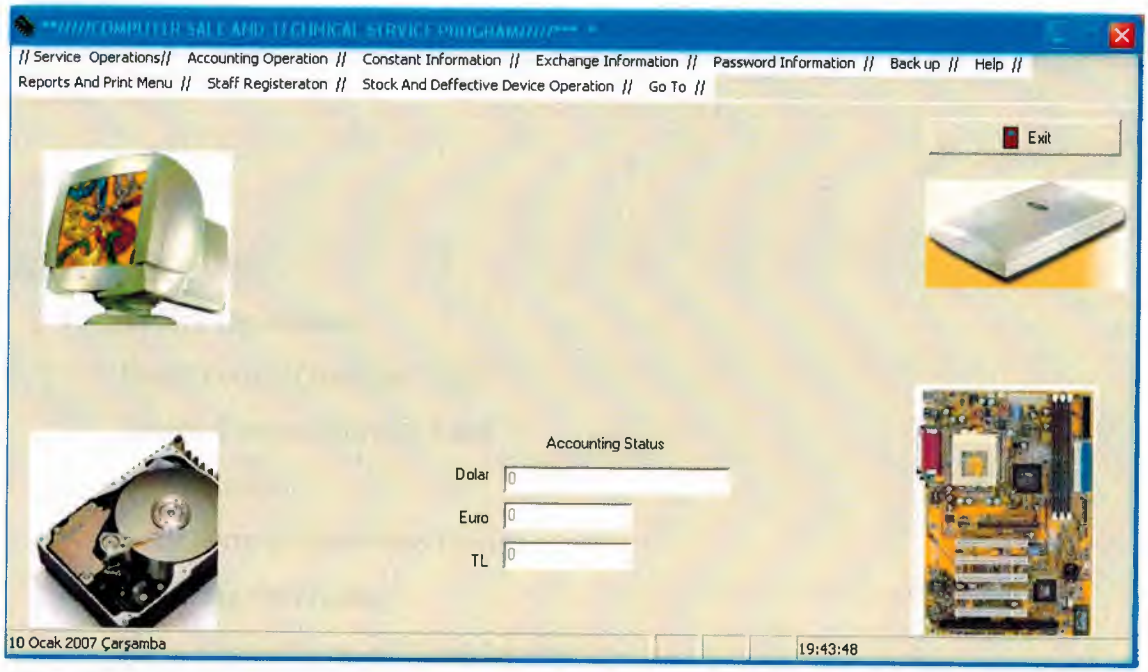


Figure 3.4. Main Form

This is the main form of " COMPUTER SALE AND TECHNICAL SERVICE PROGRAM "

This form includes the service operation ,Accounting operation constant information for device ,exchanging information,password information,back up for databases,help,reports and print menu,staff registeration stock and deffective device operation and go to submenus.shows date and time.

And the important one we can see all total Accounting operation for each type of money separetely in this form.

If we want to exit the program press the Exit button .program shows the question message box for if sure this operation.

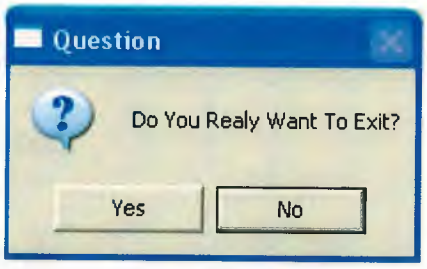


Figure 3.5.Question

when we sellect the Yes the program shutting down but othervise sellect the No program continue.

3.1.Inside a menus

1. **Service Operation :**

   . Entry Form of Service Card

   . Search Form of Service Card

   . Used Device

   .Entry Form of Hardware Device

2. **Acconting Operation**

   . Entry /Outlet Form of Cash Money

   .Outlet Form of Cash Money

   .Summery of Outlet Cash Money

3. **Constant Information**

   .Mark

   .Model

   .Hardware Device

   .Description of Deffect

   .Country /City

4. **Exchange Information**

   .Calculator For Foreign Money Exchange

5. **Password Information**

   .Change Password

6. **Back Up**

   .Save Back Up

7. **Help**

   .About

8. **Reports and Print Menu**

   .List of Stock

   .List of Defective Device

9. **Staff Registeration**

   .Register or Delete Staff

10. Stock and Deffective Device

.Entry Form of Stock Device

.Entry Form of Deffective Device

.Result Form of Deffective Device

.Failure Observation Form

11. Go To

3.2.Service Operation

Entry Form Of Service Card

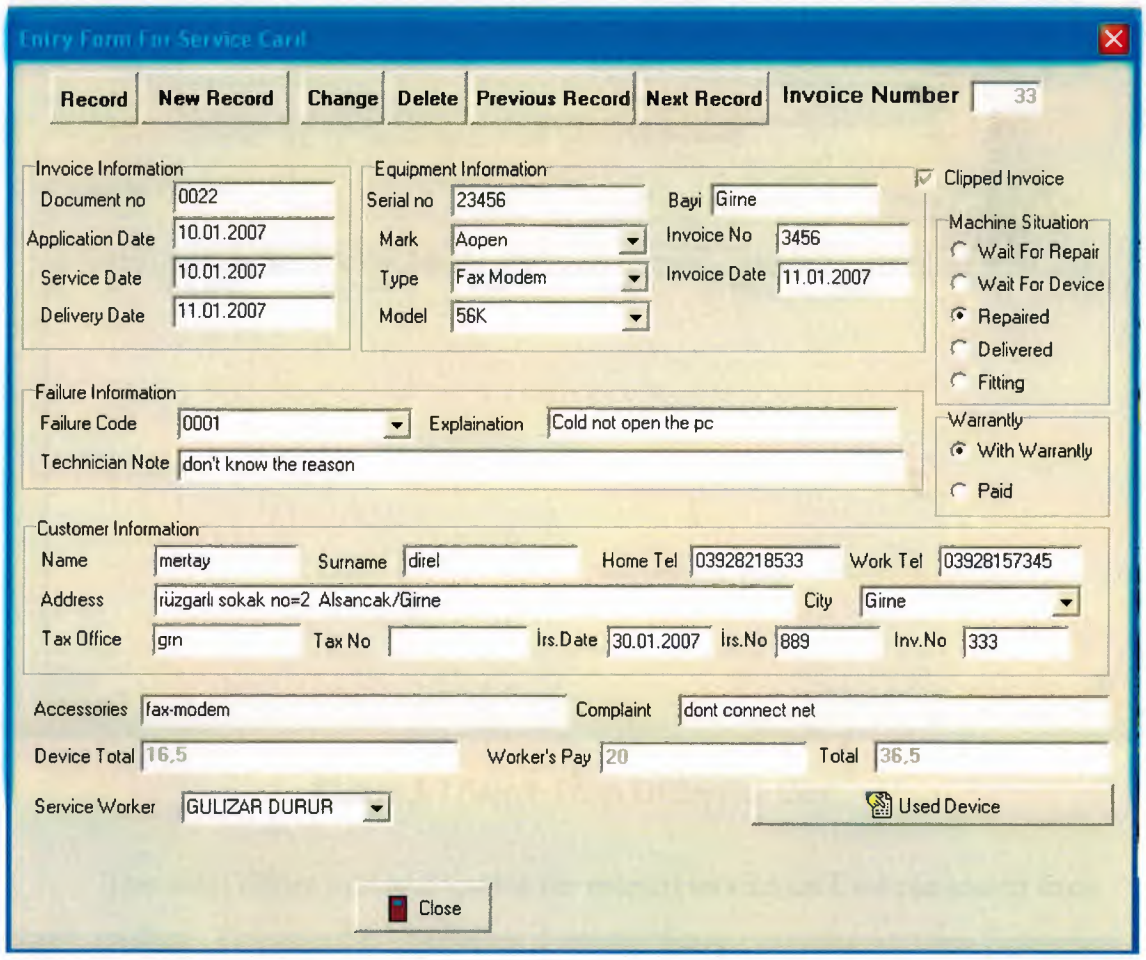


Figure 3.6. Entry Form of Service Card

This form records the invoice information,equipment information,failure information,customer information,machine situation,warrantly situation for the system

.If press the used device button we can calculate the total price after repaired.and record the used device for the every system by the invoice number.plus record the which service worker interested current system.ıf want to change anything we can do this by press the change button and record the changes.

If we enter space the serial number the system shows the message that "have to enter serial number".

Mark , Type and Model is selected by the system where the entered constant information.

The last when we want to exit this form we can do this with close button or Esc .

Search Form Of Service Card

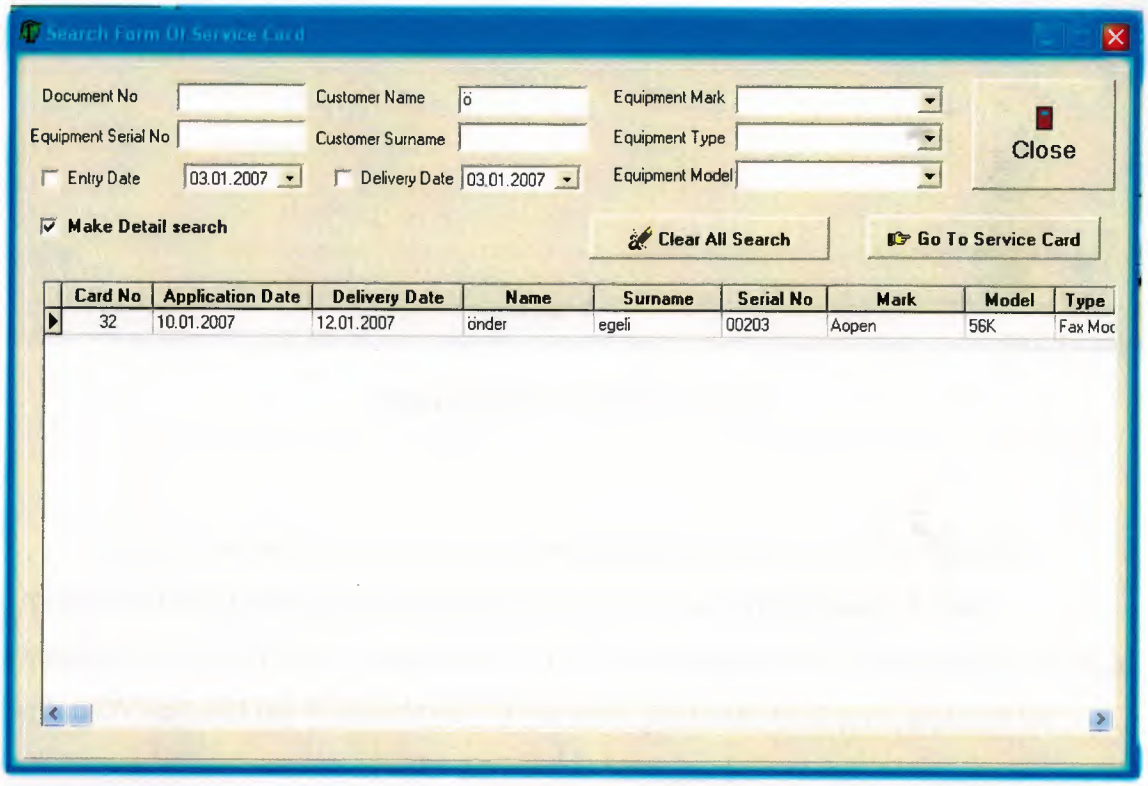


Figure 3.7.Search Form Of Service Card

This form obtain to search enable for entered service card.we can search from many attribute , Document no ,Serial no ,customer name,customer surname,Equipment Mark ,Equipment Type ,Equipment Model .we can use only one of these or all of them.

We can go to Service card (Entry Form of Service Card).press the Go To Service Card button and can see the all details about the system.

When finish the work press Close button and exit the form.

Used Device Form

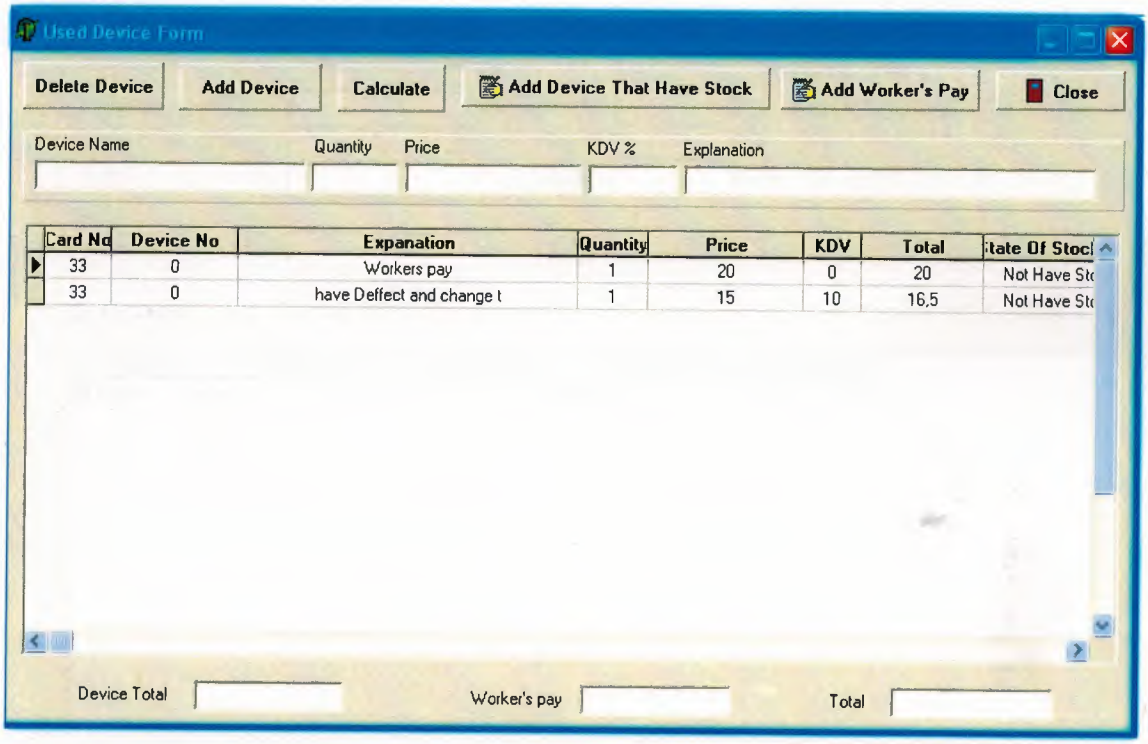


Figure 3.8. Used Device Form

This is the one of the important form.Reach this form from the Service Operations/Used Device submenu or used device button which inside Service Operations / Entry form of service card . This form calculate the total used device price with KDV and add the Worker's pay, ıf we want.shows the total price and add the service card.if we want we add the device that have stock .

Device Entry Form

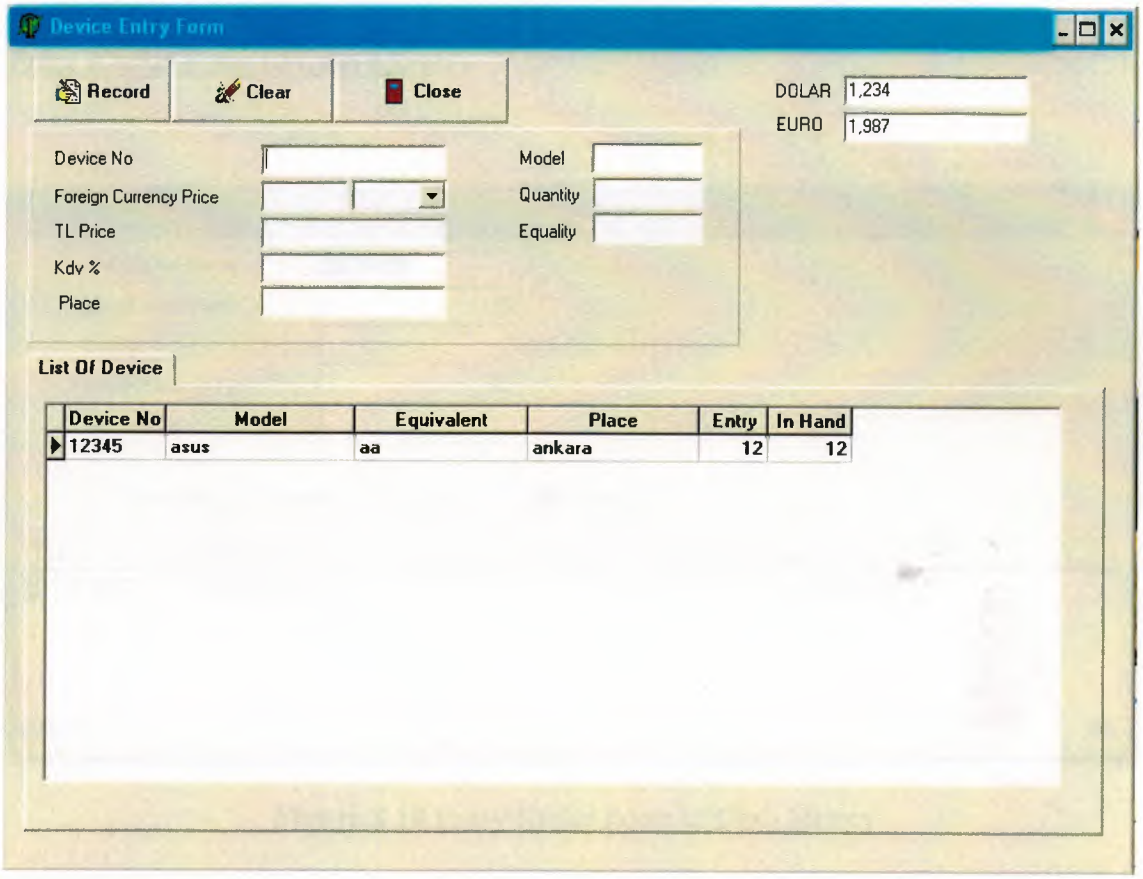


Figure 3.9.Device Entry Form

This form enter a new device information (Device no,foreign currency price,kdv,which place go , model,quantity,equality)in database

Tl price calculated from system by using the daily rate of exchange foreign money.

3.3.Accounting Operation

Entry /Outlet Form **of** Cash Money

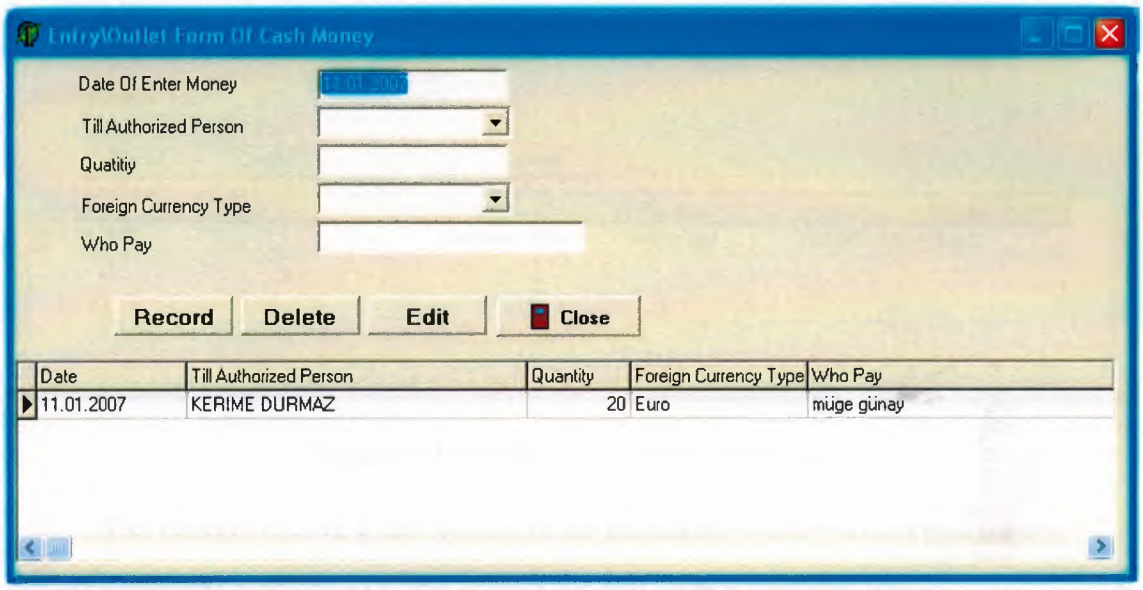


Figure 3.10.Entry\Outlet Form of Cash Money

This Form Records the entry money to till . There are three different type for dolar,euro and Tl .Date entering automatically by the system but ıf we want we can change.

We can change the recorded items with edit button .and we can delete the recorded money from the list but database cover the item.

The important process of this form is showing the entering money to the main form Accounting Status

Finished exit the close button or Esc .

The Outlet Form Of Cash Money

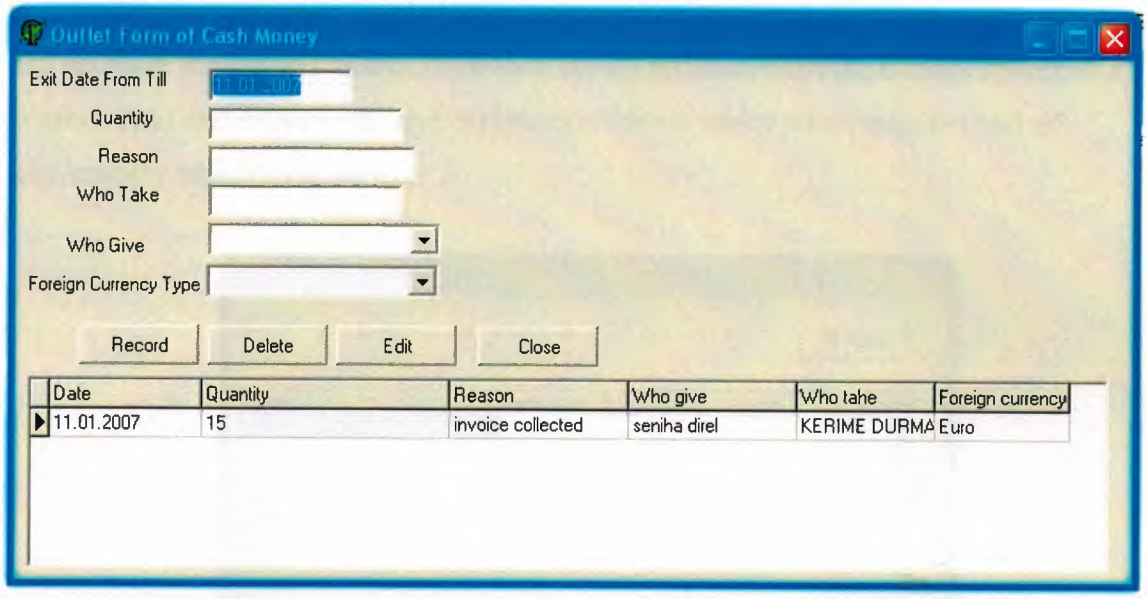


Figure 3.11.Outlet Form of Cash Money

The Outlet Form of Cash Money is the record the money which pay for the device supported company or any reason for exit money from the till.

Database recorded exit date, quantity for Dolar,Euro or Tl,name who take the money,name who give the money .

The resulted record this information the system subtract the outlet money from appropriate type(Dolar,Euro0r TL)the previous enter money and shows the main form Accountig Operation.

If prees the close button the form shut down.

3.4.Constant Information

The submenu which is below the constant Informatin main menu is the record the database the models ,marks,Hardware device ınfırmation of the current system in a market. And deffective code and explanation for use easily when required.and all city/country which have service

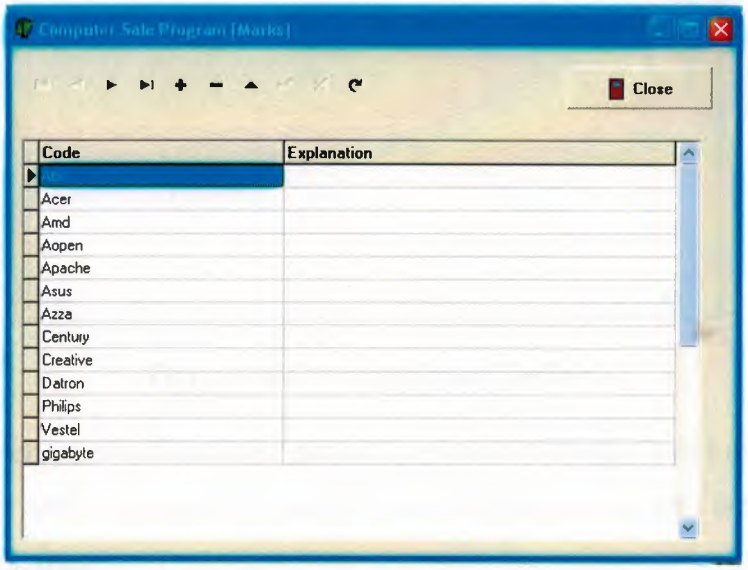


Figure 3.12.Marks

This form record the all marks which have .

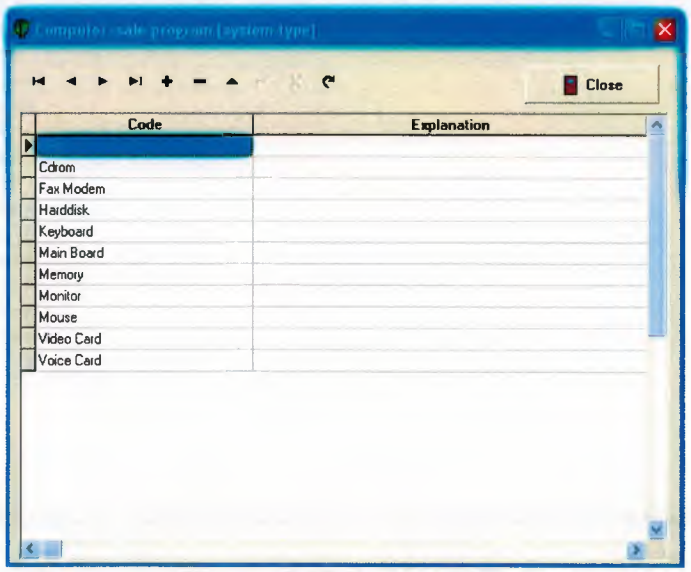


Figure 3.13.System Type

This form record the type the system. This three form depended on each other.

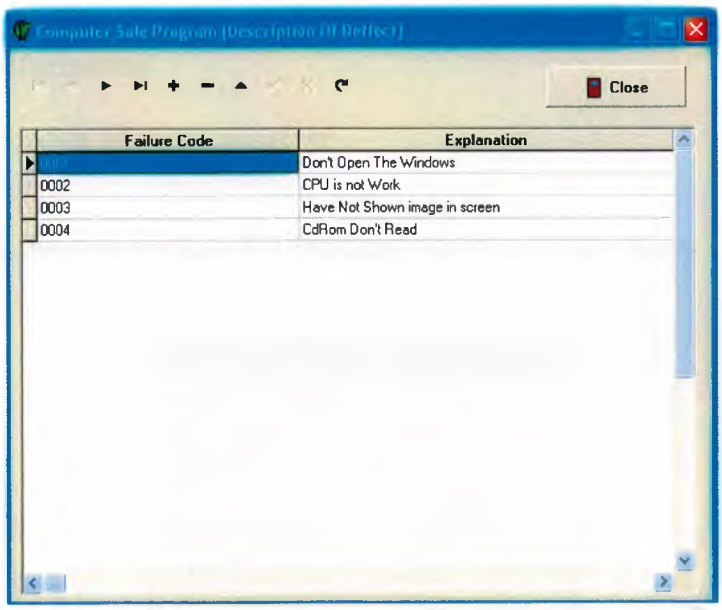


Figure 3.14.Description of Deffect

Describe ob deffect form generally used the Service Card Entry Form .if select the failure code system shows the description of deffective .this is make easily for eter Type of failure.

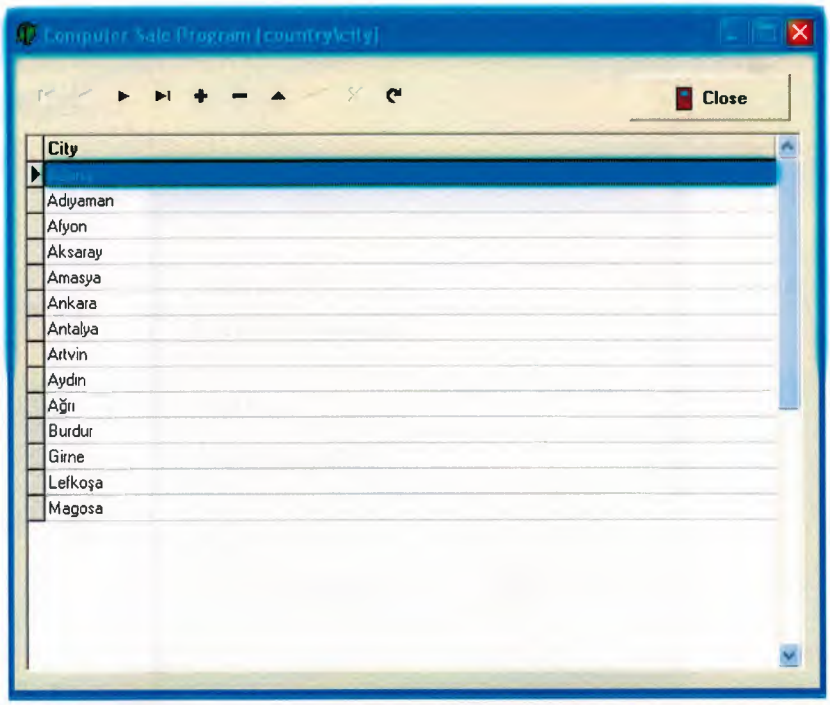


Figure 3.15.Country \City

And the last constant information form is Country\city form .The aim of form is same the others .this used for sellect easily city\country informations.

3.5.Exchange Information

Calculator For Foreign Money Exchange:

This submenu is show us the Rate Of Exchange For Dolar \Euro form .

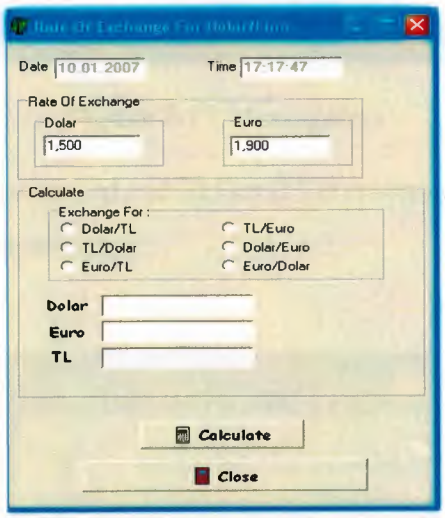


Figure 3.3. Exchange Money

3.6.Password Information

Change Password

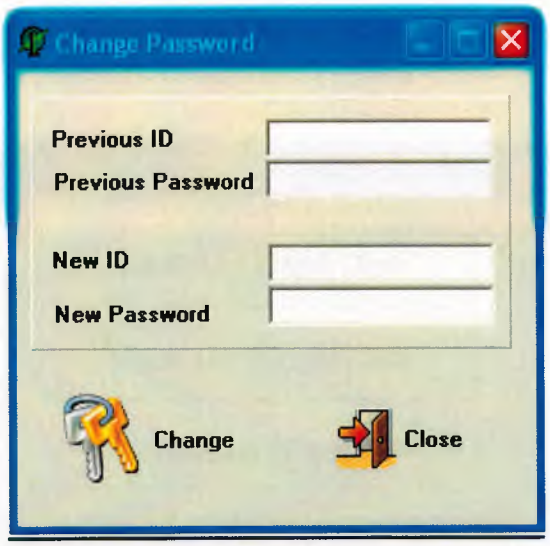


Figure 3.16. Change Password

If the user want to change password use this form .User must know the current user Id and Password.Enter the correct current Id ,Password and New Id ad Password then press the change button the system shows the message box

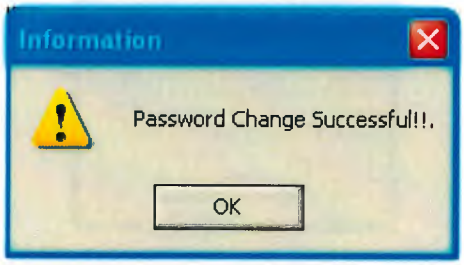


Figure 3.17.Messagebox

You must use new password and Id.But if Enter wrong current password or Id.system shows the mistake message

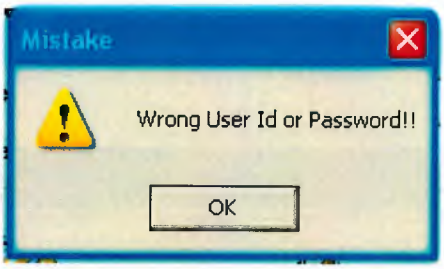


Figure 3.18.Messagebox

3.7.Back Up

Save Back Up

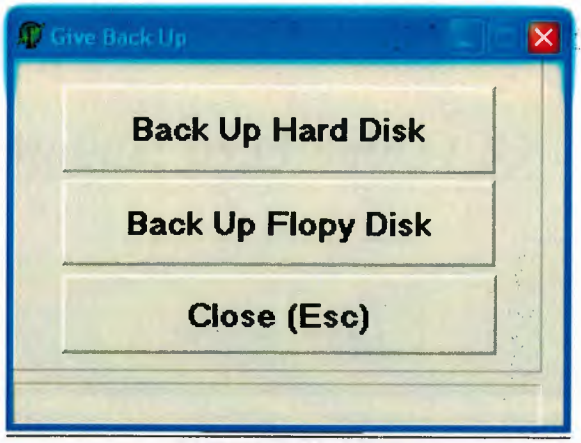


Figure3.19.Give Back Up

When press the Back up Hard Disk button .system creates a new folder inside the C:\ which name is Yedek andsaves all database inside this folder then the operation finished show the message is shown below:

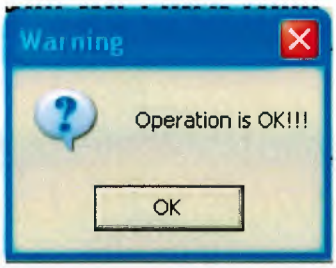


Figure 3.20. Messagebox

If Press the Back up Floppy disk System show the message that inside a disk A.then save the database inside this disk

3.8.Help

About:

Figure 3.21. About

This form include the information about who prepared.

3.9.Reports and Prints Menu

List of Stock

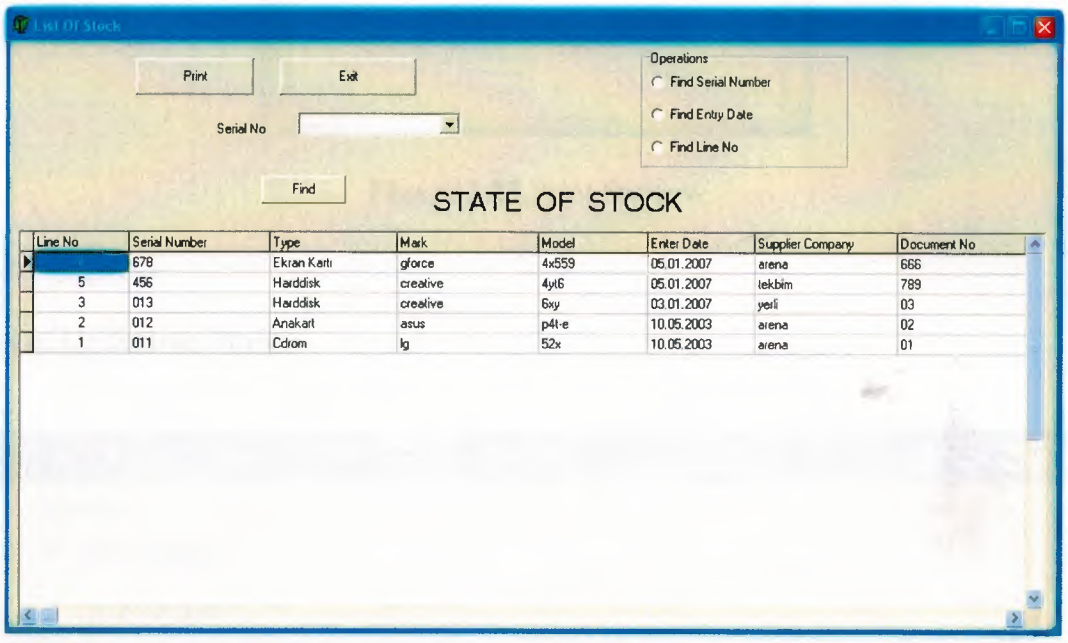


Figure 3.22.List of Stock

This form shows us state of stock. If we select the serial no, program found this serial no and show only founded item.we have three different way for see the stock.ıf we select first The stock increasing order by the serial number.ıf we select second stock increasing by the Entry date .and we selact the last one stock ıncreasing order by the line no .

The print button shows us the print preview form .ıf we want to print the stock stateuse this button . ıf we want to take print out we must sellect the print this page.

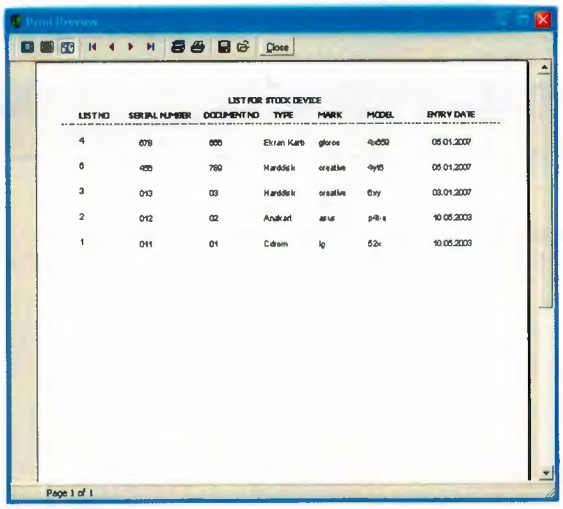


Figure 3.23. Print Preview

List of Deffective Device

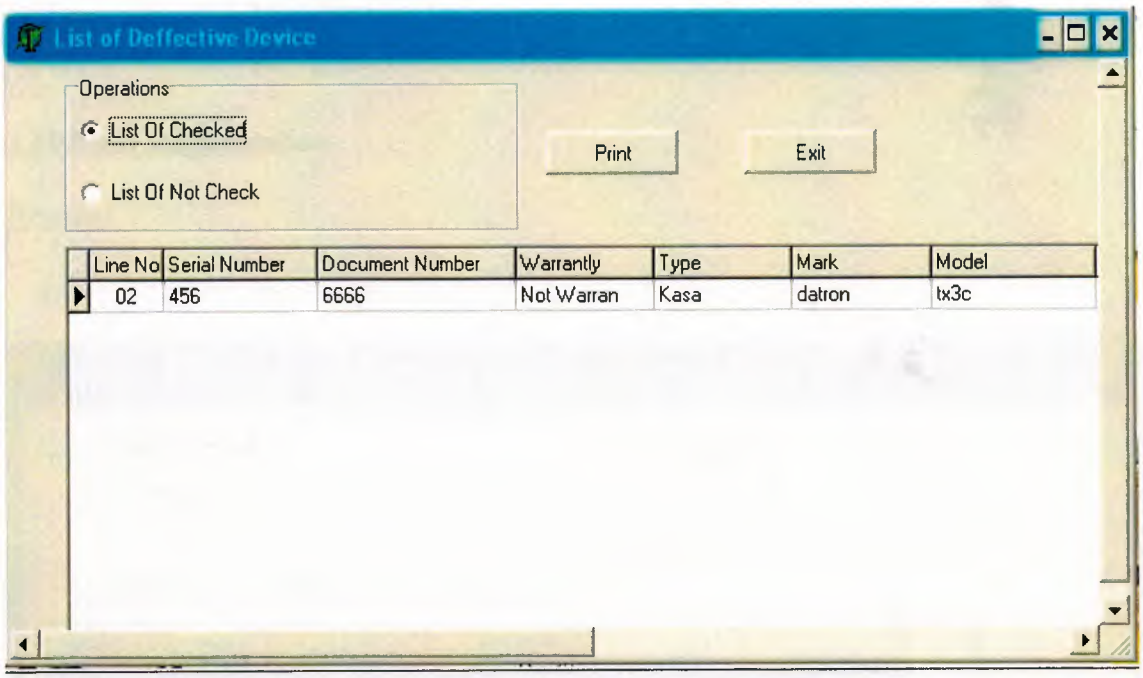


Figure 3.24. List of Deffective Device

This form lists the deffective device .If we select the first operation ,system shows the device which check that the deffect.ıf we select the second operation,system shows the not checked device.

The print button shows the print Preview page ;

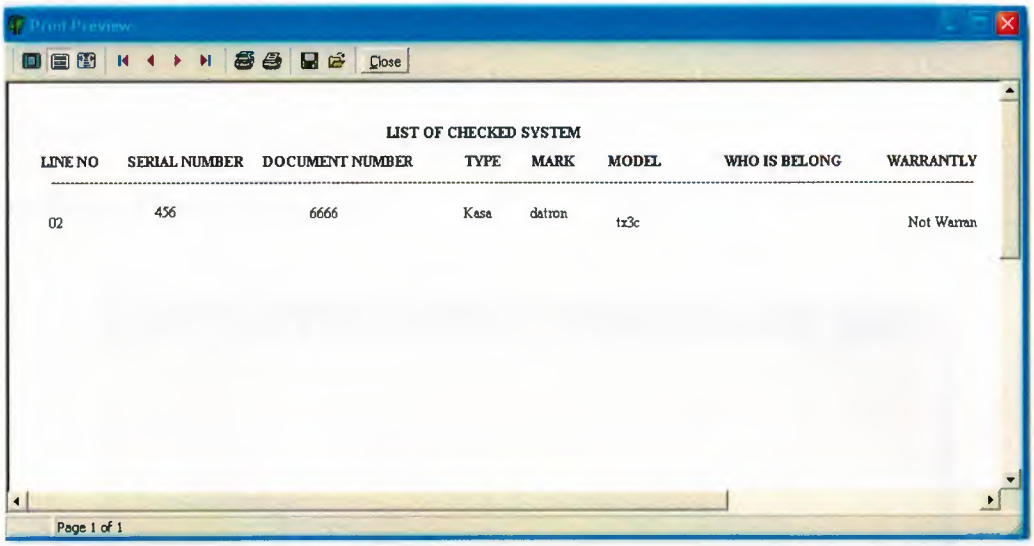


Figure 3.25. Print Preview

If we want to take print out we must sellect the print button . If we sellect close button close only Print Preview page.

3.10.Staff Registeration

Register.

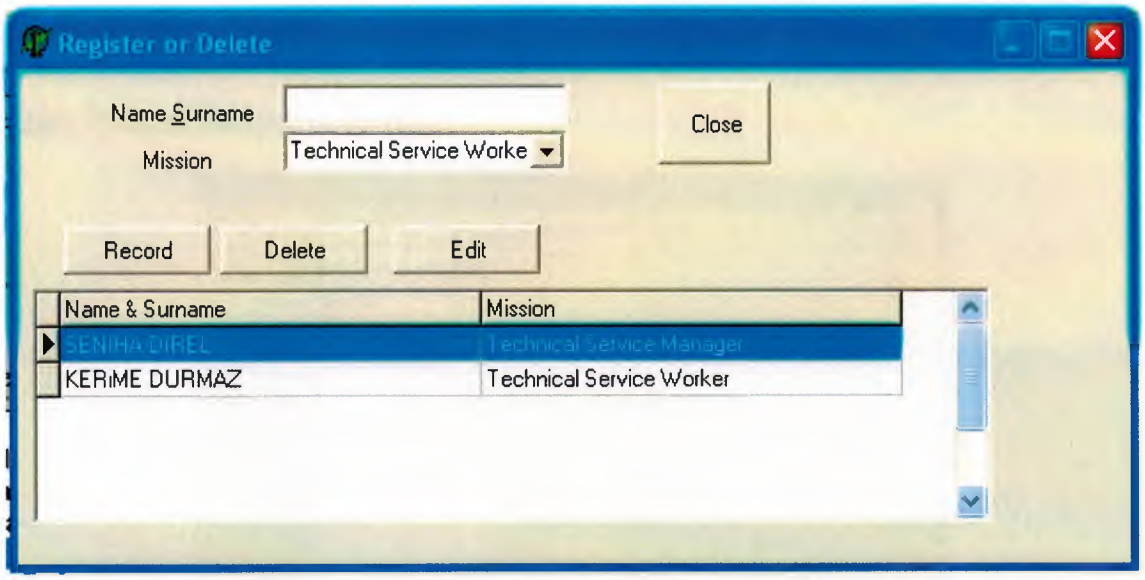


Figure 3.26. Register or Delete

Records database the workers and managers for a company with their aim.and deleted recorded worker.or edit the changes.

3.11.Stock And Deffective Device Operation

Entry Form of Stock Device

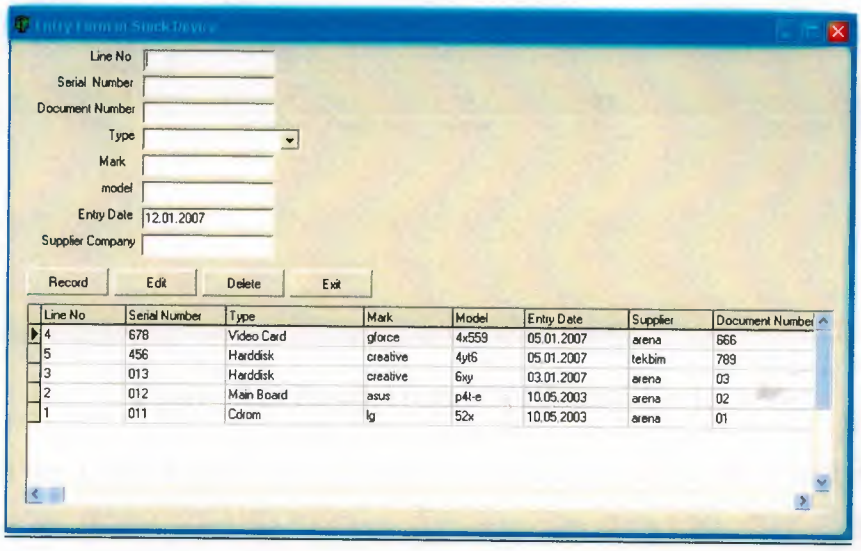


Figure 3.27.Entry Form of Stock Device

This form supply the possibility for enter the device in the stock . or delete the device from the stock .

Entry Form of Deffective Device

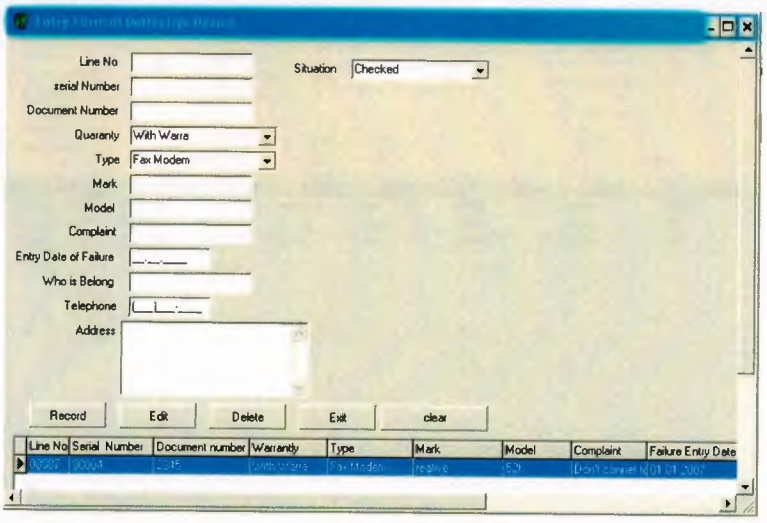


Figure 3.28. Deffective Device

This form enter the deffective device information(line no,serial no,document no,Quranty,type..etc) in the databaseand delete the deffective device from the database.

Result Form of Deffective Device

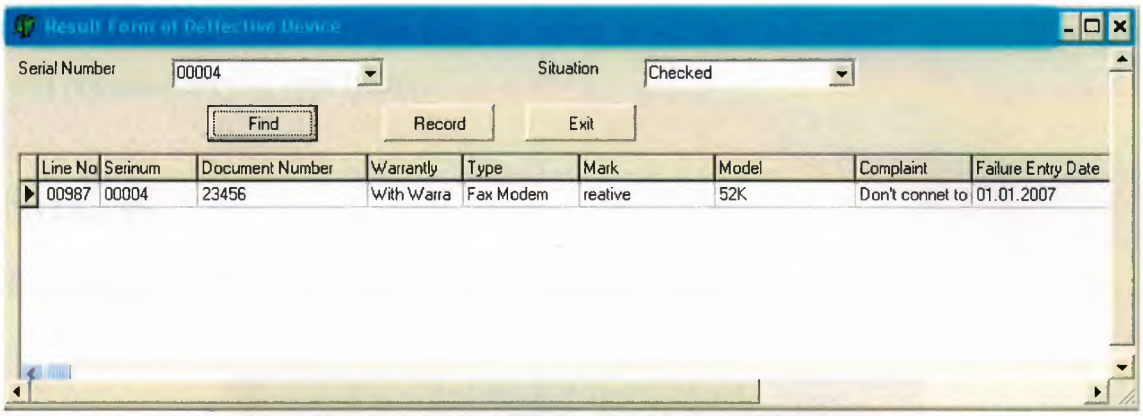


Figure 3.29.Result Form of Deffective Device

This form search and find the deffective device . we select the situation of device check or not check and press the find button then system shows the all checked or not checked device.

The record button use for change any item .If change anything inside the table and press record button the changes save in the database.

Failure Observation Form

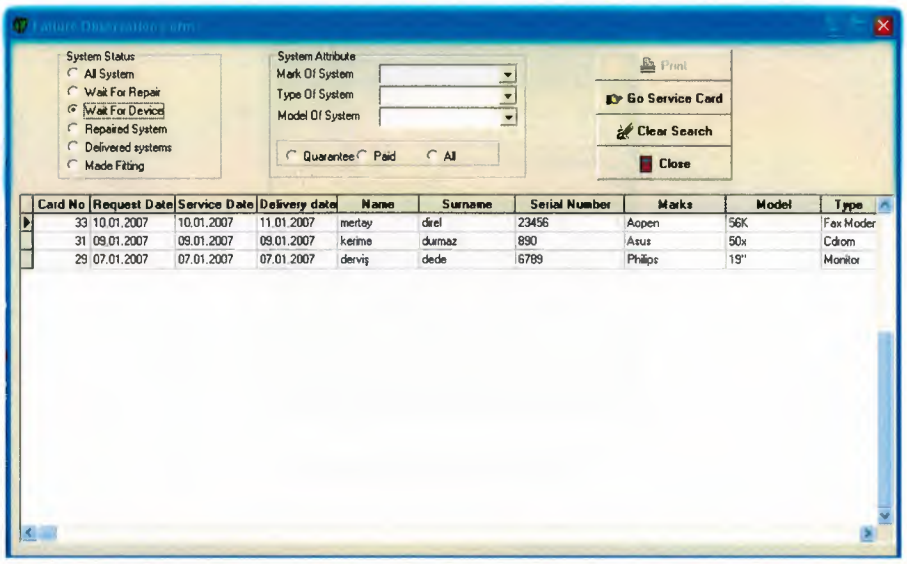


Figure 3.30. Failure Observation Form

The failure observation form help us for seperate which type of deffective device want to see.We can sellect the system type or sytem attribute together or we can use only one .For Example ; we want to see only wait system for device .we sellect "wait for device",And system show only wait for device .

If we want to go service card we select inside the table which service card want and press the service card button .

3.12.Go To

If we sellect Go To from main menu . The operations form open

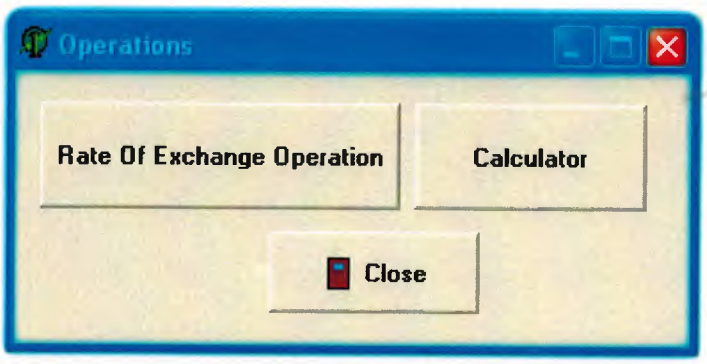


Figure 3.31. Operations

Rate of Exchange button open the "Rate Of Exchange for Dolar\Euro" form.Calculator button open the windows calculator .

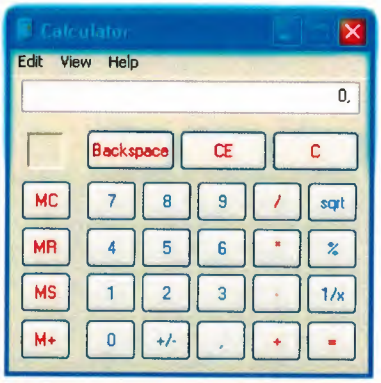


Figure 3.32. Calculator

When we want to close the All program we press the Close Exit button which is the program main form then system ask a question.

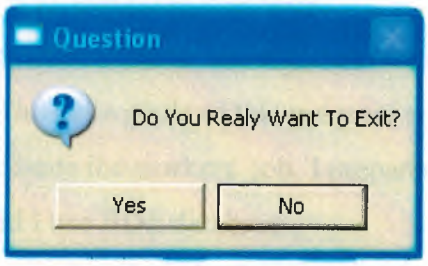


Figure 3.33.Messagebox

If we press the "Yes" button, program shutting down .or press the "No" button program continue.

CONCLUSION

In this finishing project COMPUTER SALE AND TECHNICAL SERVICE PROGRAMMING was used to computer company .

My project is record informations about the customer .or deffective device which concern the system. The main goals of this program is make easy to reach necessity information or facilitate the workers job. I prepared this project with Delphi 6 programming language and I use BDE database

I continue to get information about broken parts and systems thinking to reach them with this program which come to the computer company. Beside this I have try to control the stocks of new product which come to the company.

One thing that I must admit is that during the phase of doing this project I tried many solutions and almost all of them were correct but I tried harder to get the perfect one. Although they were all correct the reliability and consistency of them were not same. Some seemed more promising while other less promising. So the way presented here may (may not) be the way that fits you.

References

Referenca to Book:

[1] Steve Teixeira,Delphi programming: Delphi 6 Developer's Guide,2000

Referance to electronic book:

[1] Delphi 7 Delphi QuickStart .PDF
[2] Delphi_2005_Reviewers_Guide.PDF
[3] Delphi_Database_Application_Developers_Book.PDF

Referance to Electronic source:

[1] http://www.borland.com

APPENDIX A

Source Code of Program

Form 1 (Computer Sale And Technical Service Program)

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, ExtCtrls, Buttons, ToolWin, ComCtrls, StdCtrls, ImgList,
  ActnMan, ActnCtrls, jpeg, DB, DBTables;

type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    ServisFii1: TMenuItem;
    ParaBilgisi1: TMenuItem;
    Kasalemleri1: TMenuItem;
    Help1: TMenuItem;
    Kurlemleri1: TMenuItem;
    ServisFiiGiriFormu1: TMenuItem;
    N1: TMenuItem;
    ServisFiiAramaFormu1: TMenuItem;
    NakitParaGiriFormu1: TMenuItem;
    Dolar1: TMenuItem;
    About1: TMenuItem;
    ifreBilgileri1: TMenuItem;
    ifreDeitirme1: TMenuItem;
    Yedekleme1: TMenuItem;
    YedekVer1: TMenuItem;
    StatusBar1: TStatusBar;
    Timer1: TTimer;
    Timer2: TTimer;
    Image3: TImage;
    Image5: TImage;
    Image4: TImage;
    Image1: TImage;
    KullanlanParalar2: TMenuItem;
    ParaGiriFormu2: TMenuItem;
    N3: TMenuItem;
    NakitParaGirikFormu1: TMenuItem;
    Modeller1: TMenuItem;
    Cinsler1: TMenuItem;
```

```
ArzaTanmlama1: TMenuItem;
leehir1: TMenuItem;
N5: TMenuItem;
N6: TMenuItem;
Timer3: TTimer;
ImageList1: TImageList;
ImageList2: TImageList;
RaporlarveYazdrmaMenuleri1: TMenuItem;
StokdakiParalarnListesi1: TMenuItem;
ArzadakiBekleyenParalar1: TMenuItem;
NakitParakFormu1: TMenuItem;
PersonelKayd1: TMenuItem;
Kayt1: TMenuItem;
Label1: TLabel;
Label2: TLabel;
Label3: TLabel;
Edit1: TEdit;
Edit2: TEdit;
Edit3: TEdit;
Query1: TQuery;
DataSource1: TDataSource;
Label4: TLabel;
Parakzeti1: TMenuItem;
StokveArzalemleri1: TMenuItem;
Stokparagirii1: TMenuItem;
ArzaGiriformu1: TMenuItem;
ArzaSonucFormu1: TMenuItem;
SpeedButton5: TSpeedButton;
Timer4: TTimer;
GoTo1: TMenuItem;
procedure FormCreate(Sender: TObject);
procedure Timer1Timer(Sender: TObject);
procedure Timer2Timer(Sender: TObject);
procedure Dolar1Click(Sender: TObject);
procedure ServisFiiGiriFormu1Click(Sender: TObject);
procedure ServisFiiAramaFormu1Click(Sender: TObject);
procedure KullanlanParalar1Click(Sender: TObject);
procedure KullanlanParalar2Click(Sender: TObject);
procedure ParaGiriFormu2Click(Sender: TObject);
procedure NakitParaGirikFormu1Click(Sender: TObject);
procedure NakitParaGiriFormu1Click(Sender: TObject);
procedure Modeller1Click(Sender: TObject);
procedure Cinsler1Click(Sender: TObject);
procedure ArzaTanmlama1Click(Sender: TObject);
procedure leehir1Click(Sender: TObject);
procedure Timer3Timer(Sender: TObject);
procedure ifreDeitirme1Click(Sender: TObject);
procedure SpeedButton5Click(Sender: TObject);
procedure FormKeyPress(Sender: TObject; var Key: Char);
procedure N4Click(Sender: TObject);
```

```
    procedure FormShow(Sender: TObject);
    procedure YedekVer1Click(Sender: TObject);
    procedure About1Click(Sender: TObject);
    procedure Kayt1Click(Sender: TObject);

    procedure FormActivate(Sender: TObject);
    procedure NakitParakFormu1Click(Sender: TObject);
    procedure Parakzeti1Click(Sender: TObject);
    procedure Stokparagirii1Click(Sender: TObject);
    procedure ArzaGiriformu1Click(Sender: TObject);
    procedure ArzaSonucFormu1Click(Sender: TObject);
    procedure ArzadakiBekleyenParalar1Click(Sender: TObject);
    procedure StokdakiParalarnListesi1Click(Sender: TObject);
    procedure Timer4Timer(Sender: TObject);
    procedure GoTo1Click(Sender: TObject);



  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

uses  Unit2, Unit3, Unit8, Unit4, Unit5, Unit9, Unit10, Unit6, Unit11,
  Unit12, Unit13, Unit14, Unit15, Unit16, Unit17, Unit18, Unit20, Unit19,
  Unit23, Unit22, Unit24, Unit25, Unit26, Unit27, Unit29, Unit30, Unit31,
  Unit32, Unit28;

{$R *.dfm}

procedure TForm1.FormCreate(Sender: TObject);
begin
Timer1.interval:=1000;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
 statusbar1.Panels[4].Text:=TimeToStr(Time);
 end;
```

```
procedure TForm1.Timer2Timer(Sender: TObject);
var T:TKeyboardState;
   msg1,msg2,msg3:String;

begin
  GetKeyboardstate(t);
  if t[VK_NUMLOCK]<>0 then
    statusbar1.Panels[2].Text:='NUM'
  else statusbar1.Panels[2].Text:=' ';
  if t[VK_CAPITAL]<>0 then
    statusbar1.Panels[1].Text:='CAPS'
  else statusbar1.Panels[1].Text:=' ';
  if t[VK_SCROLL]<>0 then
    statusbar1.Panels[3].Text:='SCRL'
  else statusbar1.Panels[3].Text:=' ';

end;

procedure TForm1.Dolar1Click(Sender: TObject);
begin
form8.show;
end;

procedure TForm1.ServisFiiGiriFormu1Click(Sender: TObject);
begin
form4.show;
end;

procedure TForm1.ServisFiiAramaFormu1Click(Sender: TObject);
begin
form9.show;
end;

procedure TForm1.KullanlanParalar1Click(Sender: TObject);
begin
form10.show;
end;



procedure TForm1.KullanlanParalar2Click(Sender: TObject);
begin
form6.show;
end;



procedure TForm1.ParaGiriFormu2Click(Sender: TObject);
begin
form11.showmodal;
```

```
end;

procedure TForm1.NakitParaGirikFormu1Click(Sender: TObject);
begin
form12.show;

end;

procedure TForm1.NakitParaGiriFormu1Click(Sender: TObject);
begin
form13.show;
end;

procedure TForm1.Modeller1Click(Sender: TObject);
begin
form14.show;
end;

procedure TForm1.Cinsler1Click(Sender: TObject);
begin
form15.show;
end;

procedure TForm1.ArzaTanmlama1Click(Sender: TObject);
begin
form16.show;
end;

procedure TForm1.leehir1Click(Sender: TObject);
begin
form17.show;
end;

procedure TForm1.Timer3Timer(Sender: TObject);
begin
Statusbar1.Panels[0].Text:=FormatDateTime('dddddd  ',Now);
end;

procedure TForm1.ifreDeitirme1Click(Sender: TObject);
begin
form19.show;
end;


procedure TForm1.SpeedButton5Click(Sender: TObject);
var
m:byte;
begin
MessageBeep(MB_ICONQUESTION);
```

```
m:=application.MessageBox('Do You Realy Want To
Exit?','Question',4+32+256+4096);
if m=idyes then
close;
end;

procedure TForm1.FormKeyPress(Sender: TObject; var Key: Char);
begin
if key=#27 then
speedbutton5.Click;
end;

procedure TForm1.N4Click(Sender: TObject);
begin

//Form20.show;
end;




procedure TForm1.FormShow(Sender: TObject);
begin
form3.Show;
end;

procedure TForm1.YedekVer1Click(Sender: TObject);
begin
form23.showmodal;
end;

procedure TForm1.About1Click(Sender: TObject);
begin
form24.show;
form1.enabled:=false;

end;

procedure TForm1.Kayt1Click(Sender: TObject);
begin
form25.show;
form1.enabled:=false;

end;
```

```
function para(b:string):boolean;
begin
para:=false;
form1.Query1.First;
while not form1.Query1.eof do
if (b=form1.Query1.Fields[3].asstring)then
begin
para:=true;
exit;
end
else
form1.Query1.Next;
end;

procedure TForm1.FormActivate(Sender: TObject);
var yaz:integer;
begin
edit1.Enabled:=true;
edit2.Enabled:=true;
edit3.Enabled:=true;
query1.First;
while not(query1.Eof) do begin
edit1.Text:=query1.Fields[1].AsString;
query1.next;
edit2.Text:=query1.Fields[1].AsString;
query1.last;
edit3.Text:=query1.Fields[1].AsString;
end;

edit1.Enabled:=false;
edit2.Enabled:=false;
edit3.Enabled:=false;
end;

procedure TForm1.NakitParakFormu1Click(Sender: TObject);
begin
form26.show;
form1.enabled:=false;
end;

procedure TForm1.Parakzeti1Click(Sender: TObject);
begin
form27.showmodal;
end;

procedure TForm1.Stokparagirii1Click(Sender: TObject);
begin
```

```
form29.showmodal;
end;

procedure TForm1.ArzaGiriformu1Click(Sender: TObject);
begin
form30.showmodal;
end;

procedure TForm1.ArzaSonucFormu1Click(Sender: TObject);
begin
form31.showmodal;
end;

procedure TForm1.ArzadakiBekleyenParalar1Click(Sender: TObject);
begin
FORM32.SHOWMODAL;
end;

procedure TForm1.StokdakiParalarnListesi1Click(Sender: TObject);
begin
FORM22.SHOWMODAL;
end;

procedure TForm1.Timer4Timer(Sender: TObject);
begin
caption:=copy(caption,2,length(caption)-1)+caption[1];
end;
```

Form 2 (Image form)

```
unit Unit2;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, jpeg, ExtCtrls;

type
  TForm2 = class(TForm)
    Panel1: TPanel;
    Image1: TImage;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;
```

```
implementation

{$R *.dfm}

end.
```

Form 3(Sıgn In)

```
unit Unit3;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, DBTables, StdCtrls, Mask, Buttons, ExtCtrls, ComCtrls, jpeg;

type
  TForm3 = class(TForm)
    Panel2: TPanel;
    Panel1: TPanel;
    StatusBar1: TStatusBar;
    Edit1: TEdit;
    Edit2: TEdit;
    DataSource1: TDataSource;
    Query1: TQuery;
    Timer1: TTimer;
    Image1: TImage;
    SpeedButton2: TSpeedButton;
    SpeedButton1: TSpeedButton;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
    procedure FormShow(Sender: TObject);
    procedure FormKeyPress(Sender: TObject; var Key: Char);
    procedure Timer1Timer(Sender: TObject);


  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
var
  Form3: TForm3;
  a:boolean;
implementation

uses Unit1, Unit8;

{$R *.dfm}

procedure TForm3.SpeedButton1Click(Sender: TObject);
begin
form1.Close;
close;
end;

procedure TForm3.SpeedButton2Click(Sender: TObject);
var
t:string;
VolumeSerialNumber : DWORD;
MaximumComponentLength : DWORD;
FileSystemFlags : DWORD;
SerialNumber : string;

m:integer;
s:integer;
begin

    query1.SQL.Clear;
    query1.SQL.Text:='select * from pasword where KULLANICIADI
like'+#39+edit1.Text+'%'+#39;
    query1.Open;
    if query1.Fields[0].AsString=edit1.Text then
    begin
      query1.SQL.Text:='select * from pasword where SIFRE
like'+#39+edit2.Text+'%'+#39;
      query1.Open;
      if query1.Fields[1].asstring=edit2.Text then
      begin
        Form8.show;
        Form3.Release;
        form3.Close;
      end
      else
      begin
        MessageBeep(MB_ICONQUESTION);
        m:=application.MessageBox('Wrong "user ıd" or "password"!','Mistake',0+48);
        edit1.Text:='';
        edit2.Text:='';
        edit1.SetFocus;
    end;
```

```
        end;
if query1.Fields[0].AsString<>edit1.Text then
begin
   MessageBeep(MB_ICONQUESTION);
    m:=application.MessageBox('Wrong "user ıd" or "password"!','Mistake',0+48);
    edit1.Text:='';
    edit2.Text:='';
    edit1.SetFocus;

end;
end;




procedure TForm3.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
begin
key:=#0;
edit2.setfocus;
end;
end;

procedure TForm3.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
begin
key:=#0;
speedbutton2.Click;
end;
end;



procedure TForm3.FormShow(Sender: TObject);
begin
Form1.Enabled:=False;
end;

procedure TForm3.FormKeyPress(Sender: TObject; var Key: Char);
begin
if key=#27 then    speedbutton1.Click;
end;

procedure TForm3.Timer1Timer(Sender: TObject);
begin
```

```
a:=not a;
flashwindow(self.Handle,a);
end;

end.
```

Form 4 (Entry Form Of Service Card)

```
unit Unit4;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, DBCtrls, StdCtrls, Buttons, Mask, DB, DBTables,
  ComCtrls;

type
  TForm4 = class(TForm)
    Label1: TLabel;
    SpeedButton3: TSpeedButton;
    GroupBox1: TGroupBox;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    GroupBox2: TGroupBox;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    GroupBox3: TGroupBox;
    Label13: TLabel;
    Label14: TLabel;
    GroupBox4: TGroupBox;
    Label15: TLabel;
    Label16: TLabel;
    Label17: TLabel;
    Label18: TLabel;
    Label19: TLabel;
    Label20: TLabel;
    Label21: TLabel;
    Label22: TLabel;
    Label23: TLabel;
```

```
Label24: TLabel;
Label25: TLabel;
Label26: TLabel;
Label27: TLabel;
Label28: TLabel;
Label29: TLabel;
Label31: TLabel;
Label33: TLabel;
SpeedButton4: TSpeedButton;
Label34: TLabel;
Label35: TLabel;
DBEdit3: TDBEdit;
DBEdit4: TDBEdit;
DBEdit6: TDBEdit;
DBEdit7: TDBEdit;
DBEdit8: TDBEdit;
DBRadioGroup2: TDBRadioGroup;
DBRadioGroup3: TDBRadioGroup;
DBEdit9: TDBEdit;
DBEdit10: TDBEdit;
DBEdit11: TDBEdit;
DBEdit12: TDBEdit;
DBEdit13: TDBEdit;
DBEdit14: TDBEdit;
DBEdit15: TDBEdit;
DBEdit16: TDBEdit;
DBEdit17: TDBEdit;
DBEdit18: TDBEdit;
DBEdit19: TDBEdit;
DBEdit20: TDBEdit;
DBEdit21: TDBEdit;
DBEdit22: TDBEdit;
DBEdit25: TDBEdit;
DBComboBox1: TDBComboBox;
DBEdit26: TDBEdit;
DBEdit27: TDBEdit;
DBEdit28: TDBEdit;
DBComboBox2: TDBComboBox;
DBEdit29: TDBEdit;
Label36: TLabel;
DBEdit30: TDBEdit;
DBComboBox3: TDBComboBox;
DBComboBox4: TDBComboBox;
DBComboBox5: TDBComboBox;
DBComboBox6: TDBComboBox;
SpeedButton6: TSpeedButton;
SpeedButton7: TSpeedButton;
SpeedButton8: TSpeedButton;
SpeedButton9: TSpeedButton;
SpeedButton5: TSpeedButton;
```

```
  DataSource1: TDataSource;
  Query1: TQuery;
  DBCheckBox1: TDBCheckBox;
  DBEdit1: TDBEdit;
  DBEdit5: TDBEdit;
  DBEdit2: TDBEdit;
  DataSource2: TDataSource;
  Query2: TQuery;
  DataSource3: TDataSource;
  Query3: TQuery;
  Query4: TQuery;
  DataSource4: TDataSource;
  DataSource5: TDataSource;
  Query5: TQuery;
  Query6: TQuery;
  DataSource6: TDataSource;
  Query7: TQuery;
  DataSource7: TDataSource;
  SpeedButton1: TSpeedButton;
  DataSource8: TDataSource;
  Query8: TQuery;
  procedure SpeedButton3Click(Sender: TObject);
  procedure SpeedButton4Click(Sender: TObject);
  procedure SpeedButton6Click(Sender: TObject);
  procedure SpeedButton7Click(Sender: TObject);

  procedure SpeedButton5Click(Sender: TObject);
  procedure SpeedButton8Click(Sender: TObject);
  procedure FormShow(Sender: TObject);

  procedure DBEdit5Exit(Sender: TObject);
  procedure DBEdit1KeyPress(Sender: TObject; var Key: Char);
  procedure DBEdit2DblClick(Sender: TObject);
  procedure DBEdit3DblClick(Sender: TObject);
  procedure DBEdit4DblClick(Sender: TObject);
  procedure FormKeyPress(Sender: TObject; var Key: Char);
  procedure DBComboBox2Change(Sender: TObject);
  procedure SpeedButton1Click(Sender: TObject);
  procedure DBComboBox4Change(Sender: TObject);


  procedure DBComboBox3DropDown(Sender: TObject);
  procedure DBComboBox4DropDown(Sender: TObject);
  procedure DBComboBox2DropDown(Sender: TObject);
  procedure DBComboBox6DropDown(Sender: TObject);
  procedure FormActivate(Sender: TObject);
  procedure DBComboBox1DropDown(Sender: TObject);
  procedure SpeedButton9Click(Sender: TObject);
private
  { Private declarations }
```

```
  public
    { Public declarations }
  end;

var
  Form4: TForm4;

implementation

uses Unit5, Unit6, Unit13;

{$R *.dfm}


procedure TForm4.SpeedButton3Click(Sender: TObject);
begin
form4.Close;
query1.Refresh;
end;

procedure TForm4.SpeedButton4Click(Sender: TObject);
begin
form6.showmodal;
end;

procedure TForm4.SpeedButton6Click(Sender: TObject);
begin
if (dbedit1.Text<>'') and (dbedit5.Text<>'') then
begin
query1.Insert;
query1.Fields[3].AsString:=dbedit1.Text;
query1.Fields[4].AsString:=dbedit2.text;
query1.Fields[5].AsString:=Dbedit3.Text;
query1.Fields[6].AsString:=Dbedit4.Text;
query1.Fields[7].AsString:=dbedit5.Text;
query1.Fields[8].AsString:=Dbcombobox3.Text;
query1.Fields[9].AsString:=Dbcombobox4.Text;
query1.Fields[10].AsString:=Dbcombobox5.Text;
query1.Fields[11].AsString:=Dbedit6.Text;
query1.Fields[12].AsString:=Dbedit7.Text;
query1.Fields[13].AsString:=Dbedit8.Text;
query1.Fields[15].AsString:=Dbcombobox2.Text;
query1.Fields[16].AsString:=Dbedit30.Text;
query1.Fields[17].AsString:=Dbedit29.Text;
query1.Fields[19].AsString:=Dbedit9.Text;
query1.Fields[20].AsString:=Dbedit13.Text;
query1.Fields[21].AsString:=Dbedit14.Text;
query1.Fields[22].AsString:=Dbedit15.Text;
query1.Fields[23].AsString:=Dbedit10.Text;
query1.Fields[24].AsString:=Dbcombobox6.Text;
```

```
query1.Fields[25].AsString:=Dbedit11.Text;
query1.Fields[26].AsString:=Dbedit12.Text;
query1.Fields[27].AsString:=Dbedit16.text;
query1.Fields[28].AsString:=Dbedit17.Text;
query1.Fields[29].AsString:=Dbedit18.Text;
query1.Fields[30].AsString:=Dbedit19.Text;
query1.Fields[31].AsString:=Dbedit20.Text;
query1.Fields[32].AsString:=Dbedit21.Text;
query1.Fields[33].AsString:=Dbedit22.Text;
query1.Fields[36].AsString:=Dbedit25.Text;
query1.Fields[37].AsString:=dbcombobox1.Text;
query1.Fields[38].AsString:=Dbedit26.Text;
query1.Fields[39].AsString:=Dbedit27.Text;
end
else
begin
dbedit1.SetFocus;
Showmessage('Must enter "document number" and "serial number"');
end;
end;

procedure TForm4.SpeedButton7Click(Sender: TObject);
begin
query1.edit;

end;

procedure TForm4.SpeedButton5Click(Sender: TObject);
begin
query1.prior;


end;

procedure TForm4.SpeedButton8Click(Sender: TObject);
var
m:byte;
begin
MessageBeep(MB_ICONQUESTION);
m:=application.MessageBox('Are you sure?','Delete Record',4+32+256+4096);
if m=idyes then
begin
if not query1.IsEmpty then
query1.Delete;
end;
end;
procedure TForm4.FormShow(Sender: TObject);
var i:integer;
begin
dbedit1.SetFocus;
```

```
end;

procedure TForm4.DBEdit5Exit(Sender: TObject);
begin
if dbedit5.Text='' then
begin
Showmessage('Have to enter "serial number"');
dbedit5.Text:='';
Dbedit5.SetFocus;
end;
end;

procedure TForm4.DBEdit1KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
begin
key:=#0;
dbedit5.SetFocus;
end;
end;

procedure TForm4.DBEdit2DblClick(Sender: TObject);
begin
dbedit2.Text:=datetostr(Date);
end;

procedure TForm4.DBEdit3DblClick(Sender: TObject);
begin
dbedit3.Text:=datetostr(Date);
end;

procedure TForm4.DBEdit4DblClick(Sender: TObject);
begin
dbedit4.Text:=datetostr(Date);
end;

procedure TForm4.FormKeyPress(Sender: TObject; var Key:Char );
var
t:Word;

begin
  if key=#27 then form4.Close;
  if Key = #13 then
  begin
    key:=#0;{ }
    Perform(WM_NEXTDLGCTL, 0, 0);  {}
  end;
end;
procedure TForm4.DBComboBox2Change(Sender: TObject);
begin
```

```
query5.First;
dbedit30.text:='';
while not query5.Eof do
begin
if dbcombobox2.ItemIndex=strtoint(query5.Fields[0].asstring)-1 then
dbedit30.Text:=query5.Fields[1].AsString;
query5.Next;
end;
  end;
procedure TForm4.SpeedButton1Click(Sender: TObject);
begin
query1.Post;
end;

procedure TForm4.DBComboBox4Change(Sender: TObject);
var
s:string;
i:integer;
begin
for i:=0 to dbcombobox4.Items.Count do
begin
 if dbcombobox4.ItemIndex=i then
 begin
      s:=Dbcombobox4.text;
      query4.First;
      dbcombobox5.Clear;
      while not query4.Eof do
      begin
            if query4.Fields[1].AsString=s then
            begin
            dbcombobox5.Items.Add(query4.Fields[0].asstring);
            end;
      query4.Next;
      end;
 end;
end;
end;


procedure TForm4.DBComboBox3DropDown(Sender: TObject);
begin
query2.First;
dbcombobox3.Clear;
while not query2.Eof do
begin
dbcombobox3.Items.Add(query2.Fields[0].asstring);
query2.Next;
end;
end;
```

```
procedure TForm4.DBComboBox4DropDown(Sender: TObject);
begin
query3.First;
dbcombobox4.Clear;
while not query3.Eof do
begin
dbcombobox4.Items.Add(query3.Fields[0].asstring);
query3.Next;
end;

end;

procedure TForm4.DBComboBox2DropDown(Sender: TObject);
begin
query5.First;
dbcombobox2.Clear;
while not query5.Eof do
begin
dbcombobox2.Items.Add(query5.Fields[0].asstring);
query5.Next;
end;
end;


procedure TForm4.DBComboBox6DropDown(Sender: TObject);
begin
query6.First;
dbcombobox6.Clear;
while not query6.Eof do
begin
dbcombobox6.Items.Add(query6.Fields[0].asstring);
query6.Next;
end;
end;

procedure TForm4.FormActivate(Sender: TObject);
begin
query8.First;
while query8.Eof do begin
dbcombobox1.Items.Add(query8.Fields[0].asstring);
query8.Next;
end;

end;

procedure TForm4.DBComboBox1DropDown(Sender: TObject);
begin
query8.First;
```

```
while query8.Eof do begin
dbcombobox1.Items.Add(query8.Fields[0].asstring);
query8.Next;
end;
end;

procedure TForm4.SpeedButton9Click(Sender: TObject);
begin;
query1.next;


end;

end.
```

Form 6 (Used Device Form)

```
unit Unit6;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, ExtCtrls, DBCtrls, StdCtrls, Grids, DBGrids, DB,
  DBTables, Menus, Mask;

type
  TForm6 = class(TForm)
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    SpeedButton3: TSpeedButton;
    Panel1: TPanel;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Label4: TLabel;
    Label5: TLabel;
    Edit4: TEdit;
    Edit5: TEdit;
    DBGrid1: TDBGrid;
    Label6: TLabel;
    Edit6: TEdit;
    Label7: TLabel;
    Edit7: TEdit;
    Edit8: TEdit;
    Label10: TLabel;
    SpeedButton4: TSpeedButton;
```

```
    Query1: TQuery;
    DataSource1: TDataSource;
    SpeedButton5: TSpeedButton;
    SpeedButton6: TSpeedButton;
    procedure SpeedButton3Click(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
    procedure SpeedButton4Click(Sender: TObject);
    procedure SpeedButton5Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure SpeedButton6Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form6: TForm6;

implementation

uses Unit11, Unit4;

{$R *.dfm}

procedure TForm6.SpeedButton3Click(Sender: TObject);
begin
form6.Close;
edit6.Text:='';
edit7.Text:='';
edit8.Text:='';
form4.Query1.Edit;
form4.Query1.Fields[32].AsString:=query1.Fields[9].AsString;
form4.Query1.Fields[33].AsString:=query1.Fields[10].AsString;
form4.Query1.Fields[36].AsString:=query1.Fields[11].AsString;
form4.query1.Post;
end;

procedure TForm6.SpeedButton1Click(Sender: TObject);
begin
form11.showmodal;
form11.PopupMenu1.Free;
end;

procedure TForm6.FormShow(Sender: TObject);
begin
query1.SQL.Clear;
```

```
query1.SQL.Text:='Select * from KULLANILAN where FISNO
like'+#39+form4.Query1.Fields[0].AsString+'%'+#39;
query1.Open;
end;

procedure TForm6.DBGrid1DblClick(Sender: TObject);
begin
query1.Refresh;
end;

procedure TForm6.SpeedButton4Click(Sender: TObject);
begin
Query1.Delete;
end;

procedure TForm6.SpeedButton5Click(Sender: TObject);
var
T,S:string;
Toplam:double;
begin
query1.Insert;
fORM6.query1.Fields[0].AsString:=form4.Query1.Fields[0].AsString;
query1.Fields[6].AsString:='0';
query1.Fields[1].AsString:=edit1.Text;
query1.Fields[2].Asstring:=edit2.Text;
query1.Fields[3].AsString:=edit3.Text;
query1.Fields[4].AsString:=edit4.Text;
query1.Fields[5].AsString:=edit5.Text;
S:=inttostr(strtoint(form6.query1.Fields[3].AsString)*
strtoint(form6.Query1.Fields[2].Asstring));
T:=inttostr(strtoint(S)* strtoint(form6.query1.Fields[4].AsString));
Toplam:=(strtoint(T)/100);
form6.Query1.Fields[7].AsFloat:=Toplam+strtoint(S);
query1.Fields[8].AsString:='Not Have Stock';
query1.Post;
edit1.Text:='';
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
edit5.Text:='';
end;

procedure TForm6.SpeedButton2Click(Sender: TObject);
var
mesaj:string;
begin
query1.Insert;
fORM6.query1.Fields[0].AsString:=form4.Query1.Fields[0].AsString;
query1.Fields[6].AsString:='0';
query1.Fields[2].AsString:='1';
```

```
mesaj:=inputbox('Worker Pay','Enter Workers pay','20');
query1.Fields[3].AsString:=mesaj;
query1.Fields[10].AsString:=mesaj;
query1.Fields[4].AsString:='0';
query1.Fields[7].AsString:=mesaj;
query1.Fields[8].AsString:='Not Have Stock';
query1.Fields[5].AsString:='Workers pay';
query1.Post;
end;


procedure TForm6.SpeedButton6Click(Sender: TObject);
var
Toplam,Toplam2,T:double;
S,ISC:string;
kdv,kdv1:double;
m:double;
begin
Toplam2:=0;
Toplam:=0;
S:='0';
T:=0;
query1.First;
while not query1.Eof do
begin
if query1.Fields[10].asstring='' then
Toplam:=query1.Fields[7].AsFloat+Toplam;
if query1.Fields[5].AsString='Workers pay' then
edit7.Text:=Query1.Fields[10].Text;
query1.Next;
end;
Edit6.Text:=floattostr(Toplam);
if edit7.Text<>'' then
begin
Edit8.Text:=floattostr(strtofloat(edit6.Text)+strtofloat(edit7.Text));
end;
query1.Edit;
query1.Fields[11].Asstring:=edit8.Text;
query1.Fields[9].AsString:=Edit6.Text;
query1.Fields[10].AsString:=edit7.Text;
query1.Post;
end;
end.
```

Form 8 (Rate Of Exchange for Dolar\Euro)

```
unit Unit8;

interface
```

```
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, StdCtrls, ExtCtrls, Mask, DB, DBTables;

type
  TForm8 = class(TForm)
    Edit1: TEdit;
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    MaskEdit1: TMaskEdit;
    GroupBox3: TGroupBox;
    MaskEdit2: TMaskEdit;
    GroupBox4: TGroupBox;
    RadioGroup1: TRadioGroup;
    Label1: TLabel;
    Label2: TLabel;
    Edit2: TEdit;
    Edit3: TEdit;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    Edit4: TEdit;
    Label3: TLabel;
    Label4: TLabel;
    Edit5: TEdit;
    Label5: TLabel;
    DataSource1: TDataSource;
    Query1: TQuery;
    procedure FormShow(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure RadioGroup1Click(Sender: TObject);
    procedure MaskEdit1KeyPress(Sender: TObject; var Key: Char);
    procedure SpeedButton2Click(Sender: TObject);
    procedure MaskEdit2KeyPress(Sender: TObject; var Key: Char);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormKeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form8: TForm8;

implementation

uses Unit1;

{$R *.dfm}
```

```
procedure TForm8.FormShow(Sender: TObject);
begin
edit1.Text:=datetostr(date);
edit4.Text:=timetostr(Time);
end;

procedure TForm8.SpeedButton1Click(Sender: TObject);
var
sonuc:string;
sonuc2:string;
begin
if radiogroup1.ItemIndex=0 then
edit3.Text:=floattostr(strtofloat(maskedit1.Text)* strtofloat(edit2.Text));
if radiogroup1.ItemIndex=1 then
edit2.Text:=floattostr(strtofloat(edit3.Text)/strtofloat(maskedit1.Text));
if radiogroup1.ItemIndex=2 then
edit3.Text:=floattostr(strtofloat(maskedit2.Text)* strtofloat(edit5.Text));
if radiogroup1.ItemIndex=3 then
edit5.Text:=floattostr(strtofloat(edit3.Text)/strtofloat(maskedit2.Text));
if radiogroup1.ItemIndex=4 then
begin
sonuc:=floattostr(strtofloat(maskedit1.Text)* strtofloat(edit2.Text));
edit5.Text:=floattostr(strtofloat(sonuc)/strtofloat(maskedit2.Text));
end;
if radiogroup1.ItemIndex=5 then
begin
sonuc2:=floattostr(strtofloat(maskedit2.Text)* strtofloat(edit5.Text));
edit2.Text:=floattostr(strtofloat(sonuc2)/strtofloat(maskedit1.Text));
end;
end;

procedure TForm8.RadioGroup1Click(Sender: TObject);
begin
if radiogroup1.ItemIndex=0 then
begin
edit2.Enabled:=true;
edit3.Enabled:=false;
edit5.Enabled:=false;
end;
if radiogroup1.ItemIndex=1 then
begin
edit3.Enabled:=true;
edit2.Enabled:=false;
edit5.Enabled:=false;
end;
if radiogroup1.ItemIndex=2 then
begin
edit5.Enabled:=true;
edit2.Enabled:=false;
```

```
edit3.Enabled:=false;
end;
if radiogroup1.ItemIndex=3 then
begin
edit3.Enabled:=true;
edit2.Enabled:=false;
edit5.Enabled:=false;
end;
if radiogroup1.ItemIndex=4 then
begin
edit2.Enabled:=true;
edit5.Enabled:=false;
edit3.Enabled:=false;
end;
if radiogroup1.ItemIndex=5 then
begin
edit5.Enabled:=true;
edit2.Enabled:=false;
edit3.Enabled:=false;
end;

end;

procedure TForm8.MaskEdit1KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
maskedit2.SetFocus;
end;

procedure TForm8.SpeedButton2Click(Sender: TObject);
begin
query1.Edit;
query1.Fields[0].AsString:=maskedit1.Text;
query1.Fields[1].AsString:=maskedit2.Text;
query1.Post;
form1.enabled:=True;
form8.Close;
end;

procedure TForm8.MaskEdit2KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
begin
key:=#0;
Query1.insert;
Query1.Fields[0].AsString:=maskedit1.Text;
Query1.Fields[1].AsString:=maskedit2.Text;
query1.Post;
end;
end;
```

```
procedure TForm8.FormClose(Sender: TObject; var Action: TCloseAction);
begin
form1.enabled:=True;
end;

procedure TForm8.FormKeyPress(Sender: TObject; var Key: Char);
begin
if key=#27 then form8.Close;
end;

end.
```

Form 9(Serach Form Of Service Card)

```
unit Unit9;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, ComCtrls, StdCtrls, Buttons, ExtCtrls, DBCtrls,
  Mask, DB, DBTables;

type
  TForm9 = class(TForm)
    Panel1: TPanel;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    SpeedButton1: TSpeedButton;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    CheckBox3: TCheckBox;
    SpeedButton2: TSpeedButton;
    SpeedButton3: TSpeedButton;
    DBGrid1: TDBGrid;
    DateTimePicker1: TDateTimePicker;
    DateTimePicker2: TDateTimePicker;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    ComboBox1: TComboBox;
    ComboBox2: TComboBox;
```

```
    ComboBox3: TComboBox;
    Query2: TQuery;
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure SpeedButton3Click(Sender: TObject);
    procedure Edit3Change(Sender: TObject);
    procedure Edit4Change(Sender: TObject);
    procedure Edit1Change(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure ComboBox2Change(Sender: TObject);
    procedure ComboBox1Change(Sender: TObject);
    procedure ComboBox3Change(Sender: TObject);
    procedure DateTimePicker1Change(Sender: TObject);
    procedure DateTimePicker2Change(Sender: TObject);
    procedure Edit2Change(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form9: TForm9;

implementation

uses Unit4;

{$R *.dfm}

procedure TForm9.SpeedButton1Click(Sender: TObject);
begin
form9.Close;
form4.Query1.SQL.Clear;
form4.Query1.SQL.Text:='select * from GIRIS';
Form4.Query1.Open;
end;



procedure TForm9.SpeedButton2Click(Sender: TObject);
begin
edit1.Text:='';
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
combobox1.Text:='';
combobox2.Text:='';
combobox3.Text:='';
```

```
checkbox1.Checked:=False;
checkbox2.Checked:=False;
checkbox3.Checked:=False;
edit1.SetFocus;
form4.query1.SQL.Clear;
end;

procedure TForm9.SpeedButton3Click(Sender: TObject);
begin
form4.Query1.Open;
form4.showmodal;
end;

procedure TForm9.Edit3Change(Sender: TObject);
begin
if checkbox3.Checked then
begin
form4.query1.sql.Clear;
form4.query1.SQL.Text:='select * from GIRIS  where ISIM
like'+#39+edit3.Text+'%'+#39;
form4.query1.open;
end;
end;

procedure TForm9.Edit4Change(Sender: TObject);
begin
if checkbox3.Checked then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='select * from GIRIS  where SOYAD
like'+#39+edit4.Text+'%'+#39;
form4.query1.open;
end;
end;

procedure TForm9.Edit1Change(Sender: TObject);
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='select * from GIRIS  where BELGENO
like'+#39+edit1.Text+'%'+#39;
form4.query1.open;

end;

procedure TForm9.FormShow(Sender: TObject);
begin
edit1.SetFocus;
form4.query2.First;
combobox1.Clear;
while not form4.query2.Eof do
```

```
begin
combobox1.Items.Add(form4.query2.Fields[0].asstring);
form4.query2.Next;
end;

form4.query3.First;
combobox2.Clear;
while not form4.query3.Eof do
begin
combobox2.Items.Add(form4.query3.Fields[0].asstring);
form4.query3.Next;
end;

form4.query4.First;
combobox3.Clear;
while not form4.query4.Eof do
begin
combobox3.Items.Add(form4.query4.Fields[0].asstring);
form4.query4.Next;
end;


end;

procedure TForm9.ComboBox2Change(Sender: TObject);
var
s:string;
i:integer;
begin
if checkbox3.Checked then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='Select * from GIRIS where CINSI
like'+#39+combobox2.Text+'%'+#39;
form4.query1.Open;
end;
for i:=0 to combobox2.Items.Count do
begin
 if combobox2.ItemIndex=i then
 begin
      s:=combobox2.text;
      query2.First;
      combobox3.Clear;
      while not query2.Eof do
      begin
           if query2.Fields[1].AsString=s then
           begin
           combobox3.Items.Add(query2.Fields[0].asstring);
           end;
      query2.Next;
```

```
      end;
 end;
end;
end;
procedure TForm9.ComboBox1Change(Sender: TObject);
begin
if checkbox3.Checked then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='Select * from GIRIS where MARKA
like'+#39+combobox1.Text+'%'+#39;
form4.query1.Open;
end;
end;
procedure TForm9.ComboBox3Change(Sender: TObject);
begin
if checkbox3.Checked then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='Select * from GIRIS where MODELI
like'+#39+combobox3.Text+'%'+#39;
form4.query1.Open;
end;
end;

procedure TForm9.DateTimePicker1Change(Sender: TObject);
var
s:string;
begin
s:=Datetostr(datetimepicker1.Date);
if checkbox1.Checked then
begin
form4.Query1.SQL.Clear;
form4.Query1.SQL.Text:='Select * from GIRIS where BASVURUTARIHI
like'+#39+s+'%'+#39;
form4.Query1.Open;
end;
end;

procedure TForm9.DateTimePicker2Change(Sender: TObject);
var
s:string;
begin
s:=Datetostr(datetimepicker2.Date);
if checkbox2.Checked then
begin
form4.Query1.SQL.Clear;
form4.Query1.SQL.Text:='Select * from GIRIS where TESLIMTARIHI
like'+#39+s+'%'+#39;
form4.Query1.Open;
```

```
end;
end;

procedure TForm9.Edit2Change(Sender: TObject);
begin
if checkbox3.Checked then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='select * from GIRIS  where SERINO
like'+#39+edit2.Text+'%'+#39;
form4.query1.open;
end;
end;
end.
```

Form 10 (Failure Observation Form)

```
unit Unit10;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ComCtrls, StdCtrls, Buttons, ExtCtrls, Grids, DBGrids, DB,
  DBTables;

type
  TForm10 = class(TForm)
    RadioGroup1: TRadioGroup;
    GroupBox2: TGroupBox;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    RadioGroup2: TRadioGroup;
    ComboBox1: TComboBox;
    ComboBox2: TComboBox;
    ComboBox3: TComboBox;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    Edit5: TEdit;
    Label11: TLabel;
    Edit6: TEdit;
```

```
    DBGrid1: TDBGrid;
    SpeedButton3: TSpeedButton;
    SpeedButton4: TSpeedButton;
    procedure SpeedButton2Click(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton3Click(Sender: TObject);
    procedure FormShow(Sender: TObject);

    procedure RadioGroup1Click(Sender: TObject);
    procedure RadioGroup2Click(Sender: TObject);

    procedure SpeedButton4Click(Sender: TObject);
    procedure ComboBox1DropDown(Sender: TObject);
    procedure ComboBox3DropDown(Sender: TObject);
    procedure ComboBox3Change(Sender: TObject);
    procedure ComboBox1Change(Sender: TObject);
    procedure ComboBox2Change(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form10: TForm10;

implementation

uses Unit5, Unit4, Unit7, Unit9;

{$R *.dfm}

procedure TForm10.SpeedButton2Click(Sender: TObject);
begin
form4.Query1.Open;
form4.showmodal;
end;

procedure TForm10.SpeedButton1Click(Sender: TObject);
begin
form7.QuickRep1.Preview;
end;

procedure TForm10.SpeedButton3Click(Sender: TObject);
begin
form10.Close;
form4.Query1.SQL.Clear;
form4.Query1.SQL.Text:='select * from GIRIS';
Form4.Query1.Open;
```

```
end;

procedure TForm10.FormShow(Sender: TObject);
begin
form4.query2.First;
combobox1.Clear;
while not form4.query2.Eof do
begin
combobox1.Items.Add(form4.query2.Fields[0].asstring);
form4.query2.Next;
end;
form4.query3.First;
combobox3.Clear;
while not form4.query3.Eof do
begin
combobox3.Items.Add(form4.query3.Fields[0].asstring);
form4.query3.Next;
end;
form4.query4.First;
combobox2.Clear;
while not form4.query4.Eof do
begin
combobox2.Items.Add(form4.query4.Fields[0].asstring);
form4.query4.Next;
end;
end;

procedure TForm10.RadioGroup1Click(Sender: TObject);
begin
if radiogroup1.ItemIndex=0 then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='select * from GIRIS';
form4.Query1.Open;
end;
if radiogroup1.ItemIndex=1 then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='select * from GIRIS  where CIHAZDURUMU
like'+#39+'Wait For Repair'+'%'+#39;
form4.Query1.Open;
end;
if radiogroup1.ItemIndex=2 then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='select * from GIRIS  where CIHAZDURUMU
like'+#39+'Wait For Device'+'%'+#39;
form4.Query1.Open;
end;
if radiogroup1.ItemIndex=3 then
```

```
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='select * from GIRIS  where CIHAZDURUMU
like'+#39+'Repaired System'+'%'+#39;
form4.Query1.Open;
end;
if radiogroup1.ItemIndex=4 then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='select * from GIRIS  where CIHAZDURUMU
like'+#39+'Delivered Systems'+'%'+#39;
form4.Query1.Open;
end;
if radiogroup1.ItemIndex=5 then
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='select * from GIRIS  where CIHAZDURUMU
like'+#39+'Made Fitting'+'%'+#39;
form4.Query1.Open;
end;



end;

procedure TForm10.RadioGroup2Click(Sender: TObject);
begin
if radiogroup2.ItemIndex=0 then
begin
form4.query1.SQL.Text:='select * from GIRIS  where GARANTI
like'+#39+'Quarantee'+'%'+#39;
form4.Query1.Open;
end;
if radiogroup2.ItemIndex=1 then
begin
form4.query1.SQL.Text:='select * from GIRIS  where GARANTI
like'+#39+'Paid'+'%'+#39;
form4.Query1.Open;
end;
if radiogroup2.ItemIndex=2 then
begin
form4.query1.SQL.Text:='select * from GIRIS';
form4.Query1.Open;
end;

end;


procedure TForm10.SpeedButton4Click(Sender: TObject);
begin
```

```
form4.Query1.SQL.Clear;
combobox1.Text:='';
combobox2.Text:='';
combobox3.Text:='';
end;

procedure TForm10.ComboBox1DropDown(Sender: TObject);
begin
form4.query2.First;
combobox1.Clear;
while not form4.query2.Eof do
begin
combobox1.Items.Add(form4.query2.Fields[0].asstring);
form4.query2.Next;
end;

end;

procedure TForm10.ComboBox3DropDown(Sender: TObject);
begin
form4.query3.First;
combobox3.Clear;
while not form4.query3.Eof do
begin
combobox3.Items.Add(form4.query3.Fields[0].asstring);
form4.query3.Next;

end;
end;
procedure TForm10.ComboBox3Change(Sender: TObject);
var
s:string;
i:integer;
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='Select * from GIRIS where CINSI
like'+#39+combobox3.Text+'%'+#39;
form4.query1.Open;
for i:=0 to combobox3.Items.Count do
begin
if combobox3.ItemIndex=i then
begin
    s:=combobox3.text;
    form4.query2.First;
    combobox2.Clear;
    while not form4.query2.Eof do
    begin
        if form4.query2.Fields[1].AsString=s then
        begin
        combobox2.Items.Add(form4.query2.Fields[0].asstring);
```

```
            end;
        form4.query2.Next;
        end;
 end;
end;

end;

procedure TForm10.ComboBox1Change(Sender: TObject);
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='Select * from GIRIS where MARKA
like'+#39+combobox1.Text+'%'+#39;
form4.query1.Open;
end;

procedure TForm10.ComboBox2Change(Sender: TObject);
begin
form4.query1.SQL.Clear;
form4.query1.SQL.Text:='Select * from GIRIS where MODELI
like'+#39+combobox2.Text+'%'+#39;
form4.query1.Open;
end;

end.
```

Form 11 (Device Entry Form)

```
unit Unit11;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, ExtCtrls, DBCtrls, StdCtrls, Grids, DBGrids, ComCtrls,
  DB, DBTables, Menus,math;

type
  TForm11 = class(TForm)
    SpeedButton1: TSpeedButton;
    Panel1: TPanel;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label11: TLabel;
    Label17: TLabel;
    Label18: TLabel;
    Label4: TLabel;
```

```
    Edit1: TEdit;
    Edit2: TEdit;
    ComboBox1: TComboBox;
    Edit3: TEdit;
    Edit4: TEdit;
    Edit6: TEdit;
    Edit7: TEdit;
    DataSource1: TDataSource;
    Query1: TQuery;
    SpeedButton4: TSpeedButton;
    Edit9: TEdit;
    DataSource2: TDataSource;
    Query2: TQuery;
    DataSource3: TDataSource;
    Query3: TQuery;
    PopupMenu1: TPopupMenu;
    Ekle1: TMenuItem;
    Edit5: TEdit;
    Label7: TLabel;
    SpeedButton7: TSpeedButton;
    Edit8: TEdit;
    Edit10: TEdit;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Timer1: TTimer;
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    DBGrid1: TDBGrid;
    procedure SpeedButton1Click(Sender: TObject);

    procedure ComboBox1Change(Sender: TObject);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure SpeedButton4Click(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
    procedure SpeedButton5Click(Sender: TObject);
    procedure SpeedButton6Click(Sender: TObject);
    procedure Ekle1Click(Sender: TObject);
    procedure SpeedButton7Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form11: TForm11;
```

```
implementation

uses Unit6, Unit4;

{$R *.dfm}

procedure TForm11.SpeedButton1Click(Sender: TObject);
begin
form11.close;
end;



procedure TForm11.ComboBox1Change(Sender: TObject);
begin
if combobox1.ItemIndex=0 then
begin
edit3.Text:=floattostr(strtofloat(edit2.Text) * strtofloat(query2.Fields[0].AsString));
end;
if combobox1.ItemIndex=1 then
begin
edit3.Text:=floattostr(strtofloat(edit2.Text) * strtofloat(query2.Fields[1].AsString));
end;
end;

procedure TForm11.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
begin
 query1.SQL.Clear;
 query1.SQL.Text:='select * from PARCAGIRIS where PARCANO
like'+#39+edit1.Text+'%'+#39;
 query1.Open;
 if query1.Fields[0].AsString=edit1.Text then
 begin
 edit2.Text:=query1.Fields[1].AsString;
 combobox1.Text:=query1.Fields[12].AsString;
 edit3.Text:=query1.Fields[2].AsString;
 edit4.Text:=query1.Fields[3].AsString;
 edit5.Text:=query1.Fields[4].AsString;
 edit6.Text:=query1.Fields[5].AsString;
 edit9.Text:=query1.Fields[13].AsString;
 edit7.Text:=query1.Fields[6].AsString;
 end;
 end;
end;

procedure TForm11.SpeedButton4Click(Sender: TObject);
var
m:byte;
```

```
begin
 query3.Active:=false;
if edit1.Text='' then
begin
showmessage('Have to enter device no');
edit1.SetFocus;
end
else
 begin
 query1.SQL.Clear;
 query1.SQL.Text:='select * from PARCAGIRIS where PARCANO
like'+#39+edit1.Text+'%'+#39;
 query1.Open;
 if query1.Fields[0].AsString=edit1.Text then
 begin
    MessageBeep(MB_ICONQUESTION);
   m:=application.MessageBox('there are in stock this
device?','Question',4+32+256+4096);
   if m=idyes then
   begin
   query1.Edit;
   query1.Fields[1].AsString:=edit2.Text;
   query1.Fields[12].AsString:=combobox1.Text;
   query1.Fields[2].AsString:=edit3.Text;
   query1.Fields[3].AsString:=edit4.Text;
   query1.Fields[4].AsString:=edit5.Text;
   query1.Fields[5].AsString:=edit6.Text;
   query1.Fields[13].AsString:=edit9.Text;
   query1.Fields[6].AsString:=edit7.Text;
   Query1.Fields[8].Asstring:=edit9.Text;
   query1.Fields[10].AsInteger:=query1.Fields[13].asinteger +
query1.Fields[10].AsInteger;
   query1.Post;
   edit1.Text:='';
   edit2.Text:='';
   combobox1.Text:='';
   edit3.Text:='';
   edit4.Text:='';
   edit5.Text:='';
   edit6.Text:='';
   edit9.Text:='';
   edit7.Text:='';
   edit1.SetFocus;
   end;
 end;
 if query1.Fields[0].AsString<>edit1.Text then
  begin
  query1.insert;
   query1.Fields[0].AsString:=edit1.Text;
   query1.Fields[1].AsString:=edit2.Text;
```

```
    query1.Fields[12].AsString:=combobox1.Text;
    query1.Fields[2].AsString:=edit3.Text;
    query1.Fields[3].AsString:=edit4.Text;
    query1.Fields[4].AsString:=edit5.Text;
    query1.Fields[5].AsString:=edit6.Text;
    query1.Fields[13].AsString:=edit9.Text;
    query1.Fields[6].AsString:=edit7.Text;
    Query1.Fields[8].Asstring:=edit9.Text;
    query1.Fields[10].AsInteger:=query1.Fields[13].asinteger +
query1.Fields[10].AsInteger;
    query1.Post;
    end;
    end;
     query3.Active:=true;
end;
procedure TForm11.DBGrid1DblClick(Sender: TObject);
begin
query3.Refresh;
end;

procedure TForm11.SpeedButton5Click(Sender: TObject);
var
 m:byte;
begin
 query1.SQL.Clear;
 query1.SQL.Text:='select * from PARCAGIRIS where PARCANO
like'+#39+edit1.Text+'%'+#39;
 query1.Open;
 if query1.Fields[0].AsString=edit1.Text then
 begin
     MessageBeep(MB_ICONQUESTION);
    m:=application.MessageBox('there is not in stock. add?','Question',4+32+256+4096);
    if m=idyes then
    begin
    query1.Edit;
    query1.Fields[1].AsString:=edit2.Text;
    query1.Fields[12].AsString:=combobox1.Text;
    query1.Fields[2].AsString:=edit3.Text;
    query1.Fields[3].AsString:=edit4.Text;
    query1.Fields[4].AsString:=edit5.Text;
    query1.Fields[5].AsString:=edit6.Text;
    query1.Fields[6].AsString:=edit7.Text;
    query1.Fields[9].Asinteger:=strtoint(edit9.Text);
    query1.Fields[8].asinteger:=query1.Fields[8].asinteger-query1.Fields[9].AsInteger;
    query1.Fields[10].AsInteger:=query1.Fields[10].AsInteger -
query1.Fields[9].asinteger;
    query1.Post;
    end;
end;
end;
```

```
procedure TForm11.SpeedButton6Click(Sender: TObject);
begin
query1.Delete;
end;

procedure TForm11.Ekle1Click(Sender: TObject);
var
T,S:string;
Toplam:double;
Miktar:string;
begin
if query1.Fields[0].AsString<>form4.Query1.Fields[0].AsString then
begin
query3.Edit;
if query3.Fields[10].AsInteger<1 then
begin
Showmessage('there is no in stock');
end
else
begin
FORM6.query1.Insert;
fORM6.query1.Fields[0].AsString:=form4.Query1.Fields[0].AsString;
form6.Query1.Fields[1].Asstring:=form11.Query3.Fields[5].AsString;
Miktar:=inputbox('Quantity','Enter Quantity','');
form6.Query1.Fields[2].Asinteger:=strtoint(Miktar);
form6.Query1.Fields[3].Asstring:=form11.Query3.Fields[2].AsString;
form6.Query1.Fields[4].Asstring:=form11.Query3.Fields[3].AsString;
form6.Query1.Fields[6].Asstring:=form11.Query3.Fields[0].AsString;
S:=inttostr(strtoint(form6.query1.Fields[3].AsString)*
strtoint(form6.Query1.Fields[2].Asstring));
T:=inttostr(strtoint(S)* strtoint(form6.query1.Fields[4].AsString));
Toplam:=(strtoint(T)/100);
form6.Query1.Fields[7].AsFloat:=Toplam+strtoint(S);
form6.Query1.Fields[8].AsString:='Not Have Stock';
query3.Fields[11].AsInteger:=query3.Fields[11].AsInteger+form6.Query1.Fields[2].Asi
nteger;
query3.Fields[10].AsInteger:=query3.Fields[10].AsInteger-
form6.Query1.Fields[2].Asinteger;
form6.query1.Post;
query3.Post;
end;
END;
end;


procedure TForm11.SpeedButton7Click(Sender: TObject);
begin
 edit1.Text:='';
  edit2.Text:='';
```

```
    combobox1.Text:='';
    edit3.Text:='';
    edit4.Text:='';
    edit5.Text:='';
    edit6.Text:='';
    edit9.Text:='';
    edit7.Text:='';
    edit1.SetFocus;

end;

procedure TForm11.Timer1Timer(Sender: TObject);
begin
query2.Close;
query2.SQL.Clear;
query2.SQL.Add('select * from KUR');
query2.Open;
query2.Last;
edit8.Text:=query2.Fields[0].AsString;
edit10.Text:=query2.Fields[1].AsString;

end;

end.
```

Form 12 (Entry\Outlet Form Of Cash Money)

```
unit Unit12;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ComCtrls, StdCtrls, ExtCtrls, DBCtrls, Grids, DBGrids, Buttons,
  DB, DBTables;

type
  TForm12 = class(TForm)
    SpeedButton1: TSpeedButton;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    Edit3: TEdit;
    ComboBox1: TComboBox;
    ComboBox2: TComboBox;
    Edit2: TEdit;
```

```
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    Query1: TQuery;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    Query2: TQuery;
    DataSource2: TDataSource;
    Query3: TQuery;
    DataSource3: TDataSource;
    Timer1: TTimer;
    procedure SpeedButton1Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form12: TForm12;

implementation

{$R *.dfm}

procedure TForm12.SpeedButton1Click(Sender: TObject);
begin
form12.Close;
end;

procedure TForm12.FormShow(Sender: TObject);
begin
form12.query2.First;
combobox1.Clear;
while not form12.query2.Eof do
begin
combobox1.Items.Add(form12.query2.Fields[0].asstring);
form12.query2.Next;
end;
end;

function seri(b:string):boolean;
begin
```

```
seri:=false;
form12.Query3.First;
while not form12.Query3.eof do
if (b=form12.Query3.Fields[0].asstring)then
begin
seri:=true;
exit;
end
else
form12.Query3.Next;
end;

procedure TForm12.BitBtn1Click(Sender: TObject);
var dto,eto:integer;
tto:integer;
begin
query1.Insert;
query1.Fields[0].AsString:=edit1.Text;
query1.Fields[1].AsString:=Combobox1.Text;
query1.Fields[2].AsString:=edit3.text;
query1.Fields[3].AsString:=Combobox2.Text;
query1.Fields[4].AsString:=edit2.Text;
query1.post;
query1.Active:=false;
query1.Active:=true;
seri(Combobox2.Text);
if (query3.Fields[0].asstring=combobox2.Text) then
begin
dto:=strtoint(edit3.Text)+strtoint(query3.Fields[1].AsString);
query3.Edit;
query3.Fields[1].AsString:=inttostr(dto);
query3.post;
end;
edit1.Text:='';
edit2.Text:='';
edit3.Text:='';
edit1.setfocus;
end;
procedure TForm12.BitBtn3Click(Sender: TObject);
begin
 query1.edit;
query1.Fields[0].AsString:=edit1.Text;
query1.Fields[1].AsString:=Combobox1.Text;
query1.Fields[2].AsString:=edit3.text;
query1.Fields[3].AsString:=Combobox2.Text;
query1.Fields[4].AsString:=edit2.Text;
query1.post;
query1.Active:=false;
query1.Active:=true;
edit1.Text:='';
```

```
edit2.Text:='';
edit3.Text:='';
edit1.setfocus;
end;

procedure TForm12.DBGrid1DblClick(Sender: TObject);
begin
edit1.Text:=query1.Fields[0].AsString;
Combobox1.Text:=query1.Fields[1].AsString;
edit3.text:=query1.Fields[2].AsString;
Combobox2.Text:=query1.Fields[3].AsString;
edit2.Text:=query1.Fields[4].AsString;
end;

procedure TForm12.BitBtn2Click(Sender: TObject);
begin
if not query1.Eof then
query1.delete;
edit3.text:='';
edit2.Text:='';
end;

procedure TForm12.FormActivate(Sender: TObject);
begin
edit1.Text:=datetostr(date);
end;

procedure TForm12.Timer1Timer(Sender: TObject);
begin
edit1.Text:=datetostr(date);
end;

end.
```

Form 13(Computer Sale Program [marks])

```
unit Unit13;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, Grids, DBGrids, ExtCtrls, DBCtrls, DB, DBTables,
  StdCtrls, Gauges;

type
  TForm13 = class(TForm)
    DBNavigator1: TDBNavigator;
    SpeedButton1: TSpeedButton;
```

```
    DataSource1: TDataSource;
    Query1: TQuery;
    DBGrid1: TDBGrid;
    procedure SpeedButton1Click(Sender: TObject);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);

private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form13: TForm13;

implementation

{$R *.dfm}

procedure TForm13.SpeedButton1Click(Sender: TObject);
begin
form13.Close;
end;



procedure TForm13.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
begin
key:=#0;
query1.Insert;
end;
if key=#46 then
begin
key:=#0;
query1.Delete;
end;
end;

end.
```

Form 14 (Computer Sale Program [Models])

```
unit Unit14;

interface

uses
```

```
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, Buttons, ExtCtrls, DBCtrls, DB, DBTables;

type
  TForm14 = class(TForm)
    DBNavigator1: TDBNavigator;
    DBGrid1: TDBGrid;
    SpeedButton1: TSpeedButton;
    Query1: TQuery;
    DataSource1: TDataSource;
    procedure SpeedButton1Click(Sender: TObject);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form14: TForm14;

implementation

{$R *.dfm}

procedure TForm14.SpeedButton1Click(Sender: TObject);
begin
form14.Close;
end;

procedure TForm14.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
begin
key:=#0;
query1.Insert;
end;
end;

end.
```

Form 15 (Computer Sale Program [system type])

```
unit Unit15;

interface

uses
```

```
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, Buttons, ExtCtrls, DBCtrls, DB, DBTables;

type
  TForm15 = class(TForm)
    DBNavigator1: TDBNavigator;
    DBGrid1: TDBGrid;
    SpeedButton1: TSpeedButton;
    DataSource1: TDataSource;
    Query1: TQuery;
    procedure SpeedButton1Click(Sender: TObject);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form15: TForm15;

implementation

{$R *.dfm}

procedure TForm15.SpeedButton1Click(Sender: TObject);
begin
form15.Close;
end;

procedure TForm15.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
begin
key:=#0;
query1.Insert;
end;
end;

end.
```

Form 16 (Computer Sale Program [Desription Of Deffect])

```
unit Unit16;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
```

```
    Dialogs, Grids, DBGrids, Buttons, ExtCtrls, DBCtrls, DB, DBTables;

  type
    TForm16 = class(TForm)
      DBNavigator1: TDBNavigator;
      DBGrid1: TDBGrid;
      SpeedButton1: TSpeedButton;
      DataSource1: TDataSource;
      Query1: TQuery;
      procedure SpeedButton1Click(Sender: TObject);
      procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
    private
      { Private declarations }
    public
      { Public declarations }
    end;

  var
    Form16: TForm16;

  implementation

  {$R *.dfm}

  procedure TForm16.SpeedButton1Click(Sender: TObject);
  begin
  form16.Close;
  end;

  procedure TForm16.DBGrid1KeyPress(Sender: TObject; var Key: Char);
  begin
  if key=#13 then
  begin
  key:=#0;
  query1.Insert;
  end;
  end;

  end.
```

Form 17 (Computer Sale Program [Country\City])

```
unit Unit17;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, Grids, DBGrids, ExtCtrls, DBCtrls, DB, DBTables;
```

```
type
  TForm17 = class(TForm)
    DBNavigator1: TDBNavigator;
    DBGrid1: TDBGrid;
    SpeedButton1: TSpeedButton;
    DataSource1: TDataSource;
    Query1: TQuery;
    procedure SpeedButton1Click(Sender: TObject);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form17: TForm17;

implementation

{$R *.dfm}

procedure TForm17.SpeedButton1Click(Sender: TObject);
begin
form17.close;

end;

procedure TForm17.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
if key=#13 then
  begin
    key:=#0;
    query1.Insert;
  end;
end;

end.
```

Form 18 ((Go To)\Operations)

```
unit Unit18;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons;
```

```
type
  TForm18 = class(TForm)
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    SpeedButton3: TSpeedButton;
    procedure SpeedButton3Click(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form18: TForm18;

implementation

uses Unit8;

{$R *.dfm}

procedure TForm18.SpeedButton3Click(Sender: TObject);
begin
form18.Close;
end;

procedure TForm18.SpeedButton1Click(Sender: TObject);
begin
form8.show;
end;

procedure TForm18.SpeedButton2Click(Sender: TObject);
begin
Winexec('Calc.exe',SW_Show);
end;

end.
```

Form 19 (Change Password)

```
unit Unit19;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, StdCtrls, ExtCtrls, DB, DBTables;
```

```
type
  TForm19 = class(TForm)
    Panel1: TPanel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Edit5: TEdit;
    Edit6: TEdit;
    Edit7: TEdit;
    Edit8: TEdit;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    DataSource1: TDataSource;
    Query1: TQuery;
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
     procedure FormKeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form19: TForm19;

implementation

{$R *.dfm}

procedure TForm19.SpeedButton1Click(Sender: TObject);
var
m:integer;
s:integer;
begin
query1.SQL.Clear;
query1.SQL.Text:='select * from pasword where KULLANICIADI
like'+#39+edit5.Text+'%'+#39;
query1.Open;
if query1.Fields[0].AsString=edit5.Text then
begin
  query1.SQL.Text:='select * from pasword where SIFRE
like'+#39+edit6.Text+'%'+#39;
  query1.Open;
  if query1.Fields[1].asstring=edit6.Text then
    begin
     query1.Edit;
     Query1.Fields[0].asstring:=Edit7.Text;
```

```
    Query1.Fields[1].asstring:=Edit8.Text;
    query1.Post;
    m:=application.MessageBox('Password Change Successful!!.','Information',0+48);
    edit5.Text:='';
    edit6.Text:='';
    edit7.Text:='';
    edit8.Text:='';
    edit5.SetFocus;
   end
   else
   begin
    MessageBeep(MB_ICONQUESTION);
    m:=application.MessageBox('Wrong User Id or Password!!','Mistake',0+48);
    edit5.Text:='';
    edit6.Text:='';
    edit7.Text:='';
    edit8.Text:='';
    edit5.SetFocus;
  end;

  end
  else
  begin
    MessageBeep(MB_ICONQUESTION);
    m:=application.MessageBox('Wrong User Id or Password!!','Mistake',0+48);
    edit5.Text:='';
    edit6.Text:='';
    edit7.Text:='';
    edit8.Text:='';
    edit5.SetFocus;
    end;
end;

procedure TForm19.SpeedButton2Click(Sender: TObject);
begin
form19.Close;
end;



procedure TForm19.FormKeyPress(Sender: TObject; var Key: Char);
begin
    if (Key = #13) then
    begin
      Key := #0;
      Perform(WM_NEXTDLGCTL, 0, 0);
    end;
end;

end.
```

Form 22 (List Of Stock)

```
unit Unit22;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, QuickRpt, QRCtrls, ExtCtrls, DB, DBTables, StdCtrls, Grids,
  DBGrids;

type
  TForm22 = class(TForm)
    Query1: TQuery;
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    Label1: TLabel;
    Button1: TButton;
    Button2: TButton;
    Label2: TLabel;
    Button3: TButton;
    RadioGroup1: TRadioGroup;
    ComboBox1: TComboBox;
    DataSource2: TDataSource;
    Query2: TQuery;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure RadioGroup1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure Button3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form22: TForm22;

implementation

uses Unit34;

{$R *.dfm}

procedure TForm22.Button1Click(Sender: TObject);
begin
FORM34.QuickRep1.Preview;
end;
```

```
procedure TForm22.Button2Click(Sender: TObject);
begin
FORM22.Close;
end;

procedure TForm22.RadioGroup1Click(Sender: TObject);
begin
if (radiogroup1.ItemIndex=0) then
begin
query1.Active:=false;
query1.SQL.Clear;
query1.SQL.Text:='select * from stokta order by serinum ASC'
end;

if (radiogroup1.ItemIndex=1) then
begin
query1.Active:=false;
query1.SQL.Clear;
query1.SQL.Text:='select * from stokta order by gtarihi ASC'
end;

if (radiogroup1.ItemIndex=2) then
begin
query1.Active:=false;
query1.SQL.Clear;
query1.SQL.Text:='select * from stokta order by sirano ASC'
end;
query1.Active:=true;

end;

procedure TForm22.FormActivate(Sender: TObject);
begin
combobox1.Items.Clear;
query2.First;
while (not query2.Eof) do begin
combobox1.Items.add(query2.Fields[1].AsString);
query2.Next;
end;

end;
function bul(a:string):boolean;
begin
bul:=false;
form22.Query1.SQL.Clear;
form22.Query1.SQL.Text:='select * from stokta where serinum
='+#39+(form22.combobox1.text)+#39;
form22.Query1.Open;
if not form22.Query1.IsEmpty then bul:=true;
end;
```

```
procedure TForm22.Button3Click(Sender: TObject);
begin
bul(combobox1.Text);
query1.active:=true;

end;

end.
```

Form 23 (Give Back Up)

```
unit Unit23;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, ExtCtrls, ComCtrls, Gauges;

type
  TForm23 = class(TForm)
    Panel1: TPanel;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    SpeedButton3: TSpeedButton;
    ProgressBar1: TProgressBar;
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure SpeedButton3Click(Sender: TObject);
    procedure FormKeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form23: TForm23;

implementation

{$R *.dfm}

procedure TForm23.SpeedButton1Click(Sender: TObject);
var i:integer;
begin
progressbar1.Visible:=true;
winexec('yedeklehdd.bat',0);
progressbar1.Min:=0;
```

```
progressbar1.Max:=5000;
for i:=0 to 5000 do
begin
progressbar1.Position:=i;
end;
progressbar1.Visible:=false;
application.MessageBox('Operation is OK!!!','Warning',32);
end;
procedure TForm23.SpeedButton2Click(Sender: TObject);
var i:integer;
begin
application.MessageBox('Please Insert Disk to A!!','Warning',32);
progressbar1.Visible:=true;
winexec('yedeklefdd.bat',0);
progressbar1.Min:=0;
progressbar1.Max:=315000;
for i:=0 to 315000 do
begin
progressbar1.Position:=i;
end;
progressbar1.Visible:=false;
application.MessageBox('Operation is OK!!','Warning',32);
end;

procedure TForm23.SpeedButton3Click(Sender: TObject);
begin
form23.Close;
end;

procedure TForm23.FormKeyPress(Sender: TObject; var Key: Char);
begin
if key=#27 then
form23.Close;
end;

end.
```

Form 24 (About)

```
unit Unit24;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls;

type
  TForm24 = class(TForm)
    Memo1: TMemo;
```

```
    Button1: TButton;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form24: TForm24;

implementation

uses Unit1;

{$R *.dfm}

procedure TForm24.Button1Click(Sender: TObject);
begin
form24.Close;
form1.enabled:=true;
end;

end.
```

Form 25 (Register Or Delete Staff)

```
unit Unit25;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, DBTables, Grids, DBGrids, StdCtrls, Buttons;

type
  TForm25 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    ComboBox1: TComboBox;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    Query1: TQuery;
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
```

```
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form25: TForm25;

implementation

uses Unit1;

{$R *.dfm}
function bul(a:string):boolean;
begin
bul:=false;
form25.Query1.First;
while not form25.Query1.eof do
if (a=form25.Query1.Fields[0].asstring)then
begin
bul:=true;
exit;
end
else
form25.Query1.Next;
end;

procedure TForm25.BitBtn1Click(Sender: TObject);
begin
Bul(edit1.Text);
if Query1.Fields[0].asstring=edit1.Text then
application.MessageBox('Staff was Registered','Warning',32)
else
begin
query1.Insert;
query1.Fields[0].AsString:=edit1.Text;
query1.Fields[1].AsString:=combobox1.Text;
query1.Post;
query1.Active:=false;
query1.Active:=true;
edit1.Text:='';
edit1.SetFocus;
end;
end;

procedure TForm25.BitBtn3Click(Sender: TObject);
```

```
begin
query1.edit;
query1.Fields[0].AsString:=edit1.Text;
query1.Fields[1].AsString:=combobox1.Text;
query1.Post;
query1.Active:=false;
query1.Active:=true;
edit1.Text:='';
edit1.SetFocus;

end;

procedure TForm25.BitBtn2Click(Sender: TObject);
begin
bul(edit1.Text);
if query1.Fields[0].AsString=edit1.Text then
begin
query1.Delete;
query1.Active:=false;
query1.Active:=true;
edit1.Text:='';
edit1.SetFocus;
end;
end;

procedure TForm25.BitBtn4Click(Sender: TObject);
begin
form25.Close;
form1.enabled:=true;
end;

procedure TForm25.DBGrid1DblClick(Sender: TObject);
begin
edit1.Text:=query1.Fields[0].AsString;
combobox1.Text:=query1.Fields[1].AsString;
end;

end.
```

Form 26 (Outlet Form Of Cash Money)

```
unit Unit26;

interface

uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, DB, DBTables, StdCtrls, Grids, DBGrids, Buttons;

type
```

```
  TForm26 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    ComboBox1: TComboBox;
    DataSource1: TDataSource;
    Query1: TQuery;
    Query2: TQuery;
    Query3: TQuery;
    DataSource2: TDataSource;
    DataSource3: TDataSource;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    Label6: TLabel;
    ComboBox2: TComboBox;
    DBGrid1: TDBGrid;
    Label7: TLabel;
    procedure FormShow(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form26: TForm26;

implementation

uses Unit12, Unit1;

{$R *.dfm}

procedure TForm26.FormShow(Sender: TObject);
begin
  form26.query2.First;
combobox1.Clear;
```

```
while not form26.query2.Eof do
begin
combobox1.Items.Add(form26.query2.Fields[0].asstring);
form26.query2.Next;
end;

end;
function seri(b:string):boolean;
begin
seri:=false;
form26.Query3.First;
while not form26.Query3.eof do
if (b=form26.Query3.Fields[0].asstring)then
begin
seri:=true;
exit;
end
else
form26.Query3.Next;
end;

procedure TForm26.BitBtn1Click(Sender: TObject);
var dto:integer;
begin
query1.Insert;
query1.Fields[0].AsString:=edit1.Text;
query1.Fields[1].AsString:=edit2.Text;
query1.Fields[2].AsString:=edit3.Text;
query1.Fields[3].AsString:=edit4.Text;
query1.Fields[4].AsString:=combobox1.Text;
query1.Fields[5].AsString:=combobox2.Text;
query1.Post;
query1.Active:=false;
query1.Active:=true;
seri(Combobox2.Text);
if (query3.Fields[0].asstring=combobox2.Text) then
begin
dto:=strtoint(query3.Fields[1].AsString)- strtoint(edit2.Text);
query3.Edit;
query3.Fields[1].AsString:=inttostr(dto);
query3.post;
end;
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
edit1.Text:=datetostr(date());
end;

procedure TForm26.BitBtn4Click(Sender: TObject);
begin
```

```
form1.enabled:=true;
form26.Close;
end;

procedure TForm26.BitBtn2Click(Sender: TObject);
begin
if not(query1.Eof) then
begin
query1.Delete;
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
edit1.Text:=datetostr(date());
end;

end;

procedure TForm26.BitBtn3Click(Sender: TObject);
var dto:integer;
begin
query1.edit;
query1.Fields[0].AsString:=edit1.Text;
query1.Fields[1].AsString:=edit2.Text;
query1.Fields[2].AsString:=edit3.Text;
query1.Fields[3].AsString:=edit4.Text;
query1.Fields[4].AsString:=combobox1.Text;
query1.Fields[5].AsString:=combobox2.Text;
query1.Post;
query1.Active:=false;
query1.Active:=true;
seri(Combobox2.Text);
if (query3.Fields[0].asstring=combobox2.Text) then
begin
dto:=strtoint(query3.Fields[1].AsString)- strtoint(edit2.Text);
query3.Edit;
query3.Fields[1].AsString:=inttostr(dto);
query3.post;
end;
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
edit1.Text:=datetostr(date());
end;
procedure TForm26.FormActivate(Sender: TObject);
begin
edit1.Text:=datetostr(date);
end;

end.
```

Form 27 (Summery Of Outlet Cash Money)

```
 unit Unit27;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, DB, DBTables, StdCtrls, Buttons;

type
  TForm27 = class(TForm)
    DataSource1: TDataSource;
    Query1: TQuery;
    DBGrid1: TDBGrid;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    BitBtn1: TBitBtn;
    procedure BitBtn1Click(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
    procedure FormActivate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form27: TForm27;

implementation

uses Unit1;

{$R *.dfm}
function bul1(a:string;b:string):boolean;
begin
bul1:=false;
form27.Query1.First;
while not form27.Query1.eof do
if (a=form27.Query1.Fields[0].asstring)and (b=form27.Query1.Fields[3].asstring)then
begin
bul1:=true;
exit;
```

```
end
else
form27.Query1.Next;
end;
function bul2(c:string;d:string):boolean;
begin
bul2:=false;
form27.Query1.First;
while not form27.Query1.eof do
if (c=form27.Query1.Fields[5].asstring)and (d=form27.Query1.Fields[1].asstring)then
begin
bul2:=true;
exit;
end
else
form27.Query1.Next;
end;

procedure TForm27.BitBtn1Click(Sender: TObject);
begin
form27.close;
end;

procedure TForm27.DBGrid1DblClick(Sender: TObject);
var
a,b,c,d:string;
begin
  a:=dbgrid1.Fields[0].AsString; //tarih
  b:=dbgrid1.Fields[3].AsString; //kime verildiği
  c:=dbgrid1.Fields[5].AsString; //doviz
  d:=dbgrid1.Fields[1].AsString; //miktar
 bul1(a,b);
 if(query1.Fields[0].AsString=a)and (query1.Fields[3].AsString=b) then
 begin
 bul2(c,d);
 if(query1.Fields[5].AsString=c)and (query1.Fields[1].AsString=d) then
  begin
 edit1.Text:=query1.Fields[0].AsString;
 edit2.Text:=query1.Fields[3].AsString;
 edit3.Text:=query1.Fields[2].AsString;
 edit4.Text:=query1.Fields[1].AsString;
  end;
 end;
end;

procedure TForm27.FormActivate(Sender: TObject);
begin
query1.Refresh;
end;
```

```
end.
```

Form 29 (Entry Form Of Stock Device)

```
unit Unit29;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, DB, DBTables, Grids, DBGrids, Buttons;

type
  TForm29 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit4: TEdit;
    Edit5: TEdit;
    Edit6: TEdit;
    Edit7: TEdit;
    Edit8: TEdit;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    DBGrid1: TDBGrid;
    Query1: TQuery;
    DataSource1: TDataSource;
    DataSource2: TDataSource;
    Query2: TQuery;
    ComboBox1: TComboBox;
    procedure FormActivate(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
```

```
  end;

var
  Form29: TForm29;

implementation

{$R *.dfm}
function bul(a:string):boolean;
begin
bul:=false;
form29.Query1.First;
while not form29.Query1.eof do
if (a=form29.Query1.Fields[1].asstring)then
begin
bul:=true;
exit;
end
else
form29.Query1.Next;
end;

procedure TForm29.FormActivate(Sender: TObject);
begin
combobox1.Items.Clear;
query2.First;
while (not query2.Eof) do begin
combobox1.Items.Add(Query2.fields[0].asstring);
query2.next;
end;
edit6.Text:=datetostr(date());
end;

procedure TForm29.BitBtn4Click(Sender: TObject);
begin
form29.Close;
end;

procedure TForm29.BitBtn1Click(Sender: TObject);
begin
query1.Insert;
query1.Fields[0].asstring:=edit1.Text;
query1.Fields[1].asstring:=edit2.Text;
query1.Fields[2].asstring:=combobox1.Text;
query1.Fields[3].asstring:=edit4.Text;
query1.Fields[4].asstring:=edit5.Text;
query1.Fields[5].asstring:=edit6.Text;
query1.Fields[6].asstring:=edit7.Text;
query1.Fields[7].asstring:=edit8.Text;
query1.Post;
```

```
edit1.Text:='';
edit2.Text:='';
edit8.Text:='';
edit4.Text:='';
edit5.Text:='';
edit6.Text:='';
edit7.Text:='';
combobox1.Text:='';
edit1.setfocus;
edit6.Text:=datetostr(date());
query1.Refresh;
edit6.Text:=datetostr(date());
end;

procedure TForm29.BitBtn2Click(Sender: TObject);
var

a,b:integer;
begin

query1.edit;
query1.Fields[0].asstring:=edit1.Text;
query1.Fields[1].asstring:=edit2.Text;
query1.Fields[2].asstring:=combobox1.Text;
query1.Fields[3].asstring:=edit4.Text;
query1.Fields[4].asstring:=edit5.Text;
query1.Fields[5].asstring:=edit6.Text;
query1.Fields[6].asstring:=edit7.Text;
query1.Fields[7].asstring:=edit8.Text;
query1.Post;
edit1.Text:='';
edit2.Text:='';
edit8.Text:='';
edit4.Text:='';
edit5.Text:='';
edit6.Text:='';
edit7.Text:='';
combobox1.Text:='';
edit1.setfocus;
edit6.Text:=datetostr(date());
query1.Refresh;
end;

procedure TForm29.BitBtn3Click(Sender: TObject);
begin
bul(edit2.text);
if  (query1.fields[1].AsString=edit2.text) then begin
query1.Delete;
edit1.Text:='';
edit2.Text:='';
```

```
edit8.Text:='';
edit4.Text:='';
edit5.Text:='';
edit6.Text:='';
edit7.Text:='';
combobox1.Text:='';
edit1.setfocus;
query1.Refresh;
end;
edit6.Text:=datetostr(date());
end;

procedure TForm29.DBGrid1DblClick(Sender: TObject);
begin
bul(dbgrid1.Fields[1].AsString);
if (query1.fields[1].AsString=dbgrid1.Fields[1].AsString ) then begin
edit1.Text:=query1.Fields[0].asstring;
edit2.Text:=query1.Fields[1].asstring;
combobox1.Text:=query1.Fields[2].asstring;
edit4.Text:=query1.Fields[3].asstring;
edit5.Text:=query1.Fields[4].asstring;
edit6.Text:=query1.Fields[5].asstring;
edit7.Text:=query1.Fields[6].asstring;
edit8.Text:=query1.Fields[7].asstring;
end;
end;
end.
```

Form 30 (Entry Form Of Deffective Device)

```
unit Unit30;
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, DB, DBTables, Mask, Grids, DBGrids, Buttons;

type
  TForm30 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
```

```
      Label11: TLabel;
      Label12: TLabel;
      Edit1: TEdit;
      Edit2: TEdit;
      Edit3: TEdit;
      ComboBox1: TComboBox;
      Edit4: TEdit;
      Edit5: TEdit;
      Edit6: TEdit;
      Edit8: TEdit;
      MaskEdit1: TMaskEdit;
      Memo1: TMemo;
      Query1: TQuery;
      DataSource1: TDataSource;
      DataSource2: TDataSource;
      Query2: TQuery;
      DBGrid1: TDBGrid;
      Label13: TLabel;
      ComboBox3: TComboBox;
      BitBtn1: TBitBtn;
      BitBtn2: TBitBtn;
      BitBtn3: TBitBtn;
      BitBtn4: TBitBtn;
      MaskEdit2: TMaskEdit;
      ComboBox2: TComboBox;
      Button1: TButton;
      procedure FormActivate(Sender: TObject);
      procedure BitBtn1Click(Sender: TObject);
      procedure BitBtn2Click(Sender: TObject);
      procedure BitBtn3Click(Sender: TObject);
      procedure DBGrid1DblClick(Sender: TObject);
      procedure BitBtn4Click(Sender: TObject);
      procedure Button1Click(Sender: TObject);
    private
      { Private declarations }
    public
      { Public declarations }
    end;

  var
    Form30: TForm30;

  implementation

  {$R *.dfm}
  function bul(a:string):boolean;
  begin
  bul:=false;
  form30.Query2.First;
  while not form30.Query2.eof do
```

```
if (a=form30.Query2.Fields[1].asstring)then
begin
bul:=true;
exit;
end
else
form30.Query2.Next;
end;

procedure TForm30.FormActivate(Sender: TObject);
begin
combobox2.Items.Clear;
query1.First;
while (not query1.Eof) do begin
combobox2.Items.Add(Query1.fields[0].asstring);
query1.next;
 end;
end;

procedure TForm30.BitBtn1Click(Sender: TObject);
begin
    query2.Insert;
    query2.Fields[0].AsString:=edit1.Text;
    query2.Fields[1].AsString:=edit2.Text;
query2.Fields[2].AsString:=edit3.Text;
query2.Fields[3].AsString:=combobox1.Text;
query2.Fields[4].AsString:=combobox2.Text;
query2.Fields[5].AsString:=edit4.Text;
query2.Fields[6].AsString:=edit5.Text;
query2.Fields[7].AsString:=edit6.Text;
query2.Fields[8].AsString:=maskedit2.Text;
query2.Fields[9].AsString:=edit8.Text;
query2.Fields[10].AsString:=maskedit1.Text;
query2.Fields[11].AsString:=memo1.Text;
query2.Fields[12].AsString:=combobox3.Text;
query2.Post;
edit1.Text:='';
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
edit5.Text:='';
edit6.Text:='';
edit8.Text:='';
maskedit1.Text:='';
maskedit2.Text:='';
memo1.Text:='';
edit1.setfocus;
query2.Refresh;
```

```
end;

procedure TForm30.BitBtn2Click(Sender: TObject);
begin
    query2.edit;
    query2.Fields[0].AsString:=edit1.Text;
    query2.Fields[1].AsString:=edit2.Text;
query2.Fields[2].AsString:=edit3.Text;
query2.Fields[3].AsString:=combobox1.Text;
query2.Fields[4].AsString:=combobox2.Text;
query2.Fields[5].AsString:=edit4.Text;
query2.Fields[6].AsString:=edit5.Text;
query2.Fields[7].AsString:=edit6.Text;
query2.Fields[8].AsString:=maskedit2.Text;
query2.Fields[9].AsString:=edit8.Text;
query2.Fields[10].AsString:=maskedit1.Text;
query2.Fields[11].AsString:=memo1.Text;
query2.Fields[12].AsString:=combobox3.Text;
query2.Post;
edit1.Text:='';
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
edit5.Text:='';
edit6.Text:='';
edit8.Text:='';
maskedit1.Text:='';
maskedit2.Text:='';
memo1.Text:='';
edit1.setfocus;
query2.Refresh;


end;

procedure TForm30.BitBtn3Click(Sender: TObject);
begin
bul(edit2.text);
if (query1.Fields[1].AsString=edit2.text ) then
begin
query2.Delete;
edit1.Text:='';
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
```

```
edit5.Text:='';
edit6.Text:='';
edit8.Text:='';
maskedit1.Text:='';
maskedit2.Text:='';
memo1.Text:='';
edit1.setfocus;
query2.Refresh;


end;

end;

procedure TForm30.DBGrid1DblClick(Sender: TObject);
var
a:string;
begin
a:=dbgrid1.Fields[1].AsString;
bul(a);
if (query2.Fields[1].AsString=a) then
begin
edit1.Text:=query2.Fields[0].AsString;
edit2.Text:=query2.Fields[1].AsString;
edit3.Text:=query2.Fields[2].AsString;
combobox1.Text:=query2.Fields[3].AsString;
combobox2.Text:=query2.Fields[4].AsString;
edit4.Text:=query2.Fields[5].AsString;
edit5.Text:=query2.Fields[6].AsString;
edit6.Text:=query2.Fields[7].AsString;
maskedit2.Text:=query2.Fields[8].AsString;
edit8.Text:=query2.Fields[9].AsString;
maskedit1.Text:=query2.Fields[10].AsString;
memo1.Text:=query2.Fields[11].AsString;
combobox3.Text:=query2.Fields[12].AsString;
end;

end;

procedure TForm30.BitBtn4Click(Sender: TObject);
begin
form30.close;
end;

procedure TForm30.Button1Click(Sender: TObject);
begin
edit1.Text:='';
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
```

```
edit5.Text:='';
edit6.Text:='';
edit8.Text:='';
maskedit1.Text:='';
maskedit2.Text:='';
memo1.Text:='';
combobox1.Text:='';
combobox2.Text:='';
combobox3.Text:='';

edit1.setfocus;
end;
end.
```

Form 31 (Result Form Of Deffective Device)

```
unit Unit31;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, Grids, DBGrids, DB, DBTables;

type
  TForm31 = class(TForm)
    Label1: TLabel;
    DBGrid1: TDBGrid;
    ComboBox1: TComboBox;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    DataSource1: TDataSource;
    Query1: TQuery;
    ComboBox2: TComboBox;
    Query2: TQuery;
    DataSource2: TDataSource;
    BitBtn3: TBitBtn;
    Label2: TLabel;
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form31: TForm31;
```

```
implementation

{$R *.dfm}
function bul(a:string):boolean;
begin
bul:=false;
form31.Query1.SQL.Clear;
form31.Query1.SQL.Text:='select * from arizadurumu where serinum
='+#39+(form31.combobox2.text)+#39;
form31.Query1.Open;
if not form31.Query1.IsEmpty then bul:=true;
end;

procedure TForm31.BitBtn1Click(Sender: TObject);
begin
bul(combobox1.Text);
query1.Active:=true;
end;

procedure TForm31.BitBtn2Click(Sender: TObject);
begin
query1.Edit;
query1.Fields[12].AsString:=combobox1.Text;
query1.Post;
query1.Refresh;
end;

procedure TForm31.FormActivate(Sender: TObject);
begin
combobox2.Items.Clear;
query2.First;
while (not query2.Eof) do begin
combobox2.Items.Add(query2.Fields[1].AsString);
query2.Next;
end;
query1.Active:=false;

end;

procedure TForm31.BitBtn3Click(Sender: TObject);
begin
FORM31.CLOSE;
end;
end.
```

Form 32 (List Of Deffective Device)

```
unit Unit32;
```

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, DBTables, Grids, DBGrids, StdCtrls, ExtCtrls, Buttons;

type
  TForm32 = class(TForm)
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    Query1: TQuery;
    RadioGroup1: TRadioGroup;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    procedure RadioGroup1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form32: TForm32;

implementation

uses Unit33;

{$R *.dfm}

procedure TForm32.RadioGroup1Click(Sender: TObject);
begin
if (radiogroup1.ItemIndex=1) then
begin
query1.Active:=false;
query1.SQL.Clear;
query1.SQL.Text:='select * from arizadurumu where durum='+#39+('Not Check')+#39;
query1.Active:=true;

end;

if (radiogroup1.ItemIndex=0) then
begin
query1.Active:=false;
query1.SQL.Clear;
query1.SQL.Text:='select * from arizadurumu where durum='+#39+('Checked')+#39;
query1.Active:=true;
```

```
end;

end;

procedure TForm32.FormActivate(Sender: TObject);
begin
query1.Active:=false;

end;

procedure TForm32.BitBtn1Click(Sender: TObject);
begin
form32.close;
end;
function bul(a:string):boolean;
begin
bul:=false;
form33.Query1.SQL.Clear;
form33.Query1.SQL.Text:='select * from arizadurumu where durum
='+#39+('Checked')+#39;
form33.Query1.Open;
if not form33.Query1.IsEmpty then bul:=true;
end;
function bul1(a:string):boolean;
begin
bul1:=false;
form33.Query1.SQL.Clear;
form33.Query1.SQL.Text:='select * from arizadurumu where durum ='+#39+('Not
Check')+#39;
form33.Query1.Open;
if not form33.Query1.IsEmpty then bul1:=true;
end;

procedure TForm32.BitBtn2Click(Sender: TObject);
begin
if radiogroup1.ItemIndex=0 then
begin
bul('Checked');
form33.QRLabel1.Caption:='LIST OF CHECKED SYSTEM';
form33.QuickRep1.Preview;
end;
if radiogroup1.ItemIndex=1 then
begin
bul1('Not Check');
form33.QRLabel1.Caption:='LIST OF NOT CHECK SYSTEM';
form33.QuickRep1.Preview;
end;

end;
```

```
end.
```

Form 33 (Print Preview Form)

```
unit Unit33;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, DBTables, QRCtrls, QuickRpt, ExtCtrls;

type
  TForm33 = class(TForm)
    QuickRep1: TQuickRep;
    QRBand2: TQRBand;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    QRLabel9: TQRLabel;
    QRBand3: TQRBand;
    QRDBText1: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    QRDBText5: TQRDBText;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText8: TQRDBText;
    Query1: TQuery;
    DataSource1: TDataSource;
    QRLabel1: TQRLabel;
    QRLabel10: TQRLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form33: TForm33;

implementation

{$R *.dfm}
```

```
end.
```

Form 34 (Print Preview)

```
unit Unit34;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, DBTables, QRCtrls, QuickRpt, ExtCtrls;

type
  TForm34 = class(TForm)
    QuickRep1: TQuickRep;
    QRBand2: TQRBand;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    QRBand3: TQRBand;
    QRDBText1: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    QRDBText5: TQRDBText;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    Query1: TQuery;
    DataSource1: TDataSource;
    QRLabel9: TQRLabel;
    QRLabel1: TQRLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form34: TForm34;

implementation

{$R *.dfm}

end
```

APPENDIX B

DATABASE TABLES

The "ARIZA" table contains 2 fields:

- KOD
- ACIKLAMA

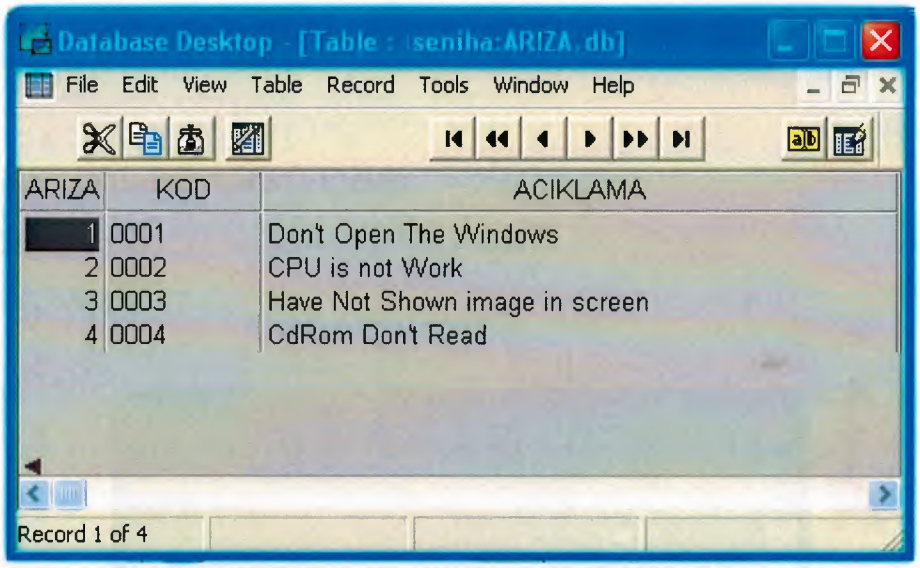


Figure 4.1.ARIZA Table

The "arizadurumu" table contains 13 fields;

- Belgeno
- Garanti
- Cinsi
- Marka
- Model
- Sikayet
- Agtarihi
- Kimeait
- Telefonu
- Adresi
- Durum

arizadurumu	Sirano	Serinum	Belgeno	Garanti	Cinsi	Marka	Model	Sikayet	Agtarihi	Kimeait
1	03	3455	444	Not Warran	Kasa	asus	965	Dont Open	24.12.2006	yusuf alak
2	02	456	6666	Not Warran	Kasa	datron	tx3c	Dont Open		

Figure 4.2.arizadurumu Table

The "CINS" table contains 2 Fields;

- KOD
- ACIKLAMA

Table : :seniha:CINS.db

CINS	KOD	ACIKLAMA
1	CPU	
2	Cdrom	
3	Fax Modem	
4	Harddisk	

Figure 4.3.CINS Table

The "GIRIS" table contains 40 fields;

- FISNO
- CIHAZBILGILERI
- FATURAKESILDI
- BELGENO
- BASVURU TARIHI
- SERVISTARIHI
- TESLIMTARIHI
- SERINO
- MARKA
- CINSI

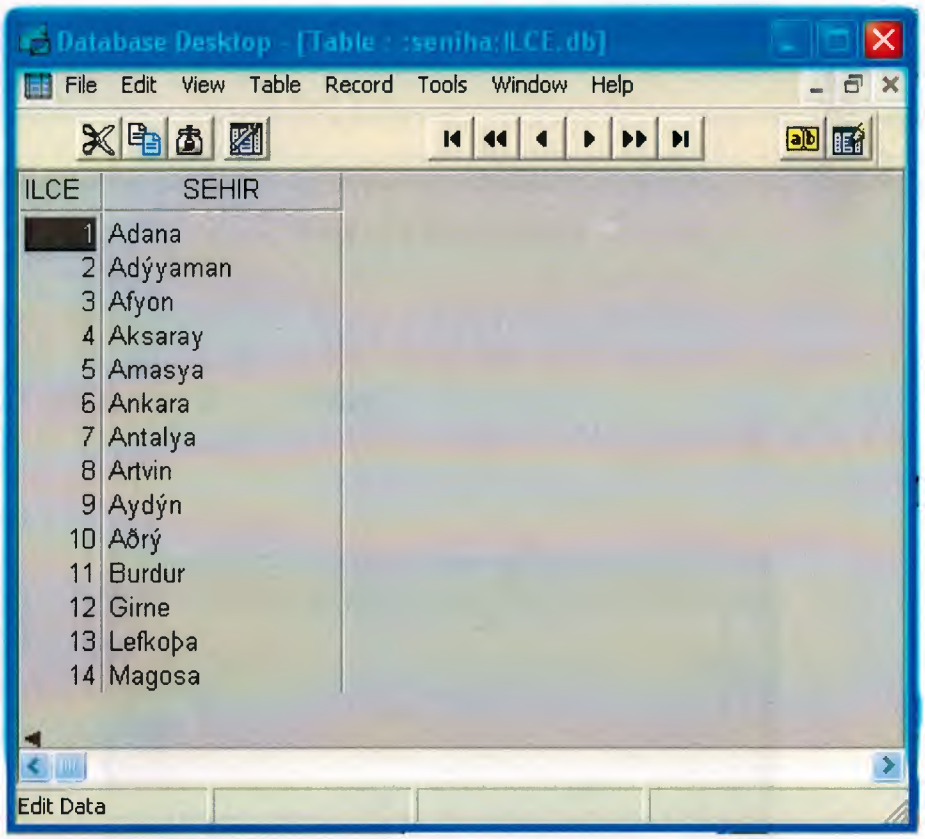
- MODELI
- BAYI
- FATURANO
- FATURATARIHI
- CIHAZDURUMU
- ARIZAKODU
- ACIKLAMA
- TEKNISYENNOTU
- GARANTI
- ISIM
- SOYAD
- EVTEL
- ISTEL
- ADRES
- SEHIR
- VDAIRESI
- VNO
- IRSTARIHI
- IRS NO
- FATTARIHI
- AKSESUAR
- SIKAYET
- PARCATOP
- ISCILIK
- ARATOP
- KDV
- GTOP
- SERVISELEMANI
- KASAYAACIKLAMA
- TAHSILEDILEN

GIRIS	FISNO	CIHAZBILGILERI	FATURAKESILDI	BELGENO	BASVURUTARIHI	SERVISTARIHI	TESLIMTARIHI	SERINO	MARKA	CINSI
1	35			0044	12.01.2007	12.01.2007	12.01.2007	3444444	Amd	CPU
2	34			0033	12.01.2007	12.01.2007	12.01.2007	235679	Philips	Monitor
3	33			0022	10.01.2007	10.01.2007	11.01.2007	23456	Aopen	Fax Modem
4	32			0022	10.01.2007	11.01.2007	12.01.2007	00203	Aopen	Fax Modem

Figure 4.4.GIRIS Table

The ILCE table contains 2 fields;

- ILCE
- SEHIR



ILCE	SEHIR
1	Adana
2	Adýyaman
3	Afyon
4	Aksaray
5	Amasya
6	Ankara
7	Antalya
8	Artvin
9	Aydýn
10	Aðrý
11	Burdur
12	Girne
13	Lefkoþa
14	Magosa

Figure 4.5 ILCE Table

The "kasa" table contains 5 fields;

- Tarih
- Kasa Sorumlusu

- Miktar
- DovizCinsi
- Geldigi yer

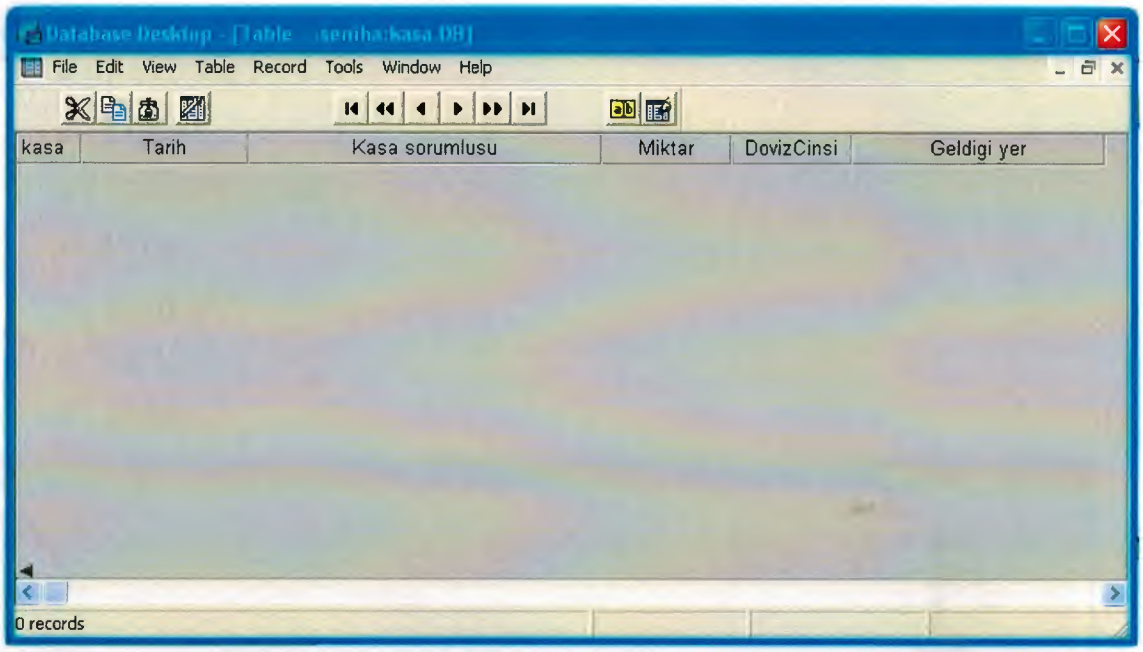


Figure 4.6 kasa Table

The "kasa1" table contains 2 fields;

- Cins
- Miktar

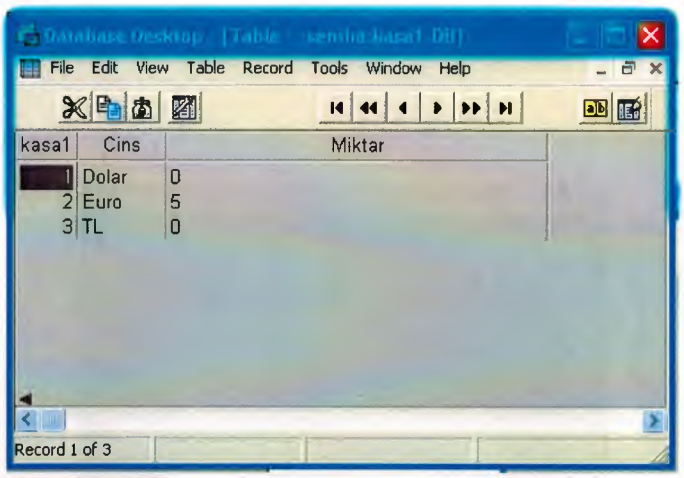


Figure 4.6 kasa1 Table

The “KULLANILAN” table contains 13 fields;

- MIKTARI
- FIYATI
- KDV
- ACIKLAMA
- PARCANO
- TUTAR
- STOKDURUMU
- PTOP
- ISCILIK
- ARATOP
- GTOP

KULLANILAN	FISNO	PARCAADI	MIKTARI	FIYATI	KDV	ACIKLAMA	PARCANO	TUTAR	STOKDURUMU	PTOP	ISCILIK	ARATOP
2	35	CPU	1,00	100	15,00	change	0,00	115	Not Have Stock	115	20	135
3	34		1,00	20	0,00	Workers pay	0,00	20	Not Have Stock		20	
4	34	monitor	1,00	50	15,00	add	0,00	57,5	Not Have Stock	57,5	20	77,5
5	33		1,00	20	0,00	Workers pay	0,00	20	Not Have Stock		20	
6	33	fax-modem	1,00	15	10,00	have Deffect and change t	0,00	16,5	Not Have Stock	16,5	20	36,5
7	32		1,00	20	0,00	Workers pay	0,00	20	Not Have Stock		20	
8	32	ekran kartý	1,00	45	15,00		0,00	51,75	Not Have Stock	51,75	20	71,75
9	31		1,00	20	0,00	Workers pay	0,00	20	Not Have Stock		20	
10	31	cd rom	1,00	10	15,00	add	0,00	11,5	Not Have Stock	11,5	20	31,5
11	29		1,00	20	0,00	Workers pay	0,00	20	Not Have Stock		20	
12	29	monitor	1,00	50	15,00	takýldý	0,00	57,5	Not Have Stock	57,5	20	77,5
13	27		1,00	20	0,00	Workers pay	0,00	20	Not Have Stock		20	
14	27	monitor	1,00	23	15,00	eklendi	0,00	26,45	Not Have Stock	26,45	20	46,45

Record 4 of 14

Figure 4.7. KULLANILAN Table

The “KUR” table contains 2 fields;

- DOLAR
- EURO
- TL

KUR	DOLAR	EURO	TL
1	1,234	7,654	
2	1,234	2,345	
3	1,234	2,345	
4	1,500	1,900	
5	1,234	1,987	

Record 1 of 5

Figure 4.8.KUR Table

The "MARKA" table contains 2 fields;

- KOD
- ACIKLAMA

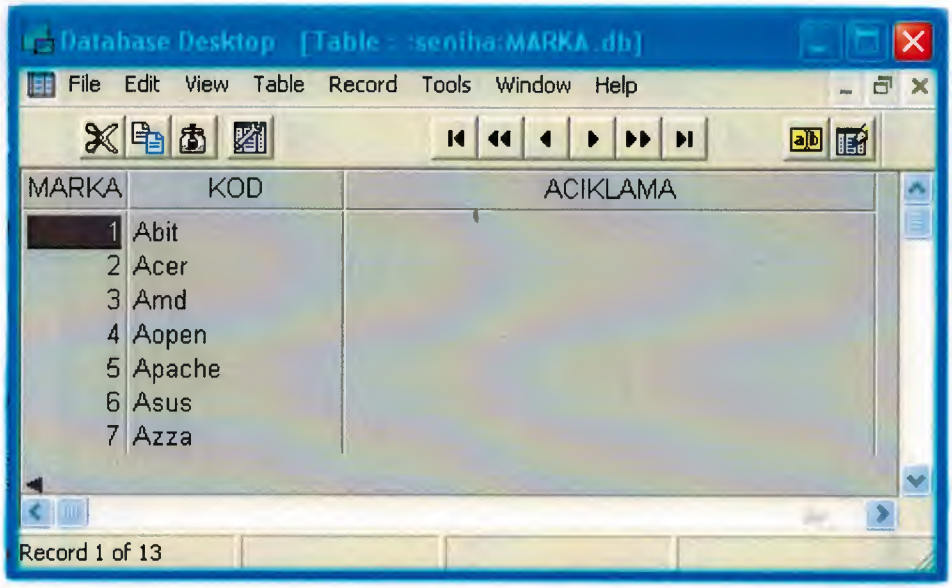


Figure 4.9 MARKA Table

The "MODEL" table contains 2 fields;

- KOD
- ACIKLAMA

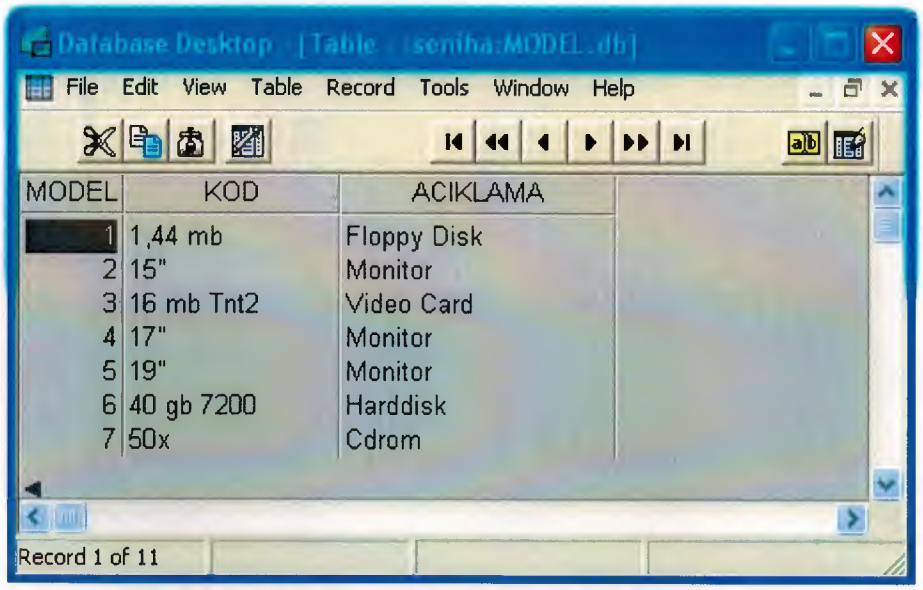


Figure 4.10 MODEL Table

The "Pasword" table contains 2 fields;

- KULLANIADI
- SIFRE

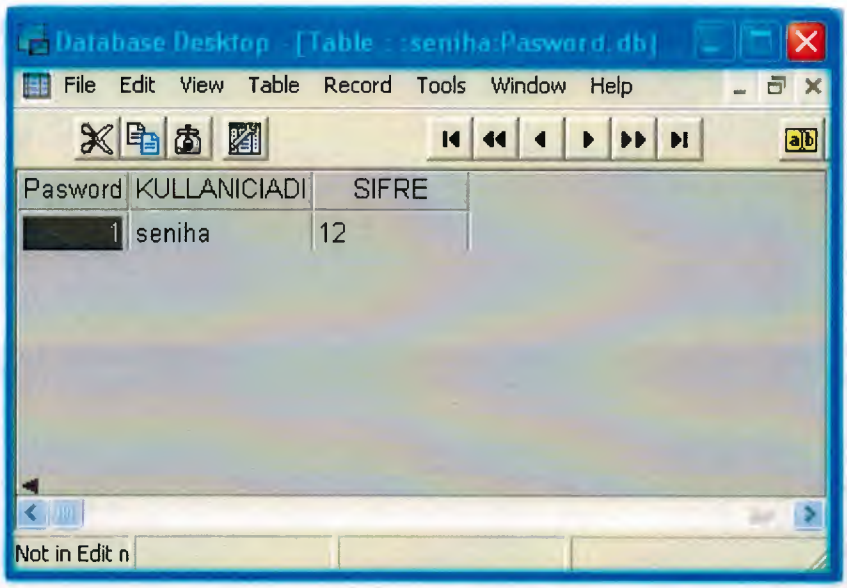


Figure 4.11 Pasword Table

The "Per" table contains 2 fields;

- AdiSoyadi
- Unvani

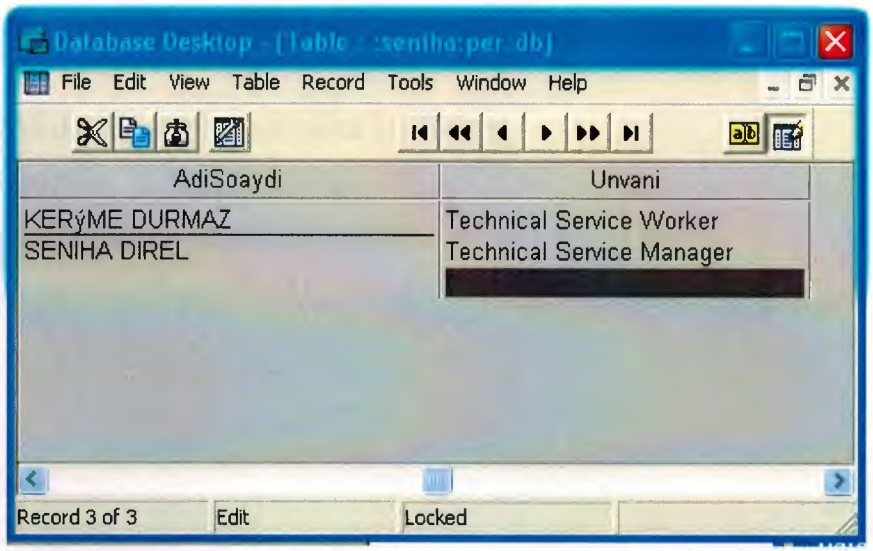


Figure 4.12 Per Table

Table " Stokta" contains 8 fields;

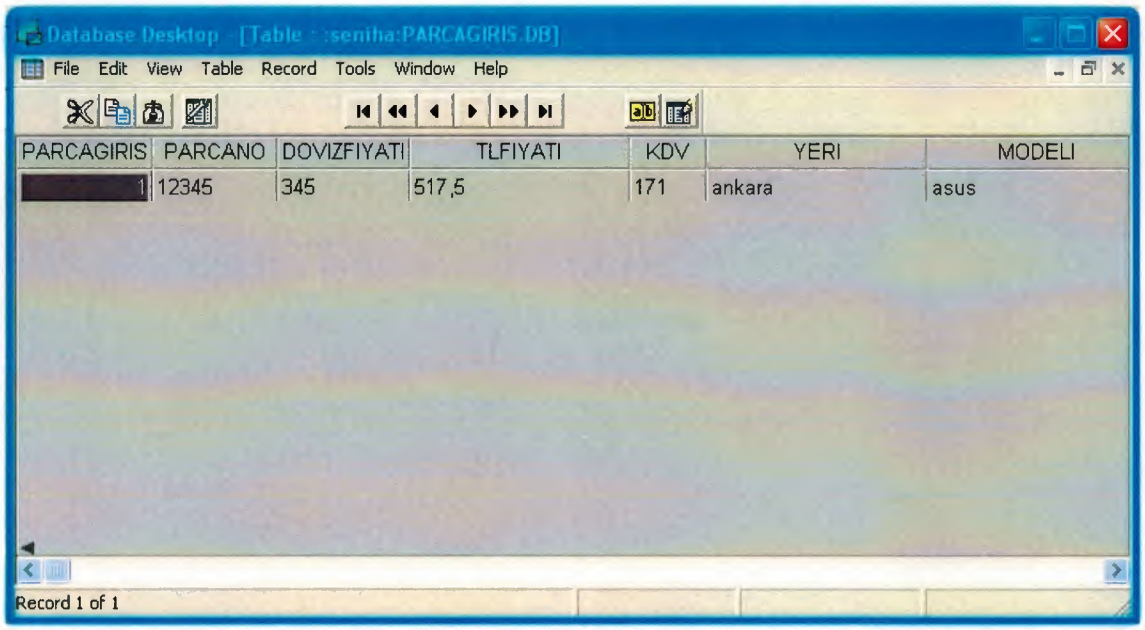
- Sirano
- Serinum
- Cinsi
- Marka
- Model
- Gtarihi
- Kimdengeldigi
- Belgeno

stokta	Sirano	Serinum	Cinsi	Marka	Gtarihi	Model	Kimdengeldigi	Belgeno
1	4	678	Video Card	gforce	05.01.2007	4x659	arena	666
2	5	456	Harddisk	creative	05.01.2007	4yt6	tekbim	789
3	3	013	Harddisk	creative	03.01.2007	6xy	yerli	03
4	2	012	Main Board	asus	10.05.2003	p4t-e	arena	02
5	1	011	Cdrom	lg	10.05.2003	52x	arena	01

Figure 4.13 Stokta Table

The "PARCAGIRIS" table contains 14 fields;

- PARCANO
- DOVIZFIYATI
- TLFIYATI
- KDV
- YERI
- MODELI
- KARSILIK



PARCAGIRIS	PARCANO	DOVIZFIYATI	TLFIYATI	KDV	YERI	MODELI
1	12345	345	517,5	171	ankara	asus

Figure 4.14 PARCAGIRIS Table