Near East University

Faculty of Engineering

Department of Computer Engineering

Using Computer Programming Technique to keep Traffic Accidents Database

Graduation Project
Com - 400

Student: Gökalp BAŞ (20021917)
Supervisor: Mr. Ümit SOYER

Nicosia - 2005

Acknowledgements

First of all, I would like to thank to my Vice Chairman Assoc. Prof. Dr. Rahib ABIYEV.He gave me a subject that I looked forward to express my decision.

I wish thank to my supervisor, Mr. Ümit SOYER. Under his guidance, I successfully overcome many deficiencies .He gave materials about my project and he taught me how to use them.

Finally, I want to thank to my family, friends and especially my fiancée; without their endless support and encouragement I could never prepare this project.

Abstract

What is actually traffic or what it should be? If you look at the dictionary it is the moving along or through an area or route the vehicles, vessels or people (page 1055 Macmillan contemporary dictionary).It is also a passage or flow of vehicles vessels, and people or the like along or through an area or route. The meanings of these sentences are clear enough .But what we do as a driver or as a (pedestrian, walker) pedestrian? Almost no one of us take care for the others; especially the drivers. They think they are the king of the roads. They drive as fast as they can, they park anywhere they like. They don't help each other so everybody has his own car. The pedestrians don't take any care, when they cross the road. They don't find any place to walk too.

The roads are not enough for the vehicles. They are too narrow and not good built. Nobody cares the future when they want to open a new road and nobody knows (?) how to built .I think that to study something doesn't mean to know that. So the roads are not sufficient for the cars .Everybody has a car but doesn't have a respect to each other.

The matters that I mentioned above are the reasons for the accidents .If a sign on a road is only to stay there and not to take care there will be more accidents .We shouldn't forget the alcohol .It doesn't stay in stomach, as it stays in a bottle. Actually the main reason of accidents is alcohol with young drivers .They are buying a car before they get a driving license and they start to drive the car while they are drunk. So who is the real responsible of that? Of course the families have the responsibility.

We can talk too many thinks about the traffic of North Cyprus .In this project our main objective is to analysis these events and try to reduce them and more. And the following project is a way of realize it.

TABLE OF CONTENTS

INTRODUCTION

Delphi is Borland's best-selling rapid application development (RAD) product for writing Windows applications. With Delphi you can write Windows programs more quickly and easily then was ever possible before. You can create win32 console applications or win32 graphical user interface (GUI programs).When Creating win32 GUI applications with Delphi; you have all the power of a true compiled programming language (Object Pascal).Wrapped in a RAD environment.

The thesis consists of the introduction and two chapters.

Chapter-1 presents what is Delphi programming? , development environment, programming language, what is database and database Programming? , Connecting to a database, PARADOX Database, how to creating an alias, sql in Delphi.

Chapter-2 presents forms of project and codes.

Chapter 1.A Brief Overview of Delphi

1.1 What is Delphi programming ?

Delphi is a programming language and software development environment. It is produced by Borland (known for a time as Inprise). The Delphi language, formerly known as Object Pascal (Pascal with object-oriented extensions) originally targeted only Microsoft Windows, but now builds native applications for Linux and the Microsoft .NET framework as well

1.1.1 Development environment

Delphi's most popular use is the development of desktop and enterprise database applications, but as a general-purpose development tool it is capable of and used for most types of development projects. It was one of the first of what came to be known as RAD tools, for Rapid Application Development, when released in 1995 for 16-bit Windows. Delphi 2, released a year later, supported 32-bit Windows environments, and a C++ version, C++Builder, followed a few years after. In 2001 a Linux version known as Kylix became available. With one new major release every year, in 2002 support for Linux (through Kylix and the CLX component library) was added and in 2003 .NET became supported in Delphi.Net (Delphi 8).

The chief architect behind Delphi, and its predecessor Turbo Pascal, was Anders Hejlsberg until he left for Microsoft in 1996 where he is the chief designer of C# and a key participant in the creation of the Microsoft .NET Framework. Full support for .NET was added in Delphi 8 (released December 2003). Delphi 8, which compiles Object Pascal code for the .NET framework, changed its IDE for the first time since its conception to a look and feel similar to Microsoft's Visual Studio for .NET.

Delphi 2005 (brand name for Delphi 9) provides both win32 and .NET code generation, and has as its most notable new feature design-time manipulation of live data from a database. It also includes a significantly improved IDE.

Delphi's proponents claim that having the Delphi Language, IDE and component library (VCL/CLX) supplied by a single vendor allows for a more internally consistent and recognizable package.
The Delphi product is distributed as various suites: Personal, Professional, Enterprise (formerly Client/Server) and Architect.

1.1.2 Programming language

The main distinguishing features of Delphi and Kylix from other IDEs are the Delphi language, the VCL/CLX (Visual Component Library), strong emphasis on database connectivity, and large number of third party components.
Notable aspects of the Delphi language include:

- Transparent handling of objects as references/pointers
- Properties as part of the language; that is, member getters and setters (aka accessors and mutators), which transparently encapsulate the access to member fields
- Index Properties and Default Properties to provide access to collections
- Delegates aka type safe method pointers which are used to wire the events triggered by the components
- Delegation of interface implementation to a field or property of the class
- Implementation of Windows message handlers by tagging a method of a class with the number/name of the windows message to handle
- COM independent interfaces with reference counted class implementations

1.2 What is database and database Programming ?

The word database is a composition of "data" and "base". "Data" are facts of the real world, which is supplied by "base". The word "data base" was first used in the beginning of 1960's, then written as "data-base" and recently "database".

There is another word, file, in the meaning of collection of facts of the real world. However, a file is an accumulation of data of the same kind of structure. On the other hand, a database is an integration of different kind of data.

This concept of database is deeply concerned with the development of computers. The computers were getting regarded as information processing machines rather than machines that calculate. As a result, people demanded to use computers to store fastly increasing data of the real world and utilize them communally with ease. For multi-purpose use, not file systems but database managemant systems(DBMS) were expected powerful in this sense. The purposes of database systems are as follows.

- to store enormous amount of data efficiently
- multi-purpose utilization of data
- effective and easy utilization of data

In case of knowledge information processing, data are more complex.

There are following good points from the view point of using integrated data communally as databases.

- Redundancy of data decreases due to integration and multi-purpose use of data
- Unnecessary to keep redundant data consistent
- Easy to protect data from attacks of users without access capabilities
- Unnecessary to create the same data for each program

Then, what fuctions of databases are required ? Let's start with the universal and basic functions.

1. Data Definition Language or DDL to represent integrated information of the real world

2. Data Manipulation Language or DML to manipulate information in databases
3. Data independence to reduce influence on programs when data changes not only in contents but also in structures
4. Concurrency control to keep consistency of data when updated by plural users or user programs at the same time
5. Integrity constraints to justify the contents of databases, which are not given to each user but to databases in order to prevent redundancy and omissions
6. Recovery from database destruction on account of hardware, software and human errors or accidents.

In a relational database, a designer can store data in separate, but related, tables and extract data in the desired form from several tables at once as if they were all one. **Access, Oracle ,PARADOX** and **MS SQL Server** are common relational database management systems. The giants of the industry are Microsoft's SQL Server, IBM's DB2 and Oracle.

Knowing one or more of the major database systems continues to be a very positive career move. "In today's business environment, information is money. The better a business collects and uses information, the more profitable it is. The ability to design computer programs that collect information, store it in a well-structured format, and use that information in more effective ways has been in demand for some time. With the advent of the Internet, the demand for this ability has become even stronger." In today's economy, many employers are seeking more highly-skilled database professionals than ever before (partially as a form of belt-tightening), but openings are definitely out there.

1.2.1 Connecting to a database

Delphi database connectivity.

With Delphi, we can connect to different types of databases: local or client/server (remote server) database. Local databases are stored on your local drive or on a local area network. Remote database servers usually reside on a remote machine. Types of local databases are Paradox, dBase and MS Access. Types of client/server databases are MS SQL Server or Oracle.
Local databases are often called single-tiered databases. A single-tiered database is a database in which any changes, such as editing the data, inserting records, or deleting records - happen immediately. Single-tiered databases are limited in how much data the tables can hold and the number of users your application can support. When the database information includes complicated relationships between several tables, or when the number of clients grows, you may want to use a two-tiered or multi-tiered application. Client applications run on local machines; the application server is typically on a server, and the database itself might be on another server. The idea behind the multi-tier architecture is that client applications can be very small because the application servers do most of the work. This enables you to write what are called thin-client applications.

When we write a database application in Delphi, we need to use some *database engine* to access a data in a database. The database engine permits you to concentrate on what data you want to access, instead of how to access it. From the first version, Delphi provides database developers with the BDE (Borland Database Engine). Beside the BDE, Delphi from the fifth version supports Microsoft ADO database interface.

1.2.2 BDE? ADO?

1.2.3 The Borland Database Engine

The BDE is a common data access layer for all of Borland's products, including Delphi and C++Builder. The BDE consists of a collection of DLLs and utilities. The beauty of the BDE is the fact that all of the data manipulation is considered "transparent" to the developer. BDE comes with a set of drivers that enables your application to talk to several different types of databases. These drivers translate high-level database commands (such as open or post) and tasks (record locking or SQL construction) into commands specific to a particular database type: **Paradox, dBASE, MS Access or any ODBC data source**. The BDE API (Application Programming Interface) consists of more than 200 procedures and functions, which are available through the BDE unit. Fortunately, you almost never need to call any of these routines directly. Instead, you use the BDE through the VCL's data access components, which are found on the Data Access page of Component Palette. To access the particular database the application only needs to know the Alias for the database and it will have access to all data in that database. The alias is set up in the BDE Administrator and specifies driver parameters and database locations. The BDE ships with a collection of database drivers, allowing access to a wide variety of data sources. The standard (native) BDE drivers include Paradox, dBase, MS Access, ASCII text. Of course, any ODBC driver can also be used by the BDE through the ODBC Administrator.

Delphi applications that use the BDE to access databases require that you distribute the BDE with the application. When deploying the BDE with an application, you must use InstallShield Express or another Borland certified installation program.

The BDE has several advantages as well as disadvantages as a database engine. It's not my intention to discuss about why and when you should (or not) use the BDE approach over some non-BDE technique.

1.2.4 The ADO programming model.

To access any kind of database with ADO, you'll of course need to have ADO/OLE DB libraries. Everything you need to use ADO is probably already on your computer: the files are distributed by Microsoft as a part of Windows 98/2000. If you or your client use Windows 95 or Windows NT you will probably need to distribute and install the ADO engine. Delphi 5's CD includes an installation of MDAC - Microsoft Data Access Components. You should always make sure to have the latest version, which is available from Microsoft. The Microsoft Data Access Components are the key technologies that enable Universal Data Access. They include ActiveX Data Objects (ADO), OLE DB, and Open Database Connectivity (ODBC).
Note: to install correctly on a Windows 95 computer, MDAC requires that DCOM95 be installed. MDAC installs components that rely on DLLs installed by DCOM95 in order to register correctly. Note that DCOM95 is not required on a Windows NT 4.0. In some cases, DCOM may not be installed on a Windows 98 computer. If it has not been installed, then DCOM98 should be installed prior to the installation of MDAC.

1.2.5 ADO Objects

The ADO programming model is built around several ADO objects that provide you with the productive means for accessing all kinds of data sources. These objects provide the functionality to connect to data sources, query and update record sets, and report errors. Delphi, through several VCL components provides wrapper components to access those objects. Let's see what are some of the Objects ADO works with:

The *Connection* object represents a connection to the data source with the connection strings. In BDE/Delphi a Connection object is a combination of the Database and Session components.

The *Command* object enables us to operate on a data source. Ir represents a command (also known as a query or statement) that can be processed to add, delete, query or update the data in a database.

The *Recordset* object is a result of a Query command. You can think of a Recordset as a Delphi Table or Query component. Each row that the Recordset returns consists of multiple *Field* objects.

1.3 PARADOX Database

One of the local Database of BDE

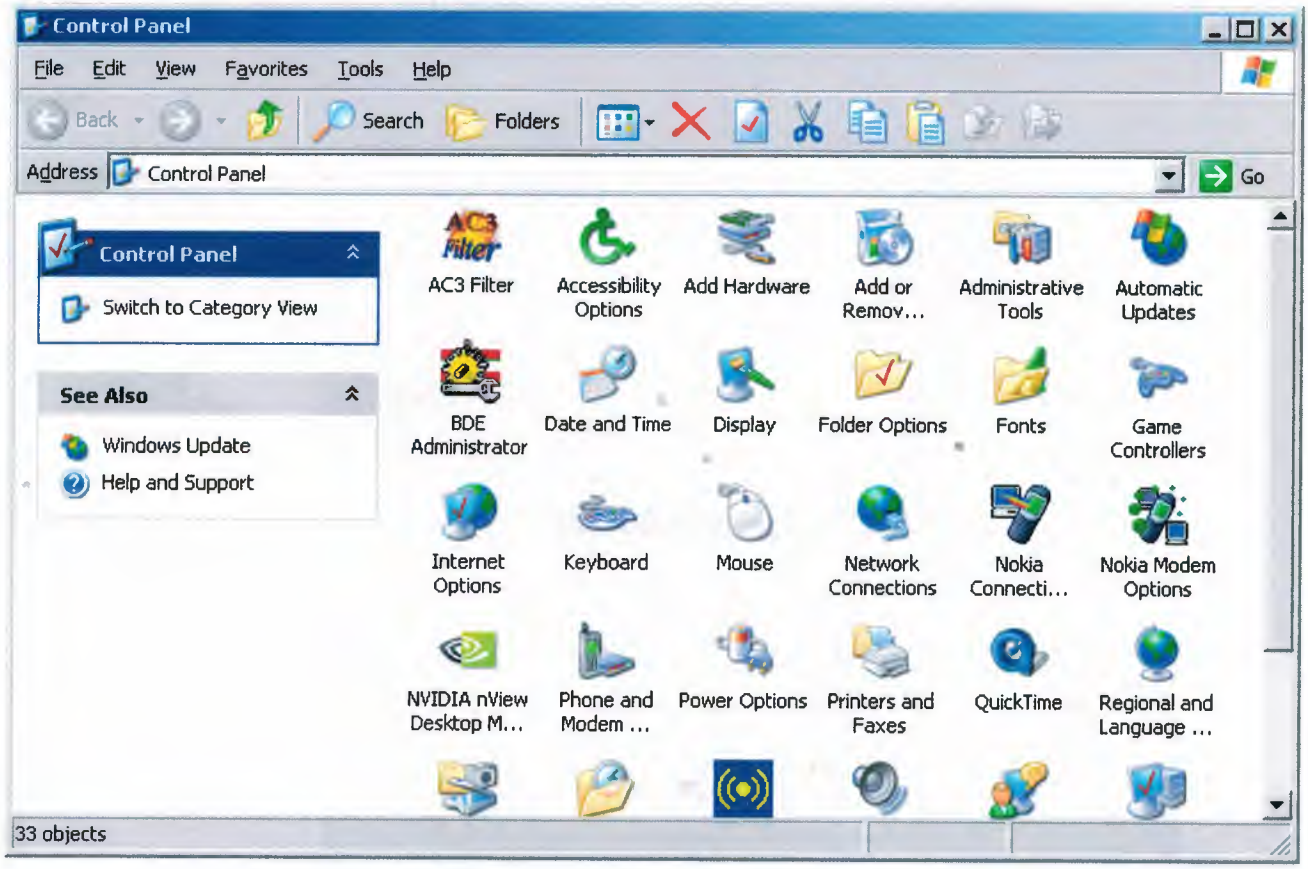
Simple Example for PARADOX:

1.3.1 Creating a Alias Under PARADOX

Before creating alias under Paradox Database Delphi 6 or newest version has to be installed on your computer.

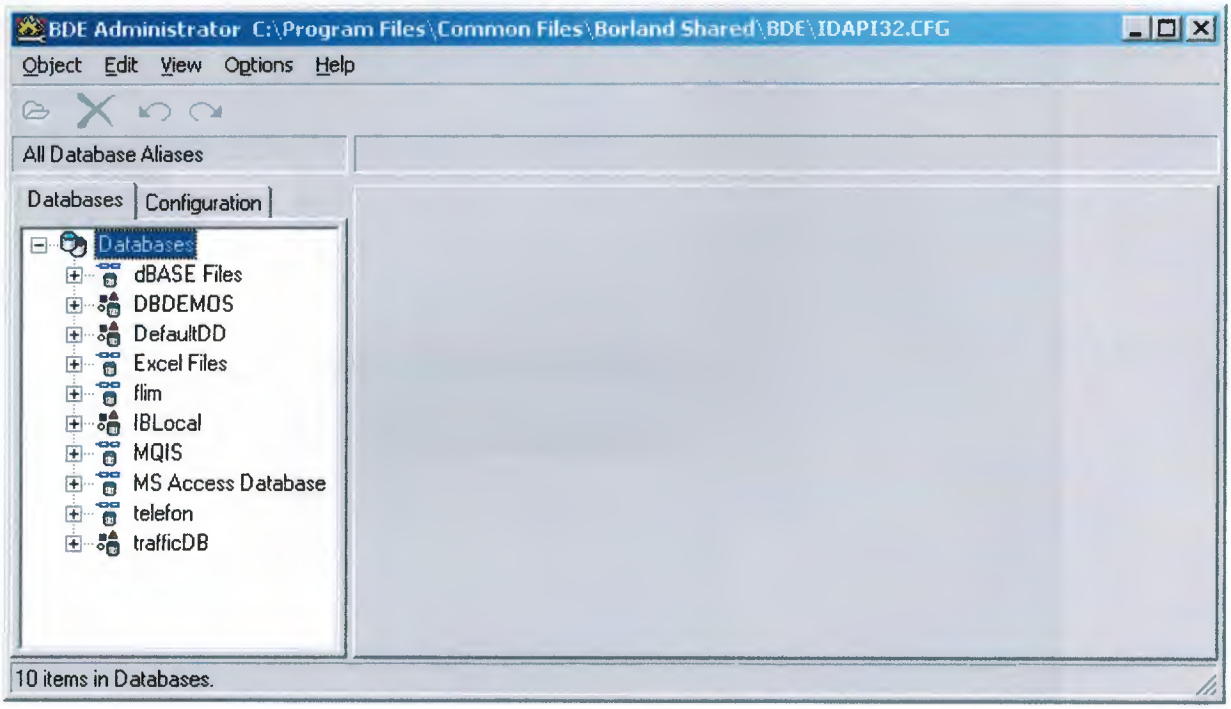
1) Start → Settings → Control Panel

and then you are required to see that image:



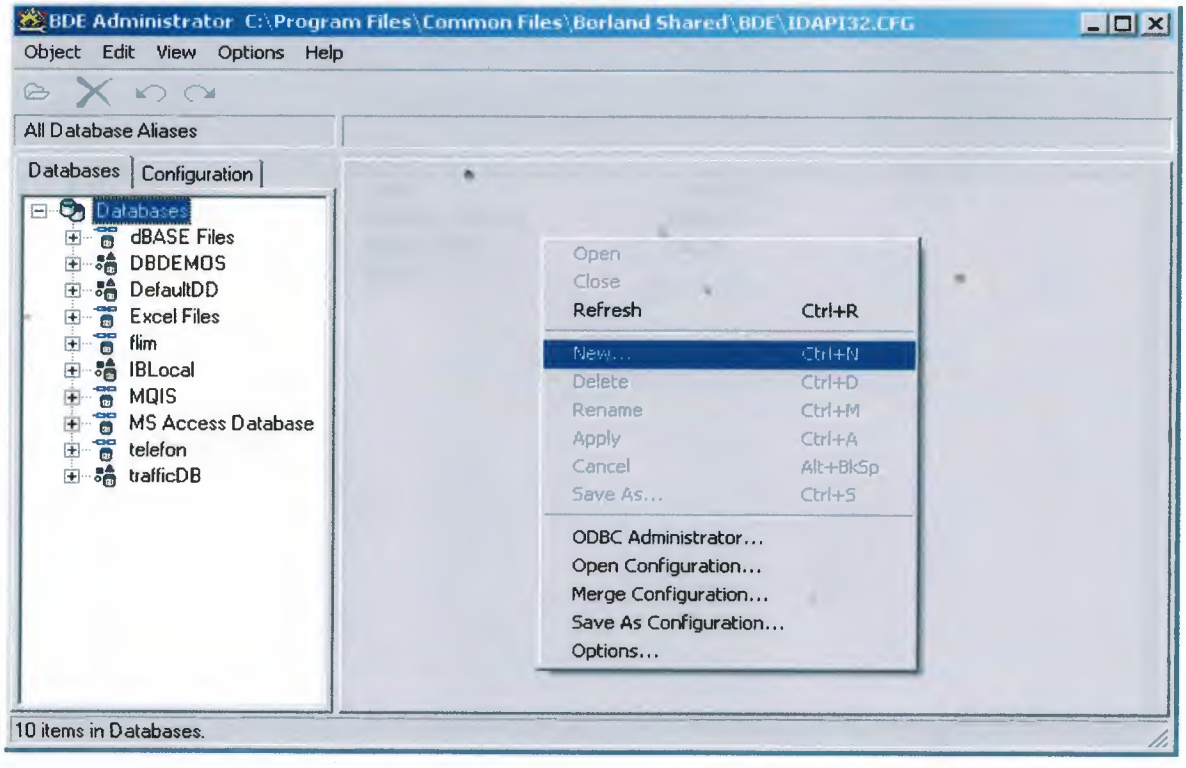
2) Double Click the BDE Adminstrator

After double clicking BDE Adminstrator you have to see the following image:

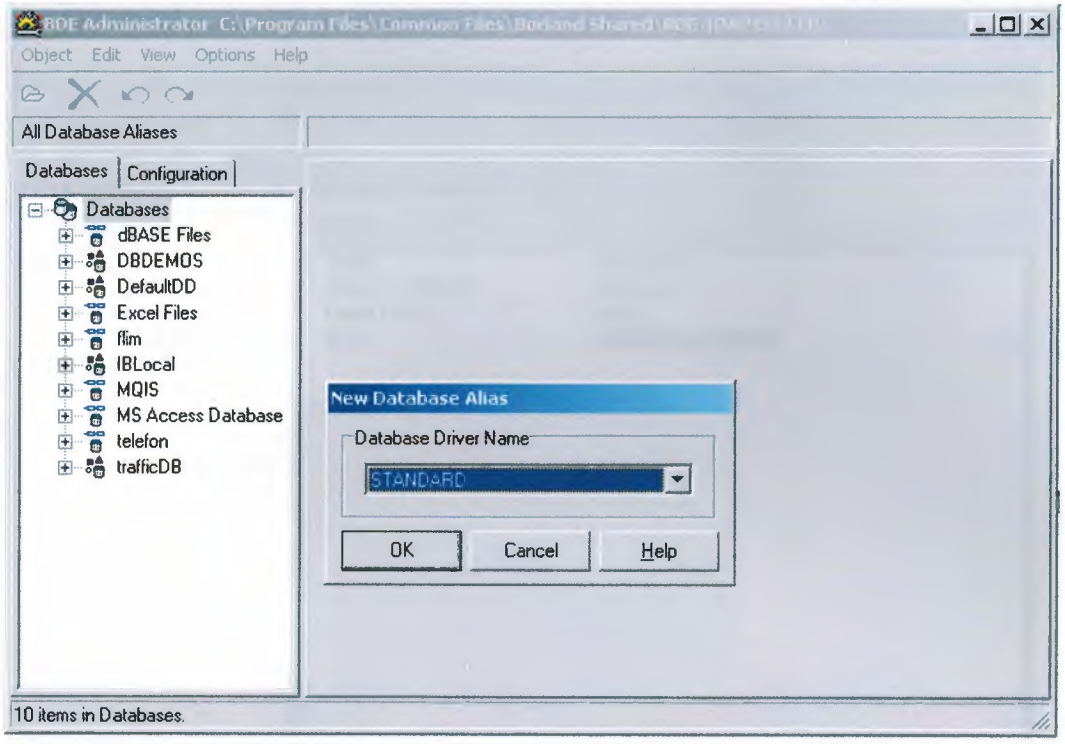


3) At the right side you will select "new" option by right clicking

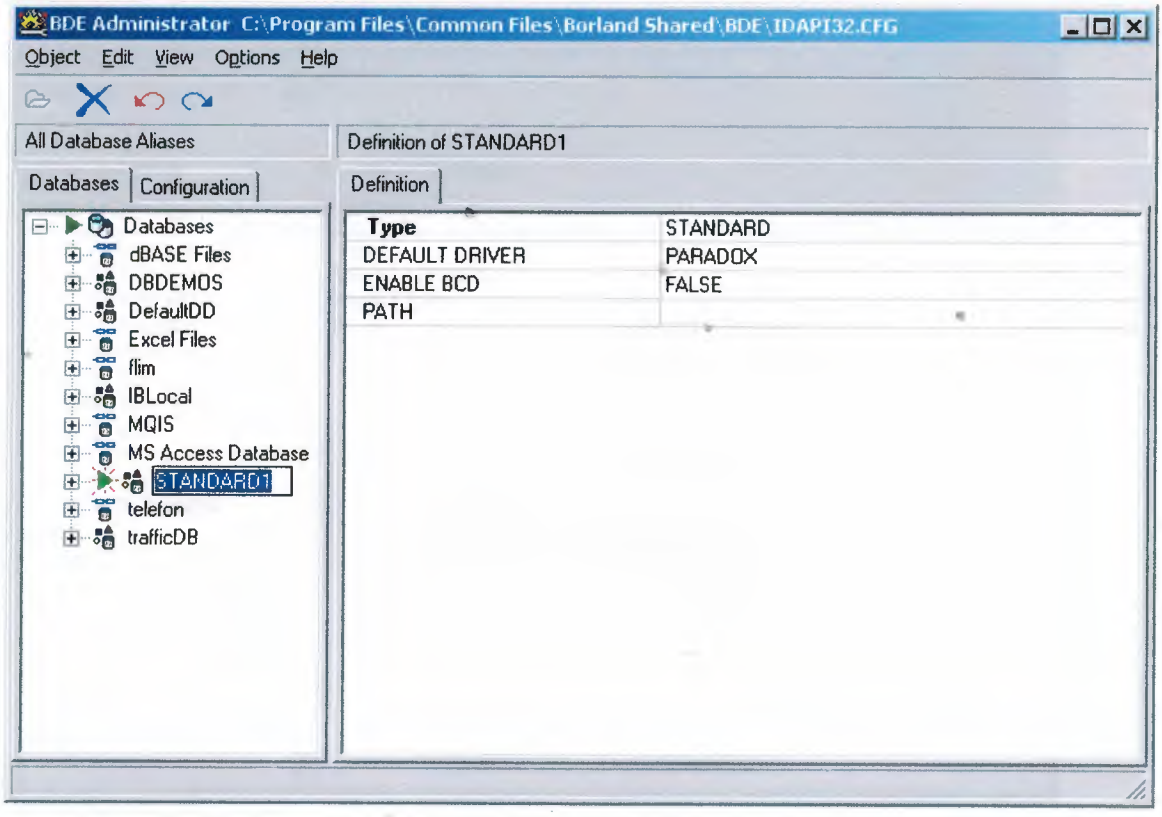
image:



4) After right clicking you will press OK button by Selecting STANDARD database driver name under title of "New Database Alias"image:

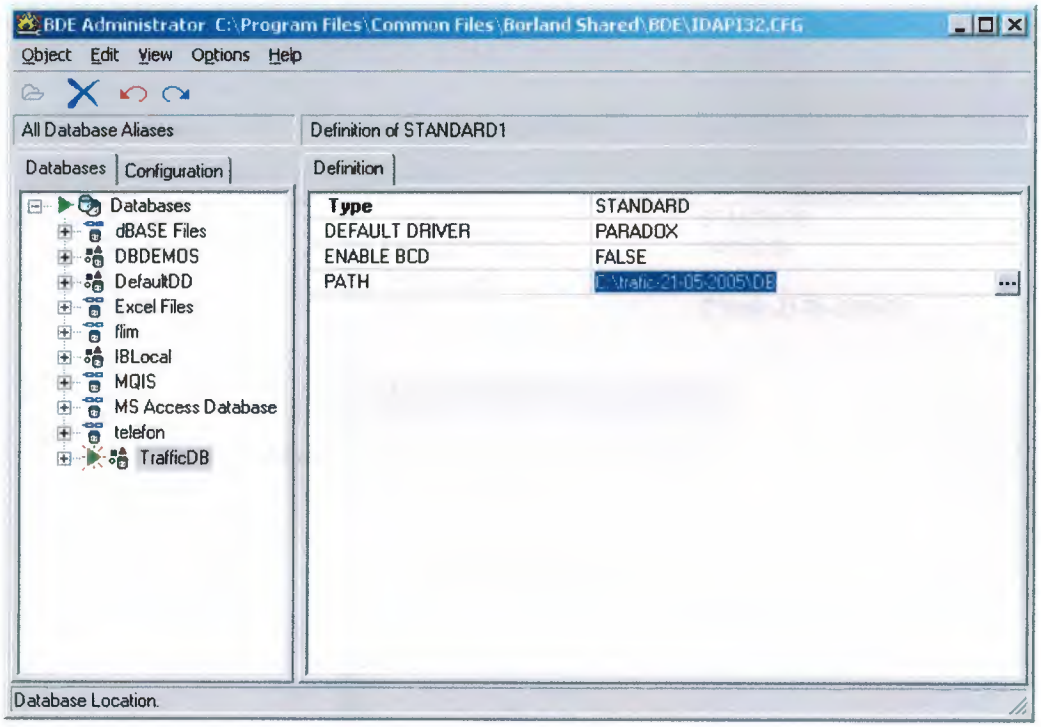


5) Now,BDE waits for your typing database name for your tables like "trafficDB".

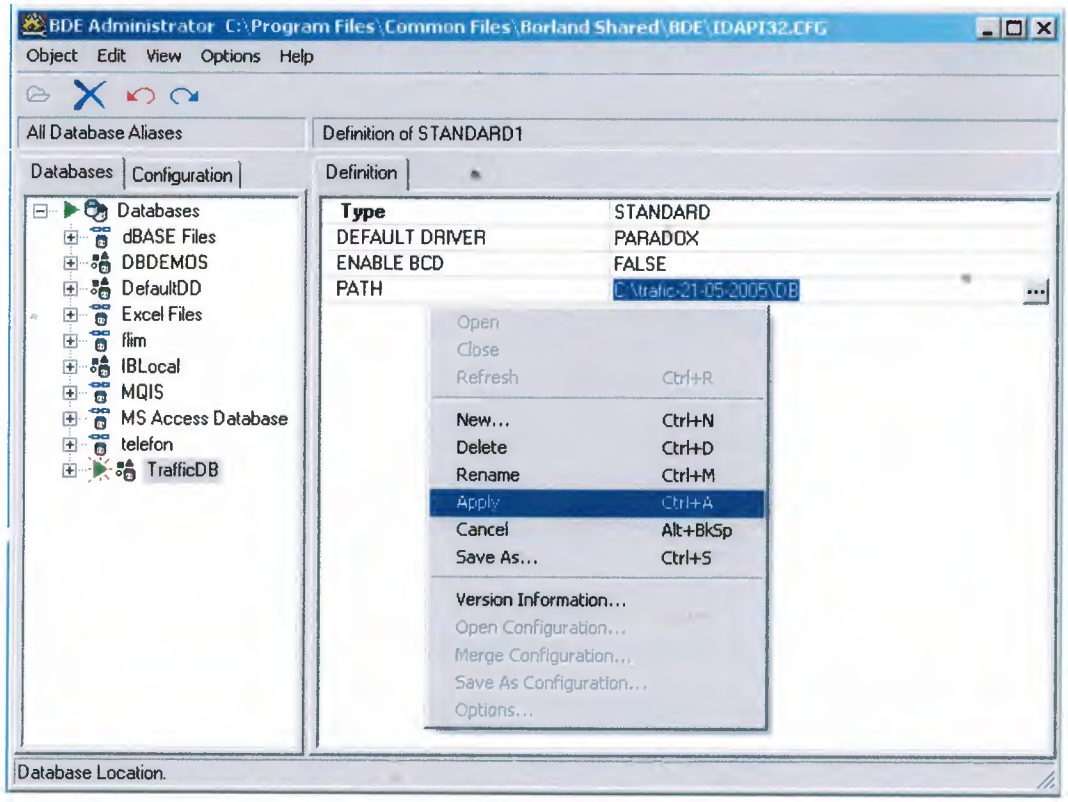


6)After wrting your database name you have to choose your program database folder for your tables by using "Path" at the right side.

image:

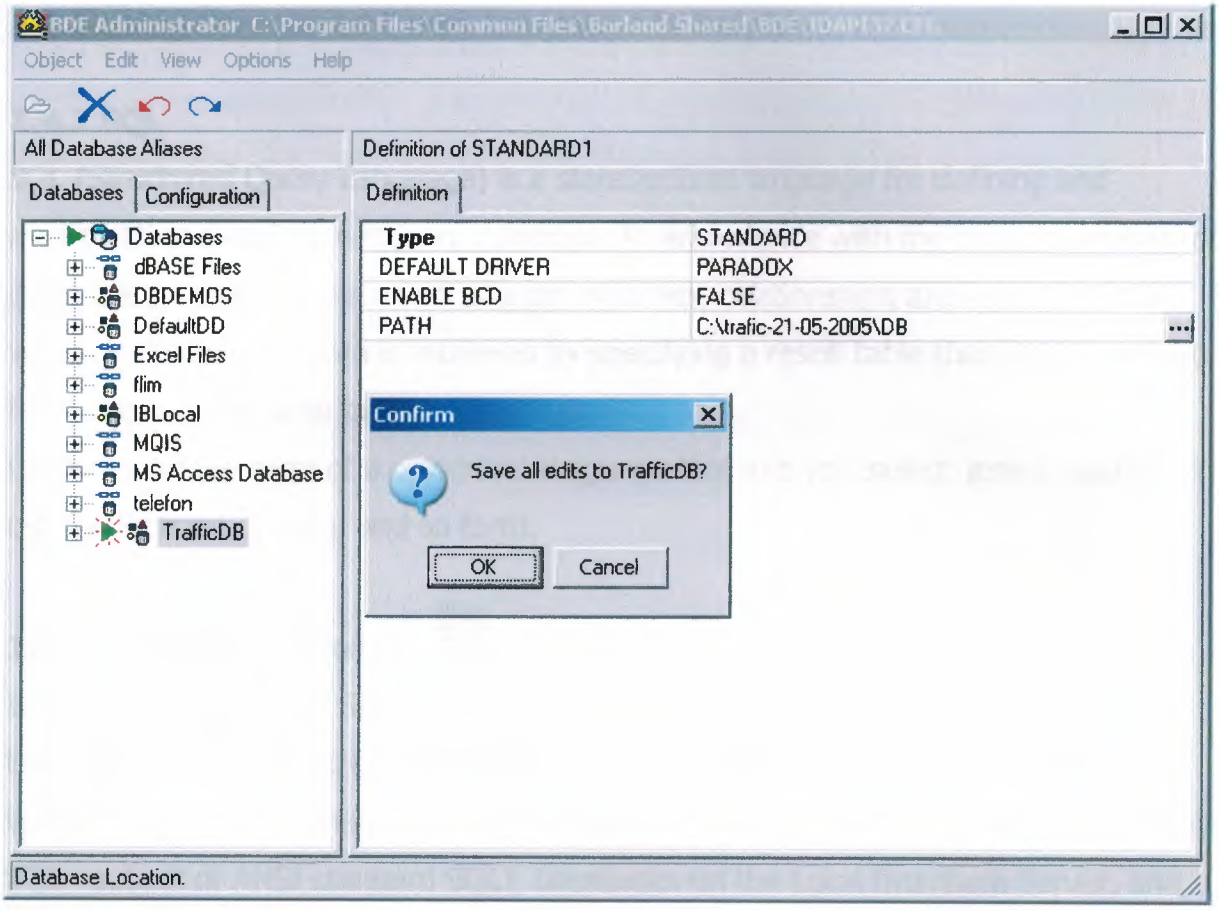


7) And, now again you will do right click with your mouse and choose "Apply".



8) At the final you will confirm the question that will be ask to you

and then press OK button.



1.4 Sql in Delphi

1.4.1 Using Structured Query Language in Delphi

1.4.2 SQL

SQL (Structured Query Language) is a standardized language for defining and manipulating data in a relational database. In accordance with the relational model of data, the database is perceived as a set of tables, relationships are represented by values in tables, and data is retrieved by specifying a result table that can be derived from one or more base tables.
Queries take the form of a command language that lets you *select, insert, update, find* out the location of data, and so forth.

1.4.3 in Delphi ... TQuery 

If you are going to use SQL in your applications, you will become very familiar with the *TQuery* component. Delphi enables your applications to use SQL syntax directly though TQuery component to access data from: Paradox and dBase tables (using local SQL - subset of ANSI standard SQL), Databases on the Local InterBase Server, and Databases on remote database servers.
Delphi also supports heterogeneous queries against more than one server or table type (for example, data from an Oracle table and a Paradox table).

TQuery has a property called *SQL*, which is used to store the SQL statement. TQuery encapsulates one or more SQL statements, executes them and provides methods by which we can manipulate the results. Queries can be divided into two categories: those that produce result sets (such as a *SELECT* statement), and those that don't (such as an *UPDATE* or *INSERT* statement). Use TQuery.Open to execute a query that produces a result set; use TQuery.ExecSQL to execute queries that do not produce result sets.

The SQL statements can be either *static* or *dynamic*, that is, they can be set at design time or include parameters (*TQuery.Params*) that vary at run time. Using

parameterized queries is very flexible, because you can change a user's view of and access to data on the fly at run time.

All executable SQL statements must be prepared before they can be executed. The result of preparation is the executable or operational form of the statement. The method of preparing an SQL statement and the persistence of its operational form distinguish static SQL from dynamic SQL. At design time a query is prepared and executed automatically when you set the query component's Active property to True. At run time, a query is prepared with a call to Prepare, and executed when the application calls the component's Open or ExecSQL methods.

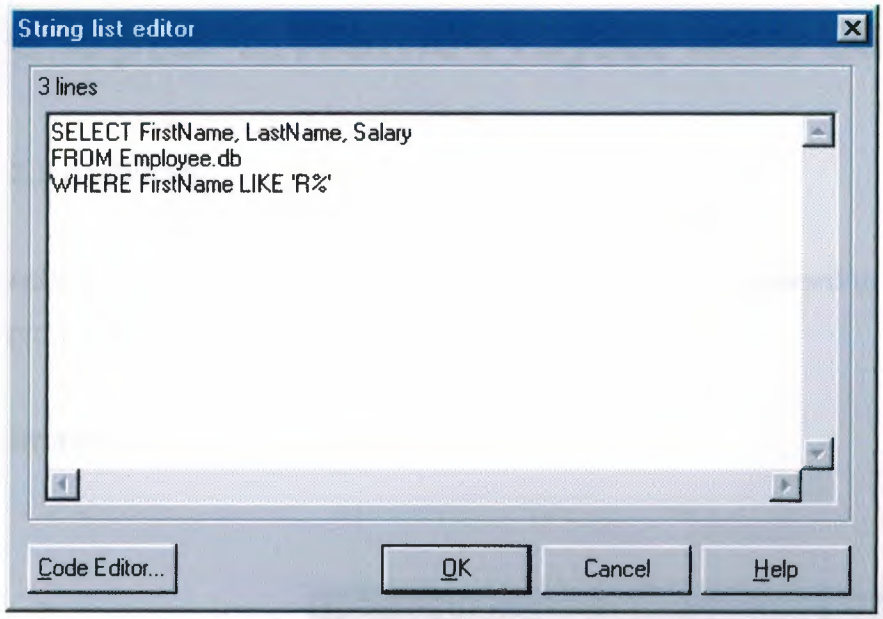
A TQuery can return two kinds of result sets: "*live*" as with TTable component (users can edit data with data controls, and when a call to Post occurs changes are send to database), "*read only*" for displaying purposes only. To request a live result set, set a query component's RequestLive property to True, and be aware that SQL statement must meet some specific requirements (no ORDER BY, SUM, AVG, etc.)

A query behaves in many ways very much like a table filter, and in some way a query is even more powerful than a filter because it lets you access: more than one table at a time ("join" in SQL), a specified subset of columns (rows and) in its underlying table(s), rather than always returning all columns (and rows).

1.4.4 Simple example of SQL.

We could now see some SQL in action. Even if we can use Database Form Wizard to create some "SQL example", let's do it step by step:

1. Place a TQuery, TDataSource, TDBGrid, TEdit, and a TButton component on the main form.
2. Set TDataSource component's DataSet property to Query1.
3. Set TDBGrid component's DataSource property to DataSource1.
4. Set TQuery component's DatabaseName property to DBDEMOS.
5. Double-click on SQL property of a TQuery to assign the SQL statement to it.



6. To make the grid display data at design time, change TQuery component's Active property to True.

7. Now assign the following code to the OnClick event of the Button1.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Query1.Close;{close the query}
//assign new SQL expression
Query1.SQL.Clear;
Query1.SQL.Add ('Select EmpNo, FirstName, LastName');
Query1.SQL.Add ('FROM Employee.db');
Query1.SQL.Add ('WHERE Salary > ' + Edit1.Text);
Query1.Open; {open query + display data}

end;
```

8. Run your application. When you click on Button (with some valid currency value in it), grid will display EmpNo, FirstName and LastName fields with records where Salary is greater than specified currency value.

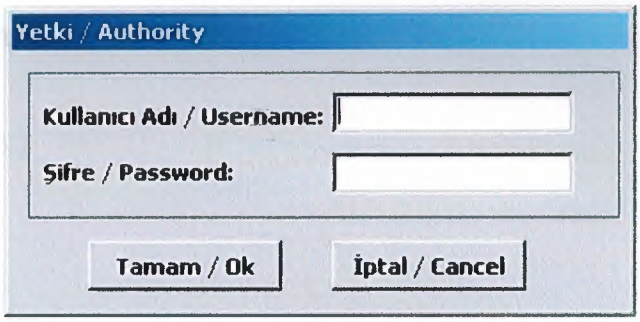
In this example we created simple static SQL statement with live result set (we haven changed any of displayed records) just for displaying purposes.

Chapter 2.Traffic Project with Codes

2.1 Authority

Asking to user for enter correct username with correct password to get access to use the program.

Image:



Codes:

```
unit Unitaccess;

interface

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

type
  TFormaccess = class(TForm)
    GroupBox1: TGroupBox;
    Label1: TLabel;
    Label2: TLabel;
    Edit2: TEdit;
    Edit1: TEdit;
```

```
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    Query1: TQuery;
    procedure FormActivate(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
   procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure FormCreate(Sender: TObject);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Formaccess: TFormaccess;
  kullanici,adi,seviye:string;
implementation

uses UnitMain;

{$R *.dfm}

procedure TFormaccess.FormActivate(Sender: TObject);
begin
 kullanici:='';
 Edit1.Text:='';
 Edit2.Text:='';
 Edit1.SetFocus;
end;

procedure TFormaccess.BitBtn2Click(Sender: TObject);
begin
 adi:='Cancel';
```

```
 edit1.Text:='';
 edit2.Text:='';
 close;
 Application.Terminate;
end;

procedure TFormaccess.BitBtn1Click(Sender: TObject);
begin
 adi:='OK';
  with Query1 do
   begin
    close;
    SQL.Clear;
    SQL.Add('select * from accessT where adi=:adi and sifre=:sifre');
    ParamByName('adi').Value:=Trim(Edit1.Text);
    ParamByName('sifre').Value:=Trim(Edit2.Text);
    Open;
   end;
   if not Query1.IsEmpty then
    begin
     kullanici:=Edit1.Text;
     seviye:=Query1.FieldValues['seviye'];
     close;
    end
   else
    begin
     kullanici:='';
    ShowMessage('KULLANICI ADI VEYA ŞİFRE YANLIŞ /  USERNAME OR PASSWORD IS
WRONG');
     exit;
    end;
end;



procedure TFormaccess.FormKeyDown(Sender: TObject; var Key: Word;
```

```
  Shift: TShiftState);
begin
if ((ssAlt in Shift) and (Key = VK_F4)) then
 Key := 0;
end;

procedure TFormaccess.FormCreate(Sender: TObject);
begin
 KeyPreview := true;
end;

procedure TFormaccess.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
if ord(key)=13 then
 BitBtn1Click(self);
end;

end.
```

2.2 Main Menu

Main Menu is allows you to choose your language to execute the program as you want to use in your language.

Image:

Codes:

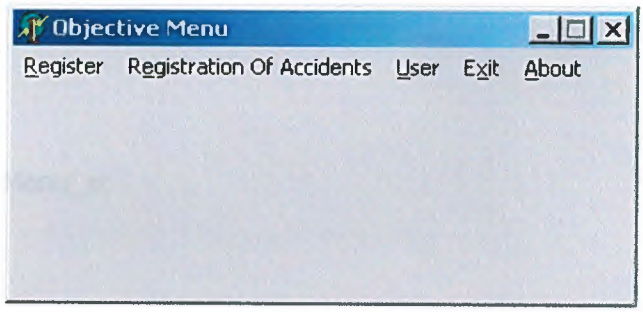
```
unit UnitMain;

interface

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

type
  TFormMain = class(TForm)
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    SpeedButton1: TSpeedButton;
    GroupBox3: TGroupBox;
    SpeedButton2: TSpeedButton;
    procedure SpeedButton1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
```

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

var
  FormMain: TFormMain;

implementation

uses UnitMenu, Unitaccess, UnitMenu_e;

{$R *.dfm}

procedure TFormMain.SpeedButton1Click(Sender: TObject);
begin
 formmenu.showmodal;
end;

procedure TFormMain.FormActivate(Sender: TObject);
begin
  Formaccess.ShowModal;
  while (kullanici='') and (adi='OK') do
   Formaccess.ShowModal;
end;

procedure TFormMain.SpeedButton2Click(Sender: TObject);
begin
 formmenu_e.showmodal;
end;

end.
```

2.3 Objective Menu:

Here, user can reach any form to do wanted thinks.

Image:



Codes:

```
unit UnitMenu_e;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, DB, DBTables;

type
  TFormMenu_e = class(TForm)
    MainMenu1: TMainMenu;
    antm1: TMenuItem;
    KazaGirilmeleri1: TMenuItem;
    k1: TMenuItem;
    Kullanc1: TMenuItem;
    Quser: TQuery;
    About1: TMenuItem;
    procedure KazaGirilmeleri1Click(Sender: TObject);
    procedure k1Click(Sender: TObject);
    procedure antm1Click(Sender: TObject);
```

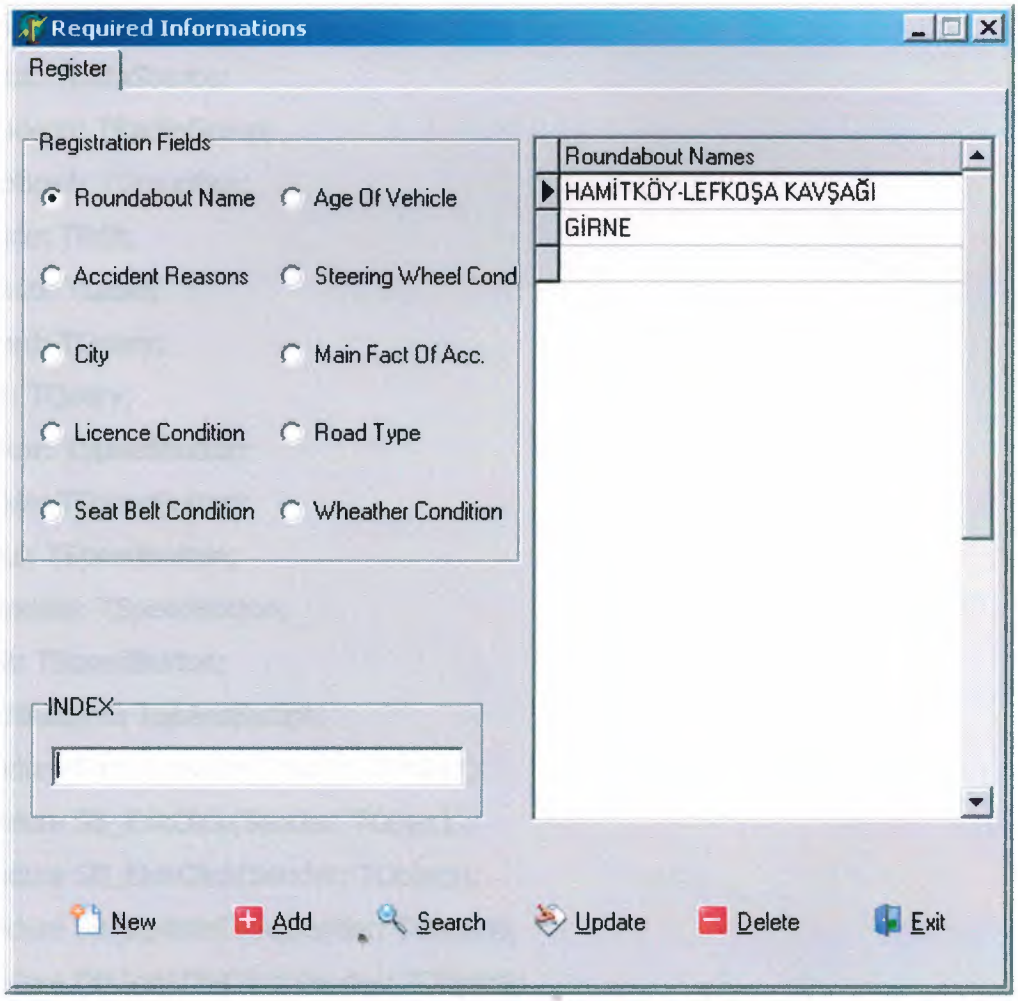
```
    procedure Kullanc1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure About1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FormMenu_e: TFormMenu_e;

implementation

uses Unit_KazaKayit, Eklenti, Unituser, Unitaccess, Eklenti_e, Unituser_e,
    Unit_KazaKayit_e ,unitHakk;

{$R *.dfm}

procedure TFormMenu_e.KazaGirilmeleri1Click(Sender: TObject);
begin
 Form_KazaKayit_e.showmodal;
end;

procedure TFormMenu_e.k1Click(Sender: TObject);
begin
 close;
end;

procedure TFormMenu_e.antm1Click(Sender: TObject);
begin
 FormEklenti_e.showmodal;
end;

procedure TFormMenu_e.Kullanc1Click(Sender: TObject);
begin
```

```
  Formuser_e.showmodal;
end;

procedure TFormMenu_e.FormActivate(Sender: TObject);
begin
 if seviye='Admin' then
    Kullanc1.Enabled:=true
 else
    Kullanc1.Enabled:=false;
end;

procedure TFormMenu_e.About1Click(Sender: TObject);
begin
formhakk.showmodal;

end;

end.
```

2.4 Register

Most important information are included here, the additional registrations will be inserted in this form

Image:



Codes:

```
unit Eklenti_e;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, ExtCtrls, Mask, DBCtrls, Grids, DBGrids, DB,
```

```
  DBTables,TabNotBk, Buttons ,
  ComCtrls,Menus, imgList, Spin, ADODB;

type
  TFormEklenti_e = class(TForm)
    PC_Sistem: TPageControl;
    TS_Ekle: TTabSheet;
    DBGrid1: TDBGrid;
    DS_List: TDataSource;
    RG_Eklenti: TRadioGroup;
    GroupBox4: TGroupBox;
    EB_Ekle: TEdit;
    L_sirano: TLabel;
    Qrecord: TQuery;
    Q_list: TQuery;
    SB_New: TSpeedButton;
    SB_Ekle: TSpeedButton;
    SB_Bul: TSpeedButton;
    SB_Update: TSpeedButton;
    SB_Sil: TSpeedButton;
    SpeedButton5: TSpeedButton;
    procedure FormActivate(Sender: TObject);
    procedure SB_ExitClick(Sender: TObject);
    procedure SB_EkleClick(Sender: TObject);
    procedure SB_UpdateClick(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
    procedure RG_EklentiClick(Sender: TObject);
    procedure SB_NewClick(Sender: TObject);
    procedure EB_EkleKeyPress(Sender: TObject; var Key: Char);
    procedure SB_BulClick(Sender: TObject);
    procedure SB_SilClick(Sender: TObject);
    procedure SpeedButton5Click(Sender: TObject);
    procedure DBGrid1CellClick(Column: TColumn);
//    procedure Clear_Screen;
  private
    { Private declarations }
```

```
  public
    { Public declarations }
  end;

var
  FormEklenti_e: TFormEklenti_e;
  table_name,display_name,field_name:string;

implementation

uses Unitaccess;
{$R *.DFM}

procedure TFormEklenti_e.FormActivate(Sender: TObject);
begin
 L_SiraNo.Caption:='';
 RG_Eklenti.ItemIndex:=0;
 EB_Ekle.SetFocus;
 RG_EklentiClick(Self);

 if seviye='Admin' then
 begin
  SB_New.Enabled:=true;
  SB_Ekle.Enabled:=true;
  SB_Bul.Enabled:=true;
  SB_Update.Enabled:=true;
  SB_Sil.Enabled:=true;
 end
 else if seviye='1.Seviye' then
 begin
  SB_New.Enabled:=True;
  SB_Ekle.Enabled:=False;
  SB_Bul.Enabled:=true;
  SB_Update.Enabled:=False;
  SB_Sil.Enabled:=False;
 end
```

```
 else if seviye='2.Seviye' then
 begin
  SB_New.Enabled:=True;
  SB_Ekle.Enabled:=True;
  SB_Bul.Enabled:=true;
  SB_Update.Enabled:=False;
  SB_Sil.Enabled:=False;
 end
 else if seviye='3.Seviye' then
 begin
  SB_New.Enabled:=true;
  SB_Ekle.Enabled:=true;
  SB_Bul.Enabled:=true;
  SB_Update.Enabled:=true;
  SB_Sil.Enabled:=true;
 end;

end;

procedure TFormEklenti_e.SB_ExitClick(Sender: TObject);
begin
 Close;
end;


procedure TFormEklenti_e.SB_EkleClick(Sender: TObject);
begin
 if trim(EB_Ekle.Text)='' Then
  begin
   showmessage('INDEX can not be empty...!');
   exit;
  End;
 Qrecord.Close;
 Qrecord.SQL.Text := 'iNSERT iNTO '+table_name+
                ' ('+field_name+')'+'VALUES ('' + EB_Ekle.Text + '')';
 Qrecord.ExecSQL;
```

```
 RG_EklentiClick(Self);
end;

procedure TFormEklenti_e.SB_UpdateClick(Sender: TObject);
var
 tus:integer;
begin
 if trim(EB_Ekle.Text)='' Then
  begin
   SHOWMESSAGE('Any Record Should be chosen..!');
   exit;
  End;
 tus:=Application.MessageBox('Are you sure want to update selected record?',
                             'Warning',MB_YESNOCANCEL+MB_DEFBUTTON1);
 if tus = 6 then
  begin
   with Qrecord do
    begin
     SQL.Text := 'UPDATE '+table_name+' SET ';
     SQL.Add(field_name+ ' = ''' + EB_Ekle.Text + '''');
     SQL.Add('WHERE ' + Q_list.Fields[0].FullName + ' = ' + intToStr(Q_list.Fields[0].Value));
     ExecSQL;
    end;
  end;
 RG_EklentiClick(Self);
end;

procedure TFormEklenti_e.DBGrid1DblClick(Sender: TObject);
begin
 L_SiraNo.Caption:=IntToStr(Q_list.Fields[0].Value);
 EB_Ekle.Text := Q_list.Fields[1].Value;
end;

procedure TFormEklenti_e.RG_EklentiClick(Sender: TObject);
begin
 table_name:='';
```

```
field_name:='';
display_name:='';
L_sirano.Caption:='';
EB_Ekle.Text:='';
if RG_Eklenti.ItemIndex=0 then
 begin
  table_name:='kavsakT';
  field_name:='kavsak_adi';
  display_name:='Roundabout Names';
 end
else if RG_Eklenti.ItemIndex=1 then
 begin
  table_name:='kazaT';
  field_name:='kaza_adi';
  display_name:='Accident Reasons';
 end
else if RG_Eklenti.ItemIndex=2 then
 begin
  table_name:='SehirT';
  field_name:='Sehir_adi';
  display_name:='City';
 end
else if RG_Eklenti.ItemIndex=3 then
 begin
  table_name:='EhliyetT';
  field_name:='Ehliyet_drm';
  display_name:='Licence Condition';
 end
else if RG_Eklenti.ItemIndex=4 then
 begin
  table_name:='Emn_KemerT';
  field_name:='Emn_kmr_drm';
  display_name:='Seat Belt Condition';
 end
else if RG_Eklenti.ItemIndex=5 then
 begin
```

```
  table_name:='imal_TarihT';
  field_name:='imal_Tarih';
  display_name:='Age Of Vehicle';
 end
else if RG_Eklenti.ItemIndex=6 then
 begin
  table_name:='DireksiyonT';
  field_name:='Direksiyon_dur';
  display_name:='Steering Wheel Condition';
 end
else if RG_Eklenti.ItemIndex=7 then
 begin
  table_name:='kazaFakT';
  field_name:='Kaza_esas_Fak';
  display_name:='Main Fact Of Accident';
 end
else if RG_Eklenti.ItemIndex=8 then
 begin
  table_name:='YoL_DrmT';
  field_name:='Yol_drm';
  display_name:='Road Type';
 end
else if RG_Eklenti.ItemIndex=9 then
 begin
  table_name:='HavaDrmT';
  field_name:='Hava_Durum';
  display_name:='Wheather Condition';
 end;
with Q_list do
 begin
  Close;
  SQL.Text := 'SELECT * FROM '+table_name;
  SQL.ADD(' ORDER BY '+field_name+' desc');
  Open;
 end;
  Q_list.Fields[0].Visible:=False;
```

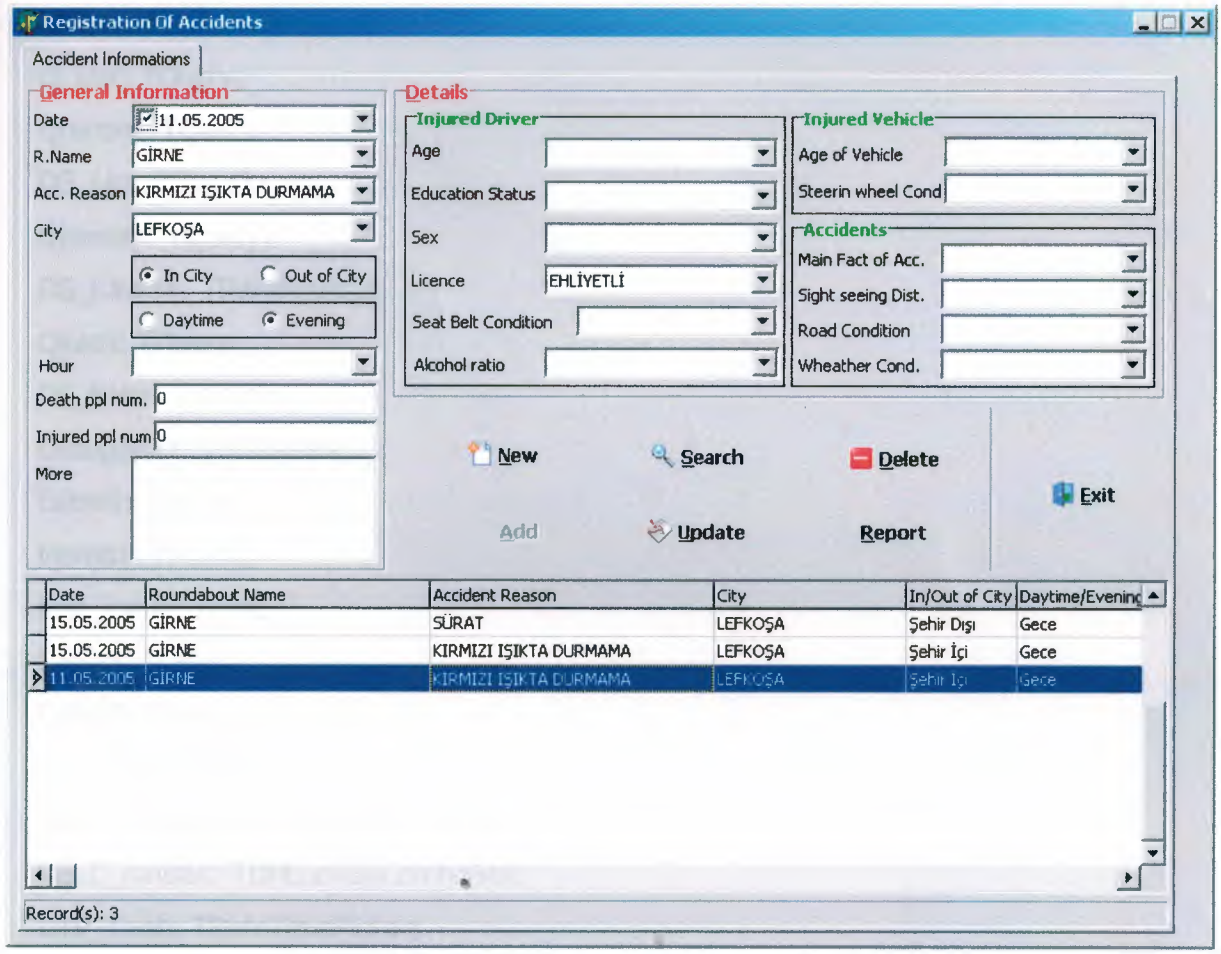
```
  Q_list.FieldByName(field_name).DisplayLabel := display_name;
  EB_Ekle.SetFocus;
end;

procedure TFormEklenti_e.SB_NewClick(Sender: TObject);
begin
 RG_EklentiClick(Self);
end;

procedure TFormEklenti_e.EB_EkleKeyPress(Sender: TObject; var Key: Char);
begin
 if key in ['İ'] then key :=key;
 if key in ['i'] then key :='İ';
 if key in ['ı'] then key :='I';
end;

procedure TFormEklenti_e.SB_BulClick(Sender: TObject);
begin
 if trim(EB_Ekle.Text)='' then
  begin
   showmessage('INDEX can not be empty...!');
   exit;
  end;
 with Q_list do
  begin
   Close;
   SQL.Text := 'SELECT * FROM '+table_name;
   if EB_Ekle.Text<>'' then
    SQL.ADD(' WHERE '+field_name+' Like '''+EB_Ekle.Text+ '%''');
   SQL.ADD(' ORDER BY '+field_name);
   Open;
  end;
   Q_list.Fields[0].Visible:=False;
   Q_list.FieldByName(field_name).DisplayLabel := display_name;
end;
```

```
procedure TFormEklenti_e.SB_SilClick(Sender: TObject);
var
 tus:integer;
begin
 if trim(EB_Ekle.Text)='' then
  begin
   SHOWMESSAGE('Any Record Should be chosen..!');
   exit;
  End;
 tus:=Application.MessageBox('Are you sure want to delete selected record?',
                        'Warning',MB_YESNOCANCEL+MB_DEFBUTTON1);
 if tus = 6 then
  begin
   with Qrecord do
    begin
     SQL.Text := 'delete from '+table_name;
     SQL.Add('WHERE ' + Q_list.Fields[0].FullName + ' = ' + intToStr(Q_list.Fields[0].Value));
     ExecSQL;
    end;
  end;
  RG_EklentiClick(self);
end;

procedure TFormEklenti_e.SpeedButton5Click(Sender: TObject);
begin
 close;
end;

procedure TFormEklenti_e.DBGrid1CellClick(Column: TColumn);
begin
 if Q_list.Fields[0].Value=0 then
  exit;
 L_SiraNo.Caption:=IntToStr(Q_list.Fields[0].Value);
 EB_Ekle.Text := Q_list.Fields[1].Value;
end;
end.
```

2.5 Registrations of Accidents

All information of data will be inserted here and these information of functions (add, update, delete, search) will be used from here.

Image:



Codes:

unit Unit_KazaKayit;

interface

uses

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

type
  TForm_KazaKayit = class(TForm)
    PC_Sistem: TPageControl;
    TS_Ekle: TTabSheet;
    DBGrid1: TDBGrid;
    Q_List: TQuery;
    Qrecord: TQuery;
    DS_List: TDataSource;
    Qkavsak: TQuery;
    DS_Kavsak: TDataSource;
    Qkaza: TQuery;
    DS_Kaza: TDataSource;
    GroupBox1: TGroupBox;
    Label6: TLabel;
    Memo1: TMemo;
    Edit_yarali: TEdit;
    Label5: TLabel;
    Label4: TLabel;
    Edit_olu: TEdit;
    DBLC_Kaza: TDBLookupComboBox;
    DBLC_Kavsak: TDBLookupComboBox;
    DTP_Tarih: TDateTimePicker;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    L_sirano: TLabel;
    DBLC_Sehir: TDBLookupComboBox;
    Label7: TLabel;
    Label8: TLabel;
    QSehir: TQuery;
    DSSehir: TDataSource;
    CB_Saat: TComboBox;
```

GroupBox2: TGroupBox;
RB_Sici: TRadioButton;
RB_Sdisi: TRadioButton;
GroupBox3: TGroupBox;
RB_gunduz: TRadioButton;
RB_Gece: TRadioButton;
GroupBox5: TGroupBox;
GroupBox6: TGroupBox;
Label10: TLabel;
Label11: TLabel;
CB_yas: TComboBox;
CB_egitim: TComboBox;
Label12: TLabel;
CB_Cinsiyet: TComboBox;
Label13: TLabel;
Label14: TLabel;
DBLC_Ehliyet: TDBLookupComboBox;
DBLC_Kem_durum: TDBLookupComboBox;
CB_Alkol: TComboBox;
Label15: TLabel;
GroupBox7: TGroupBox;
Label16: TLabel;
Label17: TLabel;
DBLC_arac_imal: TDBLookupComboBox;
DBLC_Arac_Dir_dur: TDBLookupComboBox;
GroupBox8: TGroupBox;
Label9: TLabel;
DBLC_esas_Fak: TDBLookupComboBox;
Label18: TLabel;
Label19: TLabel;
CB_Gmesafe: TComboBox;
DBLC_yol_dur: TDBLookupComboBox;
Label20: TLabel;
DBLC_hava_dur: TDBLookupComboBox;
SB_New: TSpeedButton;
SB_Ekle: TSpeedButton;

```
SB_Bul: TSpeedButton;
SB_Update: TSpeedButton;
SB_Sil: TSpeedButton;
SpeedButton5: TSpeedButton;
QEhliyet: TQuery;
DSEhliyet: TDataSource;
DSEmn_kmr_drm: TDataSource;
QEmn_Kem_Drm: TQuery;
QArac_imal: TQuery;
DSArac_imal: TDataSource;
DSDir_dur: TDataSource;
QDir_Drm: TQuery;
Qkaza_esas_faktor: TQuery;
DSKaza_esas_fak: TDataSource;
DSyol_dur: TDataSource;
Qyol_dur: TQuery;
QHava_dur: TQuery;
DSHava_Dur: TDataSource;
Q_ListId: TIntegerField;
Q_ListKavsak_id: TIntegerField;
Q_ListKazaneden_id: TIntegerField;
Q_ListSehir_id: TIntegerField;
Q_ListSehir_type: TStringField;
Q_ListGece_gunduz: TStringField;
Q_ListSaat: TStringField;
Q_ListO_sayi: TIntegerField;
Q_ListY_sayi: TIntegerField;
Q_ListAciklama: TStringField;
Q_ListYasi: TStringField;
Q_ListEgitim: TStringField;
Q_ListCinsiyet: TStringField;
Q_ListEhliyet_id: TIntegerField;
Q_ListEkemer_id: TIntegerField;
Q_ListAlkoloran: TStringField;
Q_ListImal_tarih_id: TIntegerField;
Q_ListDireksiyon_id: TIntegerField;
```

Q_ListFaktor_id: TIntegerField;
Q_ListGorus_mesafe: TStringField;
Q_ListYdurumu_id: TIntegerField;
Q_ListHdurumu_id: TIntegerField;
Q_ListTarih: TDateField;
Q_ListKavsak_id_1: TIntegerField;
Q_ListKavsak_adi: TStringField;
Q_ListKaza_id: TIntegerField;
Q_ListKaza_adi: TStringField;
Q_ListSehir_id_1: TIntegerField;
Q_ListSehir_adi: TStringField;
Q_ListEhliyet_id_1: TIntegerField;
Q_ListEhliyet_drm: TStringField;
Q_ListEmn_kmr_dur_id: TIntegerField;
Q_ListEmn_kmr_drm: TStringField;
Q_ListImal_id: TIntegerField;
Q_ListImal_Tarih: TStringField;
Q_ListDireksiyon_id_1: TIntegerField;
Q_ListDireksiyon_dur: TStringField;
Q_ListKaza_faktor_id: TIntegerField;
Q_ListKaza_esas_fak: TStringField;
Q_ListYol_dur_id: TIntegerField;
Q_ListYol_drm: TStringField;
Q_ListHava_dur_id: TIntegerField;
Q_ListHava_Durum: TStringField;
StatusBar1: TStatusBar;
QuickRep1: TQuickRep;
QRBand1: TQRBand;
QRLabel1: TQRLabel;
QRSysData1: TQRSysData;
QRLabel2: TQRLabel;
QRBand2: TQRBand;
QRLabel3: TQRLabel;
QRLabel4: TQRLabel;
QRLabel5: TQRLabel;
QRSubDetail1: TQRSubDetail;

```
Qrapor: TQuery;
IntegerField1: TIntegerField;
DateField1: TDateField;
StringField1: TStringField;
StringField2: TStringField;
StringField3: TStringField;
IntegerField2: TIntegerField;
IntegerField3: TIntegerField;
IntegerField4: TIntegerField;
StringField4: TStringField;
StringField5: TStringField;
StringField6: TStringField;
IntegerField5: TIntegerField;
IntegerField6: TIntegerField;
StringField7: TStringField;
StringField8: TStringField;
StringField9: TStringField;
StringField10: TStringField;
IntegerField7: TIntegerField;
IntegerField8: TIntegerField;
StringField11: TStringField;
StringField12: TStringField;
StringField13: TStringField;
IntegerField9: TIntegerField;
StringField14: TStringField;
IntegerField10: TIntegerField;
StringField15: TStringField;
IntegerField11: TIntegerField;
StringField16: TStringField;
StringField17: TStringField;
IntegerField12: TIntegerField;
IntegerField13: TIntegerField;
IntegerField14: TIntegerField;
IntegerField15: TIntegerField;
IntegerField16: TIntegerField;
IntegerField17: TIntegerField;
```

```
    IntegerField18: TIntegerField;
    IntegerField19: TIntegerField;
    IntegerField20: TIntegerField;
    IntegerField21: TIntegerField;
    IntegerField22: TIntegerField;
    StringField18: TStringField;
    IntegerField23: TIntegerField;
    StringField19: TStringField;
    DS_rapor: TDataSource;
    QRDBText1: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    SpeedButton1: TSpeedButton;
    QRLabel6: TQRLabel;
    Panel1: TPanel;
    procedure SB_NewClick(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure SpeedButton5Click(Sender: TObject);
    procedure Edit_oluKeyPress(Sender: TObject; var Key: Char);
    procedure Edit_yaraliKeyPress(Sender: TObject; var Key: Char);
    procedure SB_EkleClick(Sender: TObject);
    procedure DBGrid1CellClick(Column: TColumn);
    procedure SB_UpdateClick(Sender: TObject);
    procedure SB_SilClick(Sender: TObject);
    procedure SB_BulClick(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form_KazaKayit: TForm_KazaKayit;

implementation
```

```
uses UnitTarih,unitaccess;

{$R *.dfm}

procedure TForm_KazaKayit.SB_NewClick(Sender: TObject);
begin
 L_sirano.Caption:='';
 //
 DBLC_Kaza.Enabled:=true;
 DBLC_Kavsak.Enabled:=true;
 DBLC_Sehir.Enabled:=true;
 //
 if seviye='Admin' then
 begin
  SB_Update.Enabled:=false;
  SB_Sil.Enabled:=false;
  SB_Ekle.Enabled:=True;
  SB_New.Enabled:=true;
  SB_Bul.Enabled:=true;
  SpeedButton1.Enabled:=true;
 end
 else if seviye='1.Seviye' then
 begin
  SB_New.Enabled:=False;
  SB_Ekle.Enabled:=False;
  SB_Bul.Enabled:=true;
  SB_Update.Enabled:=False;
  SB_Sil.Enabled:=False;
  SpeedButton1.Enabled:=true;
 end
 else if seviye='2.Seviye' then
 begin
  SB_New.Enabled:=True;
  SB_Ekle.Enabled:=True;
  SB_Bul.Enabled:=true;
```

```
 SB_Update.Enabled:=False;
 SB_Sil.Enabled:=False;
 SpeedButton1.Enabled:=true;
end
else if seviye='3.Seviye' then
begin
 SB_Update.Enabled:=false;
 SB_Sil.Enabled:=false;
 SB_Ekle.Enabled:=True;
 SB_New.Enabled:=true;
 SB_Bul.Enabled:=true;
 SpeedButton1.Enabled:=true;
end;

 //
DTP_Tarih.DateTime:=Date;
 //
DBLC_Kavsak.KeyValue:=0;
DBLC_Kaza.KeyValue:=0;
DBLC_Sehir.KeyValue:=0;
DBLC_Ehliyet.KeyValue:=0;
DBLC_Kem_durum.KeyValue:=0;
DBLC_arac_imal.KeyValue:=0;
DBLC_Arac_Dir_dur.KeyValue:=0;
DBLC_esas_Fak.KeyValue:=0;
DBLC_yol_dur.KeyValue:=0;
DBLC_hava_dur.KeyValue:=0;
 //
Edit_olu.Text:='0';
Edit_yarali.Text:='0';
 //
CB_yas.Text:='';
CB_egitim.Text:='';
CB_Cinsiyet.Text:='';
CB_Alkol.Text:='';
CB_Gmesafe.Text:='';
```

```
CB_Saat.Text:='';
 //
RB_Sici.Checked:=False;
RB_Sdisi.Checked:=False;
RB_Gece.Checked:=False;
RB_gunduz.Checked:=False;
 //
Memo1.Clear;
 //
DTP_Tarih.SetFocus;
 //
Qkavsak.Close;
Qkavsak.Open;
Qkaza.Close;
Qkaza.Open;
 //
QSehir.Close;
QSehir.Open;
 //
QEhliyet.Close;
QEhliyet.Open;
 //
QEmn_Kem_Drm.Close;
QEmn_Kem_Drm.Open;
 //
QArac_imal.Close;
QArac_imal.Open;
 //
QDir_Drm.Close;
QDir_Drm.Open;
 //
Qkaza_esas_faktor.Close;
Qkaza_esas_faktor.Open;
 //
Qyol_dur.Close;
Qyol_dur.Open;
```

```
 //
QHava_dur.Close;
QHava_dur.Open;
 //
with Q_List do
 begin
  close;
  SQL.Clear;
  SQL.Add('select * from kazakayitT a, kavsakT b,kazaT c, sehirT d, ehliyetT e,');
  SQL.Add('emn_kemerT f, imal_tarihT g, direksiyonT h, kazaFakT k,');
  SQL.Add('yol_drmT l, havadrmT m');
  SQL.Add('where a.kavsak_id=b.Kavsak_id and');
  SQL.Add(' a.kazaneden_id=c.kaza_id and');
  SQL.Add(' a.sehir_id=d.sehir_id and');
  SQL.Add(' a.ekemer_id=f.emn_kmr_dur_id and');
  SQL.Add(' a.ehliyet_id=e.ehliyet_id and');
  SQL.Add(' a.imal_tarih_id=g.imal_id and');
  SQL.Add(' a.direksiyon_id=h.direksiyon_id and');
  SQL.Add(' a.faktor_id=k.kaza_faktor_id and');
  SQL.Add(' a.ydurumu_id=l.yol_dur_id and');
  SQL.Add(' a.hdurumu_id=m.hava_dur_id ');
  sql.add('order by id desc');
  open;
 end;
end;

procedure TForm_KazaKayit.FormActivate(Sender: TObject);
begin
 if Q_List.RecordCount = 0 then
  StatusBar1.SimpleText:='No Record yet.'
 else
  StatusBar1.SimpleText:='Record(s): '+inttostr(Q_List.RecordCount);
 SB_NewClick(self);
 //
 DBLC_Kaza.Enabled:=false;
 DBLC_Kavsak.Enabled:=false;
```

```
DBLC_Sehir.Enabled:=false;
//
if seviye='Admin' then
begin
 SB_Update.Enabled:=false;
 SB_Sil.Enabled:=false;
 SB_Ekle.Enabled:=True;
 SB_New.Enabled:=true;
 SB_Bul.Enabled:=true;
 SpeedButton1.Enabled:=true;
end
else if seviye='1.Seviye' then
begin
 SB_New.Enabled:=False;
 SB_Ekle.Enabled:=False;
 SB_Bul.Enabled:=true;
 SB_Update.Enabled:=False;
 SB_Sil.Enabled:=False;
 SpeedButton1.Enabled:=true;
end
else if seviye='2.Seviye' then
begin
 SB_New.Enabled:=True;
 SB_Ekle.Enabled:=True;
 SB_Bul.Enabled:=true;
 SB_Update.Enabled:=False;
 SB_Sil.Enabled:=False;
 SpeedButton1.Enabled:=true;
end
else if seviye='3.Seviye' then
begin
 SB_Update.Enabled:=false;
 SB_Sil.Enabled:=false;
 SB_Ekle.Enabled:=True;
 SB_New.Enabled:=true;
 SB_Bul.Enabled:=true;
```

```
  SpeedButton1.Enabled:=true;
 end;


end;

procedure TForm_KazaKayit.SpeedButton5Click(Sender: TObject);
begin
 close;
end;

procedure TForm_KazaKayit.Edit_oluKeyPress(Sender: TObject; var Key: Char);
begin
if ( StrScan('0123456789',Key) <> nil ) or
    ( Key = Char(VK_BACK) ) then
  begin

  end
  else
    Key := #0;
end;

procedure TForm_KazaKayit.Edit_yaraliKeyPress(Sender: TObject;
  var Key: Char);
begin
if ( StrScan('0123456789',Key) <> nil ) or
    ( Key = Char(VK_BACK) ) then
  begin

  end
  else
    Key := #0;
end;

procedure TForm_KazaKayit.SB_EkleClick(Sender: TObject);
var
```

```
kavsak,kaza,sehir,Ehliyet,Kem_durum,arac_imal,esas_Fak,Arac_Dir_dur,yol_dur,hava_dur:inte
ger;
begin
 if ((DBLC_Kaza.Text = '') or (DBLC_Kavsak.Text = '') or (DBLC_Sehir.Text = '')) then
  ShowMessage('Roundabout Name , Accident Reason and City can not be empty')
 else
  begin
   if DBLC_Kaza.Text='' then
    kaza:=0
   else kaza:=DBLC_Kaza.KeyValue;
   if DBLC_Kavsak.Text='' then
    kavsak:=0
   else kavsak:=DBLC_Kavsak.KeyValue;
   if DBLC_Sehir.Text='' then
    sehir:=0
   else sehir:=DBLC_Sehir.KeyValue;
   if DBLC_Ehliyet.Text='' then
    Ehliyet:=0
   else Ehliyet:=DBLC_Ehliyet.KeyValue;
   if DBLC_Kem_durum.Text='' then
    Kem_durum:=0
   else Kem_durum:=DBLC_Kem_durum.KeyValue;
   if DBLC_arac_imal.Text='' then
    arac_imal:=0
   else arac_imal:=DBLC_arac_imal.KeyValue;
   if DBLC_Arac_Dir_dur.Text='' then
    Arac_Dir_dur:=0
   else Arac_Dir_dur:=DBLC_Arac_Dir_dur.KeyValue;
   if DBLC_esas_Fak.Text='' then
    esas_Fak:=0
   else esas_Fak:=DBLC_esas_Fak.KeyValue;
   if DBLC_yol_dur.Text='' then
    yol_dur:=0
   else yol_dur:=DBLC_yol_dur.KeyValue;
   if DBLC_hava_dur.Text='' then
```

```
 hava_dur:=0
else hava_dur:=DBLC_hava_dur.KeyValue;
with Qrecord do
 begin
  close;
  SQL.Clear;
  SQL.Add('insert into KazaKayitT');
  SQL.Add('(kavsak_id,kazaneden_id,sehir_id,sehir_type,gece_gunduz,');
  SQL.Add('saat,o_sayi,y_sayi,aciklama,yasi,egitim,cinsiyet,ehliyet_id,');
  SQL.Add('ekemer_id,alkoloran,imal_tarih_id,direksiyon_id,faktor_id,');
  SQL.Add('gorus_mesafe,ydurumu_id,hdurumu_id,tarih)');
  SQL.Add('values(:xkavsak_id,:xkazaneden_id,:xsehir_id,:xsehir_type,:xgece_gunduz,');
  SQL.Add(':xsaat,:xo_sayi,:xy_sayi,:xaciklama,:xyasi,:xegitim,:xcinsiyet,:xehliyet_id,');
  SQL.Add(':xekemer_id,:xalkoloran,:ximal_tarih_id,:xdireksiyon_id,:xfaktor_id,');
  SQL.Add(':xgorus_mesafe,:xydurum_id,:xhdurum_id,:xtarih)');
  ParamByName('xkavsak_id').Value:=kavsak;
  ParamByName('xkazaneden_id').Value:=kaza;
  ParamByName('xsehir_id').Value:=Sehir;
  if RB_Sici.Checked=true then
    ParamByName('xsehir_type').Value:=RB_Sici.Caption
  else
    ParamByName('xsehir_type').Value:=RB_Sdisi.Caption;
  if RB_gunduz.Checked=true then
    ParamByName('xgece_gunduz').Value:=RB_gunduz.Caption
  else
    ParamByName('xgece_gunduz').Value:=RB_Gece.Caption;
  ParamByName('xsaat').Value:=CB_Saat.Text;
  ParamByName('xo_sayi').AsInteger:=StrToInt(Edit_olu.Text);
  ParamByName('xy_sayi').AsInteger:=StrToInt(Edit_yarali.Text);
  ParamByName('xaciklama').AsString:=Memo1.Text;
  ParamByName('xyasi').Value:=CB_yas.Text;
  ParamByName('xegitim').Value:=CB_egitim.Text;
  ParamByName('xcinsiyet').Value:=CB_Cinsiyet.Text;
  ParamByName('xehliyet_id').Value:=Ehliyet;
  ParamByName('xekemer_id').Value:=Kem_durum;
  ParamByName('xalkoloran').Value:=CB_Alkol.Text;
```

```
    ParamByName('ximal_tarih_id').Value:=arac_imal;
    ParamByName('xdireksiyon_id').Value:=Arac_Dir_dur;
    ParamByName('xfaktor_id').Value:=esas_Fak;
    ParamByName('xgorus_mesafe').Value:=CB_Gmesafe.Text;
    ParamByName('xydurum_id').Value:=yol_dur;
    ParamByName('xhdurum_id').Value:=hava_dur;
    ParamByName('xtarih').AsDate:=StrToDate(DateToStr(DTP_Tarih.Date));
    ExecSQL;
   end;
  ShowMessage('Recorded');
  SB_NewClick(self);
  SB_Update.Enabled:=false;
  SB_Sil.Enabled:=false;
  SB_Ekle.Enabled:=False;
  SB_New.Enabled:=true;
  SB_Bul.Enabled:=true;
 end;
//  DBGrid1.FieldCount
 if Q_List.RecordCount = 0 then
  StatusBar1.SimpleText:='No Record yet.'
 else
  StatusBar1.SimpleText:='Record(s): '+inttostr(Q_List.RecordCount);
end;

procedure TForm_KazaKayit.DBGrid1CellClick(Column: TColumn);
begin
 if not Q_List.IsEmpty then
  begin
  DBLC_Kaza.Enabled:=true;
  DBLC_Kavsak.Enabled:=true;
  DBLC_Sehir.Enabled:=true;
  //
  L_sirano.Caption:=Q_ListId.Text;
  DTP_Tarih.Date:=Q_ListTarih.AsDateTime;
  //
  DBLC_Kavsak.KeyValue:=Q_ListKavsak_id.Value;
```

```
 DBLC_Kaza.KeyValue:=Q_ListKazaneden_id.Value;
 DBLC_Sehir.KeyValue:=Q_ListSehir_id.Value;
 //
 if Q_ListSehir_type.Value='Şehir İçi'then
  RB_Sici.Checked:=true
 else RB_Sdisi.Checked:=true;
 if Q_ListSehir_type.Value='Gündüz' then
  RB_gunduz.Checked:=true
 else RB_Gece.Checked:=true;
 //
 CB_Saat.Text:=Q_ListSaat.Text;
 //
 Edit_yarali.Text:=Q_ListY_sayi.Text;
 Edit_olu.Text:=Q_ListO_sayi.Text;
 //
 Memo1.Text:=Q_ListAciklama.Text;
 //
 CB_yas.Text:=Q_ListYasi.Text;
 CB_egitim.Text:=Q_ListEgitim.Text;
 CB_Cinsiyet.Text:=Q_ListCinsiyet.Text;
 CB_Alkol.Text:=Q_ListAlkoloran.Text;
 CB_Gmesafe.Text:=Q_ListGorus_mesafe.Text;
 //
 DBLC_Ehliyet.KeyValue:=Q_ListEhliyet_id.Value;
 DBLC_Kem_durum.KeyValue:=Q_ListEkemer_id.Value;
 DBLC_arac_imal.KeyValue:=Q_ListImal_tarih_id.Value;
 DBLC_Arac_Dir_dur.KeyValue:=Q_ListDireksiyon_id.Value;
 DBLC_esas_Fak.KeyValue:=Q_ListFaktor_id.Value;
 DBLC_yol_dur.KeyValue:=Q_ListYdurumu_id.Value;
 DBLC_hava_dur.KeyValue:=Q_ListHdurumu_id.Value;
 //
 if seviye='Admin' then
begin
 SB_Update.Enabled:=True;
 SB_Sil.Enabled:=True;
end
```

```
 else if seviye='3.Seviye' then
 begin
  SB_Update.Enabled:=True;
  SB_Sil.Enabled:=True;
 end
 else
 begin
  SB_Update.Enabled:=false;
  SB_Sil.Enabled:=false;
 end;
  SB_Ekle.Enabled:=false;
  SB_New.Enabled:=true;
  SB_Bul.Enabled:=true;
  end;
  //SB_Ekle.Enabled:=false;
end;

procedure TForm_KazaKayit.SB_UpdateClick(Sender: TObject);
var tus:integer;

kavsak,kaza,sehir,Ehliyet,Kem_durum,arac_imal,esas_Fak,Arac_Dir_dur,yol_dur,hava_dur:inte
ger;
begin
  if DBLC_Kaza.Text='' then kaza:=0
  else kaza:=DBLC_Kaza.KeyValue;
    //
  if DBLC_Kavsak.Text='' then kavsak:=0
  else kavsak:=DBLC_Kavsak.KeyValue;
    //
  if DBLC_Sehir.Text='' then sehir:=0
  else sehir:=DBLC_Sehir.KeyValue;
    //
  if DBLC_Ehliyet.Text='' then Ehliyet:=0
  else Ehliyet:=DBLC_Ehliyet.KeyValue;
    //
  if DBLC_Kem_durum.Text='' then Kem_durum:=0
```

```
else Kem_durum:=DBLC_Kem_durum.KeyValue;
 //
if DBLC_arac_imal.Text='' then arac_imal:=0
else arac_imal:=DBLC_arac_imal.KeyValue;
 //
if DBLC_Arac_Dir_dur.Text='' then Arac_Dir_dur:=0
else Arac_Dir_dur:=DBLC_Arac_Dir_dur.KeyValue;
 //
if DBLC_esas_Fak.Text='' then esas_Fak:=0
else esas_Fak:=DBLC_esas_Fak.KeyValue;
 //
if DBLC_yol_dur.Text='' then yol_dur:=0
else yol_dur:=DBLC_yol_dur.KeyValue;
 //
if DBLC_hava_dur.Text='' then hava_dur:=0
else hava_dur:=DBLC_hava_dur.KeyValue;
 //
tus:=Application.MessageBox('Are you sure want to update selected record?',
                    'Warning',MB_YESNOCANCEL+MB_DEFBUTTON1);
if tus = 6 then
 begin
  with Qrecord do
   begin
    close;
    SQL.Clear;
    SQL.Add('update KazaKayitT set');

SQL.Add('kavsak_id=:xkavsak_id,kazaneden_id=:xkazaneden_id,sehir_id=:xsehir_id,sehir_typ
e=:xsehir_type,gece_gunduz=:xgece_gunduz,');

SQL.Add('saat=:xsaat,o_sayi=:xo_sayi,y_sayi=:xy_sayi,aciklama=:xaciklama,yasi=:xyasi,egiti
m=:xegitim,cinsiyet=:xcinsiyet,ehliyet_id=:xehliyet_id,');

SQL.Add('ekemer_id=:xekemer_id,alkoloran=:xalkoloran,imal_tarih_id=:ximal_tarih_id,direksi
yon_id=:xdireksiyon_id,faktor_id=:xfaktor_id,');
```

```
SQL.Add('gorus_mesafe=:xgorus_mesafe,ydurumu_id=:xydurum_id,hdurumu_id=:xhdurum_i
d,tarih=:xtarih');
    SQL.Add(' where id=:xid ');
    ParamByName('xkavsak_id').Value:=kavsak;
    ParamByName('xkazaneden_id').Value:=kaza;
    ParamByName('xsehir_id').Value:=Sehir;
    if RB_Sici.Checked=true then
     ParamByName('xsehir_type').Value:=RB_Sici.Caption
    else
     ParamByName('xsehir_type').Value:=RB_Sdisi.Caption;
    if RB_gunduz.Checked=true then
     ParamByName('xgece_gunduz').Value:=RB_gunduz.Caption
    else
     ParamByName('xgece_gunduz').Value:=RB_Gece.Caption;
    ParamByName('xsaat').Value:=CB_Saat.Text;
    ParamByName('xo_sayi').AsInteger:=StrToInt(Edit_olu.Text);
    ParamByName('xy_sayi').AsInteger:=StrToInt(Edit_yarali.Text);
    ParamByName('xaciklama').AsString:=Memo1.Text;
    ParamByName('xyasi').Value:=CB_yas.Text;
    ParamByName('xegitim').Value:=CB_egitim.Text;
    ParamByName('xcinsiyet').Value:=CB_Cinsiyet.Text;
    ParamByName('xehliyet_id').Value:=Ehliyet;
    ParamByName('xekemer_id').Value:=Kem_durum;
    ParamByName('xalkoloran').Value:=CB_Alkol.Text;
    ParamByName('ximal_tarih_id').Value:=arac_imal;
    ParamByName('xdireksiyon_id').Value:=Arac_Dir_dur;
    ParamByName('xfaktor_id').Value:=esas_Fak;
    ParamByName('xgorus_mesafe').Value:=CB_Gmesafe.Text;
    ParamByName('xydurum_id').Value:=yol_dur;
    ParamByName('xhdurum_id').Value:=hava_dur;
    ParamByName('xtarih').AsDate:=StrToDate(DateToStr(DTP_Tarih.Date));
    ParamByName('xid').Value:=StrToInt(L_sirano.Caption);
    ExecSQL;
   end;
  SB_NewClick(self);
```

```
    ShowMessage('Record Updated');
    SB_Update.Enabled:=False;
    SB_Sil.Enabled:=False;
    SB_Ekle.Enabled:=false;
    SB_New.Enabled:=true;
    SB_Bul.Enabled:=true;
   end;
end;

procedure TForm_KazaKayit.SB_SilClick(Sender: TObject);
var tus:integer;
begin
 if L_sirano.Caption = '' then
  exit;
 tus:=Application.MessageBox('Are you sure want to delete selected record?',
                        'Warning',MB_YESNOCANCEL+MB_DEFBUTTON1);
 if tus = 6 then
 begin
  with Qrecord do
   begin
    close;
    SQL.Clear;
    SQL.Add('delete from KazaKayitT where id=:xid');
    ParamByName('xid').Value:=strtoint(L_sirano.Caption);
    ExecSQL;
   end;
  ShowMessage('Record Deleted.');
  SB_NewClick(self);
  if Q_List.RecordCount = 0 then
   StatusBar1.SimpleText:='No Record yet.'
  else
  StatusBar1.SimpleText:='Record(s): '+inttostr(Q_List.RecordCount);
  SB_Update.Enabled:=False;
  SB_Sil.Enabled:=False;
  SB_Ekle.Enabled:=false;
  SB_New.Enabled:=true;
```

```
 SB_Bul.Enabled:=true;
 end;
end;

procedure TForm_KazaKayit.SB_BulClick(Sender: TObject);
begin
 DBLC_Kaza.Enabled:=true;
 DBLC_Kavsak.Enabled:=true;
 DBLC_Sehir.Enabled:=true;
 with Q_List do
  begin
   close;
   SQL.Clear;
   SQL.Add('select * from kazakayitT a, kavsakT b,kazaT c, sehirT d, ehliyetT e,');
   SQL.Add('emn_kemerT f, imal_tarihT g, direksiyonT h, kazaFakT k,');
   SQL.Add('yol_drmT l, havadrmT m');
   SQL.Add('where a.kavsak_id=b.Kavsak_id and');
   SQL.Add(' a.kazaneden_id=c.kaza_id and');
   SQL.Add(' a.sehir_id=d.sehir_id and');
   SQL.Add(' a.ekemer_id=f.emn_kmr_dur_id and');
   SQL.Add(' a.ehliyet_id=e.ehliyet_id and');
   SQL.Add(' a.imal_tarih_id=g.imal_id and');
   SQL.Add(' a.direksiyon_id=h.direksiyon_id and');
   SQL.Add(' a.faktor_id=k.kaza_faktor_id and');
   SQL.Add(' a.ydurumu_id=l.yol_dur_id and');
   SQL.Add(' a.hdurumu_id=m.hava_dur_id ');
   if DTP_Tarih.Checked=true then
    begin
     SQL.Add('and tarih=:xtar');
     ParamByName('xtar').AsDate:=StrToDate(DateToStr(DTP_Tarih.Date));
    end;
   if DBLC_Kavsak.KeyValue<>0 then
    begin
     sql.Add(' and a.kavsak_id=:xkavsak_id ');
     ParamByName('xkavsak_id').Value:=DBLC_Kavsak.KeyValue;
    end;
```

```
 if DBLC_Kaza.KeyValue<>0 then
  begin
   sql.Add(' and a.kazaneden_id=:xkaza_id ');
   ParamByName('xkaza_id').Value:=DBLC_Kaza.KeyValue;
  end;
 if DBLC_Sehir.KeyValue<>0 then
  begin
   sql.Add(' and a.sehir_id=:xsehir_id ');
   ParamByName('xsehir_id').Value:=DBLC_Sehir.KeyValue;
  end;
 open;
 end;
 if Q_List.RecordCount = 0 then
begin
 StatusBar1.SimpleText:='Record not found!';
 ShowMessage('Record not found!');
end
else
 StatusBar1.SimpleText:='Founded Record(s): '+inttostr(Q_List.RecordCount);

if seviye='Admin' then
begin
 SB_Update.Enabled:=True;
 SB_Sil.Enabled:=True;
end
else if seviye='3.Seviye' then
begin
 SB_Update.Enabled:=True;
 SB_Sil.Enabled:=True;
end
else
begin
 SB_Update.Enabled:=false;
 SB_Sil.Enabled:=false;
end;
 SB_Ekle.Enabled:=false;
```

```
 begin
  sql.Add(' and a.ehliyet_id=:ehliyet_id ');
  ParamByName('ehliyet_id').Value:=DBLC_Ehliyet.KeyValue;
  kriter:=kriter+' , Licence Condition :'+DBLC_Ehliyet.Text;
 end;
if CB_yas.Text<>'' then
 begin
  sql.Add(' and a.yasi=:yasi ');
  ParamByName('yasi').Value:=CB_yas.Text;
  kriter:=kriter+' , Age :'+CB_yas.Text;
 end;
if CB_Alkol.Text<>'' then
 begin
  sql.Add(' and a.alkoloran=:alkol ');
  ParamByName('alkol').Value:=CB_Alkol.Text;
  kriter:=kriter+' , Alcohol Ratio :'+CB_Alkol.Text;
 end;
 if DBLC_Kavsak.KeyValue<>0 then
begin
 sql.Add(' and a.kavsak_id=:xkavsak_id ');
 ParamByName('xkavsak_id').Value:=DBLC_Kavsak.KeyValue;
 kriter:=kriter+' , Roundabout :'+DBLC_Kavsak.Text;
end;
if DBLC_Kaza.KeyValue<>0 then
 begin
  sql.Add(' and a.kazaneden_id=:xkaza_id ');
  ParamByName('xkaza_id').Value:=DBLC_Kaza.KeyValue;
  kriter:=kriter+' , Accident Reason :'+DBLC_Kaza.Text;
 end;
if DBLC_Sehir.KeyValue<>0 then
 begin
  sql.Add(' and a.sehir_id=:xsehir_id ');
  ParamByName('xsehir_id').Value:=DBLC_Sehir.KeyValue;
  kriter:=kriter+' , City :'+DBLC_Sehir.Text;
 end;
if (RB_Sici.Checked=true) then
```

```
 begin
   sql.Add(' and a.sehir_type=:xsehir ');
   ParamByName('xsehir').Value:=RB_Sici.Caption;
   kriter:=kriter+' , In City :'+RB_Sici.Caption;
 end;
 if (RB_Sdisi.Checked=true) then
 begin
   sql.Add(' and a.sehir_type=:xsehird ');
   ParamByName('xsehird').Value:=RB_Sdisi.Caption;
   kriter:=kriter+' , Out Of City :'+RB_Sdisi.Caption;
 end;
 if (RB_gunduz.Checked=true) then
 begin
   sql.Add(' and a.gece_gunduz=:xgunduz ');
   ParamByName('xgunduz').Value:=RB_gunduz.Caption;
   kriter:=kriter+' , Daytime :'+RB_gunduz.Caption;
 end;
 if (RB_Gece.Checked=true) then
 begin
   sql.Add(' and a.gece_gunduz=:xgece ');
   ParamByName('xgece').Value:=RB_Gece.Caption;
   kriter:=kriter+' , Evening :'+RB_Gece.Caption;
 end;
 if CB_Saat.Text<>'' then
 begin
   sql.Add(' and a.saat=:xsaat ');
   ParamByName('xsaat').Value:=CB_Saat.Text;
   kriter:=kriter+' , Hour :'+CB_Saat.Text;
 end;
 if Edit_olu.Text<>'0' then
 begin
   sql.Add(' and a.o_sayi=:xosayi ');
   ParamByName('xosayi').Value:=Edit_olu.Text;
   kriter:=kriter+' , Death People Number :'+Edit_olu.Text;
 end;
 if Edit_yarali.Text<>'0' then
```

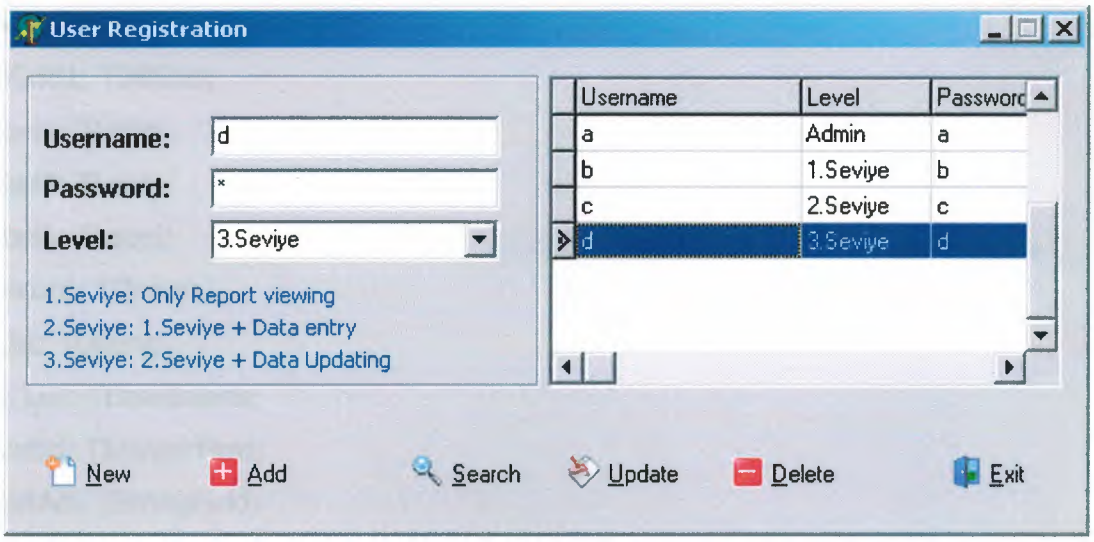
```
begin
  sql.Add(' and a.y_sayi=:xysayi ');
  ParamByName('xysayi').Value:=Edit_yarali.Text;
  kriter:=kriter+' , Injured People Number :'+Edit_yarali.Text;
end;
if CB_egitim.Text<>'' then
begin
  sql.Add(' and a.egitim=:xegitim ');
  ParamByName('xegitim').Value:=CB_egitim.Text;
  kriter:=kriter+' , Education Status :'+CB_egitim.Text;
end;
if CB_Cinsiyet.Text<>'' then
begin
  sql.Add(' and a.cinsiyet=:xcinsiyet ');
  ParamByName('xcinsiyet').Value:=CB_Cinsiyet.Text;
  kriter:=kriter+' , Sex :'+CB_Cinsiyet.Text;
end;
if DBLC_Kem_durum.KeyValue<>0 then
begin
  sql.Add(' and a.ekemer_id=:xekemer_id ');
  ParamByName('xekemer_id').Value:=DBLC_Kem_durum.KeyValue;
  kriter:=kriter+' , Seat Belt Condition :'+DBLC_Kem_durum.Text;
end;
if DBLC_arac_imal.KeyValue<>0 then
begin
  sql.Add(' and a.imal_tarih_id=:xitarih_id ');
  ParamByName('xitarih_id').Value:=DBLC_arac_imal.KeyValue;
  kriter:=kriter+' , Age Of Vehicle :'+DBLC_arac_imal.Text;
end;
if DBLC_Arac_Dir_dur.KeyValue<>0 then
begin
  sql.Add(' and a.direksiyon_id=:xdireksiyon_id ');
  ParamByName('xdireksiyon_id').Value:=DBLC_Arac_Dir_dur.KeyValue;
  kriter:=kriter+' , Steerin Wheel Condition :'+DBLC_Arac_Dir_dur.Text;
end;
if DBLC_esas_Fak.KeyValue<>0 then
```

```
  begin
    sql.Add(' and a.faktor_id=:xfaktor_id ');
    ParamByName('xfaktor_id').Value:=DBLC_esas_Fak.KeyValue;
    kriter:=kriter+' , Main Fact Of Accident :'+DBLC_esas_Fak.Text;
  end;
  if CB_Gmesafe.Text<>'' then
  begin
    sql.Add(' and a.gorus_mesafe=:xgorus_mesafe ');
    ParamByName('xgorus_mesafe').Value:=CB_Gmesafe.Text;
    kriter:=kriter+' , Sight Seeing Distance :'+CB_Gmesafe.Text;
  end;
  if DBLC_yol_dur.KeyValue<>0 then
  begin
    sql.Add(' and a.ydurumu_id=:xydurumu_id ');
    ParamByName('xydurumu_id').Value:=DBLC_yol_dur.KeyValue;
    kriter:=kriter+' , Road Condition :'+DBLC_yol_dur.Text;
  end;
  if DBLC_hava_dur.KeyValue<>0 then
  begin
    sql.Add(' and a.hdurumu_id=:xhdurumu_id ');
    ParamByName('xhdurumu_id').Value:=DBLC_hava_dur.KeyValue;
    kriter:=kriter+' , Wheather Condition :'+DBLC_hava_dur.Text;
  end;
   open;
 end;
 if not Qrapor.IsEmpty then
  begin
   Delete(kriter,17,1);
   QRLabel6.Caption:=kriter;
   QuickRep1.Preview;
  end
  else
   ShowMessage('Can not Found Searched Record Between the Certain Date(s)');
 end;
end;
end.
```

2.6 User Registration

Here, Admin of program can add or delete any user for restrict usage.

Image:



Codes:

```
unit Unituser_e;

interface

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

type
  TFormuser_e = class(TForm)
    GroupBox1: TGroupBox;
    Label3: TLabel;
    Label2: TLabel;
    Label1: TLabel;
    Edit_adi: TEdit;
    Edit_sifre: TEdit;
    ComboBox1: TComboBox;
```

```
    L_sirano: TLabel;
    SB_New: TSpeedButton;
    SB_Ekle: TSpeedButton;
    SB_Bul: TSpeedButton;
    SB_Update: TSpeedButton;
    SB_Sil: TSpeedButton;
    SpeedButton5: TSpeedButton;
    DBGrid1: TDBGrid;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Qrecord: TQuery;
    QList: TQuery;
    DS_List: TDataSource;
    QListId: TIntegerField;
    QListAdi: TStringField;
    QListSifre: TStringField;
    QListSeviye: TStringField;
    procedure SpeedButton5Click(Sender: TObject);
    procedure SB_NewClick(Sender: TObject);
    procedure SB_EkleClick(Sender: TObject);
    procedure SB_UpdateClick(Sender: TObject);
    procedure SB_SilClick(Sender: TObject);
    procedure DBGrid1CellClick(Column: TColumn);
    procedure SB_BulClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Formuser_e: TFormuser_e;

implementation

{$R *.dfm}
```

```
procedure TFormuser_e.SpeedButton5Click(Sender: TObject);
begin
 close;
end;

procedure TFormuser_e.SB_NewClick(Sender: TObject);
begin
 L_sirano.Caption:='';
 Edit_adi.Text:='';
 Edit_sifre.Clear;
 ComboBox1.Text:='';
 Edit_adi.SetFocus;
 QList.Close;
 QList.SQL.Clear;
 QList.SQL.Add('select * from accessT order by adi');
 QList.Open;
end;

procedure TFormuser_e.SB_EkleClick(Sender: TObject);
begin
if trim(Edit_adi.Text)='' then
 exit;
if trim(Edit_sifre.Text)='' then
 exit;
if trim(ComboBox1.Text)='' then
 exit;
 with Qrecord do
  begin
   close;
   SQL.Clear;
   SQL.Add('insert into accessT');
   SQL.Add('(adi,sifre,seviye)');
   SQL.Add('values(:xadi,:xsifre,:xseviye)');
   ParamByName('xadi').Value:=Edit_adi.Text;
   ParamByName('xsifre').Value:=Edit_sifre.Text;
```

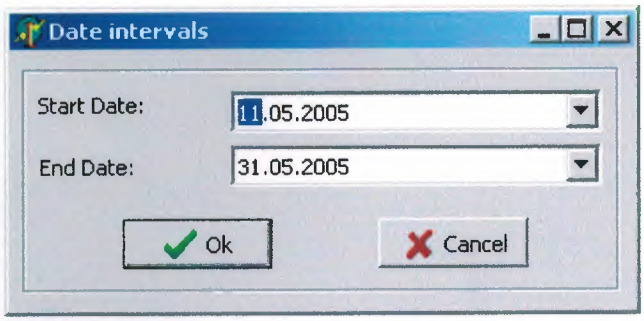
```
   ParamByName('xseviye').Value:=ComboBox1.Text;
   ExecSQL;
  end;
  SB_NewClick(self);
end;

procedure TFormuser_e.SB_UpdateClick(Sender: TObject);
begin
if L_sirano.Caption='' then exit;
 with Qrecord do
  begin
   close;
   SQL.Clear;
   SQL.Add('update accessT set');
   SQL.Add('adi=:xadi,sifre=:xsifre,seviye=:xseviye');
   SQL.Add('where id=:xid');
   ParamByName('xadi').Value:=Edit_adi.Text;
   ParamByName('xsifre').Value:=Edit_sifre.Text;
   ParamByName('xseviye').Value:=ComboBox1.Text;
   ParamByName('xid').Value:=strtoint(L_sirano.Caption);
   ExecSQL;
  end;
  SB_NewClick(self);
end;

procedure TFormuser_e.SB_SilClick(Sender: TObject);
begin
  if L_sirano.Caption='' then exit;
  with Qrecord do
   begin
    close;
    SQL.Clear;
    SQL.Add('delete from accessT where id=:xid');
    ParamByName('xid').Value:=strtoint(L_sirano.Caption);
    ExecSQL;
   end;
```

```
  SB_NewClick(self);
end;

procedure TFormuser_e.DBGrid1CellClick(Column: TColumn);
begin
  if not QList.IsEmpty then
  begin
   L_sirano.Caption:=QListId.Text;
   Edit_adi.Text:=QListAdi.Text;
   Edit_sifre.Text:=QListSifre.Text;
   ComboBox1.Text:=QListSeviye.Text;
  end;
end;

procedure TFormuser_e.SB_BulClick(Sender: TObject);
begin
  with QList do
  begin
   close;
   SQL.Clear;
   SQL.Add('select * from accessT where id>0');
    if Edit_adi.Text<>'' then
     begin
      SQL.Add(' and adi=:xadi');
      ParamByName('xadi').Value:=Edit_adi.Text;
     end;
    open;
  end;
end;

end.
```

2.7 Date intervals

Here, user can enter the certain dates to view the report

Image:



Codes:

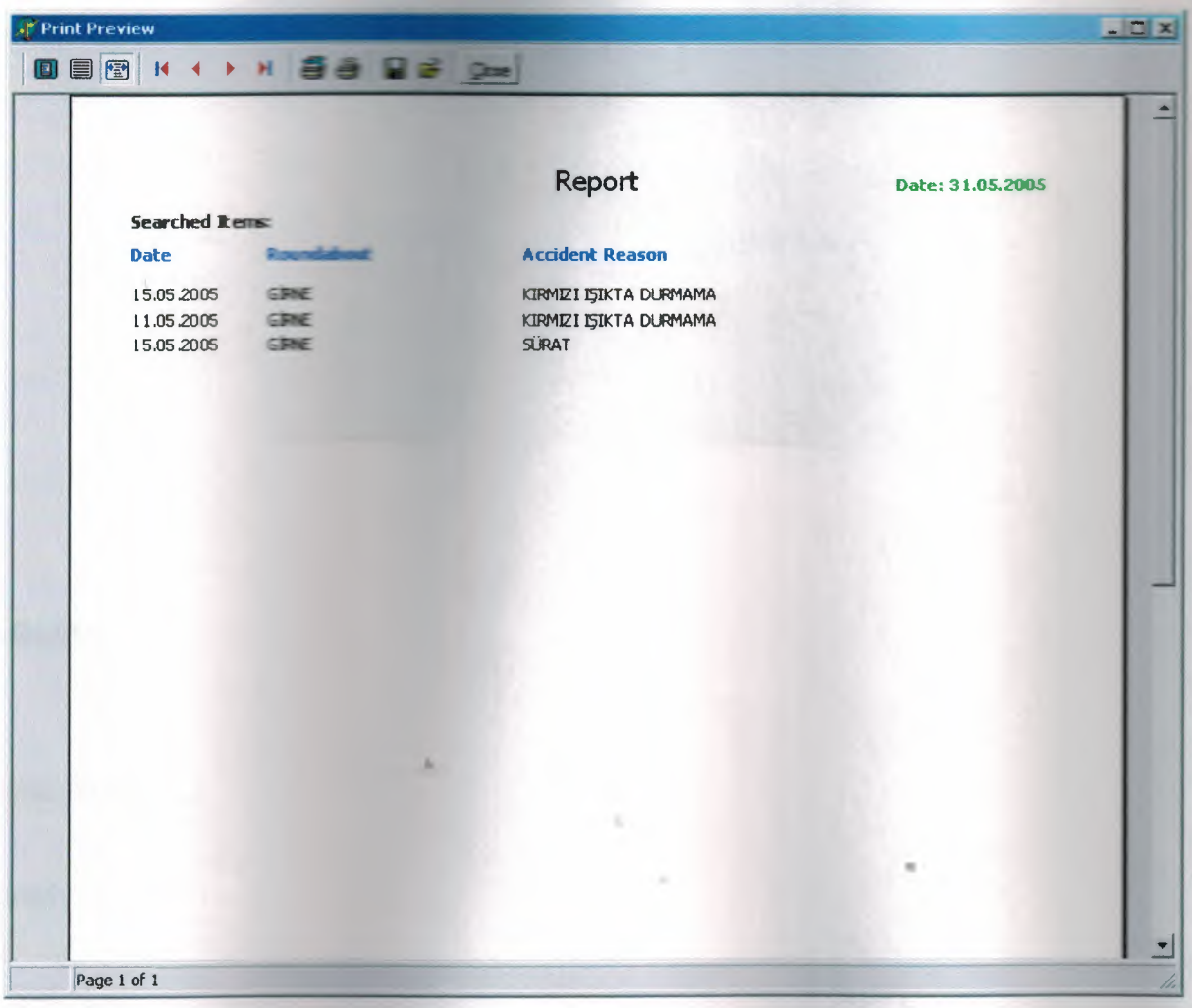
```
unit UnitTarih_e;

interface

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

type
  TFormTarih_e = class(TForm)
    GroupBox1: TGroupBox;
    DateTimePicker1: TDateTimePicker;
    DateTimePicker2: TDateTimePicker;
    Label1: TLabel;
    Label2: TLabel;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
```

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

var
  FormTarih_e: TFormTarih_e;
  buton:boolean;

implementation

{$R *.dfm}

procedure TFormTarih_e.BitBtn1Click(Sender: TObject);
begin
 buton:=true;
 close;
end;

procedure TFormTarih_e.BitBtn2Click(Sender: TObject);
begin
 buton:=false;
 close;
end;

procedure TFormTarih_e.FormActivate(Sender: TObject);
begin
 DateTimePicker1.Date:=now;
 DateTimePicker2.Date:=now;
end;

end.
```

2.8 Report

To view 'the statistics of database' as a report.

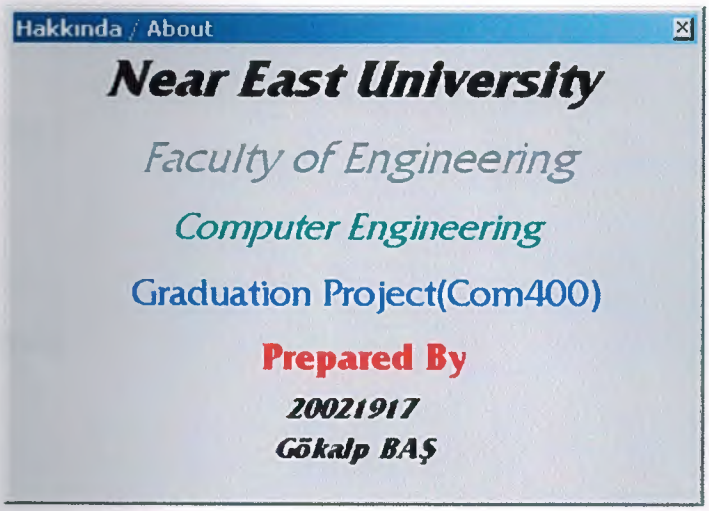
Image:



2.9 About

Information about program and programmer.

Image:



Codes:

```
unit UnitHakk;

interface

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

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

```
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    procedure FormActivate(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FormHakk: TFormHakk;

implementation

{$R *.dfm}



procedure TFormHakk.FormActivate(Sender: TObject);
var i,k,m:integer;
   //basla:boolean;
begin
i:=0;
k:=100;
 while i < 250 do
 begin
  i:=i+1;
  k:=k+1;
  FormHakk.Height:=i;
  FormHakk.Width:=k;
```

```
 end;
//
m:=180;
label1.Visible:=true;
while m > 2 do
 begin
  Application.ProcessMessages;
  m:=m-2;
  sleep(25);
  label1.Top:=m;
 end;
//
m:=196;
label2.Visible:=true;
while m > 44 do
 begin
  Application.ProcessMessages;
  m:=m-2;
  sleep(25);
  label2.Top:=m;
 end;
//
m:=204;
label3.Visible:=true;
while m > 80 do
 begin
  Application.ProcessMessages;
  m:=m-2;
  sleep(25);
  label3.Top:=m;
 end;
//
 m:=203;
 label4.Visible:=true;
 while m > 114 do
  begin
```

```
  Application.ProcessMessages;
  m:=m-2;
  sleep(25);
  label4.Top:=m;
 end;
 label5.Visible:=true;
 label6.Visible:=true;
 label7.Visible:=true;
end;

procedure TFormHakk.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 ShowMessage('Special thanks to my angel...');
 label1.Visible:=false;
 label2.Visible:=false;
 label3.Visible:=false;
 label4.Visible:=false;
 label5.Visible:=false;
 label6.Visible:=false;
 label7.Visible:=false;
end;

end.
```

REFERENCES

[1] http://Delphi.about.com

[2] http://www.Delphiturk.com

[3] Borland Delphi 7 Book