

## **ACKNOWLEDGEMENTS**

Firstly, I would like to present my special appreciation to my supervisor Prof. Dr. Dogan Ibrahim, without whom it would have not been possible for me to complete my thesis. His trust in my work and me and his priceless awareness for the project has made me do my work with full interest. His friendly behaviour with me and his words of encouragement kept me doing my thesis.

Secondly, I offer special thanks to my parents and my family, who encouraged me in every field of life and tried to help whenever I needed.

Finally, I would also like to pay my special thanks to all of my friends who helped me and encouraged me for doing my work. Their continuous encouragement and friendly environment has helped me to complete this thesis successfully. I wish to express my sincere thanks to them as they spent their time and provided very helpful suggestions to me.

# **ABSTRACT**

In industry the quality of the produced products are mainly determined by the quality of management and control system used in process control. For this reason during automation of production one of important problems is increasing the quality of used control system.

Concrete production is an important area in building construction. Concrete is produced by using special equipment and by chemically mixing cement, water, and rock. This process is usually done manually where a rotary kiln is used to mix the ingredients. The finished product is then filled into vehicles, ready for distribution to the customers.

Control and management of concrete production is traditionally carried out using programmable logic devices (PLCs). The disadvantages of using PLCs is their very high cost and the time required to adjust the production parameters. This thesis is about personal computer (PC) based concrete production automation. The system developed by the author has the advantage that the cost is very low compared to PLC design and the whole production process from the purchase of goods to the delivery of cement to the customers is automated and controlled from the central computer. A Delphi based computer program has been developed by the author to manage the process of concrete production. The program interfaces to a MySQL type database to store the various automation parameters. In addition, an 8051 microcontroller based hardware interface card has been developed by the author which can be interfaced to the serial port of a PC in order to control various phases of the concrete production process in real-time. GUI type visual output has also been provided so that the user can see production steps in real-time as the actual production takes place.

# TABLE OF CONTENTS

<b>ACKNOWLEDGEMENTS.....</b>	<b>i</b>
<b>ABSTRACT .....</b>	<b>ii</b>
<b>TABLE OF CONTENTS.....</b>	<b>iii</b>
<b>LIST OF FIGURES .....</b>	<b>iv</b>
<b>LIST OF TABLES .....</b>	<b>v</b>
<b>INTRODUCTION.....</b>	<b>1</b>
 <b>1. THE PRINCIPLES OF CONCRETE PRODUCTION.....</b>	 <b>4</b>
1.1 Overview .....	4
1.2 The Concrete Process .....	5
1.3 The Properties of Concrete.....	8
1.4 Hydration of Portland Cement .....	9
1.5 Strength of Concrete.....	12
1.6 Some Admixtures and Functions .....	14
1.7 Concrete Summary .....	15
1.8 The History of Concrete .....	17
 <b>2. CONCRETE BATCHING PLANT AND ITS AUTOMATION .....</b>	 <b>20</b>
2.1 Overview .....	20
2.2 Inline Aggregate Bunker and Wighing Belt Conveyor .....	20
2.3 Skip Hoist and Bucket Feeding .....	24
2.4 Main Chassis Superstructure .....	24
2.5 Axial Mixer, Single Shaft.....	25
2.6 Cement Silo, Silo-Top Filler, Cement Screw Conveyor .....	25
2.7 Insulated Control Cabinet.....	26
2.8 Structure of Computer Based Control of Concrete Production.....	26
 <b>3. MICROCONTROLLER BASED CONRTOL OF CONCRETE PRODUCTION .....</b>	 <b>28</b>
3.1 Overview .....	28
3.2 Block Diagram of Interface Card .....	28
3.3 Printed Circuit Diagram of Interface Card .....	31
3.4 AT89C2051 .....	33
3.5 Communication Interfaces .....	34
3.6 Bi-Directional Communications .....	34
3.7 Communicating by Bits.....	35
3.8 The Parity Bit .....	35
3.9 RS-232C .....	36
3.10 DCE and DTE Devices .....	36
3.11 9 to 25 Pin Adapters .....	39

3.12 Baud vs Bits Per Second .....	39
3.13 Cables, Null Modems, and Gender Changers .....	40
3.14 Cables Lengths .....	40
3.15 Null Modem Cables and Null Modem Adapters.....	41
3.16 Synchronous and Asynchronous Communications.....	41
 <b>4. SOFTWARE DEVELOPMENT FOR CONTROL OF CONCRETE PRODUCTION .....</b>	<b>43</b>
4.1 Overview .....	43
4.2 Software Platform .....	44
4.3 Development Tool.....	44
4.4 Developing Database Applications .....	48
4.5 Software for Control of Concrete Production .....	53
4.6 Graphical Simulation Output .....	53
4.7 System Settings .....	55
4.8 Personnel of the Company .....	56
4.9 User Rights.....	58
4.10 Stocks .....	59
4.11 Controlling the Stock In and Out Process .....	61
4.12 Products.....	64
4.13 Product Receipts.....	66
4.14 Production .....	69
4.15 Customers.....	71
4.16 Efficiency of Application of Computer Based Concrete Production Control .....	75
 <b>CONCLUSION.....</b>	<b>76</b>
 <b>REFERENCES .....</b>	<b>78</b>
 <b>APPENDIX A – INSTALLING BASCOM-8051 .....</b>	<b>80</b>
 <b>APPENDIX B – PC PROGRAM LISTING .....</b>	<b>102</b>
 <b>APPENDIX C – DATABASE STUCTURE .....</b>	<b>243</b>
 <b>APPENDIX D – INTRODUCTION TO ISO .....</b>	<b>250</b>

# LIST OF FIGURES

1.1	Schematic Diagram Of Rotary Kiln.....	6
1.2	Schematic Drawings To Demonstrate The Relationship Between The Water / Cement Ratio And Porosity.....	13
1.3	A Plot Of Concrete Strength As A Function Of The Water To Cement Ratio .....	13
1.4	History Of Cement Form 3000 B.C. To 1867.....	18
1.5	History Of Cement Form 1886 To 1992 .....	19
2.1	Computer and PLC based concrete production automation .....	20
2.2	A Typical PLC Based Concrete Automation System .....	22
2.3	Mobile Concrete Production System .....	23
2.4	Aggregate Bunker And Weighing Belt Conveyor .....	24
2.5	A Sample View Of Skip Hoist And Bucket Feeding .....	24
2.6.A	Part Of Main Chassis.....	25
2.6.B	Part Of Main Chassis .....	25
2.6.C	Part Of Main Chassis .....	25
2.7	Axial Mixer, Single Shaft.....	25
2.8.A	Cement Silo .....	26
2.8.B	Silo Top Filter.....	26
2.8.C	Cement Screw Conveyor .....	26
2.9	Control Cabinet .....	26
3.1	Block Diagram Of Interface Card .....	29
3.2	Printed Circuit Bottom View.....	30
3.3	Component Side Printed Circuit .....	30
3.4	Silk Side Of Printed Circuit .....	30
3.5	Pin Configuration Of 20-Lead PDIP/SOIC.....	32
3.6	Block Diagram Of AT89C2051 High-Performance CMOS 8-Bit Microcomputer With 2K Bytes Of Flash .....	33
3.7	RS232 DB25 Connector .....	36
3.8	RS232 DB9 Connector .....	37
4.1	PC Software Main Form .....	53
4.2	Flowchart of batching Process .....	54
4.3	Flow Control Setting .....	55
4.4	System Parametre Setting. ....	56
4.5	Aggregate Filling Order Setting. ....	56
4.6	Main Form Of Personnel Definition For The Firm. ....	57
4.7	Add A Personnel Definition For The Firm .....	57
4.8	Update Personnel Information .....	58
4.9	Delete Personnel Information.....	58
4.10	Personnel Search Dialog Form.....	59
4.11	Personnel Rights.....	59
4.12	Stock Main Form.....	60
4.13	Adding A Stock To List .....	60
4.14	Update Stock Information .....	61
4.15	Deletting The Stock Information.....	61

4.16	Stock In Or Out Process List.....	62
4.17	Stock Search Dialog Form .....	62
4.18	Add A Stock In Or Out Process .....	63
4.19	Edit Stock In Or Out Process .....	63
4.20	Delete Stock In Or Out Process.....	64
4.21	List of Products .....	64
4.22	Add New Product .....	65
4.23	Update Product Information.....	65
4.24	Delete Product Information .....	66
4.25	List Of Product Receipts .....	66
4.26	Add A Product Receipts .....	67
4.27	Update Product Receipts .....	68
4.28	Delete Product Receipts .....	68
4.29	Production List .....	69
4.30	Define the criterion of Production List .....	69
4.31	Create a New Production .....	70
4.32	Editing the Production Details .....	70
4.33	Delete the Production .....	71
4.34	Start Production .....	71
4.35	Customer List .....	72
4.36	Add A New Customer .....	72
4.37	Update Customer Details .....	73
4.38	Delete Customer Details.....	73
4.39	Accounting Details.....	74
4.40	Sale Details.....	74
4.41	Payment Details.....	75

# LIST OF TABLES

1.1	Classes Of Aggregates .....	8
1.2	Composition Of Portland Cement With Chemical Composition And Weight Percent .....	9
1.3	A Table Of Admixtures And Their Functions .....	15
3.1	A Quick Comparison of RS-232, RS-422, and RS-485 Serial Communication Interfaces .....	34
3.2	25 Pin Connector On A DTE Device .....	37
3.3	9 Pin Connector On A DTE Device .....	37
3.4	Standard 9 Pin To 25 Pin Adapter .....	39
3.5	Cable Lengths For Baud Rate .....	41
4.1	The Components List Of BDE .....	48
4.2	The Components List Of ADO .....	49
4.3	The Components List Of Dbexpress .....	50
4.4	The Components List Of Interbase .....	51
4.5	The Components List Of Data Access .....	52





# INTRODUCTION

In industry technological processes are characterized with number of controlled regime parameters. The qualitative control of these parameters directly influence the quality and on-time production of final products. To improve the quality of control system during automation of industrial processes some technologies have been developed.

In the past with the fast development of computer technology, a number of emerging software technologies can be adopted to build more powerful control systems for industrial processes. These innovative technologies include modern software engineering, object-oriented methodology, visual/graphical programming platform, graphical user interface, virtual instrumentation, componentbased system, systematic database management, dynamic data exchange, and so forth. All these technologies provide new opportunities to develop more comprehensive and reliable software artifacts than before.

In the past years before the personal computer (PC) was widely incorporated into industrial automation systems, all the faults that occurred in industrial processes were checked and dealt manually with by trained or experienced operators. For example, in the condition monitoring systems for the natural gas pipeline network, all operations were handled in a manual or semi-automatic manner, which, however, had some major drawbacks. For instance, the operator had to do the majority of the work by hand, the abnormal conditions could not be monitored and handled in real time, the remote measurement parameters could not be effectively monitored, and operators were prone to make mistakes in recording and manipulating a large amount of data.

Therefore, it is highly necessary to automate the measurement operations as well as to improve the operating efficiency. In recent decades, this picture has been dramatically changed due to the wide adoption of industrial PC in a wide range of industrial applications. A typical industrial automation system, is usually made up of the physical system, transducers, device drivers and data I/O, host computer, network server, and remote computers.

Information technologies have been rapidly developed in recent years, and they have provided sufficient technical support for building modern industrial automation systems with more open architecture with respect to the previous ones. It turns out that the computerized real-time monitoring analysis and automated technologies can realize the full automation of an industrial measurement system. The combination of emerging information technologies with traditional condition monitoring systems allows for the continuous running status monitoring for essential equipment as well as comprehensive data processing and centralized resource management. It will significantly enhance the working efficiency of system operators and decision-makers. As a result, developing such systems with the aforementioned characteristics for achieving full industrial automation has a positive practical significance in both economy and technology perspectives.

The aim of the thesis is the development of microcontroller based control and management system for concrete production. Concrete is normally produced manually by mixing chemicals such as cement, water, and rock. Automated concrete production has the advantages that the product quality is higher, the overall cost of production is lower, and the product is produced with the same consistency. A Delphi language based program has been developed by the author with a MySQL database interface. In addition, an RS232 based microcontroller interface card has been designed by the author. The card provides relay type outputs which can control various hardware points in a real plant. A graphical user interface type visual output is provided on a PC so that the user can see the automation process in real-time.

Chapter 1 provides an introduction to the process of concrete production. The chemicals used in a typical concrete production plant are described in detail in this chapter.

Chapter 2 is about concrete batching plants where the various hardware used during the concrete production process has been outlined.

Chapter 3 describes the microcontroller interface which is connected between the PC and the concrete production plant. The block diagram and the printed circuit design of the interface card and the basic principles of the RS232 serial communication is given in this chapter.

Chapter 4 describes the various computer forms developed by the author for the automation of a concrete production plant. Each form has been described in detail with examples. The software platform, and development tools are given in this chapter.

Finally, conclusions, references, and Appendices are provided at the end of the thesis.

# CHAPTER 1

## THE PRINCIPLES OF CONCRETE PRODUCTION

### 1.1 Overview

Concrete is not found in nature the way we would find aluminium, nickel or iron. Concrete is formed from combining water, a special cement and rock:



The heat, and temperature variations in general, can cause cracking problems.

A common mistake people make is to use the words cement and concrete interchangeably. It is important to remember that cement is only a component of concrete and concrete is the structural material. The cement used in concrete is not used as a building material because it would be too expensive and not as strong as concrete. So when you see a parking garage, a driveway, a sidewalk or a road remember it is made of concrete, not cement. And, by the way, that funny looking truck is a concrete mixer, not a cement mixer! But, if cement is not concrete, then what is it?

Cement is a general name for a material that binds other materials together. Yes, it is another name for glue. There are many materials which we would classify as cements and they are usually identified with certain uses, and can produce different types of "concrete". The type of cement used to make the riding surface of some of our roads (blacktop!) is called asphalt cement. It is a petroleum bi-product, and it binds rock into the road material we call asphaltic concrete.

The structural concrete used in bridges and dams and other types of road surfaces is made from Portland cement. This cement binds the rock (also called aggregate) together to form concrete. Portland cement is a mixture of processed limestone, shales, and clays which contain the following compounds: CaO (lime), Al<sub>2</sub>O<sub>3</sub> (Alumina), SiO<sub>2</sub> (silica) and iron oxides. Properties of the cement will vary depending on the relative amounts of these compounds.

Adding water to the dry cement starts a chemical reaction (hydration). While the mixture of cement, water, and rock is fluid, it can be poured into molds (called formwork) of arbitrary shape. This is a valuable property of concrete which allows us to build dams with the many different shapes which you saw in the history of dams. The compound gradually hardens into the desired final shape.

## 1.2 The Concrete Process

The importance of concrete in modern society cannot be underestimated. Look around you and you will find concrete structures everywhere such as buildings, roads, bridges, and dams. There is no escaping the impact concrete makes on your everyday life.

Concrete is a composite material which is made up of a filler and a binder. The binder (cement paste) "glues" the filler together to form a synthetic conglomerate. The constituents used for the binder are cement and water, while the filler can be fine or coarse aggregate. The role of these constituents will be discussed in this section.

Cement, as it is commonly known, is a mixture of compounds made by burning limestone and clay together at very high temperatures ranging from 1400 to 1600 °C. Although there are other cements for special purposes, this module will focus solely on portland cement and its properties.

The production of portland cement begins with the quarrying of limestone,  $\text{CaCO}_3$ . Huge crushers break the blasted limestone into small pieces. The crushed limestone is then mixed with clay (or shale), sand, and iron ore and ground together to form a homogeneous powder. However, this powder is microscopically heterogeneous.

The mixture is heated in kilns (see Figure 1.1) that are long rotating steel cylinders on an incline. The kilns may be up to 6 meters in diameter and 180 meters in length. The mixture of raw materials enters at the high end of the cylinder and slowly moves along the length of the kiln due to the constant rotation and inclination. At the low end of the kiln, a fuel is injected and burned, thus providing the heat necessary to make the materials react.

It can take up to 2 hours for the mixture to pass through the kiln, depending upon the length of the cylinder.

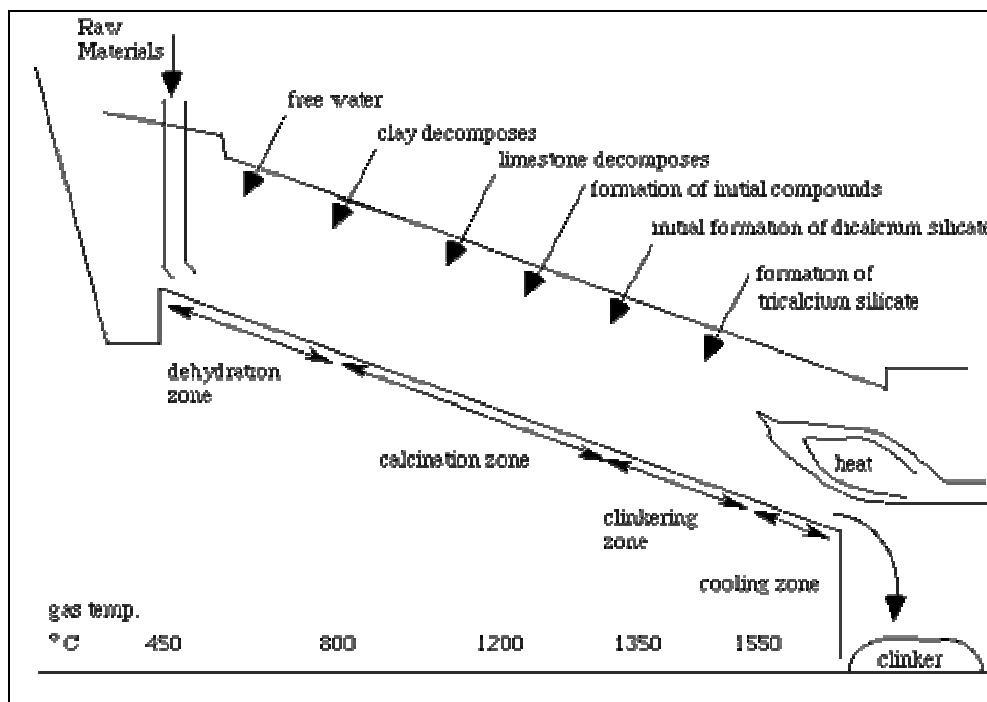


Figure 1.1. Schematic diagram of rotary kiln.

As the mixture moves down the cylinder, it progresses through four stages of transformation. Initially, any free water in the powder is lost by evaporation. Next, decomposition occurs from the loss of bound water and carbon dioxide. This is called calcination. The third stage is called clinkering. During this stage, the calcium silicates are formed. The final stage is the cooling stage.

The marble-sized pieces produced by the kiln are referred to as clinker. Clinker is actually a mixture of four compounds which will be discussed later. The clinker is cooled, ground, and mixed with a small amount of gypsum (which regulates setting) to produce the general-purpose portland cement.

Water is the key ingredient, which when mixed with cement, forms a paste that binds the aggregate together. The water causes the hardening of concrete through a process called hydration. Hydration is a chemical reaction in which the major compounds in cement form chemical bonds with water molecules and become hydrates or hydration products.

The water needs to be pure in order to prevent side reactions from occurring which may weaken the concrete or otherwise interfere with the hydration process. The role of water is important because the water to cement ratio is the most critical factor in the production of "perfect" concrete. Too much water reduces concrete strength, while too little will make the concrete unworkable.

Concrete needs to be workable so that it may be consolidated and shaped into different forms (i.e.. walls, domes, etc.). Because concrete must be both strong and workable, a careful balance of the cement to water ratio is required when making concrete.

Aggregates are chemically inert, solid bodies held together by the cement. Aggregates come in various shapes, sizes, and materials ranging from fine particles of sand to large, coarse rocks. Table 1.1 lists the classes of aggregates used in concrete production. Because cement is the most expensive ingredient in making concrete, it is desirable to minimize the amount of cement used. 70 to 80% of the volume of concrete is aggregate keeping the cost of the concrete low.

The selection of an aggregate is determined, in part, by the desired characteristics of the concrete. For example, the density of concrete is determined by the density of the aggregate. Soft, porous aggregates can result in weak concrete with low wear resistance, while using hard aggregates can make strong concrete with a high resistance to abrasion.

Aggregates should be clean, hard, and strong. The aggregate is usually washed to remove any dust, silt, clay, organic matter, or other impurities that would interfere with the bonding reaction with the cement paste. It is then separated into various sizes by passing the material through a series of screens with different size openings.

**Table 1.1.** Classes of Aggregates

Class	Examples of aggregates used	Uses
ultra-lightweight	vermiculite ceramic spheres perlite	lightweight concrete which can be sawed or nailed, also for its insulating properties
lightweight	expanded clay shale or slate crushed brick	used primarily for making lightweight concrete for structures, also used for its insulating properties.
normal weight	crushed limestone sand river gravel crushed recycled concrete	used for normal concrete projects
heavyweight	steel or iron shot steel or iron pellets	used for making high density concrete for shielding against nuclear radiation

The choice of aggregate is determined by the proposed use of the concrete. Normally sand, gravel, and crushed stone are used as aggregates to make concrete. The aggregate should be well-graded to improve packing efficiency and minimize the amount of cement paste needed. Also, this makes the concrete more workable.

### **1.3 The Properties of Concrete**

Concrete has many properties that make it a popular construction material. The correct proportion of ingredients, placement, and curing are needed in order for these properties to be optimal.

Good-quality concrete has many advantages that add to its popularity. First, it is economical when ingredients are readily available. Concrete's long life and relatively low maintenance requirements increase its economic benefits. Concrete is not as likely to rot, corrode, or decay as other building materials. Concrete has the ability to be molded or cast into almost any desired shape. Building of the molds and casting can occur on the work-site which reduces costs.



Concrete is a non-combustible material which makes it fire-safe and able withstand high temperatures. It is resistant to wind, water, rodents, and insects. Hence, concrete is often used for storm shelters.

Concrete does have some limitations despite its numerous advantages. Concrete has a relatively low tensile strength (compared to other building materials), low ductility, low strength-to-weight ratio, and is susceptible to cracking. Concrete remains the material of choice for many applications regardless of these limitations.

## 1.4 Hydration of Portland Cement

Concrete is prepared by mixing cement, water, and aggregate together to make a workable paste (see Table 1.2). It is molded or placed as desired, consolidated, and then left to harden. Concrete does not need to dry out in order to harden as commonly thought.

The concrete (or specifically, the cement in it) needs moisture to hydrate and cure (harden). When concrete dries, it actually stops getting stronger. Concrete with too little water may be dry but is not fully reacted. The properties of such a concrete would be less than that of a wet concrete.

The reaction of water with the cement in concrete is extremely important to its properties and reactions may continue for many years. This very important reaction will be discussed in detail in this section. Portland cement consists of five major compounds and a few minor compounds.

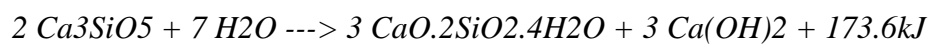
**Table 1.2.** Composition of portland cement with chemical composition and weight percent.

Cement Compound	Weight Percentage	Chemical Formula
Tricalcium silicate	50 %	$\text{Ca}_3\text{SiO}_5$ or $3\text{CaO} \cdot \text{SiO}_2$
Dicalcium silicate	25 %	$\text{Ca}_2\text{SiO}_4$ or $2\text{CaO} \cdot \text{SiO}_2$
Tricalcium aluminate	10 %	$\text{Ca}_3\text{Al}_2\text{O}_6$ or $3\text{CaO} \cdot \text{Al}_2\text{O}_3$
Tetracalcium aluminoferrite	10 %	$\text{Ca}_4\text{Al}_2\text{Fe}_2\text{O}_{10}$ or $4\text{CaO} \cdot \text{Al}_2\text{O}_3 \cdot \text{Fe}_2\text{O}_3$
Gypsum	5 %	$\text{CaSO}_4 \cdot 2\text{H}_2\text{O}$

When water is added to cement, each of the compounds undergoes hydration and contributes to the final concrete product. Only the calcium silicates contribute to strength. Tricalcium silicate is responsible for most of the early strength (first 7 days). Dicalcium silicate, which reacts more slowly, contributes only to the strength at later times. Tricalcium silicate will be discussed in the greatest detail.

The equation for the hydration of tricalcium silicate is given by:

*Tricalcium silicate + Water ---> Calcium silicate hydrate + Calcium hydroxide + heat*



Upon the addition of water, tricalcium silicate rapidly reacts to release calcium ions, hydroxide ions, and a large amount of heat. The pH quickly rises to over 12 because of the release of alkaline hydroxide (OH<sup>-</sup>) ions. This initial hydrolysis slows down quickly after it starts resulting in a decrease in heat evolved.

The reaction slowly continues producing calcium and hydroxide ions until the system becomes saturated. Once this occurs, the calcium hydroxide starts to crystallize. Simultaneously, calcium silicate hydrate begins to form. Ions precipitate out of solution accelerating the reaction of tricalcium silicate to calcium and hydroxide ions. The evolution of heat is then dramatically increased.

The formation of the calcium hydroxide and calcium silicate hydrate crystals provide "seeds" upon which more calcium silicate hydrate can form. The calcium silicate hydrate crystals grow thicker making it more difficult for water molecules to reach the unhydrated tricalcium silicate.

The speed of the reaction is now controlled by the rate at which water molecules diffuse through the calcium silicate hydrate coating. This coating thickens over time causing the production of calcium silicate hydrate to become slower and slower.

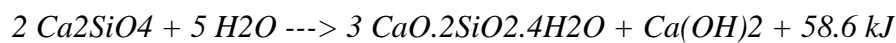
Note that the majority of space is filled with calcium silicate hydrate. That which is not filled with the hardened hydrate is primarily calcium hydroxide solution. The hydration

will continue as long as water is present and there are still unhydrated compounds in the cement paste.

Dicalcium silicate also affects the strength of concrete through its hydration. Dicalcium silicate reacts with water in a similar manner compared to tricalcium silicate, but much more slowly. The heat released is less than that by the hydration of tricalcium silicate because the dicalcium silicate is much less reactive.

The products from the hydration of dicalcium silicate are the same as those for tricalcium silicate:

*Dicalcium silicate + Water  $\rightarrow$  Calcium silicate hydrate + Calcium hydroxide + heat*



The other major components of portland cement, tricalcium aluminate and tetracalcium aluminoferrite also react with water. Their hydration chemistry is more complicated as they involve reactions with the gypsum as well. Because these reactions do not contribute significantly to strength, they will be neglected in this discussion.

Although we have treated the hydration of each cement compound independently, this is not completely accurate. The rate of hydration of a compound may be affected by varying the concentration of another.

In general, the rates of hydration during the first few days ranked from fastest to slowest are:

*tricalcium aluminate > tricalcium silicate > tetracalcium aluminoferrite > dicalcium silicate.*

Heat is evolved with cement hydration. This is due to the breaking and making of chemical bonds during hydration. The stage I hydrolysis of the cement compounds occurs rapidly with a temperature increase of several degrees. Stage II is known as the dormancy period. The evolution of heat slows dramatically in this stage.

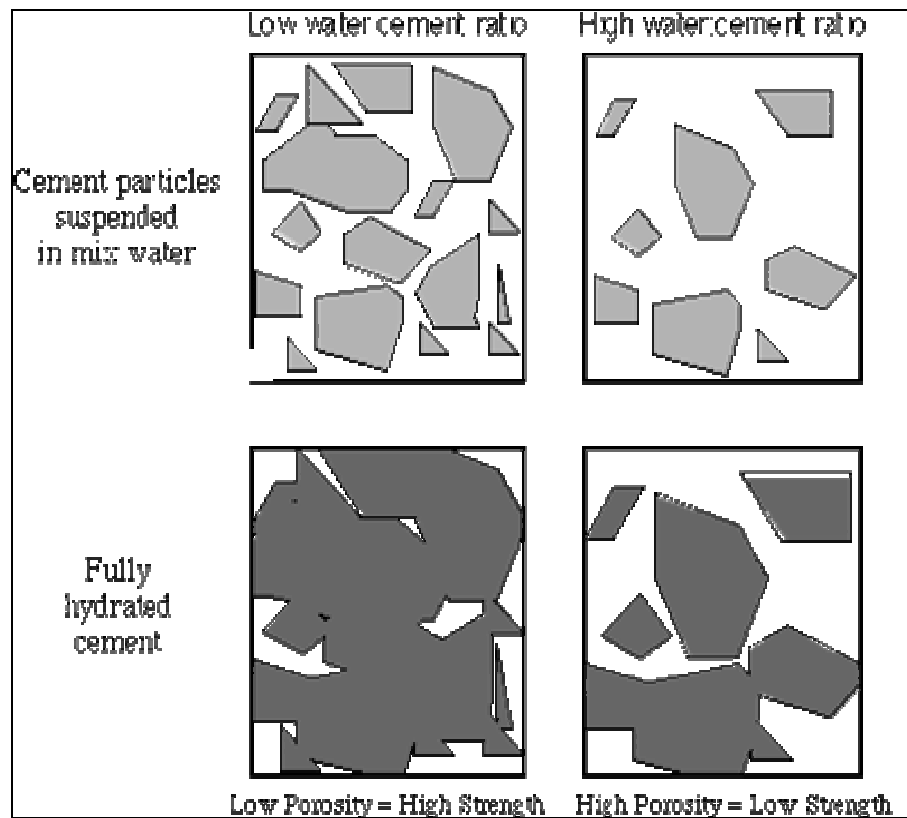
The dormancy period can last from one to three hours. During this period, the concrete is in a plastic state which allows the concrete to be transported and placed without any major difficulty. This is particularly important for the construction trade who must transport concrete to the job site. It is at the end of this stage that initial setting begins.

In stages III and IV, the concrete starts to harden and the heat evolution increases due primarily to the hydration of tricalcium silicate. Stage V is reached after 36 hours. The slow formation of hydrate products occurs and continues as long as water and unhydrated silicates are present.

## **1.5 Strength of Concrete**

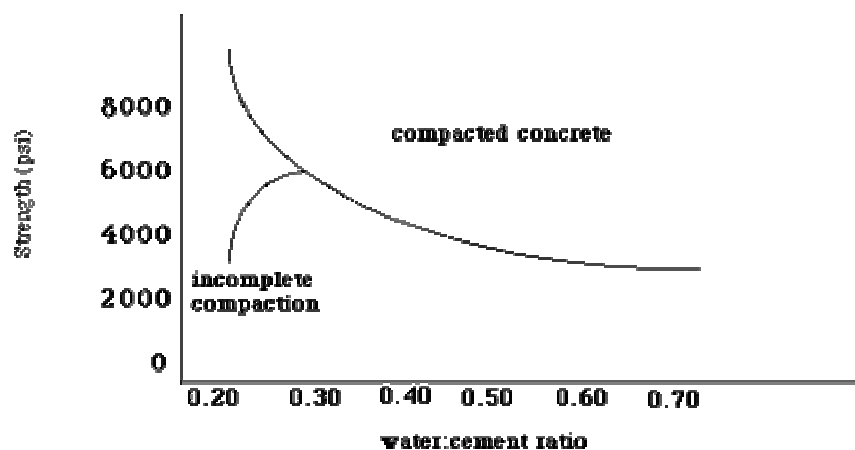
The strength of concrete is very much dependent upon the hydration reaction just discussed. Water plays a critical role, particularly the amount used. The strength of concrete increases when less water is used to make concrete. The hydration reaction itself consumes a specific amount of water. Concrete is actually mixed with more water than is needed for the hydration reactions.

This extra water is added to give concrete sufficient workability. Flowing concrete is desired to achieve proper filling and composition of the forms. The water not consumed in the hydration reaction will remain in the microstructure pore space. These pores make the concrete weaker due to the lack of strength-forming calcium silicate hydrate bonds. Some pores will remain no matter how well the concrete has been compacted. Figure 1.2 shows the relationship between the water/cement ratio and porosity.



**Figure 1.2.** Schematic drawings to demonstrate the relationship between the water/cement ratio and porosity.

The empty space (porosity) is determined by the water to cement ratio. The relationship between the water to cement ratio and strength is shown in the graph in Figure 1.3 that follows.



**Figure 1.3.** A plot of concrete strength as a function of the water to cement ratio.

Low water to cement ratio leads to high strength but low workability. High water to cement ratio leads to low strength, but good workability.

The physical characteristics of aggregates are shape, texture, and size. These can indirectly affect strength because they affect the workability of the concrete. If the aggregate makes the concrete unworkable, the contractor is likely to add more water which will weaken the concrete by increasing the water to cement mass ratio.

Time is also an important factor in determining concrete strength. Concrete hardens as time passes. Why? Remember the hydration reactions get slower and slower as the tricalcium silicate hydrate forms. It takes a great deal of time (even years!) for all of the bonds to form which determine concrete's strength. It is common to use a 28-day test to determine the relative strength of concrete.

Concrete's strength may also be affected by the addition of admixtures. Admixtures are substances other than the key ingredients or reinforcements which are added during the mixing process. Some admixtures add fluidity to concrete while requiring less water to be used. An example of an admixture which affects strength is superplasticizer.

This makes concrete more workable or fluid without adding excess water. A list of some other admixtures and their functions is given below. Note that not all admixtures increase concrete strength. The selection and use of an admixture are based on the need of the concrete user.

## **1.6 Some Admixtures And Functions**

Durability is a very important concern in using concrete for a given application. Concrete provides good performance through the service life of the structure when concrete is mixed properly and care is taken in curing it. Good concrete can have an infinite life span under the right conditions.

Water, although important for concrete hydration and hardening, can also play a role in decreased durability once the structure is built. This is because water can transport harmful chemicals to the interior of the concrete leading to various forms of deterioration.

Such deterioration ultimately adds costs due to maintenance and repair of the concrete structure. The contractor should be able to account for environmental factors and produce a durable concrete structure if these factors are considered when building concrete structures.

Table 1.3 describes briefly the admixtures and their functions.

**Table 1.3.** A table of admixtures and their functions.

Type	Function
AIR ENTRAINING	improves durability, workability, reduces bleeding, reduces freezing/thawing problems (e.g. special detergents)
SUPERPLASTICIZERS	increase strength by decreasing water needed for workable concrete (e.g. special polymers)
RETARDING	delays setting time, more long term strength, offsets adverse high temp. weather (e.g. sugar )
ACCELERATING	speeds setting time, more early strength, offsets adverse low temp. weather (e.g. calcium chloride)
MINERAL ADMIXTURES	improves workability, plasticity, strength (e.g. fly ash)
PIGMENT	adds color (e.g. metal oxides)

## 1.7 Concrete Summary

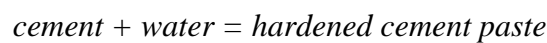
Concrete is everywhere. Take a moment and think about all the concrete encounters you have had in the last 24 hours. All of these concrete structures are created from a mixture of cement and water with added aggregate. It is important to distinguish between cement and concrete as they are not the same. Cement is used to make concrete.

$$(cement + water) + aggregate = concrete$$

Cement is made by combining a mixture of limestone and clay in a kiln at 1450[[ring]] C. The product is an intimate mixture of compounds collectively called clinker. This clinker is

finely ground into the powder form. The raw materials used to make cement are compounds containing some of the earth's most abundant elements, such as calcium, silicon, aluminum, oxygen, and iron.

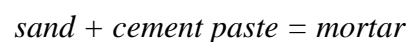
Water is a key reactant in cement hydration. The incorporation of water into a substance is known as hydration. Water and cement initially form a cement paste that begins to react and harden (set). This paste binds the aggregate particles through the chemical process of hydration. In the hydration of cement, chemical changes occur slowly, eventually creating new crystalline products, heat evolution, and other measurable signs.



The properties of this hardened cement paste, called binder, control the properties of the concrete. It is the inclusion of water (hydration) into the product that causes concrete to set, stiffen, and become hard. Once set, concrete continues to harden (cure) and become stronger for a long period of time, often up to several years.

The strength of the concrete is related to the water to cement mass ratio and the curing conditions. A high water to cement mass ratio yields a low strength concrete. This is due to the increase in porosity (space between particles) that is created with the hydration process. Most concrete is made with a water to cement mass ratio ranging from 0.35 to 0.6.

Aggregate is the solid particles that are bound together by the cement paste to create the synthetic rock known as concrete. Aggregates can be fine, such as sand, or coarse, such as gravel. The relative amounts of each type and the sizes of each type of aggregate determines the physical properties of the concrete.



Sometimes other materials are incorporated into the batch of concrete to create specific characteristics. These additives are called admixtures. Admixtures are used to: alter the fluidity (plasticity) of the cement paste; increase (accelerate) or decrease (retard) the



setting time; increase strength (both bending and compression); or to extend the life of a structure.

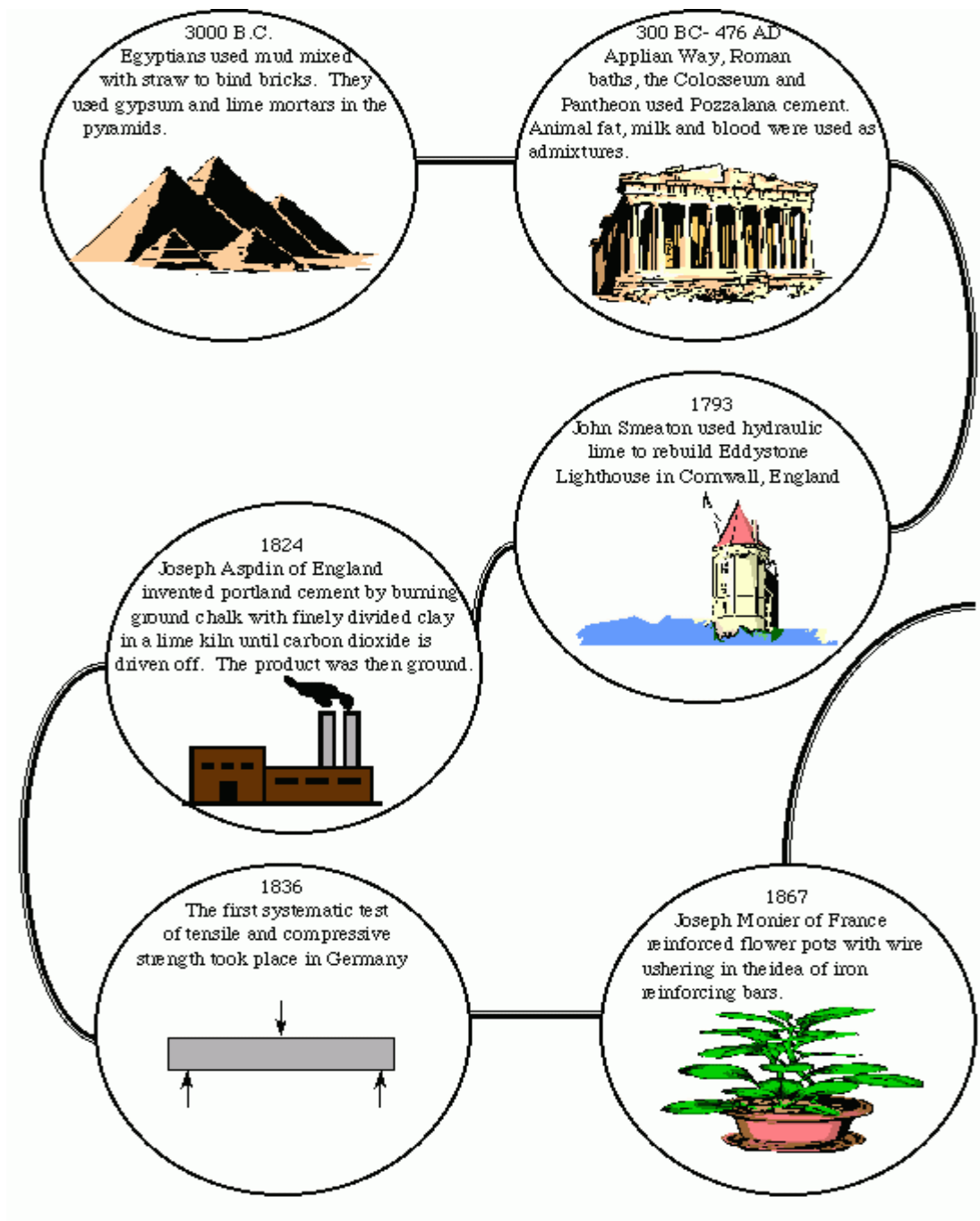
The making of concrete is a very complex process involving both chemical and physical changes. It is a material of great importance in our everyday lives.

## **1.8 The History of Concrete**

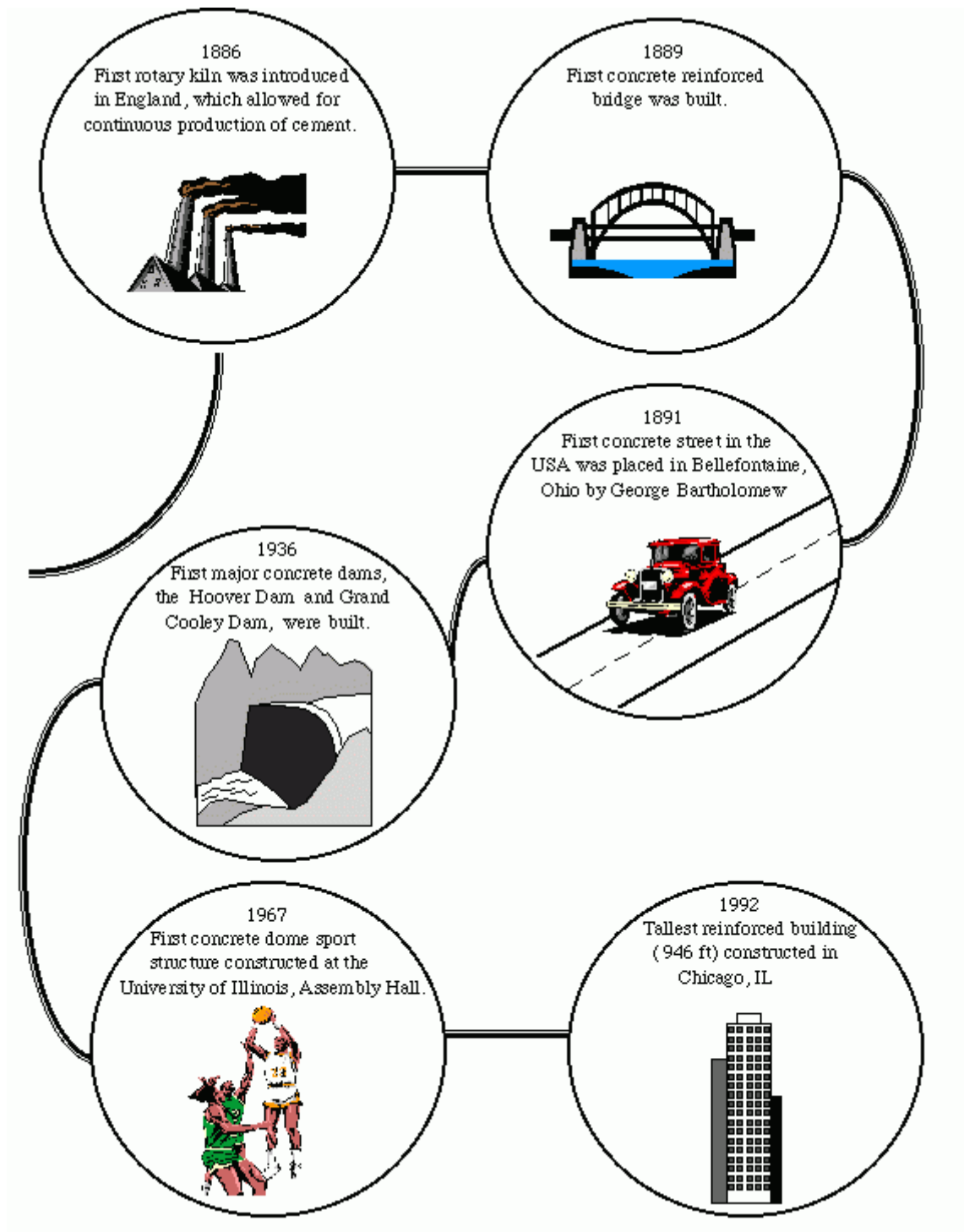
As shown in Figure 1.4 and Figure 1.5, The history of concrete dates back to 3000 B.C. where Egyptians used mud mixed straw to bind bricks. They used gypsum and lime mortars in building the pyramids. In 1793 John Smeaton used hydraulic lime to rebuild Eddystone Lighthouse in Cornwall, England. In 1824, Joseph Aspdin of England invented portland cement by burning ground chalk with finely divided clay in a lime kiln until carbon dioxide is driven off. The product was then ground. The first systematic test of tensile and compressive strength took place in Germany. Later in 1867 Joseph Monier of France reinforced flower pots with wire ushering in the idea of iron reinforcing bars.

The first rotary kiln was introduced in 1886 in England, which allowed for continuous production of cement and the first concrete reinforced bridge was built in 1889.

The first concrete street was built in the USA in 1891 in Bellefontaine by George Bartholomew. In 1936, the first major concrete dam, the Hoover Dam and Grand Coulee Dam were built. Later in 1967 the first concrete dome sport structure was built at the University of Illinois assembly hall. Finally, in 1992 the tallest reinforced building (946 feet) was constructed in Chicago.



**Figure 1.4.** History of Cement form 3000 B.C. to 1867



**Figure 1.5.** History of Cement form 1886 to 1992

# CHAPTER TWO

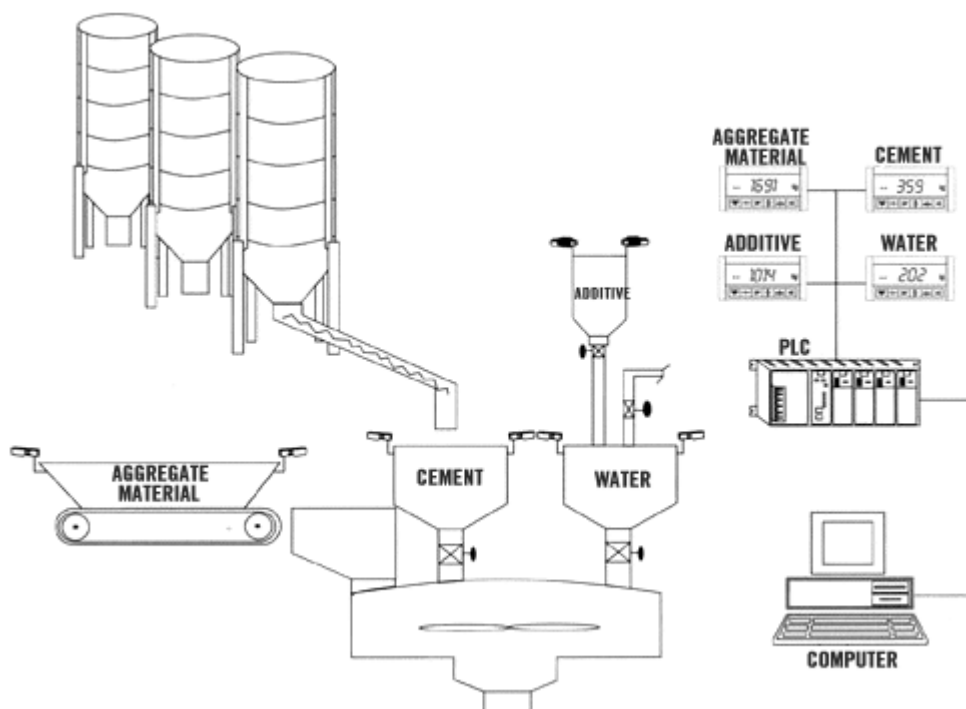
## CONCRETE BATCHING PLANT

### 2.1 Overview

This chapter describes the typical real equipment used in the production of concrete, and pictures of equipment are also given in the chapter.

### 2.2 Inline Aggregate Bunker and Weighing Belt Conveyor

Ready-mixed concrete production<sup>4</sup> requires the ingredients, that is, aggregates, cement, water, and additives to be weighed and batched as per the practice followed internationally. Figure 2.1 shows a typical concrete production plant. Usually, a PC is used to control the overall process. The PC communicates with a programmable logic controller (PLC). The inputs of the PLC are connected to load cells to measure the weight of the material. The outputs of the PLC control the various switches and solenoids so that the required amount of material is released.



**Figure 2.1** Computer and PLC based concrete production automation

A PLC is a self-contained electronic device used in automation applications. It basically consists of a processing unit, memory, and input-output unit. Just like a computer, a PLC receives inputs from sensors and switches, processes them, and provides outputs to manipulators such as motors and solenoids. PLCs do not have any keyboards or screens. The devices are normally programmed using dedicated hand-held programming devices. Alternatively, the programming can be carried out on a PC and then downloaded into the program memory of the PLC.

The main difference between a PC and a PLC is that the PC runs an operating system (e.g. Windows), it has very large program and data memory, a large hard disk, and many peripheral devices such as serial ports, parallel ports, USB ports and so on.

A PLC on the other hand is a self-contained stand alone device which runs a program stored in its non-volatile semiconductor memory. PLCs do not have any operating systems. They simply run the user program from the beginning to the end. In addition, PLCs do not have any complex interfaces such as USB ports.

As shown in Figure 2.1, the PLC receives instructions from the PC, and then controls the plant as requested. The state of the plant at any time is sent to the PC and the PC displays this data graphically on the screen.

There are many concrete production programs available in the market. Most of these programs are sold together with the hardware so that an experienced user can set up and operate a concrete production plant with little help.

ESCON<sup>5</sup> concrete production program, developed by ESIT, works under the Windows operating system. The program communicates with a PLC which is connected to the load cells, switches, and solenoids. The program has a graphical user interface which shows the state of the plant dynamically at any time. For example, the amount of material measured and released is displayed on the screen. The program provides material stock control, plant automation, stock purchase, customer database, operator database, and a display of various error conditions.

EXPER<sup>6</sup> and EVO<sup>6</sup> concrete production program is based on a client-server approach. The program runs on a Windows operating system and provides commands to a PLC. The system supports popular databases such as MySQL, ORACLE, INTERBASE, etc. A graphical user interface is provided which shows the state of the plant dynamically at any time. The program provides plant automation, stock control, operator database, stock purchase, and customer databases.

BECON-PC<sup>7</sup> is a computerized control system used for the concrete batching and mixing plants. Existing plants can be upgraded and fitted with BECON-PC. The program runs on a Windows PC and a PLC is provided for the actual control of the plant. The program runs in two modes: cycle mode, and batch mode. In cycle mode, any recipe can be programmed to run for any number of cycles, and in batch mode the recipe is programmed for any capacity mixer. A version of the program can be used in mobile mini batching operations.

AD-4820<sup>8</sup> is a ready mixed concrete batcher controller. It is basically a sophisticated PLC with a mini screen. The plant status, mixer settings, timer control settings etc can all be displayed on the screen. The PLC is connected to a PC running under the Windows operating system. Although the AD-4820 is stand-alone, using a PC makes it more user friendly.

BatchTron<sup>9</sup> is another PLC based concrete automation system. Although the system can operate on its own, it becomes more powerful when connected to a PC. A touch screen is provided which simplifies the user interface.

Figure 2.2 shows a typical PLC based concrete automation control system<sup>10</sup> consisting of power control, manual panel, weighing instruments, PLC and PC with printer. The cost of this control system is \$20,000.



**Figure 2.2** A typical PLC based concrete automation system

(Aquarius Engineers Ltd)

Some companies manufacture smaller mobile concrete production systems which can towed and pulled by a vehicle. Figure 2.3 shows<sup>11</sup> such a mobile concrete production system which is ideal in small scale concrete building applications. The cost of the system shown in Figure 2.3 is \$45,000.



**Figure 2.3** Mobile concrete production system  
(Right Manufacturing Systems)

Figure 2.4 shows a picture of the aggregate bunker and weighing belt conveyor. It is very common to use up to 4 aggregate bunkers in a typical concrete production plant. Aggregate bunker is designed and manufactured as per related DIN norms to operate under heavy working conditions for long time. There exist 2 electro-pneumatically controlled discharge gates, which provide easiness and choice in the discharge together with suitable elevation.

Aggregate weighing belt conveyor, which has weighing capacity of net 3 tons (net 1,8 m3), manufactured to operate without any problem under heavy working conditions with mechanical separator, every kind of safety precautions, air pressure measuring device and etc. as per world conditions.

Aggregate is weighed by 4 units of load cells and weighing belt conveyor has all safety switches like sensor for break off of belt, rope controlled safety switch, emergency stop button and switch that prevents belt slides.



**Figure 2.4.** Aggregate Bunker And Weighing Belt Conveyor

## 2.3 Skip Hoist And Bucket Feeding

The skip hoist and bucket feeding are used to carry the raw chemical material from the aggregate bunkers to the mixer assembly. Figure 2.5 shows the picture of a typical skip hoist and bucket feeding equipment.



**Figure 2.5.** A sample view of Skip Hoist And Bucket Feeding

## 2.4 Main Chassis Superstructure

The main chassis superstructure chassis, which provides 4250 mm useful height, is produced from section - profile and sheet iron, according to ISO9001 and all related DIN norms.

Figures 2.6a, 2.6b and 2.6c show pictures of parts of the main chassis in a typical concrete production plant.





**Figure 2.6.a**  
Part of Main Chassis



**Figure 2.6.b**  
Part of Main Chassis



**Figure 2.6.c**  
Part of Main Chassis

## 2.5 Axial Mixer, Single Shaft

Singe shaft mixers are easy to operate and long life mixers with very strong spiral shaped mixing arms, compact structure and provide fast mixing, practical loading. Water distributor and pneumatic discharge door. Figure 2.7 shows picture of an axial mixer used in a typical concrete production plant.



**Figure 2.7.** Axial Mixer, Single Shaft

## 2.6 Cement Silo, Silo-Top Filter, Cement Screw Conveyor

Figure 2.8a shows the cement silo which is typically a long cylindrical shaped container placed vertically and can be accessed via a built-in ladder. The silo top filter is shown in Figure 2.8b. Finally, the cement screw conveyor assembly is shown in Figure 2.8c



**Figure 2.8.a**  
Cement Silo



**Figure 2.8.b**  
Silo Top Filter



**Figure 2.8.c**  
Cement Screw Conveyor

## 2.7 Insulated Control Cabinet

Outer wall of the cabin is made from 0.5 mm painted galvanized material and inner wall is made from 12 mm laminated chipboard. Isolated side walls are made from 50 mm polyester hard blister and ceiling is made from 80 mm glass wool. Windows are PVC and ISICAM (heat isolated double glass). Electrical installation is imbedded. Fittings, socket and switches are exist in the cabin. Figure 2.9 shows a typical control cabinet.



**Figure 2.9** Control Cabinet

## 2.8 Structure of Computer Based Control of Concrete Production

Most computerized concrete production plants operate with a PC and a PLC. The PLC is interfaced to the plant and provides the actual physical control and monitoring of the plant. In addition, the PC provides a colourful user interface, showing the state of plant dynamically at any time. The PC program is also connected to a database to store and manipulate various plant parameters, production parameters, operator data, stock control, and customer data.

The system developed by the author is based on a PC and a microcontroller. The microcontroller is interfaced to the plant and provides the actual control of the plant. The microcontroller is under continuous control of the PC. A delphi based program runs on the PC. The program provides a colourful graphical user interface similar to the commercially available packages. As in the commercially available packages, various plant and production parameters, customer details and stock control data are stored in a database.

The advantages of the system developed by the author are:

- The system is based on a standard low-cost microcontroller which can easily be upgraded
- System architecture is simple and thus the maintenance is easy
- The overall system cost is very low compared to the commercially available systems
- Special training (e.g. PLC programming) is not required to install and operate the system
- The preparation of the necessary material, such as aggregate, cement, and water takes about 10 to 15 minutes in PLC based systems. In the system developed by the author, these activities can all be performed in less than a minute. In addition these values can be saved and then used in the future in similar applications.
- The cost of the concrete production control system shown in Figure 2.2 is \$20,000. The cost of the basic control system developed by the author is only around \$3,000, and the huge cost saving is obvious.
- Because of the complete production process is saved by the system it is possible to calculate the performance of the plant easily.

# **CHAPTER 3**

## **MICROCONTROLLER BASED CONTROL OF CONCRETE PRODUCTION**

### **3.1 Overview**

The interface card is based on the 8051 series of microcontrollers. The card is connected to the serial port (RS232 COM1 or COM2 ports) of a PC. The output of the card contains relays which are connected to the concrete production plant. The card is fitted with two power supplies: one is used to power the microcontroller, and the other one powers the relays on the interface card.

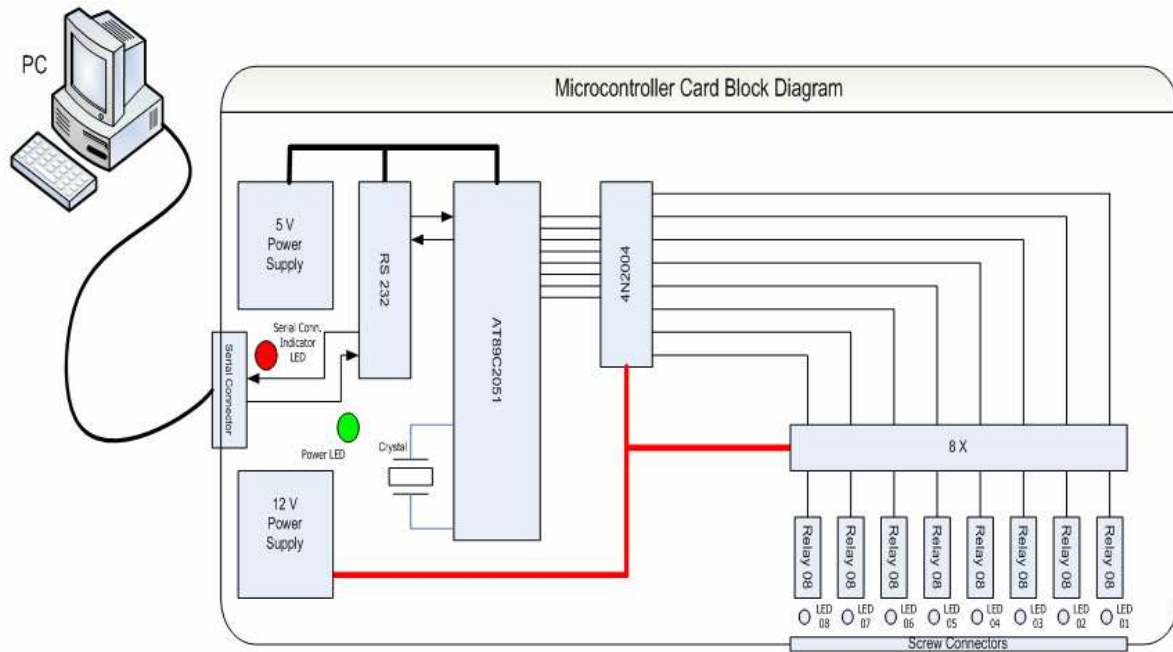
The PC sends serial data to the interface card. The card formats this data and then controls the relays so that the correct signals are sent to the concrete production plant. A graphical user interface is also provided so that the user can see the operation of the plant in real-time.

### **3.2 Block Diagram of Interface Card**

The block diagram of the interface card is shown in Figure 4.3. One end of the interface card is connected to the RS232 serial port of a PC. The other end of the interface card is connected to the concrete production plant via a set of 8 relays. A Delphi based program (developed by the author) runs on the PC and sends serial signals to the interface card. The interface card controls these relays and thus in effect controls the concrete production plant.

The card contains two power supplies: a 5V supply and a 12V supply. The 5V supply powers the microcontroller and the associated circuitry. The 12V supply powers the relays. The microcontroller drives the relays through a 4N2204 type buffer.

The outputs of the relays are also connected to a set of 8 light emitting diodes (LEDs) on a simulation board. Users can see the operation of the plant as the LEDs turn on and off while the program is working.



**Figure 3.1** Block Diagram of the Interface Card

↑  
**Relays (to the  
concrete production  
plant)**

### 3.3 Printed Circuit Diagram of Interface Card

The interface card was built on a printed circuit board. Figure 3.2 and Figure 3.3 show the bottom view and the top view of this printed circuit board. The silk screen side of the printed circuit board is shown in Figure 3.4.

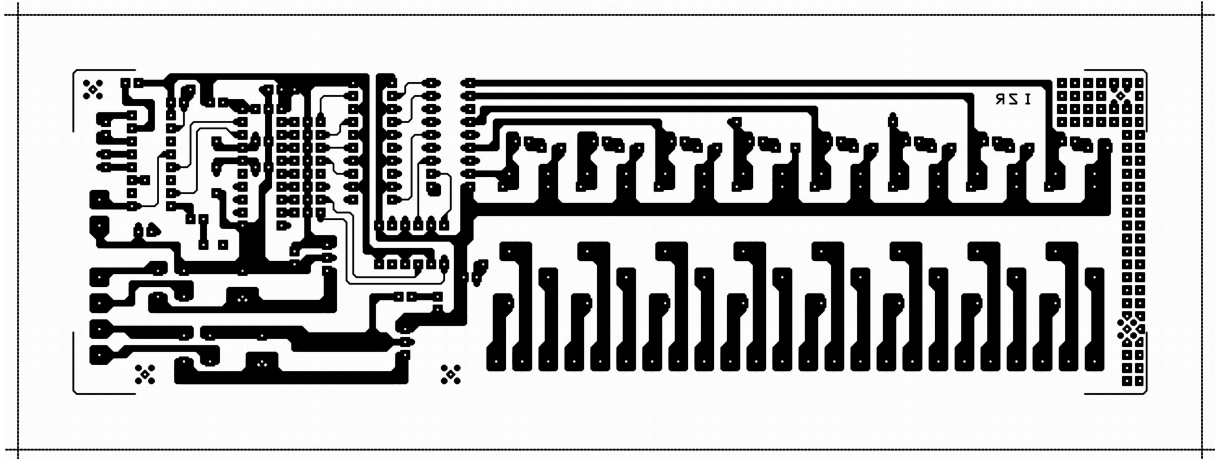


Figure 3.2 Printed circuit bottom view

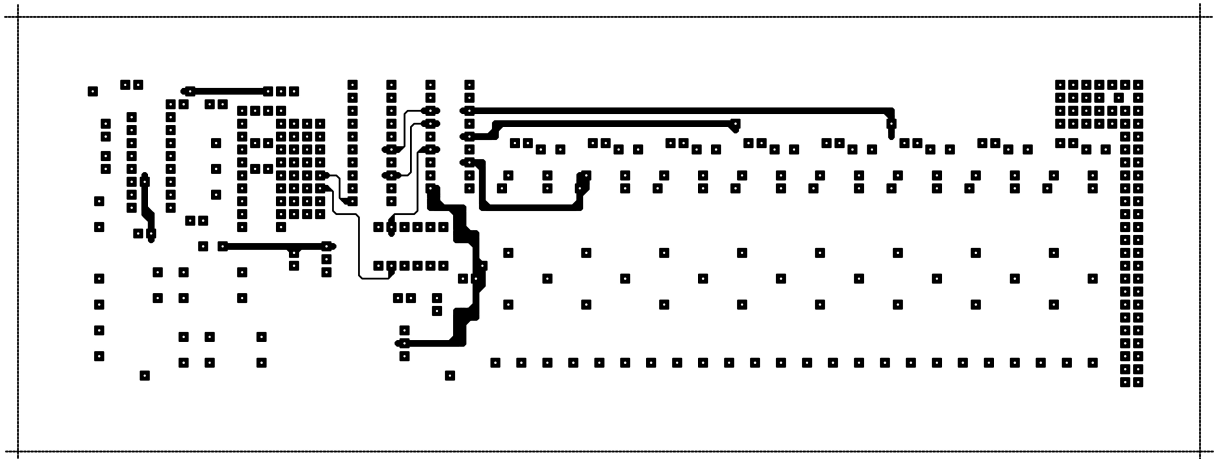


Figure 3.3 Component Side Printed Circuit

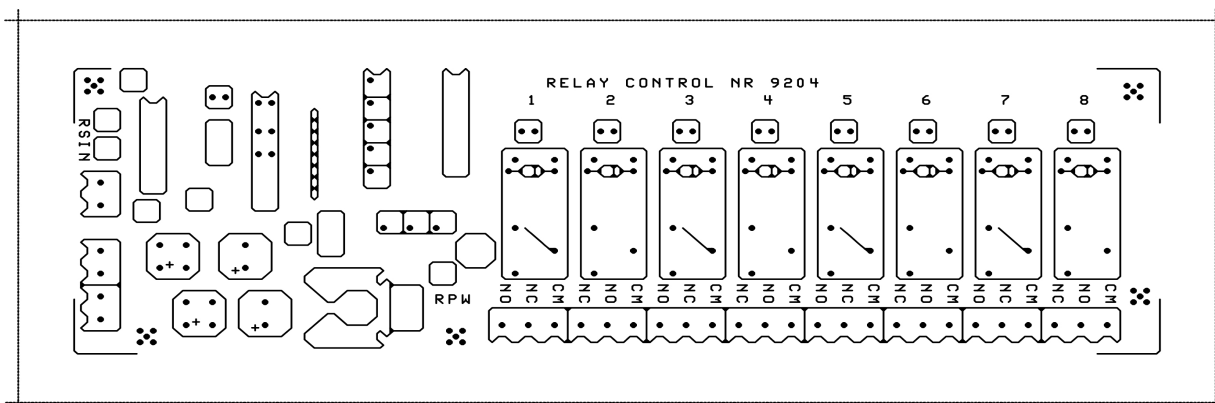


Figure 3.4 Silk Side of Printed circuit

### **3.4 AT89C2051 (8-bit Microcontroller with 2K Bytes Flash)**

The interface card is based on the AT89C2051 type of microcontroller. This is an 8-bit microcontroller which has the following specifications:

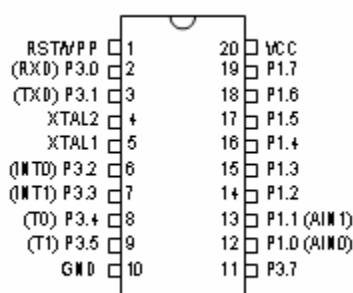
- Compatible with MCS®-51Products
- 2K Bytes of Reprogrammable Flash Memory
- Endurance: 1,000 Write/Erase Cycles
- 2.7V to 6V Operating Range
- Fully Static Operation: 0 Hz to 24 MHz
- Two-level Program Memory Lock
- 128 x 8-bit Internal RAM
- 15 Programmable I/O Lines
- Two 16-bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial UART Channel
- Direct LED Drive Outputs
- On-chip Analog Comparator
- Low-power Idle and Power-down Modes
- Green (Pb/Halide-free) Packaging Option

The AT89C2051 is a low-voltage, high-performance CMOS 8-bit microcomputer with 2K bytes of Flash programmable and erasable read-only memory (PEROM). The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard MCS-51 instruction set. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C2051 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications.

The AT89C2051 provides the following standard features: 2K bytes of Flash, 128 bytes of RAM, 15 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, a precision analog comparator, on-chip oscillator and clock circuitry.

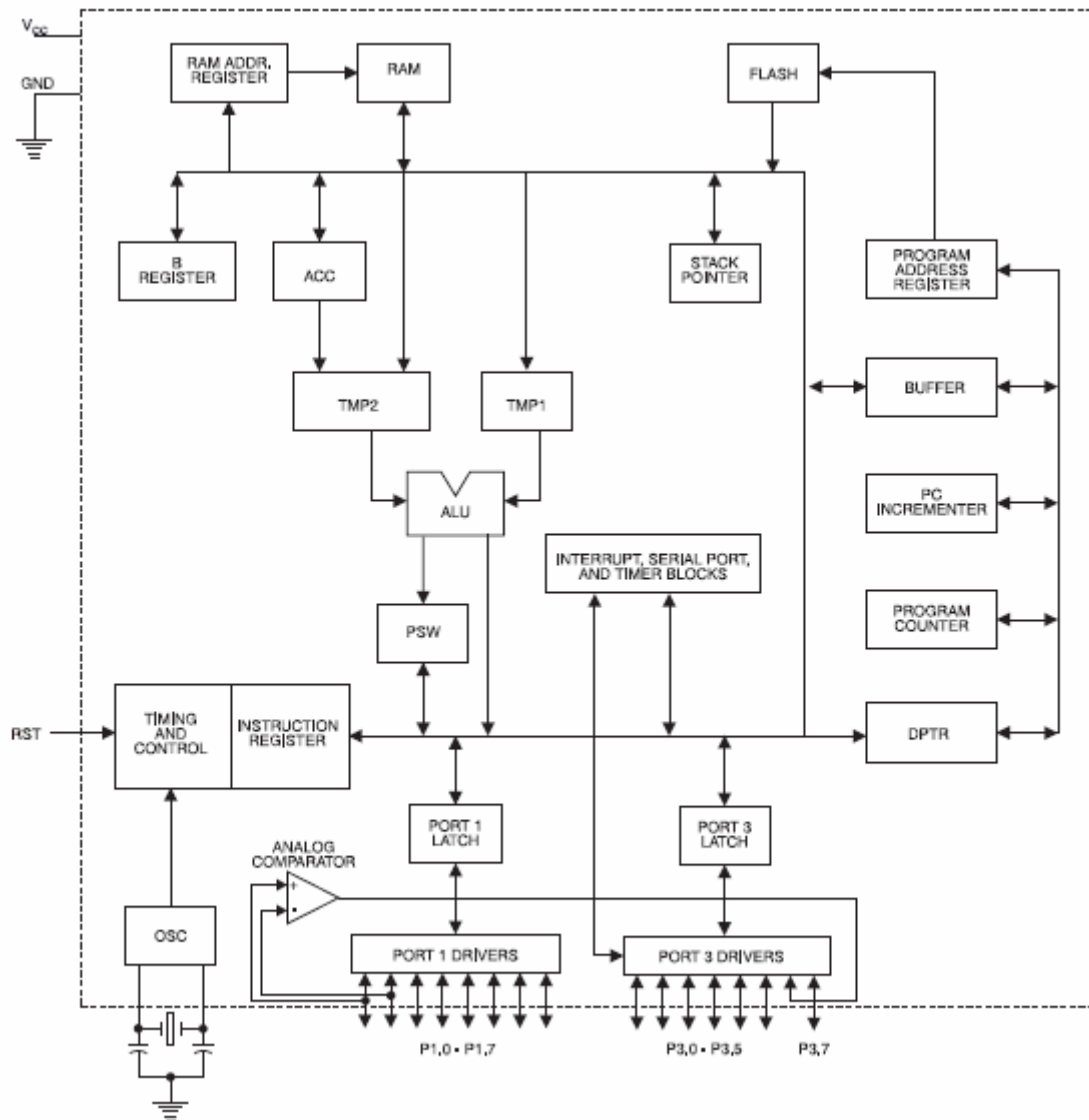
In addition, the AT89C2051 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The power-down mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

The pin configuration of the microcontroller is shown in Figure 4.1. The block diagram of the microcontroller is given in Figure 4.2.



**Figure 3.5** Pin Configuration of 20-lead PDIP/SOIC





**Figure 3.6** Block Diagram of AT89C2051 high-performance CMOS 8-bit microcomputer with 2K bytes of Flash

### 3.5 Communication Interfaces

RS-232 is the most common and ships with most PCs as COM1 and COM2. Note that RS-422/RS-485 interface is NOT available with the COM1 and COM2 ports on most PCs. However, RS-422 is the standard with Apple Macintosh computers. RS-485 is an improvement upon RS-422 and is more common in industry.

The chart below compares mode of operation, total number of drivers and receivers, maximum cable length, and maximum data rate.

**Table 3.1** A Quick Comparison of RS-232, RS-422, and RS-485 Serial Communication Interfaces

Specifications	RS-232	RS-422	RS-485
Mode of Operation	Single-Ended	Differential	Differential
Total Number of Drivers and Receivers on One Line (one driver active at a time for RS-485)	1 Driver / 1 Receiver	1 Driver / 10 Receivers	32 Driver / 32 Receiver
Maximum cable length	50 ft. (2500 pF)	4000 ft.	4000 ft.
Maximum Data Rate (40 ft. -4000 ft. for RS-422/RS485)	20 KB/s (by spec. Can be higher)	10 Mbits/s	10 Mbits/s

All IBM PC and compatible computers are typically equipped with two serial ports and one parallel port. Although these two types of ports are used for communicating with external devices, they work in different ways.

A parallel port sends and receives data eight bits at a time over 8 separate wires. This allows data to be transferred very quickly; however, the cable required is more bulky because of the number of individual wires it must contain. Parallel ports are typically used to connect a PC to a printer and are rarely used for much else.

A serial port sends and receives data one bit at a time over one wire. While it takes eight times as long to transfer each byte of data this way, only a few wires are required. In fact, two-way (full duplex) communications is possible with only three separate wires - one to send, one to receive, and a common signal ground wire.

### 3.6 Bi-Directional Communications

The serial port on the PC is a full-duplex device meaning that it can send and receive data at the same time. In order to be able to do this, it uses separate lines for transmitting and receiving data. Some types of serial devices support only one-way communications and therefore use only two wires in the cable - the transmit line and the signal ground.

### 3.7 Communicating by Bits

Once the start bit has been sent, the transmitter sends the actual data bits. There may either be 5, 6, 7, or 8 data bits, depending on the number you have selected. Both receiver and the transmitter must agree on the number of data bits, as well as the baud rate. Almost all devices transmit data using either 7 or 8 databits.

Notice that when only 7 data bits are employed, one can not send ASCII values greater than 127. Likewise, using 5 bits limits the highest possible value to 31. After the data has been transmitted, a stop bit is sent. A stop bit has a value of 1 - or a mark state - and it can be detected correctly even if the previous data bit also had a value of 1. This is accomplished by the stop bit's duration. Stop bits can be 1, 1.5, or 2 bit periods in length.

### 3.8 The Parity Bit

Besides the synchronization provided by the use of start and stop bits, an additional bit called a parity bit may optionally be transmitted along with the data. A parity bit affords a small amount of error checking, to help detect data corruption that might occur during transmission.

One can choose either even parity, odd parity, mark parity, space parity or none at all. When even or odd parity is being used, the number of marks (logical 1 bits) in each data byte are counted, and a single bit is transmitted following the data bits to indicate whether the number of 1 bits just sent is even or odd.

For example, when even parity is chosen, the parity bit is transmitted with a value of 0 if the number of preceding marks is an even number. For the binary value of *0110 0011* the parity bit would be 0. If even parity were in effect and the binary number *1101 0110* were sent, then the parity bit would be 1. Odd parity is just the opposite, and the parity bit is 0 when the number of mark bits in the preceding word is an odd number.

Parity error checking is very rudimentary. While it will tell you if there is a single bit error in the character, it doesn't show which bit was received in error. Also, if an even number of bits are in error then the parity bit would not reflect any error at all.

Mark parity means that the parity bit is always set to the mark signal condition and likewise space parity always sends the parity bit in the space signal condition. Since these two parity options serve no useful purpose whatsoever, they are almost never used.

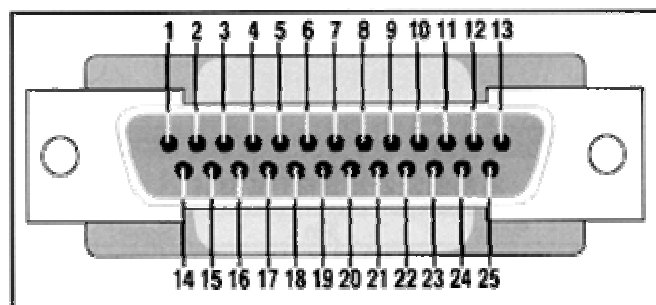
### 3.9 RS-232C

RS-232 stands for Recommend Standard number 232 and C is the latest revision of the standard. The serial ports on most computers use a subset of the RS-232C standard. The full RS-232C standard specifies a 25-pin "D" connector of which 22 pins are used. Most of these pins are not needed for normal PC communications, and indeed, most new PCs are equipped with male D type connectors having only 9 pins.

### 3.10 DCE and DTE Devices

Two terms one should be familiar with are DTE and DCE. DTE stands for Data Terminal Equipment, and DCE stands for Data Communications Equipment. These terms are used to indicate the pin-out for the connectors on a device and the direction of the signals on the pins. Your computer is a DTE device, while most other devices are usually DCE devices. If you have trouble keeping the two straight then replace the term "DTE device" with "your PC" and the term "DCE device" with "remote device" in the following discussion.

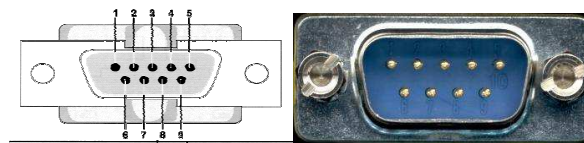
The RS-232 standard states that DTE devices use a 25-pin male connector, and DCE devices use a 25-pin female connector. You can therefore connect a DTE device to a DCE using a straight pin-for-pin connection. However, to connect two like devices, you must instead use a null modem cable.



**Figure 3.7** RS232 DB25 connector

**Table 3.2** 25 Pin Connector on a DTE device

Pin Number	Direction of signal
1	Protective Ground
2	Transmitted Data (TD) Outgoing Data (from a DTE to a DCE)
3	Received Data (RD) Incoming Data (from a DCE to a DTE)
4	Request To Send (RTS) Outgoing flow control signal controlled by DTE
5	Clear To Send (CTS) Incoming flow control signal controlled by DCE
6	Data Set Ready (DSR) Incoming handshaking signal controlled by DCE
7	Signal Ground Common reference voltage
8	Carrier Detect (CD) Incoming signal from a modem
20	Data Terminal Ready (DTR) Outgoing handshaking signal controlled by DTE
22	Ring Indicator (RI) Incoming signal from a modem

**Figure 3.8** RS232 DB9 connector**Table 3.3** 9 Pin Connector on a DTE device

Pin Number	Direction of signal
1	Carrier Detect (CD) (from DCE) Incoming signal from a modem
2	Received Data (RD) Incoming Data from a DCE
3	Transmitted Data (TD) Outgoing Data to a DCE
4	Data Terminal Ready (DTR) Outgoing handshaking signal
5	Signal Ground Common reference voltage
6	Data Set Ready (DSR) Incoming handshaking signal
7	Request To Send (RTS) Outgoing flow control signal
8	Clear To Send (CTS) Incoming flow control signal
9	Ring Indicator (RI) (from DCE) Incoming signal from a modem

The TD (transmit data) wire is the one through which data from a DTE device is transmitted to a DCE device. This name can be deceiving, because this wire is used by a DCE device to receive its data. The TD line is kept in a mark condition by the DTE device when it is idle. The RD (receive data) wire is the one on which data is received by a DTE device, and the DCE device keeps this line in a mark condition when idle.

RTS stands for Request To Send. This line and the CTS line are used when "hardware flow control" is enabled in both the DTE and DCE devices. The DTE device puts this line in a mark condition to tell the remote device that it is ready and able to receive data. If the DTE device is not able to receive data (typically because its receive buffer is almost full), it will put this line in the space condition as a signal to the DCE to stop sending data.

When the DTE device is ready to receive more data (i.e. after data has been removed from its receive buffer), it will place this line back in the mark condition.

The complement of the RTS wire is CTS, which stands for Clear To Send. The DCE device puts this line in a mark condition to tell the DTE device that it is ready to receive the data. Likewise, if the DCE device is unable to receive data, it will place this line in the space condition. Together, these two lines make up what is called RTS/CTS or "hardware" flow control.

The Software Wedge supports this type of flow control, as well as Xon/XOff or "software" flow control. Software flow control uses special control characters transmitted from one device to another to tell the other device to stop or start sending data. With software flow control the RTS and CTS lines are not used.

DTR stands for Data Terminal Ready. Its intended function is very similar to the RTS line. DSR (Data Set Ready) is the companion to DTR in the same way that CTS is to RTS. Some serial devices use DTR and DSR as signals to simply confirm that a device is connected and is turned on. The Software Wedge sets DTR to the mark state when the serial port is opened and leaves it in that state until the port is closed.

The DTR and DSR lines were originally designed to provide an alternate method of hardware handshaking. It would be pointless to use both RTS/CTS and DTR/DSR for flow control signals at the same time. Because of this, DTR and DSR are rarely used for flow control.

CD stands for Carrier Detect. Carrier Detect is used by a modem to signal that it has made a connection with another modem, or has detected a carrier tone. The last remaining line

is RI or Ring Indicator. A modem toggles the state of this line when an incoming call rings your phone.

The Carrier Detect (CD) and the Ring Indicator (RI) lines are only available in connections to a modem. Because most modems transmit status information to a PC when either a carrier signal is detected (i.e. when a connection is made to another modem) or when the line is ringing, these two lines are rarely used. ,

### 3.11 9 to 25 Pin Adapters

The following table (Table 3.4) shows the connections inside a standard 9 pin to 25 pin adapter.

**Table 3.4** Standard 9 pin to 25 pin adapter

9 Pin Connector		25 Pin Connector	
Pin 1	DCD	Pin 8	DCD
Pin 2	RD	Pin 3	RD
Pin 3	TD	Pin 2	TD
Pin 4	DTR	Pin 20	DTR
Pin 5	GND	Pin 7	GND
Pin 6	DSR	Pin 6	DSR
Pin 7	RTS	Pin 4	RTS
Pin 8	CTS	Pin 5	CTS
Pin 9	RI	Pin 22	RI

### 3.12 Baud vs. Bits Per Second

The baud unit is named after Jean Maurice Emile Baudot, who was an officer in the French Telegraph Service. He is credited with devising the first uniform-length 5-bit code for characters of the alphabet in the late 19th century. What baud really refers to is modulation rate or the number of times per second that a line changes state.

This is not always the same as bits per second (BPS). If you connect two serial devices together using direct cables then baud and BPS are in fact the same. Thus, if you are running at 19200 BPS, then the line is also changing states 19200 times per second. But when considering modems, this isn't the case.

Because modems transfer signals over a telephone line, the baud rate is actually limited to a maximum of 2400 baud. This is a physical restriction of the lines provided by the phone company. The increased data throughput achieved with 9600 or higher baud modems is accomplished by using sophisticated phase modulation, and data compression techniques.

### **3.13 Cables, Null Modems, and Gender Changers**

In a perfect world, all serial ports on every computer would be DTE devices with 25-pin male "D" connectors. All other devices would be DCE devices with 25-pin female connectors. This would allow you to use a cable in which each pin on one end of the cable is connected to the same pin on the other end.

Unfortunately, we do not live in a perfect world. Serial ports use both 9 and 25 pins, many devices can be configured as either DTE or DCE, and - as in the case of many data collection devices - may use completely non-standard or proprietary pin-outs. Because of this lack of standardization, special cables called null modem cables, gender changers and custom made cables are often required.

### **3.14 Cables Lengths**

The RS-232C standard imposes a cable length limit of 50 feet. You can usually ignore this "standard", since a cable can be as long as 10000 feet at baud rates up to 19200 if you use a high quality, well shielded cable. The external environment has a large effect on lengths for unshielded cables. In electrically noisy environments, even very short cables can pick up stray signals.

The following chart offers some reasonable guidelines for 24 gauge wire under typical conditions. You can greatly extend the cable length by using additional devices like optical isolators and signal boosters. Optical isolators use LEDs and Photo Diodes to isolate each line in a serial cable including the signal ground.

Any electrical noise affects all lines in the optically isolated cable equally - including the signal ground line. This causes the voltages on the signal lines relative to the signal



ground line to reflect the true voltage of the signal and thus canceling out the effect of any noise signals.

**Table 3.5** Cable Lengths for Baud Rate

Baud Rate	Shielded Cable Length	Unshielded Cable Length
110	5000	1000
300	4000	1000
1200	3000	500
2400	2000	500
4800	500	250
9600	250	100

### 3.15 Null Modem Cables and Null Modem Adapters

If you connect two DTE devices (or two DCE devices) using a straight RS232 cable, then the transmit line on each device will be connected to the transmit line on the other device and the receive lines will likewise be connected to each other. A Null Modem cable or Null Modem adapter simply crosses the receive and transmit lines so that transmit on one end is connected to receive on the other end and vice versa.

In addition to transmit and receive, DTR or DSR, as well as RTS or CTS are also crossed in a Null Modem connection. Null Modem adapters are available at most computer and office supply stores for under \$5.

### 3.16 Synchronous and Asynchronous Communications

There are two basic types of serial communications, synchronous and asynchronous. With synchronous communications, the two devices initially synchronize themselves to each other, and then continually send characters to stay in sync. Even when data is not really being sent, a constant flow of bits allows each device to know where the other is at any given time. That is, each character that is sent is either actual data or an idle character.

Synchronous communications allows faster data transfer rates than asynchronous methods, because additional bits to mark the beginning and end of each data byte are not required. The serial ports on IBM-style PCs are asynchronous devices and therefore only support synchronous serial communications.

Asynchronous means "no synchronization", and thus does not require sending and receiving idle characters. However, the beginning and end of each byte of data must be identified by start and stop bits. The start bit indicates when the data byte is about to begin and the stop bit signals when it ends. The requirement to send these additional two bits causes asynchronous communication to be slightly slower than synchronous however it has the advantage that the processor does not have to deal with the additional idle characters.

An asynchronous line that is idle is identified with a value of 1 (also called a mark state). By using this value to indicate that no data is currently being sent, the devices are able to distinguish between an idle state and a disconnected line. When a character is about to be transmitted, a start bit is sent. A start bit has a value of 0 (also called a space state). Thus, when the line switches from a value of 1 to a value of 0, the receiver is alerted that a data character is about to be sent.

# **CHAPTER 4**

## **SOFTWARE DEVELOPMENT FOR CONTROL OF CONCRETE PRODUCTION**

### **4.1 OVERVIEW**

This chapter describes the details of the Delphi automation program developed by the author. The program enables the user to set-up the production parameters and then to start the concrete production process. A graphical user interface is provided to show the operation of the program as the plant is controlled.

Visual programming was a hot spot in the 1990s. With the rise of the graphical user interface, a great deal of attention has been paid to the user interface design in software design. To expedite the development of the graphical user interface, Windows provides API (Application Programming Interface), which contains a huge number of functions. However, a large amount of function parameters and constants make the Windows API-based software development still very difficult.

Object Windows provides a large amount of default standard handling functions, which significantly reduce the work of application development. However, there is still a heavy burden for the developer to understand and grasp it. To resolve this, a collection of visual development tools is developed based on the Windows APIs or Object Windows of Borland C++.

In the visual development environment, the developer can operate on the interface elements, and the application software is automatically generated by the visual development tool. This type of software is usually event-based. For each event, the system generates appropriate messages and passes them to their corresponding message handling functions. These message handling functions are automatically loaded by the visual development tool in building the software.

## **4.2 Software Platform**

Microsoft's Windows operating systems have become the fastest-growing development platform in the fields of industrial measurement and control in the past several years.

Also we noticed that Linux was developing at a blindingly fast speed and more and more software developers employed it as their development platform, especially for the development of embedded real-time industrial applications.

But even up until now, for most operation and management personnel in small and medium-sized companies, they are more accustomed to operating the software in Windows platforms.

Especially in the harsh and stressful environments such as the industrial measurement and control field, they prefer the more friendly and more familiar Windows interfaces. Therefore, the Windows operating system is selected as development platform.

Although we also admit that Linux is very promising and it has many merits that the Windows-based operating systems cannot offer.

## **4.3 Development Tool**

After choosing the operating system, we need to choose the development tool for the software. The selection principle is that the tool should be able to provide a robust, fast, and efficient development environment.

Under such a principle, Borland Delphi programming language was selected, a visible programming platform, to develop the automation software.

Borland Delphi language is used to code software in order to obtain multitasking functions (e.g., simultaneous execution of data analysis on the stage and data acquisition on

the background) and intuitive graphical user interfaces (e.g., simulation map, waveform displays, and alarms list).

Delphi makes Windows development easy with drag-and-drop visual programming, and a comprehensive Visual Component Library (VCL) with a number of reusable components.

It is able to create, debug, and deploy Windows applications in an efficient manner. It is also fully supported by a wide variety of industry standards such as Win32 API, COM, ActiveX, and many others.

In addition Delphi has the following major merits:

- 1) Its visual programming characteristic enables the programming efficiency to be higher than that in the conventional Windows application development languages such as C/C++, which may greatly shorten the software development cycle.
- 2) Delphi is a suitable programming platform for industrial monitoring and control system development. For instance, its true 32-bit compiler is able to generate the high-speed executable. Therefore, the speed demand on the software can be fully satisfied.
- 3) Delphi is fully object-oriented, so it is able to support all of the object-oriented, implementations such as encapsulation, inheritance, polymorphism, dynamic binding, and so on.
- 4) Delphi is able to conveniently call the Windows API functions, so it can be used together with other languages such as assembly and C/C++ to accomplish a particular task. This feature is very vital to the development of industrial monitoring software as software functions may have to be implemented using different languages.
- 5) Delphi offers powerful database access and manipulation capability, as well as high component reusability and expandability.

- 6) Delphi provides a large amount of components, and it also allows users to develop their own components. Once the home-made components have been properly installed and registered, they can be equally used by different applications as the built-in Delphi components.

Of course, Delphi is not merely an editor or a compiler, it also contains various outstanding characteristics, which made it a comprehensive application development environment.

It turns out to be able to make the development much simpler because it has the following remarkable characteristics:

❖ **Customizable development environment:**

For many years, people kept working on the traditional integrated development tools, e.g., the editor, compiler, and debugger. The Windows development environment establishes user interfaces and automatically generates the support code.

Delphi inherits and extends these characteristics and features. The open tool API permits supporting tools to be seamlessly incorporated into Delphi's integrated development environment (IDE).

❖ **Object orientation:**

Delphi is truly object-oriented, so it permits the programmer to merge the data and code into a class (i.e., encapsulation), establish new inherited class (i.e., inheritance), and take the derived class as the parents class (i.e., polymorphism).

❖ **Component library and template:**

The screen elements of various Windows applications are extremely similar. For instance, the standard button is a gray rectangular button, and the text on its surface is used to demonstrate its function.

Delphi implements the function of this kind of buttons by enabling them to respond to mouse operations as well as text display. Delphi has a comprehensive visual component

library (VCL), which contains various kinds of objects to be used for establishing Win32 applications.

The invention of templates makes the programming much simpler. Delphi has defined four types of templates, i.e., the window, application, component, and code templates. The first three templates allow users to use a custom-made object set unlimited times in different applications, or use them as the basis for the new application development. The code template significantly reduces the repetitive programming work.

#### ❖ **Complete compilation:**

Many Windows development environments use an incompletely compiled or pseudo code. The pseudo code cannot be directly executed by the machine. It must be translated into the executable code at runtime. As a result, this disadvantage enormously reduces the software execution speed and therefore the system performance.

On the contrary, Delphi uses the complete compiler and linker so that it is able to produce pure locally executable code.

Another advantage for complete code compilation is that it may establish the dynamic link library containing any component from the component library. This type of dynamic link library can be used to expand the Delphi application, or provide services to the application developed by the simpler tool.

#### ❖ **Powerful application:**

Delphi uses the concept of Exception to handle the operation errors. Unlike the previous error handling techniques, which suppose the program to make mistakes in each execution step, Delphi presumes that each statement in the user program is correct.

In the presence of any fault, Delphi activates an Exception and then forwards it to the corresponding exception handler. This strategy enables the program to restore from faulty conditions very quickly and reliably.

#### ❖ Data accessing:

Delphi has the powerful database management ability. Because a majority of user applications inevitably have to deal with a massive volume of data, Delphi takes data handling as the heart of application development, which primarily includes data collection, processing, and presentation.

This is also an advantage of incorporating computer technologies into the software design. The objects and components provided by Delphi enormously simplify the database application development.

Due to the above reasons, I chose Delphi as our main development tool. The practical application demonstrates that using Delphi can markedly reduce the system development burden and thus enable the developer to focus on the optimization of system functionality.

## 4.4 Developing Database Applications

Delphi includes many components for accessing databases and representing the information they contain. They are grouped according to the data access mechanism:

The BDE page of the Component palette contains components that use the Borland Database Engine (BDE). The BDE defines a large API for interacting with databases. Of all the data access mechanisms, the BDE supports the broadest range of functions and comes with the most supporting utilities. It is the best way to work with data in Paradox or dBASE tables. However, it is also the most complicated mechanism to deploy.

The components on the BDE page of the Component palette (not available in all editions) enable you to connect to database information using the Borland Database Engine :

**Table 4.1** The Components List of BDE

Component	Definition
<b>Table</b>	Retrieves data from a physical database table via the BDE and supplies it to one or more data-aware components through a DataSource component. Conversely, it also sends data received from a component to a physical database via the BDE.



<b>Query</b>	Uses SQL statements to retrieve data from a physical database table via the BDE and supplies it to one or more data-aware components through a TDataSource component. Conversely, uses SQL statements to send data from a component to a physical database via the BDE.
<b>StoredProc</b>	Enables an application to access server stored procedures. Sends data received from a component to a physical database via the BDE. See Working with stored procedures.
<b>Database</b>	Sets up a persistent connection to a database, especially a remote database requiring a user login and password.
<b>Session</b>	Provides global control over a group of Database components. A default TSession component is automatically created for each database application. You must use the TSession component only if you are creating a multithreaded database application. Each database thread requires its own session component.
<b>BatchMove</b>	Copies a table structure or its data. Can be used to move entire tables from one database format to another.
<b>UpdateSQL</b>	Lets you use cached updates support with read-only datasets. For example, you could use a TUpdateSQL component with a "canned" query to provide a way to update the underlying datasets, essentially giving you the ability to post updates to a read-only dataset. You associate a TUpdateSQL component with a dataset by setting the dataset's UpdateObject property. The dataset automatically uses the TUpdateSQL component when cached updates are applied.
<b>NestedTable</b>	Retrieves the data in a nested dataset field and supplies it to data-aware controls through a datasource component.

The ADO page of the Component palette contains components that use ActiveX Data Objects (ADO) to access database information through OLEDB. ADO is a Microsoft Standard. There is a broad range of ADO drivers available for connecting to different database servers. Using ADO-based components lets you integrate your application into an ADO-based environment (for example, making use of ADO-based application servers).

The components on the ADO page of the Component palette let you connect to database information using ActiveX Data Objects (ADO):

**Table 4.2** The Components List of ADO

<b>Component</b>	<b>Definition</b>
<b>ADOConnection</b>	Sets up a persistent connection to an ADO database and provides support for transactions.
<b>ADOCommand</b>	Issues SQL commands directly against an ADO database without returning a result set.

<b>ADODataSet</b>	Represents the data from one or more tables in an ADO database and allows data-aware components to manipulate this data by connecting with a DataSource component. This is the most generic ADO dataset control, and can be used in place of ADOTable, ADOQuery, or ADOStoredProc.
<b>ADOTable</b>	Represents the data from a single database table via ADO and allows data-aware components to manipulate this data by connecting with a DataSource component.
<b>ADOQuery</b>	Uses SQL statements to retrieve data from a physical database table via ADO and allows data-aware components to manipulate this data by connecting with a DataSource component.
<b>RDSConnection</b>	Manages the marshaling of data when a Recordset object is passed from one process or machine to another. Use TRDSConnection when building multi-tier applications using business objects (Application Servers).
<b>ADOStoredProc</b>	Enables an application to access a server's stored procedures using ADO.

The dbExpress page of the Component palette contains components that use dbExpress to access database information. dbExpress is a lightweight set of drivers that provide the fastest access to database information. In addition, dbExpress components support cross-platform development because they are also available on Linux.

However, dbExpress database components also support the narrowest range of data manipulation functions. The components on the dbExpress page of the Component palette allow applications to communicate with databases using dbExpress:

**Table 4.3** The Components List of dbExpress

<b>Component</b>	<b>Definition</b>
<b>SQLConnection</b>	Encapsulates a dbExpress connection to a database server.
<b>SQLDataSet</b>	Represents data retrieved using dbExpress.
<b>SQLQuery</b>	Represents a query that is executed using dbExpress.
<b>SQLStoredProc</b>	Represents a stored procedure that is executed using dbExpress.
<b>SQLTable</b>	Represents a database table that is accessed using dbExpress.
<b>SQLMonitor</b>	Intercepts messages that pass between an SQL connection component and a database server and saves them in a string list.
<b>SimpleDataSet</b>	TSimpleDataSet is a client dataset that uses an internal TSQLDataSet and TDataSetProvider for fetching data and applying updates.

The InterBase page of the Component palette contains components that access InterBase databases directly, without going through a separate engine layer.

The components on the InterBase page of the Component palette let you connect directly to an InterBase database without using an engine such as the BDE or ActiveX Data Objects (ADO).

**Table 4.4** The Components List of InterBase

<b>Component</b>	<b>Definition</b>
<b>IBTable</b>	Represents the data from a single InterBase table or view.
<b>IBQuery</b>	Uses SQL statements to retrieve data from an InterBase table or tables. TIBQuery is more easily scaled than other IB datasets when moving from local InterBase to a remote InterBase server.
<b>IBStoredProc</b>	Executes an InterBase Execute stored procedure. IBStoredProc does not represent a result set: use IBQuery or IBDataSet for stored procedures that return a result set.
<b>IBDatabase</b>	Represents the InterBase database connection. Use this component to manage transactions or provide connection parameters for a remote database.
<b>IBTransaction</b>	Provides discrete transaction control over a one or more database connections. The IBDatabase component uses IBTransaction to represent a transaction.
<b>IBUpdateSQL</b>	Lets you use cached update support with read-only queries. For example, you could use a IBUpdateSQL component with a "canned" query to provide a way to update the underlying datasets, giving you the ability to post updates to a read-only dataset.
<b>IBDataSet</b>	Represents the result set from an SQL SELECT command. IBDataSet lets you specify separate SQL commands for inserting, deleting, and updating records.
<b>IBSQL</b>	Executes an InterBase SQL statement with minimal overhead. IBSQL has no standard interface to data-aware controls and is unidirectional.
<b>IBDatabaseInfo</b>	Returns information about an attached database, such as the version of the online disk structure (ODS), the number of cache buffers allocated, the number of database pages read from or written to, or write-ahead log information.
<b>IBSQLMonitor</b>	Monitors dynamic SQL passed to the InterBase server. It introduces a single event, OnSQL, which receives the text for every dynamic SQL statement based to the server.
<b>IBEvents</b>	Lets an application register interest in, and asynchronously handle, events posted by an InterBase server
<b>IBExtract</b>	Fetches metadata information, such as a list of tables, views, roles, and indexes from an InterBase server.
<b>IBClientDataSet</b>	Caches updates for data fetched using InterBase Express without using an external provider and client dataset.

The Data Access page of the Component palette contains components that can be used with any data access mechanism. This page includes TClientDataset, which can work with data stored on disk or, using the TDataSetProvider component also on this page, with components from one of the other groups.

The components on the Data Access page of the Component palette (not available in all editions) enable you to work with database information. These components are not associated with any particular data access mechanism (such as dbExpress or the Borland Database Engine).

**Table 4.5** The Components List of Data Access

<b>Component</b>	<b>Definition</b>
DataSource	Acts as a conduit between a dataset component and data-aware components such as TDBGrid. See Associating a data control with a dataset.
ClientDataSet	Implements a dataset caches its data in memory. Client datasets can be used with local file-based data (MyBase) or with data fetched from another dataset using a dataset provider.
DataSetProvider	Encodes data into packets than can be sent to clients and applies updates that are received from clients.
XMLTransform	Converts an XML document into a data packet or an XML data packet into an XML document with another structure.
XMLTransformProvider	Provides data from an XML document to a client dataset or XML broker and resolves updates from that client dataset or XML broker back to the source XML document.
XMLTransformClient	Converts the data from a provider into an XML document and applies updates in the form of an XML document to a provider component.

Different versions of Delphi include different drivers for accessing database servers using the BDE, ADO, or dbExpress. When designing a database application, you must decide which set of components to use. Each data access mechanism differs in its range of functional support, the ease of deployment, and the availability of drivers to support different database servers. In addition to choosing a data access mechanism, you must choose a database server. There are different types of databases and you will want to consider the advantages and disadvantages of each type before settling on a particular database server.

All types of databases contain tables which store information. In addition, most (but not all) servers support additional features such as :

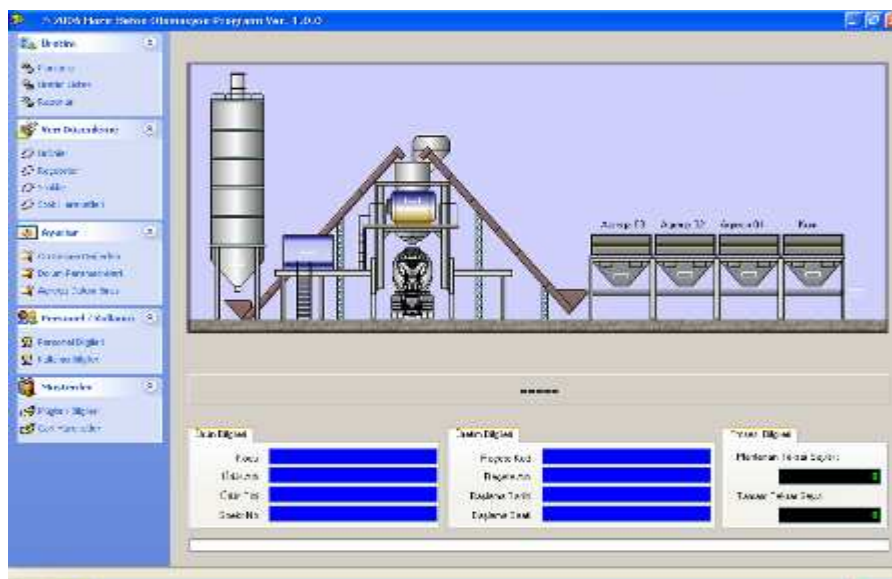
- Database security
- Transactions
- Referential integrity, stored procedures, and triggers

## 4.5 Software For Control Of Concrete Production

The PC program is a real-time program based on the Delphi language. The program is used to configure the plant parameters, and start and stop the production plant automatically. The flow diagram of the concrete batch processing is shown in Figure 4.2.

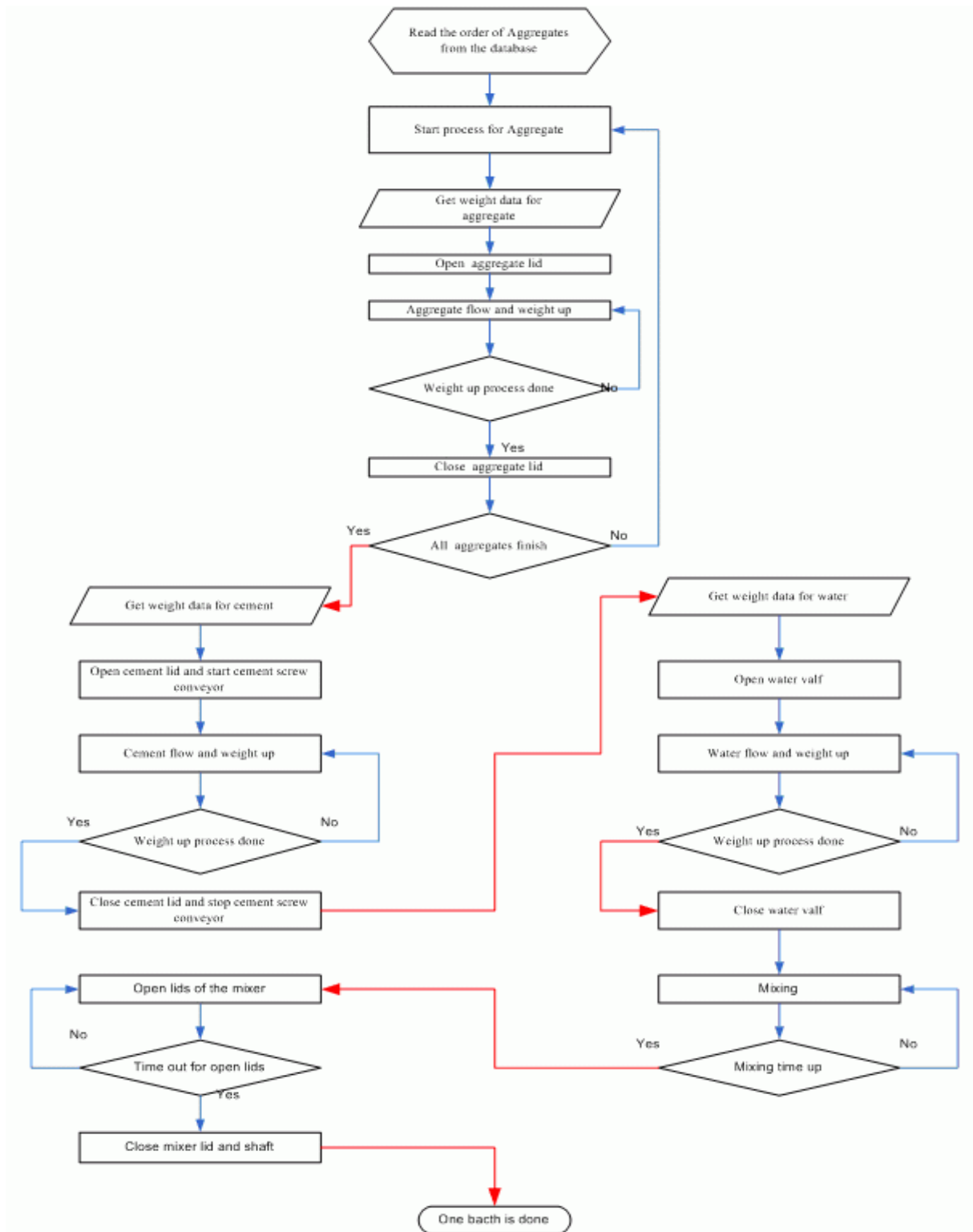
## 4.6 Graphical Simulation Output

The Delphi program provides a simulation output on the PC screen so that the user can see the simulation steps in real time. The simulation output is shown in the Figure 4.1 below:



**Figure 4.1** PC Software Main Form

Simulated lights are placed at various points of the diagram and these lights turn on and off as the simulation progresses. The interface card is also equipped with LEDs at the same points and thus the simulation is a close representation of what is being sent to the real plant.



**Figure 4.2** Flowchart Of Batching Process

## 4.7 System Settings

Aggregate, cement and sand fall down on parts which have load-cells to measure the weight. Flow must be stopped when the right amount of aggregate, cement, or sand are output.

As an example if one wants to measure 100 kg of sand one must close the lid when the weight reaches 90 kg . After flow of sand is finished we will get the correct amount. i.e 100 kg .These cutting edges are not same for aggregate, cement and sand and these values can be set using the form shown in Figure 4.3, known as the “flow control setting” form.

Agrega Ön Kesme Değerleri	
Sistemin kararlı bir şekilde çalışması için parametreleri tanımlayınız. Bilgileri kayıt için Kaydet tuşunu kullanın.	
F4 - Kaydet F10 - Kapat	
Ön Kesme Bilgileri :	
01 No`lu Agregâ :	222
02 No`lu Agregâ :	22,000 Kg
03 No`lu Agregâ :	33,000 Kg
Kum :	44,000 Kg
Çimento :	55,000 Kg
Katkı :	66,000 Kg
© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0	

**Figure 4.3** Flow Control Setting

The system parameters are also important and they must be set correctly before the start of the production. These parameters are the delay in emptying the concrete, delay in emptying the water, the time that the mixer lid should be kept open, and the aggregate weighing tolerance. Figure 4.4 shows the form that is used to set these system parameters. This form is called the “system parameter setting” form.

**Sistem Parametreleri**  
Sistemin kararlı bir şekilde çalışması için parametreleri tanımlayınız.  
Bilgileri kayıt için Kaydet tuşunu kullanın.

F4 - Kaydet F10 - Kapat

**Parametre Bilgileri**

Çimento Boşaltım Gecikmesi (Sn) :	117	Su Boşaltım Gecikmesi (Sn) :	222
Katkı Boşaltım Gecikmesi (Sn) :	249	Mixer Kapağının Açık Kalma Süresi (Sn) :	248
Agrega Tartı Toleransı (Kg) :	247		

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.4** System Parametre Setting

Order of the aggregate filling can be set by the operator as shown in Figure 4.4. This form is called the “aggregate filling order setting” form.

**Agrega Dolun Sırası**  
Agregaların dolun sıralamasını ok tuşlarıyla belirleyiniz.  
Kayıt için Kaydet tuşunu kullanın.

F4 - Kaydet F10 - Kapat

**Sıralama :**

Kum Dolun	↑ ↓
01 No`lu Agregas Dolun	
02 No`lu Agregas Dolun	
03 No`lu Agregas Dolun	

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.5** Aggregate Filling Order Setting.

## 4.8 Personnel of the Company

The details of the personnel using the program can be displayed using the “personnel listing” form shown in Figure 4.6. Unlimited number of personnel can be listed for the firm. In this example the details of 5 people are listed. The following details are displayed for each person: personnel number, name, surname, username, password, and whether or not the user is an administrator.



**Personel Listesi**  
Firmanıza personel ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni personel kayıt için ekle tuşunu kullanın, ilgili alanları doldurduktan sonra kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

Personel No	Personel Adı	Personel Soyadı	Kullanıcı Adı	Şifre	Sistem Yöneticisi
33	HAKAN	AKKUZZU	HAKAN	hakan	<input checked="" type="checkbox"/>
1	Ali	Özgür	ALI	ali	<input type="checkbox"/>
102	Kaan	Aslan	KAAN	kaan	<input type="checkbox"/>
203	Mustafa	Tuncay	MUSTAFA	mustafa	<input type="checkbox"/>
302	Hüseyin	Karahüseyin	HÜSEYİN	hüseyin	<input type="checkbox"/>

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.6** Main form of personnel definition for the firm.

Adding a personnel to the database is an easy task. The following details are required when a new personnel data is to be entered into the system: name, surname, username, password, personel number, and whether or not the person is an administrator. Figure 4.7 shows the form used to enter new personnel to the system.

**Personel Listesi**  
Firmanıza personel ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni personel kayıt için ekle tuşunu kullanın, ilgili alanları doldurduktan sonra kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

**Yeni Personel Ekle**

Adı :  Kart No :  0 Kullanıcı Adı :  ☒ Kaydet  
Soyadı :  ☐ Sistem Yöneticisi Şifre :  ☒ İptal

Personel No	Personel Adı	Personel Soyadı	Kullanıcı Adı	Şifre	Sistem Yöneticisi
33	HAKAN	AKKUZZU	HAKAN	hakan	<input checked="" type="checkbox"/>
1	Ali	Özgür	ALI	ali	<input type="checkbox"/>
102	Kaan	Aslan	KAAN	kaan	<input type="checkbox"/>
203	Mustafa	Tuncay	MUSTAFA	mustafa	<input type="checkbox"/>
302	Hüseyin	Karahüseyin	HÜSEYİN	hüseyin	<input type="checkbox"/>

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.7** Add a personnel definition for the firm.

The personnel details can be changed using the form shown in Figure 4.8. This form is named “update personnel information”. The name of the person whose details are to be changed is selected from the list and then the new details are entered and the update button is clicked to confirm the modifications.

**Personel Listesi**  
Firmanıza personel ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni personel kayıt için ekle tuşunu kullanın, ilgili alanları doldurduktan sonra kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

**Personel Bilgisini Değiştir**

Adı : HAKAN Kart No : 33 Kullanıcı Adı : HAKAN Güncelle  
Soyadı : AKKUZZU Sistem Yöneticisi Şifre : hakan İptal

Personel No	Personel Adı	Personel Soyadı	Kullanıcı Adı	Şifre	Sistem Yöneticisi
33	HAKAN	AKKUZZU	HAKAN	hakan	<input checked="" type="checkbox"/>
1	Ali	Özgür	ALI	ali	<input type="checkbox"/>
102	Kaan	Aslan	KAAN	kaan	<input type="checkbox"/>
203	Mustafa	Tuncay	MUSTAFA	mustafa	<input type="checkbox"/>
302	Hüseyin	Karahüseyin	HÜSEYİN	hüseyin	<input type="checkbox"/>

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.8** Update personnel information

Any personel data can be deleted if required. For example, if a personnel leaves the firm it will be required to delete the personnel details from the database. This is done by first selecting the personnel name to be deleted and then by clicking the delete button as shown in Figure 4.9 below:

**Personel Listesi**  
Firmanıza personel ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni personel kayıt için ekle tuşunu kullanın, ilgili alanları doldurduktan sonra kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

**Personel Bilgisini Sil**

Adı : HAKAN Kart No : 33 Kullanıcı Adı : HAKAN Sil  
Soyadı : AKKUZZU Sistem Yöneticisi Şifre : hakan İptal

Personel No	Personel Adı	Personel Soyadı	Kullanıcı Adı	Şifre	Sistem Yöneticisi
33	HAKAN	AKKUZZU	HAKAN	hakan	<input checked="" type="checkbox"/>
1	Ali	Özgür	ALI	ali	<input type="checkbox"/>
102	Kaan	Aslan	KAAN	kaan	<input type="checkbox"/>
203	Mustafa	Tuncay	MUSTAFA	mustafa	<input type="checkbox"/>
302	Hüseyin	Karahüseyin	HÜSEYİN	hüseyin	<input type="checkbox"/>

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.9** Delete personnel information

## 4.9 User Rights

All users in the system can be authorized by the following rights: View, add, update, delete and take print out. User rights can be defined for each form in the program. First select user form “personnel search dialog” form then set the rights for the user and save them.

To give permission to users use the check boxes in each colon.

Kart No	Adı	Soyadı	Kullanıcı Adı
1	Ali	Özgür	ALI
33	HAKAN	AKKUZZU	HAKAN
302	Hüseyin	Karahüseyin	HÜSEYİN
102	Kaan	Aslan	KAAN
203	Mustafa	Tuncay	MUSTAFA

Figure 4.10 Personnel search dialog form

Tanım	Kayıt	Değiştirme	Silme	Yazdırma	
Üretim					10
Planlama	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	11
Üretim Listesi	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	12
Raporlar	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	13
Veri Düzenleme					20
Ürünler	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	21
Reçeteler	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	22
Stoklar	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	23
Stok Hareketleri	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	24
Ayarlar					30
Ön Kesme Değerleri	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	31
Dolum Parametreleri	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	32
Agrega Dolum Sırası	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	33
Personel / Kullanıcı					40
Personel Bilgileri	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	41

Figure 4.11 Personnel rights

## 4.10 Stocks

Stock which are use for the production of concrete and other stock that the firm have can be described on the form shown in Figure 4.12.

**Stok Listesi**  
Firmanıza stok ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni stok kaydetmek için ekle tuşunu kullanın , ilgili alanları durup ardından kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

Stok Kod	Stok Adı	Miktar	Birim
00001	Agrega 01	10.000,000	KG
00002	Agrega 02	10.000,000	KG
00003	Agrega 03	10.000,000	KG
00004	Kum	10.000,000	KG
00005	Çimento	10.000,000	KG

5

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.12** Stock Main Form

Adding new stock can be done using the form shown in Figure 4.13. In this form stock is added by describing the stock number, description, quantity, and the unit.

**Stok Listesi**  
Firmanıza stok ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni stok kaydetmek için ekle tuşunu kullanın , ilgili alanları durup ardından kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

**Yeni Stok Ekle**

Stok No : 00017 Ölçü Birimi : Adet ☒ Kaydet

Tanım : Miktar : 0,000 ☒ İptal

Stok Kod	Stok Adı	Miktar	Birim
00001	Agrega 01	10.000,000	KG
00002	Agrega 02	10.000,000	KG
00003	Agrega 03	10.000,000	KG
00004	Kum	10.000,000	KG
00005	Çimento	10.000,000	KG

5

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.13** Adding A Stock To List.

Stock data can be updated using the form shown in Figure 4.14. The stock number of the stock whose details are to be updated is entered and then the details of the stock are updated. Clicking the update button saves the information in the database.

**Stok Listesi**  
Firmanıza stok ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni stok kaydetmek için ekle tuşunu kullanın , ilgili alanları durup ardından kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

**Stok Bilgilerini Değiştirme**

Stok No : 00001 Ölçü Birimi : KG Güncelle  
Tanım : Agregat 01 Miktar : 10.000,000 İptal

Stok Kod	Stok Adı	Miktar	Birim
00001	Agregat 01	10.000,000	KG
00002	Agregat 02	10.000,000	KG
00003	Agregat 03	10.000,000	KG
00004	Kum	10.000,000	KG
00005	Çimento	10.000,000	KG

5

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.14** Update Stock Information

Stock data can be deleted using the form shown in Figure 4.14. This is done by entering the stock number and then clicking the delete button.

**Stok Listesi**  
Firmanıza stok ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni stok kaydetmek için ekle tuşunu kullanın , ilgili alanları durup ardından kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

**Stok Bilgilerini Sil**

Stok No : 00003 Ölçü Birimi : KG Sil  
Tanım : Agregat 03 Miktar : 10.000,000 İptal

Stok Kod	Stok Adı	Miktar	Birim
00001	Agregat 01	10.000,000	KG
00002	Agregat 02	10.000,000	KG
00003	Agregat 03	10.000,000	KG
00004	Kum	10.000,000	KG
00005	Çimento	10.000,000	KG

5

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.15** Deleting stock information

## 4.11 Controlling the Stock In and Out Process

All stock in or out process can be saved without any limitation as shown in Figure 4.16. Information of the process has the following data: date, document number, definition and in or out amount.

**Stok Hareket Listesi**  
 Firmanızda stok ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
 Yeni stok hareketi kaydetmek için ekde tuşunu kullanarak ilgili alanları doldurduktan sonra kaydet tuşuna basınız.

F3 - Ara F2 - Liste F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

Stok Kod : 00001 Ürün Tanımı : Agregat 01 Stok Miktarı : 20.000,000 KG

Tarih Aralığı : 28.05.2006 28.05.2006

Tarih	Belge no	Açıklama	Stok Giriş	Stok Çıkış
Yıl : 2006				
Ay : 5				
28.05.2006	123456	Satın alma	10.000,000	0,000
			10.000,000	0,000
			10.000,000	0,000
			10.000,000	0,000

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.16** Stock in or out process list.

Before starting any process for a stock, one must select the stock from the list as shown in Figure 4.17.

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Stok Arama Formu**

Arama Türü : Kriter :  
 Ürün Adı : Ara

Kod	Ürün Tanım	Miktar	Birim
00001	Agregat 01	10000	KG
00002	Agregat 02	10000	KG
00003	Agregat 03	10000	KG
00004	Kum	10000	KG
00005	Çimento	10000	KG

Tamam İptal

**Figure 4.17** Stock Search Dialog Form.

In or out process of a stock can be added via this form showing in Figure 4.18. First define the type of process which can be stock in or stock out . The required data is: date of operation, document number (if there is), definition and amount.



**Stok Hareket Listesi**  
 Firma stok ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
 Yeni stok hareketi kaydetmek için ekle tuşunu kullanın, ilgili alanları durup ardından kaydet tuşuna basınız.

F3 - Ara F2 - Listele F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

Stok Kod : 00001 Ürün Tanımı : Agrega 01 Stok Miktarı : 10.000,000 KG

Tarih Aralığı : 28.05.2006 28.05.2006

İşlem Türü : Stok Giriş

Tarih : 28.05.2006 Açıklama : Kaydet

Belge No : 0 Miktar : 0,000 İptal

Tarih	Belge no	Açıklama	Stok Giriş	Stok Çıkış
<No data to display>				

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.18** Add a Stock in or out process .

In or out process of a stock can be edited by the form shown in Figure 4.19. Select the process to edit that changes the values required. Date of operation, document number (if there is one), definition and amount values are editable fields. After all done, click update button to make the changes permanent.

**Stok Hareket Listesi**  
 Firma stok ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
 Yeni stok hareketi kaydetmek için ekle tuşunu kullanın, ilgili alanları durup ardından kaydet tuşuna basınız.

F3 - Ara F2 - Listele F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

Stok Kod : 00001 Ürün Tanımı : Agrega 01 Stok Miktarı : 20.000,000 KG

Tarih Aralığı : 28.05.2006 28.05.2006

İşlem Türü : Stok Giriş

Tarih : 28.05.2006 Açıklama : Satın alma Güncelle

Belge No : 123456 Miktar : 10.000,000 İptal

Tarih	Belge no	Açıklama	Stok Giriş	Stok Çıkış
Yıl : 2006				
Ay : 5				
28.05.2006	123456	Satın alma	10.000,000	0,000
			10.000,000	0,000
			10.000,000	0,000
			10.000,000	0,000

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.19** Edit Stock In Or Out Process

In or out process of a stock can be deleted by the form shown in Figure 4.20. Select the process to delete then click delete button to delete record.

**Stok Hareket Listesi**  
 Firmanıza stok ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
 Yeni stok hareketi kaydetmek için ekde tuşunu kullanın, ilgili alanları doldurup ardından kaydet tuşuna basınız.

F3 - Ara F2 - Listele F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

Stok Kod : 00001 Ürün Tanımı : Agrega 01 Stok Miktarı : 20.000,000 KG

Tarih Aralığı : 28.05.2006 28.05.2006

İşlem Türü : Stok Giriş

Tarih : 28.05.2006 Açıklama : Satın alma

Belge No : 123456 Miktar : 10.000,000

✓ Sil  
✗ İptal

Tarih	Belge no	Açıklama	Stok Giriş	Stok Çıkış
Yıl : 2006				
Ay : 5				
28.05.2006	123456	Satın alma	10.000,000	0,000
			10.000,000	0,000
			10.000,000	0,000
			10.000,000	0,000

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.20 Delete Stock In Or Out Process .

## 4.12 Products

The products which are manufactured by the firm will be listed via this form shown in Figure 4.21. There is no limitation to the product record. Products have Spekt numbers which are given by the system automatically. In ISO standards all products must have unique numbers.

**Üretilen Ürünlerin Listesi**  
 Firmanız tarafından üretilen ürünleri ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
 Yeni ürün kaydetmek için ekde tuşunu kullanın, ilgili alanları doldurup ardından kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F10 - Kapat

Spekt No	Ürün Kod	Ürün Adı	Ürün Tipi
SPK00012	URN0001	Ürün 01	Ekonomik
SPK00013	URN0002	Ürün 02	Standart
SPK00014	URN0003	Ürün 03	Standart

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.21 List of Products .



Figure 4.22 shows the form used to add a new product. Product information has the following information: code, name, product type and unique spekt number.

**Üretilen Ürünlerin Listesi**  
Firmanız tarafından üretilen ürünleri ekleyebilir, mevcut bilgileri güncelleyip silabilirsiniz.  
Yeni ürün kaydetmek için ekle tuşunu kullanın, ilgili alanları doldurup ardından kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F10 - Kapat

**Yeni Ürün Kayıt**

Kodu :  Ürün Tipi :  ☒ Kaydet  
 Ürün Adı :  Spekt No :  ☒ İptal

Spekt No	Ürün Kod	Ürün Adı	Ürün Tipi
SPK00012	URN0001	Ürün 01	Ekonomik
SPK00013	URN0002	Ürün 02	Standart
SPK00014	URN0003	Ürün 03	Standart

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.22** Add New Product .

Product informatoin can be changed any time by the form in Figure 4.23. Select the product from list, make changes then press update button to apply change.

**Üretilen Ürünlerin Listesi**  
Firmanız tarafından üretilen ürünleri ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni ürün kaydetmek için ekle tuşunu kullanın, ilgili alanları doldurup ardından kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F10 - Kapat

**Ürün Bilgilerini Değiştir**

Kodu :  Ürün Tipi :  ☒ Güncelle  
 Ürün Adı :  Spekt No :  ☒ İptal

Spekt No	Ürün Kod	Ürün Adı	Ürün Tipi
SPK00012	URN0001	Ürün 01	Ekonomik
SPK00013	URN0002	Ürün 02	Standart
SPK00014	URN0003	Ürün 03	Standart

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.23** Update Product Information.

Product information can be deleted any time by the form in Figure 4.24. Select the product from the list to delete then press delete button to remove from the list.

**Üretilen Ürünlerin Listesi**

Firmanız tarafından üretilen ürünleri ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni ürün kaydetmek için ekle tuşunu kullanın, ilgili alanları doldurup ardından kaydet tuşuna basınız.

F4 - Ekle   F5 - Değiştir   F6 - Sil   F10 - Kapat

---

**Ürün Bilgilerini Sil**

Kodu :    Ürün Tipi :    ☒ Sil  
 Ürün Adı :    Spekt No :    ☒ İptal

Spekt No	Ürün Kod	Ürün Adı	Ürün Tipi
SPK00012	URN0001	Ürün 01	Ekonomik
SPK00013	URN0002	Ürün 02	Standart
SPK00014	URN0003	Ürün 03	Standart

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.24** Delete Product Information.

### 4.13 Product Receipts

In ISO standards every product manufacturing must have at least one or more receipt. A product has lost of matters and these are mixed in unlimited rates. Every changes of rate cause a new receipts. Program has the ability to save these changes.

**Üretilen Ürünler ve Onlara Ait Reçetelerin Listesi**

Ürün için reçete ekleyebilir, reçete bilgilerini değiştirip silebilirsiniz.  
Yeni reçete kaydetmek için ekle tuşunu kullanın. Reçete bilgilerini incelemek için detay tuşunu kullanın.

F2 - Detay   F4 - Ekle   F5 - Değiştir   F6 - Sil   F8 - Yazdır   F10 - Kapat

Ürün Kodu :    Ürün Tipi :   

Ürün Adı :    Spekt No :

**Ürüne Ait Reçete Listesi**

Reçete Kod	Reçete Adı	Kayıt Tarihi	Agrega 01	Agrega 02	Agrega 03	Kum	Çimento	Su	Katkı 01	Katkı 02
RCT0001	Reçete 01 (URN0001)	28.05.2006	100,000 Kg	100,000 Kg	100,000 Kg	100,000 Kg	100,000 Kg	200,000 Kg	0,000 Kg	0,000 Kg
RCT0002	Reçete 02 (URN0001)	28.05.2006	120,000 Kg	120,000 Kg	120,000 Kg	120,000 Kg	120,000 Kg	180,000 Kg	0,000 Kg	0,000 Kg

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.25** List Of Product Receipts.

Products must have a receipt that is mentioned in previous section. With the form shown in Figure 4.26, a receipt for a product can be described. Give code and name to the receipt form and the publication date. Define the other values, then select the person who has prepared and the person who has approved this receipt. Press save button to make these datas be saved in disk.

**Yeni Ürün Reçetesi Ekle**  
Ürün reçetesi eklemek için reçete adı, yayın tarihini tanımladıktan sonra;Agrega miktarlarını ve diğer bilgileri tanımlayınız.Hazırlayan ve onaylayan personeli seçtikten sonra kayıt için kaydet tuşunu kullanın.

[F4] Kaydet [ESC] İptal

Ürün Bilgileri		Reçete Bilgileri	
Ürün Kodu :	URN0001	Reçete Kodu :	
Ürün Adı :	Ürün 01	Reçete Adı :	
Ürün Tipi :	Ekonomik	Yayın Tarihi :	28.05.2006
Spekt No :	SPK00012		

Agrega Miktarları		Diğer Bilgiler	
01 No`lu Agregası :	0,000 Kg	Çimento Miktarı :	0,000 Kg
02 No`lu Agregası :	0,000 Kg	Su Miktarı :	0,000 Kg
03 No`lu Agregası :	0,000 Kg	Katkı - 1 Miktarı :	0,000 Kg
Kum Miktarı :	0,000 Kg	Katkı - 2 Miktarı :	0,000 Kg
		Karıştırma Süresi :	0 Saat 0 Dk

Hazırlayan :		Onaylayan :	
Adı :		Adı :	
Soyadı :		Soyadı :	

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.26** Add A Product Receipts.

A receipt for a product can be updated in any time with the form shown in Figure 4.27. Select a receipt form list ; Give code and name to the receipt and publication date. Define the other values, then select the person who is prepared (make the changes) and the person who is approved this this receipt. Press update button to apply changes for the receipt.

**Ürün Reçete Bilgisini Değiştirme**  
 Ürün reçetesini değiştirmek için agrega miktarlarını ve diğer bilgileri tanımlayınız.  
 Hazırlayan ve onaylayan personeli seçtikten sonra, değiştirme için güncelle tuşunu kullanın.

[F5] Güncelle [ESC] İptal

<b>Ürün Bilgileri</b> Ürün Kodu : URN0001 Ürün Adı : Ürün 01 Ürün Tipi : Ekonomik Spekt No : SPK00012	<b>Reçete Bilgileri</b> Reçete Kodu : RCT0002 Reçete Adı : Reçete 0002 (URN0001) Yayın Tarihi : 28.05.2006 Revizyon No :
<b>Agrega Miktarları</b> 01 No`lu Agregası : 120,000 Kg 02 No`lu Agregası : 120,000 Kg 03 No`lu Agregası : 120,000 Kg Kum Miktarı : 120,000 Kg	<b>Diğer Bilgiler</b> Çimento Miktarı : 120,000 Kg Su Miktarı : 180,000 Kg Katkı - 1 Miktarı : 0,000 Kg Katkı - 2 Miktarı : 0,000 Kg Karıştırma Süresi : 0 Saat 40 Dk
Değişiklik Nedeni :	
Hazırlayan : Adı : Soyadı :	Onaylayan : Adı : Soyadı :

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.27 Update Product Receipts.

A receipt for a product can be deleted at any time with the form shown in Figure 4.28. Select a receipt form list. Press delete button to remove this receipt from the list.

**Ürün Reçete Bilgisini Silme**  
 Ürün reçetesini silmek için sil tuşunu kullanın. Üretimi Yapılmış reçeteler silinemez.

[F6] Sil [ESC] İptal

<b>Ürün Bilgileri</b> Ürün Kodu : URN0001 Ürün Adı : Ürün 01 Ürün Tipi : Ekonomik Spekt No : SPK00012	<b>Reçete Bilgileri</b> Reçete Kodu : RCT0001 Reçete Adı : Reçete 01 (URN0001) Yayın Tarihi : 28.05.2006 Revizyon No :
<b>Agrega Miktarları</b> 01 No`lu Agregası : 100,000 Kg 02 No`lu Agregası : 100,000 Kg 03 No`lu Agregası : 100,000 Kg Kum Miktarı : 100,000 Kg	<b>Diğer Bilgiler</b> Çimento Miktarı : 100,000 Kg Su Miktarı : 200,000 Kg Katkı - 1 Miktarı : 0,000 Kg Katkı - 2 Miktarı : 0,000 Kg Karıştırma Süresi : 0 Saat 20 Dk

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.28 Delete Product Receipts.

## 4.14 Production

Planning is the first subject that we have to do before a production starts. The following forms show how a production is planned:

Üretim No	Ürün Adı	Reçete Kodu	Reçete Adı	R.Std.Miktar	Tekrar	Planlanan Batch	Toplam	Miktar	Gerçekleşen Batch	Tekrar	Tarih	Saat
UR00001	Ürün 01	RCT0001	Reçete 01 (URN0001)	700,000 Kg	3	2.100,000 Kg	2.100,000 Kg	3	29.01.2007	20:12		

Figure 4.29 Production List.

Ürün Adı	Reçete Kodu	Reçete Adı	R.Std.Miktar
Ürün 01	RCT0001	Reçete 01 (URN0001)	70

Figure 4.30 Define the criterion of Production List.



**Hazır Beton Üretim Emri Ekle**  
 Üretim emri eklemek için ürün seçimi yaptıktan sonra; ürüne ait spektlerden birini seçiniz. Tekrar sayısını, üretim tarihi ve saatini tanımlayıp kayıt için kaydet tuşunu kullanın.

Üretim No : URT00002 [F4] Kaydet [ESC] İptal

**Ürün Bilgileri :**  
 Kodu : URN0003  
 Ürün Adı : Ürün 03  
 Ürün Tipi : Standart  
 Spekt No : SPK00014

**Üretim Bilgileri**  
 Reçete Kod :  
 Reçete Adı :  
 Tekrar Sayısı : 0  
 Üretim Tarihi : 31.01.2007  
 Üretim Saati : 21:21:10

**Ürüne Ait Reçete Bilgileri**

Kayıt Tarihi	SpektNo	ReçeteKod	ReçeteAdi	Agrega01	Agrega02	Agrega03	Kum	Cimento
<No data to display>								

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.31** Create a New Production

**Hazır Beton Üretim Emri Değiştir**  
 Üretim emrini değiştirmek için ürün seçimi yaptıktan sonra; ürüne ait spektlerden birini seçiniz. Tekrar sayısını, üretim tarihi ve saatini tanımlayıp güncelle tuşuyla üretim emrini değiştiriniz.

Üretim No : URT00001 [F5] Güncelle [ESC] İptal

**Ürün Bilgileri :**  
 Kodu : URN0001  
 Ürün Adı : Ürün 01  
 Ürün Tipi : Ekonomik  
 Spekt No : SPK00012

**Üretim Bilgileri**  
 Reçete Kod : RCT0001  
 Reçete Adı : Reçete 01 (URN0001)  
 Tekrar Sayısı : 3  
 Üretim Tarihi : 29.01.2007  
 Üretim Saati : 20:12:00

**Ürüne Ait Reçete Bilgileri**

Kayıt Tarihi	SpektNo	ReçeteKod	ReçeteAdi	Agrega01	Agrega02	Agrega03	Kum	Cimen
28.05.2006	SPK00012	RCT0001	Reçete 01 (URN0001)	100,000 Kg	100,000 Kg	100,000 Kg	100,000 Kg	100
28.05.2006	SPK00012	RCT0002	Reçete 02 (URN0001)	120,000 Kg	120,000 Kg	120,000 Kg	120,000 Kg	120

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

**Figure 4.32** Editing the Production Details

**Hazır Beton Üretim Emri Sil**  
Sil tuşuyla üretim emrini plandan silebilirsiniz.

Üretim No : URT00001

[F6] Sil [ESC] İptal

Ürün Bilgileri :	Üretim Bilgileri
Kodu : URN0001	Reçete Kod : RCT0001
Ürün Adı : Ürün 01	Reçete Adı : Reçete 01 (URN0001)
Ürün Tipi : Ekonomik	Tekrar Sayısı : 3
Spekt No : SPK00012	Üretim Tarihi : 29.01.2007 15
	Üretim Saati : 20:12:00

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.33 Delete then Production

**Üretimi Başlat**

Ürün Bilgileri :	Üretim Bilgileri
Kodu : URN0001	Reçete Kod : RCT0002
Ürün Adı : Ürün 01	Reçete Adı : Reçete 02 (URN0001)
Ürün Tipi : Ekonomik	Üretim Tarihi : 31.01.2007 15
Spekt No : SPK00012	Üretim Saati : 21:30:41

Üretim No : URT00002

[F4] Başlat [ESC] Kapat

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.34 Start Production

## 4.15 Customers

The following forms are used to create, update, edit, and delete customer details:

**Cari Kartlar Listesi**  
 Cari Kart ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
 Yeni kart kaydetmek için ekle tuşunu kullanın, ilgili alanları durup ardından kaydet tuşuna basınız.

F4 - Ekle F5 - Değiştir F6 - Sil F8 - Yazdır F10 - Kapat

Arama Türü : T.Ünvan/Ad-Soyad Kriter : F3 - Ara

Cari Kod	Adı Soyadı/Ticari Ünvan	Borç	Alacak	Bakiye
00001	Cari Kart 01	1.200,00	800,00	400,00

1.200,00 800,00 400,00

**C.Hesap Bilgileri** **İletişim Bilgileri**

Adres : Lefkoşa Vergi Dairesi :  
 VD. No :  
 Semt/Bölge/Köy : Kayıt Tarihi : 28.05.2006  
 Şehir :

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.35 Customer List

**Cari Hesap Kartı Ekleme**  
 Yeni cari kart eklemek için gerekli bilgileri doldurduktan sonra  
 kayıt için kaydet tuşunu kullanın.

[F4] Kaydet [ESC] İptal

**C.Hesap Bilgileri** **İletişim Bilgileri**

Cari Hesap Kod : 00002  
 Ticari Ünvan :  
 Adres :  
 Semt/Bölge/Köy :  
 Şehir :  
 Vergi Dairesi :  
 VD. No :  
 Kayıt Tarihi : 31.01.2007  
 Borç : 0,00 YTL  
 Alacak : 0,00 YTL

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.36 Add A New Customer



**Cari Kart Bilgisini Deęiřtirme**  
Ürün reçetesini deęiřtirmek için agrega miktarlarını ve dięer bilgileri tanımlayınız.  
Hazırlayan ve onaylayan personeli seçtikten sonra kayıt için kaydet tuşunu kullanın.

[F5] Güncelle [ESC] İptal

**C.Hesap Bilgileri** **İletişim Bilgileri**

Cari Hesap Kod : 00001  
Ticari Ünvan : Cari Kart 01  
Adres : Lefkoşa  
Semt/Bölge/Köy :  
Şehir :  
Vergi Dairesi :  
VD. No :  
Kayıt Tarihi : 28.05.2006  
Borç : 1.200,00 YTL  
Alacak : 800,00 YTL

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.37 Update Customer Details

**Cari Hesap Kart Silme**  
Cari kart bilgilerini silmek için Sil tuşunu kullanın.

[F6] Sil [ESC] İptal

**C.Hesap Bilgileri** **İletişim Bilgileri**

Cari Hesap Kod : 00001  
Ticari Ünvan : Cari Kart 01  
Adres : Lefkoşa  
Semt/Bölge/Köy :  
Şehir :  
Vergi Dairesi :  
VD. No :  
Kayıt Tarihi : 28.05.2006  
Borç : 1.200,00 YTL  
Alacak : 800,00 YTL

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.38 Delete Customer Details

**Cari Kart Hareket Listesi**  
Cari kart hareketi ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni satış eklemek için satış tuşunu kullanın. Yeni ödeme eklemek için ödeme tuşunu kullanın.

F3 - Arama F2 - Listele Satış Ödeme İşlemler F8 - Yazdır F10 - Kapat

Cari Kod : 00001 Ad Soyad / T. Ünvan : Cari Kart 01 Bakiye : 400,00 YTL

Kriter : Tümü Tarih Aralığı :

Tarih	Belge No	Açıklama	Borç	Alacak
yıl : 2006				
ay : 5				
28.05.2006	#	Hesap Açılış Değerleri	0,00	0,00
28.05.2006	1233	satış (Urn0001)	1.200,00	0,00
28.05.2006	2233	Ödeme	0,00	800,00
			1.200,00	800,00
			1.200,00	800,00

1.200,00 800,00

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.39 Accountng Details

**Cari Kart Hareket Listesi**  
Cari kart hareketi ekleyebilir, mevcut bilgileri güncelleyip silebilirsiniz.  
Yeni satış eklemek için satış tuşunu kullanın. Yeni ödeme eklemek için ödeme tuşunu kullanın.

F3 - Arama F2 - Listele Satış Ödeme İşlemler F8 - Yazdır F10 - Kapat

Cari Kod : 00001 Ad Soyad / T. Ünvan : Cari Kart 01 Bakiye : 400,00 YTL

Kriter : Tümü Tarih Aralığı :

**Cari Kart Satış**

Tarih : 1.01.2007 Açıklama : Kaydet  
Belge No : Miktar : 0,00 İptal

Tarih	Belge No	Açıklama	Borç	Alacak
yıl : 2006				
ay : 5				
28.05.2006	#	Hesap Açılış Değerleri	0,00	0,00
28.05.2006	1233	satış (Urn0001)	1.200,00	0,00
28.05.2006	2233	Ödeme	0,00	800,00
			1.200,00	800,00
			1.200,00	800,00

1.200,00 800,00

© 2006 Hazır Beton Otomasyon Programı Ver. 1.0.0

Figure 4.40 Sale Details

## 4.16 Efficiency of Application of Computer Based concrete Production Control

Concrete production process is a complex decimal process. In the part of this process has been implemented using manual techniques where a plant operator is responsible for the complete production phases of the process. The operator used to control manual buttons to start and stop the various phases of the production. For example, pressing a button started the mixing process and so on.

The next generation of the concrete production process was implemented using PLC's. PLC based process automation has the advantage that the plant can be controlled automatically with little manual interaction. The biggest disadvantage of the PLC based system is that it is required to set-up the system parameters every time when a new production starts, and this can be very very time consuming.

The concrete production system described in this thesis described a computer based plant control. The system has many advantages over the manual and PLC based systems;

- Computer based system is very low cost ... the PLC and manual systems
- The production parameters can be stored on the PC and this makes the production an easy process.
- Special experience on training are not required to operate the plant using a computer based system.
- The system can keep a stock control of the material used in the production.
- Customer details can be kept on the computer and thus it is easy to keep track of the customer details.
- The overall system can easily be tuned using the computer based system. This makes the production a very quick process, taking no more than a few seconds to set-up and start the production.
- In PLC based systems the production was in two stages: First the production was planned and then the actual production was implemented by adjusting the parameters of the PLC. In computer based system the planning and production are carried out at the same time using the same computer. This approach makes the overall design process much more efficient, and user friendly.

## CONCLUSIONS

Traditionally, the control and management of concrete production is based on PLC technology. In such applications, many processes are carried out manually where the chemical materials such as cement, water, and rocks are mixed in order to produce concrete. The manual process has many disadvantages such as the high cost, difficulty to control the process, difficulty to obtain consistently good quality concrete and so on. The control of parameters of concrete production process on the base of PLC technology needs certain time for setting optimal values of parameters. The values of these parameters for different output products are different. To speed up these processes and improve quality of the control system, a microcontroller based control system is presented in this thesis for concrete production.

The system is based on a PC and a microcontroller based interface card. A Delphi program has been developed for the PC. The program receives the concrete production parameters and the customer details and then sends RS232 based serial signals to the interface card in order to control the concrete production plant. Most concrete production systems are based on expensive and complex PLCs, but the system developed by the author is based on a simple low-cost microcontroller card with. The advantages of the system developed by the author are:

- The system is based on a standard low-cost microcontroller which can easily be upgraded
- System architecture is simple and thus the maintenance is easy
- The overall system cost is very low compared to the commercially available systems

- Special training (e.g. PLC programming) is not required to install and operate the system
- The preparation of the necessary material, such as aggregate, cement, and water takes about 10 to 15 minutes in PLC based systems. In the system developed by the author, these activities can all be performed in less than a minute. In addition these values can be saved and then used in the future in similar applications.
- The cost of the concrete production control system shown in Figure 2.2 is \$20,000. The cost of the basic control system developed by the author is only around \$3,000, and the huge cost saving is obvious.
- Because of the complete production process is saved by the system it is possible to calculate the performance of the plant easily.

In addition, the concrete automation system developed in this thesis has the following advantages: high quality concrete can be obtained as the chemical ingredients can be controlled accurately, always consistent, and same quality product can be obtained, the cost of the product is lower since the product can be produced under optimum conditions. In addition, the automation system developed by the author has the advantage that the system can be operated without the intervention of any operator. Thus, concrete of any amount and also at any time can be produced by simply using the program developed by the author.

The automation system developed by the author can be developed further by adding some security and safety features. For example, automatic safety shutdown can be provided such that the plant shuts down automatically when an emergency occurs.

## REFERENCES

1. MAST Materials Science and Technology Teacher's Workshop  
Prepared by the Department of Materials Science and Engineering  
University of Illinois Urbana-Champaign , 2005
2. Reinforced Concrete, Preliminary design for Architects and Builders; R.E.  
Schaeffer, 1992 McGraw-Hill Inc.
3. Handout on Skyscrapers World Book Encyclopedia <http://www.portcement.org/indhist.htm>
4. Modern Industrial Automation  
Software Design Principles and Real- World Applications  
By Lingfeng Wang ,Kay Chen Tan  
Copyright 2006, by The Institute of Electrical and Electronics Engineers, Inc  
Published by John Wiley & Sons, Inc., Hoboken, New Jersey
5. MASTERING™ DELPHI™ 6 by Marco Cantù ([www.marcocantu.com](http://www.marcocantu.com))  
Published by SYBEX Inc. <http://www.sybex.com>
6. ESIT LTD. <http://www.esit.com.tr>
7. Foks Automation <http://www.foks.com.tr>
8. BHAIR Engineering Corporation <http://www.bhairengineering.com>
9. A & D Company LTD. <http://www.aandd.jp>
10. ScaleTron Ltd <http://scaletron.com>
11. Batching and Mixing Plants for Concrete ,The Indian Concrete Journal June 2002
12. dee Concrete Accessories 7350 W. Montrose Avenue • Norridge, IL 60706  
<http://www.deeconcrete.com/index.html>
13. Right Manufacturing Systems 2DH-4CL 4 Yard Towable concrete production  
system [www.constructioncomplete.com](http://www.constructioncomplete.com)
14. The RS232 Standard (renamed the "EIA232 Standard" in the early 1990's)  
by Christopher E. Strangio 1993-2006 by CAMI Research Inc., Acton,  
Massachusetts

15. The RS232 COMMS Serial Interface <http://www.datataker.com>
16. RS232 Data Interface a Tutorial on Data Interface and cables ARC Electronics  
<http://www.arcelect.com>
17. Borland Delphi Enterprise Version 7.0 (Build 4.453) 1983-2003 Borland Software Corporation <http://www.borland.com>
18. International Organization for Standardization  
<http://www.iso.org/iso/en/aboutiso/introduction/index.html>
19. BASCOM-8051 © 1995-2006 MCS Electronics <http://www.bascom.com>
20. Paul's 8051 Tools, Projects and Free Code Offerings; Paul Stoffregen  
<http://www.pjrc.com/tech/8051/index.html>
21. AT89C2051 Programmer Copyright © 1996 Silicon Studio Ltd.
20. Atmel Corporation <http://www.atmel.com>
21. Meka Concrete Batching Plants [www.meka.com.tr](http://www.meka.com.tr)

## APPENDIX A

### Installing BASCOM-8051 Compiler

The interface card used in this thesis uses the BASCOM BASIC compiler. This appendix describes how to install and use this compiler.

After the software is downloaded you should unzip it. There is only one file named setup.exe. You may run this setup.exe from within the Windows Shell but it is important to notice that when you use the commercial version, you **MUST UNZIP** the setup.exe since you need to copy the license file to the same directory as setup.exe.



One must have Administrator rights in order to be able to run setup.

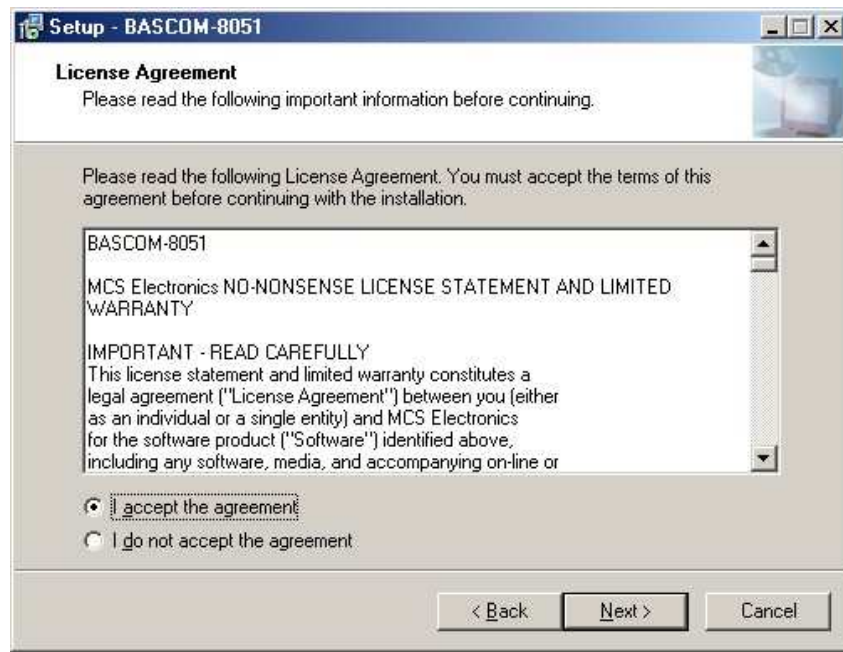
The opening screen looks like :



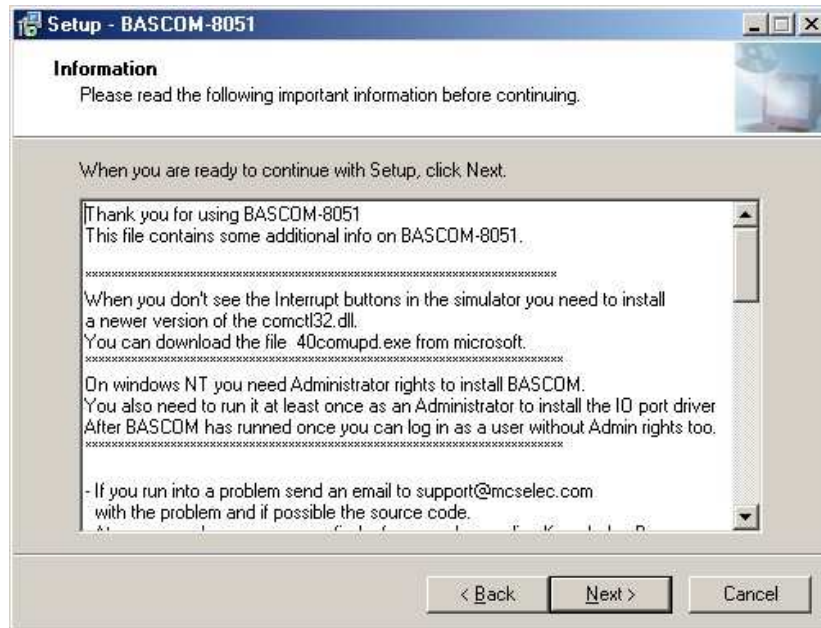
You need to click the Next-button to continue.



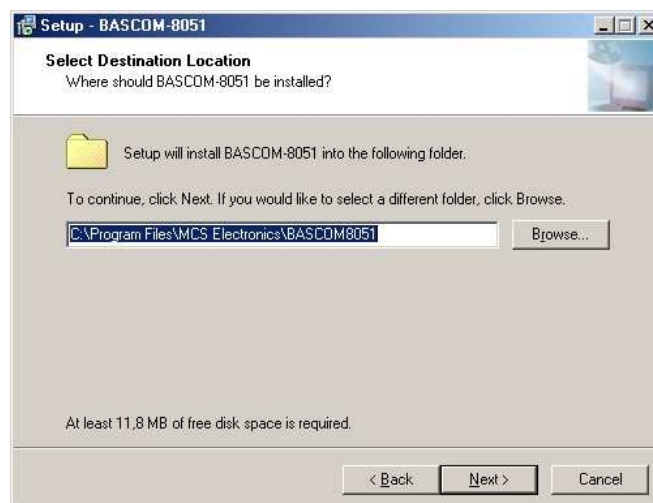
A license agreement will be shown. You need to read it and accept the agreement. This is a no-nonsense agreement where you are allowed to install/copy on as many computers as you want, providing that you use only one computer at the same time.



After clicking the 'I accept the agreement' option, you need to click the Next-button again to continue. The readme.txt file is shown. Basicly it tells you to contact [support@mcselec.com](mailto:support@mcselec.com) in case of a problem.

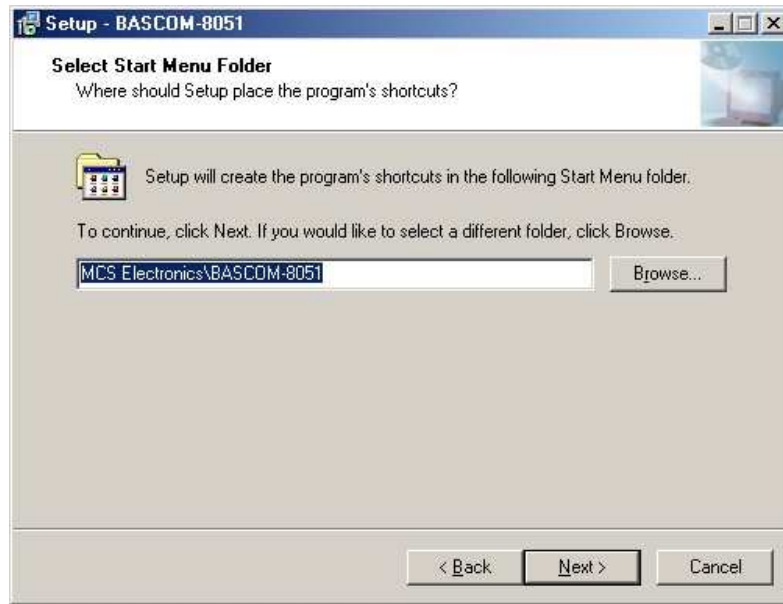


Click the Next-button again to continue with the setup. You can now select where you want to Install BASCOM-8051. The default is shown below.

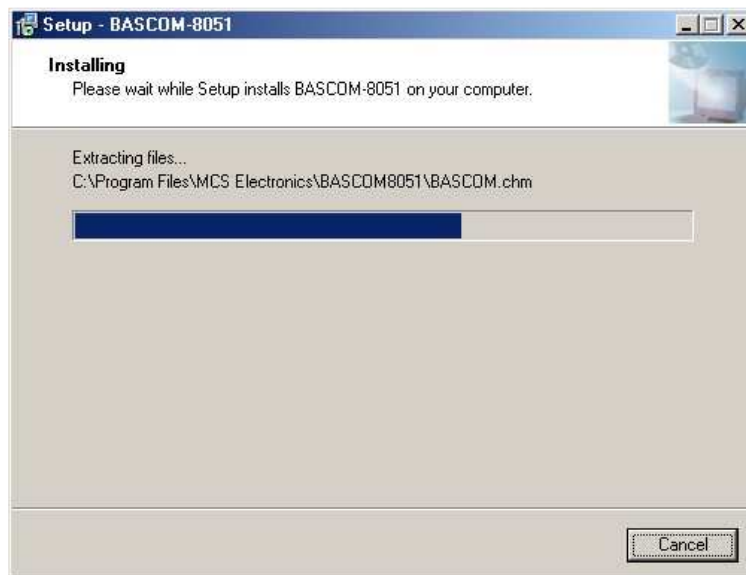


Click the Next-button again to continue.

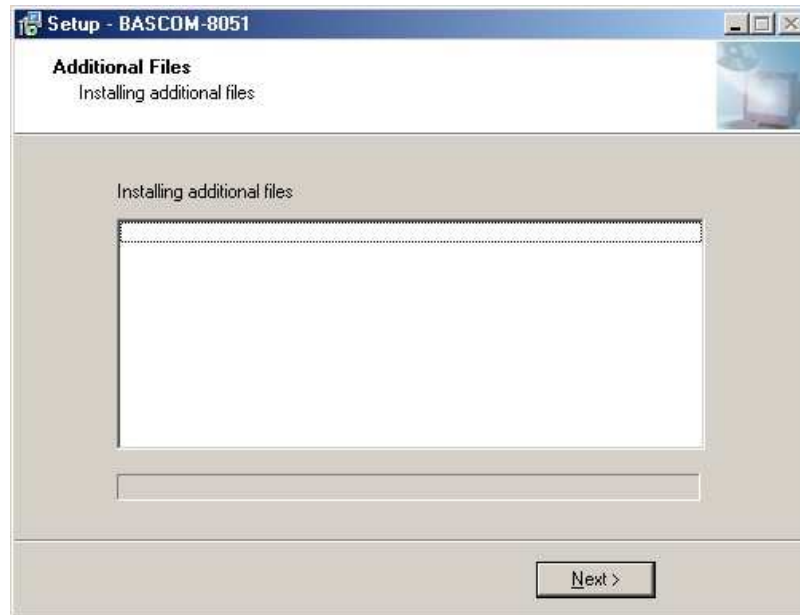
You can now select/enter the Program Group name. The default is shown below.



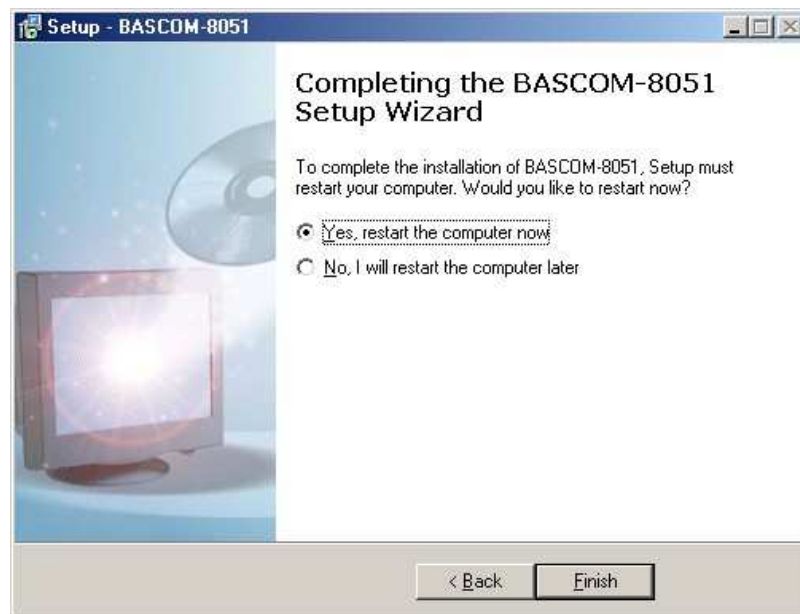
Click the Next-button again to continue. The files will now be installed. A screenshot is shown below :



When the files are installed, the installer will install some additional files :



Press the Next-Button to install the additional files. When you install from CD-ROM the setup will also copy PDF datasheets. The installation will take longer then. When setup is completely finished it will show the last screen :



You MUST reboot your PC since it will install a driver needed for the programming.

## BASCOM – 8051 Language Fundamentals

Characters from the BASCOM character set are put together to form labels, keywords, variables and operators. These in turn combine to form statements that make up a program.

This appendix describes the character set and the format of BASCOM program lines. In particular, it discusses the specific characters in the character set and the special meanings of some characters. The format of a line in a BASCOM program.

Line labels.

Program line length.

Character Set

The BASCOM BASIC character set consists of alphabetic characters, numeric characters, and special characters. The alphabetic characters in BASCOM are the uppercase letters (A-Z) and lowercase letters (az) of the alphabet.

The BASCOM numeric characters are the digits 0-9.

The letters can be used as parts of hexadecimal numbers.

The following characters have special meanings in BASCOM statements and expressions:

Character	Description
ENTER	Terminates input of a line
	Blank ( or space)
'	Single quotation mark (apostrophe)
*	Asterisks (multiplication symbol)
+	Plus sign
,	Comma
-	Minus sign
.	Period (decimal point)

/	Slash (division symbol) will be handled as \
:	Colon
"	Double quotation mark
;	Semicolon
<	Less than
=	Equal sign (assignment symbol or relational operator)
>	Greater than
\	Backslash (integer/word division symbol)

## The BASCOM program line

BASCOM program lines have the following syntax:

```
[[line-identifier]] [[statement]] [[:statement]] ... [[comment]]
```

## Using Line Identifiers

BASCOM support one type of line-identifier; alphanumeric line labels:

An alphanumeric line label may be any combination of from 1 to 32 letters and digits, starting with a letter and ending with a colon.

BASCOM keywords are not permitted. The following are valid alphanumeric line labels:

Alpha:

ScreenSUB:

Test3A:

Case is not significant. The following line labels are equivalent:

alpha:

Alpha:

ALPHA:

Line labels may begin in any column, as long as they are the first characters other than blanks on the line. Blanks are not allowed between an alphabetic label and the colon following it. A line can have only one label.

## **BASCOM Statements**

A BASCOM statement is either "executable" or "nonexecutable". An executable statement advances the flow of a programs logic by telling the program what tot do next. Non executable statement perform tasks such as allocating storage for variables, declaring and defining variable types. The following BASCOM statements are examples of nonexecutable statements:

```
REM or (starts a comment)
DIM
```

A "comment" is a nonexecutable statement used to clarify a programs operation and purpose. A comment is introduced by the REM statement or a single quote character(').

The following lines are equivalent:

```
PRINT " Quantity remaining" : REM Print report label.
PRINT " Quantity remaining" ' Print report label.
```

More than one BASCOM statement can be placed on a line, but colons(:) must separate statements, as illustrated below.

```
FOR I = 1 TO 5 : PRINT " Gday, mate." : NEXT I
```

## **BASCOM Line Length**

If you enter your programs using the built-in editor, you are not limited to any line length, although it is advised to shorten your lines to 80 characters for clarity.

## Data Types

Every variable in BASCOM has a data type that determines what can be stored in the variable. The next section summarizes the elementary data types.

### Elementary Data Types

Bit (1/8 byte)

Byte (1 byte)

Bytes are stored as unsigned 8-bit binary numbers ranging in value from 0 to 255.

Integer (two bytes).

Integers are stored as signed sixteen-bit binary numbers ranging in value from -32,768 to +32,767.

Word (two bytes).

Words are stored as unsigned sixteen-bit binary numbers ranging in value from 0 to 65535.

Long (four bytes).

Longs are stored as signed 32-bit binary numbers ranging in value from -2147483648 to 2147483647.

Single

Singles are stored as signed 32 bit binary numbers.

String (up to 254 bytes).

Strings are stored as bytes and are terminated with a 0-byte.

A string dimensioned with a length of 10 bytes will occupy 11 bytes.

Variables can be stored internal (default) or external.

### Variables

A variable is a name that refers to an object--a particular number.

A numeric variable can be assigned only a numeric value (either integer, word, byte long, single or bit). The following list shows some examples of variable assignments:

A constant value:

A = 5

C = 1.1

The value of another numeric variable:

abc = def

k = g



The value obtained by combining other variables, constants, and operators:

Temp = a + 5

Temp = C + 5

## Variable Names

A BASCOM variable name may contain up to 32 characters. The characters allowed in a variable name are letters and numbers. The first character in a variable name must be a letter.

A variable name cannot be a reserved word, but embedded reserved words are allowed.

For example, the following statement is illegal because AND is a reserved word.

AND = 8

However, the following statement is legal:

ToAND = 8

Reserved words include all BASCOM commands, statements, function names, internal registers and operator names.

(see BASCOM Reserved Words , for a complete list of reserved words).

One can specify a hexadecimal or binary number with the prefix &H or &B.

a = &HA , a = &B1010 and a = 10 are all the same.

Before assigning a variable you must tell the compiler about it with the DIM statement.

Dim b1 As Bit, I as Integer, k as Byte , s As String \* 10

One can also use DEFINT, DEFBIT, DEFBYTE and/or DEFWORD.

For example DEFINT c tells the compiler that all variables that are not dimensioned and that are beginning with the character c are of the Integer type.

## Expressions and Operators

An expression can be a numeric constant, a variable, or a single value obtained by combining constants, variables, and other expressions with operators.

Operators perform mathematical or logical operations on values. The operators provided by BASCOM can be divided into four categories, as follows:

1. Arithmetic
2. Relational
3. Logical
4. Functional

### **Arithmetic**

Arithmetic operators are +, -, \* and \.

Integer

Integer division is denoted by the backslash (\).

Example:  $Z = X \setminus Y$

Modulo Arithmetic

Modulo arithmetic is denoted by the modulus operator MOD.

Modulo arithmetic provides the remainder, rather than the quotient, of an integer division.

Example:  $X = 10 \setminus 4$  : remainder =  $10 \text{ MOD } 4$

Overflow and division by zero

Division by zero, produces an error.

At this moment there is no message, so you have to insure yourself that such won't happen.

### **Relational Operators**

Relational operators are used to compare two values as shown in the table below.

The result can be used to make a decision regarding program flow.

Operator	Relation Tested	Expression
=	Equality	$X = Y$
<>	Inequality	$X <> Y$
<	Less than	$X < Y$
>	Greater than	$X > Y$
<=	Less than or equal to	$X <= Y$
>=	Greater than or equal to	$X >= Y$

### **Logical Operators**

Logical operators perform tests on relations, bit manipulations, or Boolean operators.

There are four operators in BASCOM, they are :

Operator	Meaning
NOT	Logical complement
AND	Conjunction
OR	Disjunction
XOR	Exclusive or

It is possible to use logical operators to test bytes for a particular bit pattern. For example the AND operator can be used to mask all but one of the bits of a status byte, while OR can be used to merge two bytes to create a particular binary value.

Example

A = 63 And 19

PRINT A

A = 10 Or 9

PRINT A

Output

16

11

### **Floating point**

Single numbers conform to the IEEE binary floating point standard.

An eight-bit exponent and 24 bit mantissa are supported.

Using four bytes, the format is shown below:

31 30\_\_\_\_\_23 22\_\_\_\_\_0  
s exponent mantissa

The exponent is biased by 128. Above 128 are positive exponents and below are negative. The sign bit is 0 for positive numbers and 1 for negative. The mantissa is stored in hidden bit normalized format so that 24 bits of precision can be obtained.

All mathematical operations are supported by the single.

You can also convert a single to an integer or word or vice versa:

Dim I as Integer, S as Single

S = 100.1 'assign the single

I = S                                'will convert the single to an integer  
Take a look at the single.bas example for more information.

## **Arrays**

An array is a set of sequentially indexed elements having the same type. Each element of an array has a unique index number that identifies it. Changes made to an element of an array do not affect the other elements.

The index must be a numeric constant, a byte, an integer or a word. This means that an array can hold 65535 elements as a maximum. The minimum value is 1 and not zero as in QB.

Arrays can be used on each place where a 'normal' variable is expected but there are a few exceptions.

These exceptions are shown in the help topics.

Note that there are no BIT arrays in BASCOM-8051.

Example:

Dim a(10) as byte 'make an array named a, with 10 elements (1 to 10)

Dim c as Integer

For C = 1 To 10

    a(c) = c                    'assign array element

    Print a(c)                'print it

Next

## **Strings**

Strings can be up to 254 characters long in BASCOM.

To save memory you must specify how long each string must be with the DIM statement.

Dim S As String \* 10

This will reserve space for the string S with a length of 10 bytes. The actual length is 11 bytes because a nul(0) is used to terminate the string.

You can concatenate string with the + sign.

Dim S As String \* 10 , Z As String \* 10

S = "test"

$Z = S + \text{"abc"} + \text{var}$

In QB you can assign a string with a value and add the original string (or a part of it) too :

$S = \text{"test"}$

$S = \text{"a"} + s$

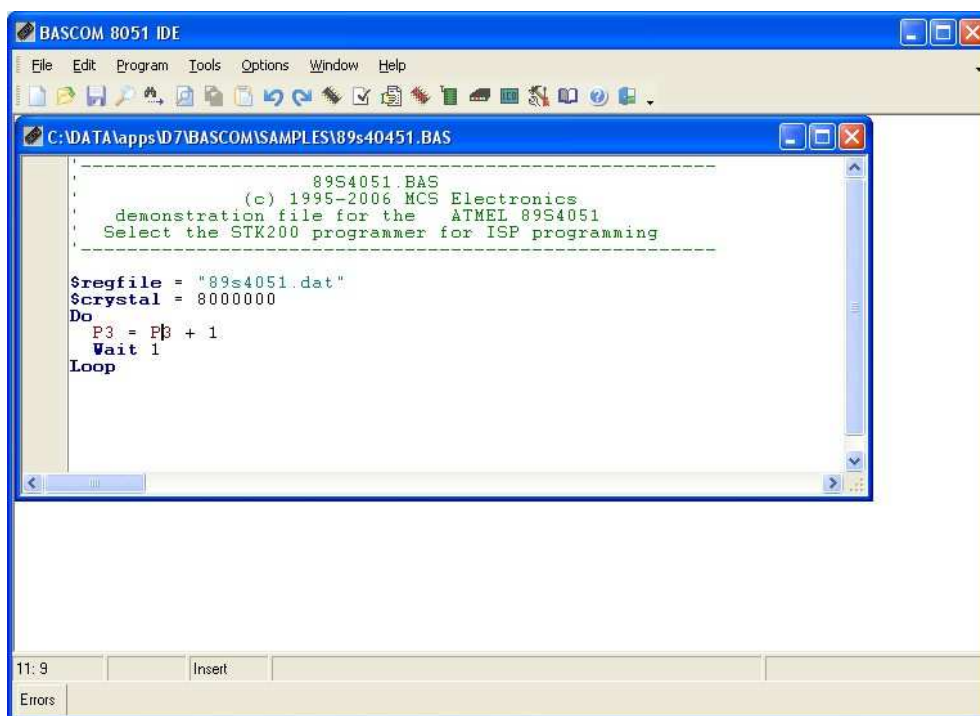
This will result in the string "atest"

In BASCOM-8051 this is NOT possible because this would require a copy of the string.

In BASCOM the string S is assigned with "a" and on that moment the original string S is destroyed. So you must make a copy of the string yourself in the event you need this functionality.

## RUNNING BASCOM-8051

When BASCOM-8051 is run, the following window will appear.



The last saved/closed program will be loaded automatically. When reformatting is enabled, the loaded program will be reformatted too. This is only meaningful for programs written with another editor.

## Keyword Reference

### 1WIRE

1Wire routines allow you to communicate with Dallas 1wire chips.

*1WRESET, 1WREAD, 1WWRITE, 1WSEARCHFIRST, 1WSEARCHNEXT, 1WIRECOUNT*

### Conditions

Conditions execute a part of the program depending on the condition

*IF, ELSE, END IF, EXIT, DO, LOOP, SELECT, FOR, NEXT, TO, THEN, WHILE .. WEND*

### Configuration

Configuration command initialize the hardware to the desired state.

*CONFIG*

### Conversion

A conversion routine is a function that converts a number or string.

*ASC, BCD, CHR, FUSING, HEX, HEXVAL, HIGH, HIGHW, MAKEDEC, MAKEBCD, MAKEINT, STR, VAL*

### Delay

Delay routines delay the program for the specified time.

*DELAY, WAIT, WAITMS*

### Directives

Directives are special instructions for the compiler. They can override a setting from the IDE.

*#IF, #ELSE, #ENDIF, \$ASM - \$END ASM, \$INCLUDE, \$BAUD, \$BGF, \$CRYSTAL, \$DEFAULT XRAM, \$IRAMSTART, \$LARGE, \$LCD, \$MAP, \$NOBREAK, \$NOINIT, \$NONAN, \$NONULL, \$NORAMCLEAR, \$NOSP, \$OBJ, \$RAMSIZE, \$RAMSTART, \$REGFILE, \$ROMSTART, \$SERIALINPUT, \$SERIALINPUT2LCD, \$SERIALOUTPUT, \$SIM*

### Graphical LCD

Graphical LCD commands extend the normal text LCD commands.

*PSET, SHOWPIC*

### I2C

I2C commands allow you to communicate with I2C chips with the TWI hardware or with emulated I2C hardware.

*I2CRECEIVE, I2CSEND, I2CSTART, I2CSTOP, I2CRBYTE, I2CWBYTE*

### Interrupts

Interrupt related routines.

*ON Interrupt , ENABLE , DISABLE , PRIORITY*

## **IO**

I/O commands are related to the I/O pins of the processor.

*ALIAS , BITWAIT , DEBOUNCE , SET , RESET*

## **Math**

Math functions

*ABS , AVG , MAX , MIN , MOD*

## **Micro**

Micro statements are highly related to the micro processor.

*BREAK , P1,P3 , IDLE , END , POWERDOWN , START , STOP , STOP TIMER*

## **Memory**

Memory functions set or read RAM , EEPROM or flash memory.

*CPEEK , ERASE , INP , OUT , PEEK , POKE , DIM , READ , RESTORE , DATA ,  
VARPTR*

## **Remote control**

Remote control statements send or receive IR commands for remote control.

*GETRC5*

## **RS-232**

RS-232 are serial routines that use the UART or emulate a UART.

*WAITKEY , PRINT , PRINTBIN , PRINTHEX , PUT , OPEN ,SPC , INKEY , INPUT ,  
INPUTBIN , INPUTHEX , GET , CLOSE*

## **SPI**

SPI routines communicate according to the SPI protocol with either hardware SPI or software emulated SPI.

*SPIIN , SPIOUT , SPIINITString*

String routines are used to manipulate strings.

*MID , LCASE , INSTR , LEFT , LEN , RIGHT , STRING , REPLACE , SPACE , UCASE*

## **Text LCD**

Text LCD routines work with the normal text based LCD displays.

*CLS , CURSOR , FOURTHLINE , HOME , LCD , LCDINIT , LCDHEX , LOCATE ,  
SHIFTLCD , SHIFTCURSOR ,DISPLAY , LOWERLINE , UPPERLINE , THIRDLINE*

## **Various**

This section contains all statements that were hard to put into another group

*CALL , CONST , COUNTER , DECR , DECLARE , DEFINT , DEFBIT ,DEFBYTE ,  
DEFLCDCHAR , DEFWORD , GETAD , GETAD2051 , GETRC , , GOSUB , GOTO ,  
INCR , LOAD , LOOKUP , LOOKUPSTR , LOW , LOWW , ON Value ,*

*READMAGCARD , REM , RETURN , RND , ROTATE , SHIFT , SHIFTIN , SHIFTOUT ,  
SOUND , SUB , SWAP*

## **Some Example Codes**

### **INKEY**

#### **Action**

Returns the ASCII value of the first character in the serial input buffer.

#### **Syntax**

```
var = INKEY()  
var = INKEY(#channel)
```

#### **Remarks**

Var	Byte, Integer, Word, Long or String variable.
Channel	The channel number of device

If there is no character waiting, a zero will be returned.

The INKEY routine can be used when you have a RS-232 interface on your uP.  
See the manual for a design of an RS-232 interface.  
The RS-232 interface can be connected to a comport of your computer.

The INKEY() function only works with the hardware UART, not the software UART.

#### **Example**

```
Dim A As Byte  
Do 'start loop  
A = Inkey() 'look for character  
If A > 0 Then 'is variable > 0?  
Print A 'yes , there was a character in the buffer  
'so print it  
End If  
Loop 'loop forever
```

#### **Example**

```
$regfile = "80517.dat"  
Open "COM2:" For Binary As #1 'open serial channel 1 on 80537  
Dim St As Byte  
St = Inkey(#1) 'get key from com2  
If St > 0 Then  
Printbin #1 , St 'send to com 2  
End If
```



Close #1

## **OPEN - CLOSE**

### **Action**

Opens and closes a device.

### **Syntax**

OPEN "device" for MODE As #channel

CLOSE #channel

### **Remarks**

Device	There are 2 hardware devices supported: COM1 and COM2. With the software UART, you must specify the port pin and the baud rate. COM3.0:9600 will use PORT 3.0 at 9600 baud. Optional is ,INVERTED this will use inverted logic so you don't need MAX232 inverters.
MODE	You can use BINARY, INPUT or OUTPUT for COM1 and COM2, but for the software UART pins, you must specify INPUT or OUTPUT.
Channel	The number of the channel to open. Must be a positive constant.

Since there are uP's such as the 80537 with 2 serial channels on board, the compiler must know which serial port you want to use. That is why the OPEN statement is implemented. With only 1 serial port on board, you don't need this statement. The statements that support the device are PRINT , PRINTHEX, INPUT and INPUTHEX.

Every opened device must be closed using the CLOSE #channel statement. Of course you must use the same channel number. The software UART, only supports the GET and PUT statements to retrieve and send data and the PRINTBIN and INPUTBIN statement. The SW UART uses timed loops and interrupts can slow down these loops. So turn interrupts off before you use the SW UART.

COM1: and COM2: are hardware ports, and can be used with PRINT etc.  
For the software UART it is important that the pin you use is bit addressable. In most cases a PORT is bit addressable but some chips have ports that are not bit addressable. When you use such a port you will get errors like : Error 208, bit variable not found.

Since the OPEN statement doesn't use real file handles like DOS but only serves as a compiler directive, it is important that you must use the CLOSE statement as the last statement in your program. The following example shows when it will NOT WORK :

```
OPEN "COM2:" FOR BINARY AS #1      'open the port
PRINT #1, "Hello"                  'print to serial 1
Gosub Test
PRINT "Hello"                      'print to serial 0
CLOSE #1
```

```
Test:
  Print #1, "test"
Return
```

Since the compiler frees the handle when it encounters the CLOSE statement, the PRINT #1, "test" code is never executed. To solve this you should put the CLOSE #1 statement under the Return statement.

```
OPEN "COM2:" FOR BINARY AS #1      'open the port
PRINT #1, "Hello"                  'print to serial 1
Gosub Test
PRINT "Hello"                      'print to serial 0
```

```
Test:
  Print #1, "test"
Return
Close #1
```

### **Example 1**

```
'only works with a 80517 or 80537
CONFIG BAUD1 = 9600                'serial 1 baudrate
OPEN "COM2:" FOR BINARY AS #1      'open the port
PRINT #1, "Hello"                  'print to serial 1
PRINT "Hello"                      'print to serial 0
CLOSE #1                           'close the channel
```

### **Example 2**

```
'works with every port pin
Dim A As Byte , S As String * 16 , I As Byte , Dum As Byte
```

```
'a software comport is named after the pin you use
'for example P3.0 will be "COM3.0:" (so there is no P)
```

'for software comports, you must provide the baudrate  
 'So for 9600 baud, the devicename is "COM3.0:9600"  
 'When you want to use the pin for sending, you must open the device for OUTPUT  
 'When you want to use the pin for receiving, you must open the device for INPUT

'At this time only variables can be sent and received with the PUT and GET statements.  
 'In the feature PRINT etc. will support these software comports.

Open "com3.1:9600" For Output As #1     'p3.1 is normally used for tx so testing is easy  
 Open "com3.0:9600,INVERTED" For Input As #2     'p3.0 is normally used for RX so testing is easy

```
S = "test this"           'assign string
Dum = Len(s)              'get length of string
For I = 1 To Dum          'for all characters from left to right
  A = Mid(s , I , 1)      'get character
  Put #1 , A              'write it to comport
Next
```

```
Do
  Get #2 , A              'get character from comport
  Put #1 , A              'write it back
  Print A                 'use normal channel
Loop
```

```
Close #1                  ' finally close device
Close #2
End
```

## GET

### Action

Retrieves a byte from the software UART.

### Syntax

GET #channel , var

### Remarks

Channel	Positive numeric constant that refers to the opened channel.
Var	A variable that receives the value from the software UART.

Note that the channel must be opened with the OPEN statement.

Also, note that the CLOSE statement, must be the last in your program. Please see comment on OPEN statement

An optional TIMEOUT can be specified so that the routine will return when no character is received.

### Example

```
Dim S As String * 12 , I As Byte , A As Byte , Dum As Byte
```

Open "com3.1:9600" For Output As #1 'p3.1 is normally used for tx so testing is easy  
Open "com3.0:9600" For Input As #2 'p3.0 is normally used for RX so testing is easy

```
S = "test this"           'assign string
Dum = Len(s)              'get length of string
For I = 1 To Dum          'for all characters from left to right
    A = Mid(s , I , 1)    'get character
    Put #1 , A            'write it to comport
Next
Do
    Get #2 , A             'get character from comport
    Put #1 , A            'write it back
    Print A               'use normal channel
Loop

Printbin #1, a            'Printbin is also supported
Inputbin #2, a            'Inputbin is also supported

Close #1                  ' finally close device
Close #2
End
```

'To use the TIMEOUT option include (without the remarks):  
'\$TIMEOUT  
' Get #2 , A TIMEOUT = 10000 'get character from comport

## PUT

### Action

Sends a byte to the software UART.

### Syntax

```
PUT #channel , var
```

**Remarks**

channel	Positive numeric constant that refers to the opened channel.
Var	A variable or constant who's value is sent to the the software UART.

**Example**

Open "com3.1:9600" For Output As #1                      'p3.1 is normally used for tx so  
testing is easy

Open "com3.0:9600" For Input As #2                      'p3.0 is normally used for RX so  
testing is easy

```
S = "test this"                      'assign string
Dum = Len(s)                      'get length of string
For I = 1 To Dum                      'for all characters from left to right
  A = Mid(s , I , 1)                      'get character
  Put #1 , A                      'write it to comport
Next
```

```
Do
  Get #2 , A                      'get character from comport
  Put #1 , A                      'write it back
  Print A                      'use normal channel
Loop
```

```
Close #1                      ' finally close device
Close #2
End
```

## APPENDIX B

### PC PROGRAM LISTING

```
unit UntAgregaDolumSirasi;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Mask, ToolEdit, CurrEdit, ComCtrls, Buttons,
  RXSpin, AdvToolBar, AdvToolBarStylers;

type
  TFrmAgregaDolumSirasi = class(TForm)
    DolumList: TListBox;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    Label2: TLabel;
    Panel4: TPanel;
    Image9: TImage;
    Panel3: TPanel;
    Image2: TImage;
    Label16: TLabel;
    Label17: TLabel;
    AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
    AdvDockPanel1: TAdvDockPanel;
    AdvToolBar1: TAdvToolBar;
    BtnKapat: TAdvToolBarButton;
    BtnEkle: TAdvToolBarButton;
    StatusBar1: TStatusBar;
    procedure FormShow(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure BtnEkleClick(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure BtnKapatClick(Sender: TObject);
    procedure Panel4MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FrmAgregaDolumSirasi: TFrmAgregaDolumSirasi;
  OldPos: integer;

implementation
```

```

uses UntGlobal, UntDm, DB, ADODB;

{$R *.dfm}

procedure TFrmAgregaDolumSirasi.FormShow(Sender: TObject);
var
  i:integer;
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
  SetWindowRgn(handle, region, true);
  KullaniciHaklari(Self.Tag,AdvToolBar1);
  with dm.Qry1 do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Select * from DolumSira Order by Sira Asc');
    Open;
    DolumList.Clear;
    for i :=1 to RecordCount do
    begin
      DolumList.Items.Add(FieldByName('Tanim').AsString);
      next;
    end;
  end;
  DolumList.SetFocus;
  DolumList.Selected[0]:=true;
end;
// ***

procedure TFrmAgregaDolumSirasi.SpeedButton1Click(Sender: TObject);
var
  i:Integer;
begin
  for i :=0 to DolumList.Items.Count-1 do
    if DolumList.Selected[i]=true then OldPos:=i;
  if OldPos>0then
  begin
    DolumList.Items.Move(OldPos,OldPos-1);
    DolumList.Selected[OldPos-1]:=true;
  end;
end;

// ***

procedure TFrmAgregaDolumSirasi.SpeedButton2Click(Sender: TObject);
var
  i:Integer;
begin
  for i :=0 to DolumList.Items.Count-1 do
    if DolumList.Selected[i]=true then OldPos:=i;
  if DolumList.Items.Count-1 > oldpos then
  begin

```

```

DolumList.Items.Move(OldPos,OldPos+1);
DolumList.Selected[OldPos+1]:=true;
end;
end;
// ***
procedure TFrmAgregaDolumSirasi.FormCreate(Sender: TObject);
begin
    Caption :=ProCaption ;
end;
// ***
procedure TFrmAgregaDolumSirasi.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
    if key=vk_F10 then btnKapat.Click;
end;
// ***
procedure TFrmAgregaDolumSirasi.BtnEkleClick(Sender: TObject);
var
    i:integer;
begin
    if Onay('Dolum Sırası Kayıt İşlemi;'+#13+'Devam Etmek İstiyor musunuz ?')=False then exit;
    with dm.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Delete * From DolumSira ');
        ExecSQL;
        for i :=0 to DolumList.Items.Count-1 do
        begin
            Close;
            SQL.Clear;
            SQL.Add('Insert Into DolumSira (Tanim,Sira) Values(:Tanim,:Sira)');
            Parameters.ParamByName('Tanim').Value :=DolumList.Items.Strings[i];
            Parameters.ParamByName('Sira').Value :=i;
            ExecSQL;
        end;
    end;
    Self.Close;
end;

// ***
procedure TFrmAgregaDolumSirasi.BtnKapatClick(Sender: TObject);
begin
    Self.Close;
end;
// ***
procedure TFrmAgregaDolumSirasi.Panel4MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;

end.

```



\*\*\*\*\*

unit UntCHDegistir;

interface

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,  
StdCtrls, Mask, ToolEdit, CurrEdit, ExtCtrls, ComCtrls, Buttons, Grids,  
BaseGrid, AdvGrid ;

type

TFrmCHDegistir = class(TForm)  
PageControl1: TPageControl;  
TabSheet1: TTabSheet;  
Image3: TImage;  
GroupBox1: TGroupBox;  
Label2: TLabel;  
Label3: TLabel;  
Label6: TLabel;  
Label7: TLabel;  
Label8: TLabel;  
Label13: TLabel;  
Label12: TLabel;  
Label15: TLabel;  
Label1: TLabel;  
Label4: TLabel;  
txtAdres01: TEdit;  
DTKayitTarihi: TDateEdit;  
txtTUnvan: TEdit;  
txtCariKod: TEdit;  
txtAdres02: TEdit;  
txtIl: TEdit;  
txtVDaire: TEdit;  
txtVNumara: TEdit;  
txtIlce: TEdit;  
txtInValue: TCurrencyEdit;  
txtOutValue: TCurrencyEdit;  
TabSheet3: TTabSheet;  
Image1: TImage;  
GroupBox3: TGroupBox;  
Label22: TLabel;  
Label23: TLabel;  
Label24: TLabel;  
Label25: TLabel;  
Label26: TLabel;  
Label27: TLabel;  
txtTelefon01: TEdit;  
txtTelefon02: TEdit;  
txtFax: TEdit;  
txtCep: TEdit;  
txtEMail: TEdit;  
txtWEB: TEdit;  
Panel5: TPanel;  
Image9: TImage;  
Image10: TImage;

```

BtnDegistir: TSpeedButton;
Label16: TLabel;
Label17: TLabel;
BtnKapat: TSpeedButton;
StatusBar1: TStatusBar;
procedure FormShow(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtnKapatClick(Sender: TObject);
procedure BtnDegistirClick(Sender: TObject);
procedure Panel5MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);

private
  { Private declarations }
public
  { Public declarations }
  iptal:Boolean;
end;

var
  FrmCHDegistir: TFrmCHDegistir;
implementation

uses UntDm, Variants, UntGlobal,
  UntUyari, UntCHListe;

{$R *.DFM}

procedure TFrmCHDegistir.FormShow(Sender: TObject);
var
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
  SetWindowRgn(handle, region, true);
  with DM.Qry1 do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Select * From CariHesap where CariKod=:bul');
    Parameters.ParamByName('bul').Value:=FrmCHListe.CHesapQuery.FieldValues['CariKod'];
    Prepared ;
    open;
    txtCariKod.Text:=FieldByName('CariKod').AsString;
    txtTUnvan.Text:=FieldByName('TUnvan').AsString;
    if FieldByName('Adres01').AsString <> "" then
      txtAdres01.Text:=FieldByName('Adres01').AsString
    else
      txtAdres01.Text:="";
    if FieldByName('Adres02').AsString <> "" then
      txtAdres02.Text:=FieldByName('Adres02').AsString
    else
      txtAdres02.Text:="";
    if FieldByName('Ilce').AsString <> "" then

```

```

    txtIlce.Text:=FieldByName('Ilce').AsString
else
    txtIlce.Text:="";
if FieldByName('Il').AsString <> "" then
    txtIl.Text:=FieldByName('Il').AsString
else
    txtIl.Text:="";
if FieldByName('VDaire').AsString <> "" then
    txtVDaire.Text:=FieldByName('VDaire').AsString
else
    txtVDaire.Text:="";
if FieldByName('VNumara').AsString <> "" then
    txtVNumara.Text:=FieldByName('VNumara').AsString
else
    txtVNumara.Text:="";
if FieldByName('Telefon01').AsString <> "" then
    txtTelefon01.Text:=FieldByName('Telefon01').AsString
else
    txtTelefon01.Text:="";
if FieldByName('Telefon02').AsString <> "" then
    txtTelefon02.Text:=FieldByName('Telefon02').AsString
else
    txtTelefon02.Text:="";
if FieldByName('Fax').AsString <> "" then
    txtFax.Text:=FieldByName('Fax').AsString
else
    txtFax.Text:="";
if FieldByName('Cep').AsString <> "" then
    txtCep.Text:=FieldByName('Cep').AsString
else
    txtCep.Text:="";
if FieldByName('EMail').AsString <> "" then
    txtEMail.Text:=FieldByName('EMail').AsString
else
    txtEMail.Text:="";
txtInValue.Value:=FieldByName('InValue').AsFloat;
txtOutValue.Value:=FieldByName('OutValue').AsFloat;
if FieldByName('WEB').AsString <> "" then
    txtWEB.Text:=FieldByName('WEB').AsString
else
    txtWEB.Text:="";
if FieldByName('KayitTarihi').AsString <> "" then
    DTKayitTarihi.Text:=FieldByName('KayitTarihi').AsString
else
    DTKayitTarihi.Text:="";
end;
end;
// ***
procedure TFrmCHDegistir.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if key=vk_escape then BtnKapat.Click;
    if key=vk_F5 then BtnDegistir.Click;
end;
// ***
procedure TFrmCHDegistir.BtnKapatClick(Sender: TObject);

```

```

begin
    iptal:=True;
    self.Close;
end;
// ***
procedure TFrmCHDegistir.BtnDegistirClick(Sender: TObject);
begin
    if txtTUnvan.Text = '' then
    begin
        Hata('Ad Soyad / Ticari Ünvan Bilgisi Eksik. ');
        exit;
    end;
    with FrmCHListe.CHesapQuery do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * From CariHesap where CariKod=:bul');
        Parameters.ParamByName('bul').Value:=txtCariKod.Text;
        Prepared ;
        open;
        Edit;
        FieldByName('TUnvan').AsString:=txtTUnvan.Text;
        FieldByName('Adres01').AsString:=txtAdres01.Text;
        FieldByName('Adres02').AsString:=txtAdres02.Text;
        FieldByName('Ilce').AsString:=txtIlce.Text;
        FieldByName('Il').AsString:=txtIl.Text;
        FieldByName('VDaire').AsString:=txtVDaire.Text;
        FieldByName('VNumara').AsString:=txtVNumara.Text;
        FieldByName('Telefon01').AsString:=txtTelefon01.Text;
        FieldByName('Telefon02').AsString:=txtTelefon02.Text;
        FieldByName('Fax').AsString:=txtFax.Text;
        FieldByName('Cep').AsString:=txtCep.Text;
        FieldByName('EMail').AsString:=txtEMail.Text;
        FieldByName('InValue').AsFloat:=txtInValue.Value;
        FieldByName('OutValue').AsFloat:=txtOutValue.Value;
        FieldByName('WEB').AsString:=txtWEB.Text;
        post;
    end;
    iptal:=False;
    self.Close;
end;
// ***
procedure TFrmCHDegistir.Panel5MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;

end.

*****

unit UntCHEkle;

interface

```

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,  
ComCtrls, ToolWin, ExtCtrls, StdCtrls, Mask, DBCtrls,  
Grids, Buttons, ToolEdit, CurrEdit, ImgList, BaseGrid, AdvGrid;

type

```
TFrmCHEkle = class(TForm)
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet3: TTabSheet;
    GroupBox3: TGroupBox;
    Label22: TLabel;
    Label23: TLabel;
    Label24: TLabel;
    Label25: TLabel;
    Label26: TLabel;
    Label27: TLabel;
    Image1: TImage;
    GroupBox1: TGroupBox;
    Label2: TLabel;
    Label3: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label13: TLabel;
    Label12: TLabel;
    Label15: TLabel;
    Image3: TImage;
    txtAdres01: TEdit;
    DTKayitTarihi: TDateEdit;
    txtTUnvan: TEdit;
    txtCariKod: TEdit;
    txtAdres02: TEdit;
    txtIl: TEdit;
    txtVDaire: TEdit;
    txtVNumara: TEdit;
    txtTelefon01: TEdit;
    txtTelefon02: TEdit;
    txtFax: TEdit;
    txtCep: TEdit;
    txtEMail: TEdit;
    txtWEB: TEdit;
    txtIlce: TEdit;
    Label1: TLabel;
    txtInValue: TCurrencyEdit;
    Label4: TLabel;
    txtOutValue: TCurrencyEdit;
    Panel5: TPanel;
    Image9: TImage;
    Image10: TImage;
    BtnKaydet: TSpeedButton;
    Label16: TLabel;
    Label17: TLabel;
    BtnKapat: TSpeedButton;
    StatusBar1: TStatusBar;
```

```

    procedure FormShow(Sender: TObject);
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure BtnKapatClick(Sender: TObject);
    procedure BtnKaydetClick(Sender: TObject);
    procedure Panel5MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
    iptal:Boolean;
  end;

var
  FrmCHEkle: TFrmCHEkle;
implementation

uses  UntGlobal, UntDm ;

{$R *.DFM}

procedure TFrmCHEkle.FormShow(Sender: TObject);
var
    region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
  SetWindowRgn(handle, region, true);
  CompTemizle(Self,1);
  txtCariKod.Text:=GetLastNumber(3);
  DTKayitTarihi.Date:=Date;
end;
// ***

procedure TFrmCHEkle.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if key=vk_F4    then BtnKaydet.Click;
  if key=vk_Escape then BtnKapat.Click;
end;
// ***

procedure TFrmCHEkle.BtnKapatClick(Sender: TObject);
begin
  iptal:=True;
  Self.Close;
end;
// ***

procedure TFrmCHEkle.BtnKaydetClick(Sender: TObject);
begin
  if txtTUnvan.Text ='' then
  begin
    Hata('Ad Soyad / Ticari Ünvan Bilgisi Eksik.' );
    Exit;
  end;

```

```

with DM.Qry1 do
begin
  Close;
  SQL.Clear;
  SQL.Add('Select * From CariHesap ');
  Prepared ;
  open;
  Append;
  FieldByName('CariKod').AsString:=txtCariKod.Text;
  FieldByName('TUnvan').AsString:=txtTUnvan.Text;
  FieldByName('Adres01').AsString:=txtAdres01.Text;
  FieldByName('Adres02').AsString:=txtAdres02.Text;
  FieldByName('Ilce').AsString:=txtIlce.Text;
  FieldByName('Il').AsString:=txtIl.Text;
  FieldByName('VDaire').AsString:=txtVDaire.Text;
  FieldByName('VNumara').AsString:=txtVNumara.Text;
  FieldByName('Telefon01').AsString:=txtTelefon01.Text;
  FieldByName('Telefon02').AsString:=txtTelefon02.Text;
  FieldByName('Fax').AsString:=txtFax.Text;
  FieldByName('Cep').AsString:=txtCep.Text;
  FieldByName('EMail').AsString:=txtEMail.Text;
  FieldByName('InValue').AsFloat:=txtInValue.Value;
  FieldByName('OutValue').AsFloat:=txtOutValue.Value;
  FieldByName('WEB').AsString:=txtWEB.Text;
  FieldByName('KayitTarihi').AsString:=DTKayitTarihi.Text;
  FieldByName('InValue').AsFloat :=txtInValue.Value;
  FieldByName('OutValue').AsFloat :=txtOutValue.Value;
  post;
  Close;
  SQL.Clear;
  SQL.Add('Select * from CariExtre2005 ');
  Prepared ;
  open;
  Append;
  FieldByName('Carikod').AsString :=txtCariKod.Text;
  FieldByName('OpDate').AsString :=DTKayitTarihi.Text;
  FieldByName('OpStatus').AsString :='A';
  FieldByName('OpNumber').AsString :='#';
  FieldByName('OpDefinition').AsString:='Hesap Açılış Değerleri';
  FieldByName('InValue').AsFloat :=txtInValue.Value;
  FieldByName('OutValue').AsFloat :=txtOutValue.Value;
  post;
end;
UpdateLastNumber(3,txtCariKod.Text);
iptal:=False;
Self.Close;
end;

// ***
procedure TFrmCHEkle.Panel5MouseDown(Sender: TObject; Button: TMouseButton;
Shift: TShiftState; X, Y: Integer);
begin
  ReleaseCapture;
  self.perform(WM_SYSCOMMAND, $F012, 0);
end;

```

end.

\*\*\*\*\*

unit UntCHHareket;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  
Dialogs, StdCtrls, Buttons, ExtCtrls, Grids, BaseGrid, AdvGrid, DBAdvGrd,  
ComCtrls, ToolWin, ImgList, Mask, ToolEdit, DB, ADODB, CurrEdit, Menus,  
cxStyles, cxCustomData, cxGraphics, cxFilter, cxData, cxDataStorage,  
cxEdit, cxDBData, cxMemo, cxGridCustomTableView, cxGridTableView,  
cxGridDBTableView, cxGridLevel, cxClasses, cxControls, cxGridCustomView,  
cxGrid, cxBlobEdit, AdvToolBar, AdvToolBarStylers, dxPSGlbl, dxPSUtl,  
dxPSEngn, dxPrnPg, dxBkgnd, dxWrap, dxPrnDev, dxPSCmpsProvider,  
dxPSFillPatterns, dxPSEdgePatterns, dxPSCore, dxPSCContainerLnk,  
dxPSCxCommon, dxPSCxGridLnk;

type

TFrmCHHareket = class(TForm)  
  pnlKriter: TPanel;  
  DTStart: TDateEdit;  
  DTEnd: TDateEdit;  
  CBoxDate: TCheckBox;  
  Panel2: TPanel;  
  Label3: TLabel;  
  lblBarkod: TLabel;  
  txtCariKod: TEdit;  
  txtTUnvan: TEdit;  
  DSQryCHHareket: TDataSource;  
  QryCHHareket: TADOQuery;  
  Label4: TLabel;  
  cmbKriter: TComboBox;  
  AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;  
  PrintPreview: TdxComponentPrinter;  
  PrintPreviewLink1: TdxGridReportLink;  
  pnlHeader: TPanel;  
  Image9: TImage;  
  Panel3: TPanel;  
  Image4: TImage;  
  Label16: TLabel;  
  Label17: TLabel;  
  AdvDockPanel1: TAdvDockPanel;  
  AdvToolBar1: TAdvToolBar;  
  BtnKapat: TAdvToolBarButton;  
  BtnSil: TAdvToolBarButton;  
  BtnDegistir: TAdvToolBarButton;  
  BtnEkle: TAdvToolBarButton;  
  BtnDetay: TAdvToolBarButton;  
  BtnYazdir: TAdvToolBarButton;  
  BtnAlacak: TAdvToolBarButton;  
  btnBorc: TAdvToolBarButton;  
  btnIslemler: TAdvToolBarButton;



```

pnlIslem: TPanel;
PnlCaption: TPanel;
Shape1: TShape;
lblCaption: TLabel;
Image2: TImage;
Label2: TLabel;
DTIslemTarihi: TDateEdit;
Label5: TLabel;
txtAciklama: TEdit;
Label6: TLabel;
txtBelgeNo: TEdit;
Label9: TLabel;
txtMiktar: TCurrencyEdit;
BtnUygula: TSpeedButton;
BtnIptal: TSpeedButton;
cxGrid1DBTableView1: TcxGridDBTableView;
cxGrid1Level1: TcxGridLevel;
cxGrid1: TcxGrid;
cxGrid1DBTableView1yil: TcxGridDBCColumn;
cxGrid1DBTableView1ay: TcxGridDBCColumn;
cxGrid1DBTableView1OpDate: TcxGridDBCColumn;
cxGrid1DBTableView1OpNumber: TcxGridDBCColumn;
cxGrid1DBTableView1OpDefinition: TcxGridDBCColumn;
cxGrid1DBTableView1InValue: TcxGridDBCColumn;
cxGrid1DBTableView1OutValue: TcxGridDBCColumn;
Label7: TLabel;
txtBakiye: TCurrencyEdit;
BtnArama: TAdvToolBarButton;
StatusBar1: TStatusBar;
procedure FormCreate(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure CBoxDateClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
procedure cmbKriterClick(Sender: TObject);
procedure BtnDetayClick(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure btnBorcClick(Sender: TObject);
procedure BtnAlacakClick(Sender: TObject);
procedure btnIslemlerClick(Sender: TObject);
procedure BtnYazdirClick(Sender: TObject);
procedure BtnIptalClick(Sender: TObject);
procedure BtnUygulaClick(Sender: TObject);
procedure BtnAramaClick(Sender: TObject);
procedure pnlHeaderMouseDown(Sender: TObject; Button: TMouseButton;
    Shift: TShiftState; X, Y: Integer);

private
{ Private declarations }
public
{ Public declarations }
procedure SetGridQuery();
procedure Islem(Durum: Boolean; BtnBaslik, lblBaslik: String);
procedure CHBorcKaydet();
procedure CHAlacakKaydet();

```

```

procedure CHBakiye();
end;

var
  FrmCHHareket: TFrmCHHareket;

implementation

uses  UntGlobal, UntDm, UntDlgCostumerSearch,UntCHIslem;

{$R *.dfm}

procedure TFrmCHHareket.CHBorcKaydet;
var
  BelgeNo:String;
begin
  if Length(Trim(txtBelgeNo.Text))=0 then
    BelgeNo:=FormatDateTime('ddmm',Date)+ FormatDateTime('hhmmss',Now)
  else
    BelgeNo:=txtBelgeNo.Text;
  if txtMiktar.Value<=0 then
    begin
      hata('Lütfen Satış Miktarını Giriniz.');
      txtMiktar.SetFocus;
      exit;
    end;
    CariHareketKayit('B',txtCariKod.Text,'--', DTIslemTarihi.Text, BelgeNo, txtAciklama.Text,
txtMiktar.Value);
    BtnIptal.Click;
    CHBakiye;
    SetGridQuery;
  end;
  // ***
procedure TFrmCHHareket.Islem(Durum: Boolean; BtnBaslik,lblBaslik: String);
begin
  pnlHeader.Enabled :=Not Durum;
  cxGrid1.Enabled :=Not Durum;
  pnlKriter.Enabled :=Not Durum;
  pnlIslem.Visible :=Durum;
  BtnUygula.Caption :=BtnBaslik;
  lblCaption.Caption:=lblBaslik;
end;
// ***
procedure TFrmCHHareket.SetGridQuery();
var
  kriter:String;
begin
  with QryCHHareket do
    begin
      Close;
      SQL.Clear;
      SQL.Add('Select * , Year(Opdate)as yıl, Month(Opdate)as ay From CariExtre2005 ');
      SQL.Add(' Where CariKod="'+txtCariKod.Text+'" ');
      case cmbKriter.ItemIndex of
        0 :kriter:=" " ; // Tümü
        1 :kriter:=" and OutValue=0 and OpStatus<>'#" ; // Yapılan Satışlar

```

```

2 :kriter:=' and InValue=0 and OpStatus <>"#" ' ; // Nakit / Çek Ödemeler
3 :kriter:=' and OpStatus="A" ' ; // Mali Yıl Açılış Değeri
end;
SQL.Add(kriter);
if CBoxDate.Checked then
begin
SQL.Add('and (opdate Between :Tarih01 and :Tarih02 )');
Parameters.ParamByName('Tarih01').Value :=DTStart.Date;
Parameters.ParamByName('Tarih02').Value :=DTEnd.Date;
end;
SQL.Add('order by Oupdate,ID Desc');
Prepared ;
open;
(FieldByName('InValue') as TNumericField ).DisplayFormat:='#,##0.00';
(FieldByName('OutValue')as TNumericField ).DisplayFormat:='#,##0.00';
Last;
end;
end;
// ***
procedure TFrmCHHareket.FormShow(Sender: TObject);
var
region: HRgn;
begin
StatusBar1.SimplePanel:=True;
StatusBar1.SimpleText:=ProCaption;
region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
SetWindowRgn(handle, region, true);
DTStart.Date:=date;
DTEnd.Date:=date;
DTIslemTarihi.Date:=Date;
// *** Hakları Ayarla...
with dm.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Select * from Haklar Where ');
SQL.Add('PerNo='+ ActPerNo+ ' And TagNo='+ IntToStr(Self.Tag));
Open;
btnBorc.Visible :=FieldByName('Kayit').AsBoolean;
BtnAlacak.Visible :=FieldByName('Kayit').AsBoolean;
BtnYazdir.Visible :=FieldByName('Yazdir').AsBoolean;
end;
end;
// ***
procedure TFrmCHHareket.CBoxDateClick(Sender: TObject);
begin
DTStart.Visible:=CBoxDate.Checked;
DTEnd.Visible:=CBoxDate.Checked;
QryCHHareket.Active:=False;
end;
// ***
procedure TFrmCHHareket.FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
if Key=Vk_F2 then BtnDetay.Click;
if Key=Vk_F3 then BtnArama.Click;

```

```

    if Key=Vk_F8 then BtnYazdir.Click;
    if Key=Vk_F10 then BtnKapat.Click;
end;
// ***
procedure TFrmCHHareket.cmbKriterClick(Sender: TObject);
begin
    QryCHHareket.Active:=False;
end;
// ***
procedure TFrmCHHareket.BtnDetayClick(Sender: TObject);
begin
    if txtCariKod.Text="" then
    begin
        Hata('Lütfen Bir Cari Kart Tanımlayınız. ');
        Exit;
    end;
    SetGridQuery;
end;

// ***
procedure TFrmCHHareket.BtnKapatClick(Sender: TObject);
begin
    Self.Close;
end;
// ***
procedure TFrmCHHareket.btnBorcClick(Sender: TObject);
begin
    if txtCariKod.Text="" then
    begin
        Hata('Lütfen Bir Cari Kart Tanımlayınız. ');
        Exit;
    end;
    if (QryCHHareket.Active=False) or (QryCHHareket.RecordCount=0) then exit;
    Islem(True,'Kaydet','Cari Kart Satış ');
    DTIslemTarihi.Date:=Date;
    txtBelgeNo.Text := "";
    txtAciklama.Text := "";
    txtMiktar.Value :=0;
    BtnUygula.Tag:=1;
    DTIslemTarihi.SetFocus;
end;
// ***
procedure TFrmCHHareket.BtnAlacakClick(Sender: TObject);
begin
    if txtCariKod.Text="" then
    begin
        Hata('Lütfen Bir Cari Kart Tanımlayınız. ');
        Exit;
    end;
    if (QryCHHareket.Active=False) or (QryCHHareket.RecordCount=0) then exit;
    Islem(True,'Kaydet','Cari Kart Ödeme ');
    DTIslemTarihi.Date:=Date;
    txtBelgeNo.Text := "";
    txtAciklama.Text := "";
    txtMiktar.Value :=0;
    BtnUygula.Tag:=2;

```

```

DTIslemTarihi.SetFocus;
end;
// ***
procedure TFrmCHHareket.btnIslemlerClick(Sender: TObject);
begin
if (QryCHHareket.Active=False) or (QryCHHareket.RecordCount<0) then exit;
if QryCHHareket.FieldName('OpNumber').AsString='#' then
begin
Hata('Açılış Değeri için Güncelleme veya Sil İşlemi Yapılamaz. ');
exit;
end;
FrmCHIslem:=TFrmCHIslem.Create(Self);
with FrmCHIslem do
begin
tag:=self.Tag;
CariKod.Text :=txtCariKod.Text;
TUnvan.Text :=txtTUnvan.Text;
txtBelgeNo.Text :=QryCHHareket.FieldName('OpNumber').AsString;
DTIslemTarihi.Text:=QryCHHareket.FieldName('OpDate').AsString;
txtAciklama.Text :=QryCHHareket.FieldName('OpDefinition').AsString;
txtMiktar.Value :=QryCHHareket.FieldName('InValue').AsFloat +
QryCHHareket.FieldName('OutValue').AsFloat;
ActRecID :=QryCHHareket.FieldName('ID').AsInteger;
if QryCHHareket.FieldName('InValue').AsFloat> QryCHHareket.FieldName('OutValue').AsFloat
then
HTur:='B'
else
HTur:='A';
ShowModal;
if iptal=False then
begin
CHBakiye;
SetGridQuery;
end;
Release;
end;
end;
end;
// ***
procedure TFrmCHHareket.BtnYazdirClick(Sender: TObject);
begin
PrintPreviewLink1.ReportTitleText:=Label16.Caption;
PrintPreviewLink1.PrinterPage.Orientation:=poPortrait;
PrintPreviewLink1.PrinterPage.PageHeader.RightTitle .Clear;
PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi:'+
FormatDateTime('dd.mm.yyyy',date));
PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Clear;
PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
PrintPreview.Preview(True,PrintPreviewLink1);
end;
// ***
procedure TFrmCHHareket.BtnIptalClick(Sender: TObject);
begin
Islem(False,'---','---');
end;
// ***
procedure TFrmCHHareket.BtnUygulaClick(Sender: TObject);

```

```

begin
  if (BtnUygula.Tag=1) and (BtnUygula.Caption='Kaydet') then CHBorcKaydet;
  if (BtnUygula.Tag=2) and (BtnUygula.Caption='Kaydet') then CHAlacakKaydet;
end;

```

```
// ***
```

```
procedure TFrmCHHareket.BtnAramaClick(Sender: TObject);
```

```
begin
```

```
  FrmDlgCostumerSearch:=TFrmDlgCostumerSearch.Create(self);
```

```
  with FrmDlgCostumerSearch do
```

```
    begin
```

```
      ShowModal;
```

```
      if iptal=False then
```

```
        begin
```

```
          txtCariKod.Text :=actCariKod;
```

```
          txtTUnvan.Text :=actTUnvan;
```

```
          txtBakiye.Value :=ActBakiye;
```

```
          QryCHHareket.Close;
```

```
        end;
```

```
        FrmDlgCostumerSearch.Release;
```

```
    end;
```

```
end;
```

```
// ***
```

```
procedure TFrmCHHareket.CHAlacakKaydet;
```

```
var
```

```
  BelgeNo:String;
```

```
begin
```

```
if Length(Trim(txtBelgeNo.Text))=0 then
```

```
  BelgeNo:=FormatDateTime('ddmm',Date)+ FormatDateTime('hhmmss',Now)
```

```
else
```

```
  BelgeNo:=txtBelgeNo.Text;
```

```
if txtMiktar.Value<=0 then
```

```
begin
```

```
  hata('Lütfen Ödeme Miktarını Giriniz.');
```

```
  txtMiktar.SetFocus;
```

```
  exit;
```

```
end;
```

```
  CariHareketKayit('A',txtCariKod.Text,
```

```
    '--',DTIslemTarihi.Text,BelgeNo,txtAciklama.Text,txtMiktar.Value);
```

```
  BtnIptal.Click;
```

```
  CHBakiye;
```

```
  SetGridQuery;
```

```
end;
```

```
// ***
```

```
procedure TFrmCHHareket.CHBakiye;
```

```
begin
```

```
with dm.QryArama do
```

```
begin
```

```
  Close;
```

```
  SQL.Clear;
```

```
  SQL.Add('Select (InValue-OutValue) as bakiye from CariHesap ');
```

```
  SQL.Add('Where CariKod="' +txtCariKod.Text+'" ');
```

```
  Open;
```

```
  txtBakiye.Value:=FieldByName('Bakiye').AsFloat;
```

```

    Close;
end;
end;
// ***
procedure TFrmCHHareket.pnlHeaderMouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;

end.

*****

unit UntCHIslem;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, Buttons, CurrEdit, Mask, ToolEdit, ComCtrls, ExtCtrls,
    AdvToolBar, AdvToolBarStylers;

type
    TFrmCHIslem = class(TForm)
        Panel1: TPanel;
        Label5: TLabel;
        lblBarkod: TLabel;
        CariKod: TEdit;
        TUnvan: TEdit;
        PageControl1: TPageControl;
        TabSheet1: TTabSheet;
        Label1: TLabel;
        Label2: TLabel;
        Label3: TLabel;
        Label4: TLabel;
        DTIslemTarihi: TDateEdit;
        txtMiktar: TCurrencyEdit;
        txtAciklama: TEdit;
        txtBelgeNo: TEdit;
        pnlHeader: TPanel;
        Image9: TImage;
        Panel3: TPanel;
        Image4: TImage;
        Label16: TLabel;
        Label17: TLabel;
        AdvDockPanel1: TAdvDockPanel;
        AdvToolBar1: TAdvToolBar;
        BtnKapat: TAdvToolBarButton;
        BtnSil: TAdvToolBarButton;
        BtnDegistir: TAdvToolBarButton;
        AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
        StatusBar1: TStatusBar;
        procedure BtnKapatClick(Sender: TObject);
        procedure FormCreate(Sender: TObject);
    end;

```

```

    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure BtnDegistirClick(Sender: TObject);
    procedure BtnSilClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
  ActRecID:Integer;
  HTur:string; // Hareket Türü;
  iptal:Boolean;
end;

var
  FrmCHIslem: TFrmCHIslem;

implementation

uses UntGlobal, UntDm;

{$R *.dfm}

procedure TFrmCHIslem.BtnKapatClick(Sender: TObject);
begin
  iptal:=True;
  Self.Close
end;
// ***

procedure TFrmCHIslem.FormCreate(Sender: TObject);
begin
  iptal:=True;
end;
// ***

procedure TFrmCHIslem.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if (key=vk_F5) and (BtnDegistir.Visible=True) then BtnDegistir.Click;
  if (key=vk_F6) and (BtnSil.Visible=True) then BtnSil.Click;
  if (key=vk_F10) then BtnKapat.Click;
end;

// ***

procedure TFrmCHIslem.BtnDegistirClick(Sender: TObject);
begin
  CariHareketSil(ActRecID);
  CariHareketKayit(HTur,CariKod.Text,"DTIslemTarihi.Text,txtBelgeNo.Text,
    txtAciklama.Text,txtMiktar.Value);
  iptal:=False;
  self.Close;
end;
// ***

procedure TFrmCHIslem.BtnSilClick(Sender: TObject);

```



```

begin
if Onay('Cari Hareket Silinecektir')=False then Exit;
CariHareketSil(ActRecID);
iptal:=False;
self.Close;
end;
// ***
procedure TFrmCHIslem.FormShow(Sender: TObject);
var
region: HRgn;
begin
StatusBar1.SimplePanel:=True;
StatusBar1.SimpleText:=ProCaption;
region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
SetWindowRgn(handle, region, true);
// *** Hakları Ayarla...
with dm.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Select * from Haklar Where ');
SQL.Add('PerNo='+ ActPerNo+ ' And TagNo='+ IntToStr(Self.Tag));
Open;
BtnDegistir.Visible:=FieldByName('Degistir').AsBoolean;
BtnSil.Visible :=FieldByName('Sil').AsBoolean;
end;
end;

end.

*****

unit UntCHListe;

interface

uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
DBGrids, Buttons, StdCtrls, DB, ADODB, Grids, AdvToolBar,
AdvToolBarStylers, XPMAN, ExtCtrls, ComCtrls, cxStyles, cxCustomData,
cxGraphics, cxFilter, cxData, cxDataStorage, cxEdit, cxDBData, DBCtrls,
cxSplitter, cxGridLevel, cxGridCustomTableView, cxGridTableView,
cxGridDBTableView, cxClasses, cxControls, cxGridCustomView, cxGrid,
cxMemo, dxPSGlbl, dxPSUtl, dxPSEngn, dxPrnPg, dxBkgnd, dxWrap, dxPrnDev,
dxPSCmpsProvider, dxPSFillPatterns, dxPSEdgePatterns, dxPSCore,
dxPScxCommon, dxPScxGridLnk;

type
TFrmCHListe = class(TForm)
AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
CHesapQuery: TADOQuery;
DSCHesapQuery: TDataSource;
Panel1: TPanel;
Label2: TLabel;
cmbKriter: TComboBox;
Label3: TLabel;

```

txtKiriter: TEdit;  
 BtnAra: TSpeedButton;  
 cxGrid1DBTableView1: TcxGridDBTableView;  
 cxGrid1Level1: TcxGridLevel;  
 cxGrid1: TcxGrid;  
 cxGrid1DBTableView1Bakiye: TcxGridDBColumn;  
 cxGrid1DBTableView1CariKod: TcxGridDBColumn;  
 cxGrid1DBTableView1TUnvan: TcxGridDBColumn;  
 cxGrid1DBTableView1InValue: TcxGridDBColumn;  
 cxGrid1DBTableView1OutValue: TcxGridDBColumn;  
 Panel2: TPanel;  
 cxSplitter1: TcxSplitter;  
 PageControl1: TPageControl;  
 TabSheet1: TTabSheet;  
 GroupBox1: TGroupBox;  
 Label6: TLabel;  
 Label7: TLabel;  
 Label8: TLabel;  
 Label13: TLabel;  
 Label12: TLabel;  
 Label15: TLabel;  
 TabSheet3: TTabSheet;  
 GroupBox2: TGroupBox;  
 Label22: TLabel;  
 Label23: TLabel;  
 Label24: TLabel;  
 Label25: TLabel;  
 Label26: TLabel;  
 Label27: TLabel;  
 Image3: TImage;  
 Image1: TImage;  
 DBText1: TDBText;  
 DBText2: TDBText;  
 DBText3: TDBText;  
 DBText4: TDBText;  
 DBText5: TDBText;  
 DBText6: TDBText;  
 DBText7: TDBText;  
 DBText8: TDBText;  
 DBText9: TDBText;  
 DBText10: TDBText;  
 DBText11: TDBText;  
 DBText12: TDBText;  
 DBText13: TDBText;  
 PrintPreview: TdxComponentPrinter;  
 PrintPreviewLink1: TdxGridReportLink;  
 pnlHeader: TPanel;  
 Image9: TImage;  
 Panel3: TPanel;  
 Image4: TImage;  
 Label16: TLabel;  
 Label17: TLabel;  
 AdvDockPanel1: TAdvDockPanel;  
 AdvToolBar1: TAdvToolBar;  
 BtnKapat: TAdvToolBarButton;  
 BtnSil: TAdvToolBarButton;

```

BtnDegistir: TAdvToolBarButton;
BtnEkle: TAdvToolBarButton;
BtnYazdir: TAdvToolBarButton;
StatusBar1: TStatusBar;
procedure FormShow(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtnEkleClick(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure BtnDegistirClick(Sender: TObject);
procedure BtnSilClick(Sender: TObject);
procedure BtnAraClick(Sender: TObject);
procedure txtKiriterKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure cmbKriterClick(Sender: TObject);
procedure BtnYazdirClick(Sender: TObject);
procedure pnlHeaderMouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);

private
  { Private declarations }
public
  { Public declarations }
  procedure UpdateGridView();
end;

var
  FrmCHListe: TFrmCHListe;

implementation

uses UntDm, UntGlobal,UntCHEkle, UntCHDegistir, UntCHSil;

{$R *.DFM}

procedure TFrmCHListe.UpdateGridView();
begin
  with CHesapQuery do
  begin
    close;
    SQL.Clear;
    SQL.Add ('Select *,(Adres01+Adres02)As Adres,(InValue-OutValue)as Bakiye ');
    SQL.Add (' from CariHesap ');
    case cmbKriter.ItemIndex of
      0 : SQL.Add('where TUnvan like :kriter ');
      1 : SQL.Add('where CariKod like :kriter ');
    end;
    SQL.Add(' order by TUnvan Asc; ');
    if cmbKriter.ItemIndex=0 then
      Parameters.ParamByName('kriter').Value:=txtKiriter.Text+'%'
    else
      Parameters.ParamByName('kriter').Value:='%'+txtKiriter.Text+'%';
    Prepared;
    Open;
    (FieldByName('InValue')as TNumericField).DisplayFormat:='0.00';
    (FieldByName('OutValue')as TNumericField).DisplayFormat:='0.00';
  end;
end;

```

```

(FieldByName('Bakiye')as TNumericField).DisplayFormat:='0.00';
first;
end;
end;
// ***
procedure TFrmCHListe.FormShow(Sender: TObject);
var
region: HRgn;
begin
StatusBar1.SimplePanel:=True;
StatusBar1.SimpleText:=ProCaption;
region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
SetWindowRgn(handle, region, true);
KullaniciHaklari(Self.Tag,AdvToolBar1);
Caption:=proCaption;
end;
// ***
procedure TFrmCHListe.FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
if Key=Vk_F3 then BtnAra.Click;
if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
if (key = vk_F5) and (BtnDegistir.Visible=True) then BtnDegistir.Click;
if (key = vk_F6) and (BtnSil.Visible=True) then BtnSil.Click;
if (key = vk_F8) and (BtnYazdir.Visible=True) then BtnYazdir.Click;
if Key=Vk_F10 then BtnKapat.Click;
end;
// ***
procedure TFrmCHListe.BtnEkleClick(Sender: TObject);
begin
FrmCHEkle:=TFrmCHEkle.Create(Self);
FrmCHEkle.ShowModal ;
if FrmCHEkle.iptal=False then UpdateGridView;
FrmCHEkle.Release ;
end;
// ***
procedure TFrmCHListe.BtnKapatClick(Sender: TObject);
begin
Self.Close;
end;
// ***
procedure TFrmCHListe.BtnDegistirClick(Sender: TObject);
begin
if (CHesapQuery.Active=False) or (CHesapQuery.RecordCount=0) then exit;
FrmCHDegistir:=TFrmCHDegistir.Create(Self);
FrmCHDegistir.ShowModal ;
if FrmCHDegistir.iptal=False then UpdateGridView;
FrmCHDegistir.Release ;
end;
// ***
procedure TFrmCHListe.BtnSilClick(Sender: TObject);
begin
if (CHesapQuery.Active=False) or (CHesapQuery.RecordCount=0) then exit;
FrmCHSil:=TFrmCHSil.Create(self);
FrmCHSil.ShowModal ;
if FrmCHSil.iptal=False then UpdateGridView;

```

```

    FrmCHSil.Release ;
end;
// ***
procedure TFrmCHListe.BtnAraClick(Sender: TObject);
begin
    UpdateGridView
end;
// ***
procedure TFrmCHListe.txtKiriterKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if key=13 then BtnAra.Click;
end;

// ***
procedure TFrmCHListe.cmbKriterClick(Sender: TObject);
begin
    txtKiriter.Text:="";
    txtKiriter.SetFocus;
end;
// ***
procedure TFrmCHListe.BtnYazdirClick(Sender: TObject);
begin
    PrintPreviewLink1.ReportTitleText:=Label16.Caption;
    PrintPreviewLink1.PrinterPage.Orientation:=poPortrait;
    PrintPreviewLink1.PrinterPage.PageHeader.RightTitle .Clear;
    PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi:'+'
    FormatDateTime('dd.mm.yyyy',date));
    PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Clear;
    PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
    PrintPreview.Preview(True,PrintPreviewLink1);
end;
// ***
procedure TFrmCHListe.pnlHeaderMouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;

end.

*****

unit UntCHSil;

interface

uses
    Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
    Buttons, StdCtrls, Mask, ToolEdit, CurrEdit, ExtCtrls, ComCtrls, Grids,
    BaseGrid, AdvGrid;

type
    TFrmCHSil = class(TForm)

```

```

PageControl1: TPageControl;
TabSheet1: TTabSheet;
Image3: TImage;
GroupBox1: TGroupBox;
Label2: TLabel;
Label3: TLabel;
Label6: TLabel;
Label7: TLabel;
Label8: TLabel;
Label13: TLabel;
Label12: TLabel;
Label15: TLabel;
Label1: TLabel;
Label4: TLabel;
txtAdres01: TEdit;
DTKayitTarihi: TDateEdit;
txtTUnvan: TEdit;
txtCariKod: TEdit;
txtAdres02: TEdit;
txtIl: TEdit;
txtVDaire: TEdit;
txtVNumara: TEdit;
txtIlce: TEdit;
txtInValue: TCurrencyEdit;
txtOutValue: TCurrencyEdit;
TabSheet3: TTabSheet;
Image1: TImage;
GroupBox3: TGroupBox;
Label22: TLabel;
Label23: TLabel;
Label24: TLabel;
Label25: TLabel;
Label26: TLabel;
Label27: TLabel;
txtTelefon01: TEdit;
txtTelefon02: TEdit;
txtFax: TEdit;
txtCep: TEdit;
txtEMail: TEdit;
txtWEB: TEdit;
Panel5: TPanel;
Image9: TImage;
Image10: TImage;
Label16: TLabel;
Label17: TLabel;
BtnKapat: TSpeedButton;
StatusBar1: TStatusBar;
BtnSil: TSpeedButton;
procedure FormShow(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtnKapatClick(Sender: TObject);
procedure BtnSilClick(Sender: TObject);
procedure Panel5MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);

```

```

private
  { Private declarations }
public
  { Public declarations }
  iptal:Boolean;
end;

var
  FrmCHSil: TFrmCHSil;

implementation

uses UntDm, UntGlobal, UntCHListe;

{$R *.DFM}

procedure TFrmCHSil.FormShow(Sender: TObject);
var
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
  SetWindowRgn(handle, region, true);
  with DM.Qry1 do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Select * From CariHesap where CariKod=:bul');
    arameters.ParamByName('bul').Value:=FrmCHListe.CHesapQuery.FieldValues['CariKod'];
    Prepared ;
    open;
    txtCariKod.Text:=FieldByName('CariKod').AsString;
    txtTUnvan.Text:=FieldByName('TUnvan').AsString;
    if FieldByName('Adres01').AsString <> "" then
      txtAdres01.Text:=FieldByName('Adres01').AsString
    else
      txtAdres01.Text:="";
    if FieldByName('Adres02').AsString <> "" then
      txtAdres02.Text:=FieldByName('Adres02').AsString
    else
      txtAdres02.Text:="";
    if FieldByName('Ilce').AsString <> "" then
      txtIlce.Text:=FieldByName('Ilce').AsString
    else
      txtIlce.Text:="";
    if FieldByName('Il').AsString <> "" then
      txtIl.Text:=FieldByName('Il').AsString
    else
      txtIl.Text:="";
    if FieldByName('VDaire').AsString <> "" then
      txtVDaire.Text:=FieldByName('VDaire').AsString
    else
      txtVDaire.Text:="";
    if FieldByName('VNumara').AsString <> "" then
      txtVNumara.Text:=FieldByName('VNumara').AsString

```

```

else
    txtVNumara.Text:="";
if FieldByName("Telefon01").AsString <> "" then
    txtTelefon01.Text:=FieldByName("Telefon01").AsString
else
    txtTelefon01.Text:="";
if FieldByName("Telefon02").AsString <> "" then
    txtTelefon02.Text:=FieldByName("Telefon02").AsString
else
    txtTelefon02.Text:="";
if FieldByName('Fax').AsString <> "" then
    txtFax.Text:=FieldByName('Fax').AsString
else
    txtFax.Text:="";
if FieldByName('Cep').AsString <> "" then
    txtCep.Text:=FieldByName('Cep').AsString
else
    txtCep.Text:="";
if FieldByName('EMail').AsString <> "" then
    txtEMail.Text:=FieldByName('EMail').AsString
else
    txtEMail.Text:="";
txtInValue.Value:=FieldByName('InValue').AsFloat;
txtOutValue.Value:=FieldByName('OutValue').AsFloat;
if FieldByName('WEB').AsString <> "" then
    txtWEB.Text:=FieldByName('WEB').AsString
else
    txtWEB.Text:="";
if FieldByName('KayitTarihi').AsString <> "" then
    DTKayitTarihi.Text:=FieldByName('KayitTarihi').AsString
else
    DTKayitTarihi.Text:="";
end;
end;
// ***
procedure TFrmCHSil.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if key=vk_F6 then BtnSil.Click;
    if key=vk_Escape then BtnKapat.Click;
end;
// ***
procedure TFrmCHSil.BtnKapatClick(Sender: TObject);
begin
    iptal:=True;
    self.Close;
end;
// ***
procedure TFrmCHSil.BtnSilClick(Sender: TObject);
begin
    if Onay('Kayıt Silinecektir'+#13 +'Devam Etmek İstiyor musunuz ?') =False then Exit;
    with DM.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * From CariExtre2005 ');
    end;
end;

```



```

SQL.Add(' Where CariKod="'+txtCariKod.Text+'" And OpStatus<>"A" ');
Prepared ;
Open;
if RecordCount>0 then
begin
  Hata('Cari Hesap Kartına ait Extre Hareketleri Olduğundan Kart Silinemez. ');
  Exit;
end
else
begin
  Close;
  SQL.Clear;
  SQL.Add('Delete * From CariHesap Where CariKod="'+txtCariKod.Text+'" ');
  Prepared;
  ExecSQL;
  Close;
  SQL.Clear;
  SQL.Add('Select * From CariExtre2005 ');
  SQL.Add(' Where CariKod="'+txtCariKod.Text+'" ');
  Prepared;
  ExecSQL;
  self.Close;
end;
end;
end;
// ***
procedure TFrmCHSil.Panel5MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  ReleaseCapture;
  self.perform(WM_SYSCOMMAND, $F012, 0);
end;

end.

*****

unit UntDlgCostumerSearch;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, Grids, BaseGrid, AdvGrid, DBAdvGrd, StdCtrls, Buttons, ExtCtrls,
DBAdvGrid, DBGrids;

type
TFrmDlgCostumerSearch = class(TForm)
  PnlCaption: TPanel;
  Shape1: TShape;
  CapImage: TImage;
  Label1: TLabel;
  Panel2: TPanel;
  BtnAra: TSpeedButton;
  Label2: TLabel;
  Label3: TLabel;

```

```

cmbKriter: TComboBox;
txtKiriter: TEdit;
Panel4: TPanel;
btnTamam: TSpeedButton;
Btntal: TSpeedButton;
DBGrid1: TDBGrid;
procedure BtnAraClick(Sender: TObject);
procedure txtKiriterKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtntalClick(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure btnTamamClick(Sender: TObject);
procedure Grd1DblClick(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure DBGrid1DblClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
private
  { Private declarations }
public
  actCariKod,actTUnvan:String;
  ActBakiye:Double;
  iptal:Boolean;
  { Public declarations }
end;

var
  FrmDlgCostumerSearch: TFrmDlgCostumerSearch;

implementation

uses UntDm, UntGlobal;

{$R *.dfm}

procedure TFrmDlgCostumerSearch.BtnAraClick(Sender: TObject);
begin
  with dm.QryArama do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Select CariKod ,TUnvan ,(InValue-OutValue) as bakiye from CariHesap ');
    case cmbKriter.ItemIndex of
    0 : SQL.Add('where TUnvan like :kriter ');
    1 : SQL.Add('where CariKod like :kriter ');
    end;
    SQL.Add(' order by Carikod Asc; ');
    Parameters.ParamByName('kriter').Value:=txtKiriter.Text+'% ' ;
    Prepared;
    Open;
  end;
end;
// ***
procedure TFrmDlgCostumerSearch.txtKiriterKeyDown(Sender: TObject;
  var Key: Word; Shift: TShiftState);
begin

```

```

    if Key=13 then BtnAra.Click;
end;
// ***
procedure TFrmDlgCostumerSearch.BtniptalClick(Sender: TObject);
begin
    iptal:=True;
    Self.Close;
end;
// ***
procedure TFrmDlgCostumerSearch.FormCreate(Sender: TObject);
begin
    iptal:=True;
with DM.QryArama do
    begin
        Close;
        SQL.Clear;
    end;
end;
// ***
procedure TFrmDlgCostumerSearch.btnTamamClick(Sender: TObject);
begin
if (Dm.QryArama.Active=False) or (Dm.QryArama.RecordCount=0) then
    exit
else
begin
    iptal :=False;
    actCariKod:=dm.QryArama.FieldValues['CariKod'];
    actTUnvan :=dm.QryArama.FieldValues['TUnvan'];
    ActBakiye :=dm.QryArama.FieldValues['Bakiye'];
    Self.Close;
end;
end;
// ***
procedure TFrmDlgCostumerSearch.Grđ1DbClick(Sender: TObject);
begin
    btnTamam.Click;
end;
// ***
procedure TFrmDlgCostumerSearch.FormShow(Sender: TObject);
begin
    Caption:=proCaption;
end;
// ***
procedure TFrmDlgCostumerSearch.DBGrid1DbClick(Sender: TObject);
begin
    btnTamam.Click;
end;
// ***
procedure TFrmDlgCostumerSearch.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if key=vk_escape then Btniptal.Click;
end;

end.

```

\*\*\*\*\*

unit UntDlgPersonalSearch;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  
Dialogs, ExtCtrls, StdCtrls, Buttons, ImgList, Grids, DBGrids, BaseGrid,  
AdvGrid, DBAdvGrd, DBAdvGrid, DB, ADODB;

type

TFrmDlgPersonalSearch = class(TForm)  
  PnlCaption: TPanel;  
  Shape1: TShape;  
  CapImage: TImage;  
  Label1: TLabel;  
  Panel2: TPanel;  
  BtnAra: TSpeedButton;  
  Label2: TLabel;  
  Label3: TLabel;  
  cmbKriter: TComboBox;  
  txtKiriter: TEdit;  
  Panel4: TPanel;  
  btnTamam: TSpeedButton;  
  BtnIptal: TSpeedButton;  
  Grd1: TDBAdvGrid;  
  DSQryPersonel: TDataSource;  
  QryPersonel: TADOQuery;  
  procedure FormCreate(Sender: TObject);  
  procedure FormKeyDown(Sender: TObject; var Key: Word;  
    Shift: TShiftState);  
  procedure BtnIptalClick(Sender: TObject);  
  procedure BtnAraClick(Sender: TObject);  
  procedure FormShow(Sender: TObject);  
  procedure btnTamamClick(Sender: TObject);  
  procedure Grd1DblClick(Sender: TObject);  
  procedure txtKiriterKeyDown(Sender: TObject; var Key: Word;  
    Shift: TShiftState);  
private  
  { Private declarations }  
public  
  { Public declarations }  
  APerAdi, APerSoyadi, APerNo, AKullanici: String;  
  iptal: Boolean;  
end;

var

  FrmDlgPersonalSearch: TFrmDlgPersonalSearch;

implementation

uses UntDm, UntGlobal;

{ \$R \*.dfm }

```

procedure TFrmDlgPersonalSearch.FormCreate(Sender: TObject);
begin
  Caption:=ProCaption;
  iptal:=true;
end;
// ***

procedure TFrmDlgPersonalSearch.FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
  if key = vk_Escape then  Btniptal.Click ;
end;
// ***

procedure TFrmDlgPersonalSearch.BtnIptalClick(Sender: TObject);
begin
  self.Close;
  iptal:=true;
end;
// ***

procedure TFrmDlgPersonalSearch.BtnAraClick(Sender: TObject);
begin
  with QryPersonel do
  begin
    close;
    SQL.Clear;
    SQL.Add('Select * from Personel Where ');
    case cmbKriter.ItemIndex of
    0 :SQL.Add(' Peradi like :kriter order by Peradi Asc ');
    1 :SQL.Add(' Persoyadi like :kriter  order by Peradi Asc ');
    2 :SQL.Add(' PerNo like :kriter  order by Peradi Asc ');
    end;
    Parameters.ParamByName('kriter').Value:=txtKriter.Text + '%';
    Prepared;
    Open;
  end;
end;
// ***

procedure TFrmDlgPersonalSearch.FormShow(Sender: TObject);
begin
  with QryPersonel do
  begin
    Close;
    SQL.Clear;
  end;
end;
// ***

procedure TFrmDlgPersonalSearch.btnTamamClick(Sender: TObject);
begin
  if (QryPersonel.Active =False) or (QryPersonel.RecordCount=0) then Exit;
  APerAdi  :=QryPersonel.FieldValues['PerAdi'];
  APerSoyadi:=QryPersonel.FieldValues['PerSoyadi'];
  APerNo   :=QryPersonel.FieldValues['PerNo'];
  AKullanici:=QryPersonel.FieldValues['KullaniciAdi'];
  iptal:=false;
  self.Close;
end;
// ***

```

```

procedure TFrmDlgPersonalSearch.Grd1DbClick(Sender: TObject);
begin
    btnTamam.Click;
end;
// ***
procedure TFrmDlgPersonalSearch.txtKiriterKeyDown(Sender: TObject;
    var Key: Word; Shift: TShiftState);
begin
    if key = 13 then btnaraClick(sender);
end;
end.
*****

unit UntDlgStockSearch;

interface

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

type
    TFrmDlgStockSearch = class(TForm)
        PnlCaption: TPanel;
        Shape1: TShape;
        CapImage: TImage;
        Label1: TLabel;
        Panel2: TPanel;
        BtnAra: TSpeedButton;
        Label2: TLabel;
        Label3: TLabel;
        cmbKriter: TComboBox;
        txtKiriter: TEdit;
        Panel4: TPanel;
        btnTamam: TSpeedButton;
        Btniptal: TSpeedButton;
        DBGrid1: TDBGrid;
        DSQryAra: TDataSource;
        QryAra: TADOQuery;
        procedure BtniptalClick(Sender: TObject);
        procedure BtnAraClick(Sender: TObject);
        procedure txtKiriterKeyDown(Sender: TObject; var Key: Word;
            Shift: TShiftState);
        procedure btnTamamClick(Sender: TObject);
        procedure Grd1DbClick(Sender: TObject);
        procedure FormKeyDown(Sender: TObject; var Key: Word;
            Shift: TShiftState);
        procedure FormShow(Sender: TObject);
        procedure DBGrid1DbClick(Sender: TObject);
    private
        { Private declarations }
    public
        { Public declarations }
        actKod,actAdi,actBirim :String;
        actMiktar:Double;
    end;
end;

```

```

    iptal:Boolean;
end;

var
    FrmDlgStockSearch: TFrmDlgStockSearch;

implementation

uses  UntGlobal, UntDm;

{$R *.dfm}

procedure TFrmDlgStockSearch.BtniptalClick(Sender: TObject);
begin
    iptal:=True;
    Self.Close;
end;
// ***

procedure TFrmDlgStockSearch.BtnAraClick(Sender: TObject);
begin
    with QryAra do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select Kod,Adi,Miktar,Birim from Stok ');
        case cmbKriter.ItemIndex of
            0 : SQL.Add('where Adi like :kriter ');
            1 : SQL.Add('where Kod like :kriter ');
        end;
        SQL.Add(' order by Kod Asc ');
        Parameters.ParamByName('kriter').Value:=txtKiriter.Text+'%';
        Prepared;
        Open;
    end;
end;
// ***

procedure TFrmDlgStockSearch.txtKiriterKeyDown(Sender: TObject;
    var Key: Word; Shift: TShiftState);
begin
    if key=13 then BtnAra.Click;
end;
//***

procedure TFrmDlgStockSearch.btnTamamClick(Sender: TObject);
begin
    if (QryAra.Active=False) or (QryAra.RecordCount=0) then
        Exit
    else
        begin
            actKod :=QryAra.FieldByName('Kod').AsString;
            actAdi :=QryAra.FieldByName('Adi').AsString;
            actBirim :=QryAra.FieldByName('Birim').AsString;
            actMiktar :=QryAra.FieldByName('Miktar').AsFloat;
            iptal:=False;
            Self.Close;
        end;
end;
end;

```

```

// ***
procedure TFrmDlgStockSearch.Grd1DbClick(Sender: TObject);
begin
    btnTamam.Click;
end;
// ***

procedure TFrmDlgStockSearch.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if Key=vk_Escape then Btntiptal.Click;
end;
// ***

procedure TFrmDlgStockSearch.FormShow(Sender: TObject);
begin
    Caption:=proCaption;
    QryAra.Close;
    QryAra.SQL.Clear;
end;
// ***

procedure TFrmDlgStockSearch.DBGrid1DbClick(Sender: TObject);
begin
    btnTamam.Click;
end;

end.

*****

unit UntDlgUrunArama;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, Grids, BaseGrid, AdvGrid, DBAdvGrd, StdCtrls, Buttons, ExtCtrls,
    DBAdvGrid;

type
    TFrmDlgUrunArama = class(TForm)
        PnlCaption: TPanel;
        Shape1: TShape;
        CapImage: TImage;
        Label1: TLabel;
        Panel2: TPanel;
        BtnAra: TSpeedButton;
        Label2: TLabel;
        Label3: TLabel;
        cmbKriter: TComboBox;
        txtKriter: TEdit;
        Panel4: TPanel;
        btnTamam: TSpeedButton;
        Btntiptal: TSpeedButton;
        DBAdvGrid1: TDBAdvGrid;
        procedure BtntiptalClick(Sender: TObject);
        procedure BtnAraClick(Sender: TObject);
        procedure txtKriterKeyDown(Sender: TObject; var Key: Word;

```



```

    Shift: TShiftState);
procedure btnTamamClick(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
procedure DBAdvGrid1DblClick(Sender: TObject);
private
    { Private declarations }
public
    { Public declarations }
    ActSpektNo,ActUrunKod,ActUrunAd,ActUrunTip:String;
    iptal:Boolean;
end;

var
    FrmDlgUrunArama: TFrmDlgUrunArama;

implementation

uses UntDm, ADODB, UntGlobal;

{$R *.dfm}

procedure TFrmDlgUrunArama.BtniptalClick(Sender: TObject);
begin
    iptal:=True;
    Self.Close;
end;
// ***

procedure TFrmDlgUrunArama.BtnAraClick(Sender: TObject);
var
    tmp:string;
begin
    if cmbKriter.ItemIndex=0 then
        tmp:= 'SPK'+txtKiriter.Text+'%'
    else
        tmp:=txtKiriter.Text+'% ' ;
    with dm.QryArama do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select SpektNo,UrunKod,UrunAd,UrunTip from Urunler ');
        case cmbKriter.ItemIndex of
            0 : SQL.Add('where SpektNo like "' +tmp+ '" ');
            1 : SQL.Add('where UrunKod like "' +tmp+ '" ');
            2 : SQL.Add('where UrunAd like "' +tmp+ '" ');
            3 : SQL.Add('where UrunTip like "' +tmp+ '" ');
        end;
        SQL.Add(' order by UrunAd Asc ');
        Open;
    end;
end;
// ***

procedure TFrmDlgUrunArama.txtKiriterKeyDown(Sender: TObject;
    var Key: Word; Shift: TShiftState);
begin

```

```

    if key=13 then BtnAra.Click;
end;
//***
procedure TFrmDlgUrunArama.btnTamamClick(Sender: TObject);
begin
if (dm.QryArama.Active=False) or (dm.QryArama.RecordCount=0) then
    Exit
else
begin
    ActSpektNo:=Dm.QryArama.FieldValues['SpektNo'];
    ActUrunKod:=Dm.QryArama.FieldValues['UrunKod'];
    ActUrunAd :=Dm.QryArama.FieldValues['UrunAd'];
    ActUrunTip:=Dm.QryArama.FieldValues['UrunTip'];
    iptal:=False;
    Self.Close;
end;
end;
// ***
procedure TFrmDlgUrunArama.FormCreate(Sender: TObject);
begin
    Caption:=ProCaption;
with dm.QryArama do
begin
    Close;
    SQL.Clear;
end;
end;
// ***
procedure TFrmDlgUrunArama.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if Key=vk_Escape then Btntal.Click;
end;
// ***
procedure TFrmDlgUrunArama.DBAdvGrid1DbClick(Sender: TObject);
begin
    btnTamam.Click;
end;
end.

*****

unit UntDolumParametreleri;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, Buttons, StdCtrls, Mask, ToolEdit, CurrEdit, ComCtrls, ExtCtrls,
    RXSpin, AdvToolBar, AdvToolBarStylers;

type
    TFrmDolumParametreleri = class(TForm)
        PageControl1: TPageControl;
        TabSheet1: TTabSheet;
        Label2: TLabel;

```

```

Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
Label6: TLabel;
SEAgrega: TRxSpinEdit;
SEMixer: TRxSpinEdit;
SEBekle03: TRxSpinEdit;
SEBekle02: TRxSpinEdit;
SEBekle01: TRxSpinEdit;
Panel4: TPanel;
Image9: TImage;
Panel3: TPanel;
Image2: TImage;
Label16: TLabel;
Label17: TLabel;
AdvDockPanel1: TAdvDockPanel;
AdvToolBar1: TAdvToolBar;
BtnKapat: TAdvToolBarButton;
BtnEkle: TAdvToolBarButton;
AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
StatusBar1: TStatusBar;
procedure FormCreate(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtnKapatClick(Sender: TObject);
procedure BtnEkleClick(Sender: TObject);
procedure Panel3MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
private
  { Private declarations }
public
  { Public declarations }
end;

var
  FrmDolumParametreleri: TFrmDolumParametreleri;

implementation

uses UntDm, DB, ADODB, UntGlobal;

{$R *.dfm}

procedure TFrmDolumParametreleri.FormCreate(Sender: TObject);
begin
  Caption := ProCaption ;
end;
// ***
procedure TFrmDolumParametreleri.FormShow(Sender: TObject);
var
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);

```

```

SetWindowRgn(handle, region, true);
KullaniciHaklari(Self.Tag,AdvToolBar1);
with dm.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Select * From Dolum');
Open;
SEBekle01.Value :=FieldByName('Bekle01').AsFloat;
SEBekle02.Value :=FieldByName('Bekle02').AsFloat;
SEBekle03.Value :=FieldByName('Bekle03').AsFloat;
SEMixer.Value :=FieldByName('Mixer').AsFloat;
SEAgrega.Value :=FieldByName('Agrega').AsFloat;
end;
end;
// ***
procedure TFrmDolumParametreleri.FormKeyDown(Sender: TObject;
var Key: Word; Shift: TShiftState);
begin
if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
if key=vk_F10 then btnKapat.Click;
end;

// ***
procedure TFrmDolumParametreleri.BtnKapatClick(Sender: TObject);
begin
Self.Close;
end;
// ***
procedure TFrmDolumParametreleri.BtnEkleClick(Sender: TObject);
begin
if Onay('Dolum Parametreleri Kayıt İşlemi;'+#13+'Devam Etmek İstiyor musunuz ?')=False then exit;
with dm.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Select * From Dolum');
Open;
if RecordCount=0 then
begin
Close;
SQL.Clear;
SQL.Add('Insert Into Dolum (Bekle01,Bekle02,Bekle03,Mixer,Agrega) ');
SQL.Add('Values (:Bekle01,:Bekle02,:Bekle03,:Mixer,:Agrega)');
Parameters.ParamByName('Bekle01').Value := SEBekle01.Value ;
Parameters.ParamByName('Bekle02').Value := SEBekle02.Value ;
Parameters.ParamByName('Bekle03').Value := SEBekle03.Value ;
Parameters.ParamByName('Mixer').Value := SEMixer.Value ;
Parameters.ParamByName('Agrega').Value := SEAgrega.Value ;
Prepared;
ExecSQL;
end
else
begin

```

```

Close;
SQL.Clear;
SQL.Add('Update Dolum Set Bekle01=:Bekle01,Bekle02=:Bekle02,');
SQL.Add('Bekle03=:Bekle03,Mixer=:Mixer,Agrega=:Agrega ');
Parameters.ParamByName('Bekle01').Value := SEBekle01.Value ;
Parameters.ParamByName('Bekle02').Value := SEBekle02.Value ;
Parameters.ParamByName('Bekle03').Value := SEBekle03.Value ;
Parameters.ParamByName('Mixer').Value := SEMixer.Value ;
Parameters.ParamByName('Agrega').Value := SEAgrega.Value ;
Prepared;
ExecSQL;
end;
end;
Self.Close;
end;

```

```

// ***
procedure TFrmDolumParametreleri.Panel3MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ReleaseCapture;
  self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

```

```

*****

```

```

unit UntEmirDegistir;

```

```

interface

```

```

uses

```

```

  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, cxDataStorage, cxEdit, DB, cxDBData, ToolEdit, ComCtrls, cxGridLevel,
  cxGridCustomTableView, cxGridTableView, cxGridDBTableView, cxClasses,
  cxControls, cxGridCustomView, cxGrid, ADODB, ExtCtrls, cxStyles,
  cxCustomData, cxGraphics, cxFilter, cxData, StdCtrls, DBCtrls, CurrEdit,
  Mask ;

```

```

type

```

```

  TFrmEmirDegistir = class(TForm)
    Panel3: TPanel;
    Panel5: TPanel;
    Image9: TImage;
    Image10: TImage;
    BtnDegistir: TSpeedButton;
    Label11: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    BtnKapat: TSpeedButton;
    DSQryRecete: TDataSource;
    QryRecete: TADOQuery;
    lblUretimNo: TLabel;
    cxGrid1: TcxGrid;

```

```

cxGrid1DBTableView1: TcxGridDBTableView;
cxGrid1DBTableView1RecTarih: TcxGridDBCColumn;
cxGrid1DBTableView1SpektNo: TcxGridDBCColumn;
cxGrid1DBTableView1ReceteKod: TcxGridDBCColumn;
cxGrid1DBTableView1ReceteAdi: TcxGridDBCColumn;
cxGrid1DBTableView1Agrega01: TcxGridDBCColumn;
cxGrid1DBTableView1Agrega02: TcxGridDBCColumn;
cxGrid1DBTableView1Agrega03: TcxGridDBCColumn;
cxGrid1DBTableView1Kum: TcxGridDBCColumn;
cxGrid1DBTableView1Cimento: TcxGridDBCColumn;
cxGrid1DBTableView1Su: TcxGridDBCColumn;
cxGrid1DBTableView1Katki01: TcxGridDBCColumn;
cxGrid1DBTableView1Katki02: TcxGridDBCColumn;
cxGrid1DBTableView1ToplamMiktar: TcxGridDBCColumn;
cxGrid1DBTableView1Sure: TcxGridDBCColumn;
cxGrid1DBTableView1Hazirlayan: TcxGridDBCColumn;
cxGrid1DBTableView1Onaylayan: TcxGridDBCColumn;
cxGrid1Level1: TcxGridLevel;
StatusBar1: TStatusBar;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label5: TLabel;
Label6: TLabel;
Label8: TLabel;
UretimSaati: TDateTimePicker;
UretimTarihi: TDateEdit;
txtTekrar: TCurrencyEdit;
Panel2: TPanel;
Label4: TLabel;
Label2: TLabel;
txtReceteAdi: TDBEdit;
txtReceteKod: TDBEdit;
DBNavigator1: TDBNavigator;
PageControl2: TPageControl;
TabSheet3: TTabSheet;
Label3: TLabel;
Label10: TLabel;
Label12: TLabel;
Label7: TLabel;
BtnUrunAra: TSpeedButton;
txtSpektNo: TEdit;
txtUrunKod: TEdit;
txtUrunAd: TEdit;
txtUrunTip: TEdit;
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure FormShow(Sender: TObject);
procedure BtnUrunAraClick(Sender: TObject);
procedure BtnDegistirClick(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure Panel5MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
private
  { Private declarations }
public
  { Public declarations }

```

```

    iptal:Boolean;
    ActUretimNo:String;
    procedure SetReceteView();
end;

var
    FrmEmirDegistir: TFrmEmirDegistir;

implementation

uses UntDm, UntDlgUrunArama, UntUyari, UntGlobal;

{$R *.dfm}

procedure TFrmEmirDegistir.SetReceteView();
begin
    with QryRecete do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * , (Saat + ":" + Dakika ) as Sure, ');
        SQL.Add('(HPerAdi + " " + HPerSoyadi ) as Hazirlayan, ');
        SQL.Add('(OPerAdi + " " + OPerSoyadi ) as Onaylayan ');
        SQL.Add(' From Recete Where SpektNo=:actSpektno');
        Parameters.ParamByName('actSpektno').Value:=txtSpektNo.Text;
        Prepared ;
        open;
        (FieldByName('Agrega01') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Agrega02') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Agrega03') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Kum') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Cimento') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Su') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Katki01') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Katki02') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('ToplamMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
    end;
end;
// ***

procedure TFrmEmirDegistir.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if Key=Vk_Escape then BtnKapat.Click;
    if Key=vk_F5 then BtnDegistir.Click;
end;
// ***

procedure TFrmEmirDegistir.FormShow(Sender: TObject);
var
    region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 23,23);
    SetWindowRgn(handle, region, true);
    with QryRecete do
    begin

```

```

Close;
SQL.Clear;
SQL.Add('Select a.*,b.UrunTip,b.SpektNo From Uretim as a,Urunler as b ');
SQL.Add(' Where (a.UrunKod=b.UrunKod) and (a.UretimNo=:ActUretimNo)');
Parameters.ParamByName('ActUretimNo').Value:=actUretimNo;
Prepared ;
Open;
lblUretimNo.caption:=FieldByName('UretimNo').AsString ;
txtUrunKod.Text :=FieldByName('UrunKod').AsString ;
txtUrunAd.Text :=FieldByName('UrunAdi').AsString ;
txtUrunTip.Text :=FieldByName('UrunTip').AsString ;
txtSpektNo.Text :=FieldByName('SpektNo').AsString ;
txtReceteKod.Text :=FieldByName('ReceteKod').AsString ;
txtReceteAdi.Text :=FieldByName('ReceteAdi').AsString ;
txtTekrar.Text :=FieldByName('PTekrar').AsString ;
UretimTarihi.Text :=FieldByName('STarih').AsString ;
UretimSaati.Time :=StrToTime(FieldByName('SSaat').AsString) ;
SetReceteView;
end;
end;
// ***
procedure TFrEmirDegistir.BtnUrunAraClick(Sender: TObject);
begin
FrmDlgUrunArama:=TFrmDlgUrunArama.Create(Self);
FrmDlgUrunArama.ShowModal;
if not FrmDlgUrunArama.iptal then
begin
txtUrunKod.Text :=FrmDlgUrunArama.ActUrunKod;
txtUrunAd.Text :=FrmDlgUrunArama.ActUrunAd;
txtUrunTip.Text :=FrmDlgUrunArama.ActUrunTip;
txtSpektNo.Text :=FrmDlgUrunArama.ActSpektNo;
SetReceteView;
end;
FrmDlgUrunArama.Release;
end;
// ***
procedure TFrEmirDegistir.BtnDegistirClick(Sender: TObject);
begin
if txtUrunKod.Text="" then
begin
Hata('Lütfen Ürün Seçimi Yapınız. ');
exit;
end;
if txtReceteKod.Text="" then
begin
Hata('Lütfen Ürün için Reçete Seçimi Yapınız. ');
exit;
end;
if txtTekrar.Value<=0 then
begin
Hata('Lütfen Tekrar Sayısını Giriniz. ');
exit;
end;
with Dm.Qry1 do
begin
Close;

```



```

SQL.Clear;
SQL.Add('Select * From Uretim Where UretimNo=:ActUretimNo');
Parameters.ParamByName('ActUretimNo').Value:=actUretimNo;
Prepared ;
Open;
Edit;
FieldByName('UrunKod').AsString      :=txtUrunKod.Text;
FieldByName('UrunAdi').AsString      :=txtUrunAd.Text;
FieldByName('ReceteKod').AsString     :=txtReceteKod.Text;
FieldByName('ReceteAdi').AsString     :=txtReceteAdi.Text;
FieldByName('RStdMiktar').AsString   :=
QryRecete.FieldByName('ToplamMiktar').AsString;
FieldByName('PTekrar').AsString       :=txtTekrar.Text;
FieldByName('GMiktar').AsString       :='0';
FieldByName('GTekrar').AsString       :='0';
FieldByName('STarih').AsString        :=UretimTarihi.Text;
FieldByName('SSaat').AsString         :=FormatDateTime('hh:mm',UretimSaati.Time);
FieldByName('ETarih').AsString        :='--.--.----';
FieldByName('ESaat').AsString         :='00:00';
Post;
end;
iptal:=False;
self.Close;
end;
// ***
procedure TFrmEmirDegistir.BtnKapatClick(Sender: TObject);
begin
    iptal:=True;
    Self.Close;
end;
// ***
procedure TFrmEmirDegistir.Panel5MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntEmirEkle;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, Buttons, cxStyles, cxCustomData, cxGraphics, cxFilter, cxData, cxDataStorage,
    cxEdit, cxDBData, cxGridLevel, cxClasses, cxControls, cxGridCustomView,
    cxGridCustomTableView, cxGridTableView, cxGridDBTableView, cxGrid, DB,
    ADODB, StdCtrls, DBCtrls, CurrEdit, Mask, ToolEdit, ComCtrls, ExtCtrls;

type
    TFrmEmirEkle = class(TForm)
        Panel3: TPanel;
        Panel5: TPanel;

```

```

Image9: TImage;
Image10: TImage;
BtnEkle: TSpeedButton;
Label11: TLabel;
Label13: TLabel;
Label14: TLabel;
lblUretimNo: TLabel;
StatusBar1: TStatusBar;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label5: TLabel;
Label6: TLabel;
Label8: TLabel;
UretimSaati: TDateTimePicker;
UretimTarihi: TDateEdit;
txtTekrar: TCurrencyEdit;
Panel2: TPanel;
Label4: TLabel;
Label2: TLabel;
txtReceteAdi: TDBEdit;
txtReceteKod: TDBEdit;
DBNavigator1: TDBNavigator;
PageControl2: TPageControl;
TabSheet3: TTabSheet;
Label3: TLabel;
Label10: TLabel;
Label12: TLabel;
Label7: TLabel;
BtnUrunAra: TSpeedButton;
txtSpektNo: TEdit;
txtUrunKod: TEdit;
txtUrunAd: TEdit;
txtUrunTip: TEdit;
BtnKapat: TSpeedButton;
cxGrid1DBTableView1: TcxGridDBTableView;
cxGrid1Level1: TcxGridLevel;
cxGrid1: TcxGrid;
QryRecete: TADOQuery;

Hata('Lütfen Tekrar Sayısını Giriniz. ');
exit;
end;
with Dm.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Select * From Uretim Where UretimNo=:ActUretimNo');
Parameters.ParamByName('ActUretimNo').Value:=actUretimNo;
Prepared ;
Open;
Edit;
FieldByName('UrunKod').AsString      :=txtUrunKod.Text;
FieldByName('UrunAdi').AsString      :=txtUrunAd.Text;
FieldByName('ReceteKod').AsString    :=txtReceteKod.Text;
FieldByName('ReceteAdi').AsString    :=txtReceteAdi.Text;
FieldByName('RStdMiktar').AsString   :=

```

```

QryRecete.FieldName('ToplamMiktar').AsString;
  FieldByName('PTekrar').AsString      :=txtTekrar.Text;
  FieldByName('GMiktar').AsString      :='0';
  FieldByName('GTekrar').AsString      :='0';
  FieldByName('STarih').AsString       :=UretimTarihi.Text;
  FieldByName('SSaat').AsString        :=FormatDateTime('hh:mm',UretimSaati.Time);
  FieldByName('ETarih').AsString       :='---.---.---';
  FieldByName('ESaat').AsString        :='00:00';
Post;
end;
iptal:=False;
self.Close;
end;
// ***
procedure TFrmEmirDegistir.BtnKapatClick(Sender: TObject);
begin
  iptal:=True;
  Self.Close;
end;
// ***
procedure TFrmEmirDegistir.Panel5MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ReleaseCapture;
  self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntEmirEkle;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, cxStyles, cxCustomData, cxGraphics, cxFilter, cxData, cxDataStorage,
  cxEdit, cxDBData, cxGridLevel, cxClasses, cxControls, cxGridCustomView,
  cxGridCustomTableView, cxGridTableView, cxGridDBTableView, cxGrid, DB,
  ADODB, StdCtrls, DBCtrls, CurrEdit, Mask, ToolEdit, ComCtrls, ExtCtrls;

type
  TFrmEmirEkle = class(TForm)
    Panel3: TPanel;
    Panel5: TPanel;
    Image9: TImage;
    Image10: TImage;
    BtnEkle: TSpeedButton;
    Label11: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    lblUretimNo: TLabel;
    StatusBar1: TStatusBar;
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    Label5: TLabel;

```

```

Label6: TLabel;
Label8: TLabel;
UretimSaati: TDateTimePicker;
UretimTarihi: TDateEdit;
txtTekrar: TCurrencyEdit;
Panel2: TPanel;
Label4: TLabel;
Label2: TLabel;
txtReceteAdi: TDBEdit;
txtReceteKod: TDBEdit;
DBNavigator1: TDBNavigator;
PageControl2: TPageControl;
TabSheet3: TTabSheet;
Label3: TLabel;
Label10: TLabel;
Label12: TLabel;
Label7: TLabel;
BtnUrunAra: TSpeedButton;
txtSpektNo: TEdit;
txtUrunKod: TEdit;
txtUrunAd: TEdit;
txtUrunTip: TEdit;
BtnKapat: TSpeedButton;
cxGrid1DBTableView1: TcxGridDBTableView;
cxGrid1Level1: TcxGridLevel;
cxGrid1: TcxGrid;
QryRecete: TADOQuery;
DSQryRecete: TDataSource;
cxGrid1DBTableView1Sure: TcxGridDBCColumn;
cxGrid1DBTableView1Hazirlayan: TcxGridDBCColumn;
cxGrid1DBTableView1Onaylayan: TcxGridDBCColumn;
cxGrid1DBTableView1SpektNo: TcxGridDBCColumn;
cxGrid1DBTableView1ReceteKod: TcxGridDBCColumn;
cxGrid1DBTableView1ReceteAdi: TcxGridDBCColumn;
cxGrid1DBTableView1RecTarih: TcxGridDBCColumn;
cxGrid1DBTableView1Agrega01: TcxGridDBCColumn;
cxGrid1DBTableView1Agrega02: TcxGridDBCColumn;
cxGrid1DBTableView1Agrega03: TcxGridDBCColumn;
cxGrid1DBTableView1Kum: TcxGridDBCColumn;
cxGrid1DBTableView1Cimento: TcxGridDBCColumn;
cxGrid1DBTableView1Su: TcxGridDBCColumn;
cxGrid1DBTableView1Katki01: TcxGridDBCColumn;
cxGrid1DBTableView1Katki02: TcxGridDBCColumn;
cxGrid1DBTableView1ToplamMiktar: TcxGridDBCColumn;
procedure FormShow(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure Image9MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure BtnEkleClick(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure BtnUrunAraClick(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }

```

```

    iptal:Boolean;
    procedure SetReceteView();

end;

var
    FrmEmirEkle: TFrmEmirEkle;

implementation

uses UntDlgUrunArama, UntDm,StrUtils, UntOnay, UntUyari, UntGlobal;

{$R *.dfm}

procedure TFrmEmirEkle.SetReceteView();
begin
    with QryRecete do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * , (Saat + ":" + Dakika ) as Sure, ');
        SQL.Add('(HPerAdi + " " + HPerSoyadi ) as Hazirlayan, ');
        SQL.Add('(OPerAdi + " " + OPerSoyadi ) as Onaylayan ');
        SQL.Add(' From Recete Where SpektNo=:actSpektno');
        Parameters.ParamByName('actSpektno').Value:=FrmEmirEkle.txtSpektNo.Text;
        Prepared ;
        open;
        (FieldByName('Agrega01') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Agrega02') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Agrega03') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Kum') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Cimento') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Su') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Katki01') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('Katki02') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('ToplamMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
    end;
end;
// ***
procedure TFrmEmirEkle.FormShow(Sender: TObject);
var
    tmp:integer;
    region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    iptal:=True;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 23,23);
    SetWindowRgn(handle, region, true);
    UretimTarihi.Date:=Date;
    UretimSaati.Time:=Now;
    with DM.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select URLastNumber From LastNumbers');
    end;
end;

```

```

Prepared ;
Open;
if FieldByName('URLastNumber').AsInteger=0 then
    lblUretimNo.Caption:='URT00001'
else
begin
    tmp:=FieldByName('URLastNumber').AsInteger+1;
    lblUretimNo.Caption:='URT'+ RightStr('00000'+IntToStr(tmp),5);
end;
end;
end;
// ***
procedure TFrmEmirEkle.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
    if Key=vk_Escape then BtnKapat.Click;
end;
// ***
procedure TFrmEmirEkle.Image9MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
// ***
procedure TFrmEmirEkle.BtnEkleClick(Sender: TObject);
begin
    if txtUrunKod.Text="" then
begin
        FrmUyari:=TFrmUyari.Create(Self);
        FrmUyari.lblCaption.Caption:='Lütfen Ürün Seçimi Yapınız.';
        FrmUyari.ShowModal;
        exit;
end;
    if txtReceteKod.Text="" then
begin
        FrmUyari:=TFrmUyari.Create(Self);
        FrmUyari.lblCaption.Caption:='Lütfen Ürün için Reçete Seçimi Yapınız.';
        FrmUyari.ShowModal;
        exit;
end;
    if txtTekrar.Value<=0 then
begin
        FrmUyari:=TFrmUyari.Create(Self);
        FrmUyari.lblCaption.Caption:='Lütfen Tekrar Sayısını Giriniz.';
        FrmUyari.ShowModal;
        exit;
end;
    with DM.Qry1 do
begin
        Close;
        SQL.Clear;
        SQL.Add('Insert Into Uretim (UretimNo,UrunKod,UrunAdi,ReceteKod,ReceteAdi,RStdMiktar, ' ');
        SQL.Add('PTekrar,GMiktar,GTekrar,STarih,SSaat,ETarih,ESaat ' ');
        SQL.Add('Values(:UretimNo,:UrunKod,:UrunAdi,:ReceteKod,:ReceteAdi,:RStdMiktar, ' ');

```

```

SQL.Add('PTekrar,:GMiktar,:GTekrar,:STarih,:SSaat,:ETarih,:ESaat' );
Parameters.ParamByName('UretimNo').Value :=lblUretimNo.caption;
Parameters.ParamByName('UrunKod').Value :=txtUrunKod.Text;
Parameters.ParamByName('UrunAdi').Value :=txtUrunAd.Text;
Parameters.ParamByName('ReceteKod').Value :=txtReceteKod.Text;
Parameters.ParamByName('ReceteAdi').Value :=txtReceteAdi.Text;
Parameters.ParamByName('RStdMiktar').Value :=QryRecete.FieldByName('ToplamMiktar').AsString;
Parameters.ParamByName('PTekrar').Value :=txtTekrar.Text;
Parameters.ParamByName('GMiktar').Value :='0';
Parameters.ParamByName('GTekrar').Value :='0';
Parameters.ParamByName('STarih').Value :=UretimTarihi.Text;
Parameters.ParamByName('SSaat').Value :=FormatDateTime('hh:mm',UretimSaati.Time);
Parameters.ParamByName('ETarih').Value :='--.--.----';
Parameters.ParamByName('ESaat').Value :='00:00';
Prepared ;
ExecSQL;
Close;
SQL.Clear;
SQL.Add('Update LastNumbers Set URLastNumber=URLastNumber+1 ');
Prepared ;
ExecSQL;
end;
iptal:=False;
self.Close;
end;
// ***
procedure TFrmEmirEkle.BtnKapatClick(Sender: TObject);
begin
iptal:=True;
Self.Close;
end;
// ***
procedure TFrmEmirEkle.BtnUrunAraClick(Sender: TObject);
begin
FrmDlgUrunArama:=TFrmDlgUrunArama.Create(Self);
FrmDlgUrunArama.ShowModal;
if not FrmDlgUrunArama.iptal then
begin
txtUrunKod.Text :=FrmDlgUrunArama.ActUrunKod;
txtUrunAd.Text :=FrmDlgUrunArama.ActUrunAd;
txtUrunTip.Text :=FrmDlgUrunArama.ActUrunTip;
txtSpektNo.Text :=FrmDlgUrunArama.ActSpektNo;
SetReceteView;
end;
FrmDlgUrunArama.Release;
end;
end.

*****

unit UntEmirSil;

interface

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

```

Dialogs, DBCtrls, CurrEdit, Mask, ToolEdit, ComCtrls, StdCtrls, Buttons, ExtCtrls;

type

```
TfrmEmirSil = class(TForm)
  Panel5: TPanel;
  Image9: TImage;
  Image10: TImage;
  BtnSil: TSpeedButton;
  Label11: TLabel;
  Label13: TLabel;
  Label14: TLabel;
  BtnKapat: TSpeedButton;
  lblUretimNo: TLabel;
  PageControl1: TPageControl;
  TabSheet1: TTabSheet;
  Label5: TLabel;
  Label6: TLabel;
  Label8: TLabel;
  Label4: TLabel;
  Label2: TLabel;
  UretimSaati: TDateTimePicker;
  UretimTarihi: TDateEdit;
  txtTekrar: TCurrencyEdit;
  txtReceteKod: TEdit;
  txtReceteAdi: TEdit;
  PageControl2: TPageControl;
  TabSheet3: TTabSheet;
  Label3: TLabel;
  Label10: TLabel;
  Label12: TLabel;
  Label7: TLabel;
  txtSpektNo: TEdit;
  txtUrunKod: TEdit;
  txtUrunAd: TEdit;
  txtUrunTip: TEdit;
  StatusBar1: TStatusBar;
  procedure FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
  procedure BtnCloseClick(Sender: TObject);
  procedure FormShow(Sender: TObject);
  procedure BtnSilClick(Sender: TObject);
  procedure BtnKapatClick(Sender: TObject);
  procedure Image9MouseDown(Sender: TObject; Button: TMouseButton;
    Shift: TShiftState; X, Y: Integer);
private
  { Private declarations }
public
  { Public declarations }
  ActUretimNo:String;
  iptal:Boolean;
end;
```

var

```
FrmEmirSil: TfrmEmirSil;
```



implementation

uses UntDm, ADODB, UntGlobal;

{ \$R \*.dfm }

*procedure TFrmEmirSil.FormKeyDown(Sender: TObject; var Key: Word;  
Shift: TShiftState);*

*begin*

if Key=Vk\_Escape then BtnKapat.Click;

if Key=vk\_F6 then BtnSil.Click;

*end;*

*// \*\*\**

*procedure TFrmEmirSil.BtnCloseClick(Sender: TObject);*

*begin*

self.Close;

*end;*

*// \*\*\**

*procedure TFrmEmirSil.FormShow(Sender: TObject);*

*var*

region: HRgn;

*begin*

StatusBar1.SimplePanel:=True;

StatusBar1.SimpleText:=ProCaption;

region:=CreateRoundRectRgn(1, 1,Width ,Height , 23,23);

SetWindowRgn(handle, region, true);

iptal:=True;

with DM.QryArama do

*begin*

Close;

SQL.Clear;

SQL.Add('Select a.\*,b.UrunTip,b.SpektNo From Uretim as a,Urunler as b ');

SQL.Add(' Where (a.UrunKod=b.UrunKod) and (a.UretimNo=:ActUretimNo)');

Parameters.ParamByName('ActUretimNo').Value:=actUretimNo;

Prepared ;

Open;

IblUretimNo.caption:=FieldByName('UretimNo').AsString ;

txtUrunKod.Text :=FieldByName('UrunKod').AsString ;

txtUrunAd.Text :=FieldByName('UrunAdi').AsString ;

txtReceteKod.Text :=FieldByName('ReceteKod').AsString ;

txtReceteAdi.Text :=FieldByName('ReceteAdi').AsString ;

txtUrunTip.Text :=FieldByName('UrunTip').AsString ;

txtSpektNo.Text :=FieldByName('SpektNo').AsString ;

txtReceteKod.Text :=FieldByName('ReceteKod').AsString ;

txtReceteAdi.Text :=FieldByName('ReceteAdi').AsString ;

txtTekrar.Text :=FieldByName('PTekrar').AsString ;

UretimTarihi.Text :=FieldByName('STarih').AsString ;

UretimSaati.Time :=StrToTime(FieldByName('SSaat').AsString) ;

*end;*

*end;*

*// \*\*\**

*procedure TFrmEmirSil.BtnSilClick(Sender: TObject);*

*begin*

if Onay('Üretim Emri Silinecektir'+#13+'Devam Etmek İstiyor musunuz ?')=False then Exit;

with DM.Qry1 do

*begin*

```

Close;
SQL.Clear;
SQL.Add('Delete * From Uretim Where UretimNo=:ActUretimNo');
Parameters.ParamByName('ActUretimNo').Value:=actUretimNo;
Prepared ;
ExecSQL;
end;
iptal:=False;
self.Close;
end;
// ***
procedure TFrmEmirSil.BtnKapatClick(Sender: TObject);
begin
    iptal:=True;
    Self.Close;
end;
// ***
procedure TFrmEmirSil.Image9MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntHaklar;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, cxStyles, cxCustomData, cxGraphics, cxFilter, cxData,
    cxDataStorage, cxEdit, DB, cxDBData, ADODB, cxGridLevel,
    cxGridCustomTableView, cxGridTableView, cxGridDBTableView, cxClasses,
    cxControls, cxGridCustomView, cxGrid, AdvToolBar, StdCtrls, ExtCtrls,
    AdvToolBarStylers, Grids, BaseGrid, AdvGrid, Buttons, ComCtrls, dxPSGlbl,
    dxPSUtl, dxPSEngn, dxPrnPg, dxBknd, dxWrap, dxPrnDev, dxPSCompsProvider,
    dxPSFillPatterns, dxPSEdgePatterns, dxPSCore, dxPSCxCommon, dxPSCxGridLnk,
    dxPSBaseGridLnk, dxPSGrLnks, dxPSStdGrLnk;

type
    TFrmHaklar = class(TForm)
        AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
        Panel4: TPanel;
        Image9: TImage;
        Panel3: TPanel;
        Image2: TImage;
        Label16: TLabel;
        Label17: TLabel;
        AdvDockPanel1: TAdvDockPanel;
        AdvToolBar1: TAdvToolBar;
        BtnKapat: TAdvToolBarButton;
        BtnEkle: TAdvToolBarButton;
        BtnYazidr: TAdvToolBarButton;
    end;

```

```

Panel1: TPanel;
GrdHaklar: TAdvStringGrid;
pnlIslem: TPanel;
Label2: TLabel;
Label3: TLabel;
Label4: TLabel;
Label1: TLabel;
BtnArama: TSpeedButton;
txtPerAdi: TEdit;
txtPerSoyadi: TEdit;
txtKullaniciAdi: TEdit;
txtPerNo: TEdit;
StatusBar1: TStatusBar;
PrintPreview: TdxComponentPrinter;
PrintPreviewLink1: TdxStringGridReportLink;
procedure GrdHaklarCanEditCell(Sender: TObject; ARow, ACol: Integer;
  var CanEdit: Boolean);
procedure GrdHaklarGetAlignment(Sender: TObject; ARow, ACol: Integer;
  var HAlign: TAlignment; var VAlign: TVAlignment);
procedure BtnKapatClick(Sender: TObject);
procedure BtnEkleClick(Sender: TObject);
procedure BtnAramaClick(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure Image9MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure BtnYazdirClick(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
  procedure Haklar();
end;

var
  FrmHaklar: TFrmHaklar;

implementation

uses UntDm, UntDlgPersonalSearch, UntGlobal;

{$R *.dfm}

procedure TFrmHaklar.GrdHaklarCanEditCell(Sender: TObject; ARow,
  ACol: Integer; var CanEdit: Boolean);
begin
  if Acol <> 2 then
    CanEdit:=True
  else
    CanEdit:=False;
end;
// **
procedure TFrmHaklar.GrdHaklarGetAlignment(Sender: TObject; ARow,
  ACol: Integer; var HAlign: TAlignment; var VAlign: TVAlignment);
begin
  if (Acol<>2) and (ARow>0) then HAlign:=taCenter;
end;

```

```

// ***
procedure TFrmHaklar.BtmKapatClick(Sender: TObject);
begin
    self.Close;
end;
// ***

procedure TFrmHaklar.BtmEkleClick(Sender: TObject);
var
    i:integer;
begin
    if txtPerNo.Text="" then
    begin
        Hata('Lütfen Personel Seçimi Yapınız. ');
        Exit;
    end;
    with dm.Qrylist do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Delete * from Haklar Where ');
        SQL.Add('PerNo='+txtPerNo.Text);
        ExecSQL;
        for i :=1 to GrdHaklar.RowCount-1 do
        begin
            if (GrdHaklar.Cells[1,i]='Y') then
            begin
                Close;
                SQL.Clear;
                SQL.Add('Insert into Haklar ');
                SQL.Add('('PerNo,Sira,Gorme,Kayit,Degistir,Sil,Yazdir,TagNo Values');
                SQL.Add('(:PerNo,:Sira,:Gorme,:Kayit,:Degistir,:Sil,:Yazdir,:TagNo)');
                Parameters.ParamByName('PerNo').Value :=txtPerNo.Text;
                Parameters.ParamByName('Sira').Value :=i;
                Parameters.ParamByName('Gorme').Value :=GrdHaklar.Cells[1,i];
                Parameters.ParamByName('Kayit').Value :=GrdHaklar.Cells[3,i];
                Parameters.ParamByName('Degistir').Value:=GrdHaklar.Cells[4,i];
                Parameters.ParamByName('Sil').Value :=GrdHaklar.Cells[5,i];
                Parameters.ParamByName('Yazdir').Value :=GrdHaklar.Cells[6,i];
                Parameters.ParamByName('TagNo').Value :=GrdHaklar.Cells[7,i];
                Prepared;
                ExecSQL;
            end;
        end;
    end;
end;
// ***

procedure TFrmHaklar.BtmAramaClick(Sender: TObject);
begin
    FrmDlgPersonalSearch:=TFrmDlgPersonalSearch.Create(Self);
    FrmDlgPersonalSearch.ShowModal;
    if not FrmDlgPersonalSearch.iptal then
    begin
        txtPerAdi.Text :=FrmDlgPersonalSearch.APerAdi ;
        txtPerSoyadi.Text :=FrmDlgPersonalSearch.APerSoyadi;
        txtPerNo.Text :=FrmDlgPersonalSearch.APerNo;
        txtKullaniciAdi.Text:=FrmDlgPersonalSearch.AKullanici;
    end;
end;

```

```

    Haklar;
end;
FrmDlgPersonalSearch.Release;
end;
// ***
procedure TFrmHaklar.Haklar;
var
    i,r,y:integer;
    tmp:string;
    modul:array of integer;
begin
    GrdHaklar.RemoveAllNodes;
    GrdHaklar.ClearRows(1,GrdHaklar.RowCount);
    GrdHaklar.MergeCells(0,0,2,1);
    with dm.Qrylist do
    begin
        close;
        SQL.Clear;
        SQL.Add('Select Count(TanimNo) as toplam From Moduller ');
        SQL.Add(' Group By ModulNo Order by ModulNo Asc');
        Open;
        SetLength(modul,RecordCount+1);
        for i:=1 to RecordCount do
        begin
            modul[i]:=FieldByName('Toplam').AsInteger;
            next;
        end;
        close;
        SQL.Clear;
        SQL.Add('Select ModulNo,TanimNo,Tanim From Moduller ');
        SQL.Add('Order by ModulNo,TanimNo Asc');
        Open;
        GrdHaklar.RowCount:=RecordCount+1;
        for i:=1 to RecordCount do
        begin
            if tmp<> FieldByName('ModulNo').AsString then
            begin
                tmp:=FieldByName('ModulNo').AsString;
                r:=r+1;
                GrdHaklar.AddNode(i,modul[r]);
                GrdHaklar.AddCheckBox(1,i,False,True);
                GrdHaklar.RowColor[i]:=clInfoBk;
            end;
            if FieldByName('TanimNo').AsInteger <> 0 then
            begin
                GrdHaklar.AddCheckBox(1,i,False,True);
                GrdHaklar.AddCheckBox(3,i,False,True);
                GrdHaklar.AddCheckBox(4,i,False,True);
                GrdHaklar.AddCheckBox(5,i,False,True);
                GrdHaklar.AddCheckBox(6,i,False,True);
            end;
            GrdHaklar.Cells[2,i]:=FieldByName('Tanim').AsString;
            GrdHaklar.Cells[7,i]:=FieldByName('ModulNo').AsString+FieldByName('TanimNo').
            AsString;
            next;
        end;
    end;
end;

```

```

Close;
SQL.Clear;
SQL.Add('Select * from Haklar Where ');
SQL.Add('PerNo='+ txtPerNo.Text );
Open;
for i :=1 to RecordCount do
begin
    y:=FieldByName('sira').AsInteger;
    GrdHaklar.Cells[1,y]:=FieldByName('Gorme').AsString;
    GrdHaklar.Cells[3,y]:=FieldByName('Kayit').AsString;
    GrdHaklar.Cells[4,y]:=FieldByName('Degistir').AsString;
    GrdHaklar.Cells[5,y]:=FieldByName('Sil').AsString;
    GrdHaklar.Cells[6,y]:=FieldByName('Yazdir').AsString;
    GrdHaklar.Cells[7,y]:=FieldByName('TagNo').AsString;
    next;
end;
end;
end;
// ***
procedure TFrmHaklar.FormShow(Sender: TObject);
var
    region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
    SetWindowRgn(handle, region, true);
end;
// ***
procedure TFrmHaklar.Image9MouseDown(Sender: TObject; Button: TMouseButton;
    Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
// ***
procedure TFrmHaklar.BtnYazidrClick(Sender: TObject);
var
    capStr:string;
begin
    capStr:='Per.No:'+txtPerNo.Text+'   Kullanıcı Adı :'+ txtKullaniciAdi.Text+ #13+
        'Adı:'+txtPerAdi.Text+'   Soyadı:'+txtPerSoyadi.Text+#13+Label16.Caption;
    PrintPreviewLink1.ReportTitleText:=capStr;
    PrintPreviewLink1.PrinterPage.Orientation:=poPortrait;
    PrintPreviewLink1.PrinterPage.PageHeader.RightTitle .Clear;
    PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi:'+
        FormatDateTime('dd.mm.yyyy',date));
    PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Clear;
    PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
    PrintPreview.Preview(True,PrintPreviewLink1);
end;
end.

*****

unit UntLogin;

```

interface

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,  
StdCtrls, ExtCtrls, Buttons, Db, DBTables, RxGIF, Animate,untGlobal,GIFCtrl,  
ComCtrls;

type

```
TFrmLogin = class(TForm)
  Label3: TLabel;
  Label4: TLabel;
  Label1: TLabel;
  txtUsername: TEdit;
  txtPassword: TEdit;
  Image5: TImage;
  BtnStart: TSpeedButton;
  SpeedButton1: TSpeedButton;
  Label2: TLabel;
  StatusBar1: TStatusBar;
  procedure FormCreate(Sender: TObject);
  procedure txtPasswordKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
  procedure FormShow(Sender: TObject);
  procedure BtnStartClick(Sender: TObject);
  procedure SpeedButton1Click(Sender: TObject);
  procedure Image5MouseDown(Sender: TObject; Button: TMouseButton;
    Shift: TShiftState; X, Y: Integer);
  procedure txtUsernameKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
private
  { Private declarations }
public
  { Public declarations }
  function FindUser():Boolean ;
end;
```

var

FrmLogin: TFrmLogin;  
icounter:integer;

implementation

uses untUyari, untDM, ADODB, UntMainForm;  
{ \$R \*.DFM }

```
function TFrmLogin.FindUser():Boolean ;
begin
  SetLength(Hakbilgi,2);
  with dm.Qrylist do
  begin
    close;
    sql.clear;
    sql.Add('Select * From Personel ');
    sql.Add('Where (KullaniciAdi=:KullaniciAdi) and (Sifre=:Sifre) ');
    Parameters.ParamByName('KullaniciAdi').Value:=trim(txtUsername.Text);
    Parameters.ParamByName('Sifre').Value :=trim(txtPassword.Text);
```

```

open;
if IsEmpty then
    Result:=False
else
begin
    Result:=True;
    ActKullanici:=FieldByName('KullaniciAdi').AsString;
    ActPerNo :=FieldByName('PerNo').AsString;
    ActPerAdi :=FieldByName('PerAdi').AsString;
    ActPerSoyadi:=FieldByName('PerSoyadi').AsString;
    ActYonetici :=FieldByName('Yonetici').AsBoolean;
end;
end;
end;
// ***
procedure TFrmLogin.FormCreate(Sender: TObject);
begin
    ProCaption :=' © 2007 Hazır Beton Otomasyon Programı Ver. 1.0.0 ' ;
end;
// ***
procedure TFrmLogin.txtPasswordKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if key=13 then btnStart.Click
end;
// ***
procedure TFrmLogin.FormShow(Sender: TObject);
var
    region: HRgn;
begin
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
    SetWindowRgn(handle, region, true);
end;
// ***
procedure TFrmLogin.BtnStartClick(Sender: TObject);
begin
    if trim(txtUsername.Text)="" then
begin
        Hata('Lütfen Kullanıcı Adını Giriniz. ');
        txtUsername.SetFocus ;
        exit;
end;
// ***
    if trim(txtPassword.Text)="" then
begin
        Hata('Lütfen Şifrenizi Giriniz. ');
        txtPassword.SetFocus ;
        exit;
end;
    if FindUser then
begin
        Hata('Sayın ' + #13 +
            ActPerAdi + ' ' + ActPerSoyadi + ' İyi Günler ... ' + #13 +
            'Hazır Beton Otomasyon Programına Hoş Geldiniz..' + #13 +
            'Çalışmalarınızda Başarılar Dileriz....');
        FrmLogin.Hide;
    end;
end;

```



```

    FrmMainForm:=TFrmMainForm.Create(Self);
    FrmMainForm.ShowModal ;
    Application.Terminate ;
end
else
begin
    icounter :=icounter +1;
    if icounter > 2 then
    begin
        Hata('3 Kez Hatalı Şifre Girişi Yapıldı.'+ #13 +
            'Bu Yüzden Program Sonlandırılacaktır.'+ #13 +
            'Lütfen Kullanıcı Adı ve Şifre Kontrol için ' + #13 +
            'Sistem Yöneticisine Başvurun. ');
        Application.Terminate ;
    end
    else
    begin
        Hata('Hatalı Kullanıcı Adı veya Şifre .' + #13 +
            'Lütfen Kullanıcı Adınızı ve Şifrenizi Kontrol Ediniz.') ;
        txtUsername.Text :=";
        txtPassword.Text :=";
        txtUsername.SetFocus ;
        exit;
    end;
end;
end;
end;
// ***
procedure TFrmLogin.SpeedButton1Click(Sender: TObject);
begin
    Application.Terminate ;
end;
// ***
procedure TFrmLogin.Image5MouseDown(Sender: TObject; Button: TMouseButton;
    Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
// ***
procedure TFrmLogin.txtUsernameKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if key=13 then
    begin
        key:=0;
        Perform(CM_DIALOGKEY,VK_TAB,0);
    end;
end;
end.

*****

unit UntMainForm;

interface

```

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,  
Dialogs, ExtCtrls, Menus, ImgList, ComCtrls,  
StdCtrls, Buttons, jpeg, XPMAN, VrControls, VrLeds, dxNavBarBase,  
dxNavBarCollns, dxNavBar, JzNavBar, VrLcd, VaClasses, VaComm, ToolEdit,  
Mask, CurrEdit ;

type

```
TfrmMainForm = class(TForm)
  ImageList2: TImageList;
  StatusBar1: TStatusBar;
  Panel1: TPanel;
  ImageList1: TImageList;
  XPManifest1: TXPManifest;
  Panel2: TPanel;
  NavBar: TdxNavBar;
  NavBarItem1: TdxNavBarItem;
  VrLcd1: TVrLcd;
  VrLcd2: TVrLcd;
  VrLcd3: TVrLcd;
  VrLcd4: TVrLcd;
  VrLcd6: TVrLcd;
  VrLcd7: TVrLcd;
  VrLcd8: TVrLcd;
  Image1: TImage;
  Label2: TLabel;
  Label3: TLabel;
  Label4: TLabel;
  Label5: TLabel;
  VaComm1: TVaComm;
  Panel3: TPanel;
  PageControl1: TPageControl;
  TabSheet1: TTabSheet;
  PageControl2: TPageControl;
  TabSheet3: TTabSheet;
  Label11: TLabel;
  Label12: TLabel;
  Label13: TLabel;
  Label14: TLabel;
  MtxtSpektNo: TEdit;
  MtxtUrunKod: TEdit;
  MtxtUrunAd: TEdit;
  MtxtUrunTip: TEdit;
  ProgressBar1: TProgressBar;
  PageControl3: TPageControl;
  TabSheet2: TTabSheet;
  Label17: TLabel;
  Label9: TLabel;
  Label10: TLabel;
  Label18: TLabel;
  MtxtPTekrar: TCurrencyEdit;
  MtxtGTekrar: TCurrencyEdit;
  Label1: TLabel;
  Label6: TLabel;
  MtxtReceteKod: TEdit;
  MtxtReceteAdi: TEdit;
```

```

MtxtBTarih: TEdit;
MtxtBSaat: TEdit;
Timer1: TTimer;
VrLed5: TVrLed;
pnlUretimNo: TPanel;
Button1: TButton;
procedure FormShow(Sender: TObject);
procedure Timer1Timer(Sender: TObject);
procedure Button1Click(Sender: TObject);

private
{ Private declarations }
procedure MenuOlustur();
procedure OnMenuItemClick(Sender: TObject);

procedure KartBilgi(RoleNo:String);
procedure Animasyon();
procedure UretimBitir();
procedure StokDus();
procedure SHKaydet(SKod:string;Miktar:Double);

public
{ Public declarations }
procedure Baslat();
end;

var
FrmMainForm: TFrmMainForm;
agrega,islem,tekrar:integer;
sira :array[0..8] of Integer;

implementation

uses UntPersonelListesi, UntUretimPlanlama, UntUretimListe,
UntUrunListesi, UntReceteListesi, UntStokKartListe,
UntOnKesmeDegerleri, UntDolumParametreleri, UntAgregaDolumSirasi,
UntCHListe, UntStokHareketListe, UntCHHareket, UntGlobal, UntDm,
ADODB, UntHaklar, DB, StrUtils, UntUretimRaporlar;

{$R *.dfm}

procedure TFrmMainForm.FormShow(Sender: TObject);
begin
Caption:=ProCaption;
MenuOlustur;
end;
// ***
procedure TFrmMainForm.MenuOlustur;
var
i:integer;
GrpNo,say:integer;
TmpItem:TdxNavBarItem;
begin
with dm.Qrylist do
begin
close;

```

```

SQL.Clear;
SQL.Add('Select M.ModulNo,M.TanimNo,M.Tanim,H.TagNo From Moduller as M , Haklar as H');
SQL.Add('Where M.TagNo=H.TagNo and H.PerNo='+ ActPerNo);
SQL.Add('Order by M.ModulNo,M.TanimNo Asc');
Open;
say:=0;
GrpNo:=0;
for i:=1 to RecordCount do
begin
if (GrpNo<> FieldByName('ModulNo').AsInteger) and (FieldByName('TanimNo').AsInteger = 0)then
begin
say:=Say+1;
GrpNo:=FieldByName('ModulNo').AsInteger;
NavBar.Groups.Add;
NavBar.Groups[say-1].Caption:=FieldByName('Tanim').AsString;
NavBar.Groups[say-1].UseSmallImages:=False;
NavBar.Groups[say-1].LargeImageIndex:=GrpNo-1;
end;
if (GrpNo=FieldByName('ModulNo').AsInteger)and (FieldByName('TanimNo').AsInteger <> 0) then
begin
TmpItem:=TdxNavBarItem.Create(Self);
with TmpItem do
begin
TmpItem.Caption:=FieldByName('Tanim').AsString;
TmpItem.Tag :=FieldByName('TagNo').AsInteger;
TmpItem.SmallImageIndex:=GrpNo ;
OnClick:=OnMenuItemClick;
end;
NavBar.Groups.Items[say-1].CreateLink(TmpItem);
end;
next;
end;
end;
end;
// ***
procedure TFrmMainForm.OnMenuItemClick(Sender: TObject);
var
FrmNo:integer;
begin
FrmNo:=(Sender as TdxNavBarItem).Tag;
case FrmNo of
11: begin // *** Üretim
FrmUretimPlanlama:=TFrmUretimPlanlama.Create(self);
FrmUretimPlanlama.Tag:=FrmNo;
FrmUretimPlanlama.ShowModal;
FrmUretimPlanlama.Release;
end;
12: begin
FrmUretimListe:=TFrmUretimListe.Create(self);
FrmUretimListe.Tag:=FrmNo;
FrmUretimListe.ShowModal;
FrmUretimListe.Release;
end;
13: begin
FrmUretimRaporlar:=TFrmUretimRaporlar.Create(self);
FrmUretimRaporlar.Tag:=FrmNo;

```

```

        FrmUretimRaporlar.ShowModal;
        FrmUretimRaporlar.Release;
    end;
21: begin // *** Veri Düzenleme
        FrmUrunListesi:=TFrmUrunListesi.Create(self);
        FrmUrunListesi.Tag:=FrmNo;
        FrmUrunListesi.ShowModal;
        FrmUrunListesi.Release;
    end;
22: begin
        FrmReceteListesi:=TFrmReceteListesi.Create(Self);
        FrmReceteListesi.Tag:=FrmNo;
        FrmReceteListesi.ShowModal;
        FrmReceteListesi.Release;
    end;
23: begin
        FrmStokKartListe:=TFrmStokKartListe.Create(self);
        FrmStokKartListe.Tag:=FrmNo;
        FrmStokKartListe.ShowModal;
        FrmStokKartListe.Release;
    end;
24: begin
        FrmStokHareketListe:=TFrmStokHareketListe.Create(self);
        FrmStokHareketListe.Tag:=FrmNo;
        FrmStokHareketListe.ShowModal;
        FrmStokHareketListe.Release;
    end;
31: begin // *** ayarlar
        FrmOnKesmeDegerleri:=TFrmOnKesmeDegerleri.Create(self);
        FrmOnKesmeDegerleri.Tag:=FrmNo;
        FrmOnKesmeDegerleri.ShowModal;
        FrmOnKesmeDegerleri.Release;
    end;
32: begin
        FrmDolumParametreleri:=TFrmDolumParametreleri.Create(self);
        FrmDolumParametreleri.Tag:=FrmNo;
        FrmDolumParametreleri.ShowModal;
        FrmDolumParametreleri.Release;
    end;
33: begin
        FrmAgregaDolumSirasi:=TFrmAgregaDolumSirasi.Create(self);
        FrmAgregaDolumSirasi.Tag:=FrmNo;
        FrmAgregaDolumSirasi.ShowModal;
        FrmAgregaDolumSirasi.Release;
    end;
34: begin
    end;
41: begin // *** Personel
        FrmPersonelListesi:=TFrmPersonelListesi.Create(self);
        FrmPersonelListesi.Tag:=FrmNo;
        FrmPersonelListesi.ShowModal;
        FrmPersonelListesi.Release;
    end;
42: begin
        FrmHaklar:=TFrmHaklar.Create(self);
        FrmHaklar.Tag:=FrmNo;

```

```

        FrmHaklar.ShowModal;
        FrmHaklar.Release;
    end;
51: begin // *** müşteriler
        FrmCHListe:=TFrmCHListe.Create(self);
        FrmCHListe.Tag:=FrmNo;
        FrmCHListe.ShowModal;
        FrmCHListe.Release;
    end;
52: begin
        FrmCHHareket:=TFrmCHHareket.Create(self);
        FrmCHHareket.Tag:=FrmNo;
        FrmCHHareket.ShowModal;
        FrmCHHareket.Release;
    end;
end;
end;
// ***
procedure TFrmMainForm.Baslat;
var
    i:integer;
    tanim:string;
begin
    with dm.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * From DolumSira order by Sira');
        open;
        for i :=1 to 4 do
        begin
            tanim:=FieldByName('Tanim').AsString;
            if LeftStr(tanim,2)='Ku' then sira[i]:=2;
            if LeftStr(tanim,2)='01' then sira[i]:=3;
            if LeftStr(tanim,2)='02' then sira[i]:=4;
            if LeftStr(tanim,2)='03' then sira[i]:=5;
            //ShowMessage(tanim + ' ---> ' + intoStr(sira[i]));
        end;
        next;
    end;
end;
tekrar:=MtxtPTekrar.AsInteger;
VrLed1.Active:=True;
VaComm1.WriteText('1'+#13);
    ProgressBar1.Max:=tekrar;
    Timer1.Enabled:=True;
end;
// ***
procedure TFrmMainForm.KartBilgi(RoleNo: String);
begin
    VaComm1.WriteText(RoleNo+#13);
end;
// ***
procedure TFrmMainForm.Timer1Timer(Sender: TObject);
begin
    if tekrar>0 then
    begin

```

```

islem:=islem+1;
Animasyon;
end;
if tekrar=0 then
begin
Timer1.Enabled:=False;
KartBilgi('0');
UretimBitir;
StokDus;
end;
end;
// ***
procedure TFrmMainForm.Animasyon;
begin
case islem of
2,4,6,8 :begin
agrega:=agrega+1; ;
case sira[agrega] of

2: begin // 2
VrLed1.Active:=True;
VrLed2.Active:=True;
VrLed3.Active:=False;
VrLed4.Active:=False;
VrLed5.Active:=False;
KartBilgi('3');
end;
3: begin // 4
VrLed1.Active:=True;
VrLed2.Active:=False;
VrLed3.Active:=True;
VrLed4.Active:=False;
VrLed5.Active:=False;
KartBilgi('5');
end;
4: begin // 8
VrLed1.Active:=True;
VrLed2.Active:=False;
VrLed3.Active:=False;
VrLed4.Active:=True;
VrLed5.Active:=False;
KartBilgi('9');
end;
5: begin // 16
VrLed1.Active:=True;
VrLed2.Active:=False;
VrLed3.Active:=False;
VrLed4.Active:=False;
VrLed5.Active:=True;
KartBilgi('17');
end;
end;
end;
10: begin
VrLed1.Active:=True;
VrLed2.Active:=False;

```

```

        VrLed3.Active:=False;
        VrLed4.Active:=False;
        VrLed5.Active:=False;
        KartBilgi('1');
    end;
11: begin
    VrLed1.Active:=True;
    VrLed6.Active:=True;
    KartBilgi('33');
    end;
14: begin
    VrLed6.Active:=False;
    VrLed1.Active:=False;
    KartBilgi('0');
    end;
15: begin
    VrLed7.Active:=True;
    KartBilgi('64');
    end;
17: begin
    VrLed7.Active:=False;
    KartBilgi('0');
    end;
22: begin
    VrLed8.Active:=True;
    KartBilgi('128');
    end;
26: begin
    VrLed8.Active:=False;
    KartBilgi('0');
    tekrar:=tekrar-1;
    MtxtGTekrar.Value:=MtxtGTekrar.Value+1;
    ProgressBar1.Position:=MtxtGTekrar.AsInteger;
    islem:=0;
    agrega:=0;
    end;
end;
end;
// ***
procedure TFrmMainForm.UretimBitir;
begin
    with dm.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Update uretim set ETarih=:ActETarih, ESaat=:ActESaat, Durum=:ActDurum,');
        SQL.Add(' GTekrar=PTekrar,GMiktar=(PTekrar*RStdMiktar)');
        SQL.Add('Where UretimNo="'+ pnlUretimNo.Caption +' "');
        Parameters.ParamByName('ActETarih').Value :=FormatDateTime('dd.mm.yyyy',Date);
        Parameters.ParamByName('ActESaat').Value :=FormatDateTime('hh:mm',Now);
        Parameters.ParamByName('ActDurum').Value :=2;
        Prepared;
        ExecSQL;
    end;
end;
// ****

```



```

procedure TFrmMainForm.StokDus;
var
  tmpstr:string;
begin
  tmpSTR:='SELECT '+
    '( R.Agrega01 * U.PTekrar) as Agregal, '+
    '( R.Agrega02 * U.PTekrar) as Agreg2, '+
    '( R.Agrega03 * U.PTekrar) as Agreg3, '+
    '( R.Kum * U.PTekrar) as TKum, '+
    '( R.Cimento * U.PTekrar) as TCimento '+
    'FROM Uretim as U ,Recete as R Where U.ReceteKod = R.ReceteKod '+
    'and U.UretimNo="'+ pnlUretimNo.Caption +"' ";
  with dm.Qry1 do
  begin
    Close;
    SQL.Clear ;
    SQL.Add(tmpStr);
    open;
    SHKaydet('00001',FieldByName('Agrega1').AsFloat);
    SHKaydet('00002',FieldByName('Agrega2').AsFloat);
    SHKaydet('00003',FieldByName('Agrega3').AsFloat);
    SHKaydet('00004',FieldByName("TKum").AsFloat);
    SHKaydet('00005',FieldByName("TCimento").AsFloat);
  close;
  end;
  end;
  // ***
procedure TFrmMainForm.SHKaydet(SKod:string;Miktar:Double);
begin
  with DM.QryArama do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Select * From StokExtre2005');
    Prepared ;
    Open;
    Append;
    FieldByName('Kod').AsString :=SKod;
    FieldByName('OpDate').AsString :=FormatDateTime('dd.mm.yyyy',Date);
    FieldByName('OpNumber').AsString:=pnlUretimNo.Caption;
    FieldByName('OpDefinition').AsString:=pnlUretimNo.Caption+' için kullanıldı';
    FieldByName('OpType').AsString :='SC';
    FieldByName('InValue').AsFloat :=0;
    FieldByName('OutValue').AsFloat :=Miktar;
  post;
  Close;
  SQL.Clear;
  SQL.Add('Update Stok Set Miktar=Miktar-:yeni Where Kod=:Bul ');
  Parameters.ParamByName('Yeni').Value:=Miktar;
  Parameters.ParamByName('bul').Value :=SKod;
  Prepared ;
  ExecSQL;
  end;
  end;
  // ***
procedure TFrmMainForm.Button1Click(Sender: TObject);

```

```

begin
    stokdus;
end;
end.

*****

unit UntOnay;

interface

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

type
    TFrmOnay = class(TForm)
        Image1: TImage;
        Label1: TLabel;
        BtnTamam: TBitBtn;
        BtnIptal: TBitBtn;
        lblCaption: TLabel;
        procedure BtnIptalClick(Sender: TObject);
        procedure FormCreate(Sender: TObject);
        procedure BtnTamamClick(Sender: TObject);
        procedure FormKeyDown(Sender: TObject; var Key: Word;
            Shift: TShiftState);
        procedure Image1MouseDown(Sender: TObject; Button: TMouseButton;
            Shift: TShiftState; X, Y: Integer);
    private
        { Private declarations }
    public
        { Public declarations }
        ActOnay: Boolean;
        ActCaption: string;
    end;

var
    FrmOnay: TFrmOnay;
implementation
{$R *.dfm}

procedure TFrmOnay.BtnIptalClick(Sender: TObject);
begin
    ActOnay:=False;
    Self.Close;
end;
// ***

procedure TFrmOnay.FormCreate(Sender: TObject);
begin
    lblCaption.Caption:= ActCaption;
end;
// ***

procedure TFrmOnay.BtnTamamClick(Sender: TObject);
begin
    ActOnay:=True;

```

```

    Self.Close;
end;
// ***
procedure TFrmOnay.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if Key=vk_Escape then BtnIptal.Click;
end;
// ***
procedure TFrmOnay.Image1MouseDown(Sender: TObject; Button: TMouseButton;
    Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntOnKesmeDegerleri;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, Buttons, StdCtrls, Mask, ToolEdit, CurrEdit, ComCtrls, ExtCtrls,
    AdvToolBar, AdvToolBarStylers;

type
    TFrmOnKesmeDegerleri = class(TForm)
        PageControl3: TPageControl;
        TabSheet2: TTabSheet;
        Label18: TLabel;
        Label20: TLabel;
        txtCimento: TCurrencyEdit;
        txtKatki: TCurrencyEdit;
        Label8: TLabel;
        txtA01: TCurrencyEdit;
        Label11: TLabel;
        txtA02: TCurrencyEdit;
        Label13: TLabel;
        txtA03: TCurrencyEdit;
        Label14: TLabel;
        txtKum: TCurrencyEdit;
        Panel4: TPanel;
        Image9: TImage;
        Panel3: TPanel;
        Image2: TImage;
        Label16: TLabel;
        Label17: TLabel;
        AdvDockPanel1: TAdvDockPanel;
        AdvToolBar1: TAdvToolBar;
        BtnKapat: TAdvToolBarButton;
        BtnEkle: TAdvToolBarButton;
        AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
        StatusBar1: TStatusBar;
    end;

```

```

    procedure btnKapatClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure FormCreate(Sender: TObject);
    procedure BtnEkleClick(Sender: TObject);
    procedure Panel4MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FrmOnKesmeDegerleri: TFrmOnKesmeDegerleri;

implementation

uses UntDm, DB, ADODB, UntGlobal;

{$R *.dfm}

procedure TFrmOnKesmeDegerleri.btnKapatClick(Sender: TObject);
begin
  Self.Close;
end;
// ***

procedure TFrmOnKesmeDegerleri.FormShow(Sender: TObject);
var
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
  SetWindowRgn(handle, region, true);
  KullaniciHaklari(Self.Tag,AdvToolBar1);
  with dm.Qry1 do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Select * from Kesme ');
    Open;
    txtA01.Value :=FieldByName('A01').AsFloat;
    txtA02.Value :=FieldbyName('A02').AsFloat;
    txtA03.Value :=FieldByName('A03').AsFloat;
    txtKum.Value :=FieldByName('Kum').AsFloat;
    txtCimento.Value:=FieldByName('Cimento').AsFloat;
    txtkatki.Value :=FieldByName('katki').AsFloat;
  end;
end;
// ***

procedure TFrmOnKesmeDegerleri.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if (key = vk_F4) and (BtnEkle.Visible=True) then  BtnEkle.Click;

```

```

    if key=vk_F10 then btnKapat.Click;
end;
// ***
procedure TFrmOnKesmeDegerleri.FormCreate(Sender: TObject);
begin
    Caption :=ProCaption ;
end;
// ***
procedure TFrmOnKesmeDegerleri.BtnEkleClick(Sender: TObject);
begin
    if Onay('Ön Kesme Değerleri Kayıt İşlemi;'+#13+'Devam Etmek İstiyor musunuz ?')=False then exit;
    with dm.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * From Kesme ');
        Open;
        if RecordCount=0 then
        begin
            Close;
            SQL.Clear;
            SQL.Add('Insert Into Kesme (A01,A02,A03,Kum,Cimento,Katki) ');
            SQL.Add('Values (:A01,,:A02,,:A03,,:Kum,,:Cimento,,:Katki) ');
            Parameters.ParamByName('A01').Value :=txtA01.Value ;
            Parameters.ParamByName('A02').Value :=txtA02.Value ;
            Parameters.ParamByName('A03').Value :=txtA03.Value ;
            Parameters.ParamByName('Kum').Value :=txtKum.Value ;
            Parameters.ParamByName('Cimento').Value:=txtCimento.Value;
            Parameters.ParamByName('Katki').Value :=txtkatki.Value ;
            Prepared;
            ExecSQL;
        end
        else
        begin
            Close;
            SQL.Clear;
            SQL.Add('Update Kesme Set A01 =:A01, A02 =:A02, A03 =:A03,');
            SQL.Add('Kum =:Kum, Cimento =:Cimento, Katki =:Katki');
            Parameters.ParamByName('A01').Value :=txtA01.Value ;
            Parameters.ParamByName('A02').Value :=txtA02.Value ;
            Parameters.ParamByName('A03').Value :=txtA03.Value ;
            Parameters.ParamByName('Kum').Value :=txtKum.Value ;
            Parameters.ParamByName('Cimento').Value:=txtCimento.Value;
            Parameters.ParamByName('Katki').Value :=txtkatki.Value ;
            Prepared;
            ExecSQL;
        end;
    end;
    self.Close;
end;
// ***
procedure TFrmOnKesmeDegerleri.Panel4MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);

```

end;  
end.

\*\*\*\*\*

unit UntPersonelListesi;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  
Dialogs, ExtCtrls, StdCtrls, Buttons,  
XPMAN, DB, ADODB, AdvToolBar, AdvToolBarStylers, ImgList,  
Mask, ToolEdit, CurrEdit, cxStyles, cxCustomData, cxGraphics, cxFilter,  
cxData, cxDataStorage, cxEdit, cxDBData, cxGridLevel, cxClasses,  
cxControls, cxGridCustomView, cxGridCustomTableView, cxGridTableView,  
cxGridDBTableView, cxGrid, ComCtrls, dxPSGlbl, dxPSUtl, dxPSEngn,  
dxPrnPg, dxBkgnd, dxWrap, dxPrnDev, dxPSCmpsProvider, dxPSFillPatterns,  
dxPSEdgePatterns, dxPSCore, dxPScxCommon, dxPScxGridLnk;

type

TFrmPersonelListesi = class(TForm)  
  DSPerQuery: TDataSource;  
  PerQuery: TADOQuery;  
  AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;  
  Panel4: TPanel;  
  Image9: TImage;  
  Panel3: TPanel;  
  Image2: TImage;  
  AdvDockPanel1: TAdvDockPanel;  
  AdvToolBar1: TAdvToolBar;  
  BtnKapat: TAdvToolBarButton;  
  BtnSil: TAdvToolBarButton;  
  BtnDegistir: TAdvToolBarButton;  
  BtnEkle: TAdvToolBarButton;  
  BtnYazdir: TAdvToolBarButton;  
  Label16: TLabel;  
  Label17: TLabel;  
  pnlIslem: TPanel;  
  PnlCaption: TPanel;  
  Shape1: TShape;  
  CapImage: TImage;  
  lblCaption: TLabel;  
  txtPerAdi: TEdit;  
  Label2: TLabel;  
  Label3: TLabel;  
  txtPerSoyadi: TEdit;  
  Label4: TLabel;  
  txtPerNo: TCurrencyEdit;  
  BtnUygula: TSpeedButton;  
  BtnIptal: TSpeedButton;  
  cxGrid1DBTableView1: TcxGridDBTableView;  
  cxGrid1Level1: TcxGridLevel;  
  cxGrid1: TcxGrid;  
  StatusBar1: TStatusBar;  
  cxGrid1DBTableView1PerAdi: TcxGridDBColumn;

```

cxGrid1DBTableView1PerSoyadi: TcxGridDBCColumn;
cxGrid1DBTableView1PerNo: TcxGridDBCColumn;
cxGrid1DBTableView1KullaniciAdi: TcxGridDBCColumn;
cxGrid1DBTableView1Sifre: TcxGridDBCColumn;
cxGrid1DBTableView1Yonetici: TcxGridDBCColumn;
Label1: TLabel;
txtKullaniciAdi: TEdit;
Label6: TLabel;
txtSifre: TEdit;
CheckBox1: TCheckBox;
PrintPreview: TdxComponentPrinter;
PrintPreviewLink1: TdxGridReportLink;
procedure FormShow(Sender: TObject);
procedure BtnEkleClick(Sender: TObject);
procedure BtnDegistirClick(Sender: TObject);
procedure BtnSilClick(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure BtnYazdirClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtnIptalClick(Sender: TObject);
procedure BtnUygulaClick(Sender: TObject);
procedure Panel4MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
private
  { Private declarations }
public
  { Public declarations }
  procedure Islem(Durum:Boolean;Baslik:String);
  procedure SetGridView();
  procedure PersonelKayit();
  procedure PersonelDegistir();
  procedure PersonelSil();
end;

var
  FrmPersonelListesi: TFrmPersonelListesi;
  ActPerID:string;

implementation

uses  UntDm,UntGlobal;

{$R *.dfm}

function KullaniciAdi(bak:String):Boolean;
begin
  with dm.Qry1 do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Select * from Personel Where KullaniciAdi=:bak ');
    Parameters.ParamByName('bak').Value :=bak;
    Open;
    if RecordCount=0 then
      KullaniciAdi:=True

```

```

else
begin
    Hata('Kullanıcı Adı Daha Önceden Kullanılmıştır.');
```

    KullaniciAdi:=False;

```

end;
end;
end;
// ***

procedure TFrmPersonelListesi.SetGridView();
begin
with PerQuery do
begin
    Close;
    SQL.Clear;
    SQL.Add('Select * From Personel');
    Prepared ;
    Open;
end;
end;
// ***

procedure TFrmPersonelListesi.FormShow(Sender: TObject);
var
region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
    SetWindowRgn(handle, region, true);
    KullaniciHaklari(Self.Tag,AdvToolBar1);
    SetGridView;
end;
// ***

procedure TFrmPersonelListesi.BtnEkleClick(Sender: TObject);
begin
    Islem(True,'Kaydet');
    txtPerAdi.Text :='';
    txtPerSoyadi.Text :='';
    txtPerNo.Value :=0;
    txtKullaniciAdi.Text:='';
    txtSifre.Text :='';
    CheckBox1.Checked :=False;
end;
// ***

procedure TFrmPersonelListesi.BtnDegistirClick(Sender: TObject);
begin
    with PerQuery do
    begin
        txtPerAdi.Text :=FieldByName('PerAdi').AsString ;
        txtPerSoyadi.Text :=FieldByName('PerSoyadi').AsString ;
        txtPerNo.Text :=FieldByName('PerNo').AsString ;
        txtKullaniciAdi.Text:=FieldByName('KullaniciAdi').AsString;
        txtSifre.Text :=FieldByName('Sifre').AsString ;
        CheckBox1.Checked :=FieldByName('Yonetici').AsBoolean ;
        ActPerID:=FieldByName('PerID').AsString;
    end;
    Islem(True,'Güncelle');
```



```

end;
// ***
procedure TFrmPersonelListesi.BtnSilClick(Sender: TObject);
begin
  with PerQuery do
  begin
    txtPerAdi.Text :=FieldByName('PerAdi').AsString ;
    txtPerSoyadi.Text :=FieldByName('PerSoyadi').AsString ;
    txtPerNo.Text :=FieldByName('PerNo').AsString ;
    txtKullaniciAdi.Text:=FieldByName('KullaniciAdi').AsString;
    txtSifre.Text :=FieldByName('Sifre').AsString ;
    CheckBox1.Checked :=FieldByName('Yonetici').AsBoolean ;
    ActPerID:=FieldByName('PerID').AsString;
  end;
  Islem(True,'Sil');
end;
// ***
procedure TFrmPersonelListesi.BtnKapatClick(Sender: TObject);
begin
  Self.Close;
end;
// ***
procedure TFrmPersonelListesi.BtnYazdirClick(Sender: TObject);
begin
  PrintPreviewLink1.ReportTitleText:=Label16.Caption;
  PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Clear;
  PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi: '+
  FormatDateTime('dd.mm.yyyy',date));
  PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Clear;
  PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
  PrintPreview.PrintTitle:=ProCaption;
  PrintPreview.Preview(True,PrintPreviewLink1);
end;
// ***
procedure TFrmPersonelListesi.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
  if (key = vk_F5) and (BtnDegistir.Visible=True) then BtnDegistir.Click;
  if (key = vk_F6) and (BtnSil.Visible=True) then BtnSil.Click;
  if (key = vk_F8) and (BtnYazdir.Visible=True) then BtnYazdir.Click;
  if key=vk_F10 then BtnKapat.Click;
  if (key=vk_escape) and (pnlIslem.Visible=True) then islem(False,'---');
end;

// ***
procedure TFrmPersonelListesi.Islem(Durum: Boolean; Baslik: String);
begin
  panel4.Enabled :=not durum;
  pnlIslem.Visible :=Durum;
  BtnUygula.Caption :=Baslik;
  cxGrid1.Enabled :=not durum;
  if BtnUygula.Caption='Kaydet' then txtPerAdi.SetFocus;
  if BtnUygula.Caption='Kaydet' then lblCaption.Caption:='Yeni Personel Ekle ';
  if BtnUygula.Caption='Güncelle' then lblCaption.Caption:='Personel Bilgisini Değiştir ';

```

```

    if BtnUygula.Caption='Sil' then lblCaption.Caption:='Personel Bilgisini Sil ' ;
end;
// ***
procedure TFrmPersonelListesi.BtnIptalClick(Sender: TObject);
begin
    Islem(False, '--');
end;
// ***
procedure TFrmPersonelListesi.BtnUygulaClick(Sender: TObject);
begin
    if BtnUygula.Caption='Kaydet' then PersonelKayit;
    if BtnUygula.Caption='Güncelle' then PersonelDegistir;
    if BtnUygula.Caption='Sil' then PersonelSil;
end;
// ***
procedure TFrmPersonelListesi.PersonelKayit;
begin
    if txtPerAdi.Text="" then
    begin
        Hata('Personel Adını Giriniz. ');
        exit;
    end;
    if txtPerSoyadi.Text="" then
    begin
        Hata('Personel Soyadını Giriniz. ');
        exit;
    end;
    if txtPerNo.AsInteger=0 then
    begin
        Hata('Personel No`sunu Giriniz. ');
        exit;
    end;
    if Length(trim(txtKullaniciAdi.Text))=0 then
    begin
        Hata('Kullanıcı Adı Giriniz. ');
        txtKullaniciAdi.SetFocus;
        exit;
    end;
    if Length(trim(txtSifre.Text))=0 then
    begin
        Hata('Şifrenizi Giriniz. ');
        txtSifre.SetFocus;
        exit;
    end;
    if KullaniciAdi(Trim(txtKullaniciAdi.Text))=False then Exit;
    with DM.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * From Personel Where PerNo='+txtPerNo.Text);
        Prepared ;
        Open;
        if recordcount>0 then
        begin
            Hata('Personel No`sü Kullanılmıştır. ');
            txtPerNo.SetFocus;

```

```

        exit;
    end;
    Close;
    SQL.Clear;
    SQL.Add('Insert Into Personel (PerAdi,PerSoyAdi,PerNo,KullaniciAdi,Sifre,Yonetici) ');
    SQL.Add('Values(:PerAdi,:PerSoyAdi,:PerNo,:KullaniciAdi,:Sifre,:Yonetici) ');
    Parameters.ParamByName('PerAdi').Value :=txtPerAdi.Text;
    Parameters.ParamByName('PerSoyAdi').Value:=txtPerSoyadi.Text;
    Parameters.ParamByName('PerNo').Value :=txtPerNo.AsInteger;
    Parameters.ParamByName('KullaniciAdi').Value:=txtKullaniciAdi.Text;
    Parameters.ParamByName('Sifre').Value :=txtSifre.Text;
    Parameters.ParamByName('Yonetici').Value := CheckBox1.Checked;
    Prepared ;
    ExecSQL;
end;
    CompTemizle(Self,1);
    txtPerNo.Value:=0;
    CheckBox1.Checked :=False;
    SetGridView;
end;
// ***
procedure TFrmPersonelListesi.PersonelDegistir;
begin
    if txtPerAdi.Text="" then
    begin
        Hata('Personel Adını Giriniz. ');
        Exit;
    end;
    if txtPerSoyadi.Text="" then
    begin
        Hata('Personel Soyadını Giriniz. ');
        exit;
    end;
    if txtPerNo.AsInteger=0 then
    begin
        Hata('Personel No`sunu Giriniz. ');
        exit;
    end;
    if Length(trim(txtKullaniciAdi.Text))=0 then
    begin
        Hata('Kullanıcı Adı Giriniz. ');
        txtKullaniciAdi.SetFocus;
        exit;
    end;
    if Length(trim(txtSifre.Text))=0 then
    begin
        Hata('Şifrenizi Giriniz. ');
        txtSifre.SetFocus;
        exit;
    end;
    with DM.Qrylist do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * From Personel Where PerNo=:ActPerNo and PerID<>' + ActPerID );
        Parameters.ParamByName('ActPerNo').Value :=txtPerNo.Value;

```

```

Prepared ;
Open;
if recordcount>0 then
begin
  Hata('Personel No`su Kullanılmıştır.');
```

exit;

```
end;
Close;
SQL.Clear;
SQL.Add('Select * From Personel Where KullaniciAdi=:ActKullanici and PerID<>'+ ActPerID );
Parameters.ParamByName('ActKullanici').Value:=Trim(txtKullaniciAdi.Text);
Prepared ;
Open;
if recordcount>0 then
begin
  Hata('Kullanıcı Adı Kullanılmıştır.');
```

exit;

```
end;
Close;
SQL.Clear;
SQL.Add('Select * From Personel Where PerID='+ActPerID);
Prepared ;
Open;
Edit;
FieldByName('PerAdi').AsString :=txtPerAdi.Text ;
FieldByName('PerSoyadi').AsString :=txtPerSoyadi.Text ;
FieldByName('PerNo').AsString :=txtPerNo.Text ;
FieldByName('KullaniciAdi').AsString:=txtKullaniciAdi.Text;
FieldByName('Sifre').AsString :=txtSifre.Text;
FieldByName('Yonetici').AsBoolean :=CheckBox1.Checked;
Post;
end;
Islem(False, '---');
SetGridView;
end;
// **

procedure TFrmPersonelListesi.PersonelSil;
begin
  if Onay('Personel Silinecektir.'+#13+'Devam Etmek İstiyor musunuz ?')=False then exit;
  with DM.Qry1 do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Delete * From Haklar Where PerNo='+txtPerNo.Text);
    Prepared ;
    ExecSQL;
    Close;
    SQL.Clear;
    SQL.Add('Delete * From Personel Where PerID='+ActPerID);
    Prepared ;
    ExecSQL;
  end;
  Islem(False, '---');
  SetGridView;
end;
// ***

```

```

procedure TFrmPersonelListesi.Panel4MouseDown(Sender: TObject;
Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ReleaseCapture;
  self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

```

```

*****

```

```

unit UntReceteDegistir;

```

```

interface

```

```

uses

```

```

  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ComCtrls, CurrEdit, Mask, ToolEdit, Buttons, ExtCtrls;

```

```

type

```

```

  TFrmReceteDegistir = class(TForm)
    Panel5: TPanel;
    Image9: TImage;
    Image10: TImage;
    BtnDegistir: TSpeedButton;
    Label16: TLabel;
    Label17: TLabel;
    BtnKapat: TSpeedButton;
    GroupBox1: TGroupBox;
    Label9: TLabel;
    HAdi: TSpeedButton;
    Label1: TLabel;
    txtHPerAdi: TEdit;
    txtHPerSoyadi: TEdit;
    GroupBox2: TGroupBox;
    Label15: TLabel;
    OAdi: TSpeedButton;
    Label23: TLabel;
    txtOPerSoyadi: TEdit;
    txtOPerAdi: TEdit;
    Label2: TLabel;
    Edit1: TEdit;
    PageControl2: TPageControl;
    TabSheet3: TTabSheet;
    Label3: TLabel;
    Label10: TLabel;
    Label12: TLabel;
    Label7: TLabel;
    txtSpektNo: TEdit;
    txtUrunKod: TEdit;
    txtUrunAd: TEdit;
    txtUrunTip: TEdit;
    PageControl4: TPageControl;
    TabSheet4: TTabSheet;
    Label5: TLabel;
    Label6: TLabel;
    Label4: TLabel;
  end;

```

```

txtReceteKod: TEdit;
txtReceteAdi: TEdit;
DTRecTarih: TDateEdit;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label8: TLabel;
Label11: TLabel;
Label13: TLabel;
Label14: TLabel;
txtAgrega01: TCurrencyEdit;
txtAgrega02: TCurrencyEdit;
txtAgrega03: TCurrencyEdit;
txtKum: TCurrencyEdit;
PageControl3: TPageControl;
TabSheet2: TTabSheet;
Label18: TLabel;
Label20: TLabel;
Label21: TLabel;
Label22: TLabel;
Label19: TLabel;
txtCimento: TCurrencyEdit;
txtSu: TCurrencyEdit;
txtKatki01: TCurrencyEdit;
txtKatki02: TCurrencyEdit;
txtSaat: TCurrencyEdit;
txtDakika: TCurrencyEdit;
txtRevNo: TEdit;
Label24: TLabel;
StatusBar1: TStatusBar;
procedure FormCreate(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure FormShow(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure BtnDegistirClick(Sender: TObject);
procedure Panel5MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
private
  { Private declarations }
public
  { Public declarations }
  actRecerteRecID:String;
end;

var
  FrmReceteDegistir: TFrmReceteDegistir;

implementation

uses UntDm, UntGlobal,StrUtils;

{$R *.dfm}

procedure TFrmReceteDegistir.FormCreate(Sender: TObject);
begin
  Caption:=ProCaption;

```

```

end;
// ***
procedure TFrmReceteDegistir.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if key = vk_f5 then BtnDegistir.Click ;
  if key = vk_Escape then BtnKapat.Click ;
end;
// ***
procedure TFrmReceteDegistir.FormShow(Sender: TObject);
var
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
  SetWindowRgn(handle, region, true);
  with DM.Qry1 do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Select * From Recete Where RecID='+actRecerteRecID);
    Prepared ;
    Open;
    txtSpektNo.Text := FieldByName('SpektNo').AsString ;
    txtReceteKod.Text := FieldByName('ReceteKod').AsString ;
    txtReceteAdi.Text := FieldByName('ReceteAdi').AsString ;
    DTRecTarih.Text := FieldByName('RecTarih').AsString ;
    txtAgrega01.Value := FieldByName('Agrega01').AsFloat ;
    txtAgrega02.Value := FieldByName('Agrega02').AsFloat ;
    txtAgrega03.Value := FieldByName('Agrega03').AsFloat ;
    txtKum.Value := FieldByName('Kum').AsFloat ;
    txtCimento.Value := FieldByName('Cimento').AsFloat ;
    txtSu.Value := FieldByName('Su').AsFloat ;
    txtKatki01.Value := FieldByName('Katki01').AsFloat ;
    txtKatki02.Value := FieldByName('Katki02').AsFloat ;
    txtSaat.AsInteger := FieldByName('Saat').AsInteger ;
    txtDakika.AsInteger := FieldByName('Dakika').AsInteger ;
  end;
end;
// ***
procedure TFrmReceteDegistir.BtnKapatClick(Sender: TObject);
begin
  self.Close;
end;
// ***
procedure TFrmReceteDegistir.BtnDegistirClick(Sender: TObject);
begin
  if txtDakika.Value>59 then
  begin
    ShowMessage('Dakika Değeri 60`dan Büyük Olamaz.'+ #13 + 'Lütfen Tanımlayınız. ');
    exit;
  end;
  with DM.Qry1 do
  begin
    Close;

```

```

SQL.Clear;
SQL.Add('Select * From Recete Where RecID='+actRecerteRecID);
Prepared ;
Open;
Edit;
FieldByName('Agrega01').AsFloat :=txtAgrega01.Value;
FieldByName('Agrega02').AsFloat :=txtAgrega02.Value;
FieldByName('Agrega03').AsFloat :=txtAgrega03.Value;
FieldByName('Kum').AsFloat :=txtKum.Value;
FieldByName('Cimento').AsFloat :=txtCimento.Value;
FieldByName('Su').AsFloat :=txtSu.Value;
FieldByName('Katki01').AsFloat :=txtKatki01.Value;
FieldByName('Katki02').AsFloat :=txtKatki02.Value;
FieldByName('ToplamMiktar').AsFloat:=txtAgrega01.Value+txtAgrega02.Value+
txtAgrega03.Value+txtKum.Value+txtCimento.Value
+ txtSu.Value+txtKatki01.Value+txtKatki02.Value;
FieldByName('Saat').AsString :=RightStr('00'+txtSaat.Text,2);
FieldByName('Dakika').AsString :=RightStr('00'+txtDakika.Text,2);
Post;
end;
Self.Close;
end;
// ***
procedure TFrmReceteDegistir.Panel5MouseDown(Sender: TObject;
Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
ReleaseCapture;
self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntReceteDetay;

interface

uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, ExtCtrls, ComCtrls, CurrEdit, Mask, ToolEdit, StdCtrls, Buttons,
Grids, BaseGrid, AdvGrid;

type
TFrmReceteDetay = class(TForm)
DetayGrid: TAdvStringGrid;
Panel5: TPanel;
Image9: TImage;
Image10: TImage;
BtnYazdir: TSpeedButton;
Label16: TLabel;
Label17: TLabel;
BtnKapat: TSpeedButton;
StatusBar1: TStatusBar;
procedure FormShow(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);

```



```

    procedure DetayGridGetAlignment(Sender: TObject; ARow, ACol: Integer;
      var HAlign: TAlignment; var VAlign: TVAlignment);
    procedure BtnYazdirClick(Sender: TObject);
    procedure BtnKapatClick(Sender: TObject);
    procedure Panel5MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
private
  { Private declarations }
public
  { Public declarations }
  actRecerteRecID:String;
end;

var
  FrmReceteDetay: TFrmReceteDetay;

implementation

uses UntDm, UntGlobal, ADODB, DB;

{$R *.dfm}

procedure TFrmReceteDetay.FormShow(Sender: TObject);
var
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
  SetWindowRgn(handle, region, true);
  DM.Qry1.Close;
  DM.Qry1.SQL.Clear;
  DM.Qry1.SQL.Add('Select * From Recete Where RecID='+actRecerteRecID);
  DM.Qry1.Prepared ;
  DM.Qry1.Open;
  with DetayGrid do
  begin
    MergeCells(0,0,4,1);
    MergeCells(0,1,2,1);
    CellProperties[0,1].FontStyle:=[fsBold];
    Colors[0,1]:=clSilver;
    Cells[0,0]:='Reçete Detay';
    Cells[0,1]:='Ürün Bilgileri';
    Cells[0,2]:='Ürün Kodu';
    Cells[0,3]:='Ürün Adı';
    Cells[0,4]:='Ürün Tipi';
    Cells[0,5]:='Spekt No';
    MergeCells(0,6,2,1);
    CellProperties[0,6].FontStyle:=[fsBold];
    Colors[0,6]:=clSilver;
    Cells[0,6] :='Agrega Miktarları';
    Cells[0,7] :='01 No`lu Agrega' ; Cells[1,7] :=FormatFloat('0.000
Kg',DM.Qry1.FieldByName('Agrega01').AsFloat);
    Cells[0,8] :='02 No`lu Agrega' ; Cells[1,8] :=FormatFloat('0.000
Kg',DM.Qry1.FieldByName('Agrega02').AsFloat);

```

```

Cells[0,9] := '03 No`lu Agrega' ; Cells[1,9] := FormatFloat('0.000
Kg',DM.Qry1.FieldName('Agrega03').AsString);
Cells[0,10] := 'Kum Miktarı' ; Cells[1,10] := FormatFloat('0.000
Kg',DM.Qry1.FieldName('Kum').AsString );
MergeCells(2,1,2,1);
CellProperties[2,1].FontStyle:=[fsBold];
Colors[2,1]:=clSilver;
Cells[2,1] := 'Reçete Bilgileri';
Cells[2,2] := 'Reçete Kodu' ; Cells[3,2] := DM.Qry1.FieldName('ReceteKod').AsString ;
Cells[2,3] := 'Reçete Adı' ; Cells[3,3] := DM.Qry1.FieldName('ReceteAdi').AsString ;
Cells[2,4] := 'Yayın Tarihi'; Cells[3,4] := DM.Qry1.FieldName('RecTarih').AsString ;
MergeCells(2,5,2,1);
CellProperties[2,5].FontStyle:=[fsBold];
Colors[2,5]:=clSilver;
Cells[2,5] := 'Diğer Bilgileri' ;
Cells[2,6] := 'Çimento Miktarı' ; Cells[3,6] := FormatFloat('0.000
Kg',DM.Qry1.FieldName('Cimento').AsString);
Cells[2,7] := 'Su Miktarı' ; Cells[3,7] := FormatFloat('0.000 Kg',DM.Qry1.FieldName('Su').AsString
);
Cells[2,8] := 'Katkı - 1 Miktarı'; Cells[3,8] := FormatFloat('0.000
Kg',DM.Qry1.FieldName('Katki01').AsString);
Cells[2,9] := 'Katkı - 2 Miktarı'; Cells[3,9] := FormatFloat('0.000
Kg',DM.Qry1.FieldName('Katki02').AsString);
Cells[2,10] := 'Karıştırma Süresi'; Cells[3,10] := DM.Qry1.FieldName('Saat').AsString+' Saat '+
DM.Qry1.FieldName('Dakika').AsString+' Dk ' ;
MergeCells(0,11,2,1);
CellProperties[0,11].FontStyle:=[fsBold];
Colors[0,11]:=clSilver;
Cells[0,11] := 'Hazırlayan ' ;
Cells[0,12] := 'Adı' ; Cells[1,12] := DM.Qry1.FieldName('HPerAdi').AsString;
Cells[0,13] := 'Soyadı'; Cells[1,13] := DM.Qry1.FieldName('HPerSoyadi').AsString;
MergeCells(2,11,2,1);
CellProperties[2,11].FontStyle:=[fsBold];
Colors[2,11]:=clSilver;
Cells[2,11] := 'Onaylayan ' ;
Cells[2,12] := 'Adı' ; Cells[3,12] := DM.Qry1.FieldName('OPerAdi').AsString;
Cells[2,13] := 'Soyadı'; Cells[3,13] := DM.Qry1.FieldName('OPerSoyadi').AsString;
end;
end;
// ***
procedure TFrmReceteDetay.FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
if key = vk_F8 then BtnYazdir.Click ;
if key = vk_Escape then BtnKapat.Click ;
end;
// ***
procedure TFrmReceteDetay.DetayGridGetAlignment(Sender: TObject; ARow,
ACol: Integer; var HAlign: TAlignment; var VAlign: TVAlignment);
begin
if ((ACol=1) or (ACol=3)) and ((ARow>5) and (ARow<11)) then
HAlign:=taRightJustify;
end;
// ***
procedure TFrmReceteDetay.BtnYazdirClick(Sender: TObject);
begin

```

```

    AdvStrYazdir(Self,DetayGrid,'Görüntüle','Dikey','');
end;
// ****
procedure TFrmReceteDetay.BtnKapatClick(Sender: TObject);
begin
    self.Close;
end;
// ***
procedure TFrmReceteDetay.Panel5MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntReceteEkle;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, ExtCtrls, StdCtrls, Buttons, Grids, BaseGrid, AdvGrid, Mask,
    ToolEdit, ComCtrls, CurrEdit, ImgList, ToolWin;

type
    TFrmReceteEkle = class(TForm)
        GroupBox1: TGroupBox;
        txtHPerAdi: TEdit;
        Label9: TLabel;
        txtHPerSoyadi: TEdit;
        GroupBox2: TGroupBox;
        Label15: TLabel;
        txtOPerSoyadi: TEdit;
        txtOPerAdi: TEdit;
        HAdi: TSpeedButton;
        OAdi: TSpeedButton;
        PageControl2: TPageControl;
        TabSheet3: TTabSheet;
        Label3: TLabel;
        Label10: TLabel;
        Label12: TLabel;
        txtSpektNo: TEdit;
        txtUrunKod: TEdit;
        txtUrunAd: TEdit;
        txtUrunTip: TEdit;
        PageControl4: TPageControl;
        TabSheet4: TTabSheet;
        Label5: TLabel;
        Label6: TLabel;
        Label4: TLabel;
        txtReceteKod: TEdit;
        txtReceteAdi: TEdit;
        DTRecTarih: TDateEdit;
    end;

```

```

PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label8: TLabel;
Label11: TLabel;
Label13: TLabel;
Label14: TLabel;
txtAgrega01: TCurrencyEdit;
txtAgrega02: TCurrencyEdit;
txtAgrega03: TCurrencyEdit;
txtKum: TCurrencyEdit;
PageControl3: TPageControl;
TabSheet2: TTabSheet;
Label18: TLabel;
Label20: TLabel;
Label21: TLabel;
Label22: TLabel;
Label19: TLabel;
txtCimento: TCurrencyEdit;
txtSu: TCurrencyEdit;
txtKatki01: TCurrencyEdit;
txtKatki02: TCurrencyEdit;
txtSaat: TCurrencyEdit;
txtDakika: TCurrencyEdit;
Label7: TLabel;
Panel5: TPanel;
Image9: TImage;
Image10: TImage;
BtnKaydet: TSpeedButton;
Label16: TLabel;
Label17: TLabel;
BtnKapat: TSpeedButton;
Label1: TLabel;
Label23: TLabel;
StatusBar1: TStatusBar;
procedure FormCreate(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure HAdiClick(Sender: TObject);
procedure BtnKaydetClick(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure Panel5MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);

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

var
  FrmReceteEkle: TFrmReceteEkle;

implementation

uses ADODB, DB, UntDlgPersonalSearch, UntDm, StrUtils, UntGlobal;

```

{ \$R \*.dfm }

```
procedure TFrmReceteEkle.FormCreate(Sender: TObject);
begin
    DTRecTarih.Date :=Date;
end;
// ***
procedure TFrmReceteEkle.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if key = vk_f4 then BtnKaydet.Click ;
    if key = vk_Escape then BtnKapat.Click ;
end;
// ***
procedure TFrmReceteEkle.HAdiClick(Sender: TObject);
begin
    FrmDlgPersonalSearch:=TFrmDlgPersonalSearch.Create(Self);
    FrmDlgPersonalSearch.ShowModal;
    if not FrmDlgPersonalSearch.iptal then
    begin
        if (Sender as TSpeedButton).Tag =0 then
        begin
            txtHPerAdi.Text :=FrmDlgPersonalSearch.APerAdi;
            txtHPerSoyadi.Text :=FrmDlgPersonalSearch.APerSoyadi;
        end
        else
        begin
            txtOPerAdi.Text :=FrmDlgPersonalSearch.APerAdi;
            txtOPerSoyadi.Text :=FrmDlgPersonalSearch.APerSoyadi;
        end;
    end;
    FrmDlgPersonalSearch.Release;
end;
// ***
procedure TFrmReceteEkle.BtnKaydetClick(Sender: TObject);
begin
    if txtReceteKod.Text="" then
    begin
        Hata('Reçete Kod Bilgisi Eksik. ' + #13 + 'Lütfen Tanımlayınız. ');
        txtReceteKod.SetFocus;
        exit;
    end;
    if txtReceteAdi.Text="" then
    begin
        Hata('Reçete Ad Bilgisi Eksik. ' + #13 + 'Lütfen Tanımlayınız. ');
        txtReceteAdi.SetFocus;
        exit;
    end;
    if (txtHPerAdi.Text="") or (txtOPerAdi.Text="") then
    begin
        Hata('Hazırlayan veya Onaylan Bilgisi Eksik. ' + #13 + 'Lütfen Tanımlayınız. ');
        exit;
    end;
    if txtDakika.Value>59 then
    begin
```

```

Hata('Dakika Değeri 60`dan Büyük Olamaz.'+ #13 + 'Lütfen Tanımlayınız. ');
exit;
end;
with DM.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Insert Into Recete (SpektNo,ReceteKod,ReceteAdi,RecTarih,Agrega01,');
SQL.Add('Agrega02,Agrega03,Kum,Cimento,Su,Katki01,Katki02,ToplamMiktar,Saat,');
SQL.Add('Dakika,HPerAdi,HPerSoyadi,OPerAdi,OPerSoyadi) ');
SQL.Add('Values (:SpektNo,:ReceteKod,:ReceteAdi,:RecTarih,:Agrega01,');
SQL.Add(':Agrega02,:Agrega03,:Kum,:Cimento,:Su,:Katki01,:Katki02,');
SQL.Add(':ToplamMiktar,:Saat,');
SQL.Add(':Dakika,:HPerAdi,:HPerSoyadi,:OPerAdi,:OPerSoyadi)');
Parameters.ParamByName('SpektNo').Value :=txtSpektNo.Text;
Parameters.ParamByName('ReceteKod').Value :=txtReceteKod.Text;
Parameters.ParamByName('ReceteAdi').Value :=txtReceteAdi.Text;
Parameters.ParamByName('RecTarih').Value :=DTRecTarih.Text;
Parameters.ParamByName('Agrega01').Value :=txtAgrega01.Value;
Parameters.ParamByName('Agrega02').Value :=txtAgrega02.Value;
Parameters.ParamByName('Agrega03').Value :=txtAgrega03.Value;
Parameters.ParamByName('Kum').Value :=txtKum.Value;
Parameters.ParamByName('Cimento').Value :=txtCimento.Value;
Parameters.ParamByName('Su').Value :=txtSu.Value;
Parameters.ParamByName('Katki01').Value :=txtKatki01.Value;
Parameters.ParamByName('Katki02').Value :=txtKatki02.Value;

Parameters.ParamByName('ToplamMiktar').Value:=txtAgrega01.Value+txtAgrega02.Value+txtAgrega03.
Value +txtKum.Value+
txtCimento.Value+txtSu.Value+txtKatki01.Value+txtKatki02.Value;
Parameters.ParamByName('Saat').Value :=RightStr('00'+txtSaat.Text,2);
Parameters.ParamByName('Dakika').Value :=RightStr('00'+txtDakika.Text,2);
Parameters.ParamByName('HPerAdi').Value :=txtHPerAdi.Text;
Parameters.ParamByName('HPerSoyadi').Value:=txtHPerSoyadi.Text;
Parameters.ParamByName('OPerAdi').Value :=txtOPerAdi.Text;
Parameters.ParamByName('OPerSoyadi').Value:=txtOPerSoyadi.Text;
Prepared ;
ExecSQL;
end;
CompTemizle(Self,1);
CompTemizle(Self,3);
txtReceteKod.SetFocus;
end;
// ***
procedure TFrmReceteEkle.BtnKapatClick(Sender: TObject);
begin
Self.Close;
end;
// ***
procedure TFrmReceteEkle.FormShow(Sender: TObject);
var
region: HRgn;
begin
StatusBar1.SimplePanel:=True;
StatusBar1.SimpleText:=ProCaption;
region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);

```

```

    SetWindowRgn(handle, region, true);
end;
// ***
procedure TFrmReceteEkle.Panel5MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntReceteListesi;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, ExtCtrls, StdCtrls, Buttons, Grids, Mask,
    ToolEdit, ComCtrls, CurrEdit, ImgList, ToolWin, DB, ADODB,
    AdvToolBar, AdvToolBarStylers, DBCtrls, XPMAN, cxStyles, cxCustomData,
    cxGraphics, cxFilter, cxData, cxDataStorage, cxEdit, cxDBData,
    cxGridCustomTableView, cxGridTableView, cxGridDBTableView, cxGridLevel,
    cxClasses, cxControls, cxGridCustomView, cxGrid, dxPSGlbl, dxPSUtl,
    dxPSEngn, dxPrnPg, dxBkgnd, dxWrap, dxPrnDev, dxPSCompsProvider,
    dxPSFillPatterns, dxPSEdgePatterns, dxPSCore, dxPScxCommon, dxPScxGridLnk;

type
    TFrmReceteListesi = class(TForm)
        AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
        DSUrunlerQuery: TDataSource;
        UrunlerQuery: TADOQuery;
        Panel1: TPanel;
        DSReceteQuery: TDataSource;
        ReceteQuery: TADOQuery;
        cxGrid1DBTableView1: TcxGridDBTableView;
        cxGrid1Level1: TcxGridLevel;
        cxGrid1: TcxGrid;
        cxGrid1DBTableView1ReceteKod: TcxGridDBCColumn;
        cxGrid1DBTableView1ReceteAdi: TcxGridDBCColumn;
        cxGrid1DBTableView1RecTarih: TcxGridDBCColumn;
        cxGrid1DBTableView1Agrega01: TcxGridDBCColumn;
        cxGrid1DBTableView1Agrega02: TcxGridDBCColumn;
        cxGrid1DBTableView1Agrega03: TcxGridDBCColumn;
        cxGrid1DBTableView1Kum: TcxGridDBCColumn;
        cxGrid1DBTableView1Cimento: TcxGridDBCColumn;
        cxGrid1DBTableView1Su: TcxGridDBCColumn;
        cxGrid1DBTableView1Katki01: TcxGridDBCColumn;
        cxGrid1DBTableView1Katki02: TcxGridDBCColumn;
        cxGrid1DBTableView1ToplamMiktar: TcxGridDBCColumn;
        Panel2: TPanel;
        cxGrid1DBTableView1Sure: TcxGridDBCColumn;
        cxGrid1DBTableView1Hazirlayan: TcxGridDBCColumn;
        cxGrid1DBTableView1Onaylayan: TcxGridDBCColumn;
        Panel4: TPanel;
        Image9: TImage;
    end;

```

```

lblCaption: TLabel;
Panel3: TPanel;
Image2: TImage;
AdvDockPanel1: TAdvDockPanel;
AdvToolBar1: TAdvToolBar;
BtnKapat: TAdvToolBarButton;
BtnSil: TAdvToolBarButton;
BtnDegistir: TAdvToolBarButton;
BtnEkle: TAdvToolBarButton;
BtnDetay: TAdvToolBarButton;
BtnYazdir: TAdvToolBarButton;
Label7: TLabel;
DBtxtKod: TDBEdit;
Label10: TLabel;
DBtxtAdi: TDBEdit;
Label12: TLabel;
Label3: TLabel;
DBtxtTipi: TDBEdit;
DBtxtSpektNo: TDBEdit;
DBNavigator1: TDBNavigator;
Label16: TLabel;
Label17: TLabel;
StatusBar1: TStatusBar;
PrintPreview: TdxComponentPrinter;
PrintPreviewLink1: TdxGridReportLink;
procedure FormCreate(Sender: TObject);
procedure BtnCloseClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtnKapatClick(Sender: TObject);
procedure BtnEkleClick(Sender: TObject);
procedure BtnDegistirClick(Sender: TObject);
procedure BtnSilClick(Sender: TObject);
procedure BtnDetayClick(Sender: TObject);
procedure DBtxtKodChange(Sender: TObject);
procedure BtnYazdirClick(Sender: TObject);
procedure ReceteGridDb1Click(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure Image9MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);

private
{ Private declarations }
public
{ Public declarations }
procedure SetReceteView();
end;

var
  FrmReceteListesi: TFrmReceteListesi;

implementation

uses
  UntDm, StrUtils, UntDlgPersonalSearch, UntReceteEkle,
  UntReceteDegistir, UntReceteSil, UntReceteDetay, UntGlobal; //UntDlgPersonalSearch,

```



```
{ $R *.dfm }
```

```
procedure TFrmReceteListesi.SetReceteView();
```

```
begin
```

```
with ReceteQuery do
```

```
begin
```

```
Close;
```

```
SQL.Clear;
```

```
SQL.Add('Select * , (Saat + ":" + Dakika ) as sure , ');
```

```
SQL.Add('(HPerAdi + " " + HPerSoyadi ) as Hazirlayan, ');
```

```
SQL.Add('(OPerAdi + " " + OPerSoyadi ) as Onaylayan ');
```

```
SQL.Add(' From Recete Where SpektNo="' +
```

```
UrunlerQuery.FieldName('SpektNo').AsString + "' ');
```

```
open;
```

```
(FieldName('Agrega01') as TNumericField).DisplayFormat:='0.000 Kg';
```

```
(FieldName('Agrega02') as TNumericField).DisplayFormat:='0.000 Kg';
```

```
(FieldName('Agrega03') as TNumericField).DisplayFormat:='0.000 Kg';
```

```
(FieldName('Kum') as TNumericField).DisplayFormat:='0.000 Kg';
```

```
(FieldName('Cimento') as TNumericField).DisplayFormat:='0.000 Kg';
```

```
(FieldName('Su') as TNumericField).DisplayFormat:='0.000 Kg';
```

```
(FieldName('Katki01') as TNumericField).DisplayFormat:='0.000 Kg';
```

```
(FieldName('Katki02') as TNumericField).DisplayFormat:='0.000 Kg';
```

```
(FieldName('ToplamMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
```

```
end;
```

```
end;
```

```
// ***
```

```
procedure TFrmReceteListesi.FormCreate(Sender: TObject);
```

```
begin
```

```
Caption:=ProCaption ;
```

```
{Table Name:Urunler;RecID           SpektNo UrunKod           UrunAd UrunTip}
```

```
with UrunlerQuery do
```

```
begin
```

```
Close;
```

```
SQL.Clear;
```

```
SQL.Add('Select RecID,SpektNo,UrunKod,UrunAd,UrunTip From Urunler Order By RecID Asc');
```

```
Prepared ;
```

```
Open;
```

```
end;
```

```
SetReceteView;
```

```
end;
```

```
// ***
```

```
procedure TFrmReceteListesi.BtnCloseClick(Sender: TObject);
```

```
begin
```

```
self.Close;
```

```
end;
```

```
// ***
```

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

```
Shift: TShiftState);
```

```
begin
```

```
if key = vk_F2 then BtnDetay.Click;
```

```
if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
```

```
if (key = vk_F5) and (BtnDegistir.Visible=True) then BtnDegistir.Click;
```

```
if (key = vk_F6) and (BtnSil.Visible=True) then BtnSil.Click;
```

```

    if (key = vk_F8) and (BtnYazdir.Visible=True) then BtnYazdir.Click;
    if key = vk_F10 then BtnKapat.Click;
end;
// ***
procedure TFrmReceteListesi.BtnKapatClick(Sender: TObject);
begin
    self.Close;
end;
// ***
procedure TFrmReceteListesi.BtnEkleClick(Sender: TObject);
begin
    if (UrunlerQuery.Active=False) or (UrunlerQuery.RecordCount=0) then Exit;
    FrmReceteEkle:=TFrmReceteEkle.Create(Self);
    with FrmReceteEkle do
    begin
        CompTemizle(FrmReceteEkle,1);
        CompTemizle(FrmReceteEkle,3);
        txtSpektNo.Text :=UrunlerQuery.FieldName('SpektNo').AsString;
        txtUrunKod.Text :=UrunlerQuery.FieldName('UrunKod').AsString;
        txtUrunAd.Text :=UrunlerQuery.FieldName('UrunAd').AsString ;
        txtUrunTip.Text :=UrunlerQuery.FieldName('UrunTip').AsString;
        ShowModal;
        Release;
    end;
    SetReceteView;
end;
// ***
procedure TFrmReceteListesi.BtnDegistirClick(Sender: TObject);
begin
    if (ReceteQuery.Active=False) or (ReceteQuery.RecordCount=0) then Exit;
    FrmReceteDegistir:=TFrmReceteDegistir.Create(Self);
    with FrmReceteDegistir do
    begin
        CompTemizle(FrmReceteDegistir,1);
        CompTemizle(FrmReceteDegistir,3);
        actRecerteRecID:=ReceteQuery.FieldName('RecID').AsString;
        txtSpektNo.Text :=UrunlerQuery.FieldName('SpektNo').AsString;
        txtUrunKod.Text :=UrunlerQuery.FieldName('UrunKod').AsString;
        txtUrunAd.Text :=UrunlerQuery.FieldName('UrunAd').AsString ;
        txtUrunTip.Text :=UrunlerQuery.FieldName('UrunTip').AsString;
        ShowModal;
        Release;
    end;
    SetReceteView;
end;
// ***
procedure TFrmReceteListesi.BtnSilClick(Sender: TObject);
begin
    if (ReceteQuery.Active=False) or (ReceteQuery.RecordCount=0) then Exit;
    FrmReceteSil:=TFrmReceteSil.Create(Self);
    with FrmReceteSil do
    begin
        CompTemizle(FrmReceteSil,1);
        CompTemizle(FrmReceteSil,3);
        actRecerteRecID:=ReceteQuery.FieldName('RecID').AsString;
        txtSpektNo.Text :=UrunlerQuery.FieldName('SpektNo').AsString;

```

```

txtUrunKod.Text :=UrunlerQuery.FieldByName('UrunKod').AsString;
txtUrunAd.Text :=UrunlerQuery.FieldByName('UrunAd').AsString ;
txtUrunTip.Text :=UrunlerQuery.FieldByName('UrunTip').AsString;
ShowModal;
Release;
end;
SetReceteView;
end;
// ***
procedure TFrmReceteListesi.BtnDetayClick(Sender: TObject);
begin
if (ReceteQuery.Active=False) or (ReceteQuery.RecordCount=0) then Exit;
FrmReceteDetay:=TFrmReceteDetay.Create(Self);
with FrmReceteDetay do
begin
BtnYazdir.Enabled:=self.BtnYazdir.Visible;
CompTemizle(FrmReceteDetay,1);
CompTemizle(FrmReceteDetay,3);
actReceteRecID:=ReceteQuery.FieldByName('RecID').AsString;
DetayGrid.Cells[1,2]:=UrunlerQuery.FieldByName('UrunKod').AsString;
DetayGrid.Cells[1,3]:=UrunlerQuery.FieldByName('UrunAd').AsString ;
DetayGrid.Cells[1,4]:=UrunlerQuery.FieldByName('UrunTip').AsString;
DetayGrid.Cells[1,5]:=UrunlerQuery.FieldByName('SpektNo').AsString;
ShowModal;
Release;
end;
end;
// ***
procedure TFrmReceteListesi.DBtxtKodChange(Sender: TObject);
begin
SetReceteView;
end;
// ***
procedure TFrmReceteListesi.BtnYazdirClick(Sender: TObject);
var
tmpstr:string;
begin
if (ReceteQuery.Active=False) or (ReceteQuery.RecordCount=0) then Exit;
tmpstr:=' Ürün Kod : ' + DBtxtKod.Text + ' / Ürün Adı : ' + DBtxtAdi.Text +
' / Ürün Tipi : ' + DBtxtTipi.Text + ' / Spekt No : ' + DBtxtSpektNo.Text;

PrintPreviewLink1.ReportTitleText:= tmpstr;
PrintPreviewLink1.PrinterPage.Orientation:=poLandscape;
PrintPreviewLink1.PrinterPage.PageHeader.RightTitle .Clear;
PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi:' +
FormatDateTime('dd.mm.yyyy',date));
PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Clear;
PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
PrintPreview.Preview(True,PrintPreviewLink1);
end;
// ***
procedure TFrmReceteListesi.ReceteGridDbClick(Sender: TObject);
begin
BtnDetay.Click;
end;
// ***

```

```

procedure TFrmReceteListesi.FormShow(Sender: TObject);
var
region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
    SetWindowRgn(handle, region, true);
    KullaniciHaklari(Self.Tag,AdvToolBar1);
end;
// ***
procedure TFrmReceteListesi.Image9MouseDown(Sender: TObject;
Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntReceteSil;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, ExtCtrls, ComCtrls, CurrEdit, Mask, ToolEdit, StdCtrls, Buttons;

type
TFrmReceteSil = class(TForm)
    Panel5: TPanel;
    Image9: TImage;
    Image10: TImage;
    BtnSil: TSpeedButton;
    Label16: TLabel;
    Label17: TLabel;
    BtnKapat: TSpeedButton;
    PageControl2: TPageControl;
    TabSheet3: TTabSheet;
    Label3: TLabel;
    Label10: TLabel;
    Label12: TLabel;
    Label7: TLabel;
    txtSpektNo: TEdit;
    txtUrunKod: TEdit;
    txtUrunAd: TEdit;
    txtUrunTip: TEdit;
    PageControl4: TPageControl;
    TabSheet4: TTabSheet;
    Label5: TLabel;
    Label6: TLabel;
    Label4: TLabel;
    Label24: TLabel;
    txtReceteKod: TEdit;
    txtReceteAdi: TEdit;

```

```

DTRecTarih: TDateEdit;
txtRevNo: TEdit;
PageControl3: TPageControl;
TabSheet2: TTabSheet;
Label18: TLabel;
Label20: TLabel;
Label21: TLabel;
Label22: TLabel;
Label19: TLabel;
txtCimento: TCurrencyEdit;
txtSu: TCurrencyEdit;
txtKatki01: TCurrencyEdit;
txtKatki02: TCurrencyEdit;
txtSaat: TCurrencyEdit;
txtDakika: TCurrencyEdit;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
Label8: TLabel;
Label11: TLabel;
Label13: TLabel;
Label14: TLabel;
txtAgrega01: TCurrencyEdit;
txtAgrega02: TCurrencyEdit;
txtAgrega03: TCurrencyEdit;
txtKum: TCurrencyEdit;
StatusBar1: TStatusBar;
procedure FormShow(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtnKapatClick(Sender: TObject);
procedure BtnSilClick(Sender: TObject);
procedure Panel5MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
private
  { Private declarations }
public
  { Public declarations }
  actRecerteRecID:String;
end;

var
  FrmReceteSil: TFrmReceteSil;

implementation

uses UntDm, UntGlobal, ADODB, UntOnay, Math;

{$R *.dfm}

procedure TFrmReceteSil.FormShow(Sender: TObject);
var
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);

```

```

SetWindowRgn(handle, region, true);
with DM.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Select * From Recete Where RecID='+actRecerteRecID);
Prepared ;
Open;
txtSpektNo.Text := FieldByName('SpektNo').AsString ;
txtReceteKod.Text := FieldByName('ReceteKod').AsString ;
txtReceteAdi.Text := FieldByName('ReceteAdi').AsString ;
DTRecTarih.Text := FieldByName('RecTarih').AsString ;
txtAgrega01.Value := FieldByName('Agrega01').AsFloat ;
txtAgrega02.Value := FieldByName('Agrega02').AsFloat ;
txtAgrega03.Value := FieldByName('Agrega03').AsFloat ;
txtKum.Value := FieldByName('Kum').AsFloat ;
txtCimento.Value := FieldByName('Cimento').AsFloat ;
txtSu.Value := FieldByName('Su').AsFloat ;
txtKatki01.Value := FieldByName('Katki01').AsFloat ;
txtKatki02.Value := FieldByName('Katki02').AsFloat ;
txtSaat.AsInteger := FieldByName('Saat').AsInteger ;
txtDakika.AsInteger := FieldByName('Dakika').AsInteger ;
end;
end;
// ***
procedure TFrmReceteSil.FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
if key = vk_f6 then BtnSil.Click ;
if key = vk_Escape then BtnKapat.Click ;
end;
// ***
procedure TFrmReceteSil.BtnKapatClick(Sender: TObject);
begin
self.Close;
end;
// ***
procedure TFrmReceteSil.BtnSilClick(Sender: TObject);
begin
if Onay('Reçete Silinecektir.'+#13+'Devam Etmek İstiyor musunuz?')=False then exit;
with DM.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Select * From Uretim Where ReceteKod=Bul');
Parameters.ParamByName('Bul').Value:=txtReceteKod.Text;
Prepared;
Open;
if Recordcount>0 then
begin
Hata('Reçeteye Ait Üretim Emri Olduğundan Reçete Silinemez. ');
exit;
end;
Close;
SQL.Clear;
SQL.Add('Delete * From Recete Where RecID='+actRecerteRecID);

```

```

    Prepared ;
    ExecSQL;
end;
    Self.Close;
end;
// ***
procedure TFrmReceteSil.Panel5MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

```

\*\*\*\*\*

```
unit UntStokHareketListe;
```

```
interface
```

```
uses
```

```

    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, Buttons, ExtCtrls, Grids, BaseGrid, AdvGrid, DBAdvGrd,
    ComCtrls, ToolWin, ImgList, Mask, ToolEdit, DB, ADODB, CurrEdit, Menus,
    DBAdvGrid, AdvToolBar, AdvToolBarStylers, cxStyles, cxCustomData,
    cxGraphics, cxFilter, cxData, cxDataStorage, cxEdit, cxDBData,
    cxGridLevel, cxClasses, cxControls, cxGridCustomView,
    cxGridCustomTableView, cxGridTableView, cxGridDBTableView, cxGrid,
    dxPSGlbl, dxPSUtl, dxPSEngn, dxPrnPg, dxBkgnd, dxWrap, dxPrnDev,
    dxPSCompsProvider, dxPSFillPatterns, dxPSEdgePatterns, dxPSCore,
    dxPScxCommon, dxPScxGridLnk;

```

```
type
```

```

TFrmStokHareketListe = class(TForm)
    Panel2: TPanel;
    Label3: TLabel;
    Label5: TLabel;
    txtStokKod: TEdit;
    txtUrunAdi: TEdit;
    DSQrySHareket: TDataSource;
    QrySHareket: TADOQuery;
    Label6: TLabel;
    txtStokMiktar: TCurrencyEdit;
    AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
    cxGrid1DBTableView1: TcxGridDBTableView;
    cxGrid1Level1: TcxGridLevel;
    cxGrid1: TcxGrid;
    Panel1: TPanel;
    CBoxDate: TCheckBox;
    DTStart: TDateEdit;
    DTEnd: TDateEdit;
    pnlHeader: TPanel;
    Image9: TImage;
    Panel3: TPanel;
    Image2: TImage;

```

```

Label16: TLabel;
Label17: TLabel;
AdvDockPanel1: TAdvDockPanel;
AdvToolBar1: TAdvToolBar;
BtnKapat: TAdvToolBarButton;
BtnAra: TAdvToolBarButton;
BtnSil: TAdvToolBarButton;
BtnDegistir: TAdvToolBarButton;
BtnEkle: TAdvToolBarButton;
BtnListele: TAdvToolBarButton;
BtnYazdir: TAdvToolBarButton;
cxGrid1DBTableView1aylar: TcxGridDBCColumn;
cxGrid1DBTableView1OpDate: TcxGridDBCColumn;
cxGrid1DBTableView1OpNumber: TcxGridDBCColumn;
cxGrid1DBTableView1OpDefinition: TcxGridDBCColumn;
cxGrid1DBTableView1InValue: TcxGridDBCColumn;
cxGrid1DBTableView1OutValue: TcxGridDBCColumn;
cxGrid1DBTableView1Yil: TcxGridDBCColumn;
pnlIslem: TPanel;
Label7: TLabel;
cmbHareketTur: TComboBox;
Label2: TLabel;
DTIslemTarihi: TDateEdit;
Label4: TLabel;
txtBelgeNo: TCurrencyEdit;
Label8: TLabel;
txtAciklama: TEdit;
Label9: TLabel;
txtMiktar: TCurrencyEdit;
BtnUygula: TSpeedButton;
BtnIptal: TSpeedButton;
StatusBar1: TStatusBar;
PrintPreview: TdxComponentPrinter;
PrintPreviewLink1: TdxGridReportLink;

procedure FormShow(Sender: TObject);
procedure BtnEkleClick(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure BtnDegistirClick(Sender: TObject);
procedure BtnAraClick(Sender: TObject);
procedure BtnIptalClick(Sender: TObject);
procedure BtnUygulaClick(Sender: TObject);
procedure BtnListeleClick(Sender: TObject);
procedure BtnSilClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure pnlHeaderMouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure BtnYazdirClick(Sender: TObject);

private
{ Private declarations }
public
{ Public declarations }
procedure SHKaydet();
procedure SHDegistir();

```



```

    procedure SHSil();
    procedure Islem(Durum: Boolean; BtnBaslik: String);
    procedure SetGridQuery();
end;

var
    FrmStokHareketListe: TFrmStokHareketListe;

implementation

uses  UntGlobal, UntDm, UntUyari, StrUtils, UntDlgStockSearch;

{$R *.dfm}

procedure TFrmStokHareketListe.SHSil();
var
    ActRecID:String;
    EskiMiktar:Double;
begin
    if Onay('Stok Hareketi Silenecek.')=False then exit;
    ActRecID :=QrySHareket.FieldByName('ID').AsString;

    EskiMiktar:=QrySHareket.FieldByName('In Value').AsFloat+QrySHareket.FieldByName('Out Value').AsFl
    oat;
    with DM.QryArama do
    begin
        Close;
        SQL.Clear;
        if cmbHareketTur.ItemIndex=0 then
            SQL.Add('Update Stok Set Miktar=Miktar-:eski Where Kod=:Bul ')
        else
            SQL.Add('Update Stok Set Miktar=Miktar+:eski Where Kod=:Bul ');
        Parameters.ParamByName('eski').Value:=EskiMiktar;
        Parameters.ParamByName('bul').Value :=txtStokKod.Text;
        Prepared ;
        ExecSQL;
        Close;
        SQL.Clear;
        SQL.Add('Delete * From StokExtre2005');
        SQL.Add('Where ID='+ActRecID);
        ExecSQL ;
        Close;
        SQL.Clear;
        SQL.Add('Select * from Stok Where Kod=:Bul ');
        Parameters.ParamByName('bul').Value :=txtStokKod.Text;
        Open;
        txtStokMiktar.Value:=FieldByName('Miktar').AsFloat;
        txtStokMiktar.DisplayFormat:='0.000 ' + FieldByName('Birim').AsString;
    end;
    SetGridQuery;
    BtnIptal.Click;
end;
// ***
procedure TFrmStokHareketListe.SHDegistir();
var
    ActRecID:String;

```

```

    EskiMiktar:Double;
begin
if txtBelgeNo.Value =0 then
begin
    Hata('Lütfen Belge No`sunu Giriniz. ');
    txtBelgeNo.SetFocus;
    exit;
end;
if txtMiktar.Value =0 then
begin
    Hata('Lütfen Miktar Giriniz. ');
    txtMiktar.SetFocus;
    exit;
end;
    ActRecID :=QrySHareket.FieldByName('ID').AsString;

EskiMiktar:=QrySHareket.FieldByName('In Value').AsFloat+QrySHareket.FieldByName('Out Value').AsFl
oat;
with DM.QryArama do
begin
    Close;
    SQL.Clear;
    if cmbHareketTur.ItemIndex=0 then
        SQL.Add('Update Stok Set Miktar=Miktar-:eski Where Kod=:Bul ');
    else
        SQL.Add('Update Stok Set Miktar=Miktar+:eski Where Kod=:Bul ');
    Parameters.ParamByName('eski').Value:=EskiMiktar;
    Parameters.ParamByName('bul').Value :=txtStokKod.Text;
    Prepared ;
    ExecSQL;
    Close;
    SQL.Clear;
    SQL.Add('Select * From StokExtre2005');
    SQL.Add('Where ID='+ActRecID);
    Prepared ;
    Open;
    Edit;
    FieldByName('Kod').AsString :=txtStokKod.Text;
    FieldByName('OpDate').AsString :=DTIslemTarihi.Text;
    FieldByName('OpNumber').AsString:=txtBelgeNo.Text;
    FieldByName('OpDefinition').AsString:=txtAciklama.Text;
    if cmbHareketTur.ItemIndex=0 then
    begin
        FieldByName('OpType').AsString :='SG';
        FieldByName('In Value').AsFloat :=txtMiktar.Value;
        FieldByName('Out Value').AsFloat :=0;
    end
    else
    begin
        FieldByName('OpType').AsString :='SC';
        FieldByName('In Value').AsFloat :=0;
        FieldByName('Out Value').AsFloat :=txtMiktar.Value;
    end;
    post;
    Close;
    SQL.Clear;

```

```

if cmbHareketTur.ItemIndex=0 then
    SQL.Add('Update Stok Set Miktar=Miktar+:yeni Where Kod=:Bul ')
else
    SQL.Add('Update Stok Set Miktar=Miktar -:yeni Where Kod=:Bul ');
Parameters.ParamByName('Yeni').Value:=txtMiktar.Value;
Parameters.ParamByName('bul').Value :=txtStokKod.Text;
Prepared ;
ExecSQL;
Close;
SQL.Clear;
SQL.Add('Select * from Stok Where Kod=:Bul ');
Parameters.ParamByName('bul').Value :=txtStokKod.Text;
Open;
txtStokMiktar.Value:=FieldByName('Miktar').AsFloat;
txtStokMiktar.DisplayFormat:='0.000 ' + FieldByName('Birim').AsString;
end;
SetGridQuery;
BtnIptal.Click;
end;
// ***
procedure TFrmStokHareketListe.SHKaydet();
begin
if txtBelgeNo.Value =0 then
begin
    Hata('Lütfen Belge No`sunu Giriniz. ');
    txtBelgeNo.SetFocus;
    exit;
end;
if txtMiktar.Value =0 then
begin
    Hata('Lütfen Miktar Giriniz. ');
    txtMiktar.SetFocus;
    exit;
end;
with DM.QryArama do
begin
    Close;
    SQL.Clear;
    SQL.Add('Select * From StokExtre2005');
    Prepared ;
    Open;
    Append;
    FieldByName('Kod').AsString :=txtStokKod.Text;
    FieldByName('OpDate').AsString :=DTIslemTarihi.Text;
    FieldByName('OpNumber').AsString:=txtBelgeNo.Text;
    FieldByName('OpDefinition').AsString:=txtAciklama.Text;
    if cmbHareketTur.ItemIndex=0 then
    begin
        FieldByName('OpType').AsString :='SG';
        FieldByName('InValue').AsFloat :=txtMiktar.Value;
        FieldByName('OutValue').AsFloat :=0;
    end
    else
    begin
        FieldByName('OpType').AsString :='SC';
        FieldByName('InValue').AsFloat :=0;
    end
end;
end;

```

```

        FieldByName('OutValue').AsFloat :=txtMiktar.Value;
    end;
post;
Close;
SQL.Clear;
if cmbHareketTur.ItemIndex=0 then
    SQL.Add('Update Stok Set Miktar=Miktar+:yeni Where Kod=:Bul ');
else
    SQL.Add('Update Stok Set Miktar=Miktar -:yeni Where Kod=:Bul ');
Parameters.ParamByName('Yeni').Value:=txtMiktar.Value;
Parameters.ParamByName('bul').Value :=txtStokKod.Text;
Prepared ;
ExecSQL;
Close;
SQL.Clear;
SQL.Add('Select * from Stok Where Kod=:Bul ');
Parameters.ParamByName('bul').Value :=txtStokKod.Text;
Open;
txtStokMiktar.Value:=FieldByName('Miktar').AsFloat;
txtStokMiktar.DisplayFormat:='0.000 ' + FieldByName('Birim').AsString;
end;
SetGridQuery;
DTIslemTarihi.Date:=Date;
txtBelgeNo.Text :=";
txtAciklama.Text :=";
txtMiktar.Value :=0;
DTIslemTarihi.SetFocus;
end;
// ***
procedure TFrmStokHareketListe.Islem(Durum: Boolean; BtnBaslik: String);
begin
    pnlHeader.Enabled :=Not Durum;
    cxGrid1.Enabled :=Not Durum;
    pnlIslem.Visible :=Durum;
    BtnUygula.Caption :=BtnBaslik;
end;
// ***
procedure TFrmStokHareketListe.SetGridQuery();
begin
    with QrySHareket do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select *, year(opdate) as yil,month(opdate) as aylar From StokExtre2005 ');
        SQL.Add(' Where Kod="'+txtStokKod.Text +"' ' );
        if CBoxDate.Checked then
            begin
                SQL.Add('and (opdate Between :Tarih01 and :Tarih02 )');
                Parameters.ParamByName('Tarih01').Value :=DTStart.Date;
                Parameters.ParamByName('Tarih02').Value :=DTEnd.Date;
            end;
        SQL.Add('order by Opdate,ID Asc');
        Prepared ;
        open;
        (FieldByName('InValue') as TNumericField ).DisplayFormat:='#,##0.000';
        (FieldByName('OutValue')as TNumericField ).DisplayFormat:='#,##0.000';
    end;
end;

```

```

end;
end;
// ***
procedure TFrmStokHareketListe.FormShow(Sender: TObject);
var
region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
    SetWindowRgn(handle, region, true);
    KullaniciHaklari(Self.Tag,AdvToolBar1);
    Islem(False,'---');
    DTStart.Date:=Date;
    DTEnd.Date:=Date;
end;
// ***
procedure TFrmStokHareketListe.BtnEkleClick(Sender: TObject);
begin
if txtStokKod.Text="" then
begin
    Hata('Lütfen Stok Tanımlayınız. ');
    Exit;
end;
Islem(True,'Kaydet');
cmbHareketTur.Enabled:=True;
DTIslemTarihi.Date:=Date;
txtBelgeNo.Text := "";
txtAciklama.Text := "";
txtMiktar.Value :=0;
DTIslemTarihi.SetFocus;
end;
// ***
procedure TFrmStokHareketListe.BtnKapatClick(Sender: TObject);
begin
    self.Close;
end;
// ***
procedure TFrmStokHareketListe.BtnDegistirClick(Sender: TObject);
begin
if (QrySHareket.Active=False) or (QrySHareket.RecordCount=0) then exit;
islem (True,'Güncelle');
if QrySHareket.FieldName('InValue').AsFloat>QrySHareket.FieldName('OutValue').AsFloat then
    cmbHareketTur.ItemIndex:=0
else
    cmbHareketTur.ItemIndex:=1;
cmbHareketTur.Enabled:=False;
DTIslemTarihi.Text:=QrySHareket.FieldName('OpDate').AsString;
txtBelgeNo.Text :=QrySHareket.FieldName('OpNumber').AsString;
txtAciklama.Text :=QrySHareket.FieldName('OpDefinition').AsString;
txtMiktar.Value :=QrySHareket.FieldName('InValue').AsFloat+
    QrySHareket.FieldName('OutValue').AsFloat;
end;
// ***
procedure TFrmStokHareketListe.BtnAraClick(Sender: TObject);
begin

```

```

FrmDlgStockSearch:=TFrmDlgStockSearch.Create(Self);
FrmDlgStockSearch.ShowModal;
if FrmDlgStockSearch.iptal =False then
begin
    txtStokKod.Text:=FrmDlgStockSearch.actKod;
    txtUrunAdi.Text:=FrmDlgStockSearch.actAdi;
    txtStokMiktar.Value:=FrmDlgStockSearch.actMiktar;
    txtStokMiktar.DisplayFormat:='0.000 ' + FrmDlgStockSearch.actBirim;
end;
FrmDlgStockSearch.Release;
end;
// ***
procedure TFrmStokHareketListe.BtnIptalClick(Sender: TObject);
begin
    Islem(False,'---');
end;

// ***
procedure TFrmStokHareketListe.BtnUygulaClick(Sender: TObject);
begin
    if BtnUygula.Caption='Kaydet' then SHKaydet;
    if BtnUygula.Caption='Güncelle' then SHDegistir;
    if BtnUygula.Caption='Sil' then SHSil;
end;
// ***
procedure TFrmStokHareketListe.BtnListeleClick(Sender: TObject);
begin
    if txtStokKod.Text="" then
    begin
        Hata('Lütfen Stok Tanımlayınız. ');
        Exit;
    end;
    SetGridQuery;
end;
// ***
procedure TFrmStokHareketListe.BtnSilClick(Sender: TObject);
begin
    if (QrySHareket.Active=False) or (QrySHareket.RecordCount=0) then exit;
    islem (True,'Sil');
    if QrySHareket.FieldByName('InValue').AsFloat>QrySHareket.FieldByName('OutValue').
AsFloat then
        cmbHareketTur.ItemIndex:=0
    else
        cmbHareketTur.ItemIndex:=1;
    cmbHareketTur.Enabled:=False;
    DTIslemTarihi.Text:=QrySHareket.FieldByName('OpDate').AsString;
    txtBelgeNo.Text :=QrySHareket.FieldByName('OpNumber').AsString;
    txtAciklama.Text :=QrySHareket.FieldByName('OpDefinition').AsString;
    txtMiktar.Value :=QrySHareket.FieldByName('InValue').AsFloat+
        QrySHareket.FieldByName('OutValue').AsFloat;
end;
// ***
procedure TFrmStokHareketListe.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin

```

```

if pnlHeader.Enabled=False then exit;
if Key=Vk_F2 then BtnListele.Click;
if Key=Vk_F3 then BtnAra.Click;
if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
if (key = vk_F5) and (BtnDegistir.Visible=True) then BtnDegistir.Click;
if (key = vk_F6) and (BtnSil.Visible=True) then BtnSil.Click;
if (key = vk_F8) and (BtnYazdir.Visible=True) then BtnYazdir.Click;
if Key=Vk_F10 then BtnKapat.Click;
end;
// ***
procedure TFrmStokHareketListe.pnlHeaderMouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ReleaseCapture;
  self.perform(WM_SYSCOMMAND, $F012, 0);
end;
// ***
procedure TFrmStokHareketListe.BtnYazdirClick(Sender: TObject);
begin
  PrintPreviewLink1.ReportTitleText:=Label16.Caption;
  PrintPreviewLink1.PrinterPage.Orientation:=poPortrait;
  PrintPreviewLink1.PrinterPage.PageHeader.RightTitle .Clear;
  PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi:' +
  FormatDateTime('dd.mm.yyyy',date));
  PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Clear;
  PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
  PrintPreview.Preview(True,PrintPreviewLink1);
end;

end.

*****

unit UntStokKartListe;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, ComCtrls, ToolWin, ExtCtrls, StdCtrls, XPMAN, DB, ADODB, OleCtrls,
  AdvToolBar, AdvToolBarStylers, cxStyles, cxCustomData, cxGraphics,
  cxFilter, cxData, cxDataStorage, cxEdit, cxDBData, cxGridLevel,
  cxClasses, cxControls, cxGridCustomView, cxGridCustomTableView,
  cxGridTableView, cxGridDBTableView, cxGrid, Buttons, Mask, ToolEdit,
  CurrEdit, dxPSGlbl, dxPSUtl, dxPSEngn, dxPrnPg, dxBkgnd, dxWrap,
  dxPrnDev, dxPSCmpsProvider, dxPSFillPatterns, dxPSEdgePatterns,
  dxPSCore, dxPScxCommon, dxPScxGridLnk;

type
  TFrmStokKartListe = class(TForm)
    XPMANifest1: TXPMANifest;
    AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
    DSQryStock: TDataSource;
    pnlHeader: TPanel;
    Image9: TImage;
    Panel3: TPanel;
  end;

```

```

Image1: TImage;
Label16: TLabel;
Label17: TLabel;
AdvDockPanel1: TAdvDockPanel;
AdvToolBar1: TAdvToolBar;
BtnKapat: TAdvToolBarButton;
BtnSil: TAdvToolBarButton;
BtnDegistir: TAdvToolBarButton;
BtnEkle: TAdvToolBarButton;
BtnYazdir: TAdvToolBarButton;
QryStock: TADOQuery;
cxGrid1DBTableView1: TcxGridDBTableView;
cxGrid1Level1: TcxGridLevel;
cxGrid1: TcxGrid;
pnlIslem: TPanel;
cxGrid1DBTableView1Kod: TcxGridDBCColumn;
cxGrid1DBTableView1Adi: TcxGridDBCColumn;
cxGrid1DBTableView1Birim: TcxGridDBCColumn;
cxGrid1DBTableView1Miktar: TcxGridDBCColumn;
StatusBar1: TStatusBar;
Label9: TLabel;
txtKod: TEdit;
Label1: TLabel;
txtAdi: TEdit;
Label26: TLabel;
cmbBirim: TComboBox;
PnlCaption: TPanel;
Shape1: TShape;
lblCaption: TLabel;
Image2: TImage;
BtnUygula: TSpeedButton;
BtnIptal: TSpeedButton;
Label2: TLabel;
txtMiktar: TCurrencyEdit;
PrintPreview: TdxComponentPrinter;
PrintPreviewLink1: TdxGridReportLink;
procedure BtnCloseClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure FormShow(Sender: TObject);
procedure BtnEkleClick(Sender: TObject);
procedure BtnDegistirClick(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure BtnSilClick(Sender: TObject);
procedure BtnUygulaClick(Sender: TObject);
procedure BtnIptalClick(Sender: TObject);
procedure pnlHeaderMouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure BtnYazdirClick(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
  procedure Islem(Durum:Boolean;BtnBaslik,lblBaslik:String);
  procedure SetGridQuery();
  procedure GetParams();

```



```

        procedure StokKaydet();
        procedure StokDegistir();
        procedure StokSil();
    end;

var
    FrmStokKartListe: TFrmStokKartListe;

implementation

uses  UntDm, UntGlobal,UntUyari, UntOnay;

{$R *.DFM}
// ***
procedure TFrmStokKartListe.GetParams();
var
    i:Integer;
begin
    with DM.QryArama do
    begin
        // *** Ölçü Birimi
        Close;
        SQL.Clear;
        SQL.Add('Select * from Params Where PType="02" order by ID');
        Prepared ;
        open;
        for i :=1 to recordcount do
        begin
            cmbBirim.Items.Add(FieldValues['PCode']);
        end;
        next;
    end;
    cmbBirim.ItemIndex:=0;
end;
// ***
procedure TFrmStokKartListe.SetGridQuery();
begin
    with QryStock do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * From Stok order by kod Asc');
        Prepared ;
        open;
        (FieldByName('Miktar')as TNumericField ).DisplayFormat:='#,##0.000';
    end;
end;
// ***
procedure TFrmStokKartListe.BtnCloseClick(Sender: TObject);
begin
    Self.close;
end;
// ***
procedure TFrmStokKartListe.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin

```

```

if pnlHeader.Enabled=False then exit;
if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
if (key = vk_F5) and (BtnDegistir.Visible=True) then BtnDegistir.Click;
if (key = vk_F6) and (BtnSil.Visible=True) then BtnSil.Click;
if (key = vk_F8) and (BtnYazdir.Visible=True) then BtnYazdir.Click;
if Key=Vk_F10 then BtnKapat.Click;
end;
// ***
procedure TFrmStokKartListe.FormShow(Sender: TObject);
var
region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
    SetWindowRgn(handle, region, true);
    KullaniciHaklari(Self.Tag,AdvToolBar1);
    GetParams;
    SetGridQuery;
end;
// ***
procedure TFrmStokKartListe.BtnEkleClick(Sender: TObject);
begin
    Islem(True,'Kaydet','Yeni Stok Ekle ');
    txtKod.Text :=GetLastNumber(0);
    txtAdi.Text :='';
    txtMiktar.Value :=0;
    cmbBirim.ItemIndex:=0;
    txtAdi.SetFocus;
end;
// ***
procedure TFrmStokKartListe.BtnDegistirClick(Sender: TObject);
begin
    if (QryStock.Active=False) or (QryStock.RecordCount=0) then exit;
    Islem(True,'Güncelle','Stok Bilgilerini Değiştirme ');
    with QryStock do
    begin
        txtKod.Text :=FieldByName('Kod').AsString;
        txtAdi.Text :=FieldByName('Adi').AsString;
        txtMiktar.Text:=FieldByName('Miktar').AsString;
        cmbBirim.ItemIndex:=cmbBirim.Items.IndexOf(FieldByName('Birim').AsString);
    end;
end;
// ***
procedure TFrmStokKartListe.BtnKapatClick(Sender: TObject);
begin
    self.Close;
end;
// ***
procedure TFrmStokKartListe.BtnSilClick(Sender: TObject);
begin
    if (QryStock.Active=False) or (QryStock.RecordCount=0) then exit;
    Islem(True,'Sil','Stok Bilgilerini Sil ');
    with QryStock do
    begin
        txtKod.Text :=FieldByName('Kod').AsString;

```

```

txtAdi.Text :=FieldByName('Adi').AsString;
txtMiktar.Text:=FieldByName('Miktar').AsString;
cmbBirim.ItemIndex:=cmbBirim.Items.IndexOf(FieldByName('Birim').AsString);
end;
end;
// ***
procedure TFrmStokKartListe.StokKaydet;
begin
// *** Kontroller.....
if Trim(txtAdi.Text) = " " Then
begin
FrmUyari:=TFrmUyari.Create(Self);
FrmUyari.IblCaption.Caption:='Lütfen Ürün Tanımını Yapınız.';
FrmUyari.ShowModal;
FrmUyari.Release;
txtAdi.SetFocus;
Exit;
end;
// *** kaydet
with DM.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Insert Into Stok (Kod,Adi,Birim,Miktar) ');
SQL.Add('Values (:Kod,:Adi,:Birim,0) ');
Parameters.ParamByName('Kod').Value :=txtKod.Text ;
Parameters.ParamByName('Adi').Value :=txtAdi.Text;
Parameters.ParamByName('Birim').Value :=cmbBirim.Text;
ExecSQL;
end;
UpdateLastNumber(0, txtKod.Text);
CompTemizle(self,1);
CompTemizle(self,3);
CompTemizle(self,5);
txtKod.Text :=GetLastNumber(0);
SetGridQuery;
txtAdi.SetFocus;
end;
// ***
procedure TFrmStokKartListe.BtnUygulaClick(Sender: TObject);
begin
if BtnUygula.Caption='Kaydet' then StokKaydet;
if BtnUygula.Caption='Güncelle' then StokDegistir;
if BtnUygula.Caption='Sil' then StokSil;
end;
// ***
procedure TFrmStokKartListe.Islem(Durum: Boolean; BtnBaslik,lblBaslik: String);
begin
pnlHeader.Enabled :=Not Durum;
cxGrid1.Enabled :=Not Durum;
pnlIslem.Visible :=Durum;
BtnUygula.Caption :=BtnBaslik;
lblCaption.Caption:=lblBaslik;
end;
// ***
procedure TFrmStokKartListe.BtnIptalClick(Sender: TObject);

```

```

begin
    Islem(False,'---','---');
    SetGridQuery;
end;
// ***
procedure TFrmStokKartListe.StokDegistir;
begin
    if Trim(txtAdi.Text) = '' Then
    begin
        FrmUyari:=TFrmUyari.Create(Self);
        FrmUyari.lblCaption.Caption:='Lütfen Ürün Tanımını Yapınız.';
        FrmUyari.ShowModal;
        FrmUyari.Release;
        txtAdi.SetFocus;
        Exit;
    end;
    with DM.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * from Stok Where Kod=:Bul');
        Parameters.ParamByName('Bul').Value:=txtKod.Text;
        open;
        Edit;
        FieldByName('Adi').AsString :=txtAdi.Text;
        FieldByName('Birim').AsString :=cmbBirim.Text;
        post;
    end;
    BtnIptal.Click;
end;

// ***
procedure TFrmStokKartListe.StokSil;
begin
    FrmOnay:=TFrmOnay.Create(Self);
    FrmOnay.lblCaption.Caption:='Stok Kartını Silmek mi İstiyormusunuz?';
    FrmOnay.ShowModal;
    if StrToInt(txtKod.Text)<6 then
    begin
        Hata('Otomasyon için kullanılan stoklar silinemez. ');
        exit;
    end;
    if FrmOnay.ActOnay=False then
    begin
        FrmOnay.Release;
        Exit;
    end;
    FrmOnay.Release;
    with DM.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * from StokExtre2005 Where Kod="'+txtKod.Text +"' ");
        Prepared ;
        Open;
    end;
end;

```

```

if RecordCount > 0 then
begin
    hata('Kartın Hareketi Olduğu için Silinemez.');
```

exit;

```
end
else
begin
    Close;
    SQL.Clear;
    SQL.Add('Delete * from Stok Where Kod="'+txtKod.Text +"' ');
    ExecSQL;
end;
end;
BtnIptal.Click;
end;
// ***
procedure TFrmStokKartListe.pnlHeaderMouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
// ***
procedure TFrmStokKartListe.BtnYazdirClick(Sender: TObject);
begin
    PrintPreviewLink1.ReportTitleText:=Label16.Caption;
    PrintPreviewLink1.PrinterPage.Orientation:=poLandscape;
    PrintPreviewLink1.PrinterPage.PageHeader.RightTitle .Clear;
    PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi:'+
    FormatDateTime('dd.mm.yyyy',date));
    PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Clear;
    PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
    PrintPreview.Preview(True,PrintPreviewLink1);
end;
end.

*****

unit UntUretimListe;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, Mask, ToolEdit, Buttons, ExtCtrls, ComCtrls, DB, ADODB, Menus, AdvToolBar,
    AdvToolBarStylers, AdvMenus, ImgList, cxStyles, cxCustomData, cxGraphics,
    cxFilter, cxData, cxDataStorage, cxEdit, cxDBData, cxGridCustomTableView,
    cxGridTableView, cxGridBandedTableView, cxGridDBBandedTableView,
    cxControls, cxGridCustomView, cxClasses, cxGridLevel, cxGrid;

type
    TFrmUretimListe = class(TForm)
        Panel2: TPanel;
        DTStart: TDateEdit;
        DTEnd: TDateEdit;
        CheckBox1: TCheckBox;
```

```

DSUretimQuery: TDataSource;
UretimQuery: TADOQuery;
AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
AdvPopupMenu1: TAdvPopupMenu;
mnuTumu: TMenuItem;
mnuBaslamamis: TMenuItem;
mnuDevamEden: TMenuItem;
mnuBitmis: TMenuItem;
ImageList1: TImageList;
Panel4: TPanel;
Image9: TImage;
lblCaption: TLabel;
Panel1: TPanel;
Image1: TImage;
AdvDockPanel1: TAdvDockPanel;
AdvToolBar1: TAdvToolBar;
BtnKapat: TAdvToolBarButton;
BtnEkle: TAdvToolBarButton;
BtnYazdir: TAdvToolBarButton;
BtnKriter: TAdvToolBarButton;
Label16: TLabel;
Label17: TLabel;
cxGrid1: TcxGrid;
cxGrid1Level1: TcxGridLevel;
cxGrid1DBBandedTableView1: TcxGridDBBandedTableView;
cxGrid1DBBandedTableView1TPlan: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1UretimNo: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1UrunKod: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1UrunAdi: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1ReceteKod: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1ReceteAdi: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1RStdMiktar: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1PTekrar: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1GMiktar: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1GTekrar: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1STarih: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1SSaat: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1ETarih: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1ESaat: TcxGridDBBandedColumn;
StatusBar1: TStatusBar;
procedure BtnKapatClick(Sender: TObject);
procedure CheckBox1Click(Sender: TObject);
procedure mnuTumuClick(Sender: TObject);
procedure mnuBaslamamisClick(Sender: TObject);
procedure mnuDevamEdenClick(Sender: TObject);
procedure mnuBitmisClick(Sender: TObject);
procedure BtnEkleClick(Sender: TObject);
procedure BtnKriterClick(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure Image9MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure BtnYazdirClick(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
private

```

```

    { Private declarations }
public
    { Public declarations }
    procedure SetRecordView(ActView:integer);
end;

var
    FrmUretimListe: TFrmUretimListe;
    ListeTuru:Integer;

implementation

uses UntGlobal, UntUretimPopUp, UntDm, UntQRUretimListesi01;

{$R *.dfm}

procedure TFrmUretimListe.SetRecordView(ActView:integer);
var
    tmpSTR:String;
begin
    case ActView of
    0 : tmpSTR:='Where Durum<>-1 ';
    1 : tmpSTR:='Where Durum=0 ';
    2 : tmpSTR:='Where Durum=1 ';
    3 : tmpSTR:='Where Durum=2 ';
    end;
    with UretimQuery do
    begin
        Close;
        SQL.Clear ;
        SQL.Add('Select *,(RStdMiktar*PTekrar) as TPlan From Uretim ');
        SQL.Add(tmpStr);
        if CheckBox1.Checked then
            SQL.Add(' and STarih Between "' + DTStart.Text + '" and "' + DTEnd.Text + '" ');
        SQL.Add(' Order By STarih Asc ');
        Prepared;
        open;
        (FieldByName('RStdMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('GMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
        (FieldByName('TPlan') as TNumericField).DisplayFormat:='0.000 Kg';
    end;
    BtnKriter.Caption:='Kriter : ' +AdvPopupMenu1.Items[ActView].Caption ;
    BtnKriter.Glyph:=nil;
    ImageList1.GetBitmap(ActView,BtnKriter.Glyph);
    Self.Repaint;
end;
// ***
procedure TFrmUretimListe.BtnKapatClick(Sender: TObject);
begin
    Self.Close;
end;
// ***
procedure TFrmUretimListe.CheckBox1Click(Sender: TObject);
begin
    DTStart.Visible:=CheckBox1.Checked;
    DTEnd.Visible:=CheckBox1.Checked;
end;

```

```

end;
// ***
procedure TFrmUretimListe.mnuTumuClick(Sender: TObject);
begin
    SetRecordView(0);
    ListeTuru:=0;
end;
// ***
procedure TFrmUretimListe.mnuBaslamamisClick(Sender: TObject);
begin
    SetRecordView(1);
    ListeTuru:=1;
end;
// ***
procedure TFrmUretimListe.mnuDevamEdenClick(Sender: TObject);
begin
    SetRecordView(2);
    ListeTuru:=2;
end;
// ***
procedure TFrmUretimListe.mnuBitmisClick(Sender: TObject);
begin
    SetRecordView(3);
    ListeTuru:=3;
end;
// ***
procedure TFrmUretimListe.BtnEkleClick(Sender: TObject);
begin
    if (UretimQuery.Active=False) or (UretimQuery.RecordCount=0) then exit;
    if ListeTuru<>1 then
        begin
            Hata('Üretimi Başlatmak İçin Üretime Girmemiş'+#13+
                'Emirler Listesinden Seçim Yapılabilir.' );
            exit;
        end;
    FrmUretimPopUp:=TFrmUretimPopUp.Create(Self);
    FrmUretimPopUp.actUretimNo:=UretimQuery.fieldByName('UretimNo').AsString;
    FrmUretimPopUp.ShowModal;
    FrmUretimPopUp.Release;
end;
// ***
procedure TFrmUretimListe.BtnKriterClick(Sender: TObject);
begin
    SetRecordView(ListeTuru);
end;
// ***
procedure TFrmUretimListe.FormShow(Sender: TObject);
var
    region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
    SetWindowRgn(handle, region, true);
    KullaniciHaklari(Self.Tag,AdvToolBar1);
    DTStart.Date:=date;

```



```

    DTEnd.Date:=date;
    UretimQuery.Close;
end;
// ***
procedure TFrmUretimListe.Image9MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
// ***
procedure TFrmUretimListe.BtnYazdirClick(Sender: TObject);
begin
    QRUretimListesi01:=TQRUretimListesi01.Create(self);
    QRUretimListesi01.QRActView:=ListeTuru;
    QRUretimListesi01.PreviewModal;
    QRUretimListesi01.Free;
end;
// ****
procedure TFrmUretimListe.FormClose(Sender: TObject;
    var Action: TCloseAction);
begin
    UretimQuery.Close;
end;
// ***
procedure TFrmUretimListe.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
    if (key = vk_F8) and (BtnYazdir.Visible=True) then BtnYazdir.Click;
    if Key=Vk_F10 then BtnKapat.Click;
end;
end.

*****

unit UntUretimPlanlama;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, ExtCtrls, StdCtrls, Buttons, ImgList, ComCtrls, Mask, ToolEdit,
    AdvToolBar, AdvToolBarStylers, Menus, AdvMenus, cxStyles, cxCustomData,
    cxGraphics, cxFilter, cxData, cxDataStorage, cxEdit, cxDBData,
    cxGridLevel, cxGridCustomTableView, cxGridTableView,
    cxGridBandedTableView, cxGridDBBandedTableView, cxClasses, cxControls,
    cxGridCustomView, cxGrid, DB, ADODB;

type
    TFrmUretimPlanlama = class(TForm)
        UretimQuery: TADOQuery;
        DSUretimQuery: TDataSource;
        AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
        Panel2: TPanel;
        DTStart: TDateEdit;

```

```

DTEnd: TDateEdit;
CheckBox1: TCheckBox;
AdvPopupMenu1: TAdvPopupMenu;
mnuTumu: TMenuItem;
mnuBasla: TMenuItem;
mnuDevamEden: TMenuItem;
mnuBitmis: TMenuItem;
ImageList1: TImageList;
Panel4: TPanel;
Image9: TImage;
Label1: TLabel;
Panel3: TPanel;
Image2: TImage;
Label16: TLabel;
Label17: TLabel;
AdvDockPanel1: TAdvDockPanel;
AdvToolBar1: TAdvToolBar;
BtnKapat: TAdvToolBarButton;
BtnSil: TAdvToolBarButton;
BtnDegistir: TAdvToolBarButton;
BtnEkle: TAdvToolBarButton;
BtnYazdir: TAdvToolBarButton;
BtnKriter: TAdvToolBarButton;
cxGrid1: TcxGrid;
cxGrid1DBBandedTableView1: TcxGridDBBandedTableView;
cxGrid1DBBandedTableView1UretimNo: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1UrunKod: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1UrunAdi: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1ReceteKod: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1ReceteAdi: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1RStdMiktar: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1PTekrar: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1TPlan: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1GMiktar: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1GTekrar: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1STarih: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1SSaat: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1ETarih: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1ESaat: TcxGridDBBandedColumn;
cxGrid1Level1: TcxGridLevel;
StatusBar1: TStatusBar;
procedure FormCreate(Sender: TObject);
procedure btnKapatClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure FormActivate(Sender: TObject);
procedure BtnEkleClick(Sender: TObject);
procedure BtnDegistirClick(Sender: TObject);
procedure BtnSilClick(Sender: TObject);
procedure mnuTumuClick(Sender: TObject);
procedure mnuBaslaClick(Sender: TObject);
procedure mnuDevamEdenClick(Sender: TObject);
procedure mnuBitmisClick(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure BtnKriterClick(Sender: TObject);
procedure CheckBox1Click(Sender: TObject);

```

```

    procedure Image9MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure BtnYazdirClick(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
  ListeTuru:Integer;
  procedure SetRecordView(ActView:integer);

end;

var
  FrmUretimPlanlama: TFrmUretimPlanlama;

implementation

uses DateUtils, UntEmirEkle, UntDm, UntGlobal, UntEmirDegistir, UntEmirSil,
  UntQRUretimListesi02;

{$R *.dfm}

procedure TFrmUretimPlanlama.SetRecordView(ActView:integer);
var
  tmpSTR:string;
begin
  case ActView of
    0 : tmpSTR:='Where Durum<>-1 ' ;
    1 : tmpSTR:='Where Durum=0 ' ;
    2 : tmpSTR:='Where Durum=1 ' ;
    3 : tmpSTR:='Where Durum=2 ' ;
  end;
  with UretimQuery do
  begin
    Close;
    SQL.Clear ;
    SQL.Add('Select *,(RStdMiktar*PTekrar) as TPlan From Uretim ');
    SQL.Add(tmpStr);
    if CheckBox1.Checked then
      SQL.Add(' and STarih Between "' + DTStart.Text + '" and "' + DTEnd.text + '" ');
    SQL.Add(' Order By STarih Asc ');
    Prepared;
    open;
    (FieldByName('RStdMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('GMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('TPlan') as TNumericField).DisplayFormat:='0.000 Kg';
  end;
  BtnKriter.Caption:='Kriter : ' +AdvPopupMenu1.Items[ActView].Caption ;
  BtnKriter.Glyph:=nil;
  ImageList1.GetBitmap(ActView,BtnKriter.Glyph);
end;
// ***

procedure TFrmUretimPlanlama.btnKapatClick(Sender: TObject);
begin
  Self.Close;
end;

```

```

// ***
procedure TFrmUretimPlanlama.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
  if (key = vk_F5) and (BtnDegistir.Visible=True) then BtnDegistir.Click;
  if (key = vk_F6) and (BtnSil.Visible=True) then BtnSil.Click;
  if (key = vk_F8) and (BtnYazdir.Visible=True) then BtnYazdir.Click;
  if key = vk_F10 then btnKapat.Click;
end;
// ***
procedure TFrmUretimPlanlama.FormActivate(Sender: TObject);
begin
  DTStart.Date:=Date;
  DTEnd.Date :=Date+5;
end;
// ***
procedure TFrmUretimPlanlama.BtnEkleClick(Sender: TObject);
begin
  FrmEmirEkle:=TFrmEmirEkle.Create(Self);
  FrmEmirEkle.ShowModal;
  if FrmEmirEkle.iptal=False then BtnKriter.Click;
  FrmEmirEkle.Release;
end;
// ****
procedure TFrmUretimPlanlama.BtnDegistirClick(Sender: TObject);
begin
  if (UretimQuery.Active=False) or (UretimQuery.RecordCount=0) then Exit;
  if ListeTuru=2 then
  begin
    hata('Üretim Prosesi Devam Ettiği İçin Değişiklik İşlemi Yapılamaz. ');
    exit;
  end;
  if ListeTuru=3 then
  begin
    hata('Üretim Prosesi Bitiği İçin Değişiklik İşlemi Yapılamaz. ');
    exit;
  end;
  FrmEmirDegistir:=TFrmEmirDegistir.Create(Self);
  FrmEmirDegistir.ActUretimNo:=UretimQuery.FieldByName('UretimNo').AsString;
  FrmEmirDegistir.ShowModal;
  if FrmEmirDegistir.iptal=False then BtnKriter.Click;
  FrmEmirDegistir.Release;
end;
// ***
procedure TFrmUretimPlanlama.BtnSilClick(Sender: TObject);
begin
  if (UretimQuery.Active=False) or (UretimQuery.RecordCount=0) then Exit;
  if ListeTuru=2 then
  begin
    hata('Üretim Prosesi Devam Ettiği İçin Silme İşlemi Yapılamaz. ');
    exit;
  end;
  if ListeTuru=3 then
  begin
    hata('Üretim Prosesi Bitiği İçin Silme İşlemi Yapılamaz. ');
  end;
end;

```

```

    exit;
end;
FrmEmirSil:=TFrmEmirSil.Create(Self);
FrmEmirSil.ActUretimNo:=UretimQuery.FieldByName('UretimNo').AsString;
FrmEmirSil.ShowModal;
if FrmEmirSil.iptal=False then BtnKriter.Click;
FrmEmirSil.Release;
end;
// ***
procedure TFrmUretimPlanlama.mnuTumuClick(Sender: TObject);
begin
    SetRecordView(0);
    ListeTuru:=0;
end;
// ***
procedure TFrmUretimPlanlama.mnuBaslaClick(Sender: TObject);
begin
    SetRecordView(1);
    ListeTuru:=1;
end;
// ***
procedure TFrmUretimPlanlama.mnuDevamEdenClick(Sender: TObject);
begin
    SetRecordView(2);
    ListeTuru:=2;
end;
// ***
procedure TFrmUretimPlanlama.mnuBitmisClick(Sender: TObject);
begin
    SetRecordView(3);
    ListeTuru:=3;
end;
// ***
procedure TFrmUretimPlanlama.FormShow(Sender: TObject);
var
    region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
    SetWindowRgn(handle, region, true);
    UretimQuery.Close;
    KullaniciHaklari(Self.Tag,AdvToolBar1);
end;
// ***
procedure TFrmUretimPlanlama.BtnKriterClick(Sender: TObject);
begin
    SetRecordView(ListeTuru);
end;
// ***
procedure TFrmUretimPlanlama.CheckBox1Click(Sender: TObject);
begin
    DTStart.Visible:=CheckBox1.Checked;
    DTEnd.Visible:=CheckBox1.Checked;
end;
// ***

```

```

procedure TFrmUretimPlanlama.Image9MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ReleaseCapture;
  self.perform(WM_SYSCOMMAND, $F012, 0);
end;
// ***
procedure TFrmUretimPlanlama.BtnYazdirClick(Sender: TObject);
begin
  QRUretimListesi02:=TQRUretimListesi02.Create(self);
  QRUretimListesi02.QRActView:=ListeTuru;
  QRUretimListesi02.PreviewModal;
  QRUretimListesi02.Free;
end;
end.

```

\*\*\*\*\*

```

unit UntUretimPopUp;

interface

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

type
  TFrmUretimPopUp = class(TForm)
    PnlTools: TPanel;
    BtnBaslat: TSpeedButton;
    BtnKapat: TSpeedButton;
    Panel1: TPanel;
    PageControl2: TPageControl;
    TabSheet3: TTabSheet;
    Label3: TLabel;
    Label10: TLabel;
    Label12: TLabel;
    Label7: TLabel;
    txtSpektNo: TEdit;
    txtUrunKod: TEdit;
    txtUrunAd: TEdit;
    txtUrunTip: TEdit;
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    Label6: TLabel;
    Label8: TLabel;
    DTBaslaSaat: TDateTimePicker;
    DTBaslaTarih: TDateEdit;
    Panel4: TPanel;
    Shape1: TShape;
    Label1: TLabel;
    Image1: TImage;
    Label4: TLabel;
    Label2: TLabel;
    lblUretimNo: TLabel;

```

```

txtReceteKod: TEdit;
txtReceteAdi: TEdit;
StatusBar1: TStatusBar;
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtnKapatClick(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure BtnBaslatClick(Sender: TObject);
procedure Panel4MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
private
  { Private declarations }
public
  { Public declarations }
  actUretimNo:String;
end;

var
  FrmUretimPopUp: TFrmUretimPopUp;

implementation

uses UntGlobal, UntDm, DB, ADODB, UntMainForm;

{$R *.dfm}

procedure TFrmUretimPopUp.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key=vk_F4 then BtnBaslat.Click;
  if Key=vk_Escape then BtnKapat.Click;
end;
// ***
procedure TFrmUretimPopUp.BtnKapatClick(Sender: TObject);
begin
  self.Close;
end;
// ***
procedure TFrmUretimPopUp.FormCreate(Sender: TObject);
begin
  Caption:=ProCaption;
  DTBaslaTarih.Date:=date;
  DTBaslaSaat.Time:=Now;
end;
// ***
procedure TFrmUretimPopUp.FormShow(Sender: TObject);
var
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
  SetWindowRgn(handle, region, true);
  with DM.QryList do
  begin

```

```

Close;
SQL.Clear;
SQL.Add('Select a.*,b.UrunTip,b.SpektNo From Uretim as a,Urunler as b ');
SQL.Add(' Where (a.UrunKod=b.UrunKod) and (a.UretimNo=:ActUretimNo)');
Parameters.ParamByName('ActUretimNo').Value:=actUretimNo;
Prepared ;
Open;
lblUretimNo.caption:=FieldByName('UretimNo').AsString ;
txtUrunKod.Text :=FieldByName('UrunKod').AsString ;
txtUrunAd.Text :=FieldByName('UrunAdi').AsString ;
txtUrunTip.Text :=FieldByName('UrunTip').AsString ;
txtSpektNo.Text :=FieldByName('SpektNo').AsString ;
txtReceteKod.Text :=FieldByName('ReceteKod').AsString ;
txtReceteAdi.Text :=FieldByName('ReceteAdi').AsString ;
Self.Tag :=FieldByName('PTekrar').AsInteger ;
end;
end;
// ***
procedure TFrmUretimPopUp.BtnBaslatClick(Sender: TObject);
begin
if Onay('Üretim Başlatılacaktır.'+#13+'Devam Etmek İstiyor musunuz?')=False then Exit;
with dm.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Update uretim set STarih=:ActSTarih,SSaat=:ActSSaat ,Durum=:ActDurum');
SQL.Add('Where UretimNo="'+ lblUretimNo.Caption +"'");
Parameters.ParamByName('ActSTarih').Value :=DTBaslaTarih.Text;
Parameters.ParamByName('ActSSaat').Value :=FormatDateTime('hh:mm',DTBaslaSaat.Time);
Parameters.ParamByName('ActDurum').Value :=1;
Prepared;
ExecSQL;
end;
with FrmMainForm do
begin
MtxtUrunKod.Text:=txtUrunKod.Text;
MtxtUrunAd.Text :=txtUrunAd.Text ;
MtxtUrunTip.Text:=txtUrunTip.Text;
MtxtSpektNo.Text:=txtSpektNo.Text;
// ***
MtxtReceteKod.Text:=txtReceteKod.Text;
MtxtReceteAdi.Text:=txtReceteAdi.Text;
MtxtBTarih.Text :=DTBaslaTarih.Text;
MtxtBSaat.Text :=FormatDateTime('hh:mm', DTBaslaSaat.time);
pnlUretimNo.Caption:=lblUretimNo.Caption;
MtxtPTekrar.Value:=Self.Tag;
Baslat;
end;
self.Close;
end;
// ***
procedure TFrmUretimPopUp.Panel4MouseDown(Sender: TObject;
Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
ReleaseCapture;
self.perform(WM_SYSCOMMAND, $F012, 0);

```



end;

end.

\*\*\*\*\*

unit UntUretimRaporlar;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  
Dialogs, ExtCtrls, StdCtrls, ComCtrls, Buttons, Mask, ToolEdit, CheckLst,  
cxStyles, cxCustomData, cxGraphics, cxFilter, cxData, cxDataStorage,  
cxEdit, DB, cxDBData, ADODB, cxGridLevel, cxGridCustomTableView,  
cxGridTableView, cxGridBandedTableView, cxGridDBBandedTableView,  
cxClasses, cxControls, cxGridCustomView, cxGrid, AdvToolBar, Menus,  
AdvMenus, AdvToolBarStylers, ImgList, dxPSGlbl, dxPSUtl, dxPSEngn,  
dxPrnPg, dxBkgnd, dxWrap, dxPrnDev, dxPSCompsProvider, dxPSFillPatterns,  
dxPSEdgePatterns, dxPSCore, dxPScxCommon, dxPScxGridLnk;

type

TFrmUretimRaporlar = class(TForm)  
  Panel1: TPanel;  
  Shape1: TShape;  
  Label2: TLabel;  
  Image1: TImage;  
  cxGrid1: TcxGrid;  
  cxGrid1DBBandedTableView1: TcxGridDBBandedTableView;  
  cxGrid1DBBandedTableView1UretimNo: TcxGridDBBandedColumn;  
  cxGrid1DBBandedTableView1UrunKod: TcxGridDBBandedColumn;  
  cxGrid1DBBandedTableView1UrunAdi: TcxGridDBBandedColumn;  
  cxGrid1Level1: TcxGridLevel;  
  DSUretimQuery: TDataSource;  
  UretimQuery: TADOQuery;  
  Panel3: TPanel;  
  AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;  
  AdvPopupMenu1: TAdvPopupMenu;  
  mnuUrtEvet: TMenuItem;  
  mnuUrtHayir: TMenuItem;  
  AdvDockPanel1: TAdvDockPanel;  
  AdvToolBar1: TAdvToolBar;  
  BtnKapat: TAdvToolBarButton;  
  BtnYazdir: TAdvToolBarButton;  
  BtnKriter: TAdvToolBarButton;  
  CheckBox1: TCheckBox;  
  DTStart: TDateEdit;  
  DTEnd: TDateEdit;  
  ImageList1: TImageList;  
  CheckBox2: TCheckBox;  
  Panel2: TPanel;  
  Label1: TLabel;  
  txtUrunKod: TEdit;  
  Label3: TLabel;  
  txtUrunAdi: TEdit;  
  SpeedButton1: TSpeedButton;

```

cxGrid1DBBandedTableView1DToplam: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1Agrega1: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1Agrega2: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1Agrega3: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1Kum: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1Cimento: TcxGridDBBandedColumn;
cxGrid1DBBandedTableView1Su: TcxGridDBBandedColumn;
StatusBar1: TStatusBar;
PrintPreview: TdxComponentPrinter;
PrintPreviewLink1: TdxGridReportLink;
procedure FormCreate(Sender: TObject);
procedure SpeedButton1Click(Sender: TObject);
procedure CheckBox1Click(Sender: TObject);
procedure mnuUrtEvetClick(Sender: TObject);
procedure mnuUrtHayirClick(Sender: TObject);
procedure CheckBox2Click(Sender: TObject);
procedure BtnKapatClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure FormShow(Sender: TObject);
procedure Label2MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure BtnYazdirClick(Sender: TObject);
private
  { Private declarations }
  procedure SetRecordView(ActView:integer);
public
  { Public declarations }
end;

var
  FrmUretimRaporlar: TFrmUretimRaporlar;

implementation

uses UntDlUrunArama, UntGlobal;

{$R *.dfm}

procedure TFrmUretimRaporlar.SetRecordView(ActView:integer);
var
  ActDurum,ActToplam,tmpSTR:String;
begin
  if (CheckBox2.Checked =False) and (txtUrunKod.Text='') then
  begin
    hata('Lütfen Ürün Tanımlayınız. ');
    exit;
  end;
  if ActView=0 then // *** Üretilmiş Ürünler
  begin
    ActToplam:=(U.GTekrar*U.GMiktar) as Toplam, '
    ActDurum:='2';
  end
  else
  begin // *** Üretilmemiş Ürünler
    ActToplam:=(RStdMiktar*PTekrar) as Toplam,';
  end;
end;

```

```

ActDurum:='0';
end;

tmpSTR:='SELECT U.UretimNo, U.UrunKod, U.UrunAdi,'+ActToplam +
        '( R.Agrega01 * U.PTekrar) as Agregal,'+
        '( R.Agrega02 * U.PTekrar) as Agrega2,'+
        '( R.Agrega03 * U.PTekrar) as Agrega3,'+
        '( R.Kum * U.PTekrar) as TKum,'+
        '( R.Cimento * U.PTekrar) as TCimento,'+
        '( R.Su * U.PTekrar) as TSu '+
        'FROM Uretim as U ,Recete as R Where U.ReceteKod = R.ReceteKod and Durum='+ActDurum;
// *** genel
if CheckBox2.Checked =False then
    tmpSTR:=tmpSTR+' and U.UrunKod="'+txtUrunKod.Text+'"';
if CheckBox1.Checked then
    tmpSTR:=tmpSTR+' and STarih Between "'+DTStart.Text+'" and "'+DTEnd.text+'"';
    tmpSTR:=tmpSTR+' Order By STarih Asc';
// for debug ShowMessage(tmpstr);
with UretimQuery do
begin
    Close;
    SQL.Clear ;
    SQL.Add(tmpStr);
    open;
    (FieldByName('Agregal') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('Agrega2') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('Agrega3') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('TKum') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('TCimento') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('TSu') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('Toplam') as TNumericField).DisplayFormat:='0.000 Kg';
end;
BtnKriter.Caption:='Kriter : ' +AdvPopupMenu1.Items[ActView].Caption ;
BtnKriter.Glyph:=nil;
ImageList1.GetBitmap(ActView,BtnKriter.Glyph);
Self.Repaint;
end;

// ***
procedure TFrmUretimRaporlar.FormCreate(Sender: TObject);
begin
    DTStart.Date:=date;
    DTEnd.Date:=Date+7;
end;
// ***
procedure TFrmUretimRaporlar.SpeedButton1Click(Sender: TObject);
begin
    FrmDlgUrunArama:=TFrmDlgUrunArama.Create(Self);
    FrmDlgUrunArama.ShowModal;
    if FrmDlgUrunArama.iptal=False then
    begin
        txtUrunKod.Text:=FrmDlgUrunArama.ActUrunKod;
        txtUrunAdi.Text:=FrmDlgUrunArama.ActUrunAd;
    end;
    FrmDlgUrunArama.Release;

```

```

end;
// ***
procedure TFrmUretimRaporlar.CheckBox1Click(Sender: TObject);
begin
    DTStart.Visible :=CheckBox1.Checked;
    DTEnd.Visible :=CheckBox1.Checked;
    UretimQuery.Close;
end;
// ***
procedure TFrmUretimRaporlar.mnuUrtEvetClick(Sender: TObject);
begin
    SetRecordView(0);
end;
// ***
procedure TFrmUretimRaporlar.mnuUrtHayirClick(Sender: TObject);
begin
    SetRecordView(1);
end;
// ****
procedure TFrmUretimRaporlar.CheckBox2Click(Sender: TObject);
begin
    Panel2.Visible:=not CheckBox2.Checked;
    UretimQuery.Close;
end;
// ***
procedure TFrmUretimRaporlar.BtnKapatClick(Sender: TObject);
begin
    Self.Close;
end;
// ***
procedure TFrmUretimRaporlar.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if (key = vk_F8) and (BtnYazdir.Visible=True) then BtnYazdir.Click;
    if Key=Vk_F10 then BtnKapat.Click;
end;
// ***
procedure TFrmUretimRaporlar.FormShow(Sender: TObject);
var
    region: HRgn;
begin
    StatusBar1.SimplePanel:=True;
    StatusBar1.SimpleText:=ProCaption;
    region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
    KullaniciHaklari(Self.Tag,AdvToolBar1);
    SetWindowRgn(handle, region, true);
end;
// ***
procedure TFrmUretimRaporlar.Label2MouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
// ***
procedure TFrmUretimRaporlar.BtnYazdirClick(Sender: TObject);

```

```

begin
PrintPreviewLink1.ReportTitleText:=Label2.Caption;
PrintPreviewLink1.PrinterPage.Orientation:=poLandscape;
PrintPreviewLink1.PrinterPage.PageHeader.RightTitle .Clear;
PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi:'+'
FormatDateTime('dd.mm.yyyy',date));
PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Clear;
PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
PrintPreview.Preview(True,PrintPreviewLink1);
end;
end.

*****

unit UntUrunerAitReceteler;

interface

uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, cxStyles, cxCustomData, cxGraphics, cxFilter, cxData,
cxDataStorage, cxEdit, DB, cxDBData, dxPSGlbl, dxPSUtl, dxPSEngn,
dxPrnPg, dxBkgn, dxWrap, dxPrnDev, dxPSCompsProvider, dxPSFillPatterns,
dxPSEdgePatterns, dxPSCore, dxPScxCommon, dxPScxGridLnk, ADODB,
AdvToolBar, AdvToolBarStylers, cxGridLevel, cxGridCustomTableView,
cxGridTableView, cxGridDBTableView, cxClasses, cxControls,
cxGridCustomView, cxGrid, ExtCtrls, StdCtrls, ComCtrls;

type
TFrmUrunerAitReceteler = class(TForm)
AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
cxGrid1DBTableView1: TcxGridDBTableView;
cxGrid1Level1: TcxGridLevel;
cxGrid1: TcxGrid;
DSUrunerQuery: TDataSource;
UrunerQuery: TADOQuery;
cxGrid1DBTableView1USpektNo: TcxGridDBCColumn;
cxGrid1DBTableView1UrunKod: TcxGridDBCColumn;
cxGrid1DBTableView1UrunAd: TcxGridDBCColumn;
cxGrid1DBTableView1UrunTip: TcxGridDBCColumn;
cxGrid1DBTableView1ReceteKod: TcxGridDBCColumn;
cxGrid1DBTableView1ReceteAdi: TcxGridDBCColumn;
cxGrid1DBTableView1RecTarih: TcxGridDBCColumn;
cxGrid1DBTableView1Agrega01: TcxGridDBCColumn;
cxGrid1DBTableView1Agrega02: TcxGridDBCColumn;
cxGrid1DBTableView1Agrega03: TcxGridDBCColumn;
cxGrid1DBTableView1Kum: TcxGridDBCColumn;
cxGrid1DBTableView1Cimento: TcxGridDBCColumn;
cxGrid1DBTableView1Su: TcxGridDBCColumn;
cxGrid1DBTableView1Katki01: TcxGridDBCColumn;
cxGrid1DBTableView1Katki02: TcxGridDBCColumn;
cxGrid1DBTableView1ToplamMiktar: TcxGridDBCColumn;
cxGrid1DBTableView1Saat: TcxGridDBCColumn;
cxGrid1DBTableView1Dakika: TcxGridDBCColumn;
cxGrid1DBTableView1DBCColumn1: TcxGridDBCColumn;
cxGrid1DBTableView1DBCColumn2: TcxGridDBCColumn;

```

```

pnlHeader: TPanel;
Image9: TImage;
Label1: TLabel;
Panel3: TPanel;
Image2: TImage;
Label16: TLabel;
Label17: TLabel;
AdvDockPanel1: TAdvDockPanel;
AdvToolBar1: TAdvToolBar;
BtnKapat: TAdvToolBarButton;
BtnYazdir: TAdvToolBarButton;
AdvToolBarButton1: TAdvToolBarButton;
BtnListele: TAdvToolBarButton;
StatusBar1: TStatusBar;
PrintPreview: TdxComponentPrinter;
PrintPreviewLink1: TdxGridReportLink;
procedure BtnKapatClick(Sender: TObject);
procedure BtnYazdirClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
procedure BtnListeleClick(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure pnlHeaderMouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
private
  { Private declarations }
public
  { Public declarations }
end;

var
  FrmUruneAitReceteler: TFrmUruneAitReceteler;

implementation

uses UntDm, UntGlobal;

{$R *.dfm}

procedure TFrmUruneAitReceteler.BtnKapatClick(Sender: TObject);
begin
  Self.Close;
end;
// ***

procedure TFrmUruneAitReceteler.BtnYazdirClick(Sender: TObject);
begin
  PrintPreviewLink1.ReportTitleText:=Label16.Caption;
  PrintPreviewLink1.PrinterPage.Orientation:=poLandscape;
  PrintPreviewLink1.PrinterPage.PageHeader.RightTitle .Clear;
  PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi:' +
  FormatDateTime('dd.mm.yyyy',date));
  PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Clear;
  PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
  PrintPreview.Preview(True,PrintPreviewLink1);
end;
// ***

```

```

procedure TFrmUrunAitReceteler.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key=vk_F2 then BtnListele.Click;
  if Key=vk_F8 then BtnYazdir.Click;
  if Key=vk_F10 then BtnKapat.Click;
end;
// ***

procedure TFrmUrunAitReceteler.BtnListeleClick(Sender: TObject);
begin
  with UrunlerQuery do
  begin
    Close;
    SQL.Clear;
    SQL.Add('Select U.* , R.* , (R.HPerAdi '+' ' '+' R.HPerSoyadi) as Hazirlayan, ');
    SQL.Add('(R.OPerAdi '+' ' '+' R.OPerSoyadi) as Onaylayan From Urunler as U, recete as R ');
    SQL.Add('Where U.SpektNo=R.SpektNo');
    Open;
  end;
end;
// ***

procedure TFrmUrunAitReceteler.FormShow(Sender: TObject);
var
  region: HRgn;
begin
  StatusBar1.SimplePanel:=True;
  StatusBar1.SimpleText:=ProCaption;
  region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
  SetWindowRgn(handle, region, true);
end;
// ***

procedure TFrmUrunAitReceteler.pnlHeaderMouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ReleaseCapture;
  self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntUrunListesi;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, StdCtrls, ExtCtrls, Buttons, ToolWin, ComCtrls,
  Grids, Mask, AdvToolBar, AdvToolBarStylers, AdvMenus, AdvMenuStylers, XPMan, cxStyles,
  cxCustomData, cxGraphics, cxFilter, cxData, cxDataStorage, cxEdit,
  cxDBData, cxGridCustomTableView, cxGridTableView, cxGridDBTableView,
  cxGridLevel, cxClasses, cxControls, cxGridCustomView, cxGrid, DB, ADODB,
  dxPSGlbl, dxPSUtl, dxPSEngn, dxPrnPg, dxBkgnd, dxWrap, dxPrnDev,
  dxPSCompsProvider, dxPSFillPatterns, dxPSEdgePatterns, dxPSCore,
  dxPScxCommon, dxPScxGridLnk;

```

```

type
TFrmUrunListesi = class(TForm)
  DSUrunlerQuery: TDataSource;
  UrunlerQuery: TADOQuery;
  AdvToolBarFantasyStyler1: TAdvToolBarFantasyStyler;
  pnlHeader: TPanel;
  Image9: TImage;
  Label1: TLabel;
  Panel3: TPanel;
  Image2: TImage;
  AdvDockPanel1: TAdvDockPanel;
  AdvToolBar1: TAdvToolBar;
  BtnKapat: TAdvToolBarButton;
  BtnSil: TAdvToolBarButton;
  BtnDegistir: TAdvToolBarButton;
  BtnEkle: TAdvToolBarButton;
  BtnYazdir: TAdvToolBarButton;
  BtnReceteler: TAdvToolBarButton;
  Label16: TLabel;
  Label17: TLabel;
  pnlIslem: TPanel;
  Panel2: TPanel;
  Shape1: TShape;
  lblCaption: TLabel;
  CapImage: TImage;
  Label7: TLabel;
  txtUrunKod: TEdit;
  Label10: TLabel;
  txtUrunAd: TEdit;
  Label12: TLabel;
  txtUrunTip: TEdit;
  Label3: TLabel;
  txtSpektNo: TEdit;
  BtnUygula: TSpeedButton;
  BtnIptal: TSpeedButton;
  StatusBar1: TStatusBar;
  cxGrid1DBTableView1: TcxGridDBTableView;
  cxGrid1Level1: TcxGridLevel;
  cxGrid1: TcxGrid;
  cxGrid1DBTableView1SpektNo: TcxGridDBColumn;
  cxGrid1DBTableView1UrunKod: TcxGridDBColumn;
  cxGrid1DBTableView1UrunAd: TcxGridDBColumn;
  cxGrid1DBTableView1UrunTip: TcxGridDBColumn;
  PrintPreview: TdxComponentPrinter;
  PrintPreviewLink1: TdxGridReportLink;

  procedure FormShow(Sender: TObject);
  procedure BtnKapatClick(Sender: TObject);
  procedure BtnEkleClick(Sender: TObject);
  procedure BtnDegistirClick(Sender: TObject);
  procedure BtnSilClick(Sender: TObject);
  procedure BtnRecetelerClick(Sender: TObject);
  procedure BtnYazdirClick(Sender: TObject);
  procedure FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
  procedure BtnIptalClick(Sender: TObject);

```



```

    procedure BtnUygulaClick(Sender: TObject);
    procedure pnlHeaderMouseDown(Sender: TObject; Button: TMouseButton;
        Shift: TShiftState; X, Y: Integer);
private
    { Private declarations }
public
    { Public declarations }
    procedure UrunKayit();
    procedure UrunDegistir();
    procedure UrunSil();
    procedure SetGridView();
    procedure Islem(Durum: Boolean; BtnBaslik,lblBaslik: String);
end;

var
    FrmUrunListesi: TFrmUrunListesi;

implementation

uses UntDm, UntGlobal,StrUtils, UntUruneAitReceteler;

{$R *.dfm}

procedure TFrmUrunListesi.Islem(Durum: Boolean; BtnBaslik,lblBaslik: String);
begin
    pnlHeader.Enabled :=Not Durum;
    cxGrid1.Enabled :=Not Durum;
    pnlIslem.Visible :=Durum;
    BtnUygula.Caption :=BtnBaslik;
    lblCaption.Caption:=lblBaslik;
end;
// ***
procedure TFrmUrunListesi.SetGridView();
begin
    with UrunlerQuery do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select RecID,SpektNo,UrunKod,UrunAd,UrunTip From Urunler Order By RecID Asc');
        Prepared ;
        Open;
    end;
end;
end;
// ***
procedure TFrmUrunListesi.FormShow(Sender: TObject);
    var
        region: HRgn;
    begin
        StatusBar1.SimplePanel:=True;
        StatusBar1.SimpleText:=ProCaption;
        region:=CreateRoundRectRgn(1, 1,Width ,Height , 20, 20);
        SetWindowRgn(handle, region, true);
        KullaniciHaklari(Self.Tag,AdvToolBar1);
        BtnReceteler.Visible:=BtnYazdir.Visible;
        SetGridView;
    end;

```

```

// ***
procedure TFrmUrunListesi.BtnKapatClick(Sender: TObject);
begin
    Self.Close;
end;
// ***

procedure TFrmUrunListesi.BtnEkleClick(Sender: TObject);
var
    tmp:integer;
begin
    with DM.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select SPLastNumber From LastNumbers');
        Prepared ;
        Open;
        if FieldByName('SPLastNumber').AsInteger=0 then
            txtSpektNo.Text:='SPK00001'
        else
            begin
                tmp:=FieldByName('SPLastNumber').AsInteger+1;
                txtSpektNo.Text:='SPK'+ RightStr('00000'+IntToStr(tmp),5);
            end;
    end;
    Islem(True,'Kaydet','Yeni Ürün Kayıt ');
    txtUrunKod.Text :='';
    txtUrunAd.Text :='';
    txtUrunTip.Text :='';
    txtUrunAd.SetFocus;
end;
// ***

procedure TFrmUrunListesi.BtnDegistirClick(Sender: TObject);
begin
    if (UrunlerQuery.Active=False) or (UrunlerQuery.RecordCount=0) then Exit;
    txtSpektNo.Text :=UrunlerQuery.FieldByName('SpektNo').AsString;
    txtUrunKod.Text :=UrunlerQuery.FieldByName('UrunKod').AsString;
    txtUrunAd.Text :=UrunlerQuery.FieldByName('UrunAd').AsString ;
    txtUrunTip.Text :=UrunlerQuery.FieldByName('UrunTip').AsString;
    Islem(True,'Güncelle','Ürün Bilgilerini Değiştir ');
end;

// ***

procedure TFrmUrunListesi.BtnSilClick(Sender: TObject);
begin
    if (UrunlerQuery.Active=False) or (UrunlerQuery.RecordCount=0) then Exit;
    txtSpektNo.Text :=UrunlerQuery.FieldByName('SpektNo').AsString;
    txtUrunKod.Text :=UrunlerQuery.FieldByName('UrunKod').AsString;
    txtUrunAd.Text :=UrunlerQuery.FieldByName('UrunAd').AsString ;
    txtUrunTip.Text :=UrunlerQuery.FieldByName('UrunTip').AsString;
    Islem(True,'Sil','Ürün Bilgilerini Sil ');
end;
// ***

procedure TFrmUrunListesi.BtnRecetelerClick(Sender: TObject);

```

```

begin
    FrmUrunAitReceteler:=TFrmUrunAitReceteler.Create(self);
    FrmUrunAitReceteler.ShowModal;
    FrmUrunAitReceteler.Release;
end;
// ***

procedure TFrmUrunListesi.BtnYazdirClick(Sender: TObject);
begin
    PrintPreviewLink1.ReportTitleText:=Label16.Caption;
    PrintPreviewLink1.PrinterPage.PageHeader.RightTitle.Add( 'Döküm Tarihi:' +
    FormatDateTime('dd.mm.yyyy',date));
    PrintPreviewLink1.PrinterPage.PageFooter.CenterTitle.Add(ProCaption);
    PrintPreview.PrintTitle:=ProCaption;
    PrintPreview.Preview(True,PrintPreviewLink1);
end;
// ***

procedure TFrmUrunListesi.FormKeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if pnlHeader.Enabled=False then exit;
    if (key = vk_F4) and (BtnEkle.Visible=True) then BtnEkle.Click;
    if (key = vk_F5) and (BtnDegistir.Visible=True) then BtnDegistir.Click;
    if (key = vk_F6) and (BtnSil.Visible=True) then BtnSil.Click;
    if (key = vk_F8) and (BtnYazdir.Visible=True) then BtnYazdir.Click;
    if Key=vk_F9 then BtnReceteler.Click;
    if Key=vk_F10 then BtnKapat.Click;
end;
// ***

procedure TFrmUrunListesi.BtnIptalClick(Sender: TObject);
begin
    Islem(False,'---','---');
    SetGridView;
end;
// ***

procedure TFrmUrunListesi.UrunKayit;
var
    tmp:integer;
begin
    if Length(trim(txtUrunKod.Text))=0 then
    begin
        Hata('Lütfen Ürün Kodunu Giriniz. ');
        txtUrunKod.SetFocus;
        exit;
    end;
    if Length(trim(txtUrunAd.Text))=0 then
    begin
        Hata('Lütfen Ürün Adını Giriniz. ');
        txtUrunAd.SetFocus;
        exit;
    end;
    if Length(trim(txtUrunTip.Text))=0 then
    begin
        Hata('Lütfen Ürün Tipini Giriniz. ');
        txtUrunTip.SetFocus;
        exit;
    end;
end;

```

```

with DM.Qry1 do
begin
  Close;
  SQL.Clear;
  SQL.Add('Select Urunkod from Urunler Where Urunkod="'+Trim(txtUrunKod.Text) + "' ');
  Open;
  if recordcount>0 then
  begin
    Hata('Ürün Kodu Kayıtlıdır.'+#13+ 'Lütfen Kontrol Edin. ');
    txtUrunKod.SetFocus;
    exit;
  end;
  Close;
  SQL.Clear;
  SQL.Add('Select UrunAd from Urunler Where UrunAd="'+Trim(txtUrunAd.Text) + "' ');
  Open;
  if recordcount>0 then
  begin
    Hata('Ürün Adı Kayıtlıdır.'+#13+ 'Lütfen Kontrol Edin. ');
    txtUrunKod.SetFocus;
    exit;
  end;
  Close;
  SQL.Clear;
  SQL.Add('Insert Into Urunler (SpektNo,UrunKod,UrunAd,UrunTip) ');
  SQL.Add('Values(:SpektNo,:UrunKod,:UrunAd,:UrunTip) ');
  Parameters.ParamByName('SpektNo').Value :=txtSpektNo.Text ;
  Parameters.ParamByName('UrunKod').Value :=txtUrunKod.Text ;
  Parameters.ParamByName('UrunAd').Value :=txtUrunAd.Text ;
  Parameters.ParamByName('UrunTip').Value :=txtUrunTip.Text ;
  Prepared ;
  ExecSQL;
  Close;
  SQL.Clear;
  SQL.Add('Update LastNumbers Set SPLastNumber=SPLastNumber+1 ');
  Prepared ;
  ExecSQL;
  CompTemizle(Self,1);
  Close;
  SQL.Clear;
  SQL.Add('Select SPLastNumber From LastNumbers');
  Prepared ;
  Open;
  tmp:=FieldByName('SPLastNumber').AsInteger+1;
  txtSpektNo.Text:='SPK'+ RightStr('00000'+IntToStr(tmp),5);
  end;
  SetGridView;
  txtUrunKod.SetFocus;
end;
// ***
procedure TFrmUrunListesi.BtnUygulaClick(Sender: TObject);
begin
  if BtnUygula.Caption='Kaydet' then UrunKayit;
  if BtnUygula.Caption='Güncelle' then UrunDegistir;
  if BtnUygula.Caption='Sil' then UrunSil;
end;

```

```

// ***
procedure TFrmUrunListesi.UrunDegistir;
begin
if Length(trim(txtUrunKod.Text))=0 then
begin
Hata('Lütfen Ürün Kodunu Giriniz. ');
txtUrunKod.SetFocus;
exit;
end;
if Length(trim(txtUrunAd.Text))=0 then
begin
Hata('Lütfen Ürün Adını Giriniz. ');
txtUrunAd.SetFocus;
exit;
end;
if Length(trim(txtUrunTip.Text))=0 then
begin
Hata('Lütfen Ürün Tipini Giriniz. ');
txtUrunTip.SetFocus;
exit;
end;
with DM.Qry1 do
begin
Close;
SQL.Clear;
SQL.Add('Select Urunkod from Urunler Where UrunKod="'+Trim(txtUrunKod.Text) + " '");
SQL.Add(' and RecID<>'+UrunlerQuery.FieldName('RecID').AsString);
Open;
if recordcount>0 then
begin
Hata('Ürün Kodu Kayıtlıdır.'+#13+ 'Lütfen Kontrol Edin. ');
txtUrunKod.SetFocus;
exit;
end;
Close;
SQL.Clear;
SQL.Add('Select UrunAd from Urunler Where UrunAd="'+Trim(txtUrunAd.Text) + " '");
SQL.Add(' and RecID<>'+UrunlerQuery.FieldName('RecID').AsString);
Open;
if recordcount>0 then
begin
Hata('Ürün Adı Kayıtlıdır.'+#13+ 'Lütfen Kontrol Edin. ');
txtUrunKod.SetFocus;
exit;
end;
Close;
SQL.Clear;
SQL.Add('Select * From Urunler Where RecID='+UrunlerQuery.FieldName('RecID').AsString);
Prepared ;
Open;
Edit;
FieldName('UrunKod').AsString :=txtUrunKod.Text ;
FieldName('UrunAd').AsString :=txtUrunAd.Text ;
FieldName('UrunTip').AsString :=txtUrunTip.Text ;
Post;
end;

```

```

    BtnIptal.Click;
end;
// ***
procedure TFrmUrunListesi.UrunSil;
begin
    if Onay('Ürün Silinecektir.'+#13+'Devam Etmek İstiyor musunuz?')= false then exit;
    with DM.Qry1 do
    begin
        Close;
        SQL.Clear;
        SQL.Add('Select * From Recete Where SpektNo=:bul');
        Parameters.ParamByName('bul').Value:=txtSpektNo.Text;
        Prepared;
        Open;
        if RecordCount>0 then
        begin
            Hata('Ürüne Ait Reçete Tanımlanmış Olduğundan Ürün Silinemez.');
```

exit;

```
        end;
        Close;
        SQL.Clear;
        SQL.Add('Delete * From Urunler Where RecID='+UrunlerQuery.FieldByName('RecID').AsString);
        Prepared ;
        ExecSQL;
    end;
    BtnIptal.Click;
end;
// ***
procedure TFrmUrunListesi.pnlHeaderMouseDown(Sender: TObject;
    Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    ReleaseCapture;
    self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntUyari;

interface

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

type
    TFrmUyari = class(TForm)
        Image1: TImage;
        BtnTamam: TBitBtn;
        lblCaption: TLabel;
        Label1: TLabel;
        procedure FormCreate(Sender: TObject);
        procedure BtnTamamClick(Sender: TObject);
        procedure Image1MouseDown(Sender: TObject; Button: TMouseButton;
            Shift: TShiftState; X, Y: Integer);
```

```

private
  { Private declarations }
public
  { Public declarations }
  ActCaption:string;
end;

var
  FrmUyari: TFrmUyari;

implementation

{$R *.dfm}

procedure TFrmUyari.FormCreate(Sender: TObject);
begin
  lblCaption.Caption:= ActCaption;
end;
// ***
procedure TFrmUyari.BtnTamamClick(Sender: TObject);
begin
  Self.Close;
end;
// ***
procedure TFrmUyari.Image1MouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  ReleaseCapture;
  self.perform(WM_SYSCOMMAND, $F012, 0);
end;
end.

*****

unit UntQRUretimListesi01;

interface

uses Windows, SysUtils, Messages, Classes, Graphics, Controls,
  StdCtrls, ExtCtrls, Forms, QuickRpt, QRCtrl;

type
  TQRUretimListesi01 = class(TQuickRep)
    TitleBand1: TQRBand;
    ColumnHeaderBand1: TQRBand;
    DetailBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel12: TQRLabel;
    QRDBText1: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;

```

```

QRDBText4: TQRDBText;
QRDBText5: TQRDBText;
QRDBText7: TQRDBText;
QRLabel8: TQRLabel;
QRLabel17: TQRLabel;
QRLabel9: TQRLabel;
QRLabel10: TQRLabel;
QRLabel19: TQRLabel;
QRDBText8: TQRDBText;
QRDBText12: TQRDBText;
QRDBText13: TQRDBText;
QRLabel13: TQRLabel;
QRLabel20: TQRLabel;
QRDBText15: TQRDBText;
QRDBText16: TQRDBText;
QRLabel22: TQRLabel;
QRLabel11: TQRLabel;
QRLabel14: TQRLabel;
QRDBText9: TQRDBText;
QRDBText10: TQRDBText;
QRLabel15: TQRLabel;
QRDBText11: TQRDBText;
QRLabel7: TQRLabel;
QRBand3: TQRBand;
procedure QuickRepBeforePrint(Sender: TCustomQuickRep;
  var PrintReport: Boolean);
private

public
  QRActView:integer;
end;

var
  QRUretimListesi01: TQRUretimListesi01;

implementation

uses UntUretimListe, UntDm,db;

{$R *.DFM}

procedure TQRUretimListesi01.QuickRepBeforePrint(Sender: TCustomQuickRep;
  var PrintReport: Boolean);
var
  tmpSTR:String;
begin
  case QRActView of
    0 : tmpSTR:='Where Durum<>-1 ';
    1 : tmpSTR:='Where Durum=0 ';
    2 : tmpSTR:='Where Durum=1 ';
    3 : tmpSTR:='Where Durum=2 ';
  end;
  with dm.Qrylist do
  begin

```



```

Close;
SQL.Clear ;
SQL.Add('Select *,(RStdMiktar*PTekrar) as TPlan From Uretim ');
SQL.Add(tmpStr);
if FrmUretimListe.CheckBox1.Checked then
    SQL.Add(' and STarih Between "'+ FrmUretimListe.DTStart.Text +'" and "'+
FrmUretimListe.DTEnd.Text + '" ');
SQL.Add(' Order By STarih Asc ');
Prepared;
open;
(FieldByName('RStdMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
(FieldByName('GMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
(FieldByName('TPlan') as TNumericField).DisplayFormat:='0.000 Kg';
end;
end;
end.

```

\*\*\*\*\*

```

unit UntQRUretimListesi02;

```

```

interface

```

```

uses Windows, SysUtils, Messages, Classes, Graphics, Controls,
    StdCtrls, ExtCtrls, Forms, QuickRpt, QRCtrl;

```

```

type

```

```

TQRUretimListesi02 = class(TQuickRep)
    TitleBand1: TQRBand;
    ColumnHeaderBand1: TQRBand;
    DetailBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel12: TQRLabel;
    QRDBText1: TQRDBText;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    QRDBText5: TQRDBText;
    QRDBText7: TQRDBText;
    QRLabel8: TQRLabel;
    QRLabel17: TQRLabel;
    QRLabel9: TQRLabel;
    QRLabel10: TQRLabel;
    QRLabel19: TQRLabel;
    QRDBText8: TQRDBText;
    QRDBText12: TQRDBText;
    QRDBText13: TQRDBText;
    QRLabel13: TQRLabel;
    QRLabel20: TQRLabel;
    QRDBText15: TQRDBText;
    QRDBText16: TQRDBText;

```

```

    QRLabel22: TQRLabel;
    QRLabel11: TQRLabel;
    QRLabel14: TQRLabel;
    QRDBText9: TQRDBText;
    QRDBText10: TQRDBText;
    QRLabel15: TQRLabel;
    QRDBText11: TQRDBText;
    QRLabel7: TQRLabel;
    QRBand3: TQRBand;
    procedure QuickRepBeforePrint(Sender: TCustomQuickRep;
      var PrintReport: Boolean);
  private

  public
    QRActView:integer;
  end;

var
  QRUretimListesi02: TQRUretimListesi02;

implementation

uses UntUretimPlanlama, UntDm,db;

{$R *.DFM}

procedure TQRUretimListesi02.QuickRepBeforePrint(Sender: TCustomQuickRep;
  var PrintReport: Boolean);
var
  tmpSTR:String;
begin
  case QRActView of
    0 : tmpSTR:='Where Durum<>-1 ' ;
    1 : tmpSTR:='Where Durum=0 ' ;
    2 : tmpSTR:='Where Durum=1 ' ;
    3 : tmpSTR:='Where Durum=2 ' ;
  end;
  with dm.Qrylist do
  begin
    Close;
    SQL.Clear ;
    SQL.Add('Select *,(RStdMiktar*PTekrar) as TPlan From Uretim ');
    SQL.Add(tmpStr);
    if FrmUretimPlanlama.CheckBox1.Checked then
      SQL.Add(' and STarih Between "'+ FrmUretimPlanlama.DTStart.Text +' and "'+
        FrmUretimPlanlama.DTEnd.text + '" ');
    SQL.Add(' Order By STarih Asc ');
    Prepared;
    open;
    (FieldByName('RStdMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('GMiktar') as TNumericField).DisplayFormat:='0.000 Kg';
    (FieldByName('TPlan') as TNumericField).DisplayFormat:='0.000 Kg';
  end;
end;
end;
end.

```

## APPENDIX C

### DATABASE STUCTURE

Table Name : CariExtre2005		Table No: 1
Field Name	Type	Size
ID	Long Integer	4
CariKod	Text	5
OpStatus	Text	1
OpDate	Date/Time	8
OpNumber	Text	50
OpType	Text	2
OpDefinition	Text	50
InValue	Double	8
OutValue	Double	8

Table Name : CariHesap		Table No: 2
Field Name	Type	Size
ID	Long Integer	4
CariKod	Text	5
TUnvan	Text	255
Adres01	Text	50
Adres02	Text	50
Ilce	Text	50
Il	Text	50
VDaire	Text	50
VNumara	Text	50
Telefon01	Text	50
Telefon02	Text	50
Fax	Text	50

Cep	Text	50
EMail	Text	50
KayitTarihi	Date/Time	8
WEB	Text	50
InValue	Double	8
OutValue	Double	8

Table Name : Dolum		Table No: 3
Field Name	Type	Size
RecID	Long Integer	4
Bekle01	Double	8
Bekle02	Double	8
Bekle03	Double	8
Mixer	Double	8
Agrega	Double	8

Table Name : DolumSira		Table No: 4
Field Name	Type	Size
RecID	Long Integer	4
Tanim	Text	50
Sira	Integer	2

Table Name : Haklar		Table No: 5
Field Name	Type	Size
RecID	Long Integer	4
PerNo	Long Integer	4
Sira	Long Integer	4
Gorme	Text	1
Kayit	Text	1

Degistir	Text	1
Sil	Text	1
Yazdir	Text	1
TagNo	Long Integer	4

Table Name : Kesme		Table No: 6
Field Name	Type	Size
RecID	Long Integer	4
A01	Double	8
A02	Double	8
A03	Double	8
Kum	Double	8
Cimento	Double	8
Katki	Double	8

Table Name : LastNumbers		Table No: 7
Field Name	Type	Size
ID	Long Integer	4
STLastNumber	Integer	2
URLastNumber	Integer	2
SPLastNumber	Integer	2
CHLastNumber	Long Integer	4

Table Name : Moduller		Table No: 8
Field Name	Type	Size
RecID	Long Integer	4
ModulNo	Long Integer	4
TanimNo	Long Integer	4
Tanim	Text	50

Sira	Long Integer	4
TagNo	Long Integer	4

Table Name : Params		Table No: 9
Field Name	Type	Size
ID	Long Integer	4
PType	Text	2
PCode	Text	10
PDefinition	Text	30

Table Name : Passwords		Table No: 10
Field Name	Type	Size
RecID	Long Integer	4
KullaniciAdi	Text	10
PerNo	Long Integer	4
Sifre	Text	8
Yonetici	Text	1
Gorme	Text	1
Kayit	Text	1
Degistir	Text	1
Sil	Text	1
Yazdir	Text	1

Table Name : Personel		Table No: 11
Field Name	Type	Size
PerID	Long Integer	4

PerAdi	Text	50
PerSoyadi	Text	50
PerNo	Long Integer	4
KullaniciAdi	Text	10
Sifre	Text	8
Yonetici	Yes/No	1

Table Name : Recete		Table No: 12
Field Name	Type	Size
RecID	Long Integer	4
SpektNo	Text	50
ReceteKod	Text	50
ReceteAdi	Text	50
RecTarih	Date/Time	8
Agrega01	Double	8
Agrega02	Double	8
Agrega03	Double	8
Kum	Double	8
Cimento	Double	8
Su	Double	8
Katki01	Double	8
Katki02	Double	8
ToplamMiktar	Double	8
Saat	Text	2
Dakika	Text	2
HPerAdi	Text	50
HPerSoyadi	Text	50
OPerAdi	Text	50
OPerSoyadi	Text	50

Table Name : Stok		Table No: 13
Field Name	Type	Size
ID	Long Integer	4
Kod	Text	6
Adi	Text	30
Birim	Text	10
Miktar	Double	8

Table Name : StokExtre2005		Table No: 14
Field Name	Type	Size
ID	Long Integer	4
Kod	Text	5
OpDate	Date/Time	8
OpNumber	Text	20
OpType	Text	2
OpDefinition	Text	50
InValue	Double	8
OutValue	Double	8

Table Name : Uretim		Table No: 15
Field Name	Type	Size
RecID	Long Integer	4
UretimNo	Text	20
UrunKod	Text	20
UrunAdi	Text	30
ReceteKod	Text	30
ReceteAdi	Text	50
RStdMiktar	Double	8
PTekrar	Long Integer	4
GMiktar	Double	8



GTekrar	Long Integer	4
STarih	Text	20
SSaat	Text	20
ETarih	Text	20
ESaat	Text	20
Durum	Integer	2

Table Name : Urunler		Table No: 16
Field Name	Type	Size
RecID	Long Integer	4
SpektNo	Text	50
UrunKod	Text	50
UrunAd	Text	50
UrunTip	Text	50

## **APPENDIX D**

### **INTRODUCTION TO ISO**

#### **Breif information to ISO 9000 and ISO 14000**

ISO has been developing voluntary technical standards over almost all sectors of business, industry and technology since 1947. Then, in 1987, came ISO 9000, followed nearly 10 years later by ISO 14000, which have brought ISO to the attention of a much wider business community.

With the exception of ISO 9000 and ISO 14000, the vast majority of ISO standards are highly specific. They are documented agreements containing technical specifications or other precise criteria to be used consistently as rules, guidelines, or definitions of characteristics to ensure that materials, products, processes and services are fit for their purpose.

The ISO 9000 and ISO 14000 families are among ISO's most widely known standards ever. ISO 9000 and ISO 14000 standards are implemented by some 887 770 organizations in 161 countries. ISO 9000 has become an international reference for quality management requirements in business-to-business dealings, and ISO 14000 is well on the way to achieving as much, if not more, in enabling organizations to meet their environmental challenges.

The ISO 9000 family is primarily concerned with "quality management". This means what the organization does to fulfil:

- the customer's quality requirements, and
- applicable regulatory requirements, while aiming to
- enhance customer satisfaction, and
- achieve continual improvement of its performance in pursuit of these objectives.

The ISO 14000 family is primarily concerned with "environmental management". This means what the organization does to:

- minimize harmful effects on the environment caused by its activities, and to
- achieve continual improvement of its environmental performance.

The vast majority of ISO standards are highly specific to a particular product, material, or process. However, the standards that have earned the ISO 9000 and ISO 14000 families a worldwide reputation are known as "generic management system standards".

"**Generic**" means that the same standards can be applied:

- to any organization, large or small, whatever its product,
- including whether its "product" is actually a service,
- in any sector of activity,
- whether it is a business enterprise, a public administration, or a government department.

"**Generic**" also signifies that no matter what the organization's scope of activity, if it wants to establish a quality management system or an environmental management system, then such a system has a number of essential features for which the relevant standards of the ISO 9000 or ISO 14000 families provide the requirements.

"**Management system**" refers to the organization's structure for managing its processes - or activities - that transform inputs of resources into a product or service which meet the organization's objectives, such as satisfying the customer's quality requirements, complying to regulations, or meeting environmental objectives.

## **Quality Management Principles**

Eight quality management principles on which the quality management system standards of the revised ISO 9000:2000 series are based. These principles can be used by senior management as a framework to guide their organizations towards improved performance.

The principles are derived from the collective experience and knowledge of the international experts who participate in ISO Technical Committee ISO/TC 176, Quality management and quality assurance, which is responsible for developing and maintaining the ISO 9000 standards.

The eight quality management principles are defined in ISO 9000:2000, Quality management systems Fundamentals and vocabulary, and in ISO 9004:2000, Quality management systems Guidelines for performance improvements.

This document gives the standardized descriptions of the principles as they appear in ISO 9000:2000 and ISO 9004:2000. In addition, it provides examples of the benefits derived from their use and of actions that managers typically take in applying the principles to improve their organizations' performance.

**Principle 1 : Customer focus ;**

Organizations depend on their customers and therefore should understand current and future customer needs, should meet customer requirements and strive to exceed customer expectations.

***Key benefits:***

- Increased revenue and market share obtained through flexible and fast responses to market opportunities.
- Increased effectiveness in the use of the organization's resources to enhance customer satisfaction.
- Improved customer loyalty leading to repeat business.
- Applying the principle of customer focus typically leads to:
- Researching and understanding customer needs and expectations.
- Ensuring that the objectives of the organization are linked to customer needs and expectations.
- Communicating customer needs and expectations throughout the organization.
- Measuring customer satisfaction and acting on the results.
- Systematically managing customer relationships.
- Ensuring a balanced approach between satisfying customers and other interested parties (such as owners, employees, suppliers, financiers, local communities and society as a whole).

**Principle 2 : Leadership ;**

Leaders establish unity of purpose and direction of the organization. They should create and maintain the internal environment in which people can become fully involved in achieving the organization's objectives.

***Key benefits:***

- People will understand and be motivated towards the organization's goals and objectives.
- Activities are evaluated, aligned and implemented in a unified way.
- Miscommunication between levels of an organization will be minimized.
- Applying the principle of leadership typically leads to:
- Considering the needs of all interested parties including customers, owners, employees, suppliers, financiers, local communities and society as a whole.
- Establishing a clear vision of the organization's future.
- Setting challenging goals and targets.
- Creating and sustaining shared values, fairness and ethical role models at all levels of the organization.
- Establishing trust and eliminating fear.
- Providing people with the required resources, training and freedom to act with responsibility and accountability.
- Inspiring, encouraging and recognizing people's contributions.

**Principle 3 : Involvement of people;**

People at all levels are the essence of an organization and their full involvement enables their abilities to be used for the organization's benefit.

***Key benefits:***

- Motivated, committed and involved people within the organization.
- Innovation and creativity in furthering the organization's objectives.
- People being accountable for their own performance.
- People eager to participate in and contribute to continual improvement.

- Applying the principle of involvement of people typically leads to:
- People understanding the importance of their contribution and role in the organization.
- People identifying constraints to their performance.
- People accepting ownership of problems and their responsibility for solving them.
- People evaluating their performance against their personal goals and objectives.
- People actively seeking opportunities to enhance their competence, knowledge and experience.
- People freely sharing knowledge and experience.
- People openly discussing problems and issues.

#### **Principle 4 : Process approach ;**

A desired result is achieved more efficiently when activities and related resources are managed as a process.

##### **Key benefits:**

- Lower costs and shorter cycle times through effective use of resources.
- Improved, consistent and predictable results.
- Focused and prioritized improvement opportunities.
- Applying the principle of process approach typically leads to:
- Systematically defining the activities necessary to obtain a desired result.
- Establishing clear responsibility and accountability for managing key activities.
- Analysing and measuring of the capability of key activities.
- Identifying the interfaces of key activities within and between the functions of the organization.
- Focusing on the factors such as resources, methods, and materials that will improve key activities of the organization.
- Evaluating risks, consequences and impacts of activities on customers, suppliers and other interested parties.

**Principle 5 : System Approach To Management;**

Identifying, understanding and managing interrelated processes as a system contributes to the organization's effectiveness and efficiency in achieving its objectives.

***Key benefits:***

- Integration and alignment of the processes that will best achieve the desired results.
- Ability to focus effort on the key processes.
- Providing confidence to interested parties as to the consistency, effectiveness and efficiency of the organization.
- Applying the principle of system approach to management typically leads to:
- Structuring a system to achieve the organization's objectives in the most effective and efficient way.
- Understanding the interdependencies between the processes of the system.
- Structured approaches that harmonize and integrate processes.
- Providing a better understanding of the roles and responsibilities necessary for achieving common objectives and thereby reducing cross-functional barriers.
- Understanding organizational capabilities and establishing resource constraints prior to action.
- Targeting and defining how specific activities within a system should operate.
- Continually improving the system through measurement and evaluation.

**Principle 6 : Continual Improvement;**

Continual improvement of the organization's overall performance should be a permanent objective of the organization.

***Key benefits:***

- Performance advantage through improved organizational capabilities.
- Alignment of improvement activities at all levels to an organization's strategic intent.
- Flexibility to react quickly to opportunities.
- Applying the principle of continual improvement typically leads to:

- Employing a consistent organization-wide approach to continual improvement of the organization's performance.
- Providing people with training in the methods and tools of continual improvement.
- Making continual improvement of products, processes and systems an objective for every individual in the organization.
- Establishing goals to guide, and measures to track, continual improvement.
- Recognizing and acknowledging improvements.

### **Principle 7 : Factual Approach To Decision Making ;**

Effective decisions are based on the analysis of data and information

#### ***Key benefits:***

- Informed decisions.
- An increased ability to demonstrate the effectiveness of past decisions through reference to factual records.
- Increased ability to review, challenge and change opinions and decisions.
- Applying the principle of factual approach to decision making typically leads to:
- Ensuring that data and information are sufficiently accurate and reliable.
- Making data accessible to those who need it.
- Analysing data and information using valid methods.
- Making decisions and taking action based on factual analysis, balanced with experience and intuition.

### **Principle 8 : Mutually Beneficial Supplier Relationships;**

An organization and its suppliers are interdependent and a mutually beneficial relationship enhances the ability of both to create value.

#### ***Key benefits:***

- Increased ability to create value for both parties.
- Flexibility and speed of joint responses to changing market or customer needs and expectations.



- Optimization of costs and resources.
- Applying the principles of mutually beneficial supplier relationships typically leads to:
  - Establishing relationships that balance short-term gains with long-term considerations.
  - Pooling of expertise and resources with partners.
  - Identifying and selecting key suppliers.
  - Clear and open communication.
  - Sharing information and future plans.
  - Establishing joint development and improvement activities.
  - Inspiring, encouraging and recognizing improvements and achievements by suppliers.

### **The Next Step:**

This document provides a general perspective on the quality management principles underlying the ISO 9000:2000 series. It gives an overview of these principles and shows how, collectively, they can form a basis for performance improvement and organizational excellence.

There are many different ways of applying these quality management principles. The nature of the organization and the specific challenges it faces will determine how to implement them. Many organizations will find it beneficial to set up quality management systems based on these principles.