NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

VETERINERIAN APPLICATION PROGRAM WITH DELPHI

Graduation Project
COM400

Student: Gerçek SEZGİN (20021500)

Supervisor: Mr. Elburus IMANOV

Lefkoşa-2007

TABLE OF CONTENT

ACKNOWLEDGMENT

To begin with, I am proud and happy to complete the task which I had given with blessing of God and also I am grateful to all the people in my life who have, supported me, advised me. Taught me and who have always encouraged me to follow my dreams and ambitions.

I wish to thank my supervisor, ***Elburus Imanov****, for intellectual support, encouragement, and enthusiasm, which made this project possible, and his patience for correcting both my stylistic and scientific errors.*

And thank my dearest parents who encouraged me to continue beyond my undergraduate studies, to my father, ***Metin Sezgin*** *who proceeded before me and to my mother,* ***Perihan Sezgin*** *who encouraged me along the way.*

Thankful my life is very important for past and future who my fiancée, ***Begüm Vardar****.*

To all my friends, especially ***Hakan Kılıç, Ahmet Kayabaş*** *for sharing wonderful moments, advice, and for making me feel at home.*

As last, I thank God for giving me stamina and courage to achieve my objectives.

GERÇEK SEZGİN

ABSTRACT

In the universe not only person life is important.In the same time other entity lives with us.We are not alone on the earth.Animals share life with us.Ilnesses are not only for person.In the same time whole alive interested with illnesses.

How Doctor is important for us like Veterinerian is important for animals.Todays Doctors use application program.Because of to keep knowledge of patient, to facility diagnosis of illness, to reach background of patient efficiently and easly.

Well Veterinerian application program is important like the program that is used person health.Also much more important then others.Because animal can not keep the illnesses knowledge.And also papers of the animal can lost.

This project has as its aim to develop software, processing information about activities of a veterinerian application software. Software developed in this project like not only for animal.At the same time for staff and for owner of the animal.All records keep in the other Database program.It acts easly and fast access.Veterinerian can keep all records in the program as concentment.

INTRODUCTION

In the near future, the technology is developed a lot and started to use by anyone in the world no matter who he/she is. Because of the technology is entered to every platform of our life person needed to combine both software and hardware. Without software the machines are nothing. They need software to operate.

The automation is also became a part of our lives. The people operate with automation systems in everywhere. This Automation is used to keep the information about the receiving, coming ang going documents. Begsoft is Lifelong Program System.

Begsoft project which is my project.In this software veterinerian can keep animal knowledge, cracked background knowledge of the animal, owner of the animal knowledge.With this software veterinerian will make record process easily and safetly.

The software can be used at every animal clinic easly. They can access to only their task process.In the same time in the program veterinerian can get obligation as daily.

CHAPTER 1

DELPHI

1.1 INTRODUCTION TO DELPHI

The name "Delphi" was never a term with which either Olaf Helmer or Norman Dalkey (the founders of the method) were particular happy. Since many of the early Delphi studies focused on utilizing the technique to make forecasts of future occurrences, the name was first applied by some others at Rand as a joke. However, the name stuck. The resulting image of a priestess, sitting on a stool over a crack in the earth, inhaling sulfur fumes, and making vague and jumbled statements that could be interpreted in many different ways, did not exactly inspire confidence in the method.

The straightforward nature of utilizing an iterative survey to gather information "sounds" so easy to do that many people have done "one" Delphi, but never a second. Since the name gives no obvious insight into the method and since the number of unsuccessful Delphi studies probably exceeds the successful ones, there has been a long history of diverse definitions and opinions about the method. Some of these misconceptions are expressed in statements such as the following that one finds in the literature:

It is a method for predicting future events.

It is a method for generating a quick consensus by a group.

It is the use of a survey to collect information.

It is the use of anonymity on the part of the participants.

It is the use of voting to reduce the need for long discussions.

It is a method for quantifying human judgement in a group setting.

Some of these statements are sometimes true; a few (e.g. consensus) are actually contrary to the purpose of a Delphi. Delphi is a communication structure aimed at producing detailed critical examination and discussion, not at forcing a quick

compromise. Certainly quantification is a property, but only to serve the goal of quickly identifying agreement and disagreement in order to focus attention. It is often very common, even today, for people to come to a view of the Delphi method that reflects a particular application with which they are familiar. In 1975 Linstone and Turoff proposed a view of the Delphi method that they felt best summarized both the technique and its objective:

"Delphi may be characterized as a method for structuring a group communication process, so that the process is effective in allowing a group of individuals, as a whole, to deal with complex problems." The essence of Delphi is structuring of the group communication process. Given that there had been much earlier work on how to facilitate and structure face-to-face meetings, the other important distinction was that Delphi was commonly applied utilizing a paper and pencil communication process among groups in which the members were dispersed in space and time. Also, Delphis were commonly applied to groups of a size (30 to 100 individuals) that could not function well in a face-to-face environment, even if they could find a time when they all could get together.

Additional opportunity has been added by the introduction of Computer Mediated Communication Systems (Hiltz and Turoff, 1978; Rice and Associates, 1984; Turoff, 1989; Turoff, 1991). These are computer systems that support group communications in either a synchronous (Group Decision Support Systems, Desanctis et. al., 1987) or an asynchronous manner (Computer Conferencing). Techniques that were developed and refined in the evolution of the Delphi Method (e.g. anonymity, voting) have been incorporated as basic facilities or tools in many of these computer based systems. As a result, any of these systems can be used to carry out some form of a Delphi process or Nominal Group Technique (Delbecq, et. al., 1975).

The result, however, is not merely confusion due to different names to describe the same things; but a basic lack of knowledge by many people working in these areas as to what was learned in the studies of the Delphi Method about how to properly employ these techniques and their impact on the communication process. There seems to be a great deal of "rediscovery" and repeating of earlier misconceptions and difficulties.

Given this situation, the primary objective of this chapter is to review the specific properties and methods employed in the design and execution of Delphi Exercises and to examine how they may best be translated into a computer based environment.

1.2 WHAT IS DELPHI?

Delphi is an object oriented, component based, visual, rapid development environment for event driven Windows applications, based on the Pascal language. Unlike other popular competing Rapid Application Development (RAD) tools, Delphi compiles the code you write and produces really tight, natively executable code for the target platform. In fact the most recent versions of Delphi optimise the compiled code and the resulting executables are as efficient as those compiled with any other compiler currently on the market.The term "visual" describes Delphi very well. All of the user interface development is conducted in a What You See Is What You Get environment (WYSIWYG), which means you can create polished, user friendly interfaces in a very short time, or prototype whole applications in a few hours.

Delphi is, in effect, the latest in a long and distinguished line of Pascal compilers (the previous versions of which went by the name "Turbo Pascal") from the company formerly known as Borland, now known as Inprise. In common with the Turbo Pascal compilers that preceded it, Delphi is not just a compiler, but a complete development environment. 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 What's more, the development environment itself is extensible, and there are a number of add ins available to perform functions such as memory leak detection and profiling.

In short, Delphi includes just about everything you need to write applications that will run on an Intel platform under Windows, but if your target platform is a Silicon Graphics running IRIX, or a Sun Sparc running SOLARIS, or even a PC running LINUX, then you will need to look elsewhere for your development tools.

This specialisation on one platform and one operating system, makes Delphi a very strong tool. The code it generates runs very rapidly, and is very stable, once your own bugs have been ironed out!

1.3 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

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.

1.4 VERSIONS ARE THERE AND HOW DO THEY DIFFER?

Borland (as they were then) has a long tradition in the creation of high speed compilers. One of their best known products was Turbo Pascal - a tool that many programmers cut their teeth on. With the rise in importance of the Windows environment, it was only a matter of time before development tools started to appear that were specific to this new environment.

In the very beginning, Windows produced SDKs (software development kits) that were totally non-visual (user interface development was totally separated from the development of the actual application), and required great patience and some genius to

get anything working with. Whilst these tools slowly improved, they still required a really good understanding of the inner workings of Windows.

To a great extent these criticisms were dispatched by the release of Microsoft's Visual Basic product, which attempted to bring Windows development to the masses. It achieved this to a great extent too, and remains a popular product today. However,it suffered from several drawbacks:

1) It wasn't as stable as it might have been

2) It was an interpreted language and hence was slow to run

3) It had as its underlying language BASIC, and most "real" programmers weren't so keen!

Into this environment arrived the eye opening Delphi I product, and in many ways the standard for visual development tools for Windows was set. This first version was a 16 bit compiler, and produced executable code that would run on Windows 3.1 and Windows 3.11. Of course, Microsoft have ensured (up to now) that their 32 bit operating systems (Win95, Win98, and Win NT) will all run 16 bit applications, however, many of the features that were introduced in these newer operating systems are not accessible to the 16 bit applications developed with Delphi I.

Delphi 2 was released quite soon after Delphi I, and in fact included a full distribution of Delphi I on the same CD. Delphi 2, (and all subsequent versions) have been 32 bit compilers, producing code that runs exclusively on 32bit Windows platforms. (We ignore for simplicity the WIN32S DLLs which allow Win 3.1x to run some 32 bit applications).

Delphi is currently standing at Version 4.0, with a new release (version 5.0) expected shortly. In its latest version, Delphi has become somewhat feature loaded, and as a result, we would argue, less stable than the earlier versions. However, in its defence, Delphi (and Borland products in general) have always been more stable than their competitors products, and the majority of Delphi 4's glitches are minor and forgivable -

just don't try and copy/paste a selection of your code, midway through a debugging session!

The reasons for the version progression include the addition of new components, improvements in the development environment, the inclusion of more internet related support and improvements in the documentation. Delphi at version 4 is a very mature product, and Inprise has always been responsive in developing the product in the direction that the market requires it to go. Predominantly this means right now, the inclusion of more and more Internet, Web and CORBA related tools and components - a trend we are assured continues with the release of version 5.0

For each version of Delphi there are several sub-versions, varying in cost and features, from the most basic "Developer" version to the most complete (and expensive) "Client Server" version. The variation in price is substantial, and if you are contemplating a purchase, you should study the feature list carefully to ensure you are not paying for features you will never use. Even the most basic "Developer" version contains the vast majority of the features you are likely to need on a day to day basis. Don't assume that you will need Client Server, simply because you are intending to write a large database application - The developer edition is quitcapable ofthis.

1.5 SOME KNOWLEDGE ABOUT 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 7. There are more recent versions available (2005 and 2006) however Delphi 7 should be available inexpensively compared to the new versions which will set you back a lot of money. Delphi 7 will more than likely be available in a magazine for free.

1.5.1 Example: Try First Delphi Program

First thing is first, fire up your copy of Delphi and open the Project > Options menu. To compile a console application you need to change a setting on the Linker tab called 'Generate console application', check the box and click OK. Now select File > Close All if anything is already loaded. Then select File > New > Other > Console Application.

Notice the first line refers to the keyword program. You can rename this to HelloWorld. You can also remove the commented portion enclosed in curly brackets. The uses keyword allows you to list all units that you want to use in the program. At the moment just leave it as it is, SysUtils is all we need.

Your unit should now look like this:

Delphi Code:

```
program HelloWorld;

  {$APPTYPE CONSOLE}

 uses

  SysUtils;

 begin

 end.
```

Now what we have just done is written a program, it currently doesn't do a thing however. Hit the run button and see the result. Now wasn't that completely worthless.

Luckily this isn't the end of the article so we'll actually have a worthwhile program at the end of it. All we need to do is insert some code in the main procedure we have just made.

Every good programmer's first program was 'Hello World' and you'll be no exception. All we need to do is use the WriteLn procedure to write 'Hello World!' to the console, simple.Notice the semicolon at the end of the line, at the end of any statement you need to add a semicolon. Run the program and see the results…

Now I don't know about you but I saw hello world flash up and go away in a second, if you didn't write the program you wouldn't even know what it said. To solve this problem we need to tell the program to leave the console open until the user is ready to close it. We can use ReadLn for this which reads the users input from the console.

Delphi Code:

```
program HelloWorld;

{$APPTYPE CONSOLE}

uses

  SysUtils;

begin

  WriteLn('Hello World!' + #13#10 + #13#10 +

    'Press RETURN to end...');

  ReadLn;

end.
```

I have added a few extra things into the 'Hello World' string so the user knows what to do to end the program as it could be a bit confusing. '#13#10' is to insert a carriage

return as 13 and 10 are the ASCII codes for a carriage return followed by a new line feed. ASCII can be inserted in this way into strings.

1.5.2 Delphi Style

Coding style, the way you format your code and the way in which you present it on the page.At the end of the day who cares about my style, I can read it, and Delphi strips all the spaces out of it and doesn't care if I indent. Why waste my time?

Neatly present code which conforms to the accepted standards not only makes your code much easier for you to read and debug but also but any one else who might read your code to help you, or learn from you can do so with ease. After all which code is easier to follow, example 1 or 2?

Delphi Code:

```
// Example 1

procedure xyz();

var

x,y,z,a:integer;

begin

x:=1;y:=2;

for z:=x to y do begin

a:=power(z,y);

showmessage(inttostr(a));

end;
```

```
end;
```

Delphi Code:

```
// Example 2
procedure XYZ();
var
X,Y,Z,A: Integer;
begin
  X := 1;
  Y := 2;
  for Z := X to Y do
  begin
        A := Power(Z, Y);
        ShowMessage(IntToStr(A));
  end; // for end
end; // procedure end
```

Design patterns are frequently recurring structures and relationships in object-oriented design. Getting to know them can help you design better, more reusable code and also help you learn to design more complex systems.

Much of the ground-breaking work on design patterns was presented in the book Design Patterns: Elements of Reusable Object-Oriented Software by Gamma, Helm, Johnson and Vlissides. You might also have heard of the authors referred to as "the Gang of Four". If you haven't read this book before and you're designing objects, it's an excellent

primer to help structure your design. To get the most out of these examples, I recommend reading the book as well.

Another good source of pattern concepts is the book Object Models: Strategies, Patterns and Applications by Peter Coad. Coad's examples are more business oriented and he emphasises learning strategies to identify patterns in your own work.

1.6 HOW DELPHI HELPS YOU DEFINE PATTERNS

Delphi implements a fully object-oriented language with many practical refinements that simplify development.

The most important class attributes from a pattern perspective are the basic inheritance of classes; virtual and abstract methods; and use of protected and public scope. These give you the tools to create patterns that can be reused and extended, and let you isolate varying functionality from base attributes that are unchanging.

Delphi is a great example of an extensible application, through its component architecture, IDE interfaces and tool interfaces. These interfaces define many virtual and abstract constructors and operations.

1.6.1 Delphi Examples of Design Patterns

I should note from the outset, there may be alternative or better ways to implement these patterns and I welcome your suggestions on ways to improve the design. The following patterns from the book *Design Patterns* are discussed and illustrated in Delphi to give you a starting point for implementing your own Delphi patterns.

Pattern Name	*Definition*
Singleton	"Ensure a class has only one instance, and provide a global point of access to it."
Adapter	"Convert the interface of a class into another interface clients expect. Adapter lets classes work together that couldn't

	otherwise because of incompatible interfaces."
Template Method	"Define the skeleton of an algorithm in an operation, deferring some steps to subclasses. Template Method lets subclasses redefine certain steps of an algorithm without changing the algorithm's structure."
Builder	"Separate the construction of a complex object from its representation so that the same construction process can create different representations."
Abstract Factory	"Provide an interface for creating families of related or dependant objects without specifying their concrete classes."
Factory Method	"Define an interface for creating an object, but let subclasses decide which class to instantiate. Factory method lets a class defer instantiation to subclasses."

Note: These definitions are taken from *Design Patterns*.

1.6.2 Pattern: Singleton

1.6.2.1 Definition

"Ensure a class has only one instance, and provide a global point of access to it."

This is one of the easiest patterns to implement.

1.6.2.2 Applications in Delphi

There are several examples of this sort of class in the Delphi VCL, such as TApplication, TScreen or TClipboard. The pattern is useful whenever you want a single global object in your application. Other uses might include a global exception handler, application security, or a single point of interface to another application.

1.6.2.3 Implementation Example

To implement a class of this type, override the constructor and destructor of the class to refer to a global (interface) variable of the class.

Abort the constructor if the variable is assigned, otherwise create the instance and assign the variable.

In the destructor, clear the variable if it refers to the instance being destroyed.

Note: To make the creation and destruction of the single instance automatic, include its creation in the initialization section of the unit. To destroy the instance, include its destruction in an ExitProc (Delphi 1) or in the finalization section of the unit (Delphi 2).

1.6.3 Pattern: Adapter

1.6.3.1 Definition

"Convert the interface of a class into another interface clients expect. Adapter lets classes work together that couldn't otherwise because of incompatible interfaces."

1.6.3.2 Applications in Delphi

A typical example of this is the wrapper Delphi generates when you import a VBX or OCX. Delphi generates a new class which translates the interface of the external control into a Pascal compatible interface. Another typical case is when you want to build a single interface to old and new systems.

Note Delphi does not allow class adaption through multiple inheritance in the way described in Design Patterns. Instead, the adapter needs to refer to a specific instance of the old class.

1.6.3.3 Implementation Example

The following example is a simple (read only) case of a new customer class, an adapter class and an old customer class. The adapter illustrates handling the year 2000 problem, translating an old customer record containing two digit years into a new date format. The client using this wrapper only knows about the new customer class. Translation between classes is handled by the use of virtual access methods for the properties. The old customer class and adapter class are hidden in the implementation of the unit.

1.6.4 Pattern: Template Method

1.6.4.1 Definition

"Define the skeleton of an algorithm in an operation, deferring some steps to subclasses. Template Method lets subclasses redefine certain steps of an algorithm without changing the algorithm's structure."

This pattern is essentially an extension of abstract methods to more complex algorithms.

1.6.4.2 Applications in Delphi

Abstraction is implemented in Delphi by abstract virtual methods. Abstract methods differ from virtual methods by the base class not providing any implementation. The descendant class is completely responsible for implementing an abstract method. Calling an abstract method that has not been overridden will result in a runtime error.

1.6.4.3 A typical example of abstraction is the TGraphic class.

TGraphic is an abstract class used to implement TBitmap, TIcon and TMetafile. Other developers have frequently used TGraphic as the basis for other graphics objects such as PCX, GIF, JPG representations. TGraphic defines abstract methods such as Draw, LoadFromFile and SaveToFile which are then overridden in the concrete classes. Other objects that use TGraphic, such as a TCanvas only know about the abstract Draw method, yet are used with the concrete class at runtime.

Many classes that use complex algorithms are likely to benefit from abstraction using the template method approach. Typical examples include data compression, encryption and advanced graphics processing.

1.6.4.4 Implementation Example

To implement template methods you need an abstract class and concrete classes for each alternate implementation. Define a public interface to an algorithm in an abstract base class. In that public method, implement the steps of the algorithm in calls to protected abstract methods of the class. In concrete classes derived from the base class, override each step of the algorithm with a concrete implementation specific to that class.

1.6.5 Pattern: Builder

1.6.5.1 Definition

"Separate the construction of a complex object from its representation so that the same construction process can create different representations."

A Builder seems similar in concept to the Abstract Factory. The difference as I see it is the Builder refers to single complex objects of different concrete classes but containing multiple parts, whereas the abstract factory lets you create whole families of concrete classes. For example, a builder might construct a house, cottage or office. You might employ a different builder for a brick house or a timber house, though you would give them both similar instructions about the size and shape of the house. On the other hand the factory generates parts and not the whole. It might produce a range of windows for buildings, or it might produce a quite different range of windows for cars.

1.6.5.2 Applications in Delphi

The functionality used in Delphi's VCL to create forms and components is similar in concept to the builder. Delphi creates forms using a common interface, through Application.CreateForm and through the TForm class constructor. TForm implements a

common constructor using the resource information (DFM file) to instantiate the components owned by the form. Many descendant classes reuse this same construction process to create different representations. Delphi also makes developer extensions easy. TForm's OnCreate event also adds a hook into the builder process to make the functionality easy to extend.

1.6.5.3 Implementation Example

The following example includes a class TAbstractFormBuilder and two concrete classes TRedFormBuilder and TBlueFormBuilder. For ease of development some common functionality of the concrete classes has been moved into the shared TAbstractFormBuilder class.

1.6.6 Pattern: Abstract Factory

1.6.6.1 Definition

"Provide an interface for creating families of related or dependant objects without specifying their concrete classes."

The Factory Method pattern below is commonly used in this pattern.

1.6.6.2 Applications in Delphi

This pattern is ideal where you want to isolate your application from the implementation of the concrete classes. For example if you wanted to overlay Delphi's VCL with a common VCL layer for both 16 and 32 bit applications, you might start with the abstract factory as a base.

1.6.6.3 Implementation Example

The following example uses an abstract factory and two concrete factory classes to implement different styles of user interface components. TOAbstractFactory is a singleton class, since we usually want one factory to be used for the whole application.

At runtime, our client application instantiates the abstract factory with a concrete class and then uses the abstract interface. Parts of the client application that use the factory don't need to know which concrete class is actually in use.

1.6.7 Pattern: Factory Method

1.6.7.1 Definition

"Define an interface for creating an object, but let subclasses decide which class to instantiate. Factory method lets a class defer instantiation to subclasses."

The Abstact Factory pattern can be viewed as a collection of Factory Methods.

1.6.7.2 Applications in Delphi

This pattern is useful when you want to encapsulate the construction of a class and isolate knowledge of the concrete class from the client application through an abstract interface.

One example of this might arise if you had an object oriented business application potentially interfacing to multiple target DBMS. The client application only wants to know about the business classes, not about their implementation-specific storage and retrieval.

1.6.7.3 Implementation Example

In the Abstract Factory example, each of the virtual widget constructor functions is a Factory Method. In their implementation we define a specific widget class to return.

1.7 KEY ELEMENTS OF DELPHI CLASS DEFINITIONS

1.7.1 Unit Structure

Delphi units (.PAS files) allow declaration of interface and implementation sections. The interface defines the part that is visible to other units using that unit. The keyword *uses* can be added to a unit's interface or implementation section to list the other units that your unit uses. This indicates to the compiler that your unit refers to parts of the used unit's interface. Parts of a unit declared in the implementation section are all private to that unit, i.e. never visible to any other unit. Types, functions and procedures declared in the interface of a unit must have a corresponding implementation, or be declared as external (e.g. a call to a function in a DLL).

1.7.2 Class Interfaces

Classes are defined as types in Delphi and may contain fields of standard data types or other objects, methods declared as functions or procedures, and properties. The type declaration of a class defines its interface and the scope of access to fields, methods and properties of the class. Class interfaces are usually defined in the interface of a unit to make them accessible to other modules using that unit. However they don't need to be. Sometimes a type declaration of a class may be used only within the implementation part of a unit.

1.7.3 Properties

Properties are a specialised interface to a field of a defined type, allowing access control through read and write methods. Properties are not virtual, you can replace a property with another property of the same name, but the parent class doesn't know about the new property. It is however possible to make the access methods of a property virtual.

1.7.4 Inheritance

Delphi's inheritance model is based on a single hierarchy. Every class inherits from TObject and can have only one parent.

A descendant class inherits all of the interface and functionality of its parent class, subject to the scope described below.

Multiple inheritance from more than one parent is not allowed directly. It can be implemented by using a container class to create instances one or more other classes and selectively expose parts of the contained classes.

Private, Protected, Public and Published ScopeScope refers to the visibility of methods and data defined in the interface of a class, i.e. what parts of the class are accessible to the rest of the application or to descendant classes.

The default scope is public, for instance the component instances you add to a form at design time. Public says "come and get me"; it makes the data or method visible to everything at runtime.

Published parts of a class are a specialized form of Public scope. They indicate special behaviour for classes derived from TPersistent. A persistent class can save and restore its published properties to persistent storage using Delphi's standard streaming methods. Published properties also interact with Delphi Object Inspector in the IDE. A class must descend from TPersistent in order to use Published. There's also not much point in publishing methods, since you can't store them, although Delphi's compiler doesn't stop you. Published also lets another application access details of the class through Delphi's runtime type information. This would be rarely used, except in Delphi's design time interaction with its VCL.

Encapsulation or information hiding is essential to object orientation, so Protected and Private scope let you narrow the access to parts of a class.

Protected parts are visible only to descendant classes, or to other classes defined in the same unit.

Private parts are visible only to the defining class, or to other classes defined in the same unit.

It's important to note that once something is given public or published scope, it cannot be hidden in descendant classes.

Static, Virtual and Dynamic Methods; Override and Inherited

Methods declared as virtual or dynamic let you change their behaviour using override in a descendant class. You're unlikely to see a virtual method in the private part of a class, since it could only be overridden in the same unit, although Delphi's compiler doesn't stop you from doing this.

Override indicates that your new method replaces the method of the same name from the parent class. The override must be declared with the same name and parameters as the original method.

When a method is overridden, a call to the parent class's method actually executes the override method in the real class of the object.

Static methods on the other hand have no virtual or override declaration. You can replace a method of a class in a descendant class by redeclaring another method, however this is not object oriented. If you reference your descendant class as the parent type and try to call the replaced method, the static method of the parent class is executed. So in most cases, it's a bad idea to replace a static method.

Virtual and dynamic methods can be used interchangeably. They differ only in their treatment by the compiler and runtime library. Delphi's help explains that dynamic methods have their implementation resolved at compile time and run slightly faster, whereas virtual methods are resolved at runtime, resulting in slightly slower access but a smaller compiled program. Virtual is usually the preferred declaration. Delphi's help suggests using dynamic when you have a base class with many descendants that may not override the method.

The inherited directive lets you refer back to a property or method as it was declared in the parent class. This is most often used in the implementation of an override method, to call the inherited method of the parent class and then supplement its behaviour.

1.7.5 Abstract Methods

Abstract is used in base classes to declare a method in the interface and defer its implementation to a descendant class. I.e. it defines an interface, but not the underlying

operation. Abstract must be used with the virtual or dynamic directive. Abstract methods are never implemented in the base class and must be implemented in descendant classes to be used. A runtime error occurs if you try to execute an abstract method that is not overridden. Calling inherited within the override implementation of an abstract method will also result in a runtime error, since there is no inherited behaviour.

1.7.6 Messages

Delphi's handling of Windows messages is a special case of virtual methods. Message handlers are implemented in classes that descend from TControl. I.e classes that have a handle and can receive messages. Message handlers are always virtual and can be declared in the private part of a class interface, yet still allow the inherited method to be called. Inherited in a message handler just uses the keyword inherited, there is no need to supply the name of the method to call.

1.7.7 Events

Events are also an important characteristic of Delphi, since they let you delegate extensible behaviour to instances of a class. Events are properties that refer to a method of another object. Events are not inherited in Delphi 1; Delphi 2 extends this behaviour to let you use inherited in an event. . Inherited in an event handler just uses the keyword inherited, there is no need to supply the name of the method to call.

Events are particularly important to component developers, since they provide a hook for the user of the component to modify its behaviour in a way that may not be foreseen at the time the component is written.

1.7.8 Constructors and Destructors

The constructor and destructor are two special types of methods. The constructor initializes a class instance (allocates memory initialized to 0) and returns a reference (pointer) to the object. The destructor deallocates memory used by the object (but not the memory of other objects created by the object).

Classes descended from TObject have a static constructor, Create, and a virtual destructor Destroy.

TComponent introduces a new public property, the Owner of the component and this must be initialized in the constructor. TComponent's constructor is declared virtual, i.e. it can be overridden in descendant classes.It is essential when you override a virtual constructor or destructor in a TComponent descendant to include a call to the inherited method.

1.8 THE VCL TO APPLICATIONS DEVELOPERS

Applications Developers create complete applications by interacting with the Delphi visual environment (as mentioned earlier, this is a concept nonexistent in many other frameworks). These people use the VCL to create their user-interface and the other elements of their application: database connectivity, data validation, business rules, etc..

Applications Developers should know which properties, events, and methods each component makes available. Additionally, by understanding the VCL architecture, Applications Developers will be able to easily identify where they can improve their applications by extending components or creating new ones. Then they can maximize the capabilities of these components, and create better applications.

1.8.1 The VCL to Component Writers

Component Writers expand on the existing VCL, either by developing new components, or by increasing the functionality of existing ones. Many component writers make their components available for Applications Developers to use.

A Component Writer must take their knowledge of the VCL a step further than that of the Application Developer. For example, they must know whether to write a new component or to extend an existing one when the need for a certain characteristic arises. This requires a greater knowledge of the VCL's inner workings.

1.8.2 The VCL is made up of components

Components are the building blocks that developers use to design the user-interface and to provide some non-visual capabilities to their applications. To an Application Developer, a component is an object most commonly dragged from the Component palette and placed onto a form. Once on the form, one can manipulate the component's properties and add code to the component's various events to give the component a specific behavior. To a Component Writer, components are objects in Object Pascal code. Some components encapsulate the behavior of elements provided by the system, such as the standard Windows 95 controls. Other objects introduce entirely new visual or non-visual elements, in which case the component's code makes up the entire behavior of the component.

The complexity of different components varies widely. Some might be simple while others might encapsulate a elaborate task. There is no limit to what a component can do or be made up of. You can have a very simple component like a TLabel, or a much more complex component which encapsulates the complete functionality of a spreadsheet.

1.8.3 Component Types, structure, and VCL hierarchy

Components are really just special types of objects. In fact, a component's structure is based on the rules that apply to Object Pascal. There are three fundamental keys to understanding the VCL.

First, you should know the special characteristics of the four basic component types: standard controls, custom controls, graphical controls and non-visual components.

Second, you must understand the VCL structure with which components are built. This really ties into your understanding of Object Pascal's implementation. Third, you should be familiar with the VCL hierarchy and you should also know where the four component types previously mentioned fit into the VCL hierarchy. The following paragraphs will discuss each of these keys to understanding the VCL.

1.8.4 Component Types

As a component writer, there four primary types of components that you will work with in Delphi: standard controls, custom controls, graphical controls, and non-visual components. Although these component types are primarily of interest to component writers, it's not a bad idea for applications developers to be familiar with them. They are the foundations on which applications are built.

1.8.4.1 Standard Components

Some of the components provided by Delphi 2.0 encapsulate the behavior of the standard Windows controls: TButton, TListbox and Tedit, for example. You will find these components on the *Standard* page of the Component Palette. These components are Windows' common controls with Object Pascal wrappers around them.

Each standard component looks and works like the Windows' common control which it encapsulates. The VCL wrapper's simply makes the control available to you in the form of a Delphi component-it doesn't define the common control's appearance or functionality, but rather, surfaces the ability to modify a control's appearance/functionality in the form of methods and properties. If you have the VCL source code, you can examine how the VCL wraps these controls in the file STDCTRLS.PAS.

If you want to use these standard components unchanged, there is no need to understand how the VCL wraps them. If, however, you want to extend or change one of these components, then you must understand how the Window's common control is wrapped by the VCL into a Delphi component.

For example, the Windows class LISTBOX can display the list box items in multiple columns. This capability, however, isn't surfaced by Delphi's TListBox component (which encapsulates the Windows LISTBOX class). (TListBox only displays items in a single column.) Surfacing this capability requires that you override the default creation of the TListBox component.

This example also serves to illustrate why it is important for Applications Developers to understand the VCL. Just knowing this tidbit of information helps you to identify where enhancements to the existing library of components can help make your life easier and more productive.

1.8.4.2 Custom components

Unlike standard components, custom components are controls that don't already have a method for displaying themselves, nor do they have a defined behavior. The Component Writer must provide to code that tells the component how to draw itself and determines how the component behaves when the user interacts with it. Examples of existing custom components are the TPanel and TStringGrid components.

It should be mentioned here that both standard and custom components are *windowed* controls. A "windowed control" has a window associated with it and, therefore, has a window handle. Windowed controls have three characteristics: they can receive the input focus, they use system resources, and they can be parents to other controls. (Parents is related to containership, discussed later in this paper.) An example of a component which can be a container is the TPanel component.

1.8.4.3 Graphical components

Graphical components are visual controls which cannot receive the input focus from the user. They are non-windowed controls. Graphical components allow you to display something to the user without using up any system resources; they have less "overhead" than standard or custom components. Graphical components don't require a window handle-thus, they cannot can't get focus. Some examples of graphical components are the TLabel and TShape components.

Graphical components cannot be containers of other components. This means that they cannot own other components which are placed on top of them.

1.8.4.4 Non-visual components

Non-visual components are components that do not appear on the form as controls at run-time. These components allow you to encapsulate some functionality of an entity

within an object. You can manipulate how the component will behave, at design-time, through the Object Inspector. Using the Object Inspector, you can modify a non-visual component's properties and provide event handlers for its events. Examples of such components are the TOpenDialog, TTable, and TTimer components.

1.8.4.5 Structure of a component

All components share a similar structure. Each component consists of common elements that allow developers to manipulate its appearance and function via properties, methods and events. The following sections in this paper will discuss these common elements as well as talk about a few other characteristics of components which don't apply to all components.

1.8.4.6 Component properties

Properties provide an extension of an object's fields. Unlike fields, properties do not store data: they provide other capabilities. For example, properties may use methods to read or write data to an object field to which the user has no access. This adds a certain level of protection as to how a given field is assigned data. Properties also cause "side effects" to occur when the user makes a particular assignment to the property. Thus what appears as a simple field assignment to the component user could trigger a complex operation to occur behind the scenes.

1.9 PROPERTIES PROVIDE ACCESS TO INTERNAL STORAGE FIELDS

There are two ways that properties provide access to internal storage fields of components: directly or through access methods. Examine the code below which illustrates this process.

```
TCustomEdit = class(TWinControl)
private
  FMaxLength: Integer;
protected
  procedure SetMaxLength(Value: Integer);
```

```
...
published
  property MaxLength: Integer read
     FMaxLength write SetMaxLength default 0;
...
end;
```

The code above is snippet of the TCustomEdit component class. TCustomEdit is the base class for edit boxes and memo components such as TEdit, and TMemo.

TCustomEdit has an internal field FMaxLength of type Integer which specifies the maximum length of characters which the user can enter into the control. The user doesn't directly access the FMaxLength field to specify this value. Instead, a value is added to this field by making an assignment to the MaxLength property.

The property MaxLength provides the access to the storage field FMaxLength. The property definition is comprised of the property name, the property type, a read declaration, a write declaration and optional default value.

The read declaration specifies how the property is used to read the value of an internal storage field. For instance, the MaxLength property has direct read access to FMaxLength. The write declaration for MaxLength shows that assignments made to the MaxLength property result in a call to an *access method* which is responsible for assigning a value to the FMaxLength storage field. This access method is SetMaxLength.

1.9.1 Property-access methods

Access methods take a single parameter of the same type as the property. One of the primary reasons for write access methods is to cause some side-effect to occur as a result of an assignment to a property. Write access methods also provide a method layer over assignments made to a component's fields. Instead of the component user making the assignment to the field directly, the property's write access method will assign the

value to the storage field if the property refers to a particular storage field. For example, examine the implementation of the SetMaxLength method below.

```
procedure TCustomEdit.SetMaxLength(Value: Integer);

begin

 if FMaxLength <> Value then

 begin

  FMaxLength := Value;

  if HandleAllocated then

     SendMessage(Handle, EM_LIMITTEXT, Value, 0);

 end;

end;
```

The code in the SetMaxLength method checks if the user is assigning the same value as that which the property already holds. This is done as a simple optimization. The method then assigns the new value to the internal storage field, FMaxLength. Additionally, the method then sends an EM_LIMITTEXT Windows message to the window which the TCustomEdit encapsulates. The EM_LIMITTEXT message places a limit on the amount of text that a user can enter into an edit control. This last step is what is referred to as a *side-effect* when assigning property values. Side effects are any additional actions that occur when assigning a value to a property and can be quite sophisticated.

Providing access to internal storage fields through property access methods offers the advantage that the Component Writer can modify the implementation of a class without modifying the interface. It is also possible to have access methods for the read access of a property. The read access method might, for example, return a type which is different that that of a properties storage field. For instance, it could return the string representation of an integer storage field.

Another fundamental reason for properties is that properties are accessible for modification at run-time through Delphi's Object Inspector. This occurs whenever the declaration of the property appears in the published section of a component's declaration.

1.9.2 Types of properties

Properties can be of the standard data types defined by the Object Pascal rules. Property types also determine how they are edited in Delphi's Object Inspector. The table below shows the different property types as they are defined in Delphi's online help.

Property type	Object Inspector treatment
Simple	Numeric, character, and string properties appear in the Object Inspector as numbers, characters, and strings, respectively. The user can type and edit the value of the property directly.
Enumerated	Properties of enumerated types (including Boolean) display the value as defined in the source code. The user can cycle through the possible values by double-clicking the value column. There is also a drop-down list that shows all possible values of the enumerated type.
Set	Properties of set types appear in the Object Inspector looking like a set. By expanding the set, the user can treat each element of the set as a Boolean value: True if the element is included in the set or False if it's not included.
Object	Properties that are themselves objects often have their own property editors. However, if the object that is a property also has published properties, the Object Inspector allows the user to expand the list of object properties and edit them individually. Object properties must descend from TPersistent.
Array	Array properties must have their own property editors. The Object Inspector has no built-in support for editing array properties.

For more information on properties, refer to the "Component Writers Guide" which ships with Delphi.

1.9.3 Methods

Since components are really just objects, they can have methods. We will discuss some of the more commonly used methods later in this paper when we discuss the different levels of the VCL hierarchy.

1.9.4 Events

Events provide a means for a component to notify the user of some pre-defined occurrence within the component. Such an occurrence might be a button click or the pressing of a key on a keyboard.

Components contain special properties called events to which the component user assigns code. This code will be executed whenever a certain event occurs. For instance, if you look at the events page of a TEdit component, you'll see such events as OnChange, OnClick and OnDblClick. These events are nothing more than pointers to methods.

When the user of a component assigns code to one of those events, the user's code is referred to as an event handler. For example, by double clicking on the events page for a particular event causes Delphi to generate a method and places you in the Code Editor where you can add your code for that method. An example of this is shown in the code below, which is an OnClick event for a TButton component.

It becomes clearer that events are method pointers when you assign an event handler to an event programmatically. The above example was Delphi generated code. To link your own an event handler to a TButton's OnClick event at run time you must first create a method that you will assign to this event. Since this is a method, it must belong to an existing object. This object can be the form which owns the TButton component although it doesn't have to be. In fact, the event handlers which Delphi creates belong to the form on which the component resides. The code below illustrates how you would create an event handler method.

When you define methods for event handlers, these methods must be defined as the same type as the event property and the field to which the event property refers. For

instance, the OnClick event refers to an internal data field, FOnClick. Both the property OnClick, and field FOnClick are of the type TNotifyEvent. TNotifyEvent is a procedural type as shown below:

TNotifyEvent = procedure (Sender: TObject) of object;

Note the use of the of object specification. This tells the compiler that the procedure definition is actually a method and performs some additional logic like ensuring that an implicit Self parameter is also passed to this method when called. Self is just a pointer reference to the class to which a method belongs.

1.9.5 Containership

Some components in the VCL can own other components as well as be parents to other components. These two concepts have a different meaning as will be discussed in the section to follow.

1.9.6 Ownership

All components may be owned by other components but not all components can own other components. A component's Owner property contains a reference to the component which owns it.

The basic responsibility of the owner is one of resource management. The owner is responsible for freeing those components which it owns whenever it is destroyed. Typically, the form owns all components which appear on it, even if those components are placed on another component such as a TPanel. At design-time, the form automatically becomes the owner for components which you place on it. At run-time, when you create a component, you pass the owner as a parameter to the component's constructor. For instance, the code below shows how to create a TButton component at run-time and passes the form's implicit Self variable to the TButton's Create constructor. TButton.Create will then assign whatever is passed to it, in this case Self or rather the form, and assign it to the button's Owner property.

MyButton := TButton.Create(self);

When the form that now owns this TButton component gets freed, MyButton will also be freed.

You can create a component without an owner by passing nil to the component's Create constructor, however, you must ensure that the component is freed when it is no longer needed. The code below shows you how to do this for a TTable component.

1.9.7 Parenthood

Parenthood is a much different concept from ownership. It applies only to windowed components, which can be parents to other components. Later, when we discuss the VCL hierarchy, you will see the level in the hierarchy which introduces windowed controls.

Parent components are responsible for the display of other components. They call the appropriate methods internally that cause the children components to draw themselves. The Parent property of a component refers to the component which is its parent. Also, a component's parent does not have to be it's owner. Although the parent component is mainly responsible for the display of components, it also frees children components when it is destroyed.

Windowed components are controls which are visible user interface elements such as edit controls, list boxes and memo controls. In order for a windowed component to be displayed, it must be assigned a parent on which to display itself. This task is done automatically by Delphi's design-time environment when you drop a component from the Component Palette onto your form.

CHAPTER 2

DATABASE

Every thing around us has a particular identity. To identify anything system, actor or person in words we need a data or information. So this information is valuable and in this advanced era we can store it in database and access this data by the blink of eye.

For an instant if we go through the definitions of database we may find following definitions.

A database is a collection of related information.

A database is an organized body of related information.

2.1 DEMERITS OF ABSENCE OF DATABASE

A glance on the past will may help us to reveal the drawbacks in case of absence of database.

In the past when there wasn't proper system of database, Much paper work was need to do and to handle great deal of written paper documentation was giant among the problems itself.

In the huge networks to deal with equally bulky data, more workers are needed which affidavit cost much labor expanses.

The old criteria for saving data and making identification was much time consuming such as if we want to search the particular data of a person.

Before the Development of Computer database it was a great problem to search for some thing. Efforts to avoid the headache of search often results in new establishments of data.

Before the development of database it seemed very unsafe to keep the worthy information. In Some situation some big organization had to employee the special persons in order to secure the data.

Before the implementation of database any firm had to face the plenty of difficulties in order to maintain their Management. To hold the check on the expenses of the firm, the manager faced difficulties.

2.2 MERITS OF DATABASE

The modern era is known as the golden age computer sciences and technology. In a simple phrase we can express that the modern age is built on the foundation of database.

If we carefully watch our daily life we can examine that some how our daily life is being connected with database.

There are several benefits of database developments.

Now with the help of computerized database we can access data in a second.

By the development of the database we can make data more secure.

By the development of database we can reduce the cost.

2.3 DATABASE DESIGN

The design of a database has to do with the way data is stored and how that data is related. The design process is performed after you determine exactly what information needs to be stored and how it is to be retrieved.

A collection of programs that enables you to store, modify, and extract information from a database. There are many different types of DBMS ranging from small systems that run on personal computers to huge systems that run on mainframes. The following are examples of database applications:

Computerized library systems

Automated teller machines

Flight reservation systems

Computerized parts inventory systems

From a technical standpoint, DBMS can differ widely. The terms relational, network, flat, and hierarchical all refer to the way a DBMS organizes information internally. The internal organization can affect how quickly and flexibly you can extract information.

Requests for information from a database are made in the form of a query.

Database design is a complex subject. A properly designed database is a model of a business, Country Database or some other in the real world. Like their physical model counterparts, data models enable you to get answers about the facts that make up the objects being modeled. It's the questions that need answers that determine which facts need to be stored in the data model.

In the relational model, data is organized in tables that have the following characteristics: every record has the same number of facts, every field contains the same type of facts (Data) in each record, and there is only one entry for each fact. No two records are exactly the same.

The more carefully you design, the better the physical database meets users' needs. In the process of designing a complete system, you must consider user needs from a variety of viewpoints.

2.4 DATABASE MODELS

Various techniques are used to model data structures. Certain models are more easily implemented by some types of database management systems than others. For any one logical model various physical implementation may be possible. An example of this is the relational model: in larger systems the physical implementation often has indexes which point to the data; this is similar to some aspects of common implementations of the network model. But in small relational database the data is often stored in a set of

files, one per table, in a flat, un-indexed structure. There is some confusion below and elsewhere in this article as to logical data model vs. its physical implementation.

2.4.1 Flat Model

The flat (or table) model consists of a single, two dimensional array of data elements, where all members of a given column are assumed to be similar values, and all members of a row are assumed to be related to one another. For instance, columns for name and password might be used as a part of a system security database. Each row would have the specific password associated with a specific user. Columns of the table often have a type associated with them, defining them as character data, date or time information, integers, or floating point numbers. This model is the basis of the spreadsheet.

2.4.2 Network Model

The network model allows multiple datasets to be used together through the use of pointers (or references). Some columns contain pointers to different tables instead of data. Thus, the tables are related by references, which can be viewed as a network structure. A particular subset of the network model, the hierarchical model, limits the relationships to a tree structure, instead of the more general directed graph structure implied by the full network model.

2.4.3 Relational Model

The relational data model was introduced in an academic paper by E.F. Cod in 1970 as a way to make database management systems more independent of any particular application. It is a mathematical model defined in terms of predicate logic and set theory.

Although the basic idea of a relational database has been very popular, relatively few people understand the mathematical definition and only a few obscure DBMSs implement it completely and without extension. Oracle, for example, can be used in a purely relational way, but it also allow tables to be defined that allow duplicate rows an extension (or violation) of the relational model. In common English usage, a DBMS is

called relational if it supports relational operational operations, regardless of whether it enforces strict adherence to the relational model. The following is an informal, not-technical explanation of how "relational" database management systems commonly work.

A relational database contains multiple tables, each similar to the one in the "flat" database model. However, unlike network databases, the tables are not linked by pointers. Instead, keys are used to match up rows of data in different tables. A key is just one or more columns in one table that correspond to columns in other tables. Any column can be a key, or multiple columns can be grouped together into a single key. Unlike pointers, it's not necessary to define all the keys in advance; a column can be used as a key even if it wasn't originally intended to be one.

A key that can be used to uniquely identify a row in a table is called a unique key. Typically one of the unique keys is the preferred way to refer to row; this is defined as the table's primary key.

When a key consists of data that has an external, real-world meaning (such as a person's name, a book's ISBN, or a car's serial number), it's called a "natural" key. If no nature key is suitable, an arbitrary key can be assigned (such as by given employees ID numbers). In practice, most databases have both generated and natural keys, because generated keys can be used internally to create links between rows that can't break, while natural keys can be used, less reliably, for searches and for integration with other databases. (For example, records in two independently developed databases could be matched up by social security number, except when the social security numbers are incorrect, missing, or have changed).

2.4.3.1 Why we use a Relational Database Design

Maintaining a simple, so-called flat database consisting of a single table doesn't require much knowledge of database theory. On the other hand, most database worth maintaining are quite a bit more complicated than that. Real life databases often have hundreds of thousands or even millions of records, with data that are very intricately related. This is where using a full-fledged relational database program becomes essential. Consider, for example, the Library of Congress, which has over 16 million

books in its collection. For reasons that will become apparent soon, a single table simply will not do for this database.

2.5 RELATIONSHIPS BETWEEN TABLES

When you create tables for an application, you should also consider the relationships between them. These relationships give a relational database much of its power. There are three types of relationships between tables: one-to-one, one-to-many and many-to-many relationships.

2.5.2 One-To-One Relationships

In a one-to-one relationship, each record in one table corresponds to a single record in a second table. This relationship is not very common, but it can offer several benefits. First, you can put the fields from both tables into a single, combined table. One reason for using two tables is that each field is a property of a separate entity, such as owner operators and their tracks. Each operator can operate just one truck at a time, but the fields for the operator and truck tables refer to different entities.

A one-to-one relationship can also reduce the time needed to open a large table by placing some of the table's columns in a second, separate table. This approach makes particular sense when a table has some fields that are used infrequently. Finally, a one-to-one relationship can support in a table requires security, placing them in a separate table lets your application restrict to certain fields. Your application can link the restricted table back to the main table via a one-to-one relationship so that people with proper permissions can edit, delete, and add new records to these fields.

2.5.3 One-To-Many Relationships

A one-to-many relationship, in which a row from one table corresponds to one or more rows from a second table, is more common. This kind of relationship can form the basis for a Many-To-Many relationship as well.

2.6 DATA MODELING

In information system design, data modeling is the analysis and design of the information in the system, concentrating on the logical entities and the logical dependencies between these entities. Data modeling is an abstraction activity in that the details of the values of individual data observations are ignored in favor of the structure, relationships, names and formats of the data of interest, although a list of valid values is frequently recorded. It is by the data model that definitions of what the data means is related to the data structures.

While a common term for this activity is "Data Analysis" the activity actually has more in common with the ideas and methods of synthesis (putting things together), than it does in the original meaning of the term analysis (taking things apart). This is because the activity strives to bring the data structures of interest together in a cohesive, inseparable, whole by eliminating unnecessary data redundancies and relating data structures by relationships.

2.6.1 Database Normalization

Database normalization is a series of steps followed to obtain a database design that allows for consistent storage and efficient access of data in a relational database. These steps reduce data redundancy and the risk of data becoming inconsistent.

However, many relational DBMS lack sufficient separation between the logical database design and the physical implementation of the data store, such that queries against a fully normalized database often perform poorly. In this case de-normalizations are sometimes used to improve performance, at the cost of reduced consistency.

2.6.2 Primary Key

In database design, a primary key is a value that can be used to identify a particular row in a table. Attributes are associated with it. Examples are names in a telephone book (to look up telephone numbers), words in a dictionary (to look up definitions) and Dewey Decimal Numbers (to look up books in a library).

In the relational model of data, a primary key is a candidate key chosen as the main method of uniquely identifying a relation. Practical telephone books, dictionaries and libraries can not use names, words or Dewey Decimal System Numbers as candidate keys because they do not uniquely identify telephone numbers, word definitions or books. In some design situations it is impossible to find a natural key that uniquely identifies a relation. A surrogate key can be used as the primary key. In other situations there may be more than one candidate key for a relation, and no candidate key is obviously preferred. A surrogate key may be used as the primary key to avoid giving one candidate key artificial primacy over the others. In addition to the requirement that the primary key be a candidate key, there are several other factors which may make a particular choice of key better than others for a given relation.

The primary key should generally be short to minimize the amount of data that needs to be stored by other relations that reference it. A compound key is usually not appropriate. (However, this is a design consideration, and some database management systems may be better than others in this regard.)

The primary key should be immutable, meaning its value should not be changed during the course of normal operations of the database. (Recall that a primary key is the means of uniquely identifying a tuple, and that identity by definition, never changes.) This avoids the problem of dangling references or orphan records created by other relations referring to a tuple whose primary key has changed. If the primary key is immutable, this can never happen.

2.6.3 Foreign Key

A foreign key (FK) is a field in a database record under one primary key that points to a key field of another database record in another table where the foreign key of one table refers to the primary key of the other table. This way references can be made to link information together and it is an essential part of database normalization.

For example, a person sending an e-mail needs not to include the entire text of a book in the e-mail. Instead, they can include the ISBN of the book, and interested persons can then use the number to get information about the book, or even the book itself. The ISBN is the primary key of the book, and it is used as a foreign key in the e-mail.

Note that using a foreign key often assumes its existence as a primary key somewhere else. Improper foreign key/primary key relationships are the source of many database problems.

2.6.4 Compound Key

In database design, a compound key (also called a composite key) is a key that consists on 2 or more attributes.

No restriction is applied to the attribute regarding their (initial) ownership within the data model. This means that any one, none or all, of the multiple attributes within the compound key can be foreign keys. Indeed, a foreign key may, itself, be a compound key.

Compound keys almost always originate from attributive or associative entities (tables) within the model, but this is not an absolute value.

CHAPTER 3
MYSQL

3.1 INTRODUCTION TO MYSQL

This chapter provides a tutorial introduction to MySQL by showing how to use the mysql client program to create and use a simple database. mysql (sometimes referred to as the ``terminal monitor'' or just ``monitor'') is an interactive program that allows you to connect to a MySQL server, run queries, and view the results. mysql may also be used in batch mode: you place your queries in a file beforehand, then tell mysql to execute the contents of the file. Both ways of using mysql are covered here.

To see a list of options provided by mysql, invoke it with the --help option:

shell> mysql --help

This chapter assumes that mysql is installed on your machine and that a MySQL server is available to which you can connect. If this is not true, contact your MySQL administrator. (If you are the administrator, you will need to consult other sections of this manual.)

This chapter describes the entire process of setting up and using a database. If you are interested only in accessing an already-existing database, you may want to skip over the sections that describe how to create the database and the tables it contains.

Because this chapter is tutorial in nature, many details are necessarily left out. Consult the relevant sections of the manual for more information on the topics covered here.

3.2 WHAT IS MYSQL?

3.2.1 Definition

MySQL is an open source software relational database management system (RDBMS) which

uses a SQL (Structured Query Language)

SQL is the standard language used for interacting with databases.

3.3 WHY CHOOSE MYSQL?

There are many relational databases available to use, so why choose MySQL?

We are specifically interested in databases which PHP supports; these include Oracle, IBM's DB2 and Microsoft's SQL Server (all of which cost money).

The two main open source (free) alternatives to these are PostgreSQL and MySQL. PostgreSQL is arguably the better of the two, but MySQL is better supported on Windows, and is a popular choice among Web hosts that provide support for PHP.

Here are some of MySQL's advantages

- It's fast
- It's free to use, and commercial licenses are reasonable
- It's easy to use
- It is cross platform
- There is a wide community of technical support
- It's secure
- It supports large databases
- It is designed specifically for web base applications and hence works very well partnered with PHP

3.4 PREPARING THE WINDOWS MYSQL ENVIRONMENT

Starting with MySQL X.2X.X8, the Windows distribution includes both the normal and the MySQL- Max server binaries. Here is a list of the different MySQL servers you can use:

mysqld	Compiled with full debugging and automatic memory allocation checking, symbolic links, InnoDB and DBD tables.
mysql-opt	Optimized binary with no support for transactional tables.
mysqld-nt	Optimized binary for NT with support for named pipes. You can run this version on Win98, but in this case no named pipes are created and you must have TCP/IP installed.
mysqld-max	Optimized binary with support for symbolic links, InnoDB and DBD tables.
mysqld-max-nt	Like mysqld-max, but compiled with support for named pipes.

All of the above binaries are optimized for the Pentium Pro processor but should work on any Intel processor >=iX86

In the following circumstance, you will need to use the MySQL configuration file:

- The install/data directories are different than the default 'c:\mysql' and 'c:\mysql\data'.
- If you want to use one of these servers:
  - mysqld.exe
  - mysqld-max.exe
  - mysqld-max-nt.exe
- If you need to tune the server settings.

There are two configuration files with the same function: 'my.cnf' and 'my.ini' file, however, only one of these can/should be used. Both files are plain text. The 'my.cnf' file should be created in the root directory of drive C and the 'my.ini' file in the WinDir directory e.g.: C:\WINDOWS or C:\WINNT. If your PC uses a boot loader where the C drive isn't the boot drive, then your only option is to use the 'my.ini' file. Also note that if you use the WinMySQLAdmin tool, only the

'my.ini' file is used. The '\mysql\bin' directory contains a help file with instructions for using this tool.

Using Notepad, create the configuration file and edit the base section and keys:

[mysqld]

basedir = the_install_path # e.g. 'c:\mysql'

datadir = the_data_path # e.g. 'c:\mysql\data' or 'd:\mydata\data'

If the data directory is other than the default 'c:\mysql\data', you must cut the whole

'\data\mysql' directory and paste it on the your option new directory, e.g. 'd:\mydata\mysql'.

If you want to use the InnoDB transaction tables, you need to manually create two new directories to hold the InnoDB data and log files, e.g. 'c:\ibdata' and 'c:\iblogs'. You will also need to create some extra lines to the configuration file.

If you don't want to use, add the skip-innodb option to the configuration file.

Now you are ready to test starting the server.

3.5 STARTING THE SERVER FOR THE FIRST TIME

Testing from a DOS command prompt is the best thing to do because the server prints messages, so if something is wrong with your configuration, you will see a more accurate error message which will make it easier to identify and fix any problems.

Make sure you're in the right directory *(C:\>cd \mysql\bin),*

 # To install mysqld as a standalone program, enter:

C:\mysql\bin> mysqld-max --standalone

You should see the below print messages:

```
InnoDB: The first specified data file c:\ibdata\ibdata1 did not exist:
InnoDB: a new database to be created!
InnoDB: Setting file c:\ibdata\ibdata1 size to 209715200
InnoDB: Database physically writes the file full: wait...
InnoDB: Log file c:\iblogs\ib_logfile0 did not exist: new to be created
InnoDB: Setting log file c:\iblogs\ib_logfile0 size to 31457280
InnoDB: Log file c:\iblogs\ib_logfile1 did not exist: new to be created
InnoDB: Setting log file c:\iblogs\ib_logfile1 size to 31457280
InnoDB: Log file c:\iblogs\ib_logfile2 did not exist: new to be created
InnoDB: Setting log file c:\iblogs\ib_logfile2 size to 31457280
InnoDB: Doublewrite buffer not found: creating new
InnoDB: Doublewrite buffer created
InnoDB: creating foreign key constraint system tables
InnoDB: foreign key constraint system tables created
011024 10:58:25  InnoDB: Started
```

 # To install mysql as a service (Windows 2000), enter:

C:\mysql\bin> mysqld-nt --install

Now you can start and stop mysqld as follows:

C:\>NET START MySQL C:\>NET STOP MySQL

C:\>NET START MySQL

 # To start the MySQL Monitor, enter:

The MySql service is starting.

The MySQL service was started successfully.

C:\>cd \mysql

C:\mysql>bin\mysql

Welcome to the MySQL Monitor. Commands end with ; or \g. Your MySQL connection id

is 1 to server version X.2X.49-nt Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> (enter a command or enter 'QUIT' to quit)

mysql> QUIT Bye

C:\mysql>NET STOP MySQL The MySQL service is stopping.

The MySQL service was stopped successfully.

C:\mysql>

3.6 CONNECTING TO AND DISCONNECTING FROM THE SERVER

To connect to the server, you'll usually need to provide a MySQL user name when you invoke mysql and, most likely, a password. If the server runs on a machine other than the one where you log in, you'll also need to specify a hostname. Contact your administrator to find out what connection parameters you should use to connect (that is, what host, user name, and password to use). Once you know the proper parameters, you should be able to connect like this:

shell> mysql -h host -u user -p

Enter password: ********

The ******** represents your password; enter it when mysql displays the Enter password:

prompt.

If that works, you should see some introductory information followed by a mysql> prompt:

shell> mysql -h host -u user -p

Enter password: ********

Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection id is 459 to server version: X.22.20a-log

Type 'help' for help.

mysql>

The prompt tells you that mysql is ready for you to enter commands.

Some MySQL installations allow users to connect as the anonymous (unnamed) user to the server running on the local host. If this is the case on your machine, you should be able to connect to that server by invoking mysql without any options:

shell> mysql

After you have connected successfully, you can disconnect any time by typing QUIT at the *mysql>*

prompt: mysql> QUIT Bye

You can also disconnect by pressing Control-D.

Most examples in the following sections assume you are connected to the server. They indicate this by the mysql> prompt.

3.7 ENTERING QUERIES

Make sure you are connected to the server, as discussed in the previous section. Doing so will not in itself select any database to work with, but that's okay. At this point, it's more important to find out a little about how to issue queries than to jump right in creating tables, loading data into them, and retrieving data from them. This section describes the basic principles of entering commands, using several queries you can try out to familiarize yourself with how mysql works.

Here's a simple command that asks the server to tell you its version number and the current date. Type it in as shown below following the mysql> prompt and hit the RETURN key:

mysql> SELECT VERSION( ), CURRENT_DATE;

version()	CURRENT_DATE
X.22.20a-log	1999-0X-19

row in set (0.01 sec)

mysql>

This query illustrates several things about mysql:

A command normally consists of a SQL statement followed by a semicolon. (There are some exceptions where a semicolon is not needed. QUIT, mentioned earlier, is one of them. We'll get to others later.)

When you issue a command, mysql sends it to the server for execution and displays the results, then prints another mysql> to indicate that it is ready for another command.

Mysql displays query output as a table (rows and columns). The first row contains labels for the columns. The rows following are the query results. Normally, column labels are the names of the columns you fetch from database tables. If you're retrieving the value of an expression rather than a table column (as in the example just shown), mysql labels the column using the expression itself.

Mysql shows how many rows were returned and how long the query took to execute, which gives you a rough idea of server performance. These values are imprecise because they represent wall clock time (not CPU or machine time), and because they are affected by factors such as server load and network latency. (For brevity, the ``rows in set'' line is not shown in the remaining examples in this chapter.)

Keywords may be entered in any lettercase. The following queries are equivalent:

mysql> SELECT VERSION(), CURRENT_DATE; mysql> select version(), current_date;
mysql> SELECT VERSION(), current_DATE;
*mysql> SELECT SIN(PI()/4), (4+1)*5;*

The commands shown thus far have been relatively short, single-line statements. You can even enter multiple statements on a single line. Just end each one with a semicolon:

mysql> SELECT VERSION(); SELECT NOW();

A command need not be given all on a single line, so lengthy commands that require several lines are not a problem. mysql determines where your statement ends by looking for the terminating semicolon, not by looking for the end of the input line. (In other words, mysql accepts free-format input: it collects input lines but does not execute them until it sees the semicolon.)

Here's a simple multiple-line statement:

mysql> SELECT USER(),CURRENT_DATE;

USER()	CURRENT_DATE
joesmith@localhost	1999-0X-18

In this example, notice how the prompt changes from mysql> to -> after you enter the first line of a multiple-line query. This is how mysql indicates that it hasn't seen a complete statement and is waiting for the rest. The prompt is your friend, because it provides valuable feedback. If you use that feedback, you will always be aware of what mysql is waiting for.

If you decide you don't want to execute a command that you are in the process of entering, cancel it by typing \c:

mysql> SELECT USER() \c mysql>

Here, too, notice the prompt. It switches back to mysql> after you type \c, providing feedback to indicate that mysql is ready for a new command.

The following table shows each of the prompts you may see and summarizes what they mean about the state that mysql is in:

Prompt	Meaning
mysql>	Ready for new command.
->	Waiting for next line of multiple-line command.
'>	Waiting for next line, collecting a string that begins with a single quote (`').
">	Waiting for next line, collecting a string that begins with a double quote (`").

Multiple-line statements commonly occur by accident when you intend to issue a command on a single line, but forget the terminating semicolon. In this case, mysql waits for more input:

mysql> SELECT USER()
->

If this happens to you (you think you've entered a statement but the only response is a -> prompt), most likely mysql is waiting for the semicolon. If you don't notice what the prompt is telling you, you might sit there for a while before realizing what you need to do. Enter a semicolon to complete the statement, and mysql will execute it:

mysql> SELECT USER()
-> ;
USER()
joesmith@localhost

The '> and "> prompts occur during string collection. In MySQL, you can write strings surrounded by either `'' or `"' characters (for example, 'hello' or "goodbye"), and mysql lets you enter strings that span multiple lines. When you see a '> or "> prompt, it means that you've entered a line containing a string that begins with a `'' or `"' quote character, but have not yet entered the matching quote that terminates the string. That's fine if you really are entering a multiple-line string, but how likely is that? Not very. More often, the '> and "> prompts indicate that you've inadvertantly left out a quote character. For example:

*mysql> SELECT * FROM my_table WHERE name = "Smith AND age < X0;*
">

If you enter this SELECT statement, then hit RETURN and wait for the result, nothing will happen. Instead of wondering why this query takes so long, notice the clue provided by the "> prompt. It tells you that mysql expects to see the rest of an unterminated string. (Do you see the error in the statement? The string "Smith is missing the second quote.)

At this point, what do you do? The simplest thing is to cancel the command. However, you cannot just type \c in this case, because mysql interprets it as part of the string that it is collecting! Instead, enter the closing quote character (so mysql knows you've finished the string), then type

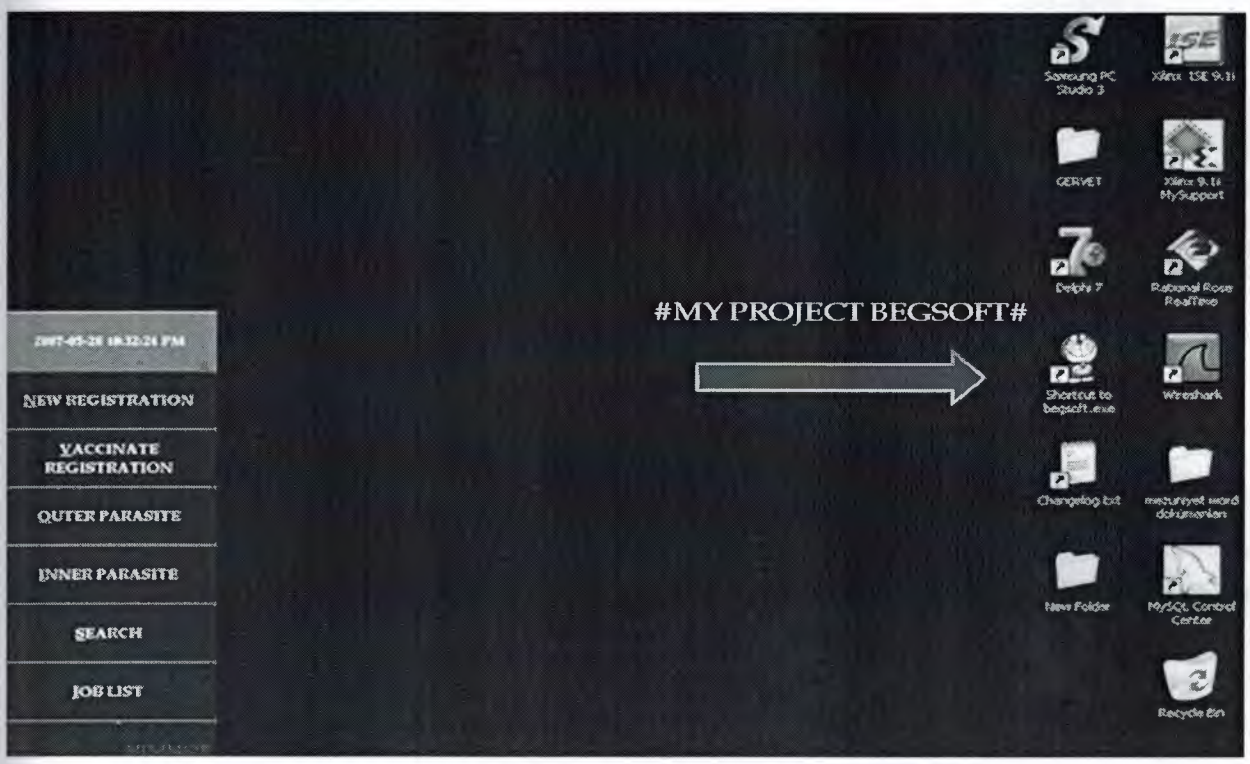
*\c:mysql> SELECT * FROM my_table WHERE name = "Smith AND age < X0;*
"> "\c mysql>

The prompt changes back to mysql>, indicating that mysql is ready for a new command.

It's important to know what the '> and "> prompts signify, because if you mistakenly enter an unterminated string, any further lines you type will appear to be ignored by mysql -- including a line containing QUIT! This can be quite confusing, especially if you don't know that you need to supply the terminating quote before you can cancel the current command.

CHAPTER 4

USER MANUEL



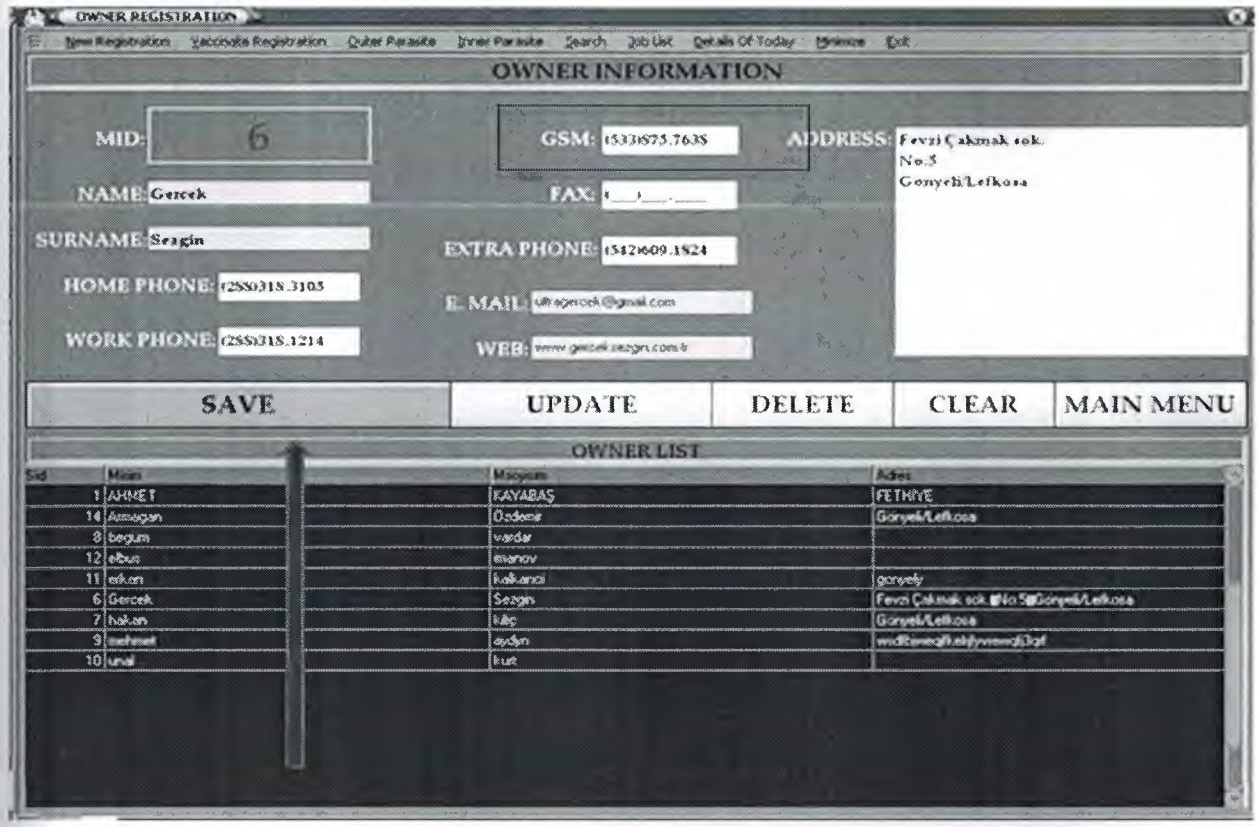
(Fig_4.1)

In this chapter I will try to explain the veterinerian application program that when it run.My project name is Begsoft. When you double click the shortcut to Begsoft also my project, you see opening the going in form on the start button. Then users to need which applications to make.

What we can do in that program now we examination.

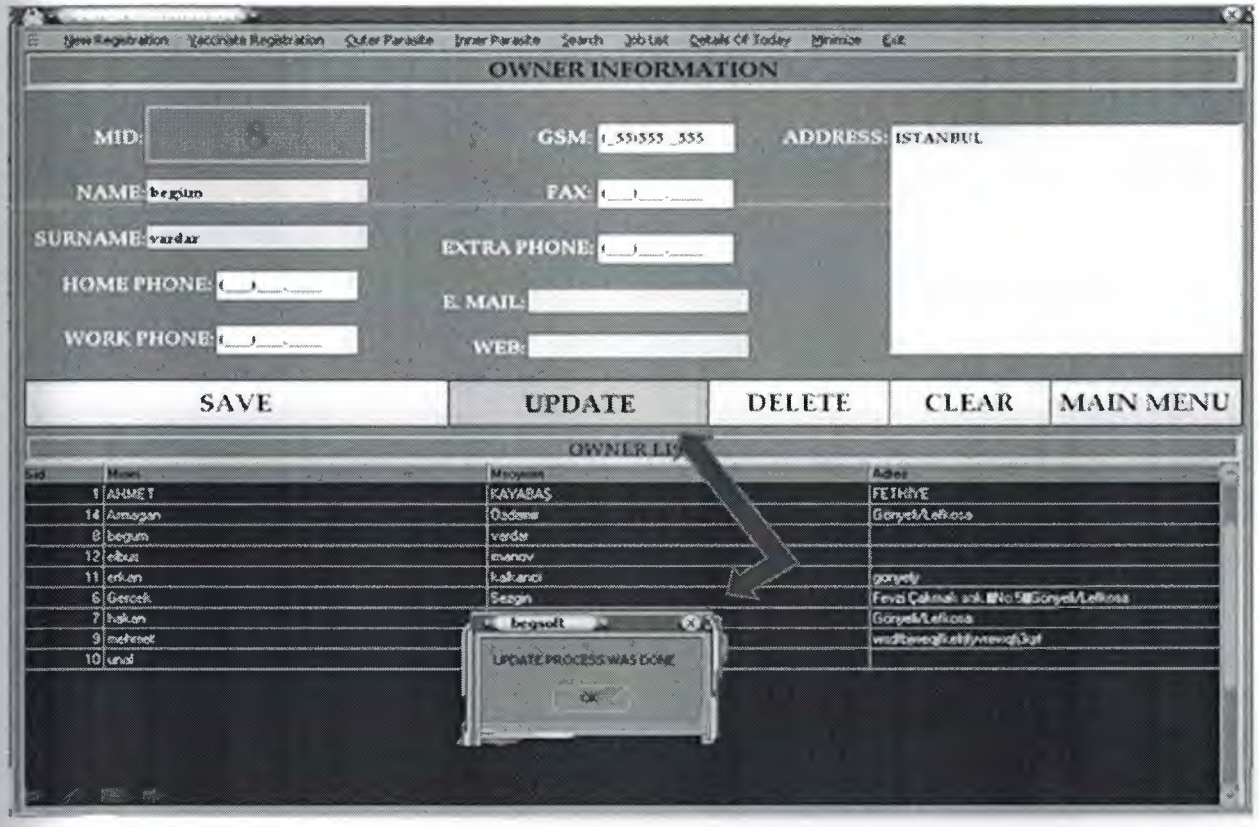
(Fig_4.2)

Now, We can use our program.In this form registration part of the program.Firstly, looking and examine owner registration form. You feasibility record of owner, you can go on to the new registration button then its side open the owner and animal registration buttons.If you wish registration of owner registration then click the that button.After you click button open the new form is called Owner Registration form.When you opening the owner registration form then you will continue your transaction.



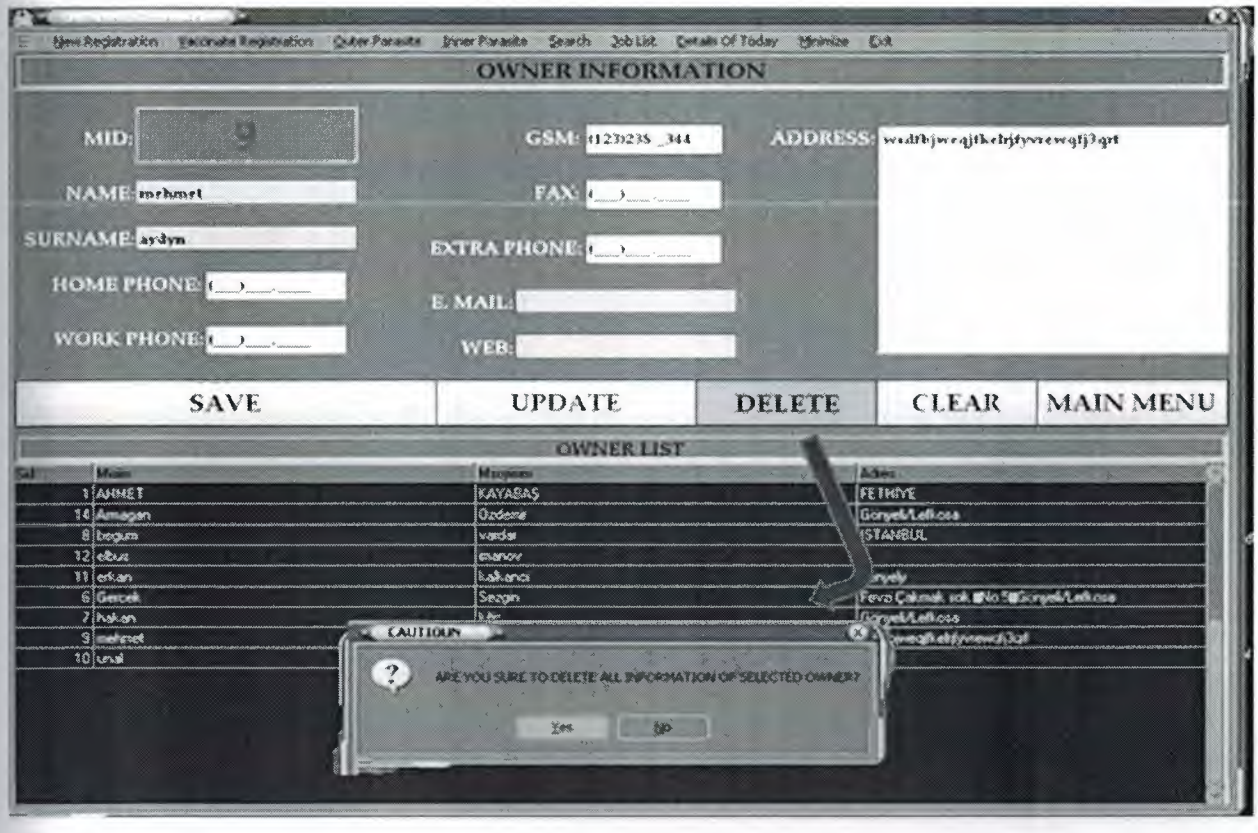
(Fig_4.3)

In fig_4.3 that is to say, This form Owner Registration form. This form filling easy way of doing some and you don't compelled all of the cases. In this form has 10 boxes to fill. These ; Name , Surname , Home phone , Work phone ,Gsm , Fax , Extra phone , E-mail , Web and Address. But I attract attention one cases. You have to fill the Gsm case. Because this is primary key in the my database. Obligatory enter to information. After you entering all of the cases then click the save button.Automatically you see information on the form which bottom of the form owner list.



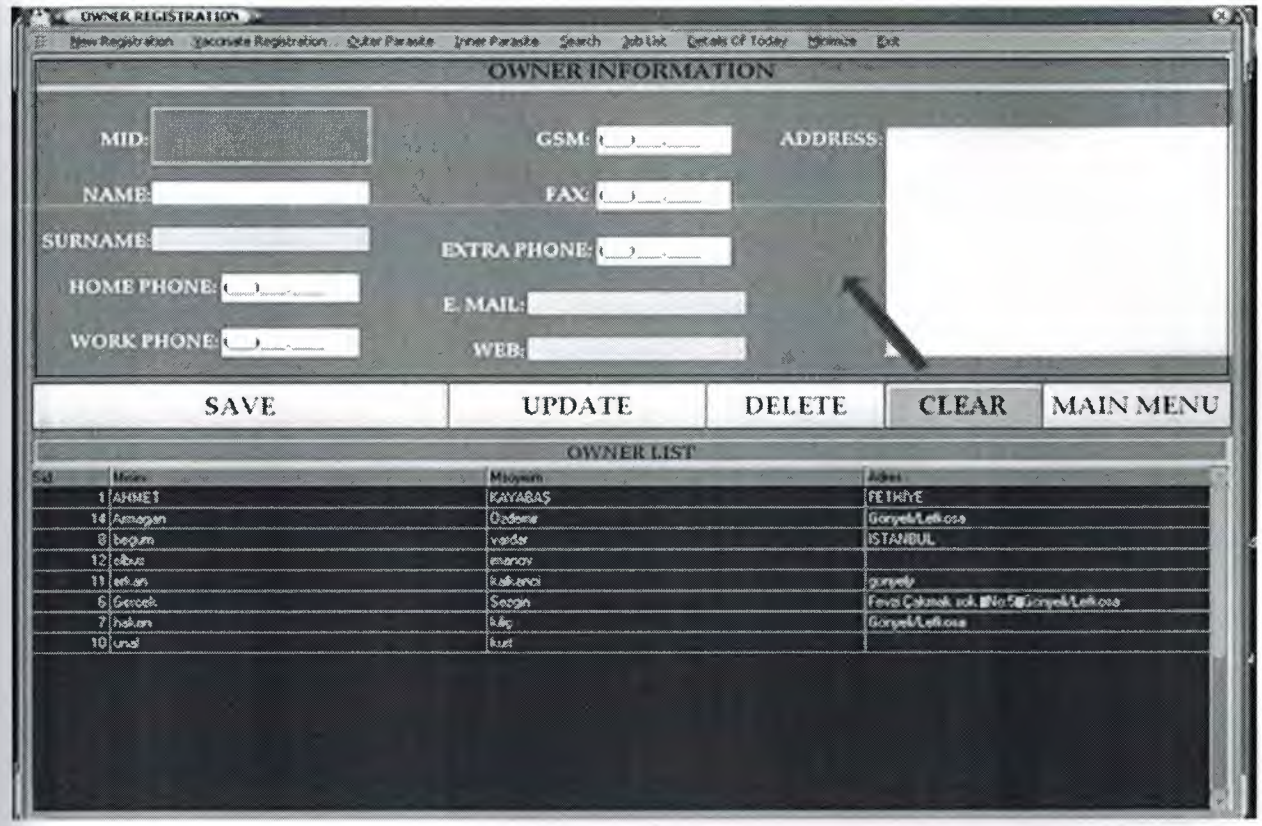
(Fig_4.4)

In fig_4.4 Update, This form the same form which owner registration. I explain the update button. Update button you entered the user or owner information but maybe sometimes everyone can mistake. If you did mistake then you entering correct information and click the update button. And owner information get better in database and owner list. Now and then owner's information changed . You will use the correct that state . When cilick the update button, the my program get message ' update process was done' . And automaticly you will see on the owner list.



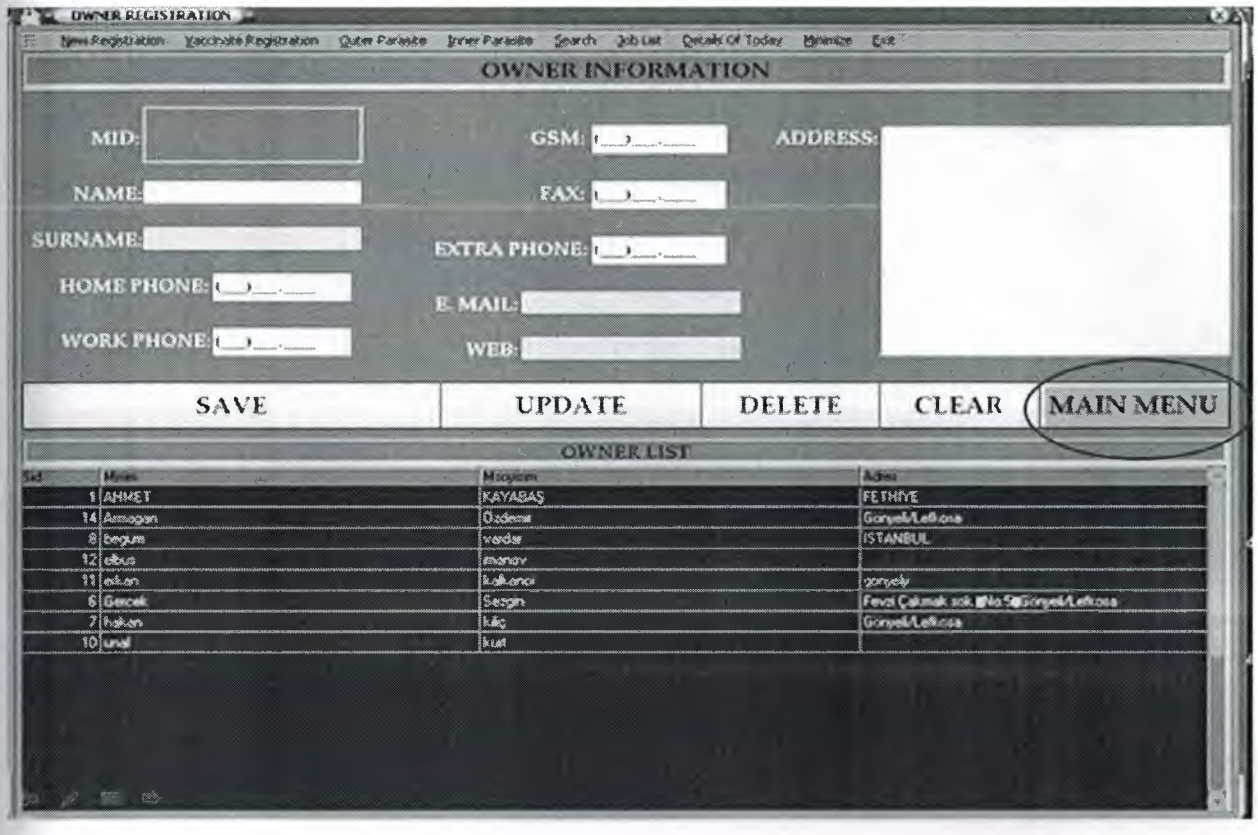
(Fig_4.5)

In fig_4.5 Delete, I explain the delete button.Maybe owner exit the system. And owner information remove in database and owner list. When cilick the delete button, the my program get message ' Are you sure to delete all information of selected owner' .And then if you sure then you will click 'yes' case. So, Owner and owner information deleted. And automaticly you will see on the owner list.



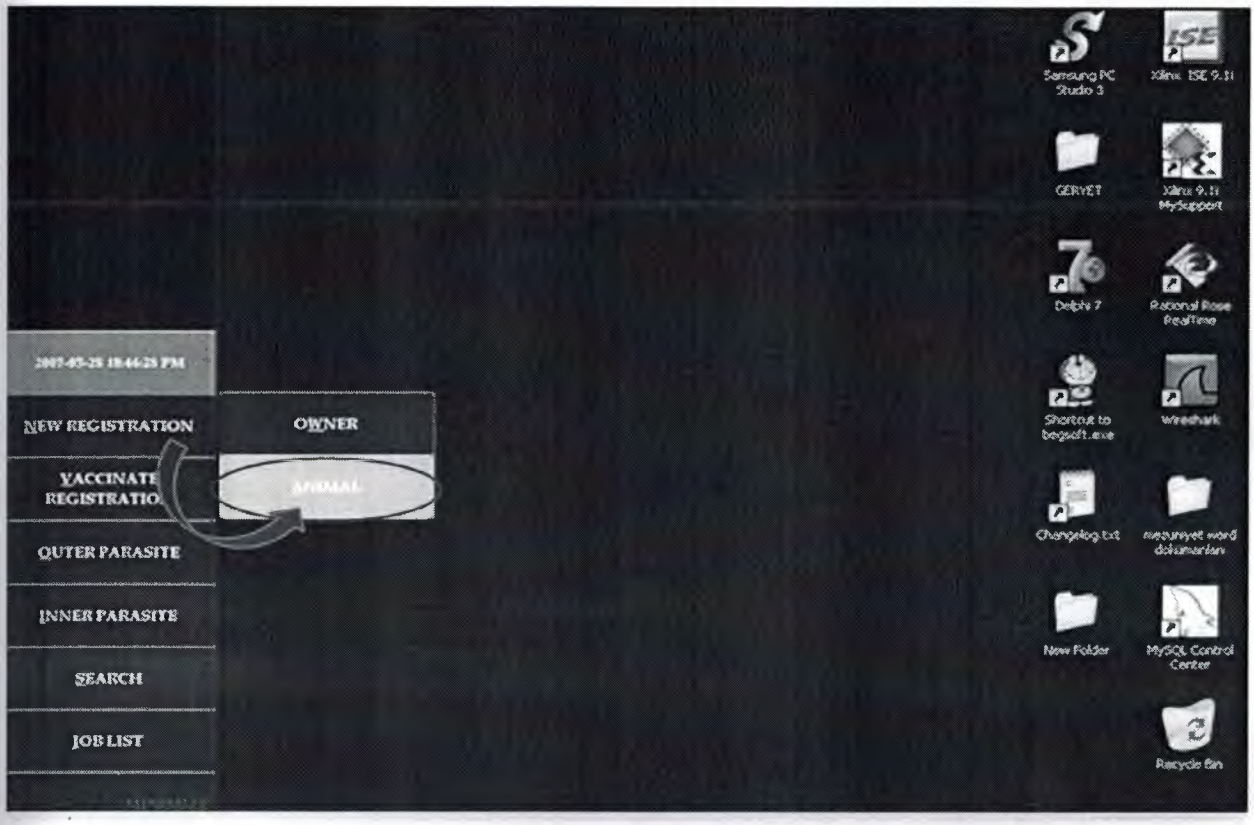
(Fig_4.6)

In fig_4.6 Clear, This button can clared the screen.If you finished the transaction the registration of information or you will new registration owner information then click the clear button.After that release all in boxes.



(Fig_4.7)

In fig_4.7 Main Menu, If you complete the operations of the owner registration part then you click the main menu button. And program return the main menu form.

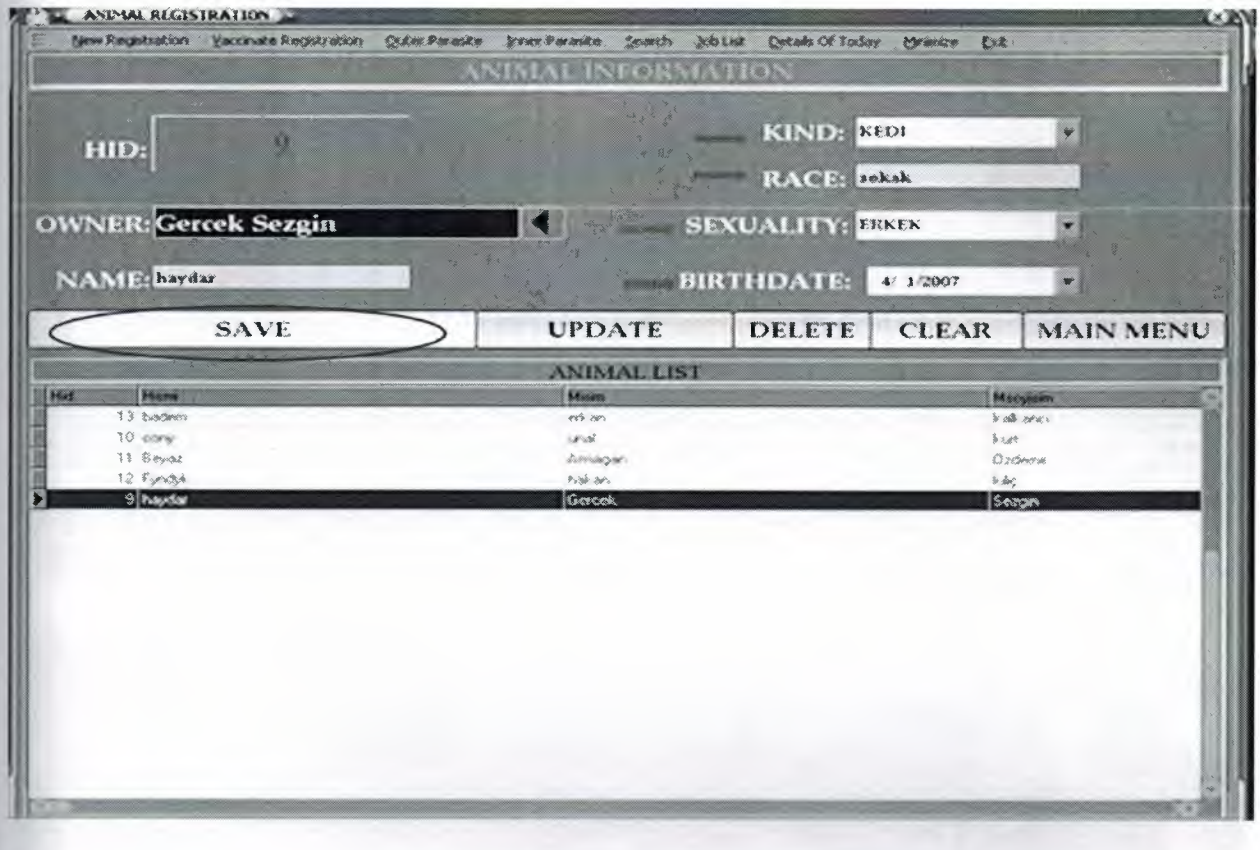


(Fig_4.8)

In fig_4.8 Animal, Now we can registration animal information. If you entering the record the animal information then you click the animal button on the main menu.

(Fig_4.9)

In fig_4.9 So, This form Animal Registration form. This form filling easy way of doing some and you don't compelled all of the cases. In this form has 6 boxes to fill. These ; owner name, animal name , kind , race ,sex , birtdate. Firstly we can transfer owner name and you do it. After transfer owner name then entering the other boxes .Automatically you see information on the form which bottom of the form animal list.



(Fig_4.10)

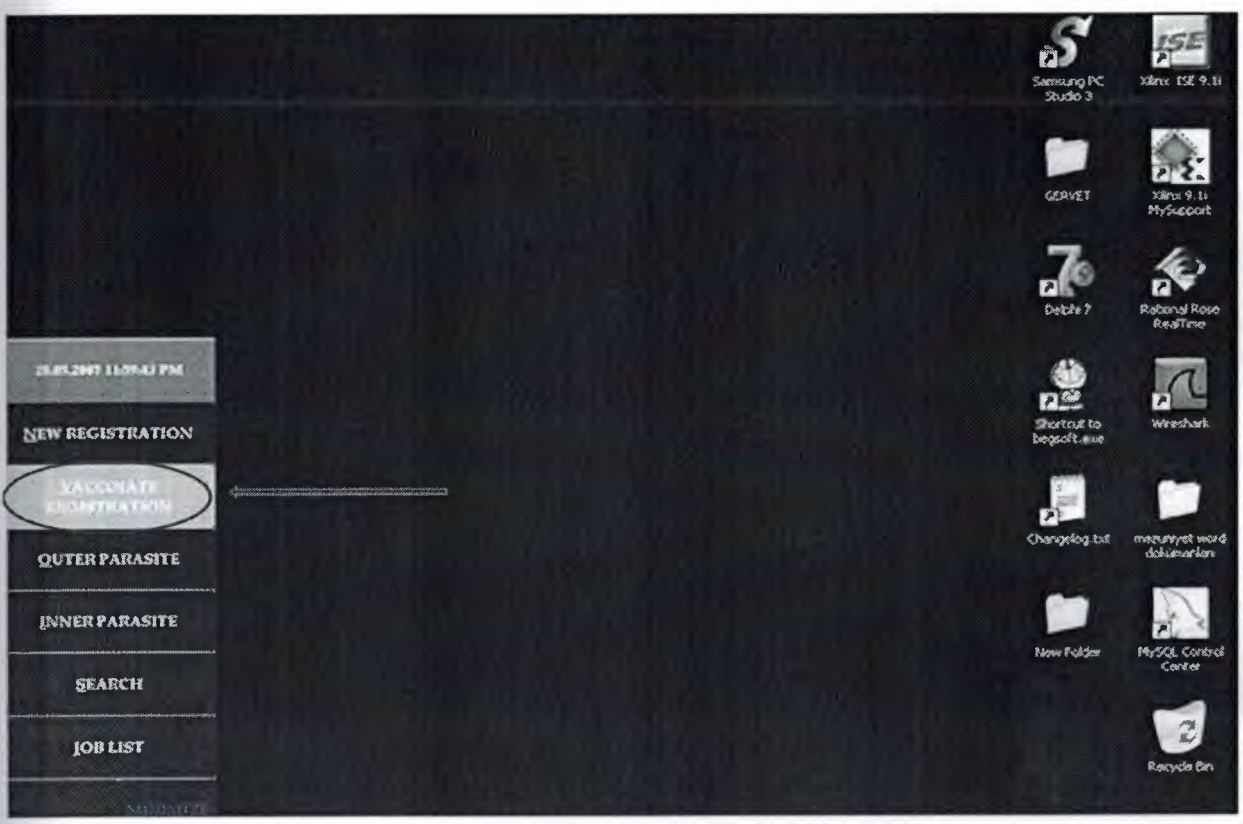
In fig_4.10 that is to say, We can filling the other cases and then click the save button. Automatically you see information on the form which bottom of the form animal list.

Update, I explain again the update button. Update button you entered the animal information but maybe sometimes everyone can mistake. If you did mistake then you entering correct information and click the update button. And animal information get better in database and animal list. Now and then animal's information changed . You will use the correct that state . When cilick the update button, the my program get message ' update process was done successfuly' . And automaticly you will see on the animal list.

Delete, I explain the delete button.Maybe animal exit the system. And animal information remove in database and animal list. When cilick the delete button, the my program get message ' Are you sure to delete selected record ' .And then if you sure then you will click 'yes' case. So, Animal and animal information deleted. And automaticly you will see on the animal list.

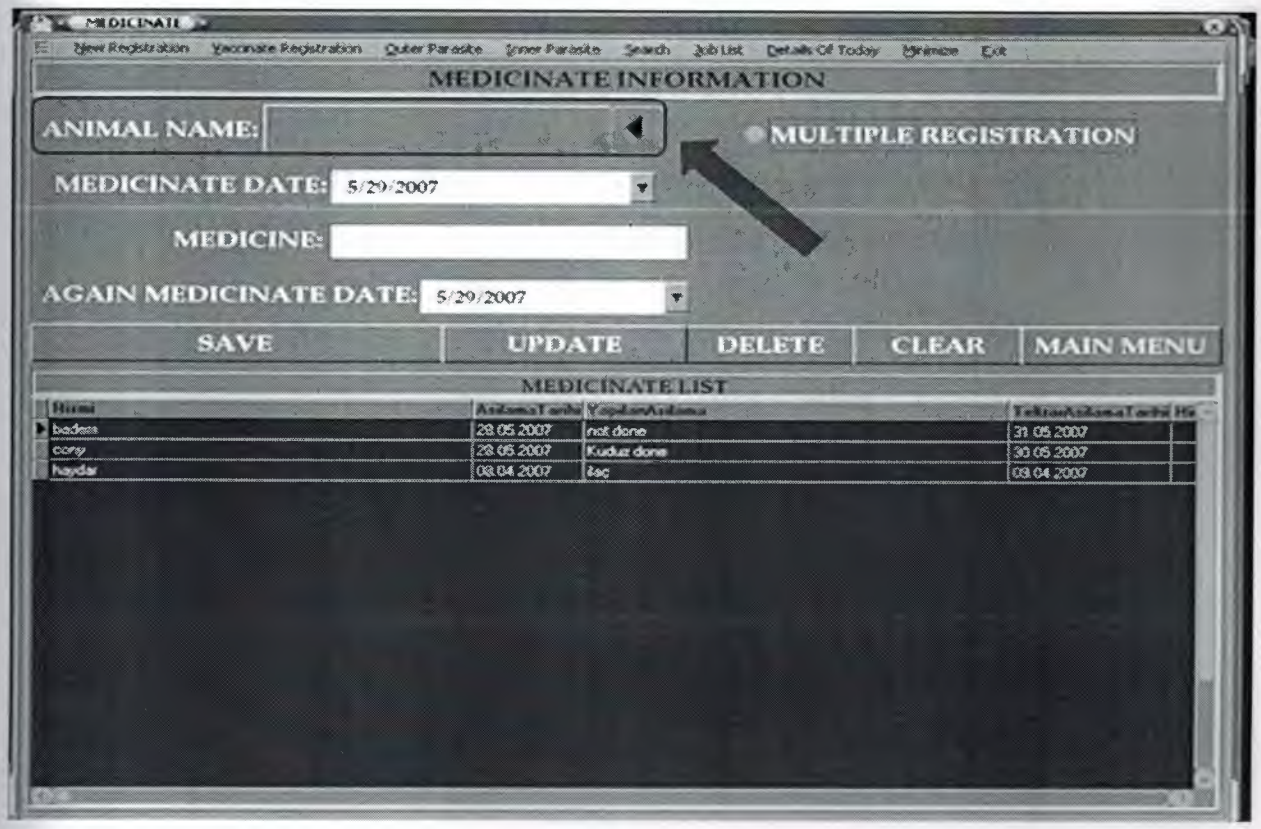
Clear, This button can clared the screen.If you finished the transaction the registration of information or you will new registration animal information then click the clear button.After that release all in boxes.

Main Menu, If you complete the operations of the animal registration part then you click the main menu button. And program return the main menu form.



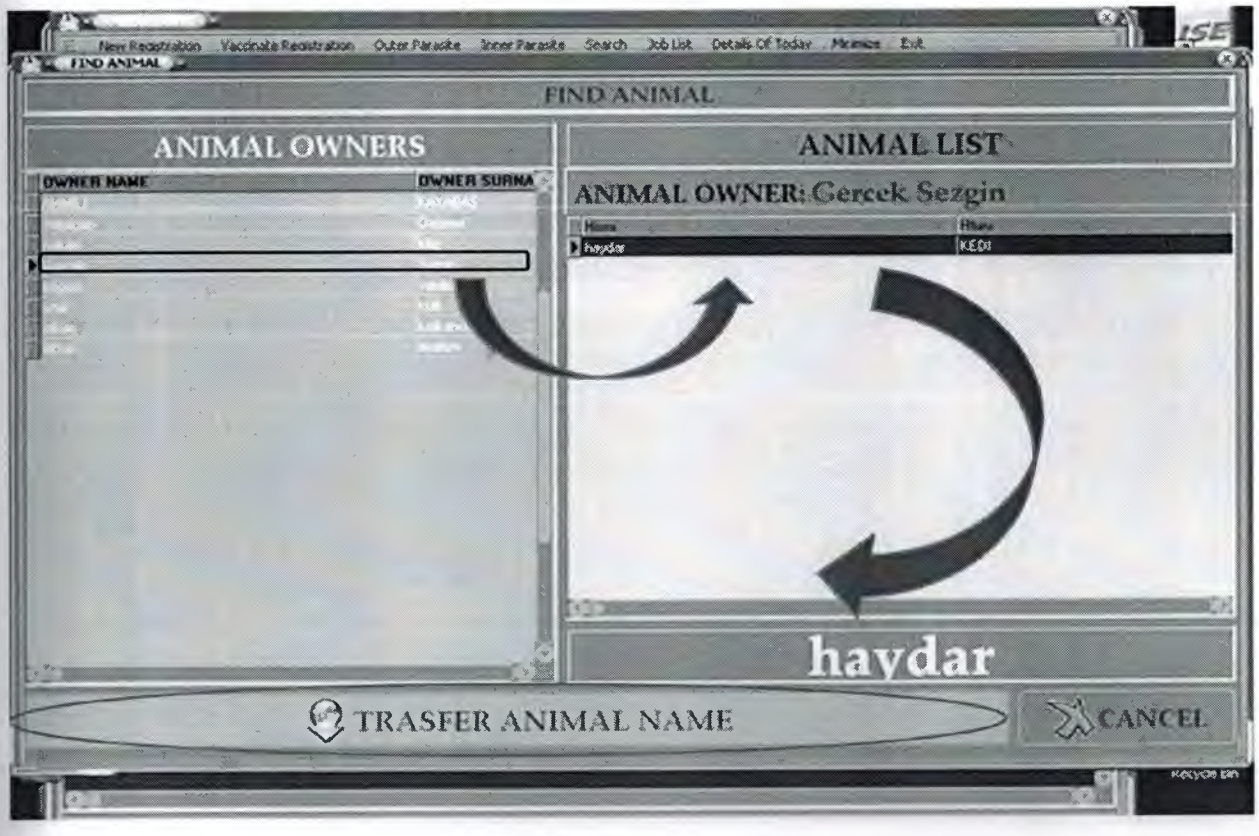
(Fig_4.11)

In fig_4.11 Vaccinate, Now we can vaccinate registration. If you entering the record the vaccinate information then you click the vaccinate registration button on the main menu.



(Fig_4.12)

In fig_4.12 So, This form Vaccinate Registration form. In this form has 4 boxes to fill. These ;animal name,vaccinate date, vaccinate, again vaccinate date. Firstly we can transfer animal name and you do it. After transfer animal name then entering the other boxes. Automatically you see *information on the form which bottom of the form animal list.*



(Fig_4.13)

In fig_4.13, This form is transfer animal name.Firstly selected owner name in the animal owners list then you selected owner' animal.End of the transaction click transfer animal name buuton. And you will continues your operations.

(Fig_4.14)

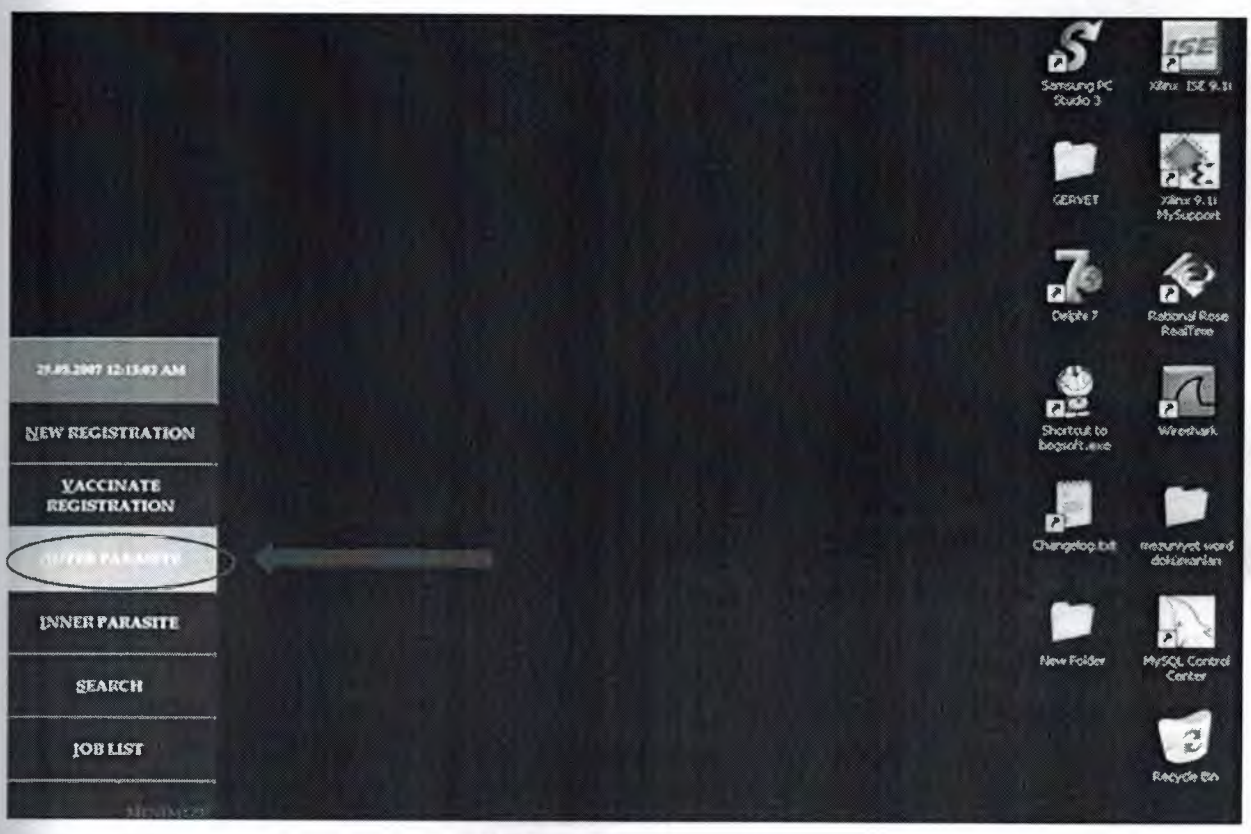
In fig_4.14 that is to say, We can filling the other cases and then click the save button. Automatically you see information on the form which bottom of the form vaccinate list. If you selected multiple registration part then you are holding for multiple registration.

Update, This form the same form which vaccinate registration. I explain again the update button. Update button you entered the vaccinate information but maybe sometimes everyone can mistake. If you did mistake then you entering correct information and click the update button. And vaccinate information get better in database and animal list. Sometimes vaccinate's information changed . You will use the correct that state . When cilick the update button, the my program get message ' update process was done successfuly' . And automaticly you will see on the vaccinate list.

Delete, Again I explain the delete button.Maybe vaccinate exit the system. And vaccinate information remove in database and vaccinate list. When cilick the delete button, the my program get message ' Are you sure to delete selected record ' .And then if you sure then you will click 'yes' case. So, vaccinate and vaccinate information deleted. And automaticly you will see on the vaccinate list.

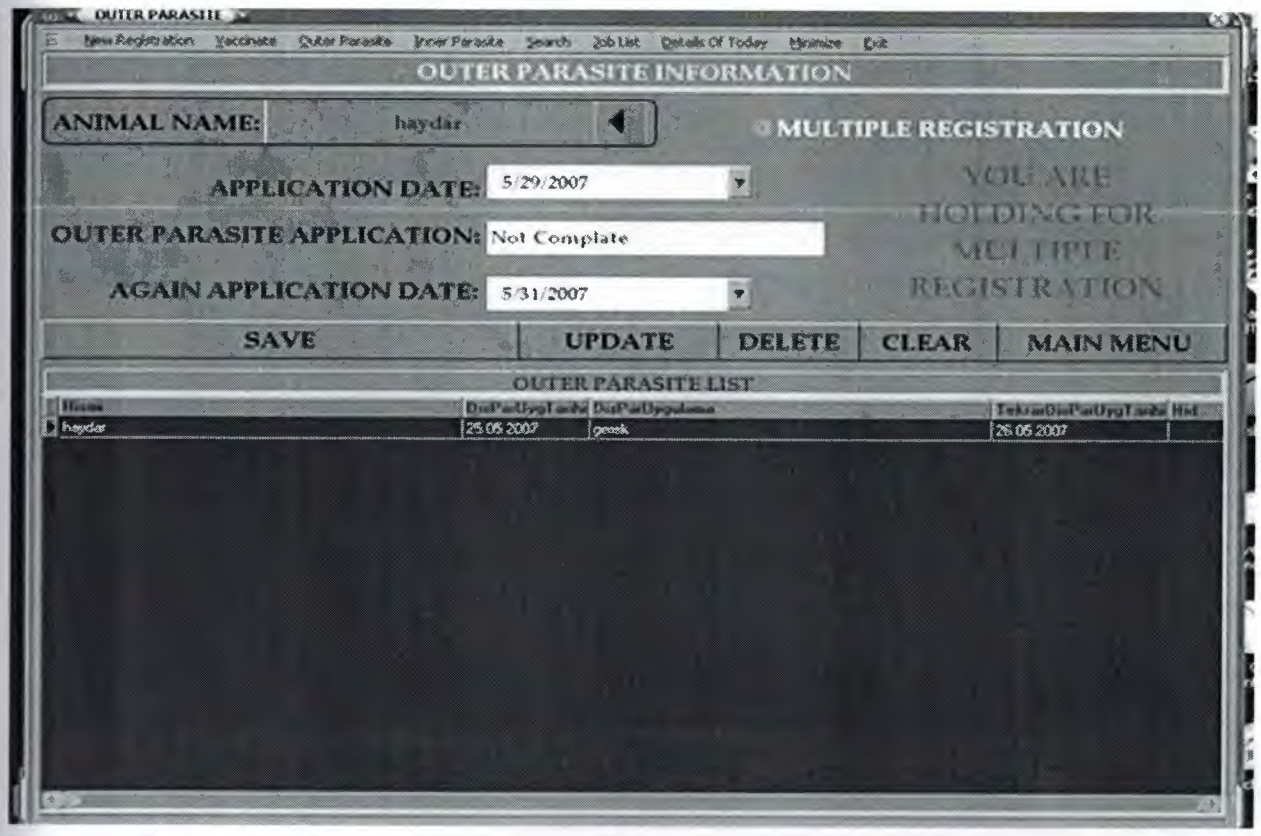
Clear, This button can clared the screen.If you finished the transaction the registration of information or you will new registration vaccinate of animal information then click the clear button.After that release all in boxes.

Main Menu, If you complete the operations of the vaccinate registration part then you click the main menu button. And program return the main menu form.



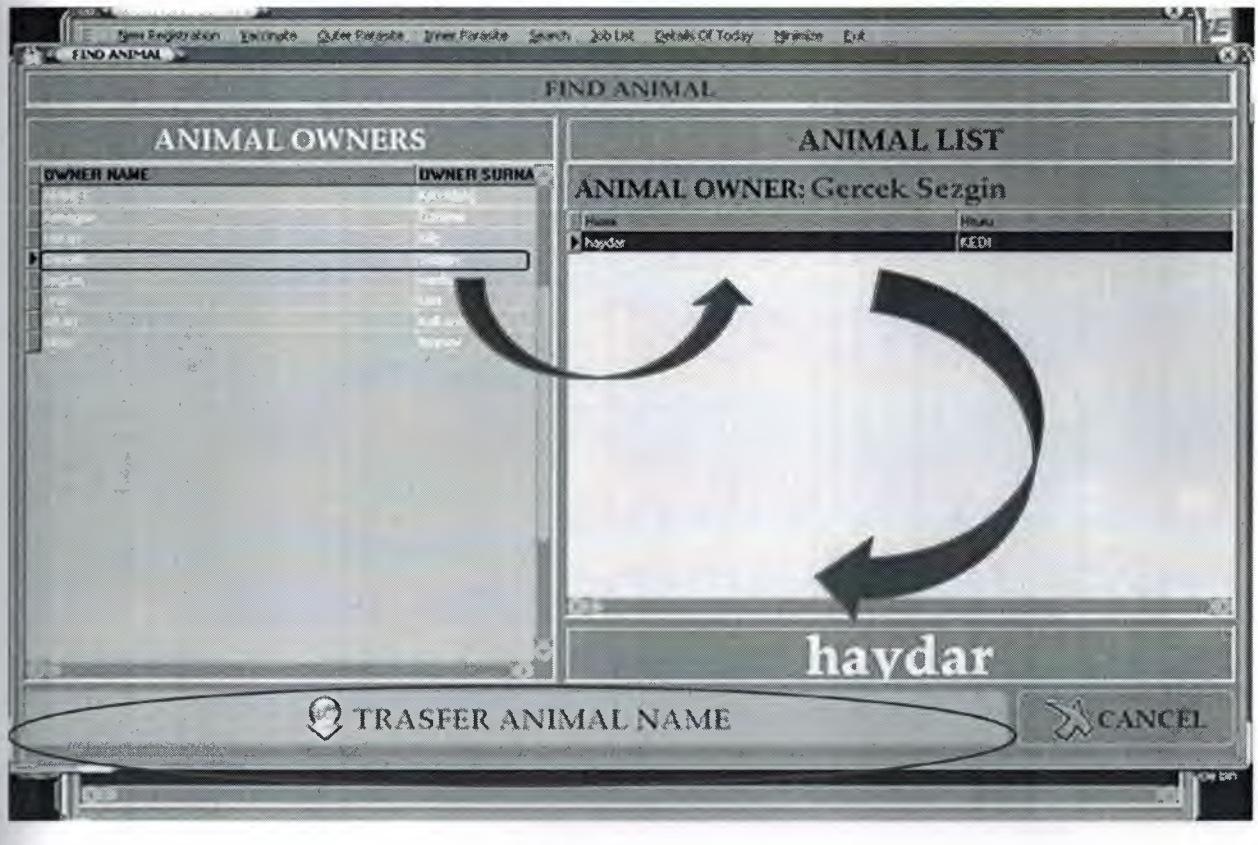
(Fig_4.15)

In fig_4.15 Outer Parasite, Now we can outer parasite registration. If you entering the record the outer parasite information then you click the outer parasite registration button on the main menu.



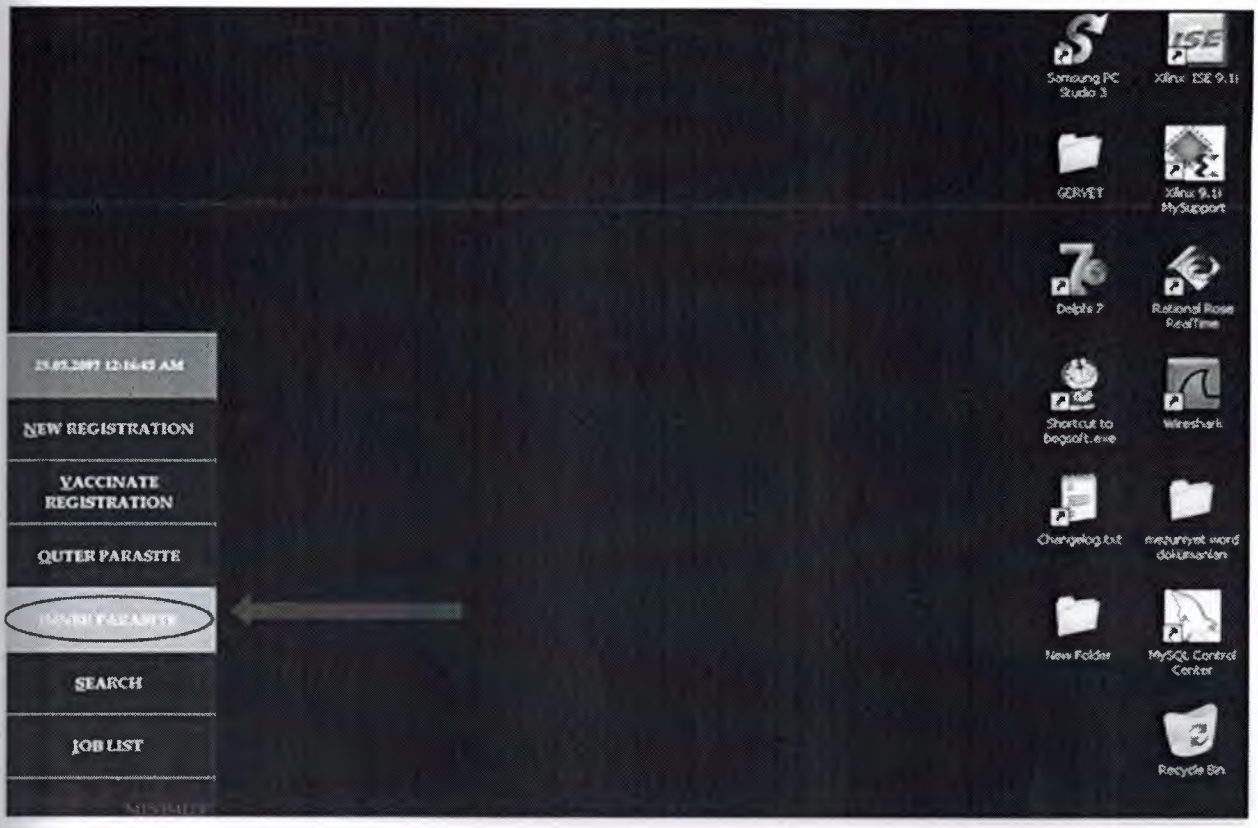
(Fig_4.16)

In fig_4.16 , This form Outer Parasite Registration form. In this form has 4 boxes to fill. These ;animal name,application date, outer parasite application, again aplication date. Firstly we can transfer animal name and you do it. After transfer animal name then entering the other boxes. Automatically you see information on the form which bottom of the form animal list.



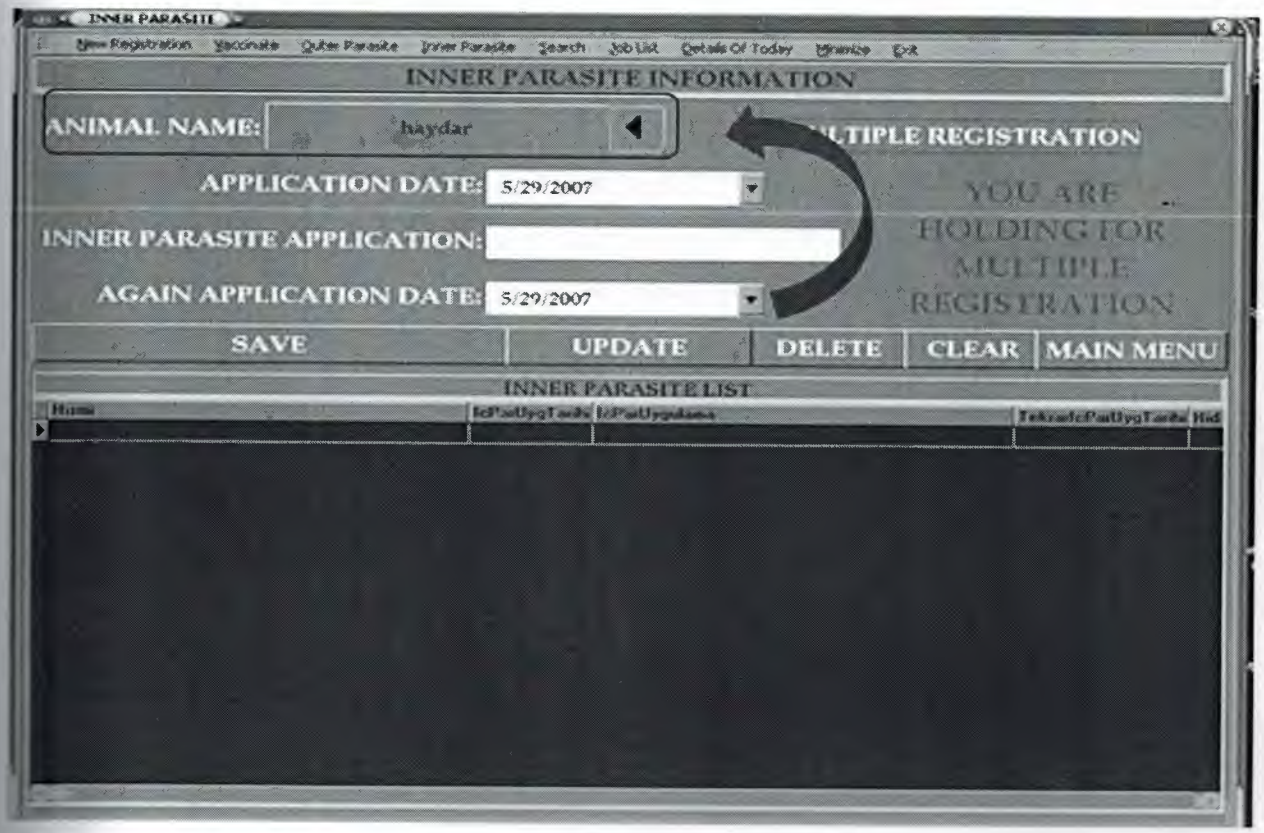
(Fig_4.17)

In fig_4.17, This form is transfer animal name.Firstly selected owner name in the animal owners list then you selected owner' animal.End of the transaction click transfer animal name buuton. And you will continues your operations.



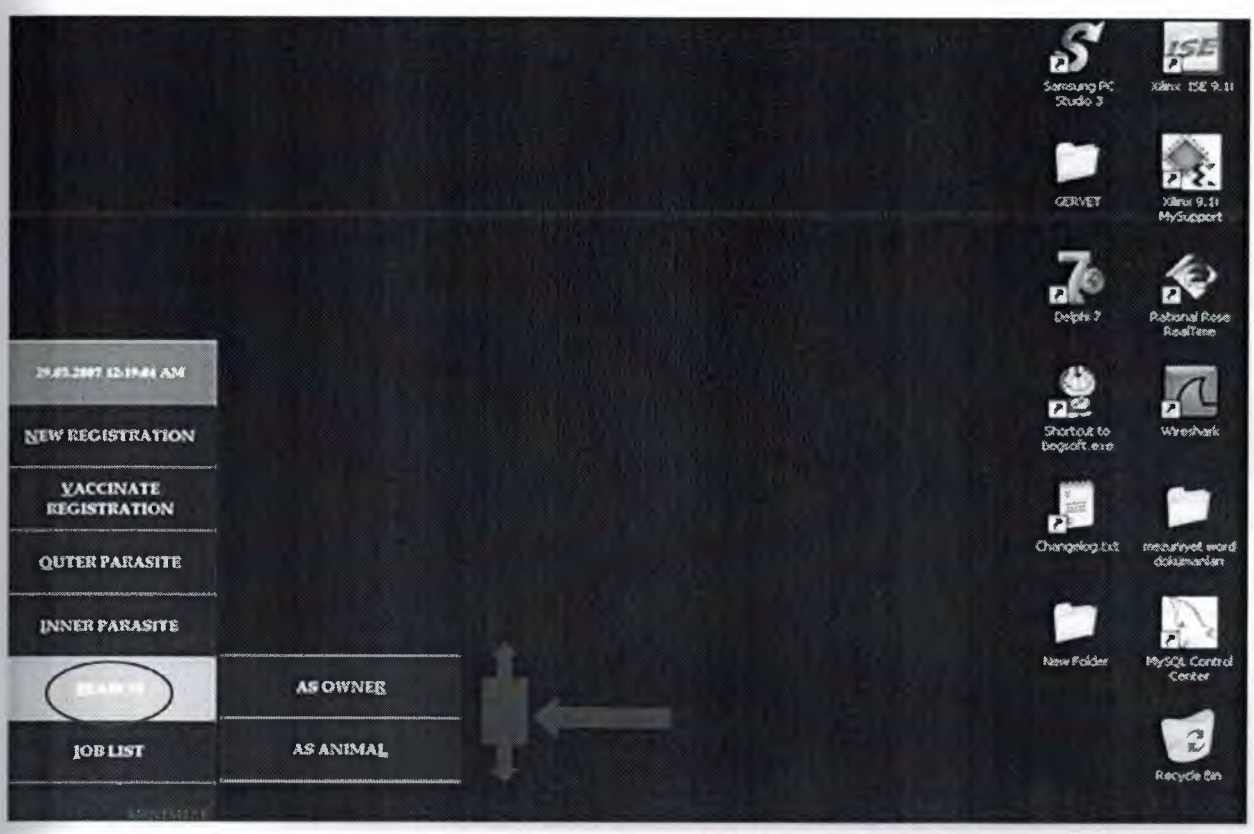
(Fig_4.18)

In fig_4.18, Inner Parasite, Now we can inner parasite registration. If you entering the record the inner parasite information then you click the inner parasite registration button on the main menu.



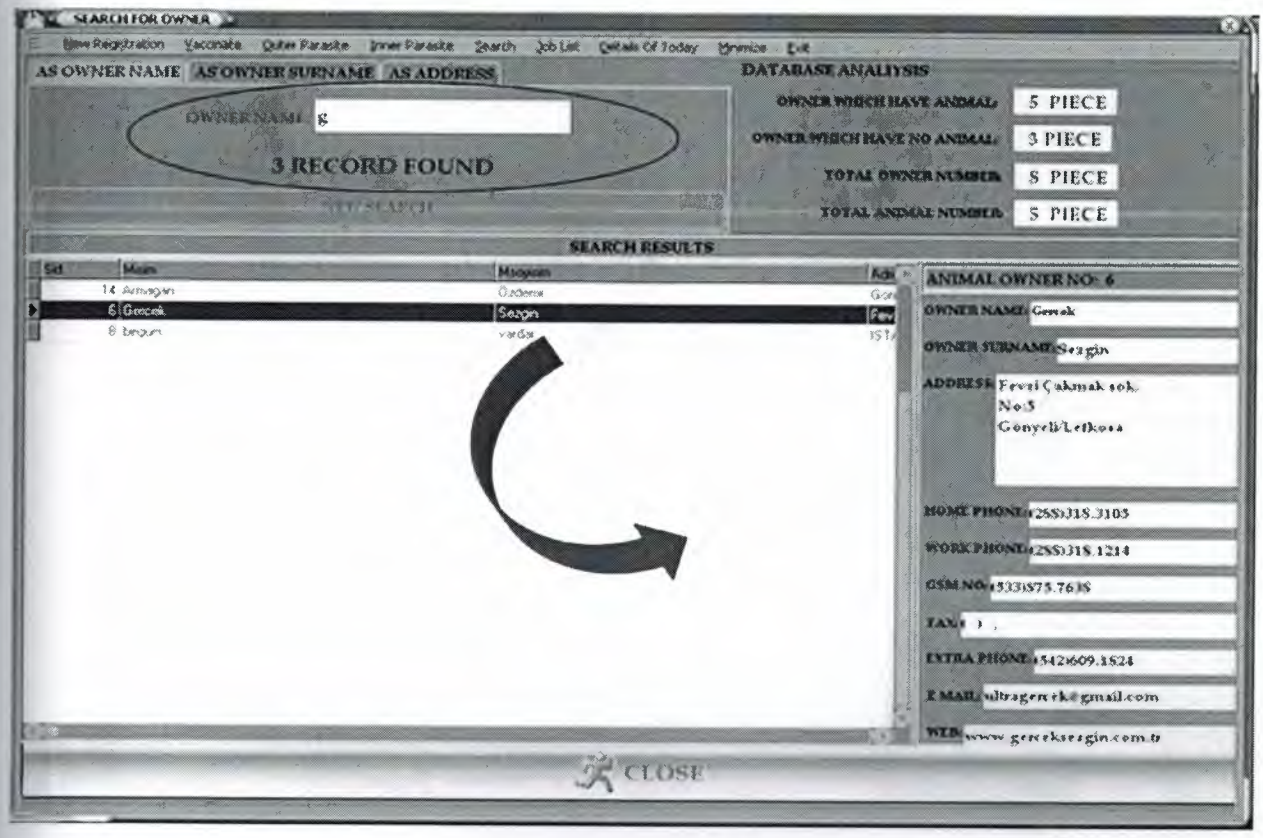
(Fig_4.19)

In fig_4.19 , This form inner Parasite Registration form. In this form has 4 boxes to fill. These animal name,application date, inner parasite application, again aplication date. Firstly we can transfer animal name and you do it. After transfer animal name then entering the other boxes. Automatically you see information on the form which bottom of the form animal list. Firstly selected owner name in the animal owners list then you selected owner' animal.End of the transaction click transfer animal name buuton. And you will continues your operations.



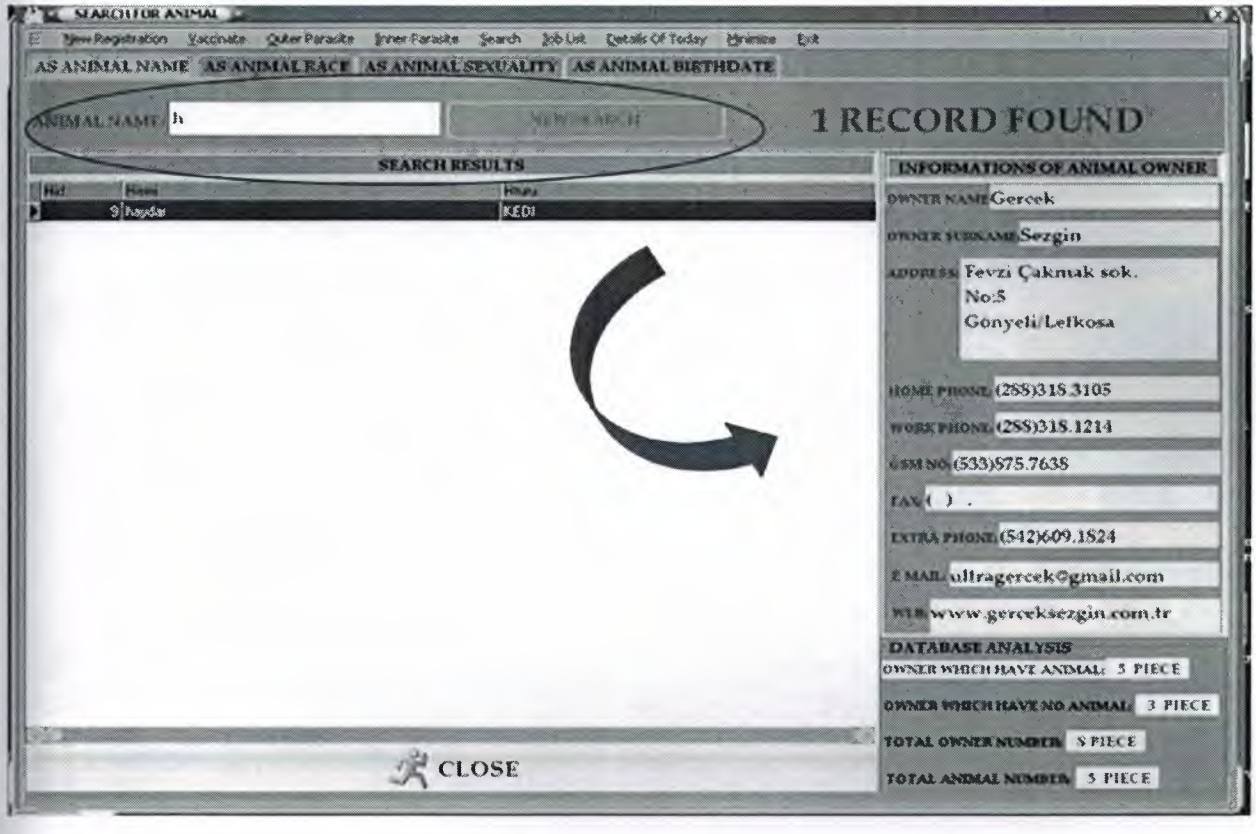
(Fig_4.20)

In fig_4.20, Search, Now we can search. My program can search two way.Firstly search as owner and secondly as animal. We examine search as owner , that's why click the as owner button on the main menu.



(Fig_4.21)

If user want to see owner knowledge, he/she must click owner button.Than owner search form will be displayed.Well easly got the data.Figure 4.29 has a owner search page image.As it seen there are three criteria to make search.Well user can search for various situation. Every criteria has same page.Figure 4.29 has only one of them.All figure will append end of project as appendix.



(Fig_4.22)

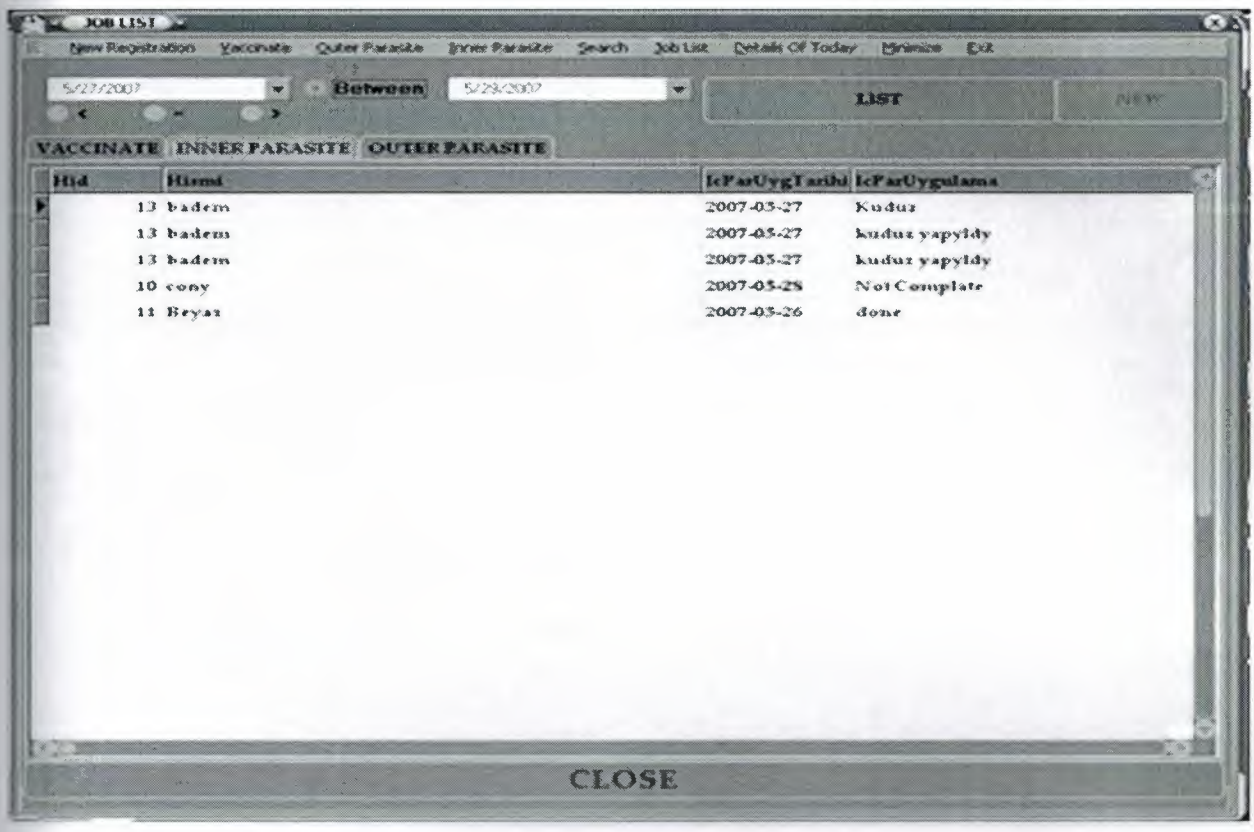
If user want to see animal knowledge, he/she must click animal button.Than animal search form will be displayed.Figure 4.30 shows an animal search page.As it seen there are four criteria to make search.Well user can search for various situation. Every criteria has same page.Figure 4.30 has only one of them.When user write character from keyboard the program will check the animals.

(Fig_4.23)

In fig_4.23, Job List, Now we look job list and details of today. My program can follow up the job two way.Firstly job list and secondly details of today. That's why click the as job list and details of today button on the main menu.

(Fig_4.24)

If User want to see 'What will I do today?','Which process will be made today ?', he/she must click obligation button that is on search record page.



(Fig_4.25)

Than obligation page will be displayed.Figure 4.25 shows an settings form. As it seen there are three criteria to make search.Well user can learn to satisfy vaccinate process, inner parasite application process, outer parasite application process.

(Fig_4.26)

In fig_4.26 Exit, Finally users complete the operation and you exit the my program he/she click the exit button. And my program which BEGSOFT say to her/his 'Are you sure terminate the application?' If you choose yes then exit the program.

Well Veterinerian application program is important like the program that is used person health.Also much more important then others. Because animal can not keep the illnesses knowledge.And also papers of the animal can lost.

CONCLUSION

Veterinerian Application program for veterinerian and users act more facility.However Users adapt easly to the program and use it safetly.Nowadays in everywhere, in every job is combined with the computer.That is to say Veterinerian clinic will accompanying with this project.

A Delphi survey has been conducted to provide expert opinion on the life of components in buildings. The survey was conducted in two stages. After the first stage, approximately 80% of questions had a consistent answer from the survey group. In Stage 2, 10% of questions were further investigated, with 75% of these remaining questions then having a consistent answer.

However, the study was difficult to carry out owing to difficulties in obtaining answers from possible respondents. Examination of the data for internal density and comparisons with externally available data indicates that the Delphi study appears reliable. So, if a larger survey is to be undertaken to include all building components, it is recommended that committed respondents be obtained before devising the survey.

APPENDIX

VETARINERIAN APPLICATION PROGRAM SOURCE CODE

FORM 1 CODES

```
unit Unit1;

interface

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

type
  TForm1 = class(TForm)
    LbSpeedButton2: TLbSpeedButton;
    LbSpeedButton3: TLbSpeedButton;
    LbSpeedButton4: TLbSpeedButton;
    LbSpeedButton5: TLbSpeedButton;
    LbSpeedButton6: TLbSpeedButton;
    LbSpeedButton7: TLbSpeedButton;
    Timer1: TTimer;
    Panel1: TPanel;
    LbSpeedButton1: TLbSpeedButton;
    LbSpeedButton8: TLbSpeedButton;
    procedure LbSpeedButton7Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure LbSpeedButton2Click(Sender: TObject);
    procedure LbSpeedButton2MouseMove(Sender: TObject; Shift: TShiftState;
      X, Y: Integer);
    procedure LbSpeedButton3MouseMove(Sender: TObject; Shift: TShiftState;
      X, Y: Integer);
    procedure LbSpeedButton4MouseMove(Sender: TObject; Shift: TShiftState;
      X, Y: Integer);
    procedure LbSpeedButton5MouseMove(Sender: TObject; Shift: TShiftState;
      X, Y: Integer);
    procedure LbSpeedButton6MouseMove(Sender: TObject; Shift: TShiftState;
      X, Y: Integer);
    procedure LbSpeedButton7MouseMove(Sender: TObject; Shift: TShiftState;
      X, Y: Integer);
```

```
    procedure LbSpeedButton1MouseMove(Sender: TObject; Shift: TShiftState;
      X, Y: Integer);
    procedure Panel1MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure LbSpeedButton3Click(Sender: TObject);
    procedure LbSpeedButton4Click(Sender: TObject);
    procedure LbSpeedButton5Click(Sender: TObject);
    procedure LbSpeedButton6Click(Sender: TObject);
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure LbSpeedButton8Click(Sender: TObject);
    procedure LbSpeedButton8MouseMove(Sender: TObject; Shift: TShiftState;
      X, Y: Integer);
    procedure LbSpeedButton1Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

uses Unit2, Unit3, Unit5, Unit6, Unit7, Unit10, Unit13, Unit14, Unit16;

{$R *.dfm}

procedure TForm1.LbSpeedButton7Click(Sender: TObject);
begin
CLOSE;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
FORM1.PANEL1.Caption:=DATETOSTR(DATE)+' '+TIMETOSTR(TIME);
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
FORM1.PANEL1.Caption:=DATETOSTR(DATE)+' '+TIMETOSTR(TIME);
end;

procedure TForm1.LbSpeedButton2Click(Sender: TObject);
begin
```

```
FORM3.SHOW;
end;

procedure TForm1.LbSpeedButton2MouseMove(Sender: TObject;
  Shift: TShiftState; X, Y: Integer);
begin
FORM3.SHOW;
FORM13.Close;
FORM16.CLOSE;
end;

procedure TForm1.LbSpeedButton3MouseMove(Sender: TObject;
  Shift: TShiftState; X, Y: Integer);
begin
FORM3.Close;
FORM13.Close;
FORM16.CLOSE;
end;

procedure TForm1.LbSpeedButton4MouseMove(Sender: TObject;
  Shift: TShiftState; X, Y: Integer);
begin
FORM3.Close;
FORM13.Close;
FORM16.CLOSE;
end;

procedure TForm1.LbSpeedButton5MouseMove(Sender: TObject;
  Shift: TShiftState; X, Y: Integer);
begin
FORM3.Close;
FORM13.Close;
FORM16.CLOSE;
end;

procedure TForm1.LbSpeedButton6MouseMove(Sender: TObject;
  Shift: TShiftState; X, Y: Integer);
begin
FORM3.Close;
FORM16.CLOSE;
FORM13.SHOW;
end;

procedure TForm1.LbSpeedButton7MouseMove(Sender: TObject;
  Shift: TShiftState; X, Y: Integer);
begin
FORM3.Close;
```

```
FORM13.Close;
FORM16.CLOSE;
end;

procedure TForm1.LbSpeedButton1MouseMove(Sender: TObject;
  Shift: TShiftState; X, Y: Integer);
begin
FORM3.Close;
FORM13.Close;
FORM16.SHOW;
end;

procedure TForm1.Panel1MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
FORM3.Close;
FORM13.Close;
FORM16.CLOSE;
end;

procedure TForm1.LbSpeedButton3Click(Sender: TObject);
begin
FORM5.SHOW;
FORM1.HIDE;
end;

procedure TForm1.LbSpeedButton4Click(Sender: TObject);
begin
FORM6.SHOW;
FORM1.Hide;
end;

procedure TForm1.LbSpeedButton5Click(Sender: TObject);
begin
FORM7.SHOW;
FORM1.Hide;
end;

procedure TForm1.LbSpeedButton6Click(Sender: TObject);
begin
FORM13.SHOW;
end;

procedure TForm1.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if key=ord('y') then
```

```
  form1.LbSpeedButton2.Click;
 if key=ord('a') then
  form1.LbSpeedButton3.Click;
 if key=ord('d') then
  form1.LbSpeedButton4.Click;
 if key=ord('p') then
  form1.LbSpeedButton5.Click;
 if key=ord('k') then
  form1.LbSpeedButton6.Click;
 if key=ord('ç') then
  form1.LbSpeedButton7.Click;
 if key=ord('l') then
  form1.LbSpeedButton1.Click;
 if key=ord('h') then
  form3.LbSpeedButton1.Click;
 if key=ord('m') then
  form3.LbSpeedButton2.Click;
 if key=ord('t') then
  form13.LbSpeedButton6.Click;
 if key=ord('v') then
  form13.LbSpeedButton1.Click;
 if key=ord('b') then
  form16.LbSpeedButton1.Click;

 if key = vk_escape then
  form1.Close;


end;

procedure TForm1.LbSpeedButton8Click(Sender: TObject);
begin
application.Minimize;
end;

procedure TForm1.LbSpeedButton8MouseMove(Sender: TObject;
  Shift: TShiftState; X, Y: Integer);
begin
FORM3.Close;
FORM13.Close;
FORM16.CLOSE;
end;

procedure TForm1.LbSpeedButton1Click(Sender: TObject);
begin
FORM14.LABEL1.VISIBLE:=false;
```

```
FORM14.PANEL1.VISIBLE:=true;
FORM14.Caption:='İŞ LİSTESİ HAZIRLA';
FORM14.SHOW;
FORM16.Hide;
FORM1.HIDE;
end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
var
CLS:WORD;
begin
 MESSAGEBEEP(MB_ICONASTERISK);
 CLS:=APPLICATION.MessageBox('PROGRAMI SONLANDIRMAK İSTEDİĞİNİZDEN
EMİNMİSİNİZ?','PROGRAM SONLANDIR',MB_YESNO+MB_ICONQUESTION);
 IF CLS= IDNO THEN
 BEGIN
  action:=caNone ;
  //FORM1.Hide;
 END;


end;

end.
```

FORM 2 CODES

```
unit Unit2;

interface

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

type
 TForm2 = class(TForm)
  DBGrid1: TDBGrid;
  tfXPEdit2: TtfXPEdit;
  tfXPEdit3: TtfXPEdit;
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Panel1: TPanel;
  Memo1: TMemo;
  Label4: TLabel;
```

```
Label6: TLabel;
Label7: TLabel;
Label8: TLabel;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
MaskEdit1: TMaskEdit;
MaskEdit2: TMaskEdit;
MaskEdit3: TMaskEdit;
MaskEdit4: TMaskEdit;
MaskEdit5: TMaskEdit;
Panel2: TPanel;
tfXPEdit1: TtfXPEdit;
tfXPEdit4: TtfXPEdit;
LbSpeedButton1: TLbSpeedButton;
LbSpeedButton2: TLbSpeedButton;
LbSpeedButton3: TLbSpeedButton;
Panel3: TPanel;
DataSource1: TDataSource;
LbSpeedButton4: TLbSpeedButton;
LbSpeedButton5: TLbSpeedButton;
ADOConnection1: TADOConnection;
ADOQuery1: TADOQuery;
ADOQuery2: TADOQuery;
DataSource2: TDataSource;
MainMenu1: TMainMenu;
M1: TMenuItem;
MTERKAYIT1: TMenuItem;
HAYVANKAYIT1: TMenuItem;
A1: TMenuItem;
DParazit1: TMenuItem;
Parazit1: TMenuItem;
KaytArama1: TMenuItem;
MterismineGre1: TMenuItem;
HayvanaGreArama1: TMenuItem;
Hazirla1: TMenuItem;
BugnnDetaylar1: TMenuItem;
k1: TMenuItem;
Gizle1: TMenuItem;
ADOQuery3: TADOQuery;
DataSource3: TDataSource;
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure LbSpeedButton5Click(Sender: TObject);
procedure LbSpeedButton1Click(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
```

```
    procedure tfXPEdit2KeyPress(Sender: TObject; var Key: Char);
    procedure MaskEdit1KeyPress(Sender: TObject; var Key: Char);
    procedure tfXPEdit3KeyPress(Sender: TObject; var Key: Char);
    procedure MaskEdit2KeyPress(Sender: TObject; var Key: Char);
    procedure MaskEdit3KeyPress(Sender: TObject; var Key: Char);
    procedure MaskEdit5KeyPress(Sender: TObject; var Key: Char);
    procedure MaskEdit4KeyPress(Sender: TObject; var Key: Char);
    procedure tfXPEdit1KeyPress(Sender: TObject; var Key: Char);
    procedure tfXPEdit4KeyPress(Sender: TObject; var Key: Char);
    procedure DBGrid1CellClick(Column: TColumn);
    procedure FormShow(Sender: TObject);
    procedure LbSpeedButton2Click(Sender: TObject);
    procedure LbSpeedButton3Click(Sender: TObject);
    procedure MaskEdit3Exit(Sender: TObject);
    procedure LbSpeedButton4Click(Sender: TObject);
    procedure DBGrid1KeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure DBGrid1Exit(Sender: TObject);
    procedure MTERKAYIT1Click(Sender: TObject);
    procedure A1Click(Sender: TObject);
    procedure DParazit1Click(Sender: TObject);
    procedure Parazit1Click(Sender: TObject);
    procedure MterismineGre1Click(Sender: TObject);
    procedure HayvanaGreArama1Click(Sender: TObject);
    procedure Hazirla1Click(Sender: TObject);
    procedure BugnnDetaylar1Click(Sender: TObject);
    procedure k1Click(Sender: TObject);
    procedure HAYVANKAYIT1Click(Sender: TObject);
    procedure Gizle1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;
  dlt:string;
  nm:integer;
implementation

uses Unit1, Unit9, Unit16, Unit3, Unit13;

{$R *.dfm}

procedure TForm2.FormClose(Sender: TObject; var Action: TCloseAction);
begin
```

```
form2.Hide;
FORM1.SHOW;

end;

procedure TForm2.LbSpeedButton5Click(Sender: TObject);
begin
form2.LbSpeedButton4.Click;
FORM2.hide;
FORM1.Show;

end;

procedure TForm2.LbSpeedButton1Click(Sender: TObject);
var
 gn,ay,yil,saat,dak,sn,sl:word;
ta:string;
begin
 if (form2.tfXPEdit2.Text <> '') and (form2.tfXPEdit3.Text <> '') and (form2.MaskEdit3.Text
<>'(   )   .   ') then
 begin
  form2.ADOQuery1.SQL.Text:='select * from musteri where
Misim='+#39+form2.tfXPEdit2.Text+#39+' and Msoyisim='+#39+form2.tfXPEdit3.Text+#39+'
and ceptel='+#39+form2.MaskEdit3.Text+#39;
  form2.ADOQuery1.Open;
  if form2.ADOQuery1['Misim'] = null then
  begin
    decodedate(date, yil, ay, gn);
    decodetime(time,saat,dak,sn,sl);

ta:='M'+inttostr(gn)+inttostr(ay)+inttostr(yil)+inttostr(saat)+inttostr(dak)+inttostr(sn)+inttostr(sl);

    FORM2.ADOQuery1.SQL.Text:='INSERT INTO musteri
(Misim,Msoyisim,Adres,Evtel,Istel,Ceptel,Fax,Extratel,Eposta,Web,Mislemkodu) values
('+#39+form2.tfXPEdit2.Text+#39+','+#39+form2.tfXPEdit3.Text+#39+','+#39+form2.Memo1.
Text+#39+','+#39+form2.MaskEdit1.Text+#39+','+#39+form2.MaskEdit2.Text+#39+','+#39+for
m2.MaskEdit3.Text+#39+','+#39+form2.MaskEdit4.Text+#39+','+#39+form2.MaskEdit5.Text+#
39+','+#39+form2.tfXPEdit1.Text+#39+','+#39+form2.tfXPEdit4.Text+#39+','+#39+ta+#39+')';
    form2.ADOQuery1.ExecSQL;
    {form2.ADOQuery1.Close;
    form2.ADOQuery1.SQL.Clear;
    form2.ADOQuery1.SQL.Text:='select * from musteri';
    form2.ADOQuery1.Open;
    form2.ADOQuery1.Insert;
    form2.ADOQuery1.FieldByName('Misim').AsString:=form2.tfXPEdit2.Text;
    form2.ADOQuery1.FieldByName('Msoyisim').AsString:=form2.tfXPEdit3.Text;
    form2.ADOQuery1.FieldByName('Adres').AsString:=form2.Memo1.Text;
```

```
    form2.ADOQuery1.FieldByName('Evtel').AsString:=form2.MaskEdit1.Text;
    form2.ADOQuery1.FieldByName('Istel').AsString:=form2.MaskEdit2.Text;
    form2.ADOQuery1.FieldByName('Ceptel').AsString:=form2.MaskEdit3.Text;
    form2.ADOQuery1.FieldByName('Fax').AsString:=form2.MaskEdit4.Text;
    form2.ADOQuery1.FieldByName('ExtraTel').AsString:=form2.MaskEdit5.Text;
    form2.ADOQuery1.FieldByName('Eposta').AsString:=form2.tfXPEdit1.Text;
    form2.ADOQuery1.FieldByName('Web').AsString:=form2.tfXPEdit4.Text;
    form2.ADOQuery1.FieldByName('Mislemkodu').AsString:=ta;
    form2.ADOQuery1.Post;}
    SHOWMESSAGE('THE RECORD SAVED'+#13+'ACCEPTANCE CODE:'+TA);
    FORM2.LbSpeedButton4.Click;
   end
   else
    BEGIN
    //MESSAGEBEEP(MB_ICONHAND);
    APPLICATION.MessageBox('OWNER SAVED
BEFORE','CAUTIOUN',MB_OK+MB_ICONSTOP);
    END;
 end
 else if (form2.tfXPEdit2.Text='') and (form2.tfXPEdit3.Text='') and (form2.MaskEdit3.Text='(
)  .  ') then
 begin
  MESSAGEBEEP(MB_ICONASTERISK);
  APPLICATION.MessageBox('PLEASE FILL NAME, SURNAME AND GSM
NO','ATTENTION',MB_OK + MB_ICONEXCLAMATION);
 END
 else if (form2.tfXPEdit2.Text='') and (form2.tfXPEdit3.Text='') THEN
 begin
  beep;
  showmessage('PLEASE FILL NAME AND SURNAME');
 END
 else if (form2.tfXPEdit2.Text='') and (form2.MASKEDIT3.Text='(  )  .  ') THEN
 begin
  beep;
  showmessage('PLEASE FILL NAME AND GSM NO');
 END
 else if (form2.tfXPEdit3.Text='') and (form2.MaskEdit3.Text='(  )  .  ') THEN
 begin
  beep;
  showmessage('PLEASE FILL SURNAME AND GSM NO');
 END
 else if (form2.tfXPEdit2.Text='')THEN
 begin
  beep;
  showmessage('PLEASE FILL NAME');
 END
 else if (form2.tfXPEdit3.Text='')THEN
```

```
  begin
    beep;
    showmessage('PLEASE FILL SURNAME');
  END
  else if (form2.MASKEDIT3.Text='(   )   .    ')THEN
  begin
    beep;
    showmessage('PLEASE FILL GSM NO');
  END
end;

procedure TForm2.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if key = vk_f2 then
    form2.LbSpeedButton1.Click;//kaydet
  if key = vk_f3 then
    form2.LbSpeedButton2.Click;//düzelt
  if key = vk_f4 then
    form2.LbSpeedButton3.Click;//sil
  if key = vk_f5 then
    form2.LbSpeedButton4.Click;//temizle
  if key = vk_f6 then
    form2.LbSpeedButton5.Click;//ana menü
  if key = vk_escape then
    form2.Close;
end;

procedure TForm2.tfXPEdit2KeyPress(Sender: TObject; var Key: Char);
begin
if key = #13 then
form2.tfXPEdit3.SetFocus;
end;

procedure TForm2.MaskEdit1KeyPress(Sender: TObject; var Key: Char);
begin
if key = #13 then
form2.MaskEdit2.SetFocus;
end;

procedure TForm2.tfXPEdit3KeyPress(Sender: TObject; var Key: Char);
begin
if key = #13 then
form2.MaskEdit1.SetFocus;
end;

procedure TForm2.MaskEdit2KeyPress(Sender: TObject; var Key: Char);
```

```
begin
if key = #13 then
form2.MaskEdit3.SetFocus;
end;

procedure TForm2.MaskEdit3KeyPress(Sender: TObject; var Key: Char);
begin
if key = #13 then
form2.MaskEdit4.SetFocus;
end;

procedure TForm2.MaskEdit5KeyPress(Sender: TObject; var Key: Char);
begin
if key = #13 then
form2.tfXPEdit1.SetFocus;
end;

procedure TForm2.MaskEdit4KeyPress(Sender: TObject; var Key: Char);
begin
if key = #13 then
form2.MaskEdit5.SetFocus;
end;

procedure TForm2.tfXPEdit1KeyPress(Sender: TObject; var Key: Char);
begin
if key = #13 then
form2.tfXPEdit4.SetFocus;
end;

procedure TForm2.tfXPEdit4KeyPress(Sender: TObject; var Key: Char);
begin
if key = #13 then
form2.Memo1.SetFocus;
end;

procedure TForm2.DBGrid1CellClick(Column: TColumn);
begin
 IF FORM2.DBGrid1.FieldCount <> 0 THEN
 BEGIN
  form2.Panel1.Caption:=form2.DBGrid1.Fields[0].Text;
  form2.tfXPEdit2.Text:=form2.DBGrid1.Fields[1].Text;
  form2.tfXPEdit3.Text:=form2.DBGrid1.Fields[2].Text;
  form2.MaskEdit1.Text:=form2.DBGrid1.Fields[4].Text;
  form2.MaskEdit2.Text:=form2.DBGrid1.Fields[5].Text;
  form2.MaskEdit3.Text:=form2.DBGrid1.Fields[6].Text;
  form2.MaskEdit4.Text:=form2.DBGrid1.Fields[7].Text;
  form2.MaskEdit5.Text:=form2.DBGrid1.Fields[8].Text;
```

```
   form2.tfXPEdit1.Text:=form2.DBGrid1.Fields[9].Text;
   form2.tfXPEdit4.Text:=form2.DBGrid1.Fields[10].Text;
   form2.Memo1.Text:=form2.DBGrid1.Fields[3].Text;
  END

end;

procedure TForm2.FormShow(Sender: TObject);
begin
  form2.ADOQuery2.SQL.Text:='select * from musteri ORDER BY Misim';
  form2.ADOQuery2.Open;
  form2.LbSpeedButton4.Click;
end;

procedure TForm2.LbSpeedButton2Click(Sender: TObject);
begin
  if (form2.Panel1.Caption<>'')then
  begin
  form2.ADOQuery1.SQL.Text:='update musteri set Misim='+#39+form2.tfXPEdit2.Text+#39+',
Msoyisim='+#39+form2.tfXPEdit3.Text+#39+', Adres='+#39+form2.Memo1.Text+#39+',
Evtel='+#39+form2.MaskEdit1.Text+#39+', Istel='+#39+form2.MaskEdit2.Text+#39+',
Ceptel='+#39+form2.MaskEdit3.Text+#39+', Fax='+#39+form2.MaskEdit4.Text+#39+',
Extratel='+#39+form2.MaskEdit5.Text+#39+', Eposta='#39+form2.tfXPEdit1.Text+#39+',
Web='+#39+form2.tfXPEdit4.Text+#39+' where
Sid='+#39+form2.DBGrid1.Fields[0].Text+#39;
  form2.ADOQuery1.ExecSQL;
  SHOWMESSAGE('UPDATE PROCESS WAS DONE');
  form2.ADOQuery2.SQL.Text:='select * from musteri ORDER BY Misim';
  form2.ADOQuery2.Open;
  end
  else
    APPLICATION.MessageBox('PLEASE SELECT OWNER FROM LIST TO MAKE
UPDATE','SELECT OWNER',MB_OK+MB_ICONINFORMATION);
end;

procedure TForm2.LbSpeedButton3Click(Sender: TObject);
VAR
A:WORD;
begin
  if (form2.Panel1.Caption<>'')then
  begin
    MESSAGEBEEP(MB_ICONEXCLAMATION);
    A:=APPLICATION.MessageBox('ARE YOU SURE TO DELETE ALL INFORMATION OF
SELECTED OWNER?','CAUTIOUN',MB_YESNO+MB_ICONQUESTION);
    IF A=IDYES THEN
    BEGIN
      dlt:=form2.panel1.caption;
```

```
    FORM2.ADOQuery1.SQL.Text:='DELETE FROM musteri where
Sid='+#39+form2.Panel1.Caption+#39;
    form2.ADOQuery1.ExecSQL;

    form2.ADOQuery3.SQL.Text:='select Hid from hayvan where Sid='+#39+dlt+#39;
    form2.ADOQuery3.Open;
    form2.ADOQuery3.First;
    while not form2.ADOQuery3.Eof do
    begin
     dlt:=form2.ADOQuery3['Hid'];
     FORM2.ADOQuery1.SQL.Text:='DELETE FROM hayvan where Hid='+#39+dlt+#39;
     form2.ADOQuery1.ExecSQL;
     FORM2.ADOQuery1.SQL.Text:='DELETE FROM asilama where Hid='+#39+dlt+#39;
     form2.ADOQuery1.ExecSQL;
     FORM2.ADOQuery1.SQL.Text:='DELETE FROM disparazit where Hid='+#39+dlt+#39;
     form2.ADOQuery1.ExecSQL;
     FORM2.ADOQuery1.SQL.Text:='DELETE FROM icparazit where Hid='+#39+dlt+#39;
     form2.ADOQuery1.ExecSQL;
     form2.ADOQuery3.Next;
    end;
    SHOWMESSAGE('DELETE PROCESS WAS DONE');
    FORM2.LbSpeedButton4.Click;
    dlt:='';
   END;
 end
 else
   APPLICATION.MessageBox('PLEASE SELECT OWNER FROM LIST TO
DELETE','SELECT OWNER',MB_OK+MB_ICONINFORMATION);
end;

procedure TForm2.MaskEdit3Exit(Sender: TObject);
begin
  if form2.MaskEdit3.Text='(   )   .   ' then
  APPLICATION.MessageBox('ATTENTION YOU MUST ENTER GSM NO'+#13+'IF YOU
 HAVE NO GSM NO, YOU CAN ENTER OTHER PHONE THERE','IMPORTANT
 CAUTION',MB_OK+MB_ICONEXCLAMATION);



end;

procedure TForm2.LbSpeedButton4Click(Sender: TObject);
begin
  form2.Panel1.Caption:='';
  form2.tfXPEdit2.Clear;
  form2.tfXPEdit2.SetFocus;
  form2.tfXPEdit3.Clear;
```

```
 form2.MaskEdit1.Clear;
 form2.MaskEdit2.Clear;
 form2.MaskEdit3.Clear;
 form2.MaskEdit4.Clear;
 form2.MaskEdit5.Clear;
 form2.tfXPEdit1.Clear;
 form2.tfXPEdit4.Clear;
 form2.Memo1.Lines.Clear;
 form2.ADOQuery1.Close;
 form2.ADOQuery2.SQL.Text:='select * from musteri ORDER BY Misim';
 form2.ADOQuery2.Open;
 form2.DBGrid1.Refresh;
end;

procedure TForm2.DBGrid1KeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
 IF FORM2.DBGrid1.FieldCount <> 0 THEN
 BEGIN
  form2.Panel1.Caption:=form2.DBGrid1.Fields[0].Text;
  form2.tfXPEdit2.Text:=form2.DBGrid1.Fields[1].Text;
  form2.tfXPEdit3.Text:=form2.DBGrid1.Fields[2].Text;
  form2.MaskEdit1.Text:=form2.DBGrid1.Fields[4].Text;
  form2.MaskEdit2.Text:=form2.DBGrid1.Fields[5].Text;
  form2.MaskEdit3.Text:=form2.DBGrid1.Fields[6].Text;
  form2.MaskEdit4.Text:=form2.DBGrid1.Fields[7].Text;
  form2.MaskEdit5.Text:=form2.DBGrid1.Fields[8].Text;
  form2.tfXPEdit1.Text:=form2.DBGrid1.Fields[9].Text;
  form2.tfXPEdit4.Text:=form2.DBGrid1.Fields[10].Text;
  form2.Memo1.Text:=form2.DBGrid1.Fields[3].Text;
 END
end;

procedure TForm2.DBGrid1Exit(Sender: TObject);
begin
//form2.LbSpeedButton4.Click;
end;

procedure TForm2.Hazirla1Click(Sender: TObject);
begin
form2.LbSpeedButton4.Click;
form2.Hide;
form1.LbSpeedButton1.Click;
end;

procedure TForm2.BugnnDetaylar1Click(Sender: TObject);
begin
```

```
form2.LbSpeedButton4.Click;
form2.Hide;
form16.LbSpeedButton1.Click;
end;

procedure TForm2.Parazit1Click(Sender: TObject);
begin
form2.LbSpeedButton4.Click;
form2.Hide;
form1.LbSpeedButton5.Click;
end;

procedure TForm2.DParazit1Click(Sender: TObject);
begin
form2.LbSpeedButton4.Click;
form2.Hide;
form1.LbSpeedButton4.Click;
end;

procedure TForm2.A1Click(Sender: TObject);
begin
form2.LbSpeedButton4.Click;
form2.Hide;
form1.LbSpeedButton3.Click;
end;

procedure TForm2.MTERKAYIT1Click(Sender: TObject);
begin
form3.LbSpeedButton1.Click;
end;


procedure TForm2.MterismineGre1Click(Sender: TObject);
begin
form2.LbSpeedButton4.Click;
form2.Hide;
form13.LbSpeedButton6.Click;
end;

procedure TForm2.HayvanaGreArama1Click(Sender: TObject);
begin
form2.LbSpeedButton4.Click;
form2.Hide;
form13.LbSpeedButton1.Click;
end;

procedure TForm2.k1Click(Sender: TObject);
```

```
begin
form1.LbSpeedButton7.Click;
end;


procedure TForm2.HAYVANKAYIT1Click(Sender: TObject);
begin
form2.LbSpeedButton4.Click;
form2.Hide;
form3.LbSpeedButton2.Click;
end;

procedure TForm2.Gizle1Click(Sender: TObject);
begin
form1.LbSpeedButton8.Click;
end;

end.
```

FORM 3 CODES

```
unit Unit3;

interface

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

type
  TForm3 = class(TForm)
    LbSpeedButton1: TLbSpeedButton;
    LbSpeedButton2: TLbSpeedButton;
    procedure LbSpeedButton1Click(Sender: TObject);
    procedure LbSpeedButton2Click(Sender: TObject);
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form3: TForm3;
```

```
implementation

uses Unit2, Unit1, Unit4;

{$R *.dfm}

procedure TForm3.LbSpeedButton1Click(Sender: TObject);
begin
form2.show;
form3.Close;
form1.hide;
end;

procedure TForm3.LbSpeedButton2Click(Sender: TObject);
begin
FORM4.SHOW;
FORM3.Close;
FORM1.Hide;
end;

procedure TForm3.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if key = vk_escape then
    form3.Close;
end;

end.
```

FORM 4 CODES

```
unit Unit4;

interface

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

type
  TForm4 = class(TForm)
    Panel1: TPanel;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
```

```
Label5: TLabel;
Label6: TLabel;
Label7: TLabel;
tfXPEdit1: TtfXPEdit;
tfXPEdit2: TtfXPEdit;
ComboBox1: TComboBox;
ComboBox2: TComboBox;
DateTimePicker1: TDateTimePicker;
Edit1: TEdit;
Panel3: TPanel;
SpeedButton1: TSpeedButton;
Panel4: TPanel;
LbSpeedButton1: TLbSpeedButton;
LbSpeedButton2: TLbSpeedButton;
LbSpeedButton3: TLbSpeedButton;
LbSpeedButton4: TLbSpeedButton;
LbSpeedButton5: TLbSpeedButton;
ADOQuery1: TADOQuery;
DataSource1: TDataSource;
DBGrid1: TDBGrid;
Edit2: TEdit;
DataSource2: TDataSource;
ADOQuery2: TADOQuery;
Label8: TLabel;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
MainMenu1: TMainMenu;
M1: TMenuItem;
MTERKAYIT1: TMenuItem;
HAYVANKAYIT1: TMenuItem;
A1: TMenuItem;
DParazit1: TMenuItem;
Parazit1: TMenuItem;
KaytArama1: TMenuItem;
MterismineGre1: TMenuItem;
HayvanaGreArama1: TMenuItem;
Hazirla1: TMenuItem;
BugnnDetaylar1: TMenuItem;
Gizle1: TMenuItem;
k1: TMenuItem;
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure LbSpeedButton5Click(Sender: TObject);
procedure SpeedButton1Click(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure LbSpeedButton1Click(Sender: TObject);
```

```
    procedure FormCreate(Sender: TObject);
    procedure DBGrid1CellClick(Column: TColumn);
    procedure LbSpeedButton2Click(Sender: TObject);
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure tfXPEdit1KeyPress(Sender: TObject; var Key: Char);
    procedure tfXPEdit2KeyPress(Sender: TObject; var Key: Char);
    procedure tfXPEdit2EndDock(Sender, Target: TObject; X, Y: Integer);
    procedure ComboBox1CloseUp(Sender: TObject);
    procedure ComboBox2CloseUp(Sender: TObject);
    procedure LbSpeedButton4Click(Sender: TObject);
    procedure LbSpeedButton3Click(Sender: TObject);
    procedure Edit1Change(Sender: TObject);
    procedure tfXPEdit1Exit(Sender: TObject);
    procedure tfXPEdit2Exit(Sender: TObject);
    procedure ComboBox1Exit(Sender: TObject);
    procedure ComboBox2Exit(Sender: TObject);
    procedure DBGrid1KeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure MTERKAYIT1Click(Sender: TObject);
    procedure HAYVANKAYIT1Click(Sender: TObject);
    procedure A1Click(Sender: TObject);
    procedure DParazit1Click(Sender: TObject);
    procedure Parazit1Click(Sender: TObject);
    procedure MterismineGre1Click(Sender: TObject);
    procedure HayvanaGreArama1Click(Sender: TObject);
    procedure Hazirla1Click(Sender: TObject);
    procedure BugnnDetaylar1Click(Sender: TObject);
    procedure k1Click(Sender: TObject);
    procedure Gizle1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form4: TForm4;
  MN:INTEGER;
implementation

uses Unit1, Unit9, Unit2, Unit16, Unit13, Unit3;

{$R *.dfm}

procedure TForm4.FormClose(Sender: TObject; var Action: TCloseAction);
begin
```

```
FORM4.hide;
FORM1.SHOW;
end;

procedure TForm4.LbSpeedButton5Click(Sender: TObject);
begin
form4.LbSpeedButton4.Click;
FORM4.hide;
FORM1.SHOW;

end;

procedure TForm4.SpeedButton1Click(Sender: TObject);
begin
MN:=1;
FORM9.SHOW;
end;

procedure TForm4.FormShow(Sender: TObject);
begin
  FORM4.ADOQuery2.Close;
  form4.ADOQuery2.SQL.Text:='SELECT
H.Hid,H.Hismi,M.Misim,M.Msoyisim,H.Hturu,H.Hirki,H.Hcinsiyeti,H.Hdogumtarihi,H.Sid,H.H
islemkodu from musteri M,hayvan H where H.Sid=M.Sid';
  form4.ADOQuery2.Open;
  form4.DateTimePicker1.Date:=date;
end;

procedure TForm4.LbSpeedButton1Click(Sender: TObject);
VAR
  gn,ay,yil,saat,dak,sn,sl:word;
ta:string;
begin
dateseparator := '-';    // Burada tarih'in ayraçlarını MySql database sisteminin anlayacağı şekle
dönüştürdüm...
shortdateformat := 'yyyy/m/d';
  if (form4.Edit1.Text <> '') and (form4.tfXPEdit1.Text <> '') and (form4.ComboBox1.Text <>
'SELECT ONE...') AND (FORM4.tfXPEdit2.Text <> '') AND (FORM4.ComboBox2.Text <>
'SELECT ONE...') THEN
  BEGIN
    form4.ADOQuery1.SQL.Text:='select * from hayvan where
Hismi='+#39+form4.tfXPEdit1.Text+#39+' and Hturu='+#39+form4.ComboBox1.Text+#39+'
and Hirki='+#39+form4.tfXPEdit2.Text+#39+' and
Hcinsiyeti='+#39+form4.ComboBox2.Text+#39+' and Sid='+#39+form4.Edit1.Text+#39;
    form4.ADOQuery1.Open;
    if form4.ADOQuery1['Hismi'] = null then
    begin
```

```
    decodedate(date, yil, ay, gn);
    decodetime(time,saat,dak,sn,sl);

ta:='H'+inttostr(gn)+inttostr(ay)+inttostr(yil)+inttostr(saat)+inttostr(dak)+inttostr(sn)+inttostr(sl);

    FORM4.ADOQuery1.Close;
    FORM4.ADOQuery1.SQL.Clear;
    FORM4.ADOQuery1.SQL.Text:='SELECT * FROM hayvan';
    form4.ADOQuery1.Open;
    FORM4.ADOQuery1.Insert;
    FORM4.ADOQuery1.FieldByName('Hismi').AsString:=form4.tfXPEdit1.Text;
    FORM4.ADOQuery1.FieldByName('Hturu').AsString:=form4.ComboBox1.Text;
    form4.ADOQuery1.FieldByName('Hirki').AsString:=form4.tfXPEdit2.Text;
    FORM4.ADOQuery1.FieldByName('Hcinsiyeti').AsString:=form4.ComboBox2.Text;

form4.ADOQuery1.FieldByName('Hdogumtarihi').AsString:=datetostr(form4.DateTimePicker1.
Date);
    form4.ADOQuery1.FieldByName('Sid').AsString:=form4.Edit1.Text;
    form4.ADOQuery1.FieldByName('Hislemkodu').AsString:=ta;
    form4.ADOQuery1.Post;
    SHOWMESSAGE('THE RECORD SAVED SUCCESSFULY'+#13+'ACCEPTANCE
CODE:'+TA);
    FORM4.LbSpeedButton4.Click;
   END
   else
    APPLICATION.MessageBox('THE ANIMAL SAVED
BEFORE','CAUTION',MB_OK+MB_ICONSTOP);
 end
 ELSE if (form4.Edit1.Text = '') or (form4.tfXPEdit1.Text = '') or (form4.ComboBox1.Text =
'LÜTFEN SEÇİNİZ...') or (FORM4.tfXPEdit2.Text = '') or (FORM4.ComboBox2.Text =
'LÜTFEN SEÇİNİZ...') THEN
 begin
  APPLICATION.MessageBox('PLEASE FILL EMPTY
PLACE','ATTENTION',MB_OK+MB_ICONEXCLAMATION);
  if form4.Edit1.Text = '' then
   form4.Label8.Visible:=true;
  if form4.tfXPEdit1.Text = '' then
   form4.Label9.Visible:=true;
  if FORM4.ComboBox1.Text = 'SELECT ONE...' then
   form4.Label10.Visible:=true;
  if form4.tfXPEdit2.Text = '' then
   form4.Label11.Visible:=true;
  if FORM4.ComboBox2.Text = 'SELECT ONE...' then
   form4.Label12.Visible:=true;
 end
```

```
end;

procedure TForm4.FormCreate(Sender: TObject);
begin
//dateseparator := '-';    // Burada tarih'in ayraçlarını MySql database sisteminin anlayacağı şekle
dönüştürdüm...
//shortdateformat := 'yyyy/m/d';
end;

procedure TForm4.DBGrid1CellClick(Column: TColumn);
begin
 if form4.DBGrid1.FieldCount <> 0 then
 begin
  //form4.LbSpeedButton1.Enabled:=false;
  DATESEPARATOR:='.';
  SHORTDATEFORMAT:='DD/MM/YYYY';
  FORM4.Panel1.Caption:=FORM4.DBGrid1.Fields[0].Text;
  FORM4.Edit2.Text:=FORM4.DBGrid1.Fields[2].Text+' '+FORM4.DBGrid1.Fields[3].Text;
  FORM4.tfXPEdit1.Text:=FORM4.DBGrid1.Fields[1].Text;
  FORM4.ComboBox1.Text:=FORM4.DBGrid1.Fields[4].Text;
  FORM4.tfXPEdit2.Text:=FORM4.DBGrid1.Fields[5].Text;
  FORM4.ComboBox2.Text:=FORM4.DBGrid1.Fields[6].Text;
  FORM4.DateTimePicker1.Date:=STRTODATE(FORM4.DBGrid1.Fields[7].Text);
  FORM4.Edit1.Text:=FORM4.DBGrid1.Fields[8].Text
 end;



end;

procedure TForm4.LbSpeedButton2Click(Sender: TObject);
begin
 if (form4.Panel1.Caption <> '') then
 begin
  dateseparator:='-';
  shortdateformat:='yyyy/m/d';
  form4.ADOQuery1.Close;
  form4.ADOQuery1.SQL.Text:='update hayvan set
Hismi='+#39+form4.tfXPEdit1.Text+#39+', Hturu='+#39+form4.ComboBox1.Text+#39+',
Hirki='+#39+form4.tfXPEdit2.Text+#39+', Hcinsiyeti='+#39+form4.ComboBox2.Text+#39+',
Sid='+#39+form4.Edit1.Text+#39+' WHERE Hid='+#39+form4.Panel1.Caption+#39;
  form4.ADOQuery1.ExecSQL;
  showmessage('UPDATE PROCESS WAS DONE SUCCESSFULY');
  FORM4.ADOQuery2.Close;
  form4.ADOQuery2.SQL.Text:='SELECT
H.Hid,H.Hismi,M.Misim,M.Msoyisim,H.Hturu,H.Hirki,H.Hcinsiyeti,H.Hdogumtarihi,H.Sid,H.H
islemkodu from musteri M,hayvan H where H.Sid=M.Sid';
```

```
    form4.ADOQuery2.Open;
  END
  ELSE
    APPLICATION.MessageBox('PLEASE SELECT ANIMAL FROM LIST TO MAKE
UPDATE','SELECT ANIMAL',MB_OK+MB_ICONINFORMATION);
end;

procedure TForm4.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if key = vk_f2 then
    form4.LbSpeedButton1.Click;//kaydet
  if key = vk_f3 then
    form4.LbSpeedButton2.Click;//düzelt
  if key = vk_f4 then
    form4.LbSpeedButton3.Click;//sil
  if key = vk_f5 then
    form4.LbSpeedButton4.Click;//temizle
  if key = vk_f6 then
    form4.LbSpeedButton5.Click;//ana menü
  if key = vk_escape then
    form4.Close;
end;

procedure TForm4.tfXPEdit1KeyPress(Sender: TObject; var Key: Char);
begin
  if key = #13 then
  begin
    form4.ComboBox1.DroppedDown:=true;
    form4.ComboBox1.SetFocus;
    end;
end;

procedure TForm4.tfXPEdit2KeyPress(Sender: TObject; var Key: Char);
begin
  if key = #13 then
  begin
    form4.ComboBox2.DroppedDown:=true;
    form4.ComboBox2.SetFocus;
    end;
end;

procedure TForm4.tfXPEdit2EndDock(Sender, Target: TObject; X, Y: Integer);
begin
    form4.DateTimePicker1.SetFocus;
end;
```

```
procedure TForm4.ComboBox1CloseUp(Sender: TObject);
begin
  if form4.ComboBox1.Text <> 'SELECT ONE...' then
    form4.Label10.Visible:=false;
  form4.tfXPEdit2.SetFocus;
end;

procedure TForm4.ComboBox2CloseUp(Sender: TObject);
begin
  if form4.ComboBox2.Text <> 'SELECT ONE...' then
    form4.Label12.Visible:=false;
  form4.DateTimePicker1.SetFocus;
end;

procedure TForm4.LbSpeedButton4Click(Sender: TObject);
begin
  form4.LbSpeedButton1.Enabled:=true;
  FORM4.Panel1.Caption:='';
  FORM4.Edit1.Clear;
  FORM4.Edit2.Clear;
  FORM4.tfXPEdit1.Clear;
  FORM4.ComboBox1.Text:='SELECT ONE...';
  FORM4.tfXPEdit2.Clear;
  FORM4.ComboBox2.Text:='SELECT ONE...';
  FORM4.DateTimePicker1.Date:=DATE;
  form4.Label8.Visible:=false;
  form4.Label9.Visible:=false;
  form4.Label10.Visible:=false;
  form4.Label11.Visible:=false;
  form4.Label12.Visible:=false;
  form4.ADOQuery2.SQL.Text:='SELECT
H.Hid,H.Hismi,M.Misim,M.Msoyisim,H.Hturu,H.Hirki,H.Hcinsiyeti,H.Hdogumtarihi,H.Sid,H.H
islemkodu from musteri M,hayvan H where H.Sid=M.Sid';
  form4.ADOQuery2.Open;
end;

procedure TForm4.LbSpeedButton3Click(Sender: TObject);
VAR
A:WORD;
begin
  if (form4.Panel1.Caption <> '') then
  begin
    A:=APPLICATION.MessageBox('ARE YOU SURE TO DELETE SELECTED
RECORD?','CAUTION',MB_YESNO+MB_ICONEXCLAMATION);
    IF A=IDYES THEN
    BEGIN
      form4.ADOQuery1.Close;
```

```
    form4.ADOQuery1.SQL.Text:='DELETE from hayvan WHERE
Hid='+#39+form4.Panel1.Caption+#39;
    form4.ADOQuery1.ExecSQL;
    showmessage('DELETE PROCESS WAS DONE SUCCESSFULY');
    FORM4.LbSpeedButton4.Click;
   END;
  END
  ELSE
   APPLICATION.MessageBox('PLEASE SELECT ANIMAL FROM LIST TO
DELETE','SELECT ANIMAL',MB_OK+MB_ICONINFORMATION);
end;

procedure TForm4.Edit1Change(Sender: TObject);
begin
  if form4.Edit1.Text <> '' then
  form4.Label8.Visible:=false;
end;

procedure TForm4.tfXPEdit1Exit(Sender: TObject);
begin
  if form4.Edit1.Text <> '' then
  form4.Label9.Visible:=false;
end;

procedure TForm4.tfXPEdit2Exit(Sender: TObject);
begin
  if form4.Edit1.Text <> '' then
  form4.Label11.Visible:=false;
end;

procedure TForm4.ComboBox1Exit(Sender: TObject);
begin
  if form4.ComboBox1.Text <> 'SELECT ONE...' then
  form4.Label10.Visible:=false;
end;

procedure TForm4.ComboBox2Exit(Sender: TObject);
begin
  if form4.ComboBox2.Text <> 'SELECT ONE...' then
  form4.Label12.Visible:=false;
end;

procedure TForm4.DBGrid1KeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if form4.DBGrid1.FieldCount <> 0 then
  begin
```

```
  form4.LbSpeedButton1.Enabled:=false;
  DATESEPARATOR:='.';
  SHORTDATEFORMAT:='DD/MM/YYYY';
  FORM4.Panel1.Caption:=FORM4.DBGrid1.Fields[0].Text;
  FORM4.Edit2.Text:=FORM4.DBGrid1.Fields[2].Text+' '+FORM4.DBGrid1.Fields[3].Text;
  FORM4.tfXPEdit1.Text:=FORM4.DBGrid1.Fields[1].Text;
  FORM4.ComboBox1.Text:=FORM4.DBGrid1.Fields[4].Text;
  FORM4.tfXPEdit2.Text:=FORM4.DBGrid1.Fields[5].Text;
  FORM4.ComboBox2.Text:=FORM4.DBGrid1.Fields[6].Text;
  FORM4.DateTimePicker1.Date:=STRTODATE(FORM4.DBGrid1.Fields[7].Text);
  FORM4.Edit1.Text:=FORM4.DBGrid1.Fields[8].Text
 end;
end;


procedure TForm4.Hazirla1Click(Sender: TObject);
begin
form4.LbSpeedButton4.Click;
form4.Hide;
form1.LbSpeedButton1.Click;
end;

procedure TForm4.BugnnDetaylar1Click(Sender: TObject);
begin
form4.LbSpeedButton4.Click;
form4.Hide;
form16.LbSpeedButton1.Click;
end;

procedure TForm4.Parazit1Click(Sender: TObject);
begin
form4.LbSpeedButton4.Click;
form4.Hide;
form1.LbSpeedButton5.Click;
end;

procedure TForm4.DParazit1Click(Sender: TObject);
begin
form4.LbSpeedButton4.Click;
form4.Hide;
form1.LbSpeedButton4.Click;
end;

procedure TForm4.A1Click(Sender: TObject);
begin
form4.LbSpeedButton4.Click;
form4.Hide;
```

```
form1.LbSpeedButton3.Click;
end;

procedure TForm4.MTERKAYIT1Click(Sender: TObject);
begin
form4.LbSpeedButton4.Click;
form4.Hide;
form3.LbSpeedButton1.Click;
end;


procedure TForm4.MterismineGre1Click(Sender: TObject);
begin
form4.LbSpeedButton4.Click;
form4.Hide;
form13.LbSpeedButton6.Click;
end;

procedure TForm4.HayvanaGreArama1Click(Sender: TObject);
begin
form4.LbSpeedButton4.Click;
form4.Hide;
form13.LbSpeedButton1.Click;
end;

procedure TForm4.k1Click(Sender: TObject);
begin
form1.LbSpeedButton7.Click;
end;


procedure TForm4.HAYVANKAYIT1Click(Sender: TObject);
begin
form3.LbSpeedButton2.Click;
end;
procedure TForm4.Gizle1Click(Sender: TObject);
begin
form1.LbSpeedButton8.Click;
end;

end.
```

FORM 5 CODES

```
unit Unit5;
```

```
interface

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

type
  TForm5 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    DateTimePicker1: TDateTimePicker;
    DateTimePicker2: TDateTimePicker;
    LbSpeedButton1: TLbSpeedButton;
    LbSpeedButton2: TLbSpeedButton;
    LbSpeedButton3: TLbSpeedButton;
    LbSpeedButton4: TLbSpeedButton;
    LbSpeedButton5: TLbSpeedButton;
    Edit1: TEdit;
    Panel1: TPanel;
    SpeedButton1: TSpeedButton;
    Edit2: TEdit;
    DBGrid1: TDBGrid;
    Panel3: TPanel;
    Panel4: TPanel;
    DataSource1: TDataSource;
    CheckBox1: TCheckBox;
    ADOQuery1: TADOQuery;
    ADOQuery2: TADOQuery;
    DataSource2: TDataSource;
    Label5: TLabel;
    ADOQuery3: TADOQuery;
    DataSource3: TDataSource;
    Label8: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label9: TLabel;
    MainMenu1: TMainMenu;
    M1: TMenuItem;
    MTERKAYIT1: TMenuItem;
    HAYVANKAYIT1: TMenuItem;
    A1: TMenuItem;
    DParazit1: TMenuItem;
    Parazit1: TMenuItem;
```

```
  KaytArama1: TMenuItem;
  MterismineGre1: TMenuItem;
  HayvanaGreArama1: TMenuItem;
  Hazirla1: TMenuItem;
  BugnnDetaylar1: TMenuItem;
  Gizle1: TMenuItem;
  k1: TMenuItem;
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure LbSpeedButton5Click(Sender: TObject);
  procedure SpeedButton1Click(Sender: TObject);
  procedure FormShow(Sender: TObject);
  procedure LbSpeedButton1Click(Sender: TObject);
  procedure Edit1Change(Sender: TObject);
  procedure LbSpeedButton4Click(Sender: TObject);
  procedure CheckBox1Click(Sender: TObject);
  procedure DateTimePicker2Change(Sender: TObject);
  procedure Edit2Exit(Sender: TObject);
  procedure DateTimePicker1Change(Sender: TObject);
  procedure LbSpeedButton2Click(Sender: TObject);
  procedure DBGrid1KeyUp(Sender: TObject; var Key: Word;
    Shift: TShiftState);
  procedure LbSpeedButton3Click(Sender: TObject);
  procedure FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
  procedure Hazirla1Click(Sender: TObject);
  procedure BugnnDetaylar1Click(Sender: TObject);
  procedure Parazit1Click(Sender: TObject);
  procedure DParazit1Click(Sender: TObject);
  procedure A1Click(Sender: TObject);
  procedure MTERKAYIT1Click(Sender: TObject);
  procedure HAYVANKAYIT1Click(Sender: TObject);
  procedure MterismineGre1Click(Sender: TObject);
  procedure HayvanaGreArama1Click(Sender: TObject);
  procedure k1Click(Sender: TObject);
  procedure Gizle1Click(Sender: TObject);
  procedure Edit2KeyPress(Sender: TObject; var Key: Char);
  procedure DateTimePicker1CloseUp(Sender: TObject);
  procedure DBGrid1CellClick(Column: TColumn);
  procedure DBGrid1MouseMove(Sender: TObject; Shift: TShiftState; X,
    Y: Integer);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
```

```
 Form5: TForm5;
 hyv:integer;
 KD,DAK:STRING;
 SS:integer=0;
implementation

uses Unit1, Unit8, Unit2, Unit4, Unit16, Unit3, Unit13;

{$R *.dfm}

procedure TForm5.FormClose(Sender: TObject; var Action: TCloseAction);
begin
FORM5.Hide;
FORM1.SHOW;
FORM5.CheckBox1.Checked:=FALSE;

end;

procedure TForm5.LbSpeedButton5Click(Sender: TObject);
begin
FORM5.CheckBox1.Checked:=FALSE;
FORM5.Hide;
FORM1.SHOW;
end;

procedure TForm5.SpeedButton1Click(Sender: TObject);
begin
hyv:=5;
FORM8.SHOW;
end;

procedure TForm5.FormShow(Sender: TObject);
begin
  form5.DateTimePicker1.Date:=date;
  form5.DateTimePicker2.Date:=date;
  KD:='0';
  FORM5.LbSpeedButton4.Click;
  //FORM5.Panel1.SetFocus;
end;

procedure TForm5.LbSpeedButton1Click(Sender: TObject);
var
  dr,gn,ay,yil,saat,dak,sn,sl:word;
  ta:string;
begin
  if (form5.Edit1.Text <> '') and (datetostr(form5.DateTimePicker1.Date) <>
datetostr(form5.DateTimePicker2.Date)) and (form5.Edit2.Text <> '') then
```

```
 begin
   dateseparator:='-';
   shortdateformat:='yyyy/mm/dd';

   decodedate(date, yil, ay, gn);
   decodetime(time,saat,dak,sn,sl);

ta:='A'+inttostr(gn)+inttostr(ay)+inttostr(yil)+inttostr(saat)+inttostr(dak)+inttostr(sn)+inttostr(sl);

   form5.ADOQuery1.SQL.Text:='select * from asilama';
   form5.ADOQuery1.Open;
   form5.ADOQuery1.Insert;
   form5.ADOQuery1.FieldByName('Hid').AsString:=form5.Edit1.Text;

form5.ADOQuery1.FieldByName('AsilamaTarihi').AsString:=datetostr(form5.DateTimePicker1.
Date);
   form5.ADOQuery1.FieldByName('YapilanAsilama').AsString:=form5.Edit2.Text;

form5.ADOQuery1.FieldByName('TekrarAsilamaTarihi').AsString:=datetostr(form5.DateTimePi
cker2.Date);
   form5.ADOQuery1.FieldByName('Aislemkodu').AsString:=ta;

   form5.ADOQuery3.SQL.Text:='select * from asilama where
Hid='+#39+form5.Edit1.Text+#39+' and
AsilamaTarihi='+#39+datetostr(form5.DateTimePicker1.Date)+#39+' and
YapilanAsilama='+#39+form5.Edit2.Text+#39+' and
TekrarAsilamaTarihi='+#39+datetostr(form5.DateTimePicker2.Date)+#39;
   form5.ADOQuery3.Open;
   if form5.ADOQuery3.RecordCount = 0 then
   begin
    form5.ADOQuery1.Post;
    SHOWMESSAGE('THE RECORD SAVED SUCCESSFULY'+#13+'ACCEPTANCE
CODE:'+TA);
    FORM5.LbSpeedButton4.Click;
   end
   else
   begin
    dr:=application.MessageBox('THE INFORMATIONS SAVED BEFORE'+#13+'STILL DO
YOU WANT TO SAVE AGAIN?','ATTENTION',MB_YESNO+MB_ICONEXCLAMATION);
    IF DR=IDYES THEN
    BEGIN
     form5.ADOQuery1.Post;
     SHOWMESSAGE('THE RECORD SAVED SUCCESSFULY'+#13+'ACCEPTANCE
CODE:'+TA);
     FORM5.LbSpeedButton4.Click;
    END;
   END;
```

```
 END
 ELSE
 begin
  APPLICATION.MessageBox('PLEASE FILL EMPTY
PLACE','CAUTION',MB_OK+MB_ICONEXCLAMATION);
  if form5.Edit1.Text = '' then
   form5.Label6.Visible:=true;

  if datetostr(form5.DateTimePicker1.Date) = datetostr(form5.DateTimePicker2.Date) then
  begin
   //showmessage('1: '+datetostr(form5.DateTimePicker1.Date)+#13+#13+'2:
'+datetostr(form5.DateTimePicker2.Date));
   form5.Label7.Visible:=true;
   form5.Label9.Visible:=true;
  end;
  if form5.Edit2.Text = '' then
   form5.Label8.Visible:=true;
 end;




end;

procedure TForm5.Edit1Change(Sender: TObject);
begin
 if form5.Edit1.Text <> '' then
 begin
  form5.ADOQuery2.SQL.Text:='select
H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi,A.Hid,A.Aislemkodu from
hayvan H,asilama A where H.Hid='+#39+form5.Edit1.Text+#39+'and H.Hid=A.Hid order by
TekrarAsilamaTarihi DESC';
  form5.ADOQuery2.Open;
  form5.Label6.Visible:=false;
 end;
end;

procedure TForm5.LbSpeedButton4Click(Sender: TObject);
begin
 DATESEPARATOR:='.';
 SHORTDATEFORMAT:='DD/MM/YYYY';
 FORM5.CheckBox1.Enabled:=TRUE;
 FORM5.LbSpeedButton1.Enabled:=TRUE;
 IF FORM5.CheckBox1.Checked = FALSE THEN
 BEGIN
  FORM5.Panel1.Caption:='';
```

```
   FORM5.Edit1.Clear;
   form5.ADOQuery2.SQL.Text:='select
H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi,A.Hid,A.Aislemkodu from
hayvan H,asilama A where H.Hid=A.Hid order by TekrarAsilamaTarihi DESC';
   form5.ADOQuery2.Open;
  END
  ELSE
  BEGIN
   form5.ADOQuery2.SQL.Text:='select
H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi,A.Hid,A.Aislemkodu from
hayvan H,asilama A where H.Hid='+#39+form5.Edit1.Text+#39+' and H.Hid=A.Hid order by
TekrarAsilamaTarihi DESC';
   form5.ADOQuery2.Open;
  END;
  FORM5.DateTimePicker1.Date:=DATE;
  FORM5.Edit2.Clear;
  FORM5.DateTimePicker2.Date:=DATE;
  KD:='0';
  FORM5.Label6.Visible:=FALSE;
  FORM5.Label7.Visible:=FALSE;
  FORM5.Label8.Visible:=FALSE;
  FORM5.Label9.Visible:=FALSE;

end;

procedure TForm5.CheckBox1Click(Sender: TObject);
begin
  if form5.CheckBox1.Checked=true then
   form5.Label5.Visible:=true
  else
   form5.Label5.Visible:=false;
end;

procedure TForm5.DateTimePicker2Change(Sender: TObject);
begin
  if datetostr(form5.DateTimePicker1.Date) <> datetostr(form5.DateTimePicker2.Date) then
  begin
   form5.Label7.Visible:=false;
   form5.Label9.Visible:=false;
  end;
end;

procedure TForm5.Edit2Exit(Sender: TObject);
begin
  if form5.Edit2.Text <> '' then
  form5.Label8.Visible:=false;
end;
```

```
procedure TForm5.DateTimePicker1Change(Sender: TObject);
begin
 //if datetostr(form5.DateTimePicker1.Date) <> datetostr(form5.DateTimePicker2.Date) then
 //begin
   form5.Label7.Visible:=false;
   //form5.Label9.Visible:=false;
// end;
end;

procedure TForm5.LbSpeedButton2Click(Sender: TObject);
begin
 if (form5.Edit1.Text <> '') and (datetostr(form5.DateTimePicker1.Date) <>
datetostr(form5.DateTimePicker2.Date)) and (form5.Edit2.Text <> '') then
 begin
   dateseparator:='-';
   shortdateformat:='yyyy/mm/dd';
   form5.ADOQuery1.SQL.Text:='update asilama set Hid='+#39+form5.Edit1.Text+#39+',
AsilamaTarihi='+#39+datetostr(form5.DateTimePicker1.Date)+#39+',
YapilanAsilama='+#39+form5.Edit2.Text+#39+',
TekrarAsilamaTarihi='+#39+datetostr(form5.DateTimePicker2.Date)+#39+' where
Aislemkodu='+#39+KD+#39;
   form5.ADOQuery1.ExecSQL;
   showmessage('UPDATE PROCESS WAS DONE SUCCESSFULY');
   dateseparator:='.';
   shortdateformat:='DD/MM/YYYY';
   form5.ADOQuery2.SQL.Text:='select
H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi,A.Hid,A.Aislemkodu from
hayvan H,asilama A where H.Hid='+#39+form5.Edit1.Text+#39+' and H.Hid=A.Hid order by
TekrarAsilamaTarihi DESC';
   form5.ADOQuery2.Open;
 END
 ELSE
 BEGIN
   IF FORM5.Edit1.Text='' THEN//FORM5.DBGrid1.FieldCount = 0 THEN
     APPLICATION.MessageBox('PLEASE BEFORE SELECT ANIMAL, THEN SELECT
INFORMATION OF ANIMAL THAT YOU
SELECTED','ATTENTION',MB_OK+MB_ICONEXCLAMATION)
   ELSE IF FORM5.DBGrid1.FieldCount <> 0 THEN
   BEGIN
     IF KD='0' THEN//FORM5.DBGrid1.SelectedRows.CurrentRowSelected = FALSE THEN
       APPLICATION.MessageBox('PLEASE SELECT INFORMATION OF ANIMAL FROM
LIST FOR CORRECTION PROCESS','ATTENTION',MB_OK+MB_ICONEXCLAMATION);
     IF (DATETOSTR(form5.DateTimePicker1.Date) =
DATETOSTR(form5.DateTimePicker2.Date)) OR (FORM5.Edit1.Text ='') THEN
       APPLICATION.MessageBox('PLEASE CHECK
ERRORS','ERROR',MB_OK+MB_ICONEXCLAMATION);
```

```
    if datetostr(form5.DateTimePicker1.Date) = datetostr(form5.DateTimePicker2.Date) then
    begin
      form5.Label7.Visible:=true;
      form5.Label9.Visible:=true;
    end;
    if form5.Edit2.Text = '' then
      form5.Label8.Visible:=true;
  END;
 END;
end;

procedure TForm5.DBGrid1KeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
 if form5.DBGrid1.FieldCount <> 0 then
 begin
  //FORM5.CheckBox1.Checked:=FALSE;
  //FORM5.CheckBox1.Enabled:=FALSE;
  //FORM5.LbSpeedButton1.Enabled:=FALSE;
  DATESEPARATOR:='.';
  SHORTDATEFORMAT:='DD/MM/YYYY';
  FORM5.Panel1.Caption:=FORM5.DBGrid1.Fields[0].Text;
  FORM5.DateTimePicker1.Date:=STRTODATE(FORM5.DBGrid1.Fields[1].Text);
  FORM5.Edit2.Text:=FORM5.DBGrid1.Fields[2].Text;
  FORM5.DateTimePicker2.Date:=STRTODATE(FORM5.DBGrid1.Fields[3].Text);
  FORM5.Edit1.Text:=FORM5.DBGrid1.Fields[4].Text;
  KD:=FORM5.DBGrid1.Fields[5].Text;
 END
end;

procedure TForm5.LbSpeedButton3Click(Sender: TObject);
VAR
DRM:WORD;
begin
 IF FORM5.Edit1.Text='' THEN //FORM5.DBGrid1.FieldCount = 0 THEN
  APPLICATION.MessageBox('PLEASE BEFORE SELECT ANIMAL, THEN SELECT
INFORMATION OF ANIMAL THAT YOU
SELECTED','ATTENTION',MB_OK+MB_ICONEXCLAMATION)
 ELSE
 BEGIN
  if KD <> '0' then
  begin
   DRM:=APPLICATION.MessageBox('ARE YOU SURE TO DELETE SELECTED
RECORD?','CAUTION',MB_YESNO+MB_ICONEXCLAMATION);
    IF DRM=IDYES THEN
    BEGIN
```

```
      form5.ADOQuery1.SQL.Text:='delete from asilama where Aislemkodu='+#39+KD+#39;
      form5.ADOQuery1.ExecSQL;
      showmessage('DELETE PROCESS WAS DONE SUCCESSFULY');
      form5.ADOQuery2.SQL.Text:='select
H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi,A.Hid,A.Aislemkodu from
hayvan H,asilama A where H.Hid='+#39+form5.Edit1.Text+#39+' and H.Hid=A.Hid order by
TekrarAsilamaTarihi DESC';
      form5.ADOQuery2.Open;
      KD:='0';
      FORM5.Edit2.Clear;
      FORM5.DateTimePicker1.Date:=DATE;
      FORM5.DateTimePicker2.Date:=DATE;
      //form5.LbSpeedButton4.Click;
    END
    ELSE
    BEGIN
      KD:='0';
      FORM5.Edit2.Clear;
      FORM5.DateTimePicker1.Date:=DATE;
      FORM5.DateTimePicker2.Date:=DATE;
    END;
  end
  ELSE
    APPLICATION.MessageBox('PLEASE SELECT INFORMATION OF ANIMAL FROM
LIST TO DELETE','ATTENTION',MB_OK+MB_ICONEXCLAMATION);
 END;

end;

procedure TForm5.FormKeyDown(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
 if key = vk_f2 then
  form5.LbSpeedButton1.Click;//kaydet
 if key = vk_f3 then
  form5.LbSpeedButton2.Click;//düzelt
 if key = vk_f4 then
  form5.LbSpeedButton3.Click;//sil
 if key = vk_f5 then
  form5.LbSpeedButton4.Click;//temizle
 if key = vk_f6 then
  form5.LbSpeedButton5.Click;//ana menü
 if key = vk_escape then
  form5.Close;
end;

procedure TForm5.Hazirla1Click(Sender: TObject);
```

```
begin
FORM5.LbSpeedButton4.Click;
form5.Hide;
form1.LbSpeedButton1.Click;
end;

procedure TForm5.BugnnDetaylar1Click(Sender: TObject);
begin
FORM5.LbSpeedButton4.Click;
form5.Hide;
form16.LbSpeedButton1.Click;
end;

procedure TForm5.Parazit1Click(Sender: TObject);
begin
FORM5.LbSpeedButton4.Click;
form5.Hide;
form1.LbSpeedButton5.Click;
end;

procedure TForm5.DParazit1Click(Sender: TObject);
begin
FORM5.LbSpeedButton4.Click;
form5.Hide;
form1.LbSpeedButton4.Click;
end;

procedure TForm5.A1Click(Sender: TObject);
begin
form1.LbSpeedButton3.Click;
end;

procedure TForm5.MTERKAYIT1Click(Sender: TObject);
begin
FORM5.LbSpeedButton4.Click;
form5.Hide;
form3.LbSpeedButton1.Click;
end;

procedure TForm5.HAYVANKAYIT1Click(Sender: TObject);
begin
FORM5.LbSpeedButton4.Click;
form5.Hide;
form3.LbSpeedButton2.Click;
end;

procedure TForm5.MterismineGre1Click(Sender: TObject);
```

```
begin
FORM5.LbSpeedButton4.Click;
form5.Hide;
form13.LbSpeedButton6.Click;
end;

procedure TForm5.HayvanaGreArama1Click(Sender: TObject);
begin
FORM5.LbSpeedButton4.Click;
form5.Hide;
form13.LbSpeedButton1.Click;
end;

procedure TForm5.k1Click(Sender: TObject);
begin
form1.LbSpeedButton7.Click;
end;

procedure TForm5.Gizle1Click(Sender: TObject);
begin
form1.LbSpeedButton8.Click;
end;

procedure TForm5.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
 IF key=#13 then
   form5.DateTimePicker2.SetFocus;
end;

procedure TForm5.DateTimePicker1CloseUp(Sender: TObject);
begin
form5.Edit2.SetFocus;
end;

procedure TForm5.DBGrid1CellClick(Column: TColumn);
begin
 if (form5.DBGrid1.FieldCount <> 0) AND (FORM5.DBGrid1.Fields[0].Text <> '') then
 begin
   //FORM5.CheckBox1.Checked:=FALSE;
   //FORM5.CheckBox1.Enabled:=FALSE;
   //FORM5.LbSpeedButton1.Enabled:=FALSE;
   DATESEPARATOR:='.';
   SHORTDATEFORMAT:='DD/MM/YYYY';
   FORM5.Panel1.Caption:=FORM5.DBGrid1.Fields[0].Text;
   FORM5.DateTimePicker1.Date:=STRTODATE(FORM5.DBGrid1.Fields[1].Text);
   FORM5.Edit2.Text:=FORM5.DBGrid1.Fields[2].Text;
   FORM5.DateTimePicker2.Date:=STRTODATE(FORM5.DBGrid1.Fields[3].Text);
```

```
   FORM5.Edit1.Text:=FORM5.DBGrid1.Fields[4].Text;
   KD:=FORM5.DBGrid1.Fields[5].Text;
  END
end;

procedure TForm5.DBGrid1MouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
begin
  if (form5.Edit2.Text = '') and (form5.Edit2.Text = '') then
  BEGIN
   form5.DBGrid1.Hint:='WHEN YOU SELECTED A RECORD THE SYSTEM ACCEPT
ANIMAL NAME.'+#13+'THATS WHY YOU MUST SELECT RECORD THAT YOU WANT
AGAIN';
   FORM5.DBGrid1.ShowHint:=TRUE;
  END;
end;

end.
```

FORM 6 CODES

```
unit Unit6;

interface

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

type
  TForm6 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    DateTimePicker1: TDateTimePicker;
    DateTimePicker2: TDateTimePicker;
    LbSpeedButton1: TLbSpeedButton;
    LbSpeedButton2: TLbSpeedButton;
    LbSpeedButton3: TLbSpeedButton;
    LbSpeedButton4: TLbSpeedButton;
    LbSpeedButton5: TLbSpeedButton;
    Edit1: TEdit;
    Panel1: TPanel;
    SpeedButton1: TSpeedButton;
    Edit2: TEdit;
```

```
DBGrid1: TDBGrid;
Panel3: TPanel;
Panel4: TPanel;
Label5: TLabel;
Label8: TLabel;
Label9: TLabel;
CheckBox1: TCheckBox;
Label6: TLabel;
Label7: TLabel;
ADOQuery1: TADOQuery;
ADOQuery2: TADOQuery;
ADOQuery3: TADOQuery;
DataSource1: TDataSource;
DataSource2: TDataSource;
DataSource3: TDataSource;
MainMenu1: TMainMenu;
M1: TMenuItem;
MTERKAYIT1: TMenuItem;
HAYVANKAYIT1: TMenuItem;
A1: TMenuItem;
DParazit1: TMenuItem;
Parazit1: TMenuItem;
KaytArama1: TMenuItem;
MterismineGre1: TMenuItem;
HayvanaGreArama1: TMenuItem;
Hazirla1: TMenuItem;
BugnnDetaylar1: TMenuItem;
Gizle1: TMenuItem;
k1: TMenuItem;
procedure LbSpeedButton5Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure SpeedButton1Click(Sender: TObject);
procedure LbSpeedButton1Click(Sender: TObject);
procedure LbSpeedButton2Click(Sender: TObject);
procedure DBGrid1CellClick(Column: TColumn);
procedure Edit1Change(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure LbSpeedButton4Click(Sender: TObject);
procedure LbSpeedButton3Click(Sender: TObject);
procedure DateTimePicker1Change(Sender: TObject);
procedure Edit2Exit(Sender: TObject);
procedure DateTimePicker2Change(Sender: TObject);
procedure CheckBox1Click(Sender: TObject);
procedure DBGrid1KeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
```

```
    procedure MTERKAYIT1Click(Sender: TObject);
    procedure HAYVANKAYIT1Click(Sender: TObject);
    procedure A1Click(Sender: TObject);
    procedure DParazit1Click(Sender: TObject);
    procedure Parazit1Click(Sender: TObject);
    procedure MterismineGre1Click(Sender: TObject);
    procedure HayvanaGreArama1Click(Sender: TObject);
    procedure Hazirla1Click(Sender: TObject);
    procedure BugnnDetaylar1Click(Sender: TObject);
    procedure k1Click(Sender: TObject);
    procedure Gizle1Click(Sender: TObject);
    procedure DateTimePicker1CloseUp(Sender: TObject);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form6: TForm6;
  DKD:string;
implementation

uses Unit1, Unit8, Unit5, Unit2, Unit16, Unit3, Unit13;

{$R *.dfm}

procedure TForm6.LbSpeedButton5Click(Sender: TObject);
begin
FORM6.Hide;
FORM1.SHOW;
end;

procedure TForm6.FormClose(Sender: TObject; var Action: TCloseAction);
begin
FORM6.CheckBox1.Checked:=FALSE;
FORM6.Hide;
FORM1.SHOW;
end;

procedure TForm6.SpeedButton1Click(Sender: TObject);
begin
hyv:=6;
FORM8.SHOW;
end;
```

```
procedure TForm6.LbSpeedButton1Click(Sender: TObject);
var
 dr,gn,ay,yil,saat,dak,sn,sl:word;
 ta:string;
begin
 if (form6.Edit1.Text <> '') and (datetostr(form6.DateTimePicker1.Date) <>
datetostr(form6.DateTimePicker2.Date)) and (form6.Edit2.Text <> '') then
 begin
  dateseparator:='-';
  shortdateformat:='yyyy/mm/dd';

  decodedate(date, yil, ay, gn);
  decodetime(time,saat,dak,sn,sl);

ta:='D'+inttostr(gn)+inttostr(ay)+inttostr(yil)+inttostr(saat)+inttostr(dak)+inttostr(sn)+inttostr(sl);

  form6.ADOQuery1.SQL.Text:='select * from disparazit';
  form6.ADOQuery1.Open;
  form6.ADOQuery1.Insert;
  form6.ADOQuery1.FieldByName('Hid').AsString:=form6.Edit1.Text;

form6.ADOQuery1.FieldByName('DisParUygTarihi').AsString:=datetostr(form6.DateTimePicke
r1.Date);
  form6.ADOQuery1.FieldByName('DisParUygulama').AsString:=form6.Edit2.Text;

form6.ADOQuery1.FieldByName('TekrarDisParUygTarihi').AsString:=datetostr(form6.DateTim
ePicker2.Date);
  form6.ADOQuery1.FieldByName('Dislemkodu').AsString:=ta;

  form6.ADOQuery3.SQL.Text:='select * from disparazit where
Hid='+#39+form6.Edit1.Text+#39+' and
DisParUygTarihi='+#39+datetostr(form6.DateTimePicker1.Date)+#39+' and
DisParUygulama='+#39+form6.Edit2.Text+#39+' and
TekrarDisParUygTarihi='+#39+datetostr(form6.DateTimePicker2.Date)+#39;
  form6.ADOQuery3.Open;
  if form6.ADOQuery3.RecordCount = 0 then
  begin
   form6.ADOQuery1.Post;
   SHOWMESSAGE('THE RECORD SAVED SUCCESSFULY'+#13+'ACCEPTANCE
CODE:'+TA);
   FORM6.LbSpeedButton4.Click;
  end
  else
  begin
   dr:=application.MessageBox('THE INFORMATIONS SAVED BEFORE'+#13+'STILL DO
YOU WANT TO SAVE AGAIN?','ATTENTION',MB_YESNO+MB_ICONEXCLAMATION);
   IF DR=IDYES THEN
```

```
    BEGIN
     form6.ADOQuery1.Post;
     SHOWMESSAGE('THE RECORD SAVED SUCCESSFULY'+#13+'ACCEPTANCE
CODE:'+TA);
     FORM6.LbSpeedButton4.Click;
    END;
   END;
  END
  ELSE
  begin
   APPLICATION.MessageBox('PLEASE FILL EMPTY
PLACE','CAUTION',MB_OK+MB_ICONEXCLAMATION);
   if form6.Edit1.Text = '' then
    form6.Label6.Visible:=true;

   if datetostr(form6.DateTimePicker1.Date) = datetostr(form6.DateTimePicker2.Date) then
   begin
    //showmessage('1: '+datetostr(form5.DateTimePicker1.Date)+#13+#13+'2:
'+datetostr(form5.DateTimePicker2.Date));
    form6.Label7.Visible:=true;
    form6.Label9.Visible:=true;
   end;
   if form6.Edit2.Text = '' then
    form6.Label8.Visible:=true;
  end;
end;

procedure TForm6.LbSpeedButton2Click(Sender: TObject);
begin
if (form6.Edit1.Text <> '') and (datetostr(form6.DateTimePicker1.Date) <>
datetostr(form6.DateTimePicker2.Date)) and (form6.Edit2.Text <> '') then
 begin
  dateseparator:='-';
  shortdateformat:='yyyy/mm/dd';
  form6.ADOQuery1.SQL.Text:='update disparazit set Hid='+#39+form6.Edit1.Text+#39+',
DisParUygTarihi='+#39+datetostr(form6.DateTimePicker1.Date)+#39+',
DisParUygulama='+#39+form6.Edit2.Text+#39+',
TekrarDisParUygTarihi='+#39+datetostr(form6.DateTimePicker2.Date)+#39+' where
Dislemkodu='+#39+DKD+#39;
  form6.ADOQuery1.ExecSQL;
  dateseparator:='.';
  shortdateformat:='DD/MM/YYYY';
  form6.ADOQuery2.SQL.Text:='select
H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi,D.Hid,D.Dislemkod
e from hayvan H,disparazit D where H.Hid='+#39+form6.Edit1.Text+#39+' and H.Hid=D.Hid
order by TekrarDisParUygTarihi DESC';
  form6.ADOQuery2.Open;
```

```
  showmessage('UPDATE PROCESS WAS DONE SUCCESSFULY');
 END
 ELSE
 BEGIN
  IF FORM6.Edit1.Text='' THEN//FORM6.DBGrid1.FieldCount = 0 THEN
   APPLICATION.MessageBox('PLEASE BEFORE SELECT ANIMAL, THEN SELECT
INFORMATION OF ANIMAL THAT YOU
SELECTED','ATTENTION',MB_OK+MB_ICONEXCLAMATION)
  ELSE IF FORM6.DBGrid1.FieldCount <> 0 THEN
  BEGIN
   IF DKD = '0' THEN//FORM6.DBGrid1.SelectedRows.CurrentRowSelected = FALSE THEN
    APPLICATION.MessageBox('PLEASE SELECT INFORMATION OF ANIMAL FROM
LIST FOR CORRECTION PROCESS','ATTENTION',MB_OK+MB_ICONEXCLAMATION);
   IF (DATETOSTR(form6.DateTimePicker1.Date) =
DATETOSTR(form6.DateTimePicker2.Date)) OR (FORM6.Edit1.Text ='') THEN
    APPLICATION.MessageBox('PLEASE CHECK
ERRORS','ERROR',MB_OK+MB_ICONEXCLAMATION);

   //ELSE IF {(FORM5.DBGrid1.SelectedRows.CurrentRowSelected = TRUE) AND}
(DATETOSTR(form5.DateTimePicker1.Date) = DATETOSTR(form5.DateTimePicker2.Date))
and (form5.Edit2.Text = '') OR {(FORM5.DBGrid1.SelectedRows.CurrentRowSelected =
TRUE) AND} (DATETOSTR(form5.DateTimePicker1.Date) =
DATETOSTR(form5.DateTimePicker2.Date)) THEN
    {BEGIN
     APPLICATION.MessageBox('LÜTFEN HATALI KISIMLARI KONTROL
EDİNİZ','HATALARI KONTROL EDİNİZ...',MB_OK+MB_ICONEXCLAMATION);
     if form5.Edit1.Text = '' then
      form5.Label6.Visible:=true;}

     if datetostr(form5.DateTimePicker1.Date) = datetostr(form5.DateTimePicker2.Date) then
     begin
      //showmessage('1: '+datetostr(form5.DateTimePicker1.Date)+#13+#13+'2:
'+datetostr(form5.DateTimePicker2.Date));
      form6.Label7.Visible:=true;
      form6.Label9.Visible:=true;
     end;
    if form6.Edit2.Text = '' then
      form6.Label8.Visible:=true;
   // END;

  END;
 END;
end;

procedure TForm6.DBGrid1CellClick(Column: TColumn);
begin
 if (form6.DBGrid1.FieldCount <> 0) AND (FORM6.DBGrid1.Fields[0].Text <> '') then
```

```
 begin
  //FORM6.CheckBox1.Checked:=FALSE;
  //FORM6.CheckBox1.Enabled:=FALSE;
  //FORM6.LbSpeedButton1.Enabled:=FALSE;
  DATESEPARATOR:='.';
  SHORTDATEFORMAT:='DD/MM/YYYY';
  FORM6.Panel1.Caption:=FORM6.DBGrid1.Fields[0].Text;
  FORM6.DateTimePicker1.Date:=STRTODATE(FORM6.DBGrid1.Fields[1].Text);
  FORM6.Edit2.Text:=FORM6.DBGrid1.Fields[2].Text;
  FORM6.DateTimePicker2.Date:=STRTODATE(FORM6.DBGrid1.Fields[3].Text);
  FORM6.Edit1.Text:=FORM6.DBGrid1.Fields[4].Text;
  DKD:=FORM6.DBGrid1.Fields[5].Text;
 END
end;

procedure TForm6.Edit1Change(Sender: TObject);
begin
 if form6.Edit1.Text <> '' then
 begin
  form6.ADOQuery2.SQL.Text:='select
H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi,D.Hid,D.Dislemkod
u from hayvan H,disparazit D where H.Hid='+#39+form6.Edit1.Text+#39+'and H.Hid=D.Hid
order by TekrarDisParUygTarihi DESC';
  form6.ADOQuery2.Open;
  form6.Label6.Visible:=false;
 end;
end;

procedure TForm6.FormShow(Sender: TObject);
begin
 form6.DateTimePicker1.Date:=date;
 form6.DateTimePicker2.Date:=date;
 DKD:='0';
 FORM6.LbSpeedButton4.Click;
 FORM6.Panel1.SetFocus;
end;

procedure TForm6.LbSpeedButton4Click(Sender: TObject);
begin
 DATESEPARATOR:='.';
 SHORTDATEFORMAT:='DD/MM/YYYY';
 FORM6.CheckBox1.Enabled:=TRUE;
 FORM6.LbSpeedButton1.Enabled:=TRUE;
 IF FORM6.CheckBox1.Checked = FALSE THEN
 BEGIN
  FORM6.Panel1.Caption:='';
  FORM6.Edit1.Clear;
```

```
  form6.ADOQuery2.SQL.Text:='select
H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi,D.Hid,D.Dislemkod
u from hayvan H,disparazit D where H.Hid=D.Hid order by TekrarDisParUygTarihi DESC';
  form6.ADOQuery2.Open;
 END
 ELSE
 BEGIN
  form6.ADOQuery2.SQL.Text:='select
H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi,D.Hid,D.Dislemkod
u from hayvan H,disparazit D where H.Hid='+#39+form6.Edit1.Text+#39+' and H.Hid=D.Hid
order by TekrarDisParUygTarihi DESC';
   form6.ADOQuery2.Open;
  END;
  FORM6.DateTimePicker1.Date:=DATE;
  FORM6.Edit2.Clear;
  FORM6.DateTimePicker2.Date:=DATE;
  DKD:='0';
  FORM6.Label6.Visible:=FALSE;
  FORM6.Label7.Visible:=FALSE;
  FORM6.Label8.Visible:=FALSE;
  FORM6.Label9.Visible:=FALSE;

end;

procedure TForm6.LbSpeedButton3Click(Sender: TObject);

VAR
DRM:WORD;
begin
  IF FORM6.Edit1.Text='' THEN// FORM6.DBGrid1.FieldCount = 0 THEN
   APPLICATION.MessageBox('PLEASE BEFORE SELECT ANIMAL, THEN SELECT
INFORMATION OF ANIMAL THAT YOU
SELECTED','ATTENTION',MB_OK+MB_ICONEXCLAMATION)
  ELSE
  BEGIN
   if DKD <> '0' then
   begin
    DRM:=APPLICATION.MessageBox('ARE YOU SURE TO DELETE SELECTED
RECORD?','CAUTION',MB_YESNO+MB_ICONEXCLAMATION);
    IF DRM=IDYES THEN
    BEGIN
     form6.ADOQuery1.SQL.Text:='delete from disparazit where
Dislemkodu='+#39+DKD+#39;
     form6.ADOQuery1.ExecSQL;
     showmessage('DELETE PROCESS WAS DONE SUCCESSFULY');
     form6.ADOQuery2.SQL.Text:='select
H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi,D.Hid,D.Dislemkod
```

```
u from hayvan H,disparazit D where H.Hid='+#39+form6.Edit1.Text+#39+' and H.Hid=D.Hid
order by TekrarDisParUygTarihi DESC';
      form6.ADOQuery2.Open;
      DKD:='0';
      FORM6.Edit2.Clear;
      FORM6.DateTimePicker1.Date:=DATE;
      FORM6.DateTimePicker2.Date:=DATE;
      //form5.LbSpeedButton4.Click;
    END
    ELSE
    BEGIN
      DKD:='0';
      FORM6.Edit2.Clear;
      FORM6.DateTimePicker1.Date:=DATE;
      FORM6.DateTimePicker2.Date:=DATE;
    END;
  end
  ELSE
    APPLICATION.MessageBox('PLEASE SELECT INFORMATION OF ANIMAL FROM
LIST TO DELETE','ATTENTION',MB_OK+MB_ICONEXCLAMATION);
 END;
end;

procedure TForm6.DateTimePicker1Change(Sender: TObject);
begin
  form6.Label7.Visible:=false;
end;

procedure TForm6.Edit2Exit(Sender: TObject);
begin
 if form6.Edit2.Text <> '' then
 BEGIN
  form6.Label8.Visible:=false;
 END;
end;

procedure TForm6.DateTimePicker2Change(Sender: TObject);
begin
 if datetostr(form6.DateTimePicker1.Date) <> datetostr(form6.DateTimePicker2.Date) then
 begin
  form6.Label7.Visible:=false;
  form6.Label9.Visible:=false;
 end;
end;

procedure TForm6.CheckBox1Click(Sender: TObject);
begin
```

```
 if form6.CheckBox1.Checked=true then
   form6.Label5.Visible:=true
 else
   form6.Label5.Visible:=false;
end;

procedure TForm6.DBGrid1KeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
 if form6.DBGrid1.FieldCount <> 0 then
 begin
   //FORM6.CheckBox1.Checked:=FALSE;
   //FORM6.CheckBox1.Enabled:=FALSE;
   //FORM6.LbSpeedButton1.Enabled:=FALSE;
   DATESEPARATOR:='.';
   SHORTDATEFORMAT:='DD/MM/YYYY';
   FORM6.Panel1.Caption:=FORM6.DBGrid1.Fields[0].Text;
   FORM6.DateTimePicker1.Date:=STRTODATE(FORM6.DBGrid1.Fields[1].Text);
   FORM6.Edit2.Text:=FORM6.DBGrid1.Fields[2].Text;
   FORM6.DateTimePicker2.Date:=STRTODATE(FORM6.DBGrid1.Fields[3].Text);
   FORM6.Edit1.Text:=FORM6.DBGrid1.Fields[4].Text;
   DKD:=FORM6.DBGrid1.Fields[5].Text;
 END
end;

procedure TForm6.FormKeyDown(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
 if key = vk_f2 then
   form6.LbSpeedButton1.Click;//kaydet
 if key = vk_f3 then
   form6.LbSpeedButton2.Click;//düzelt
 if key = vk_f4 then
   form6.LbSpeedButton3.Click;//sil
 if key = vk_f5 then
   form6.LbSpeedButton4.Click;//temizle
 if key = vk_f6 then
   form6.LbSpeedButton5.Click;//ana menü
 if key = vk_escape then
   form6.Close;
end;

procedure TForm6.Hazirla1Click(Sender: TObject);
begin
form6.LbSpeedButton4.Click;
form6.Hide;
form1.LbSpeedButton1.Click;
```

```
end;

procedure TForm6.BugnnDetaylar1Click(Sender: TObject);
begin
form6.LbSpeedButton4.Click;
form6.Hide;
form16.LbSpeedButton1.Click;
end;

procedure TForm6.Parazit1Click(Sender: TObject);
begin
form6.LbSpeedButton4.Click;
form6.Hide;
form1.LbSpeedButton5.Click;
end;

procedure TForm6.DParazit1Click(Sender: TObject);
begin
form1.LbSpeedButton4.Click;
end;

procedure TForm6.A1Click(Sender: TObject);
begin
form6.LbSpeedButton4.Click;
form6.Hide;
form1.LbSpeedButton3.Click;
end;

procedure TForm6.MTERKAYIT1Click(Sender: TObject);
begin
form6.LbSpeedButton4.Click;
form6.Hide;
form3.LbSpeedButton1.Click;
end;


procedure TForm6.MterismineGre1Click(Sender: TObject);
begin
form6.LbSpeedButton4.Click;
form6.Hide;
form13.LbSpeedButton6.Click;
end;

procedure TForm6.HayvanaGreArama1Click(Sender: TObject);
begin
form6.LbSpeedButton4.Click;
form6.Hide;
```

```
form13.LbSpeedButton1.Click;
end;

procedure TForm6.k1Click(Sender: TObject);
begin
form1.LbSpeedButton7.Click;
end;


procedure TForm6.HAYVANKAYIT1Click(Sender: TObject);
begin
form6.LbSpeedButton4.Click;
form6.Hide;
form3.LbSpeedButton2.Click;
end;


procedure TForm6.Gizle1Click(Sender: TObject);
begin
form1.LbSpeedButton8.Click;
end;

procedure TForm6.DateTimePicker1CloseUp(Sender: TObject);
begin
  FORM6.Edit2.SetFocus;
end;

procedure TForm6.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
  if key=#13 then
    FORM6.DateTimePicker2.SetFocus;
end;

end.
```

FORM 7 CODES

```
unit Unit7;

interface

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

type
```

```
TForm7 = class(TForm)
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Label4: TLabel;
  DateTimePicker1: TDateTimePicker;
  DateTimePicker2: TDateTimePicker;
  LbSpeedButton1: TLbSpeedButton;
  LbSpeedButton2: TLbSpeedButton;
  LbSpeedButton3: TLbSpeedButton;
  LbSpeedButton4: TLbSpeedButton;
  LbSpeedButton5: TLbSpeedButton;
  Edit1: TEdit;
  Panel1: TPanel;
  SpeedButton1: TSpeedButton;
  Edit2: TEdit;
  DBGrid1: TDBGrid;
  Panel3: TPanel;
  Panel4: TPanel;
  DataSource1: TDataSource;
  Label5: TLabel;
  Label8: TLabel;
  CheckBox1: TCheckBox;
  Label6: TLabel;
  Label7: TLabel;
  Label9: TLabel;
  ADOQuery1: TADOQuery;
  ADOQuery2: TADOQuery;
  ADOQuery3: TADOQuery;
  DataSource2: TDataSource;
  DataSource3: TDataSource;
  MainMenu1: TMainMenu;
  M1: TMenuItem;
  MTERKAYIT1: TMenuItem;
  HAYVANKAYIT1: TMenuItem;
  A1: TMenuItem;
  DParazit1: TMenuItem;
  Parazit1: TMenuItem;
  KaytArama1: TMenuItem;
  MterismineGre1: TMenuItem;
  HayvanaGreArama1: TMenuItem;
  Hazirla1: TMenuItem;
  BugnnDetaylar1: TMenuItem;
  Gizle1: TMenuItem;
  k1: TMenuItem;
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure LbSpeedButton5Click(Sender: TObject);
```

```
    procedure SpeedButton1Click(Sender: TObject);
    procedure LbSpeedButton1Click(Sender: TObject);
    procedure LbSpeedButton2Click(Sender: TObject);
    procedure DBGrid1CellClick(Column: TColumn);
    procedure Edit1Change(Sender: TObject);
    procedure LbSpeedButton3Click(Sender: TObject);
    procedure LbSpeedButton4Click(Sender: TObject);
    procedure CheckBox1Click(Sender: TObject);
    procedure DateTimePicker1Change(Sender: TObject);
    procedure Edit2Exit(Sender: TObject);
    procedure DateTimePicker2Change(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure A1Click(Sender: TObject);
    procedure MTERKAYIT1Click(Sender: TObject);
    procedure HAYVANKAYIT1Click(Sender: TObject);
    procedure DParazit1Click(Sender: TObject);
    procedure Parazit1Click(Sender: TObject);
    procedure MterismineGre1Click(Sender: TObject);
    procedure HayvanaGreArama1Click(Sender: TObject);
    procedure Hazirla1Click(Sender: TObject);
    procedure BugnnDetaylar1Click(Sender: TObject);
    procedure k1Click(Sender: TObject);
    procedure Gizle1Click(Sender: TObject);
    procedure DBGrid1KeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure DateTimePicker1CloseUp(Sender: TObject);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form7: TForm7;
  iKD:string;
implementation

uses Unit1, Unit8, Unit2, Unit5, Unit6, Unit16, Unit3, Unit13;

{$R *.dfm}

procedure TForm7.FormClose(Sender: TObject; var Action: TCloseAction);
begin
FORM7.CheckBox1.Checked:=FALSE;
```

```
FORM7.Hide;
FORM1.SHOW;
end;

procedure TForm7.LbSpeedButton5Click(Sender: TObject);
begin
FORM7.Hide;
FORM1.SHOW;
end;

procedure TForm7.SpeedButton1Click(Sender: TObject);
begin
hyv:=7;
FORM8.SHOW;
end;

procedure TForm7.LbSpeedButton1Click(Sender: TObject);
var
  dr,gn,ay,yil,saat,dak,sn,sl:word;
  ta:string;
begin
  if (form7.Edit1.Text <> '') and (datetostr(form7.DateTimePicker1.Date) <>
datetostr(form7.DateTimePicker2.Date)) and (form7.Edit2.Text <> '') then
  begin
    dateseparator:='-';
    shortdateformat:='yyyy/mm/dd';

    decodedate(date, yil, ay, gn);
    decodetime(time,saat,dak,sn,sl);

ta:='I'+inttostr(gn)+inttostr(ay)+inttostr(yil)+inttostr(saat)+inttostr(dak)+inttostr(sn)+inttostr(sl);

    form7.ADOQuery1.SQL.Text:='select * from icparazit';
    form7.ADOQuery1.Open;
    form7.ADOQuery1.Insert;
    form7.ADOQuery1.FieldByName('Hid').AsString:=form7.Edit1.Text;

form7.ADOQuery1.FieldByName('IcParUygTarihi').AsString:=datetostr(form7.DateTimePicker
1.Date);
    form7.ADOQuery1.FieldByName('IcParUygulama').AsString:=form7.Edit2.Text;

form7.ADOQuery1.FieldByName('TekrarIcParUygTarihi').AsString:=datetostr(form7.DateTime
Picker2.Date);
    form7.ADOQuery1.FieldByName('Iislemkodu').AsString:=ta;

    form7.ADOQuery3.SQL.Text:='select * from icparazit where
Hid='+#39+form7.Edit1.Text+#39+' and
```

```
IcParUygTarihi='+#39+datetostr(form7.DateTimePicker1.Date)+#39+' and
IcParUygulama='+#39+form7.Edit2.Text+#39+' and
TekrarIcParUygTarihi='+#39+datetostr(form7.DateTimePicker2.Date)+#39;
   form7.ADOQuery3.Open;
   if form7.ADOQuery3.RecordCount = 0 then
   begin
    form7.ADOQuery1.Post;
    SHOWMESSAGE('THE RECORD SAVED SUCCESSFULY'+#13+'ACCEPTANCE
CODE:'+TA);
    FORM7.LbSpeedButton4.Click;
   end
   else
   begin
    dr:=application.MessageBox('THE INFORMATIONS SAVED BEFORE'+#13+'STILL DO
YOU WANT TO SAVE AGAIN?','ATTENTION',MB_YESNO+MB_ICONEXCLAMATION);
    IF DR=IDYES THEN
    BEGIN
     form7.ADOQuery1.Post;
     SHOWMESSAGE('THE RECORD SAVED SUCCESSFULY'+#13+'ACCEPTANCE
CODE:'+TA);
     FORM7.LbSpeedButton4.Click;
    END;
   END;
  END
  ELSE
  begin
   APPLICATION.MessageBox('PLEASE FILL EMPTY
PLACE','CAUTION',MB_OK+MB_ICONEXCLAMATION);
   if form7.Edit1.Text = '' then
    form7.Label6.Visible:=true;

   if datetostr(form7.DateTimePicker1.Date) = datetostr(form7.DateTimePicker2.Date) then
   begin
    //showmessage('1: '+datetostr(form5.DateTimePicker1.Date)+#13+#13+'2:
'+datetostr(form5.DateTimePicker2.Date));
    form7.Label7.Visible:=true;
    form7.Label9.Visible:=true;
   end;
   if form7.Edit2.Text = '' then
    form7.Label8.Visible:=true;
  end;
end;

procedure TForm7.LbSpeedButton2Click(Sender: TObject);
begin
if (form7.Edit1.Text <> '') and (datetostr(form7.DateTimePicker1.Date) <>
datetostr(form7.DateTimePicker2.Date)) and (form7.Edit2.Text <> '') then
```

```
 begin
   dateseparator:='-';
   shortdateformat:='yyyy/mm/dd';
   form7.ADOQuery1.SQL.Text:='update icparazit set Hid='+#39+form7.Edit1.Text+#39+',
IcParUygTarihi='+#39+datetostr(form7.DateTimePicker1.Date)+#39+',
IcParUygulama='+#39+form7.Edit2.Text+#39+',
TekrarIcParUygTarihi='+#39+datetostr(form7.DateTimePicker2.Date)+#39+' where
Iislemkodu='+#39+iKD+#39;
   form7.ADOQuery1.ExecSQL;
   dateseparator:='.';
   shortdateformat:='DD/MM/YYYY';
   form7.ADOQuery2.SQL.Text:='select
H.Hismi,i.IcParUygTarihi,i.IcParUygulama,i.TekrarIcParUygTarihi,i.Hid,i.Iislemkodu from
hayvan H,icparazit i where H.Hid='+#39+form7.Edit1.Text+#39+' and H.Hid=i.Hid order by
TekraricParUygTarihi DESC';
   form7.ADOQuery2.Open;
   showmessage('UPDATE PROCESS WAS DONE SUCCESSFULY');
 END
 ELSE
 BEGIN
   IF FORM7.Edit1.Text='' THEN//FORM7.DBGrid1.FieldCount = 0 THEN
    APPLICATION.MessageBox('PLEASE BEFORE SELECT ANIMAL, THEN SELECT
INFORMATION OF ANIMAL THAT YOU
SELECTED','ATTENTION',MB_OK+MB_ICONEXCLAMATION)
   ELSE IF FORM7.DBGrid1.FieldCount <> 0 THEN
   BEGIN
    IF iKD = '0' THEN//FORM6.DBGrid1.SelectedRows.CurrentRowSelected = FALSE THEN
     APPLICATION.MessageBox('PLEASE SELECT INFORMATION OF ANIMAL FROM
LIST FOR CORRECTION PROCESS','ATTENTION',MB_OK+MB_ICONEXCLAMATION);
    IF (DATETOSTR(form7.DateTimePicker1.Date) =
DATETOSTR(form7.DateTimePicker2.Date)) OR (FORM7.Edit1.Text ='') THEN
     APPLICATION.MessageBox('PLEASE CHECK
ERRORS','ERROR',MB_OK+MB_ICONEXCLAMATION);

    //ELSE IF {(FORM5.DBGrid1.SelectedRows.CurrentRowSelected = TRUE) AND}
(DATETOSTR(form5.DateTimePicker1.Date) = DATETOSTR(form5.DateTimePicker2.Date))
and (form5.Edit2.Text = '') OR {(FORM5.DBGrid1.SelectedRows.CurrentRowSelected =
TRUE) AND} (DATETOSTR(form5.DateTimePicker1.Date) =
DATETOSTR(form5.DateTimePicker2.Date)) THEN
     {BEGIN
      APPLICATION.MessageBox('LÜTFEN HATALI KISIMLARI KONTROL
EDİNİZ','HATALARI KONTROL EDİNİZ...',MB_OK+MB_ICONEXCLAMATION);
      if form5.Edit1.Text = '' then
       form5.Label6.Visible:=true;}

      if datetostr(form7.DateTimePicker1.Date) = datetostr(form7.DateTimePicker2.Date) then
      begin
```

```
        //showmessage('1: '+datetostr(form5.DateTimePicker1.Date)+#13+#13+'2:
'+datetostr(form5.DateTimePicker2.Date));
         form7.Label7.Visible:=true;
         form7.Label9.Visible:=true;
       end;
      if form7.Edit2.Text = '' then
         form7.Label8.Visible:=true;
     // END;

   END;
  END;
end;

procedure TForm7.DBGrid1CellClick(Column: TColumn);
begin
 if (form7.DBGrid1.FieldCount <> 0) AND (FORM7.DBGrid1.Fields[0].Text <> '') then
 begin
   //FORM7.CheckBox1.Checked:=FALSE;
   //FORM7.CheckBox1.Enabled:=FALSE;
   //FORM7.LbSpeedButton1.Enabled:=FALSE;
   DATESEPARATOR:='.';
   SHORTDATEFORMAT:='DD/MM/YYYY';
   FORM7.Panel1.Caption:=FORM7.DBGrid1.Fields[0].Text;
   FORM7.DateTimePicker1.Date:=STRTODATE(FORM7.DBGrid1.Fields[1].Text);
   FORM7.Edit2.Text:=FORM7.DBGrid1.Fields[2].Text;
   FORM7.DateTimePicker2.Date:=STRTODATE(FORM7.DBGrid1.Fields[3].Text);
   FORM7.Edit1.Text:=FORM7.DBGrid1.Fields[4].Text;
   iKD:=FORM7.DBGrid1.Fields[5].Text;
  END
end;

procedure TForm7.Edit1Change(Sender: TObject);
begin
  if form7.Edit1.Text <> '' then
  begin
    form7.ADOQuery2.SQL.Text:='select
H.Hismi,i.IcParUygTarihi,i.IcParUygulama,i.TekrarIcParUygTarihi,i.Hid,i.Iislemkodu from
hayvan H,icparazit i where H.Hid='+#39+form7.Edit1.Text+#39+' and H.Hid=i.Hid order by
TekraricParUygTarihi DESC';
    form7.ADOQuery2.Open;
    form7.Label6.Visible:=false;
  end;
end;

procedure TForm7.LbSpeedButton3Click(Sender: TObject);
VAR
DRM:WORD;
```

```
begin
 IF FORM7.Edit1.Text='' THEN//FORM7.DBGrid1.FieldCount = 0 THEN
  APPLICATION.MessageBox('PLEASE BEFORE SELECT ANIMAL, THEN SELECT
INFORMATION OF ANIMAL THAT YOU
SELECTED','ATTENTION',MB_OK+MB_ICONEXCLAMATION)
 ELSE
 BEGIN
  if iKD <> '0' then
  begin
   DRM:=APPLICATION.MessageBox('ARE YOU SURE TO DELETE SELECTED
RECORD?','CAUTION',MB_YESNO+MB_ICONEXCLAMATION);
   IF DRM=IDYES THEN
   BEGIN
    form7.ADOQuery1.SQL.Text:='delete from icparazit where Iislemkodu='+#39+iKD+#39;
    form7.ADOQuery1.ExecSQL;
    showmessage('DELETE PROCESS WAS DONE SUCCESSFULY');
    form7.ADOQuery2.SQL.Text:='select
H.Hismi,i.IcParUygTarihi,i.IcParUygulama,i.TekrarIcParUygTarihi,i.Hid,i.Iislemkodu from
hayvan H,icparazit i where H.Hid='+#39+form7.Edit1.Text+#39+' and H.Hid=i.Hid order by
TekraricParUygTarihi DESC';
    form7.ADOQuery2.Open;
    iKD:='0';
    FORM7.Edit2.Clear;
    FORM7.DateTimePicker1.Date:=DATE;
    FORM7.DateTimePicker2.Date:=DATE;
    //form5.LbSpeedButton4.Click;
   END
   ELSE
   BEGIN
    iKD:='0';
    FORM7.Edit2.Clear;
    FORM7.DateTimePicker1.Date:=DATE;
    FORM7.DateTimePicker2.Date:=DATE;
   END;
  end
  ELSE
   APPLICATION.MessageBox('PLEASE SELECT INFORMATION OF ANIMAL FROM
LIST TO DELETE','ATTENTION',MB_OK+MB_ICONEXCLAMATION);
 END;

end;

procedure TForm7.LbSpeedButton4Click(Sender: TObject);
begin
 DATESEPARATOR:='.';
 SHORTDATEFORMAT:='DD/MM/YYYY';
 FORM7.CheckBox1.Enabled:=TRUE;
```

```
  FORM7.LbSpeedButton1.Enabled:=TRUE;
  IF FORM7.CheckBox1.Checked = FALSE THEN
  BEGIN
    FORM7.Panel1.Caption:='';
    FORM7.Edit1.Clear;
    form7.ADOQuery2.SQL.Text:='select
H.Hismi,i.IcParUygTarihi,i.IcParUygulama,i.TekrarIcParUygTarihi,i.Hid,i.Iislemkodu from
hayvan H,icparazit i where H.Hid=i.Hid order by TekraricParUygTarihi DESC';
    form7.ADOQuery2.Open;
  END
  ELSE
  BEGIN
    form7.ADOQuery2.SQL.Text:='select
H.Hismi,i.IcParUygTarihi,i.IcParUygulama,i.TekrarIcParUygTarihi,i.Hid,i.Iislemkodu from
hayvan H,icparazit i where H.Hid='+#39+form7.Edit1.Text+#39+' and H.Hid=i.Hid order by
TekraricParUygTarihi DESC';
    form7.ADOQuery2.Open;
  END;
  FORM7.DateTimePicker1.Date:=DATE;
  FORM7.Edit2.Clear;
  FORM7.DateTimePicker2.Date:=DATE;
  iKD:='0';
  FORM7.Label6.Visible:=FALSE;
  FORM7.Label7.Visible:=FALSE;
  FORM7.Label8.Visible:=FALSE;
  FORM7.Label9.Visible:=FALSE;
end;

procedure TForm7.CheckBox1Click(Sender: TObject);
begin
  if form7.CheckBox1.Checked=true then
    form7.Label5.Visible:=true
  else
    form7.Label5.Visible:=false;
end;

procedure TForm7.DateTimePicker1Change(Sender: TObject);
begin
    form7.Label7.Visible:=false;
end;

procedure TForm7.Edit2Exit(Sender: TObject);
begin
  if form7.Edit2.Text <> '' then
  form7.Label8.Visible:=false;
end;
```

```
procedure TForm7.DateTimePicker2Change(Sender: TObject);
begin
 if datetostr(form7.DateTimePicker1.Date) <> datetostr(form7.DateTimePicker2.Date) then
 begin
  form7.Label7.Visible:=false;
  form7.Label9.Visible:=false;
 end;
end;

procedure TForm7.FormShow(Sender: TObject);
begin
 form7.DateTimePicker1.Date:=date;
 form7.DateTimePicker2.Date:=date;
 iKD:='0';
 FORM7.LbSpeedButton4.Click;
 FORM7.Panel1.SetFocus;
end;

procedure TForm7.FormKeyDown(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
 if key = vk_f2 then
  form7.LbSpeedButton1.Click;//kaydet
 if key = vk_f3 then
  form7.LbSpeedButton2.Click;//düzelt
 if key = vk_f4 then
  form7.LbSpeedButton3.Click;//sil
 if key = vk_f5 then
  form7.LbSpeedButton4.Click;//temizle
 if key = vk_f6 then
  form7.LbSpeedButton5.Click;//ana menü
 if key = vk_escape then
  form7.Close;
end;

procedure TForm7.Hazirla1Click(Sender: TObject);
begin
form7.LbSpeedButton4.Click;
form7.Hide;
form1.LbSpeedButton1.Click;
end;

procedure TForm7.BugnnDetaylar1Click(Sender: TObject);
begin
form7.LbSpeedButton4.Click;
form7.Hide;
form16.LbSpeedButton1.Click;
```

```
end;

procedure TForm7.Parazit1Click(Sender: TObject);
begin
form1.LbSpeedButton5.Click;
end;

procedure TForm7.DParazit1Click(Sender: TObject);
begin
form7.LbSpeedButton4.Click;
form7.Hide;
form1.LbSpeedButton4.Click;
end;

procedure TForm7.A1Click(Sender: TObject);
begin
form6.LbSpeedButton4.Click;
form7.Hide;
form1.LbSpeedButton3.Click;
end;

procedure TForm7.MTERKAYIT1Click(Sender: TObject);
begin
form7.LbSpeedButton4.Click;
form7.Hide;
form3.LbSpeedButton1.Click;
end;


procedure TForm7.MterismineGre1Click(Sender: TObject);
begin
form7.LbSpeedButton4.Click;
form7.Hide;
form13.LbSpeedButton6.Click;
end;

procedure TForm7.HayvanaGreArama1Click(Sender: TObject);
begin
form7.LbSpeedButton4.Click;
form7.Hide;
form13.LbSpeedButton1.Click;
end;

procedure TForm7.k1Click(Sender: TObject);
begin
form1.LbSpeedButton7.Click;
end;
```

```
procedure TForm7.HAYVANKAYIT1Click(Sender: TObject);
begin
form7.LbSpeedButton4.Click;
form7.Hide;
form3.LbSpeedButton2.Click;
end;

procedure TForm7.Gizle1Click(Sender: TObject);
begin
form1.LbSpeedButton8.Click;
end;

procedure TForm7.DBGrid1KeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
 if (form7.DBGrid1.FieldCount <> 0) AND (FORM7.DBGrid1.Fields[0].Text <> '') then
 begin
  //FORM7.CheckBox1.Checked:=FALSE;
  //FORM7.CheckBox1.Enabled:=FALSE;
  //FORM7.LbSpeedButton1.Enabled:=FALSE;
  DATESEPARATOR:='.';
  SHORTDATEFORMAT:='DD/MM/YYYY';
  FORM7.Panel1.Caption:=FORM7.DBGrid1.Fields[0].Text;
  FORM7.DateTimePicker1.Date:=STRTODATE(FORM7.DBGrid1.Fields[1].Text);
  FORM7.Edit2.Text:=FORM7.DBGrid1.Fields[2].Text;
  FORM7.DateTimePicker2.Date:=STRTODATE(FORM7.DBGrid1.Fields[3].Text);
  FORM7.Edit1.Text:=FORM7.DBGrid1.Fields[4].Text;
  iKD:=FORM7.DBGrid1.Fields[5].Text;
 END
end;

procedure TForm7.DateTimePicker1CloseUp(Sender: TObject);
begin
 form7.DateTimePicker1.SetFocus;
end;

procedure TForm7.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
 if key=#13 then
  form7.Edit2.SetFocus;
end;

end.
```

FORM 8 CODES

```
unit Unit8;

interface

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

type
  TForm8 = class(TForm)
    DBGrid1: TDBGrid;
    DBGrid2: TDBGrid;
    Panel1: TPanel;
    SpeedButton1: TSpeedButton;
    DataSource1: TDataSource;
    Panel2: TPanel;
    Panel3: TPanel;
    Panel4: TPanel;
    SpeedButton2: TSpeedButton;
    ADOTable1: TADOTable;
    DataSource2: TDataSource;
    ADOQuery1: TADOQuery;
    Label1: TLabel;
    Panel5: TPanel;
    Label2: TLabel;
    Label3: TLabel;
    procedure SpeedButton2Click(Sender: TObject);
    procedure DBGrid2CellClick(Column: TColumn);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure DBGrid1CellClick(Column: TColumn);
    procedure SpeedButton1Click(Sender: TObject);
    procedure DBGrid1KeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
    procedure FormShow(Sender: TObject);
    procedure DBGrid2KeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure DBGrid2KeyPress(Sender: TObject; var Key: Char);
    procedure FormKeyPress(Sender: TObject; var Key: Char);
    procedure FormHide(Sender: TObject);
    procedure DBGrid2DblClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
```

```
 end;

var
 Form8: TForm8;

implementation

uses Unit2, Unit5, Unit6, Unit7;

{$R *.dfm}

procedure TForm8.SpeedButton2Click(Sender: TObject);
begin
FORM8.Hide;
end;

procedure TForm8.DBGrid2CellClick(Column: TColumn);
begin
  if form8.DBGrid2.FieldCount <> 0 then
    FORM8.Panel1.Caption:=FORM8.DBGrid2.Fields[0].Text;
end;

procedure TForm8.FormClose(Sender: TObject; var Action: TCloseAction);
begin
form8.Panel1.Caption:='';
form8.Label3.Caption:='';
form8.ADOQuery1.Close;
FORM8.Hide;
end;

procedure TForm8.DBGrid1CellClick(Column: TColumn);
begin
  IF FORM8.DBGrid1.FieldCount <> 0 THEN
  BEGIN
    FORM8.ADOQuery1.SQL.Text:='SELECT Hismi,Hturu,Hirki,Hcinsiyeti,Hdogumtarihi,Hid
from hayvan where Sid='+#39+form8.DBGrid1.Fields[2].Text+#39;
    //form8.ADOQuery1.SQL.Text:='select * from hayvan where
Sid='+#39+form8.DBGrid1.Fields[2].Text;
    form8.ADOQuery1.Open;
    if form8.ADOQuery1.RecordCount = 0  then
    begin
      form8.DBGrid2.Visible:=false;
      form8.Label1.Visible:=true;
    end
    else
    begin
      form8.DBGrid2.Visible:=true;
```

```
    form8.Label1.Visible:=false;
   end;
   if (form8.DBGrid1.Fields[0].Text <> '') and (form8.DBGrid1.Fields[1].Text <> '') then
    FORM8.Label3.Caption:=form8.DBGrid1.Fields[0].Text+' '+form8.DBGrid1.Fields[1].Text;
   form8.Panel1.Caption:='';
  END;

end;

procedure TForm8.SpeedButton1Click(Sender: TObject);
begin
 if form8.Panel1.Caption <> '' then
 begin
  if (hyv = 5) then
  begin
   form5.Panel1.Caption:=form8.DBGrid2.Fields[0].Text;
   form5.Edit1.Text:=form8.DBGrid2.Fields[5].Text;
   form8.Hide;
  end
  else if (hyv = 6) then
  begin
   form6.Panel1.Caption:=form8.DBGrid2.Fields[0].Text;
   form6.Edit1.Text:=form8.DBGrid2.Fields[5].Text;
   form8.Hide;
  end
  else if (hyv = 7) then
  begin
   form7.Panel1.Caption:=form8.DBGrid2.Fields[0].Text;
   form7.Edit1.Text:=form8.DBGrid2.Fields[5].Text;
   form8.Hide;
  end;
 end
 else
  application.MessageBox('BEFORE SELECT OWNER'+#13+'THEN SELECT ANIMAL OF
OWNER','ATTENTION',MB_OK+MB_ICONEXCLAMATION);

end;

procedure TForm8.DBGrid1KeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
 IF (FORM8.DBGrid1.FieldCount <> 0) THEN
 BEGIN
 if (key <> vk_return)  then
  begin
   FORM8.ADOQuery1.SQL.Text:='SELECT Hismi,Hturu,Hirki,Hcinsiyeti,Hdogumtarihi,Hid
from hayvan where Sid='+#39+form8.DBGrid1.Fields[2].Text+#39;
```

```
    //form8.ADOQuery1.SQL.Text:='select * from hayvan where
Sid='+#39+form8.DBGrid1.Fields[2].Text;
    form8.ADOQuery1.Open;
    if form8.ADOQuery1.RecordCount = 0  then
    begin
     form8.DBGrid2.Visible:=false;
     form8.Label1.Visible:=true;
    end
    else
    begin
     form8.DBGrid2.Visible:=true;
     form8.Label1.Visible:=false;
    end;
   end;
   if (form8.DBGrid1.Fields[0].Text <> '') and (form8.DBGrid1.Fields[1].Text <> '') then
    FORM8.Label3.Caption:=form8.DBGrid1.Fields[0].Text+' '+form8.DBGrid1.Fields[1].Text;
   form8.Panel1.Caption:='';
  END;
end;

procedure TForm8.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
 if (key = #13) and (form8.DBGrid2.Visible=true) and (form8.DBGrid2.FieldCount <> 0) then
 begin                                  //formun açılışında dbg2 null olduğu için hata veriyordu
  form8.DBGrid2.SetFocus;
  form8.DBGrid2.SelectedIndex:=1;
 end;
end;

procedure TForm8.FormShow(Sender: TObject);
begin
 form8.ADOTable1.Active:=false;
 form8.ADOTable1.Active:=true;
 form8.Panel1.Caption:='';
 form8.DBGrid1.SetFocus;
end;

procedure TForm8.DBGrid2KeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
   FORM8.Panel1.Caption:=FORM8.DBGrid2.Fields[0].Text;
end;

procedure TForm8.DBGrid2KeyPress(Sender: TObject; var Key: Char);
begin
 if (key = #13) and (form8.Panel1.Caption <> '') then
   form8.SpeedButton1.Click;
```

```
end;

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

procedure TForm8.FormHide(Sender: TObject);
begin
form8.Panel1.Caption:='';
form8.Label3.Caption:='';
form8.ADOQuery1.Close;
end;

procedure TForm8.DBGrid2DblClick(Sender: TObject);
begin
form8.SpeedButton1.Click;

end;

end.
```

FORM 9 CODES

```
unit Unit9;

interface

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

type
  TForm9 = class(TForm)
    Panel1: TPanel;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    Panel2: TPanel;
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    ADOTable1: TADOTable;
    procedure SpeedButton2Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure SpeedButton1Click(Sender: TObject);
    procedure DBGrid1CellClick(Column: TColumn);
```

```
  procedure FormShow(Sender: TObject);
  procedure DBGrid1DblClick(Sender: TObject);
  procedure FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
  procedure DBGrid1KeyUp(Sender: TObject; var Key: Word;
    Shift: TShiftState);
  procedure FormHide(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 Form9: TForm9;

implementation

uses Unit2, Unit4;

{$R *.dfm}

procedure TForm9.SpeedButton2Click(Sender: TObject);
begin
FORM9.Hide;
end;

procedure TForm9.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 MN:=0;
 form9.Panel1.Caption:='';
 form9.DBGrid1.SelectedRows.CurrentRowSelected:=false;
 FORM9.Hide;

end;

procedure TForm9.SpeedButton1Click(Sender: TObject);
begin
 if form9.Panel1.Caption <> '' then
 begin
  if mn = 1 then
  begin
   FORM4.Edit1.Text:=form9.DBGrid1.Fields[0].Text; // FORM9.Panel1.Caption;
   FORM4.Edit2.Text:=FORM9.DBGrid1.Fields[1].Text+' '+FORM9.DBGrid1.Fields[2].Text;
   form4.show;
   form9.Hide;
   FORM4.tfXPEdit1.SetFocus;
```

```
  end;
 end
 else
 BEGIN
  application.MessageBox('LÜTFEN LİSTEDEN HAYVAN SAHİBİNİ SEÇİNİZ','KİŞİ
SEÇ',MB_OK+MB_ICONEXCLAMATION);
 END;

end;

procedure TForm9.DBGrid1CellClick(Column: TColumn);
begin
 IF FORM9.DBGrid1.FieldCount <> 0 THEN
  form9.Panel1.Caption:=FORM9.DBGrid1.Fields[1].Text+' '+FORM9.DBGrid1.Fields[2].Text;
//form9.DBGrid1.Fields[0].text;
end;

procedure TForm9.FormShow(Sender: TObject);
begin
FORM9.ADOTable1.Active:=FALSE;
FORM9.ADOTable1.Active:=TRUE;

//form4.Panel1.Caption:=FORM9.DBGrid1.Fields[1].Text+' '+FORM9.DBGrid1.Fields[2].Text;

end;

procedure TForm9.DBGrid1DblClick(Sender: TObject);
begin
{if form9.Panel1.Caption <> '' then
 begin
  if mn = 1 then
  begin
   FORM4.Edit1.Text:=FORM9.Panel1.Caption;
   FORM4.Edit2.Text:=FORM9.DBGrid1.Fields[1].Text+' '+FORM9.DBGrid1.Fields[2].Text;
   form4.show;
   form9.Hide;
   FORM4.tfXPEdit1.SetFocus;
  end;
 end
 else
 BEGIN
  application.MessageBox('LÜTFEN LİSTEDEN HAYVAN SAHİBİNİ SEÇİNİZ','KİŞİ
SEÇ',MB_OK+MB_ICONEXCLAMATION);
  MN:=0;
 END;}
 form9.SpeedButton1.Click;
end;
```

```
procedure TForm9.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
    if key = vk_return then
    form9.SpeedButton1.Click;
  if key = vk_escape then
    form9.SpeedButton2.Click;
end;

procedure TForm9.DBGrid1KeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  IF FORM9.DBGrid1.FieldCount <> 0 THEN
    form9.Panel1.Caption:=FORM9.DBGrid1.Fields[1].Text+' '+FORM9.DBGrid1.Fields[2].Text;
//form9.DBGrid1.Fields[0].text;
end;

procedure TForm9.FormHide(Sender: TObject);
begin
  MN:=0;
  form9.Panel1.Caption:='';
  form9.DBGrid1.SelectedRows.CurrentRowSelected:=false;
end;

end.
```

FORM 10 CODES

```
unit Unit10;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, Buttons, Grids, DBGrids, ComCtrls, StdCtrls, XP_Edit, DB,
  ADODB;

type
  TForm10 = class(TForm)
    GroupBox2: TGroupBox;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label17: TLabel;
    Label18: TLabel;
```

```
Label19: TLabel;
tfXPEdit7: TtfXPEdit;
tfXPEdit8: TtfXPEdit;
tfXPEdit9: TtfXPEdit;
tfXPEdit10: TtfXPEdit;
GroupBox1: TGroupBox;
Label1: TLabel;
Label2: TLabel;
Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
Label6: TLabel;
Label7: TLabel;
Label8: TLabel;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
tfXPEdit1: TtfXPEdit;
tfXPEdit2: TtfXPEdit;
tfXPEdit3: TtfXPEdit;
Memo1: TMemo;
tfXPEdit4: TtfXPEdit;
tfXPEdit5: TtfXPEdit;
tfXPEdit6: TtfXPEdit;
tfXPEdit11: TtfXPEdit;
tfXPEdit12: TtfXPEdit;
tfXPEdit13: TtfXPEdit;
tfXPEdit14: TtfXPEdit;
tfXPEdit15: TtfXPEdit;
tfXPEdit16: TtfXPEdit;
tfXPEdit17: TtfXPEdit;
tfXPEdit18: TtfXPEdit;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
TabSheet2: TTabSheet;
TabSheet3: TTabSheet;
DBGrid1: TDBGrid;
DBGrid2: TDBGrid;
DBGrid3: TDBGrid;
SpeedButton1: TSpeedButton;
SpeedButton2: TSpeedButton;
SpeedButton3: TSpeedButton;
SpeedButton4: TSpeedButton;
MainMenu1: TMainMenu;
A1: TMenuItem;
Gizle1: TMenuItem;
```

```
  ADOQuery1: TADOQuery;
  ADOQuery2: TADOQuery;
  ADOQuery3: TADOQuery;
  ADOQuery4: TADOQuery;
  ADOQuery5: TADOQuery;
  DataSource1: TDataSource;
  DataSource2: TDataSource;
  DataSource3: TDataSource;
  DataSource4: TDataSource;
  DataSource5: TDataSource;
  procedure SpeedButton4Click(Sender: TObject);
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure MTERKAYIT1Click(Sender: TObject);
  procedure HAYVANKAYIT1Click(Sender: TObject);
  procedure A1Click(Sender: TObject);
  procedure DParazit1Click(Sender: TObject);
  procedure Parazit1Click(Sender: TObject);
  procedure MterismineGre1Click(Sender: TObject);
  procedure HayvanaGreArama1Click(Sender: TObject);
  procedure Hazirla1Click(Sender: TObject);
  procedure BugnnDetaylar1Click(Sender: TObject);
  procedure Gizle1Click(Sender: TObject);
  procedure k1Click(Sender: TObject);
  procedure FormShow(Sender: TObject);
  procedure SpeedButton1Click(Sender: TObject);
  procedure SpeedButton2Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 Form10: TForm10;
 tbs:string;
implementation

uses Unit1, Unit16, Unit3, Unit13, Unit11, Unit12, Unit2;

{$R *.dfm}

procedure goster();
begin
 if gnlsnc <> 0 THEN
 BEGIN
  FORM10.ADOQuery2.SQL.Text:='SELECT * FROM hayvan where
Hid='+#39+inttostr(gnlsnc)+#39;
```

```
  form10.ADOQuery2.Open;
  form10.tfXPEdit7.Text:=form10.ADOQuery2['Hid'];
  form10.tfXPEdit8.Text:=form10.ADOQuery2['Hismi'];
  form10.tfXPEdit9.Text:=form10.ADOQuery2['Hturu'];
  form10.tfXPEdit10.Text:=form10.ADOQuery2['Hirki'];
  form10.tfXPEdit16.Text:=form10.ADOQuery2['Hcinsiyeti'];
  form10.tfXPEdit17.Text:=form10.ADOQuery2['Hdogumtarihi'];
  form10.tfXPEdit18.Text:=form10.ADOQuery2['Hislemkodu'];

  tbs:=form10.ADOQuery2['Sid'];
  form10.ADOQuery1.SQL.Text:='select * from musteri where Sid='+#39+tbs+#39;
  form10.ADOQuery1.Open;
  form10.tfXPEdit1.Text:=form10.ADOQuery1['Sid'];
  form10.tfXPEdit2.Text:=form10.ADOQuery1['Misim'];
  form10.tfXPEdit3.Text:=form10.ADOQuery1['Msoyisim'];
  form10.tfXPEdit11.Text:=form10.ADOQuery1['Evtel'];
  form10.tfXPEdit12.Text:=form10.ADOQuery1['Istel'];
  form10.tfXPEdit13.Text:=form10.ADOQuery1['Ceptel'];
  form10.tfXPEdit14.Text:=form10.ADOQuery1['Fax'];
  form10.tfXPEdit15.Text:=form10.ADOQuery1['Extratel'];
  form10.tfXPEdit4.Text:=form10.ADOQuery1['Eposta'];
  form10.tfXPEdit5.Text:=form10.ADOQuery1['Web'];
  form10.memo1.Text:=form10.ADOQuery1['Adres'];
  form10.tfXPEdit6.Text:=form10.ADOQuery1['Mislemkodu'];

  form10.ADOQuery3.SQL.Text:='select * from asilama where
Hid='+#39+inttostr(gnlsnc)+#39+'ORDER BY TekrarAsilamaTarihi DESC';
  form10.ADOQuery3.Open;

  form10.ADOQuery4.SQL.Text:='select * from disparazit where
Hid='+#39+inttostr(gnlsnc)+#39+' order by TekrarDisParUygTarihi DESC';
  form10.ADOQuery4.Open;

  form10.ADOQuery5.SQL.Text:='select * from icparazit where
Hid='+#39+inttostr(gnlsnc)+#39+' order by TekrarIcParUygTarihi DESC';
  form10.ADOQuery5.Open;
 end;
end;
procedure TForm10.SpeedButton4Click(Sender: TObject);
begin
gnlsnc:=0;
form10.tfXPEdit1.Clear;
form10.tfXPEdit2.Clear;
form10.tfXPEdit3.Clear;
form10.tfXPEdit4.Clear;
form10.tfXPEdit5.Clear;
form10.tfXPEdit6.Clear;
```

```
form10.tfXPEdit7.Clear;
form10.tfXPEdit8.Clear;
form10.tfXPEdit9.Clear;
form10.tfXPEdit10.Clear;
form10.tfXPEdit11.Clear;
form10.tfXPEdit12.Clear;
form10.tfXPEdit13.Clear;
form10.tfXPEdit14.Clear;
form10.tfXPEdit15.Clear;
form10.tfXPEdit16.Clear;
form10.tfXPEdit17.Clear;
form10.tfXPEdit18.Clear;
form10.Memo1.Clear;
form11.ADOQuery1.Close;
form11.ADOQuery2.Close;
form11.ADOQuery3.Close;
form11.ADOQuery4.Close;
form11.ADOQuery5.Close;
FORM10.Hide;
end;

procedure TForm10.FormClose(Sender: TObject; var Action: TCloseAction);
begin
FORM10.Hide;
FORM1.SHOW;
end;

procedure TForm10.Hazirla1Click(Sender: TObject);
begin
form10.SpeedButton3.Click;
form10.Hide;
form1.LbSpeedButton1.Click;
end;

procedure TForm10.BugnnDetaylar1Click(Sender: TObject);
begin
form10.SpeedButton3.Click;
form10.Hide;
form16.LbSpeedButton1.Click;
end;

procedure TForm10.Parazit1Click(Sender: TObject);
begin
form10.SpeedButton3.Click;
form10.Hide;
form1.LbSpeedButton5.Click;
end;
```

```
procedure TForm10.DParazit1Click(Sender: TObject);
begin
form10.SpeedButton3.Click;
form10.Hide;
form1.LbSpeedButton4.Click;
end;

procedure TForm10.A1Click(Sender: TObject);
begin
form10.Hide;
end;

procedure TForm10.MTERKAYIT1Click(Sender: TObject);
begin
form10.SpeedButton3.Click;
form10.Hide;
form3.LbSpeedButton1.Click;
end;


procedure TForm10.MterismineGre1Click(Sender: TObject);
begin
form10.SpeedButton3.Click;
form10.Hide;
form13.LbSpeedButton6.Click;
end;

procedure TForm10.HayvanaGreArama1Click(Sender: TObject);
begin
form10.SpeedButton3.Click;
form10.Hide;
form13.LbSpeedButton1.Click;
end;

procedure TForm10.k1Click(Sender: TObject);
begin
FORM12.Close;
form1.LbSpeedButton7.Click;
end;


procedure TForm10.HAYVANKAYIT1Click(Sender: TObject);
begin
form10.SpeedButton3.Click;
form10.Hide;
form3.LbSpeedButton2.Click;
```

```
end;


procedure TForm10.Gizle1Click(Sender: TObject);
begin
form1.LbSpeedButton8.Click;
end;


procedure TForm10.FormShow(Sender: TObject);

begin
  goster();
end;

procedure TForm10.SpeedButton1Click(Sender: TObject);
begin
  if form10.ADOQuery2.Eof=false then
  begin
    gnlsnc:=gnlsnc+1;
    goster();
    FORM10.ADOQuery2.Next;
  end
  else
    Application.MessageBox('THE LAST
RECORD','ATTENTION',MB_OK+MB_ICONHAND);
end;

procedure TForm10.SpeedButton2Click(Sender: TObject);
begin
if form10.ADOQuery2.Bof=false then
  begin
    gnlsnc:=gnlsnc-1;
    goster();
    FORM10.ADOQuery2.Prior;
  end
  else
    Application.MessageBox('THE FIRST
RECORD','ATTENTION',MB_OK+MB_ICONHAND);
end;

end.
```

FORM 11 CODES

```
unit Unit11;
```

```
interface

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

type
  TForm11 = class(TForm)
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    TabSheet3: TTabSheet;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Edit1: TEdit;
    DBGrid1: TDBGrid;
    Panel1: TPanel;
    SpeedButton1: TSpeedButton;
    Edit3: TEdit;
    SpeedButton3: TSpeedButton;
    ADOQuery1: TADOQuery;
    DataSource1: TDataSource;
    LbSpeedButton1: TLbSpeedButton;
    ADOQuery2: TADOQuery;
    ADOQuery3: TADOQuery;
    ADOQuery4: TADOQuery;
    ADOQuery5: TADOQuery;
    DataSource2: TDataSource;
    DataSource3: TDataSource;
    DataSource4: TDataSource;
    DataSource5: TDataSource;
    Panel2: TPanel;
    Panel3: TPanel;
    Label4: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label17: TLabel;
    Label18: TLabel;
    Label19: TLabel;
    Label20: TLabel;
    Label21: TLabel;
    Label22: TLabel;
    Memo1: TMemo;
    Panel4: TPanel;
    Edit4: TEdit;
```

```
Edit5: TEdit;
Edit6: TEdit;
Edit7: TEdit;
Edit8: TEdit;
Edit9: TEdit;
Edit10: TEdit;
Edit11: TEdit;
Label23: TLabel;
Edit12: TEdit;
GroupBox1: TGroupBox;
Label7: TLabel;
Label8: TLabel;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
Label13: TLabel;
Label14: TLabel;
MainMenu1: TMainMenu;
M1: TMenuItem;
MTERKAYIT1: TMenuItem;
HAYVANKAYIT1: TMenuItem;
A1: TMenuItem;
DParazit1: TMenuItem;
Parazit1: TMenuItem;
KaytArama1: TMenuItem;
MterismineGre1: TMenuItem;
HayvanaGreArama1: TMenuItem;
Hazirla1: TMenuItem;
BugnnDetaylar1: TMenuItem;
Gizle1: TMenuItem;
k1: TMenuItem;
SpeedButton2: TSpeedButton;
Edit2: TEdit;
Panel5: TPanel;
Panel6: TPanel;
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure Edit1Change(Sender: TObject);
procedure LbSpeedButton1Click(Sender: TObject);
procedure DBGrid1CellClick(Column: TColumn);
procedure DBGrid1KeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure SpeedButton1Click(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure MTERKAYIT1Click(Sender: TObject);
procedure HAYVANKAYIT1Click(Sender: TObject);
procedure A1Click(Sender: TObject);
```

```
    procedure DParazit1Click(Sender: TObject);
    procedure Parazit1Click(Sender: TObject);
    procedure MterismineGre1Click(Sender: TObject);
    procedure HayvanaGreArama1Click(Sender: TObject);
    procedure Hazirla1Click(Sender: TObject);
    procedure BugnnDetaylar1Click(Sender: TObject);
    procedure Gizle1Click(Sender: TObject);
    procedure k1Click(Sender: TObject);
    procedure Edit2Change(Sender: TObject);
    procedure Edit3Change(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure SpeedButton3Click(Sender: TObject);
    procedure PageControl1Change(Sender: TObject);
    procedure FormHide(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form11: TForm11;
  i:integer=0;
implementation

uses Unit1, Unit2, Unit16, Unit3, Unit13;

{$R *.dfm}

PROCEDURE temizle11();
begin
  form11.Panel4.Caption:='ANIMAL OWNER NO:';
  form11.Edit4.Clear;
  form11.Edit5.Clear;
  form11.Edit6.Clear;
  form11.Edit7.Clear;
  form11.Edit8.Clear;
  form11.Edit9.Clear;
  form11.Edit10.Clear;
  form11.Edit11.Clear;
  form11.Edit12.Clear;
  form11.Memo1.Clear;
end;

PROCEDURE SONUCANALIZLERI();
BEGIN
//TOPLAM MÜŞTERİ SAYISI
```

```
FORM11.ADOQuery2.SQL.Text:='SELECT COUNT(Sid) FROM musteri';
form11.ADOQuery2.Open;
FORM11.Label13.Caption:='  '+INTTOSTR(FORM11.ADOQuery2['Count(Sid)'])+'  PIECE  ';

//TOPLAM HAYVAN SAYISI
FORM11.ADOQuery3.SQL.Text:='SELECT COUNT(Hid) FROM hayvan';
form11.ADOQuery3.Open;
FORM11.Label14.Caption:='  '+INTTOSTR(FORM11.ADOQuery3['Count(Hid)'])+'  PIECE  ';

//HAYVAN SAHİBİ MÜŞTERİLERİN SAHİBİ
FORM11.ADOQuery4.SQL.Text:='SELECT H.Sid,M.Sid from Hayvan H,Musteri M where
H.Sid=M.Sid';
form11.ADOQuery4.Open;
FORM11.Label11.Caption:='  '+INTTOSTR(FORM11.ADOQuery4.RecordCount)+'  PIECE  ';

form11.ADOQuery5.SQL.Text:='select Hismi,Misim from musteri left outer join (hayvan) on
musteri.Sid=hayvan.Sid';
form11.ADOQuery5.Open;
form11.ADOQuery5.First;
while not form11.ADOQuery5.Eof do
begin
 if form11.ADOQuery5['Hismi'] = null then
  i:=i+1;
 form11.ADOQuery5.Next;
end;
form11.Label12.Caption:='  '+inttostr(i)+' PIECE  ';
i:=0;


END;

procedure TForm11.FormClose(Sender: TObject; var Action: TCloseAction);
begin
FORM11.Hide;
FORM1.SHOW;
end;

procedure TForm11.Edit1Change(Sender: TObject);
begin
 IF FORM11.Edit1.Text <> '' THEN
 BEGIN
  FORM11.ADOQuery1.SQL.Text:='SELECT * FROM musteri where Misim
LIKE'+#39+'%'+form11.Edit1.Text+'%'+#39;
  form11.ADOQuery1.Open;
  FORM11.PANEL2.Caption:=INTTOSTR(FORM11.ADOQuery1.RecordCount)+' RECORD
FOUND';
 END;
```

```
SONUCANALIZLERI();

end;

procedure TForm11.LbSpeedButton1Click(Sender: TObject);
begin
FORM11.Edit1.Clear;
FORM11.Edit2.Clear;
FORM11.Edit3.Clear;
FORM11.Panel2.Caption:='';
FORM11.Panel5.Caption:='';
FORM11.Panel6.Caption:='';
FORM11.Hide;
FORM1.SHOW;
end;

procedure TForm11.DBGrid1CellClick(Column: TColumn);
begin
 IF (FORM11.DBGrid1.FieldCount <> 0) THEN
 BEGIN
  form11.Panel4.Caption:='ANIMAL OWNER NO:  '+form11.DBGrid1.Fields[0].Text;
  form11.EDIT4.Text:=form11.DBGrid1.Fields[1].Text;
  form11.EDIT5.Text:=form11.DBGrid1.Fields[2].Text;
  form11.EDIT6.Text:=form11.DBGrid1.Fields[4].Text;
  form11.EDIT7.Text:=form11.DBGrid1.Fields[5].Text;
  form11.EDIT8.Text:=form11.DBGrid1.Fields[6].Text;
  form11.EDIT9.Text:=form11.DBGrid1.Fields[7].Text;
  form11.EDIT10.Text:=form11.DBGrid1.Fields[8].Text;
  form11.EDIT11.Text:=form11.DBGrid1.Fields[9].Text;
  form11.EDIT12.Text:=form11.DBGrid1.Fields[10].Text;
  form11.Memo1.Text:=form11.DBGrid1.Fields[3].Text;
 END
end;

procedure TForm11.DBGrid1KeyUp(Sender: TObject; var Key: Word;
 Shift: TShiftState);
begin
 IF FORM11.DBGrid1.FieldCount <> 0 THEN
 BEGIN
  form11.Panel4.Caption:='ANIMAL OWNER NO:  '+form11.DBGrid1.Fields[0].Text;
  form11.EDIT4.Text:=form11.DBGrid1.Fields[1].Text;
  form11.EDIT5.Text:=form11.DBGrid1.Fields[2].Text;
  form11.EDIT6.Text:=form11.DBGrid1.Fields[4].Text;
  form11.EDIT7.Text:=form11.DBGrid1.Fields[5].Text;
  form11.EDIT8.Text:=form11.DBGrid1.Fields[6].Text;
  form11.EDIT9.Text:=form11.DBGrid1.Fields[7].Text;
  form11.EDIT10.Text:=form11.DBGrid1.Fields[8].Text;
```

```
   form11.EDIT11.Text:=form11.DBGrid1.Fields[9].Text;
   form11.EDIT12.Text:=form11.DBGrid1.Fields[10].Text;
   form11.Memo1.Text:=form11.DBGrid1.Fields[3].Text;
  END
end;


procedure TForm11.SpeedButton1Click(Sender: TObject);
begin
  FORM11.Edit1.Clear;
  PANEL2.Caption:='';
  temizle11();
  SONUCANALIZLERI();
  FORM11.ADOQuery1.SQL.Text:='SELECT * FROM musteri';
  form11.ADOQuery1.Open;
end;

procedure TForm11.FormShow(Sender: TObject);
begin
  FORM11.ADOQuery1.SQL.Text:='SELECT * FROM musteri';
  form11.ADOQuery1.Open;
  SONUCANALIZLERI();
end;

procedure TForm11.Hazirla1Click(Sender: TObject);
begin
Form11.LbSpeedButton1.Click;
form11.Hide;
form1.LbSpeedButton1.Click;
end;

procedure TForm11.BugnnDetaylar1Click(Sender: TObject);
begin
Form11.LbSpeedButton1.Click;
form11.Hide;
form16.LbSpeedButton1.Click;
end;

procedure TForm11.Parazit1Click(Sender: TObject);
begin
Form11.LbSpeedButton1.Click;
form11.Hide;
form1.LbSpeedButton5.Click;
end;

procedure TForm11.DParazit1Click(Sender: TObject);
begin
```

```
Form11.LbSpeedButton1.Click;
form11.Hide;
form1.LbSpeedButton4.Click;
end;

procedure TForm11.A1Click(Sender: TObject);
begin
Form11.LbSpeedButton1.Click;
form11.Hide;
form1.LbSpeedButton3.Click;
end;

procedure TForm11.MTERKAYIT1Click(Sender: TObject);
begin
Form11.LbSpeedButton1.Click;
form11.Hide;
form3.LbSpeedButton1.Click;
end;


procedure TForm11.MterismineGre1Click(Sender: TObject);
begin
form13.LbSpeedButton6.Click;
end;

procedure TForm11.HayvanaGreArama1Click(Sender: TObject);
begin
Form11.LbSpeedButton1.Click;
form11.Hide;
form13.LbSpeedButton1.Click;
end;

procedure TForm11.k1Click(Sender: TObject);
begin
form1.LbSpeedButton7.Click;
end;


procedure TForm11.HAYVANKAYIT1Click(Sender: TObject);
begin
Form11.LbSpeedButton1.Click;
form11.Hide;
form3.LbSpeedButton2.Click;
end;


procedure TForm11.Gizle1Click(Sender: TObject);
```

```
begin
form1.LbSpeedButton8.Click;
end;


procedure TForm11.Edit2Change(Sender: TObject);
begin
 IF FORM11.Edit2.Text <>'' THEN
 BEGIN
  FORM11.ADOQuery1.SQL.Text:='SELECT * FROM musteri where Msoyisim
LIKE'+#39+'%'+form11.Edit2.Text+'%'+#39;
  form11.ADOQuery1.Open;
  FORM11.PANEL5.Caption:=INTTOSTR(FORM11.ADOQuery1.RecordCount)+' RECORD
FOUND';
 END;
 SONUCANALIZLERI();
end;

procedure TForm11.Edit3Change(Sender: TObject);
begin
 IF FORM11.Edit3.Text <>'' THEN
 BEGIN
  FORM11.ADOQuery1.SQL.Text:='SELECT * FROM musteri where Adres
LIKE'+#39+'%'+form11.Edit3.Text+'%'+#39;
  form11.ADOQuery1.Open;
  FORM11.PANEL6.Caption:=INTTOSTR(FORM11.ADOQuery1.RecordCount)+' RECORD
FOUND';
 END;
 SONUCANALIZLERI();
end;

procedure TForm11.SpeedButton2Click(Sender: TObject);
begin
 FORM11.Edit2.Clear;
 PANEL5.Caption:='';
 temizle11();
 SONUCANALIZLERI();
 FORM11.ADOQuery1.SQL.Text:='SELECT * FROM musteri';
 form11.ADOQuery1.Open;
end;

procedure TForm11.SpeedButton3Click(Sender: TObject);
begin
 FORM11.Edit3.Clear;
 PANEL6.Caption:='';
 temizle11();
 SONUCANALIZLERI();
```

```
  FORM11.ADOQuery1.SQL.Text:='SELECT * FROM musteri';
  form11.ADOQuery1.Open;
end;

procedure TForm11.PageControl1Change(Sender: TObject);
begin
  FORM11.Edit1.Clear;
  FORM11.Edit2.Clear;
  FORM11.Edit3.Clear;
  form11.Panel2.Caption:='';
  form11.Panel5.Caption:='';
  form11.Panel6.Caption:='';
  temizle11();
  SONUCANALIZLERI();
  FORM11.ADOQuery1.SQL.Text:='SELECT * FROM musteri';
  form11.ADOQuery1.Open;
end;

procedure TForm11.FormHide(Sender: TObject);
begin
  FORM11.Edit1.Clear;
  FORM11.Edit2.Clear;
  FORM11.Edit3.Clear;
  form11.Panel2.Caption:='';
  form11.Panel5.Caption:='';
  form11.Panel6.Caption:='';
  temizle11();
  SONUCANALIZLERI();
  FORM11.ADOQuery1.SQL.Text:='SELECT * FROM musteri';
  form11.ADOQuery1.Open;
end;

end.
```

FORM 12 CODES

```
unit Unit12;

interface

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

type
  TForm12 = class(TForm)
```

```
PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label1: TLabel;
SpeedButton1: TSpeedButton;
Edit1: TEdit;
TabSheet2: TTabSheet;
Label2: TLabel;
SpeedButton2: TSpeedButton;
Edit2: TEdit;
TabSheet3: TTabSheet;
Label3: TLabel;
SpeedButton3: TSpeedButton;
DBGrid1: TDBGrid;
Panel1: TPanel;
ComboBox1: TComboBox;
TabSheet4: TTabSheet;
DateTimePicker1: TDateTimePicker;
Label7: TLabel;
RadioButton1: TRadioButton;
RadioButton2: TRadioButton;
RadioButton3: TRadioButton;
DateTimePicker2: TDateTimePicker;
RadioButton4: TRadioButton;
GroupBox1: TGroupBox;
RadioButton5: TRadioButton;
RadioButton6: TRadioButton;
SpeedButton4: TSpeedButton;
SpeedButton5: TSpeedButton;
Panel2: TPanel;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
Memo1: TMemo;
Label15: TLabel;
Label16: TLabel;
Label17: TLabel;
Panel3: TPanel;
Edit3: TEdit;
Edit4: TEdit;
Edit5: TEdit;
Edit6: TEdit;
Edit7: TEdit;
Edit8: TEdit;
Label13: TLabel;
Label14: TLabel;
Edit9: TEdit;
```

```
Edit10: TEdit;
MainMenu1: TMainMenu;
M1: TMenuItem;
MTERKAYIT1: TMenuItem;
HAYVANKAYIT1: TMenuItem;
A1: TMenuItem;
DParazit1: TMenuItem;
Parazit1: TMenuItem;
KaytArama1: TMenuItem;
MterismineGre1: TMenuItem;
HayvanaGreArama1: TMenuItem;
Hazirla1: TMenuItem;
BugnnDetaylar1: TMenuItem;
Gizle1: TMenuItem;
k1: TMenuItem;
GroupBox2: TGroupBox;
Label18: TLabel;
Label19: TLabel;
Label20: TLabel;
Label21: TLabel;
Label22: TLabel;
Label23: TLabel;
Label24: TLabel;
Label25: TLabel;
ADOQuery1: TADOQuery;
ADOQuery2: TADOQuery;
ADOQuery3: TADOQuery;
ADOQuery4: TADOQuery;
ADOQuery5: TADOQuery;
DataSource1: TDataSource;
DataSource2: TDataSource;
DataSource3: TDataSource;
DataSource4: TDataSource;
DataSource5: TDataSource;
Label26: TLabel;
Edit11: TEdit;
LbSpeedButton1: TLbSpeedButton;
Panel4: TPanel;
Panel5: TPanel;
Panel6: TPanel;
Panel7: TPanel;
ADOQuery6: TADOQuery;
DataSource6: TDataSource;
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure DBGrid1CellClick(Column: TColumn);
procedure MTERKAYIT1Click(Sender: TObject);
procedure HAYVANKAYIT1Click(Sender: TObject);
```

```
    procedure A1Click(Sender: TObject);
    procedure DParazit1Click(Sender: TObject);
    procedure Parazit1Click(Sender: TObject);
    procedure MterismineGre1Click(Sender: TObject);
    procedure HayvanaGreArama1Click(Sender: TObject);
    procedure Hazirla1Click(Sender: TObject);
    procedure BugnnDetaylar1Click(Sender: TObject);
    procedure Gizle1Click(Sender: TObject);
    procedure k1Click(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure SpeedButton3Click(Sender: TObject);
    procedure SpeedButton5Click(Sender: TObject);
    procedure LbSpeedButton1Click(Sender: TObject);
    procedure Edit1Change(Sender: TObject);
    procedure Edit2Change(Sender: TObject);
    procedure ComboBox1Change(Sender: TObject);
    procedure DBGrid1KeyUp(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure RadioButton4Click(Sender: TObject);
    procedure RadioButton2Click(Sender: TObject);
    procedure RadioButton3Click(Sender: TObject);
    procedure RadioButton1Click(Sender: TObject);
    procedure SpeedButton4Click(Sender: TObject);
    procedure PageControl1Change(Sender: TObject);
    procedure FormHide(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form12: TForm12;
  a:integer=0;
  gnlsnc:integer;

implementation

uses Unit1, Unit10, Unit16, Unit3, Unit13, Unit2, Unit11;

{$R *.dfm}
procedure temizle12();
begin
  form12.Edit3.Clear;
```

```
 form12.Edit4.Clear;
 form12.Edit5.Clear;
 form12.Edit6.Clear;
 form12.Edit7.Clear;
 form12.Edit8.Clear;
 form12.Edit9.Clear;
 form12.Edit10.Clear;
 form12.Edit11.Clear;
 form12.Memo1.Clear;
end;

PROCEDURE SONUCANALIZLERI12();
BEGIN
//TOPLAM MÜŞTERİ SAYISI
FORM12.ADOQuery2.SQL.Text:='SELECT COUNT(Sid) FROM musteri';
form12.ADOQuery2.Open;
FORM12.Label24.Caption:='  '+INTTOSTR(FORM12.ADOQuery2['Count(Sid)'])+' PIECE  ';

//TOPLAM HAYVAN SAYISI
FORM12.ADOQuery3.SQL.Text:='SELECT COUNT(Hid) FROM hayvan';
form12.ADOQuery3.Open;
FORM12.Label25.Caption:='  '+INTTOSTR(FORM12.ADOQuery3['Count(Hid)'])+'  PIECE  ';

//HAYVAN SAHİBİ MÜŞTERİLERİN SAHİBİ
FORM12.ADOQuery4.SQL.Text:='SELECT H.Sid,M.Sid from Hayvan H,Musteri M where
H.Sid=M.Sid';
form12.ADOQuery4.Open;
FORM12.Label22.Caption:='  '+INTTOSTR(FORM12.ADOQuery4.RecordCount)+'  PIECE  ';

//HAYVAN SAHİBİ olmayan MÜŞTERİLERİN SAHİBİ
form12.ADOQuery5.SQL.Text:='select Hismi,Misim from musteri left outer join (hayvan) on
musteri.Sid=hayvan.Sid';
form12.ADOQuery5.Open;
form12.ADOQuery5.First;
while not form12.ADOQuery5.Eof do
begin
  if form12.ADOQuery5['Hismi'] = null then
    a:=a+1;
  form12.ADOQuery5.Next;
end;
form12.Label23.Caption:='  '+inttostr(a)+'  PIECE  ';
a:=0;


END;
```

```
procedure TForm12.FormClose(Sender: TObject; var Action: TCloseAction);
begin
FORM12.Hide;
FORM1.SHOW;
end;

procedure TForm12.DBGrid1CellClick(Column: TColumn);
begin
 IF FORM12.DBGrid1.FieldCount <> 0 THEN
 BEGIN
  if (form12.DBGrid1.Fields[6].Text <> null) and (form12.DBGrid1.Fields[6].Text <> '') then
  begin
   form12.ADOQuery6.SQL.Text:='select
Misim,Msoyisim,Adres,Evtel,Istel,Ceptel,Fax,Extratel,Eposta,Web from musteri where
Sid='+#39+form12.DBGrid1.Fields[6].Text+#39;
   form12.ADOQuery6.Open;
   form12.EDIT3.Text:=form12.ADOQuery6['Misim'];
   form12.EDIT4.Text:=form12.ADOQuery6['Msoyisim'];
   form12.EDIT5.Text:=form12.ADOQuery6['Evtel'];
   form12.EDIT6.Text:=form12.ADOQuery6['Istel'];
   form12.EDIT7.Text:=form12.ADOQuery6['Ceptel'];
   form12.EDIT8.Text:=form12.ADOQuery6['Fax'];
   form12.EDIT9.Text:=form12.ADOQuery6['Extratel'];
   form12.EDIT10.Text:=form12.ADOQuery6['Eposta'];
   form12.EDIT11.Text:=form12.ADOQuery6['Web'];
   form12.Memo1.Text:=form12.ADOQuery6['Adres'];
  end;
 END
end;

procedure TForm12.Hazirla1Click(Sender: TObject);
begin
form12.LbSpeedButton1.Click;
form12.Hide;
form1.LbSpeedButton1.Click;
end;

procedure TForm12.BugnnDetaylar1Click(Sender: TObject);
begin
form12.LbSpeedButton1.Click;
form12.Hide;
form16.LbSpeedButton1.Click;
end;

procedure TForm12.Parazit1Click(Sender: TObject);
begin
form12.LbSpeedButton1.Click;
```

```
form12.Hide;
form1.LbSpeedButton5.Click;
end;

procedure TForm12.DParazit1Click(Sender: TObject);
begin
form12.LbSpeedButton1.Click;
form12.Hide;
form1.LbSpeedButton4.Click;
end;

procedure TForm12.A1Click(Sender: TObject);
begin
form12.LbSpeedButton1.Click;
form12.Hide;
form1.LbSpeedButton3.Click;
end;

procedure TForm12.MTERKAYIT1Click(Sender: TObject);
begin
form12.LbSpeedButton1.Click;
form12.Hide;
form3.LbSpeedButton1.Click;
end;


procedure TForm12.MterismineGre1Click(Sender: TObject);
begin
form12.LbSpeedButton1.Click;
form12.Hide;
form13.LbSpeedButton6.Click;
end;

procedure TForm12.HayvanaGreArama1Click(Sender: TObject);
begin
form13.LbSpeedButton1.Click;
end;

procedure TForm12.k1Click(Sender: TObject);
begin
form1.LbSpeedButton7.Click;
end;


procedure TForm12.HAYVANKAYIT1Click(Sender: TObject);
begin
form12.LbSpeedButton1.Click;
```

```
form12.Hide;
form3.LbSpeedButton2.Click;
end;


procedure TForm12.Gizle1Click(Sender: TObject);
begin
form1.LbSpeedButton8.Click;
end;

procedure TForm12.SpeedButton1Click(Sender: TObject);
begin
  form12.Edit1.Clear;
  form12.Panel4.Caption:='';
  temizle12();
  SONUCANALIZLERI12();
  form12.ADOQuery1.SQL.Text:='select * from hayvan';
  form12.ADOQuery1.Open;
end;

procedure TForm12.FormShow(Sender: TObject);
begin
  SONUCANALIZLERI12();
  FORM12.DateTimePicker1.Date:=DATE;
  FORM12.DateTimePicker2.Date:=DATE;
  form12.ADOQuery1.SQL.Text:='select * from hayvan';
  form12.ADOQuery1.Open;
end;

procedure TForm12.SpeedButton2Click(Sender: TObject);
begin
  temizle12();
  SONUCANALIZLERI12();
  panel5.Caption:='';
  form12.ADOQuery1.SQL.Text:='select * from hayvan';
  form12.ADOQuery1.Open;
end;

procedure TForm12.SpeedButton3Click(Sender: TObject);
begin
  form12.ComboBox1.text:='SELECT...';
  form12.Panel6.Caption:='';
  temizle12();
  form12.ADOQuery1.SQL.Text:='select * from hayvan';
  form12.ADOQuery1.Open;
  SONUCANALIZLERI12();
end;
```

```
procedure TForm12.SpeedButton5Click(Sender: TObject);
begin
 form12.RadioButton1.Checked:=false;
 form12.RadioButton2.Checked:=false;
 form12.RadioButton3.Checked:=false;
 form12.RadioButton4.Checked:=false;
 form12.RadioButton5.Checked:=false;
 form12.RadioButton6.Checked:=false;
 form12.Panel7.Caption:='';
 form12.DateTimePicker2.Enabled:=false;
 form12.DateTimePicker1.Date:=date;
 form12.DateTimePicker2.Date:=date;
 temizle12();
 form12.ADOQuery1.SQL.Text:='select * from hayvan';
 form12.ADOQuery1.Open;
 SONUCANALIZLERI12();
end;

procedure TForm12.LbSpeedButton1Click(Sender: TObject);
begin
FORM12.Hide;
FORM1.SHOW;
end;

procedure TForm12.Edit1Change(Sender: TObject);
begin
 IF FORM12.Edit1.Text <> '' THEN
 BEGIN
   FORM12.ADOQuery1.SQL.Text:='SELECT * FROM HAYVAN where Hismi
LIKE'+#39+'%'+form12.Edit1.Text+'%'+#39;
   form12.ADOQuery1.Open;
   FORM12.PANEL4.Caption:=INTTOSTR(FORM12.ADOQuery1.RecordCount)+' RECORD
FOUND';
 END;
 SONUCANALIZLERI12();
end;

procedure TForm12.Edit2Change(Sender: TObject);
begin
 IF FORM12.Edit2.Text <> '' THEN
 BEGIN
   FORM12.ADOQuery1.SQL.Text:='SELECT * FROM HAYVAN where Hirki
LIKE'+#39+'%'+form12.Edit2.Text+'%'+#39;
   form12.ADOQuery1.Open;
   FORM12.PANEL5.Caption:=INTTOSTR(FORM12.ADOQuery1.RecordCount)+' RECORD
FOUND';
```

```
  END;
  SONUCANALIZLERI12();
end;

procedure TForm12.ComboBox1Change(Sender: TObject);
begin
  IF form12.ComboBox1.Text<>'SELECT...' THEN
  BEGIN
    FORM12.ADOQuery1.SQL.Text:='SELECT * FROM HAYVAN where
Hcinsiyeti='+#39+form12.ComboBox1.Text+#39;
    form12.ADOQuery1.Open;
    FORM12.PANEL6.Caption:=INTTOSTR(FORM12.ADOQuery1.RecordCount)+' RECORD
FOUND';
    END;
  SONUCANALIZLERI12();
end;

procedure TForm12.DBGrid1KeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
   IF FORM12.DBGrid1.FieldCount <> 0 THEN
  BEGIN
    if (form12.DBGrid1.Fields[6].Text <> null) and (form12.DBGrid1.Fields[6].Text <> '') then
    begin
      form12.ADOQuery6.SQL.Text:='select
Misim,Msoyisim,Adres,Evtel,Istel,Ceptel,Fax,Extratel,Eposta,Web from musteri where
Sid='+#39+form12.DBGrid1.Fields[6].Text+#39;
      form12.ADOQuery6.Open;
      form12.EDIT3.Text:=form12.ADOQuery6['Misim'];
      form12.EDIT4.Text:=form12.ADOQuery6['Msoyisim'];
      form12.EDIT5.Text:=form12.ADOQuery6['Evtel'];
      form12.EDIT6.Text:=form12.ADOQuery6['Istel'];
      form12.EDIT7.Text:=form12.ADOQuery6['Ceptel'];
      form12.EDIT8.Text:=form12.ADOQuery6['Fax'];
      form12.EDIT9.Text:=form12.ADOQuery6['Extratel'];
      form12.EDIT10.Text:=form12.ADOQuery6['Eposta'];
      form12.EDIT11.Text:=form12.ADOQuery6['Web'];
      form12.Memo1.Text:=form12.ADOQuery6['Adres'];
    end;
  END
end;

procedure TForm12.RadioButton4Click(Sender: TObject);
begin
form12.DateTimePicker2.Enabled:=true;
end;
```

```
procedure TForm12.RadioButton2Click(Sender: TObject);
begin
form12.DateTimePicker2.Enabled:=false;
end;

procedure TForm12.RadioButton3Click(Sender: TObject);
begin
form12.DateTimePicker2.Enabled:=false;
end;

procedure TForm12.RadioButton1Click(Sender: TObject);
begin
form12.DateTimePicker2.Enabled:=false;
end;

procedure TForm12.SpeedButton4Click(Sender: TObject);
var
chk,cns:string;
begin
  dateseparator:='-';
  shortdateformat:='yyyy/mm/dd';
  if (form12.RadioButton1.Checked=true) or (form12.RadioButton2.Checked=true) or
(form12.RadioButton3.Checked=true) or (form12.RadioButton4.Checked=true) then
  begin
    if form12.RadioButton1.Checked=true then
      chk:='< '+#39+datetostr(form12.DateTimePicker1.Date)+#39
    else if form12.RadioButton2.Checked=true then
      chk:='> '+#39+datetostr(form12.DateTimePicker1.Date)+#39
    else if form12.RadioButton3.Checked=true then
      chk:='= '+#39+datetostr(form12.DateTimePicker1.Date)+#39
    else if form12.RadioButton4.Checked=true then
      chk:=' between '+#39+datetostr(form12.DateTimePicker1.Date)+#39+' and
'+#39+datetostr(form12.DateTimePicker2.Date)+#39;

    if (form12.RadioButton5.Checked=false) and (form12.RadioButton6.Checked=false) then
    begin
      cns:='';
    end

    else if (form12.RadioButton5.Checked=true) then
    begin
      cns:=' and Hcinsiyeti='+#39+'MALE'+#39;
    end

    else if (form12.RadioButton6.Checked=true) then
    begin
      cns:=' and Hcinsiyeti='+#39+'FEMALE'+#39;
```

```
  end;

  //showmessage(chk+#13+cns);
  form12.ADOQuery1.SQL.Text:='select * from hayvan where Hdogumtarihi'+chk+cns;
  form12.ADOQuery1.Open;
  FORM12.PANEL7.Caption:=INTTOSTR(FORM12.ADOQuery1.RecordCount)+' RECORD
FOUND';
  cns:='';
  chk:='';

 end
 else
  application.MessageBox('PLEASE SELECT ANY DATE CRITERIA','SELECT
CRITERIA',MB_OK+MB_ICONEXCLAMATION);
 dateseparator:='.';
 shortdateformat:='dd/mm/yyyy';
end;

procedure TForm12.PageControl1Change(Sender: TObject);
begin
 form12.Edit1.Clear;
 form12.Edit2.Clear;
 form12.ComboBox1.Text:='SELECT...';
 form12.Panel4.Caption:='';
 form12.Panel5.Caption:='';
 form12.Panel6.Caption:='';
 form12.SpeedButton5.Click;
end;

procedure TForm12.FormHide(Sender: TObject);
begin
 form12.Edit1.Clear;
 form12.Edit2.Clear;
 form12.ComboBox1.Text:='SELECT...';
 form12.Panel4.Caption:='';
 form12.Panel5.Caption:='';
 form12.Panel6.Caption:='';
 form12.SpeedButton5.Click;
end;

procedure TForm12.DBGrid1DblClick(Sender: TObject);
begin
 IF (FORM12.DBGrid1.FieldCount <> 0) and (form12.DBGrid1.Fields[0].Text <> null) and
(form12.DBGrid1.Fields[0].Text <> '') THEN
  BEGIN
   gnlsnc:=strtoint(form12.DBGrid1.Fields[0].text);
   form10.Show;
```

```
  end;
end;

end.
```

FORM 13 CODES

```
unit Unit13;

interface

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

type
  TForm13 = class(TForm)
    LbSpeedButton6: TLbSpeedButton;
    LbSpeedButton1: TLbSpeedButton;
    procedure LbSpeedButton1Click(Sender: TObject);
    procedure LbSpeedButton6Click(Sender: TObject);
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form13: TForm13;

implementation

uses Unit12, Unit1, Unit11;

{$R *.dfm}

procedure TForm13.LbSpeedButton1Click(Sender: TObject);
begin
FORM12.SHOW;
FORM13.Close;
FORM1.HIDE;
end;

procedure TForm13.LbSpeedButton6Click(Sender: TObject);
```

```
begin
FORM11.SHOW;
FORM13.Close;
FORM1.HIDE;
end;

procedure TForm13.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if key=vk_escape then
    form13.Close;

end;

end.
```

FORM 14 CODES

```
unit Unit14;

interface

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

type
  TForm14 = class(TForm)
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    TabSheet3: TTabSheet;
    DBGrid1: TDBGrid;
    Label1: TLabel;
    DBGrid2: TDBGrid;
    DBGrid3: TDBGrid;
    SpeedButton1: TSpeedButton;
    Panel1: TPanel;
    DateTimePicker1: TDateTimePicker;
    RadioButton1: TRadioButton;
    RadioButton2: TRadioButton;
    RadioButton3: TRadioButton;
    RadioButton4: TRadioButton;
    DateTimePicker2: TDateTimePicker;
    SpeedButton2: TSpeedButton;
```

```
  SpeedButton3: TSpeedButton;
  MainMenu1: TMainMenu;
  M1: TMenuItem;
  MTERKAYIT1: TMenuItem;
  HAYVANKAYIT1: TMenuItem;
  A1: TMenuItem;
  DParazit1: TMenuItem;
  Parazit1: TMenuItem;
  KaytArama1: TMenuItem;
  MterismineGre1: TMenuItem;
  HayvanaGreArama1: TMenuItem;
  Hazirla1: TMenuItem;
  BugnnDetaylar1: TMenuItem;
  Gizle1: TMenuItem;
  k1: TMenuItem;
  ADOQuery1: TADOQuery;
  ADOQuery2: TADOQuery;
  ADOQuery3: TADOQuery;
  DataSource1: TDataSource;
  DataSource2: TDataSource;
  DataSource3: TDataSource;
  procedure FormShow(Sender: TObject);
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure SpeedButton1Click(Sender: TObject);
  procedure MTERKAYIT1Click(Sender: TObject);
  procedure HAYVANKAYIT1Click(Sender: TObject);
  procedure A1Click(Sender: TObject);
  procedure DParazit1Click(Sender: TObject);
  procedure Parazit1Click(Sender: TObject);
  procedure MterismineGre1Click(Sender: TObject);
  procedure HayvanaGreArama1Click(Sender: TObject);
  procedure Hazirla1Click(Sender: TObject);
  procedure BugnnDetaylar1Click(Sender: TObject);
  procedure Gizle1Click(Sender: TObject);
  procedure k1Click(Sender: TObject);
  procedure RadioButton4Click(Sender: TObject);
  procedure RadioButton2Click(Sender: TObject);
  procedure RadioButton3Click(Sender: TObject);
  procedure RadioButton1Click(Sender: TObject);
  procedure SpeedButton2Click(Sender: TObject);
  procedure SpeedButton3Click(Sender: TObject);
  procedure DBGrid1DblClick(Sender: TObject);
  procedure DBGrid2DblClick(Sender: TObject);
  procedure DBGrid3DblClick(Sender: TObject);

private
  { Private declarations }
```

```
  public
    { Public declarations }
  end;

var
  Form14: TForm14;
  TBM:INTEGER;

implementation

uses Unit1, Unit16, Unit3, Unit13, Unit2, Unit10, Unit12;

{$R *.dfm}



procedure TForm14.FormShow(Sender: TObject);
begin
FORM14.Label1.Caption:=FORMATDATETIME('dddddd',NOW)+#13+'JOB LIST';
form14.DateTimePicker1.Date:=date;
form14.DateTimePicker2.Date:=date;
end;

procedure TForm14.FormClose(Sender: TObject; var Action: TCloseAction);
begin
Form14.SpeedButton3.Click;
FORM14.Hide;
FORM1.SHOW;
end;



procedure TForm14.SpeedButton1Click(Sender: TObject);
begin
FORM14.Hide;
FORM1.Show;
end;

procedure TForm14.Hazirla1Click(Sender: TObject);
begin
form1.LbSpeedButton1.Click;
end;

procedure TForm14.BugnnDetaylar1Click(Sender: TObject);
begin
form16.LbSpeedButton1.Click;
end;
```

```
procedure TForm14.Parazit1Click(Sender: TObject);
begin
form14.SpeedButton1.Click;
form14.Hide;
form1.LbSpeedButton5.Click;
end;

procedure TForm14.DParazit1Click(Sender: TObject);
begin
form14.SpeedButton1.Click;
form14.Hide;
form1.LbSpeedButton4.Click;
end;

procedure TForm14.A1Click(Sender: TObject);
begin
form14.SpeedButton1.Click;
form14.Hide;
form1.LbSpeedButton3.Click;
end;

procedure TForm14.MTERKAYIT1Click(Sender: TObject);
begin
form14.SpeedButton1.Click;
form14.Hide;
form3.LbSpeedButton1.Click;
end;


procedure TForm14.MterismineGre1Click(Sender: TObject);
begin
form14.SpeedButton1.Click;
form14.Hide;
form13.LbSpeedButton6.Click;
end;

procedure TForm14.HayvanaGreArama1Click(Sender: TObject);
begin
form14.SpeedButton1.Click;
form14.Hide;
form13.LbSpeedButton1.Click;
end;

procedure TForm14.k1Click(Sender: TObject);
begin
form1.LbSpeedButton7.Click;
```

```
end;


procedure TForm14.HAYVANKAYIT1Click(Sender: TObject);
begin
form14.SpeedButton1.Click;
form14.Hide;
form3.LbSpeedButton2.Click;
end;


procedure TForm14.Gizle1Click(Sender: TObject);
begin
form1.LbSpeedButton8.Click;
end;

procedure TForm14.RadioButton4Click(Sender: TObject);
begin
if form14.RadioButton4.Checked=true then
  form14.DateTimePicker2.Enabled:=true
else
  form14.DateTimePicker2.Enabled:=false;
end;

procedure TForm14.RadioButton2Click(Sender: TObject);
begin
form14.DateTimePicker2.Enabled:=false;
end;

procedure TForm14.RadioButton3Click(Sender: TObject);
begin
form14.DateTimePicker2.Enabled:=false;
end;

procedure TForm14.RadioButton1Click(Sender: TObject);
begin
form14.DateTimePicker2.Enabled:=false;
end;

procedure TForm14.SpeedButton2Click(Sender: TObject);
begin
   if (form14.RadioButton1.Checked = true) or (form14.RadioButton2.Checked = true) or
(form14.RadioButton3.Checked = true) or (form14.RadioButton4.Checked = true) then
   begin
     dateseparator:='-';
     shortdateformat:='yyyy/mm/dd';
     if (form14.RadioButton1.Checked = true) then
```

```
    begin
      form14.ADOQuery1.SQL.Text:='select
A.Hid,H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi from asilama
A,hayvan H where TekrarAsilamaTarihi <
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and A.Hid=H.Hid';
      form14.ADOQuery1.Open;
      form14.ADOQuery3.SQL.Text:='select
D.Hid,H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi from
disparazit D,hayvan H where TekrarDisParUygTarihi <
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and D.Hid=H.Hid';
      form14.ADOQuery3.Open;
      form14.ADOQuery2.SQL.Text:='select
I.Hid,H.Hismi,I.IcParUygTarihi,I.IcParUygulama,I.TekrarIcParUygTarihi from icparazit
I,hayvan H where TekrarIcParUygTarihi <
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and I.Hid=H.Hid';
      form14.ADOQuery2.Open;
    end
    else if (form14.RadioButton2.Checked = true) then
    begin
      form14.ADOQuery1.SQL.Text:='select
A.Hid,H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi from asilama
A,hayvan H where TekrarAsilamaTarihi >
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and A.Hid=H.Hid';
      form14.ADOQuery1.Open;
      form14.ADOQuery3.SQL.Text:='select
D.Hid,H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi from
disparazit D,hayvan H where TekrarDisParUygTarihi >
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and D.Hid=H.Hid';
      form14.ADOQuery3.Open;
      form14.ADOQuery2.SQL.Text:='select
I.Hid,H.Hismi,I.IcParUygTarihi,I.IcParUygulama,I.TekrarIcParUygTarihi FROM icparazit
I,hayvan H where TekrarIcParUygTarihi >
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and I.Hid=H.Hid';
      form14.ADOQuery2.Open;
    end
    else if (form14.RadioButton3.Checked = true) then
    begin
      form14.ADOQuery1.SQL.Text:='select
A.Hid,H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi from asilama
A,hayvan H where TekrarAsilamaTarihi =
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and A.Hid=H.Hid';
      form14.ADOQuery1.Open;
      form14.ADOQuery3.SQL.Text:='select
D.Hid,H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi from
disparazit D,hayvan H where TekrarDisParUygTarihi =
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and D.Hid=H.Hid';
      form14.ADOQuery3.Open;
```

```
      form14.ADOQuery2.SQL.Text:='select
I.Hid,H.Hismi,I.IcParUygTarihi,I.IcParUygulama,I.TekrarIcParUygTarihi FROM icparazit
I,hayvan H where TekrarIcParUygTarihi =
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and I.Hid=H.Hid';
       form14.ADOQuery2.Open;
     end
     else if (form14.RadioButton4.Checked = true) then
     begin
      form14.ADOQuery1.SQL.Text:='select
A.Hid,H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi from asilama
A,hayvan H where TekrarAsilamaTarihi between
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and
'+#39+datetostr(form14.DateTimePicker2.Date)+#39+' and A.Hid=H.Hid';
      form14.ADOQuery1.Open;
      form14.ADOQuery3.SQL.Text:='select
D.Hid,H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi from
disparazit D,hayvan H where TekrarDisParUygTarihi between
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and
'+#39+datetostr(form14.DateTimePicker2.Date)+#39+' and D.Hid=H.Hid';
      form14.ADOQuery3.Open;
      form14.ADOQuery2.SQL.Text:='select
I.Hid,H.Hismi,I.IcParUygTarihi,I.IcParUygulama,I.TekrarIcParUygTarihi from icparazit I,
hayvan H where TekrarIcParUygTarihi between
'+#39+datetostr(form14.DateTimePicker1.Date)+#39+' and
'+#39+datetostr(form14.DateTimePicker2.Date)+#39+' and I.Hid=H.Hid';
      form14.ADOQuery2.Open;
    end
   end
   else
    application.MessageBox('PLEASE SELECT ANY DATE CRITERIA','SELECT
CRITERIA',MB_OK+MB_ICONEXCLAMATION);

end;

procedure TForm14.SpeedButton3Click(Sender: TObject);
begin
 FORM14.RadioButton1.Checked:=FALSE;
 FORM14.RadioButton2.Checked:=FALSE;
 FORM14.RadioButton3.Checked:=FALSE;
 FORM14.RadioButton4.Checked:=FALSE;
 FORM14.DateTimePicker2.Enabled:=FALSE;
 FORM14.DateTimePicker1.Date:=date;
 FORM14.DateTimePicker2.Date:=date;
 FORM14.ADOQuery1.Close;
 FORM14.ADOQuery2.Close;
 FORM14.ADOQuery3.Close;
end;
```

```
procedure TForm14.DBGrid1DblClick(Sender: TObject);
begin
  IF (FORM14.DBGrid1.FieldCount <> 0) and (form14.DBGrid1.Fields[0].Text <> null) and
(form14.DBGrid1.Fields[0].Text <> '') THEN
  BEGIN
    gnlsnc:=strtoint(form14.DBGrid1.Fields[0].text);
    form10.Show;
  end;
end;

procedure TForm14.DBGrid2DblClick(Sender: TObject);
begin
  IF (FORM14.DBGrid2.FieldCount <> 0) and (form14.DBGrid2.Fields[0].Text <> null) and
(form14.DBGrid2.Fields[0].Text <> '') THEN
  BEGIN
    gnlsnc:=strtoint(form14.DBGrid2.Fields[0].text);
    form10.Show;
  end;
end;

procedure TForm14.DBGrid3DblClick(Sender: TObject);
begin
  IF (FORM14.DBGrid3.FieldCount <> 0) and (form14.DBGrid3.Fields[0].Text <> null) and
(form14.DBGrid3.Fields[0].Text <> '') THEN
  BEGIN
    gnlsnc:=strtoint(form14.DBGrid3.Fields[0].text);
    form10.Show;
  end;
end;

end.
```

FORM 15 CODES

```
unit Unit15;

interface

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

type
  TForm15 = class(TForm)
    Label1: TLabel;
```

```
    GroupBox1: TGroupBox;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Panel1: TPanel;
    ADOQuery1: TADOQuery;
    DataSource1: TDataSource;
    ADOQuery2: TADOQuery;
    ADOQuery3: TADOQuery;
    DataSource2: TDataSource;
    DataSource3: TDataSource;
    procedure FormShow(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form15: TForm15;

implementation

uses Unit2;

{$R *.dfm}

procedure TForm15.FormShow(Sender: TObject);
begin
//FORM15.Memo1.Lines.Clear;
//form15.Label1.Caption:='BUGUN
'+FORMATDATETIME('dddddd',NOW)+#13+'YAPILACAK İŞLER'+#13+#13+'AŞILAMA: 1
TANE'+#13+'DIŞ PARAZİT: '+#13+'İÇ PARAZİT: 2 TANE'+#13+#13+'DETAYLARI İŞ
HAZIRLA BÖLÜMÜNDEN'+#13+'GÖREBİLİRSİNİZ'+#13+#13+#13+'PASTEUR
VETERİNER KLİNİĞİ');
//FORM15.Memo1.Lines.Add(' ');
form15.Label1.Caption:='BUGÜN '+FORMATDATETIME('DDDDDD',NOW);
{FORM15.Memo1.Lines.Add('');
FORM15.Memo1.Lines.Add('YAPILACAK İŞLER:');
FORM15.Memo1.Lines.Add('');
FORM15.Memo1.Lines.Add('AŞILAMA: 2 TANE');
FORM15.Memo1.Lines.Add('DIŞ PARAZİT: YOK');
FORM15.Memo1.Lines.Add('İÇ PARAZİT: 1 TANE');
```

```
FORM15.Memo1.Lines.Add('');
FORM15.Memo1.Lines.Add('DETAYLARI İŞ HAZIRLA BÖLÜMÜNDEN');
FORM15.Memo1.Lines.Add('GÖREBİLİRSİNİZ');
FORM15.Memo1.Lines.Add('');
 FORM15.Memo1.Lines.Add('');
 FORM15.Memo1.Lines.Add('PASTEUR VETERİNER KLİNİĞİ');}
  dateseparator:='-';
  shortdateformat:='yyyy/mm/dd';

  FORM15.ADOQuery1.SQL.Text:='SELECT COUNT(TekrarAsilamaTarihi) as sayi from
 asilama where TekrarAsilamaTarihi='+#39+datetostr(date)+#39;
  form15.ADOQuery1.Open;
  if form15.ADOQuery1['sayi'] <> 0 then
    form15.Label5.Caption:=inttostr(form15.ADOQuery1['sayi'])+' TANE'
  else
    form15.Label5.Caption:='YOK';

  FORM15.ADOQuery2.SQL.Text:='SELECT COUNT(TekrarDisParUygTarihi) as Dsayi from
 disparazit where TekrarDisParUygTarihi='+#39+datetostr(date)+#39;
  form15.ADOQuery2.Open;
  if form15.ADOQuery2['Dsayi'] <> 0 then
    form15.Label6.Caption:=inttostr(form15.ADOQuery2['Dsayi'])+' TANE'
  else
    form15.Label6.Caption:='YOK';

  FORM15.ADOQuery3.SQL.Text:='SELECT COUNT(TekrarIcParUygTarihi) as Isayi from
 icparazit where TekrarIcParUygTarihi='+#39+datetostr(date)+#39;
  form15.ADOQuery3.Open;
  if form15.ADOQuery3['Isayi'] <> 0 then
    form15.Label7.Caption:=inttostr(form15.ADOQuery3['Isayi'])+' TANE'
  else
    form15.Label7.Caption:='YOK';

    dateseparator:='.';
 shortdateformat:='dd/mm/yyyy';

 end;

 end.
```

FORM 16 CODES

```
unit Unit16;

interface

uses
```

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

type
 TForm16 = class(TForm)
  LbSpeedButton1: TLbSpeedButton;
  procedure LbSpeedButton1Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 Form16: TForm16;

implementation

uses Unit14, Unit1;

{$R *.dfm}

procedure TForm16.LbSpeedButton1Click(Sender: TObject);
begin
 FORM14.LABEL1.VISIBLE:=TRUE;
 FORM14.PANEL1.VISIBLE:=FALSE;
 FORM14.Caption:='DETAILS OF TODAY';
 FORM14.SHOW;
      form14.ADOQuery1.SQL.Text:='select
A.Hid,H.Hismi,A.AsilamaTarihi,A.YapilanAsilama,A.TekrarAsilamaTarihi from asilama
A,hayvan H where TekrarAsilamaTarihi = '+#39+datetostr(Date)+#39+' and A.Hid=H.Hid';
      form14.ADOQuery1.Open;
      form14.ADOQuery3.SQL.Text:='select
D.Hid,H.Hismi,D.DisParUygTarihi,D.DisParUygulama,D.TekrarDisParUygTarihi from
disparazit D,hayvan H where TekrarDisParUygTarihi = '+#39+datetostr(Date)+#39+' and
D.Hid=H.Hid';
      form14.ADOQuery3.Open;
      form14.ADOQuery2.SQL.Text:='select
I.Hid,H.Hismi,I.IcParUygTarihi,I.IcParUygulama,I.TekrarIcParUygTarihi from icparazit
I,hayvan H where TekrarIcParUygTarihi = '+#39+datetostr(Date)+#39+' and I.Hid=H.Hid';
      form14.ADOQuery2.Open;
 FORM16.Hide;
 FORM1.HIDE;
end;

end.
```

BEGSOFT PROJECT CODES

```
program begsoft;

uses
  Forms,
  WINDOWS,
  Unit1 in 'Unit1.pas' {Form1},
  Unit2 in 'Unit2.pas' {Form2},
  Unit3 in 'Unit3.pas' {Form3},
  Unit4 in 'Unit4.pas' {Form4},
  Unit5 in 'Unit5.pas' {Form5},
  Unit6 in 'Unit6.pas' {Form6},
  Unit7 in 'Unit7.pas' {Form7},
  Unit8 in 'Unit8.pas' {Form8},
  Unit9 in 'Unit9.pas' {Form9},
  Unit10 in 'Unit10.pas' {Form10},
  Unit11 in 'Unit11.pas' {Form11},
  Unit12 in 'Unit12.pas' {Form12},
  Unit13 in 'Unit13.pas' {Form13},
  Unit14 in 'Unit14.pas' {Form14},
  Unit15 in 'Unit15.pas' {Form15},
  Unit16 in 'Unit16.pas' {Form16};
{$R *.res}


begin
```

```
  Application.Initialize;
  Application.CreateForm(TForm1, Form1);
  Application.CreateForm(TForm2, Form2);
  Application.CreateForm(TForm3, Form3);
  Application.CreateForm(TForm4, Form4);
  Application.CreateForm(TForm5, Form5);
  Application.CreateForm(TForm6, Form6);
  Application.CreateForm(TForm7, Form7);
  Application.CreateForm(TForm8, Form8);
  Application.CreateForm(TForm9, Form9);
  Application.CreateForm(TForm10, Form10);
  Application.CreateForm(TForm11, Form11);
  Application.CreateForm(TForm12, Form12);
  Application.CreateForm(TForm13, Form13);
  Application.CreateForm(TForm14, Form14);
  Application.CreateForm(TForm16, Form16);
  //Application.CreateForm(TForm15, Form15);
  FORM15:= TFORM15.CREATE(APPLICATION);
  FORM15.SHOW;
  FORM15.UPDATE;
  SLEEP(4000);
  FORM15.HIDE;
  FORM15.FREE;
  Application.Run;
end.
```

DATABASE CREATION CODES

```
# Host: localhost
# Database: pasteur
# Table: 'vaccinate'
#
CREATE TABLE `vaccinate` (
  `Hid` int(6) NOT NULL default '0',
  `Vaccinate Date` date NOT NULL default '0000-00-00',
  `Done Vaccinate` varchar(50) NOT NULL default '',
  `Again Vaccinate Date` date NOT NULL default '0000-00-00',
  `Aoperationcode` varchar(25) NOT NULL default '',
  PRIMARY KEY  (`Aoperationcode`)
) TYPE=MyISAM;

# Host: localhost
# Database: pasteur
# Table: 'outerparasite'
#
CREATE TABLE `outerparasite` (
  `Hid` int(6) NOT NULL default '0',
  `OuterparasiteDoneDate` date NOT NULL default '0000-00-00',
  `OuterparasiteDone` varchar(50) NOT NULL default '',
```

```
  ` AgainOuterparasiteDoneDate ` date NOT NULL default '0000-00-00',
  `Doperationcode` varchar(25) NOT NULL default '',
  PRIMARY KEY  (`Doperationcode`)
) TYPE=MyISAM;

# Host: localhost
# Database: pasteur
# Table: 'animal'
#
CREATE TABLE `animal` (
  `Hid` int(6) NOT NULL auto_increment,
  `Hname` varchar(50) NOT NULL default '',
  `Hkind` varchar(50) NOT NULL default '',
  `Hrace` varchar(50) NOT NULL default '',
  `Hsex` varchar(10) NOT NULL default '',
  `Hbirthdate` date NOT NULL default '0000-00-00',
  `Sid` int(6) NOT NULL default '0',
  `Hoperationcode` varchar(25) NOT NULL default '',
  PRIMARY KEY  (`Hid`)
) TYPE=MyISAM;
# Host: localhost
# Database: pasteur
# Table: 'innerparasite'
CREATE TABLE `innerparasite` (
  `Hid` int(6) NOT NULL default '0',
```

```
`InnerparasiteDoneDate` date NOT NULL default '0000-00-00',
`InnerparasiteDone` varchar(50) NOT NULL default '',
` AgainInnerparasiteDoneDate ` date NOT NULL default '0000-00-00',
`Ioperationcode` varchar(25) NOT NULL default '',
PRIMARY KEY (`Iperationcode`)
) TYPE=MyISAM;
# Host: localhost
# Database: pasteur
# Table: 'customer'
CREATE TABLE `customer` (
`Sid` int(6) NOT NULL auto_increment,
` Cname ` varchar(50) NOT NULL default '',
`Csurname` varchar(50) NOT NULL default '',
`Address` varchar(250) NOT NULL default '',
`Homephone` varchar(14) NOT NULL default '',
`Mobilphone` varchar(14) NOT NULL default '',
`Workphone` varchar(14) NOT NULL default '',
`Fax` varchar(13) NOT NULL default '',
`Extraphone` varchar(13) NOT NULL default '',
`Eposta` varchar(100) NOT NULL default '',
`Web` varchar(100) NOT NULL default '',
Mislemkodu` varchar(25) NOT NULL default '',
PRIMARY KEY (`Sid`)
TYPE=MyISAM;
```