DELPHİ

Uzantı Açılımı Açıklama (DCU Delphi Compiled Unit)
Delphi Derlenmiş Birimi
Derleme sırasında bir Pascal dosyasının derlenmesi sonucunda oluşur. Kaynak kodu olmadığı zaman derleme için bunu kullanabilirsiniz. DFM Delphi Form File
Delphi Form Dosyası
Bir formun veya veri modülünün ve içerdiği bileşenlerin özelliklerinin tanımlarını içeren ikilik dosya. Geliştirme sırasında oluşturulur. ~DF Delphi Form Dosyası(DFM) Yedeği. DOF Delphi Options File
Delphi Seçenekler Dosyası
Proje seçeneklerinin mevcut ayarlarını içeren bir metin dosyası. Geliştirme sırasında oluşturulur. Delphi 1'de uzantısı OPT idi. DPR Delphi Proje Dosyası Geliştirme sırasında oluşturulan bu dosya gerçek Pascal kaynak kodu içerir. ~DP Delphi Proje Dosyası (DPR) yedeği. DSK Masaüstü dosyası Bu dosya pencerelerin konumları,editördeki açık dosyalar ve diğer masaüstü ayarları hakkında bilgiler içerir. Projeyi yeni bir dizine kopyalarken silmeniz gerekir. PAS Pascal dosyası Bir pascal biriminin kaynak kodu, bir formla ilgili veya bağımsız bir birim. ~PA Pascal dosyası (PAS) yedeği. RES Resource (Kaynak) dosyası Proje ile ilgili ve genellikle simgesini içeren ikilik dosya. Bu türden başka dosyalarda projeye ekleyebilirsiniz. BPG Borland Project Group Birden fazla projeyi ayni anda derlemek için kullanılır. CFG Proje seçeneklerini içeren ayar dosyası. Sadece özel derleyici seçenekleri ayarlandığı zaman oluşur. DPK Delphi Package Bir paketin kaynak kodunu içerir. TLB Type Library
Tip Kütüphanesi OLE Server uygulamaları için Type Library Editor tarafından otomatik olarak oluşturulan bir dosyadır. TODO Tüm projedeki yapılacak isler listesini içeren dosya ULD Microsoft Data Link ADO tarafından bir veri tedarikçisine gönderme yapmak için kullanılır. BDE'deki alias'a benzer. DEM Delphi Edit Mask Düzenleme maskeleri için ülkelere özgü formatlar içerir.

VERİ TABANI

Delphi'nin tercih edilmesinin en büyük nedenlerinden biriside iyi bir veritabanı desteğine sahip olması ve veritabanı programı yazmayı son derece kolaylaştırmasıdır. Hatta Delphi ile hiç kod yazmadan bir veritabanı programı yapabilirsiniz. Ancak buna rağmen veritabanı programı yazmak zor ve tecrübe isteyen bir istir. Burada her şeyin anlatılması mümkün değil. Elimizden geldiği kadar veritabanının genel olarak yapısını ve veritabanı programcılığının temelini anlatmaya çalışacağız.
Delphinin bildiğiniz gibi 3 sürümü var. Standart, Profesyonel ve Client/Server (C/S). Bu sürümlerin sunduğu veritabanı destecide farklı. Profesyonel sürüm standart sürümden daha iyi veritabanı desteği ve araçları sunuyor, C/S ise profesyonelden daha iyi.
Database ile uğraşırken bir çok terim ile karsılaşacaksınız. BDE, SQL, client, server, table, query vs. Bunları fazla problem etmeyin. Bunları zamanla öğrenirsiniz.
Veritabanını, verilerin depolandığı tablolar seklinde düşünebiliriz. Tablolar ise ad, soyad, telefon gibi alanlardan (field) oluşur. Bu alanların tamamı ise bir kayıtı (record) oluşturur.
Veritabanı tek tablodan oluşabileceği gibi birden fazla tablodan da oluşabilir. Yüzlerce tabloya sahip veritabanlarınız olabilir. Yine bu tablolarda 100 kayıt tutabileceğiniz gibi milyonlarca kayıtta tutabilirsiniz.
Indeksler(index) veritabanlarını hızlandırmak için kullanılır. Bir tabloda bir veya birden fazla indeks olabilir. Indeksleri seçerken en çok işlem yaptığınız alanları seçmeniz iyi olacaktır.

Mesela raporlarınızı daha çok tarihe göre ve il bazında alıyorsanız tarih ve il_kodu alanlarını index tanımlamalısınız.
Tablolarda alan(field) ve kayıt(record) karıştırmamanız lazım. Mesela ad, soyad, telefon ve adres'ten oluşan bir tablonuz var. Bunu bir excel sayfası olarak düşünebilirsiniz. AdSoyadTelefonAdresAliAkin0 216 123 45 67....ZaferYildirim0 212 123 45 67...gibi. Burada yukarıdan aşağı doğru olan kolonlar alan'dır. Yani ad alanı, soyad alanı gibi. Soldan sağa doğru olan sütunlar ise kayıttır. Yani (Ali, Akın, 0 216 123 45 67, ...) bir kayıttır.
Veritabanları etkin kayıtın yerini göstermek için kürsör(cursor) denen bir yapı kullanırlar. Mesela siz programla Zafer Yıldırım isimli kayıtı görüntülediğiniz zaman veritabanında kür sor bu kayıtı gösterir.
Birde dataset kavramı var. Dataseti söyle açıklayabiliriz: Birden çok tablodan elde ettiğimiz veri topluluğu. Mesela müşterilerinizin adini ve adresini sakladığınız bir tablo, siparişlerini sakladığınız bir tablo ve siparişlerin detayını sakladığınız bir tablo olmak üzere 3 tablonuz var. Siz Ali Tas isimli müşterinin Ocak ayında yaptığı siparişlerin ayrıntılarını görmek istiyorsunuz. Bu 3 tablodan da veri alırsınız. Bu bir dataset olur.

Lokal ve C/S veritabanları
Veritabanlarını iki kısımda ele alabiliriz. Lokal veritabanları ve C/S veritabanları.
Lokal veritabanları tek bir makinede çalışmak üzere tasarlanmış veritabanlarıdır. Bu verilere sadece sizin programınız erişecek ve başkaları bu verilere erişmeyecekse lokal veritanını kullanabilirsiniz. Paradox, dBase ve Access lokal veritabanlarıdır.
C/S Veritabanları: Veritabanınız bir server üzerinde durur ve birden fazla kullanıcı (client) bu verilere erişir ve işlem yapar. C/S veritabanları bir kayıta birden fazla kişinin ayni anda erişmesi gibi olayları kendileri kontrol ederler ve bu tip durumların üstesinden nasıl geleceklerini bilirler. Interbase, Oracle, SQL Server, Sybase, Informix, DB2 kullanılan C/S veritabanlarıdır.
C/S veritabanlarının bir dezavantajı veritabanını kullanan kullanıcı basına para ödemenizdir. Şirketler burada iki farklı lisans yöntemi sunuyorlar. Birincisi kullanan kullanıcı başına para ödüyorsunuz (per seat). 40 tane kullanıcınız(client) varsa 40 client lisansı almalısınız. Diyelim ki kullanıcı sayınız arttı ve 50 oldu. 10 tane daha client lisansı almanız gerekir. İkinci bir yöntem ise veritabanına ayni anda kaç kullanıcı bağlanıyorsa o kadar lisans almaktır (per connection). Mesela 100 kullanıcınız veritabanını kullanıyor ancak ayni anda en fazla 30 kişi veritabanına bağlanıyorsa 30 lisans almalısınız.
C/S veritabanları içinde su anda Oracle en iyisi. Onun dışında Türkiye piyasasında SQL Server'da çoğunlukla kullanılmakta. Benim tercihim Oracle'dan yana. Eğer C/S veritabanlarında kendinizi geliştirmek istiyorsanız Oracle veya SQL Server seçin.
Lokal veritabanları ile da birden çok kullanıcının kullandığı programlar yazmak mümkün. Ancak çoğu şeyi programla halletmeniz gerekiyor. Genellikle de problemler çıkmakta.

Tek katmanlı, iki katmanlı ve çok katmanlı database yapıları
Tek katmanlı(single-tier): Genel olarak lokal veritabanları tek katmanlıdır. Tek katmanlı yapıda program veritabanına direk ulaşır ve yapılan işlemler (kayıt ekleme, kayıt silme, değiştirme) anında gerçekleştirilir.
İki katmanlı(two-tier): Burada program client tarafında çalışır. Client tarafında çalışan program gerekli sürücüleri kullanarak serverdaki veritabanına ulaşır.
Çok katmanlı(multi-tier): Burada program yine client tarafında çalışır. Program database server'la direk bağlantı kurmaz. Server tarafında çalışan bir application server ile bağlantı kurar. Bu yapı genellikle güvenlik ve hız amacıyla kullanılır.

BDE (Borland Database Engine)
BDE Delphi'nin lokal ve C/S veritabanlarına bağlanmak için kullandığı dll ve uygulamalardır. C/S veritabanlarına bağlanmak için Delphi'nin C/S sürümünü kullanmak zorundasınız. Bu sürümle birlikte gelen SQL Links, BDE tarafından C/S veritabanlarına bağlantı için kullanılır.

Delphi BDE gibi bir yapıyı niye kullanıyor?
Normalde veritabanlarının yapıları ve API'leri farklı farklıdır. BDE programcıyı tüm bu yapıları öğrenmekten kurtararak daha üst düzey komutlarla program yapımına imkan tanır.
Delphi ile gelen sürücüler kullandığınız Delphi sürümüne göre değişir. Delphinin tüm sürümleri ile Paradox ve dBase bağlanmayı sağlayan sürücüler gelir. Bu sürücüler STANDART olarak adlandırılır ve paradox ve dBase ile yapacağınız tüm işlemleri yapmanızı sağlar. Delphi'nin C/S sürümü ile Oracle, SQL Server, Interbase, Sybase, Informix gibi C/S veritabanlarına bağlanmanızı sağlayacak sürücülerde gelir.

Alias (Takma isim/Rumuz)
Alias'lar veritabanlarına bağlantı ve bağlantının özelliklerini ayarlamak için kullanılır. Bir alias BDE'ye hangi tür bir veritabanına bağlanacağı, veritabanı dosyalarının diskte nerede olduğu gibi bilgileri bildirir. Ayrıca eğer C/S bir veritabanı kullanıyor iseniz açılış modu, kullanıcı ismi, BLOB alanların büyüklüğü gibi verileri BDE'ye bildirir.

Lokal veritabanları için Alias oluşturma
1. BDE Administrator programını çalıştırın.
2. Object menüsü altından New... komutunu verin veya klavyeden Ctrl+N tuşlarına basın.
3. Database Driver Name kısmından STANDART'ı seçin.
4. Alias'in ismini yazın ve Object menüsünden Apply komutunu verin.
5. Definition kısmında Default Driver kısmından kullandığınız veritabanını seçin.
6. Path kısmına veritabanınızın olduğu yeri seçin.
7. Object menüsünden Apply komutunu verin veya klavyeden Ctrl+A tuşlarına basın.

VERİ TABANI BİLEŞENLERİ
Bilesen paletinde **Data Access** ve **Data Controls** sayfalarında bulunurlar. Genel olarak bir veri tabanına bağlanıp veriler üzerinde insert, update,delete veya belli kayıtların görüntülenmesi için kullanılır. Bileşenler aşağıdaki şekillerde görülmektedir.

Data Access sayfasındaki bileşenler unvisible (yani program çalıştığı zaman ekranda gözükmeyen) bileşenlerdir. Bu bileşenler **Data Controls** sayfasındaki bileşenler yardımı ile görüntülenecek veriler için veritabanları ile köprü vazifesi görürler.
TTable: En önemli iki özelliği Database Name'i ve Table Name'dir. Database Name'e BDE içinden tanımladığınız herhangi bir alias'i, projeniz içindeki Database bileşeninin Database Name'ini (bunlar combobox içinde otomatik olarak gelirler) veya paradox vb. gibi tablolar için tabloların bulunduğu dizinin adini verebilirsiniz. Bu bileşenin diğer önemli özelliklerinden biri de Index Name-Index Fields özelliğidir. Bu özelliği herhangi bir kayıta ulaşmak için kullanılır. Herhangi bir kayıta ulaşmak veya istenilen bir kayıtı bulmak için genel olarak üç çeşit yöntem kullanılır. Bunlar

FindKey()
Locate()

Lookup()
prosedürleridir. Birbirinden farkları sudur. Eğer findkey'i kullanıyorsanız index name ve index fields özellikleri belirtilmiş olmalıdır. Bu fonksiyon istenilen kayıt bulunmuşsa True aksi halde False değerini döndürür. Kullanılışı su şekildedir :
Table1.FindKey([degisken1,degisken2,…]) gibi. Buradaki değişken şayisi index fields özelliğinde tanımlanan veya index name ile belirtilen indexin sahip olduğu alan sayısına eşit olmalıdır ve o alanlara karşılık gelen değerler verilmelidir.
Örnek: Index Fields=Numara olsun. Bu durumda kod su şekilde olmalıdır.
Table1.FindKey([2500]) gibi. Burada indekste belirtilen alanın tipi ile koda yazdığımız tip birbirini tutmalıdır. Eğer Index Fields=Adi;Soyadı seklinde ise kod
Table1.FindKey(['Ahmet','SAVAS']) seklinde olmalıdır.
Locate prosedürü de FindKey gibi çalışır. Fakat bunda alan isimlerini de kendiniz verirsiniz. Eğer belirlediğiniz alanlara ait bir index varsa kullanılır, yoksa sıralı arama yapılır. Kullanılışı:
Table1.Locate('adi;soyadi',VarArrayof(['Ahmet','SAVAS']),[loCaseInsensitive,loPartialKey]) seklindedir.
loCaseInsensitive: Büyük harf-küçük harf ayrımı yapılmaz.
loPartialKey: Bunu kullanırsanız eğer sadece SAVAS'i delilde eğer SAVASÇI da varsa onuda bulabilirsiniz.
Lookup da ayni şekildedir. Farklı yani ise sudur; FindKey ve Locate de cursor bulunan kayıtın üstüne gider, lookup da ise nerde iseniz orda durur. Bir de bu fanksiyonları kullanırken dikkat edeceğiniz diğer bir husus en son yaptığınız işlemdir. Bu prosedürlerden herhangi birini çağırdığınız zaman en son yapılan işlem otomatik olarak kaydedilir. Örneğin iptal etmeniz gereken bir işlem varsa veya tablo insert modandaysa bu durumlarıda kontrol etmelisiniz. Diğer sık kullanılan prosedürleri ise Edit, Insert, Post'dur.
Tablonun hangi durumda olduğunu Table1.State ile öğrenebilirsiniz. Örneğin Table1.State in [dsEdit] tablonun edit modunda olduğunu gösterir.
TQuery: Bu bileşende TTable bileşeni ile hemen hemen aynidir. Fakat bu bilesen ve SQL yardımı ile kayıtlar üzerinde sıralama, sadece belli kayıtları görüntüleme vb. işlemler çok daha rahat yapılabilmektedir. Ayni şeyler TTable bileşenin Filter, Range gibi özellikleri kullanılarak da yapılabilir. Fakat, performans açısından bakıldığında TQuery'leri kullanmak her zaman faydalıdır. Query'ler normalde **Read-Only**'dirler, yani kayıtlar üzerinde değişiklik yapamazsınız. Eğer kayıtlar üzerinde değişiklik yapmak istiyorsanız, **RequestLive** özelliğini True yapmalısınız. Bu şekilde kayıtlar üzerinde değişiklik yapıp, yapılan değişiklikleri table'da olduğu gibi kaydedebilirsiniz. Fakat SQL cümleciğiniz birkaç tablodan veri alıp getiriyorsa o zaman RequestLive özelliğini kullanmazsınız. Bu şekildeki query'ler üzerinde değişiklik yapabilmeniz için önce **CachedUpdates** özelliğini True yapmalısınız. (Bu arada RequestLive, False olmalıdır.) CachedUpdates özelliği True yapıldığında kayıtlar üzerinde güncelleme, değiştirme ve silme yapabildiğinizi göreceksiniz. Fakat bu değişiklikler sadece programda kalır ve fiziksel veritabanını etkilemez. Yaptığımız değişikliklerin kalıcı olması için **TUpdateSQL** bileşenini kullanırız.Bu işlem için aşağıdaki adımları takip etmelisiniz.

- TQuery ve TUpdateSQL bileşenlerini yerleştirin
- TQuery'nin SQL ifadesini yazın.
- TQuery'nin UpdateQbject özelliğini UpdateSQL1 olarak seçin(veya ne isim verdiyseniz)
- UpdateSQL1 bileşeninin seçip mouse'un sağ tuşuna basın. Oradan **UpdateSQL Editor…**'ü tıklayın.

İlk önce üzerinde değişiklik yapmak istediğiniz tabloyu seçmelisiniz. Daha sonra Key Fields ile belirtilen genellikle sizin üzerinde değişiklik yapmadığınız alanlar seçilmelidir. Numara vb. Daha sonra değiştirilecek alanlar Update Fields da belirtilen alanlardan seçilmelidir. Daha sonra ise **Generate SQL** butonuna bastığınızda sizin için gerekli SQL'lerin oluştuğunu göreceksiniz. Daha sonra kod içinde istediğiniz herhangi bir yerde UpdateSQL1.ExecSQL() prosedürünü kullanarak yaptığımız işlemin kalıcı olmasını sağlayabiliriz. ExecSQL'in alacağı parametreler;
ukModify : Güncelleme işleminin kalıcı olması için
ukInsert : Yeni girilen bir kayıtın veritabanında olması için
ukDelete : Sildiğiniz bir kayıtın veritabanından da silinmesi için. TTable üzerinde geçerli olan işlemlerin birçoğu TQuery içinde geçerlidir.

TStoredProc: Bu bilesen SQL tabanlı veritabanları üzerinde yazdığınız prosedürleri veya fonksiyonları kullanmanızı sağlayan bir bileşendir. Eğer Oracle, Sysbase, SQL Server gibi veritabanı kullanmıyorsanız bu bilesene ihtiyacınız yok demektir. Fakat adi anılanlardan herhangi biri veya bunlara benzer bir veritabanı sistemi kullanıyorsanız bu bilesen çok isinize yarayacaktır. Avantajları temel olarak şunlardır. Bu çalıştıracağınız prosedürler veritabanı üzerindedir. Normal kullandığınız query'lere göre çok daha hızlı çalışırlar. Network trafiğini asgari seviyeye indirirler. Yazacağınız prosedürle ilgili kullandığınız veritabanının dökümatasyonuna bakmalısınız.
Örn: Oracle'da prosedür söyle olsun.

```
PROSEDURE TEST (ISIM OUT VARHAR2, NUMARA NUMBER) IS
BEGIN
SELECT ADI INTO :ISIM FROM ABONE WHERE ABONE_NO=:NUMARA;
END;
```

Bunu Delphi'de aşağıdaki gibi kullanabilirsiniz;

```
StoredProc1.ParamByName('Numara').Value:='1000';
StoredProc1.GetResults;
Abone_Ismi:= StoredProc1.ParamByName('Isim').AsString;
```

TDatabase: Bu bileşeni projeniz içinde yönetim kolaylığı sağlamak için kullanabilirsiniz. Mesela TEST diye BDE alias'iniz var. Projedeki bütün table'lar query'ler vs. hepsi buna bağlı. Bunun ismini değiştirdiğinizde bütün projede gidip aliaslari değiştirmeniz gerekir. Bunun yerine TDatabase bileşenini kullanırsanız, sadece bu bileşenin alias'ini değiştirdiğinizde projede buna bağlı ne kadar bilesen varsa hepsini etkileyecektir.
TDataSource: Verilerinizin data controls sayfasındaki bileşenler yardımı ile görüntülenmesi için table,query vb. gibi datasetleri mutlaka bir **DataSource**'a bağlamanız gerekir.

Data Controls:
TDBGrid: Verilerin gösterilmesi için kullanılır. Gösterilecek alanlar ayarlanabilir, verilerin fontu ve grid'in başlık fontu ayrı ayrı ayarlanabilir.
TDBNavigator: Veriler üzerinde güncelleme, silme, yeni kayıt, ileri-geri gitme vs. gibi işlemlerin yapıldığı araç çubuğudur. Bunu bir datasource'a bağladığınız zaman bu datasource'un bağlı olduğu dataset navigator de yaptığınız tüm işlemlerden etkilenir.
TDBText: Label ile aynıdır. Fakat bağlı olduğu tablodan belirtilen alan bilgisini görüntüler. Genelde üzerinde değişiklik yapılmayacak alanların gösterilmesi için uygundur.
TDBEdit: DBText ile benzerdir, ek olarak eğer imkan dahilinde ise veriler üzerinde değişiklik yapılabilir.
TDBMemo: Birden fazla satirin yada 255 karakterden daha uzun verilerin saklanması ve gösterilmesi için kullanılır.

TDBImage: Veritabanlarında resim içeren alanların islenmesi için kullanılır.
TDBListBox: Bunun özelliği şudur. Verdiginiz alan degeri eger listeni içinde bulunuyorsa otomatik olarak seçilir. Bilesenin listesini siz kendiniz doldurmak zorundasiniz. Verdiğiniz alan ile ilgili değerler otomatik olarak gelmez. Mesela listesi Table1'in Isim alanına bağladınız. Etkin kayıttaki isim 'Ahmet' ve liste elemanları içinde de 'Ahmet' varsa bu eleman otomatik olarak işaretlenir. Aksi halde yani liste elemanları içinde 'Ahmet' yoksa listede hiçbir eleman işaretlenmez. Büyük-küçük harf fark etmez.
TDBComboBox: TListBox ile aynidir. Eğer elemanları arasında etkin kaydın ilgili değeri varsa görüntülenir, diğer halde bos olarak gözükür.
TDBCheckBox: Genelde iki durumlu veriler için kullanılır. Mesela, personel tablosu için düşünecek olursanız kişi yönetici veya değil. Bunun için kullanılabilir. Yönetici olması 1, olmaması 0 ile gösterilecek olursa **ValueChecked**=1, **ValueUnChecked**=0 olmalıdır. Eğer veriyi sadece göstermek amacı ile bu bileşeni göstermek kullanmak isterseniz o zaman **ValueChecked ve ValueUnChecked** özelliklerine birden fazla değer girebilirsiniz. Mesela 2 üst düzey yöneticiyi göstersin. Eğer detay gerekmiyorsa yönetici olduğunu göstermek için ValueChecked=1;2 seklinde değer girebilirsiniz.
TDBRadioGroup: Bu bilesen tablolarda kod olarak tutulan fakat programda açık olarak yazılması gereken alanlar için kullanılır. Mesela, 0-Isçi, 1-Yönetici, 2-Üst düzey yönetici ise bu bileşenin Items özelliğine alt alta Isçi, Yönetici ve Üst Düzey Yönetici, Values özelliğine de yine alt alta 0,1 ve 2 yazarsanız veritabanında 0 olduğu zaman Isçi, 1 olduğu zaman Yönetici ve 2 olduğu zaman Üst Düzey Yönetici otomatik olarak RadioGroup içinde seçilecektir. Ayni bunları yeni veri girişi yaparken de kullanabilirsiniz. (Son kullanıcılar personelin durumunu girerken 0'dan pek bir şey anlamazlar. :))
TDBLookupListBox: Bu bilesen başka bir tablonun belirtilen alanında bulunan tüm verileri görmek ve ayni zamanda istenirse bu verilerle ilgili bilgileri başka bir tabloya kaydetmek için kullanılır. Örn: Personel tablonuzda personelin oturduğu il kodunu tutuyorsunuz. Bunun için birde IlKodu,IlAdi'ni içeren Iller tablonuz var. O zaman bu bileşenin Datasource'u olarak personel, datafield'i ilkodu, ListSource'u iller, listFields'i il_adi ve keyField'i da il_kodu verdiğiniz zaman illerden Ankara'yı seçtiğiniz zaman personelin ilkodu alanına '06' otomatik olarak yazılacaktır. Ayni şekilde ilkodu '34' olan bir personelde Illerden Istanbul otomatik olarak seçilmektir.
TDBLookupComboBox: TDBLookupListBox ile aynidir.
TDBRichEdit: Eğer veritabanlarında 64Kb'den daha büyük bilgini islenmesi gerekiyor bu bilesen kullanılır.
TDBCtrlGrid: DBGrid gibidir. Fakat bunda her bir kayıtın için gösterilecek alanları, bu olanların nasıl gösterileceğini (her biri ayrı bir renkte, farklı büyüklükte gibi) belirleyebilirsiniz.
TDBChart: Veritabanındaki verilerden otomatik grafik oluşturmak için kullanılır.

SQL

SQL (**S**tructured **Q**uery **L**anguage) veri tabanlarındaki verileri islemek için kullanılan yapısal sorgulama dilidir.
Bu dil yardımıyla veritabanlarındaki tüm işlemler yapılabilir. Backup almadan tutunda bir tabloya veri girmeye varıncaya kadar hersek.
SQL'i su anda piyasada bulunan hemen hemen her veritabanında kullanabilirsiniz. SQL'de her veritabanında kullanılan ortak ifadeler olmasına karşın, veritabanlarının kendine özgü ifadeleri de

vardır. Mesela Oracle'da SQL ile yapabildiğiniz bazı şeyleri başka veritabanlarında yapamayabilirsiniz.
SQL temel olarak su ifadelerle kullanılır. SELECT, FROM, WHERE, ORDER BY, GROUP BY, HAVING, UPDATE, DELETE, INSERT.
Burada kullandığımız SQL cümleleri ISCI adli bir tablo üzerine yazılmıştır. Alanlar ISCI_NO, ISCI_ADI, YAS, GIRIS_TARIHI, MAAS.
SELECT: Tablodan seçmek istediğimiz alanları belirtmek için kullanılır. Eğer tablodan tüm alanları seçmek istiyorsak o zaman alan isimleri yerine * işareti konur.
FROM: Üzerinde işlem yapılacak tablo/tabloları belirtmek için kullanılır.
WHERE: Tablodan eğer tüm kayıtları delilde istediğimiz bazı kayıtları elde etmek istiyorsak, örnekte maaşı 250 milyondan fazla olan isçilerin numarası ve adi gibi, o zaman buraya istediğimiz kriteri yazarız.

```
SELECT ISCI_NO, ISCI_ADI
FROM ISCI
WHERE MAAS>250000000
```

DISTINCT: Birbirinin ayni olan satırların listelenmemesi için bu ifade kullanılır. Mesela ISCI tablosunda bulunan birbirinin ayni olmayan isimleri listelemek istersek

```
SELECT DISTINCT ISCI_ADI
FROM ISCI
```

seklinde bir SQL ifadesi yazarız.
IN: Koşul belirtirken kullanırız. Mesela ismi AHMET, ALI veya MUSTAFA olan isçilerin bilgilerini listelemek için

```
SELECT *
FROM ISCI
WHERE ISCI_ADI='AHMET' OR ISCI_ADI='ALI' OR ISCI_ADI='MUSTAFA'
```

seklinde bir ifade kullanırız. Bunun yerine

```
SELECT *
FROM ISCI
WHERE ISCI_ADI IN ('AHMET' ,'ALI' ,'MUSTAFA')
```

ifadesini de kullanabiliriz. Yani listenin içindeki herhangi bir değerin bulunması kayıtın seçilmesi için yeterlidir.
LIKE: Eğer aradığımız kayıtın bulunması için tam bir karsılaştırma yapamıyorsak o zaman kullanırız. Mesela isminin bas harfi A ile başlayan isimleri bulmak için

```
SELECT *
FROM ISCI
WHERE ISCI_ADI LIKE 'A%' ifadesi kullanılır.
```

% işareti uzunluğu önemsiz olmak üzere yazıldığı yere her türlü ifade gelebilir anlamındadır.
? işareti ise bir karakter olmak üzere her türlü değeri alabilir anlamındadır. Mesela isminin sondan üçüncü harfi A, ve son harfi Z olan kayıtları listelemek istersek sondan ikinci harfin ne olduğu önemli değildir. O zaman o harf yerine aşağıda görüldüğü üzere ? işaretini kullanırız.

```
SELECT *
FROM ISCI
WHERE ISCI_ADI LIKE '%A?Z' ifadesi kullanılır.
```

BETWEEN: Koşul belirtirken iki değer arasını belirtmek için kullanılır. Örnek: Yası 30 ile 40 arasındaki isçilerin kayıtlarını listelemek için

```
SELECT *
FROM ISCI
WHERE YAS BETWEEN 30 AND 40
```
ifadesi kullanılır. Bunu ayni zamanda aşağıdaki ifade ile de yapabilirsiniz. BETWEEN yazım kolaylığı sağlar.
```
SELECT *
FROM ISCI
WHERE YAS>=30 AND YAS<=40
```
SUM: Seçilen değerlerin toplamını bulur. İşçilerin aldığı toplam ücreti görmek için
```
SELECT SUM(UCRET)
FROM ISCI
```
ifadesi kullanılır.
MAX, MIN, AVG: Verilen değerin en büyüğünü, en küçüğünü ve ortalamasını bulur. 1999 yılında giren işçilerin en yüksek ücretinin, en düşük ücretinin ve ortalamasının ne kadar olduğunu öğrenmek istersek aşağıdaki ifadeyi kullanırız.
```
SELECT MAX(UCRET), MIN(UCRET), AVG(UCRET)
FROM ISCI
WHERE GIRIS_TARIHI>'01.01.1999'
```
MAX en büyük değeri, MIN en küçük değeri, AVG ise seçilen değerlerin ortalamasını bulur.
ORDER BY: Tablodan seçtiğimiz kayıtları sıralamak için kullanılır. Yukarıdaki örnekte isimleri alfabetik sıra ile görmek istersek
```
SELECT DISTINCT ISCI_ADI
FROM ISCI
ORDER BY ISCI_ADI
```
yazarız. Eğer sıralamayı tersine çevirmek istersek
```
SELECT DISTINCT ISCI_ADI
FROM ISCI
ORDER BY ISCI_ADI DESC
```
yazarız.
GROUP BY: Genelde istatistik amaçlar için kullanılır. Mesela hangi tarihte kaç işçinin ise alındığını bulmak için
```
SELECT GIRIS_TARIHI,COUNT(*)
FROM ISCI
GROUP BY GIRIS_TARIHI
```
yazmanız yeterli olacaktır. Bu ifade size gün bazında kaç işçinin ise alındığını gösterecektir. Eğer belli bir tarihten önce ye da sonrasını isterseniz veya sadece sayının 10'dan büyük olduğu günleri görmek isterseniz o zaman ifadeyi su şekilde yazmak gerekir
```
SELECT GIRIS_TARIHI,COUNT(*)
FROM ISCI
WHERE GIRIS_TARIHI>'01.01.1999'
GROUP BY GIRIS_TARIHI
HAVING COUNT(*)>10
```
HAVING, grup fonksiyonlarının kriterleri için kullanılır. SUM, COUNT vb. gibi.
UPDATE: Tabloda bulunan bir istediğiniz bir veya daha fazla alanın güncellenmesi amacıyla kullanılır. Mesela işçilerin maaşlarına % 20 zam yapıldığını düşünürsek aşağıdaki ifade ile bunu tabloda gerçekleştirebiliriz.

UPDATE ISCI
SET MAAS=MAAS*1.2
Eğer maaşlarla birlikte aldıkları primleri de %20 oranında artırmak isterseniz
UPDATE ISCI
SET MAAS=MAAS*1.2 , PRIM=PRIM*1.2
seklinde bir ifade kullanılır. Ayni zamanda WHERE ifadesini kullanarak sadece belli kayıtlar üzerinde güncelleme yapabilirsiniz.
DELETE: Tabloda bulunan kayıtları silmek için kullanılır. Eğer
DELETE FROM ISCI
derseniz tüm kayıtları gönderirsiniz. DELETE ifadesini kullanırken dikkatli olun. Buladada yine WHERE ifadesini kullanarak sadece belli kritere uyan kayıtların silinmesini sağlayabilirsiniz. Kötü bir örnek ama olsun, patron 45 yasından büyük isçileri isten attı (burası Türkiye, olmaz demeyin) ve kayıtlarının silinmesi isteniyor. O zaman
DELETE FROM ISCI
WHERE YAS>45
ifadesi kullanılır.
INSERT: Tablolara veri girişi yapmak amacıyla kullanılır.
INSERT INTO ISCI (ISCI_NO,ADI,SOYADI) VALUES (1000,'AHMET','SAVAS');
Eğer giriş yaparken tablonun bütün alanları kullanılacaksa alan isimlerini vermeye gerek yoktur.
Not: UPDATE, DELETE ve INSERT ifadelerini kullanırken dikkatli olmalısınız. Eğer SQL tabanlı bir veri tabanı kullanıyorsanız bu ifadeleri veritabanlarının kendi tool'lari üzerinde

kullanın. Çünkü Delphi ile gelen SQL Explorer'da işaretine basmayı unutursanız yaptığınız işlemin geri dönüsü olmayabilir. Yani en son yaptığınız işlemi Rollback yapamazsınız ve eğer gerçek veritabanı üzerinde yaptıysanız işlemi basınız bayağı ağrıyabilir veya o is yerinde yazdığınız son SQL ifadesi olabilir. :-))

LABEL

Label(leybil diye okunur) Component Palette'in Standart sayfasında bulunur. En çok kullanılan bileşenlerden biridir. Label bir metni ekran üzerinde göstermek için kullanılır. Label bileşeni penceresiz bir bileşendir. Yani başka bileşenleri içeremez ve bir tutamağı (handle) yoktur. Penceresiz bileşenler pencereli bileşenlere göre daha az sistem kaynağı tüketirler.
Özellikleri (Properties)
AlignColorDesignInfoNameTopAlignmentComObjectDragCursorOwnerTransparentAutoSizeComponentCountDragModeParentVCLComObjectBoundsRectComponentIndexEnabledParentColorVisibleCanvasComponentsFocusControlParentFontWidthCaptionComponentStateFontParentShowHintWindowProcClientHeightComponentStyleHeightPopupMenuWordWrapClientOriginControlStateHintShowAccelCharClientRectControlStyleLayoutShowHintClientWidthCursorLeftTag
Bu özelliklerin bazıları Object Inspector'da gözükürken bazıları gözükmez. Mesela Name,Visible gibi özellikler Object Inspector'da listelenir. Ancak mesela Owner özelliği Object Inspector'da listelenmez. Yani lafı şuraya getirmeye çalışıyoruz. Bir bileşenin Object Inspector'da listelenenden başka özellikleride olabilir. Bir bileşenin özelliklerinin tam listesini görmek için yardim dosyasına bakin.
Label'in özelliklerinden bazılarını açıklayalım:
Align:Label'in ebeveyn denetim kullanım alanına nasıl hizalanacağını gösterir. Caption:Label'in en önemli özelliği. Label'in göstereceği metin. Cursor:Fare imleci label üzerindeyken kullanılan fare imleci. Enabled:Label'in etkin veya kullanılamaz (gri renkte) olup olamadığını belirler.

Font:Label'da görüntülenen metnin yazı tipini belirler. Height:Label'in yüksekliği. Hint:Label üzerine gelindiğinde gösterilecek ipucu. İpucunun görünmesi için ShowHint seçeneği True olmalı. Left:Label'in sol üst kösesinin yatay koordinatı. Name:Label'in ismi. Kaynak kodunda buraya yazdığınız isim kullanılır. Owner:Yalnızca çalışma anında label'in sahibini gösterir. Parent:Yalnızca çalışma anında label'in ebeveynini gösterir. ParentColor:Label'in ebeveyni ile ayni rengi kullanıp kullanmayacağını gösterir. ParentCtl3D:Label'in ebeveyni ile ayni Ctl3D'yi kullanıp kullanmayacağını gösterir. ParentFont:Label'in ebeveyni ile ayni fontu kullanıp kullanmayacağını gösterir. ParentShowHint:Label'in ebeveyni ile ayni ShowHint'i kullanıp kullanmayacağını gösterir. PopupMenu:Label'a sağ tıkladığınızda gösterilen açılır. menü.ShowHint:İpuçlarının gösterilip gösterilmeyeceğini belirler.Tag:Özel ve tanımsız verilerin saklanması için kullanılabilecek bir uzun tamsayı.Top:Label'in sol üst kösesinin düşey koordinatı.Visible:Label'in görünür olup olamayacağını belirler.Width:Label'in yeni.**Olayları (Events)**

OnClickOnDragOverOnMouseMoveOnDblClickOnEndDragOnMouseUpOnDragDropOnMouseDownOnStartDrag

OnClick:Farenin sol tuşuyla label'a tıkladığınızda OnClick olayı gerçekleşir. OnDblClick:Fareyle label'a çift tıkladığınızda OnDblClick olayı gerçekleşir.OnDragDrop:Bir sürükleme bırakma operasyonu bilesen üzerinde sonlandığında, sürükleme operasyonunu alan bilesen tarafından gönderilir. OnDragOver:Kullanıcı fareyi label üzerinde sürüklediğinde.OnEndDrag:Sürükleme sonlandığında, sürüklemem operasyonunu başlatan bilesen tarafından gönderilir. OnMouseDown:Kullanıcı label üzerinde fare düğmelerinden birine bastığında gerçekleşir. OnMouseMove:Kullanıcı fareyi label üzerinde gezdirdiğinde gerçekleşir. OnMouseUp:Kullanıcı label üzerinde fare düğmelerinden birini bıraktığında gerçekleşir.OnStartDrag:Kullanıcı sürüklemeye başladığında, sürükleme operasyonunun menşesi olan bilesene gönderilir.

EDIT

Edit(edit diye okunur) Component Palette'in Standart sayfasında bulunur. En çok kullanılan bileşenlerden biridir.

Edit kullanıcının giriş yapması veya girilen metni ekran üzerinde göstermek için kullanılır. Edit bileşeni pencereli bir bileşendir.

Özellikleri (Properties)

AlignComObjectDragKindOEMConvertShowHintAnchorsComponentCountDragModeOwnerShowingAutoSelectComponentIndexEnabledParentTabOrderAutoSizeComponentsFontParentBiDiModeTabStopBiDiModeComponentStateHandleParentColorTagBorderStyleComponentStyleHeightParentCtl3DTextBoundsRectConstraintsHelpContextParentFontTopBrushControlCountHideSelectionParentShowHintVCLComObjectCanUndoControlsHintParentWindowVisibleCharCaseControlStateImeModePasswordCharWidthClientHeightControlStyleImeNamePopupMenuWindowProcClientOriginCtl3DLeftReadOnlyClientRectCursorMaxLengthSelLengthClientWidthDesignInfoModifiedSelStartColorDragCursorNameSelText

Edit'in en çok kullanılan özellikleri:

AutoSelect:True yaparsanız edit'e geçtiğiniz zaman içindeki metin seçili hale gelir. CharCaseEdit'e girilen metinin büyük harf olmasını (ecUpperCase) veya küçük harf olmasını (ecLowerCase) veya girldigi şekilde kalmasını (ecNormal) sağlar. Cursor:Fare imleci edit üzerindeyken kullanılan fare imleci. Font:Edit'teki metnin yazı tipini belirler. Height:Edit'in yüksekliği. Hint:Edit üzerine gelindiğinde gösterilecek ipucu. İpucunun görünmesi için ShowHint

seçeneği True olmalı. Left:Edit'in sol üst kösesinin yatay koordinatı. MaxLength:Edit'e girilebilecek maximum karakter şayisi. Bu değer 0 ise kullanıcı 255 karaktere kadar girebilir. Ancak bir sayı belirtirseniz maksimum o kadar karakter girebilir. Buraya 20 vermişseniz kullanıcı edit'e en fazla 20 karakter girebilir. Name:Edit'in ismi. Kaynak kodunda buraya yazdığınız isim kullanılır. PasswordChar:Eğer edit'i şifre girdirmek amacıyla kullanacaksanız buraya şifre karakterini girin (mesela *). Kullanıcının girdiği karakterler ekranda * seklinde gözükecek, ancak siz yine koddan normal şekilde girilen metni görebileceksiniz. PopupMenu:Edit'e sağ tıkladığınızda gösterilen açılır menü. Edit'e herhangi bir popup menü atamasanız bile Windows'tan dolayı bir popup menü gösterilir. ShowHint:İpuçlarının gösterilip gösterilmeyeceğini belirler.Tag:Özel ve tanımsız verilerin saklanması için kullanılabilecek bir uzun tamsayı.Text:Edit'in en önemli özelliği. Edit'e girilen metinle ilgili işlemleri bu özellik sayesine yaparsınız. Top:Edit'in sol üst kösesinin düşey koordinatı.Visible:Edit'in görünür olup olamayacağını belirler.Width:Edit'in eni.**Olayları (Events)**
OnChangeOnEndDockOnKeyPressOnStartDockOnClickOnEndDragOnKeyUpOnStartDragOnDblClickOnEnterOnMouseDownOnDragDropOnExitOnMouseMoveOnDragOverOnKeyDownOnMouseUp
OnClick:Farenin sol tuşuyla edit'e tıkladığınızda OnClick olayı gerçekleşir. OnDblClick:Fareyle edit'e çift tıkladığınızda OnDblClick olayı gerçekleşir.OnEnter:Başka bir kontrol'den edit'e geçtiğiniz zaman bu olay gerçekleşir. OnExit:Edit'ten başka bir kontrol'e geçtiğiniz zaman gerçekleşir. Edit'te en çok kullanılan olaydır. Girilen metnin sizin istediğiniz şekilde olup olmadığını kontrol edip, kullanıcı düzgün şekilde girene kadar başka bir kontrole gedmesini engelleyebilirsiniz. OnKeyPress:Edit'te karakterleri girerken her tuşa basılınca bu olay gerçekleşir. Bunu yine girilen metnin doğruluğunu kontrol için kullanabilirsiniz. Daha metnin girilişi sırasında yanlış karakter girilmesini önleyebilirsiniz.

DELPHİ PROGRAM ÖRNEKLERİ VE KODLARI :

TFileStream Kullanarak Dosya Kopyalama

```
Procedure FileCopy( Const sourcefilename, targetfilename: String );
Var
S, T: TFileStream;
Begin
S := TFileStream.Create( sourcefilename, fmOpenRead );
try
T := TFileStream.Create( targetfilename,
fmOpenWrite or fmCreate );
try
T.CopyFrom(S, S.Size ) ;
finally
T.Free;
end;
finally
S.Free;
end;
End;
```

Dosya Kopyalama (1)

```
procedure FileCopy(const FromFile, ToFile: string);
var
FromF, ToF: file;
NumRead, NumWritten: Word;
Buf: array[1..2048] of Char;
begin
AssignFile(FromF, FromFile);
Reset(FromF, 1); { Record size = 1 }
AssignFile(ToF, ToFile); { Open output file }
Rewrite(ToF, 1); { Record size = 1 }
repeat
BlockRead(FromF, Buf, SizeOf(Buf), NumRead);
BlockWrite(ToF, Buf, NumRead, NumWritten);
until (NumRead = 0) or (NumWritten <> NumRead);
CloseFile(FromF);
CloseFile(ToF);
end;
```

Dosya Kopyalama (2)

```
procedure CopyFile(FromFileName, ToFileName: string);
var
FromFile, ToFile: File;
begin
AssignFile(FromFile, FromFileName); { Assign FromFile to FromFileName }
AssignFile(ToFile, ToFileName); { Assign ToFile to ToFileName }
Reset(FromFile); { Open file for input }
try
Rewrite(ToFile); { Create file for output }
try
if LZCopy(TFileRec(FromFile).Handle, TFileRec(ToFile).Handle) < 0
then
raise EInOutError.Create('Error using LZCopy')
finally
CloseFile(ToFile); { Close ToFile }
end;
finally
CloseFile(FromFile); { Close FromFile }
end;
end;
```

Directory Adı değiştirme

SysUtils unitesinin içindeki RenameFile function bu isi görmektedir.

Treeview componentinin durumunu kaydetme ve gösterme

```
TreeView.SaveToFile('Dosya.adi');
TreeView.LoadFromFile('Dosya.adi');
```

Dosyayı yanlızca okumak (readonly) şeklinde açma

```
AssignFile(F, Dosya);
FileMode := 0; (read only modunda açmak}
Reset(F);
.
.
.
CloseFile(F);
```

Açılan dosyanın tarih ve zamanını ayarlamak

```
Var
f: file;
begin
Assign(f, DirInfo.Name);
Reset(f);
SetFTime(f, Time);
Close(f);
end;
```

Deltree

```
{$I-} {$I+}
procedure delTree (DirName: string);
var
FileSearch: SearchRec;
begin
chDir (DirName);
FindFirst ('*.*', Directory, FileSearch);
while (DosError = 0) do begin
if (FileSearch.name <> '.') AND (FileSearch.name <> '..') AND
( (FileSearch.attr AND Directory) <> 0)
then begin
if DirName[length(DirName)] = '\' then
delTree (DirName+FileSearch.Name)
else
delTree (DirName+'\'+FileSearch.Name);
ChDir (DirName);
end;
FindNext (FileSearch)
end;

FindFirst ('*.*', AnyFile, FileSearch);
while (DosError = 0) do begin
if (FileSearch.name <> '.') AND (FileSearch.name <> '..') then
Remove (workdir);
end;
FindNext (FileSearch)
```

```
end;
rmDir (DirName)
end;
```

Dbgrid'de (Ctrl-Del diyince) dosya silmesini engelleme

```
if (ssctrl in shift) and (key=vk_delete) then
begin
key:=0;
end;
```

String'i renk'e renk'i stringe çevirme

```
Uses graphics;
form1.Color:=stringtocolor('121');
label1.caption:= ColorToString(form1.color);
```

Mouse'un yerini degistirmek

```
randomize;
SetCursorPos(random(100),random(100));
```

ComboBox'in asagiya listelemesinin farkli bir yolu (DropComboBox)

```
SendMessage(ComboBox1.handle , 1039, 1, 0);
```

O anki sürücünün kapasitesini ve sürücüdeki bos yer miktarini bulmak;

```
DiskFree(0) //o anki sürücüdeki bos yer miktarini byte cinsinden döndürür.
DiskSize(0) //o anki sürücünün kapasitesini byte cinsinden döndürür.
DiskSize(0) div 1024 //o anki sürücünün kapasitesini KB cinsinden döndürür.
```

Bir menü ögesinin enabled özelligini false yapmak;

```
mainmenu1.items[0].items[1].enabled:=False;
```

Bir programın çalıştırılması

```
WinExec('c:\windows\calc.exe',sw_show);
WinExec('C:\WINDOWS\NOTEPAD.EXE C:\WINDOWS\WIN.INI', SW_SHOWNORMAL);
WinExec('COMMAND.COM', SW_SHOWNORMAL);
WinExec('COMMAND.COM /C DIR *.*', SW_SHOWNORMAL);
```

Listbox veya Combobox'ta seçili bir veya birden fazla öğeyi seçilmemiş duruma getirmek için;

```
Listbox1.itemindex:=-1;
```

Listbox, Combobox ve Memo'ya bir seferde ekleme yapmak;

```
Listbox1.items.SetText('aaa'#13'bbb'#13'ccc');
Memo1.Lines.SetText('aaa'#13'bbb'#13'ccc');
```

Harddiskin seri numarasının bulunması

```
procedure TForm1.Button1Click(Sender: TObject);
var
VolumeSerialNumber : DWORD;
MaximumComponentLength : DWORD;
FileSystemFlags : DWORD;
SerialNumber : string;
begin
GetVolumeInformation('C:\',
nil,
0,
@VolumeSerialNumber,
MaximumComponentLength,
FileSystemFlags,
nil,
0);
SerialNumber := IntToHex(HiWord(VolumeSerialNumber), 4) +
'-' +
IntToHex(LoWord(VolumeSerialNumber), 4);
Memo1.Lines.Add(SerialNumber);
end;
```

Bir string'in başındaki ve sonundaki boşlukları atmak için;

```
Trim(string)
TrimLeft (string) //stringin sadece basindaki bosluklari atmak için
TrimRight (string) //stringin sadece sonundaki bosluklari atmak için
```

Şifreli bir table için programın sifre istememesi için;

Table'in Active özelligini False yapin ve Form'un OnCreate olayina asagidaki kodu ekleyin

```
Session.AddPassword('sifre');
Table1.Active:=True;
```

Pencereyi minimize etmek;

```
Application.Minimize; //taskbar'a minimize
CloseWindow(handle)
WindowState := wsMinimized;
```

Windows'u kapatmak veya yeniden baslatmak(reboot);

```
var
i:dword;
begin
ExitWindowsEx(EWX_SHUTDOWN); //yeniden baslatmak için EWX_REBOOT
```

```
end;
```

Help menüsünden About kismina girin. Alt tusuna basili tutarak TEAM veya DEVELOPERS yazin. Delphi'yi gelistirenlerin isimlerini görebilirsiniz. Database Desktop'tada Help menüsünden About kismina girin ve delphi yazin.

ico'dan bmp'ye çevirme;

```
var
Icon : TIcon;
Bitmap : TBitmap;
begin
Icon := TIcon.Create;
Bitmap := TBitmap.Create;
Icon.LoadFromFile('c:\picture.ico');
Bitmap.Width := Icon.Width;
Bitmap.Height := Icon.Height;
Bitmap.Canvas.Draw(0, 0, Icon );
Bitmap.SaveToFile('c:\picture.bmp');
Icon.Free;
Bitmap.Free;
end;
```

CD-Rom sürücüyü açmak ve kapamak;

uses kismina MMSystem unitini ekleyin.

```
mciSendString('Set cdaudio door open wait', nil, 0, handle); //aç
mciSendString('Set cdaudio door closed wait', nil, 0, handle); //kapa
```

CapsLock ve Numlock tuslarini açip-kapama;

```
procedure TMyForm.Button1Click(Sender: TObject);
Var
KeyState : TKeyboardState;
begin
GetKeyboardState(KeyState);
if (KeyState[VK_CAPITAL] = 0) then
KeyState[VK_CAPITAL] := 1
else
KeyState[VK_CAPITAL] := 0;
SetKeyboardState(KeyState);
end;
```

Numlock tusu için VK_CAPITAL yerine VK_NUMLOCK yaziniz.

Menü'ye bitmap (resim) ekleme;

```
procedure TForm1.FormCreate(Sender: TObject);
var
```

```
Bmp1 : TPicture;
begin
Bmp1 := TPicture.Create;
Bmp1.LoadFromFile('c:\deneme\turkey.bmp');
SetMenuItemBitmaps( deneme1.Handle,
0,
MF_BYPOSITION,
Bmp1.Bitmap.Handle,
Bmp1.Bitmap.Handle);
end;
```

Alt + Tab ve Ctrl + Esc tuslarinin kullanilmaz hale getirilmesi;

```
var
OldVal : LongInt;
begin
SystemParametersInfo (97, Word (True), @OldVal, 0)
//Word(False) ile kullanirsaniz tuslari tekrar kullanabilirsiniz.
```

Windows ve System klasörlerinin bulunmasi

```
procedure TForm1.Button1Click(Sender: TObject);
var
a : Array[0..144] of char;
begin
GetWindowsDirectory(a, sizeof(a));
ShowMessage(StrPas(a));
GetSystemDirectory(a, sizeof(a));
ShowMessage(StrPas(a));    end;
```

Speakerdan Beep sesi çikartma

```
MessageBeep(word(-1));
```

Belgeler menüsüne bir dosya ekleme

uses kismina ShlOBJ unitini ekleyin;

```
procedure TForm1.Button1Click(Sender: TObject);
var
s : string;
begin
s := 'C:\DownLoad\deneme.html';
SHAddToRecentDocs(SHARD_PATH, pChar(s));
end;
```

Belgeler menüsünü temizleme

uses kismina ShlOBJ unitini ekleyin;

```
SHAddToRecentDocs(SHARD_PATH, nil);
```

Bir web adresini açma

uses kismina Shellapi unitini ekleyin;
ShellExecute(Handle,
'open',
'http://delphiworld.8m.com',
nil,
nil,
sw_ShowMaximized);

Bir DOS programini çalistirma ve çalismasi bitince penceresini kapatma

WinExec("command.com /c progdos.exe",sw_ShowNormal); //progdos.exe çalistiriliyor.
//eger ikinci paremetreyi sw_Hide yaparsaniz kullanici programin çalistigini görmez.

Uygulamanizin Görev Çubugundaki butonunu gizleme

Uygulamanizin Görev Çubugundaki butonunu gizlemek için programinizin ana formunun OnCreate olayina asagidaki kodu yazin;
SetWindowLong(Application.Handle,GWL_EXSTYLE, WS_EX_TOOLWINDOW);

Ekran koruyucusunu kapatmak ve açmak

```
//kapatmak için
SystemParametersInfo(SPI_SETSCREENSAVEACTIVE,
0,
nil,
0);
//açmak için
SystemParametersInfo(SPI_SETSCREENSAVEACTIVE,
1,
nil,
0);
```

Programın kapanmaması için
Formun OnCreate olayina;
KeyPreview := true;
Formun OnKeyDown olayina;

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

Bir string'in basindaki ve sonundaki bosluklari atmak için
Trim(string)
TrimLeft (string) //stringin sadece basindaki bosluklari atmak için
TrimRight (string) //stringin sadece sonundaki bosluklari atmak için

Listbox'a, Memo'ya ve Combobox'a bir seferde birden çok eleman eklemek

```
Listbox1.items.SetText('Ali'#13'Veli'#13'kirkdokuzelli');
Memo1.Lines.SetText('Ali'#13'Veli'#13'kirkdokuzelli');
Combobox1.items.SetText('Ali'#13'Veli'#13'kirkdokuzelli');
```
II. Yol : Mustafa Kilinç tarafindan gönderilmistir.
```
procedure TForm1.Button1Click(Sender: TObject);
var
gelen : TStringList;
begin
gelen := TStringList.Create;
gelen.sorted := True;
gelen.Duplicates := dupIgnore;
gelen.Add('evli');
gelen.Add('bekar');
gelen.Add('Dul');
ComboBox1.Items.Assign(gelen);
gelen.free;
end;
```

Memo içinde imlecin hangi satir ve kolonda oldugunu bulma
```
var
LineNum:logint;
CharsBeforeLine:logint;
begin
LineNum:=SendMessage(Memo1.Handle,EM_LINEFROMCHAR,Memo1.SelStart,0);
CharsBeforeLine:=SendMessage(Memo1.Handle,EM_LINEINDEX,LineNum,0);
Label1.Caption:='Satir'+IntToStr(LineNum+1);
Label2.Caption:='Kolon'+IntToStr((Memo1.SelStart-CharsBeforeLine)+1);
```

ListBox veya ComboBox'ta seçili bir veya birden fazla ögeyi seçilmemis hale getirme
```
ListBox1.ItemIndex:=-1;
Combobox1.ItemIndex:=-1;
```

Bir menü ögesini kullanilamaz hale getirmek
```
MainMenu1.Items[0].Items[1].Enabled:=False; //ilk menünün, ikinci elemani
```

Edit'e sadece sayi girilsin

Bir edit'e sadece istediginiz karakterlerin girilmesini saglayabilirsiniz. Bunun için Edit'in OnKeyPress olayina asagidaki kodu yazin.
```
if not (key in ['0'..'9',#8]) then
begin
Key:=#0; //girilen karakter rakam veya backspace degilse null(#0)'a dönüstür
Beep; //bip sesi ile kullaniciyi uyar.
end;
```
NOT: Kullanici Edit'e rakamlarin disinda karakter giremez, ancak Paste ile Edit'e bir metni kopyalayabilir. Bunuda Edit'in OnExit olayinda kontrol edip, girilen degerin istediginiz sekilde olup olmadigini kontrol edebilirsiniz.

Bir Popup menüyü kod ile gösterme

```
PopupMenu1.Popup(Form1.Left+60,Form1.Top+140);
```

Sistem tarihini ve saatini degistirmek

Sistemin tarihini ve saatini degistirmek için SetLocalTime fonksiyonunu kullanabilirsiniz.

```
var
  t:TSystemTime;
begin
  t.wYear:=1998;
  t.wMonth:=5;
  t.wDay:=23;
  t.wHour:=12;
  t.wMinute:=34;
  SetLocalTime(t);
end;
```

Sayilari virgüllerle yazmak

Bu is için FormatFloat fonksiyonunu kullanabilirsiniz. Sayi windows'unuz ayarina göre 12.345.678 veya 12,345,678 seklinde gösterilir.

```
procedure TForm1.Button1Click(Sender: TObject);
var
i : integer;
begin
i := 12345678;
Memo1.Lines.Add(FormatFloat('#,', i));
```

Edit'e girilen metnin ilk harfini büyük harf yapma

Bunun için Edit'in OnKeyPress olayina asagidaki kodu ekleyin.

```
with Sender as TEdit do
   if (SelStart = 0) or
      (Text[SelStart] = ' ') then
         if Key in ['a'..'z'] then
            Key := UpCase(Key);
```

Fareyi mesgul sekilde göstermek

Bir islem yaparken makinenin mesgul oldugunu göstermek için fareyi kum saati seklinde gösterip sonra eski haline getirmek için asagidaki gibi bir kod kullanabilirsiniz.

```
try
  Screen.Cursor := crHourGlass;
  {buraya kodunuzu yazin...}
finally
  Screen.Cursor := crDefault;
end;
Application.ProcessMessages;
```

Çok Satirli Ipucu

```
procedure TForm1.FormCreate(Sender: TObject);
```

```
begin
  SpeedButton1.Hint:='Çok satirli ipucunu '+chr(13)+
                     'mutlaka denemelisiniz '+chr(13)+
                     'çok güzel';
end;
```

Form'un arka kismina bir resmi dösemek

```
Bitmap: TBitmap;

procedure TForm1.FormCreate(Sender: TObject);
begin
  Bitmap := TBitmap.Create;
  Bitmap.LoadFromFile('C:\WINDOWS\cars.BMP');
end;

procedure TForm1.FormPaint(Sender: TObject);
var
  X, Y, W, H: LongInt;
begin
  with Bitmap do begin
    W := Width;
    H := Height;
  end;
  Y := 0;
  while Y < Height do begin
    X := 0;
    while X < Width do begin
      Canvas.Draw(X, Y, Bitmap);
      Inc(X, W);
    end;
    Inc(Y, H);
  end;
end;
```

Hareketli Imleç(animated cursor)leri kullanma

```
procedure TForm1.Button1Click(Sender:TObject);
var
  h : THandle;
begin
  h := LoadImage(0,
              'C:\TheWall\Magic.ani',
              IMAGE_CURSOR,
              0,
              0,
              LR_DEFAULTSIZE or
              LR_LOADFROMFILE);
  if h = 0 then ShowMessage('Cursor not loaded') else begin
```

```
    Screen.Cursors[1] := h;
    Form1.Cursor := 1;
  end;
end;
```

Sürücünün kapasitesini ve sürücüdeki bos yer miktarini bulmak

DiskFree(0) //o anki sürücüdeki bos yer miktarini byte cinsinden döndürür.
DiskSize(0) //o anki sürücünün kapasitesini byte cinsinden döndürür.
DiskSize(0) div 1024 //o anki sürücünün kapasitesini KB cinsinden döndürür.

Bir form üzerindeki tüm bilesenleri read only(salt okunur) yapma

uses kismina typinfo unitini ekleyin.

```
procedure TForm1.SetReadOnly( Value : boolean ) ;
var
PropInfo : PPropInfo ;
Component : TComponent ;
i : integer ;
begin
for i := 0 to ComponentCount - 1 do begin
Component := Components[ i ] ;
if Component is TControl then begin
PropInfo := GetPropInfo( Component.ClassInfo, 'ReadOnly' ) ;
if Assigned( PropInfo ) and
( PropInfo^.PropType^.Kind = tkEnumeration ) then
SetOrdProp( Component, PropInfo, integer( Value ) ) ;
end ;
end ;
end ;
procedure TForm1.Button1Click(Sender: TObject);
begin
SetReadOnly( true ) ;
end;
```

Dikdörtgen olmayan Edit'ler

Degisik sekilde bir edit elde etmek için formun OnCreate olayina asagidaki kodu yazin.

```
SetWindowRgn(Edit1.handle,
CreateRoundRectRgn(2,2,Edit1.Width-2,Edit1.Height-2,15,15),
True);
```

Bir klasörün boyutunu ögrenmek

Bir klasördeki dosyalarin kaç byte yer kapladigini ögrenmek için

```
function TForm1.GetDirectorySize(const ADirectory: string): Integer;
var
Dir: TSearchRec;
Ret: integer;
Path: string;
begin
```

```
Result := 0;
Path := ExtractFilePath(ADirectory);
Ret := Sysutils.FindFirst(ADirectory, faAnyFile, Dir);

if Ret <> NO_ERROR then
exit;

try
while ret=NO_ERROR do
begin
inc(Result, Dir.Size);
if (Dir.Attr in [faDirectory]) and (Dir.Name[1] <> '.') then
Inc(Result, GetDirectorySize(Path + Dir.Name + '\*.*'));
Ret := Sysutils.FindNext(Dir);
end;
finally
Sysutils.FindClose(Dir);
end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
Showmessage(intToStr(getdirectorysize('C:\windows')));
end;
```

Bir dosyanin özelliklerini gösterme

```
procedure TForm1.Button1Click(Sender: TObject);
var
sei : TShellExecuteInfo;
begin
FillChar(sei,SizeOf(sei),#0);
sei.cbSize:=SizeOf(sei);
sei.lpFile:=PChar('c:\windows\notepad.exe');
sei.lpVerb:='properties';
sei.fMask:=SEE_MASK_INVOKEIDLIST;
ShellExecuteEx(@sei);
end;
```

Programim hangi klasörde çalisiyor

I.Yol :

```
procedure TForm1.Button1Click(Sender: TObject);
var
path: string;
begin
Path := ExtractFilePath(ParamStr(0));
Showmessage (path);
end;
```

II.Yol :

```
procedure TForm1.Button1Click(Sender: TObject);
begin
MessageDlg('Programiniz ' +ExtractFilePath( Application.ExeName ) + 'klasöründe çalisiyor. ',
mtInformation, [mbOk], 0 );
end;
```

Bir dosyayi geri dönüsüm kutusuna (recyle bin) atmak

uses kismina shellapi unitini ekleyin.

```
procedure TForm1.Button1Click(Sender: TObject);
var
T : TSHFileOpStruct;
begin
FillChar(T,SizeOf(TSHFileOpStruct),#0);
with T do
begin
Wnd:=0;
wFunc:=FO_DELETE;
pFrom:=Pchar('c:\test\2.avi');
fFlags:=FOF_ALLOWUNDO;
end;
SHFileOperation(T);
end;
```

Bir dosyanin boyutunu bulmak

```
procedure TForm1.Button1Click(Sender: TObject);
var
srFileSR: TSearchRec;
sFileName, sFileSize: string;
begin
sFileName := 'c:\test\2.avi';
FindFirst(sFileName,faAnyFile,srFileSR);
sFileSize := IntToStr(srFileSR.Size);
Showmessage(sFileSize);
end;
```

Bmp dosyasini JPEG'e dönüstürme

uses kismina jpeg unitini ekleyin.

```
procedure TForm1.Button1Click(Sender: TObject);
var
MyJPEG : TJPEGImage;
MyBMP : TBitmap;
begin
MyBMP := TBitmap.Create;
with MyBMP do
try
LoadFromFile('c:\winnt\ACD Wallpaper.bmp');
MyJPEG := TJPEGImage.Create;
```

```
with MyJPEG do begin
Assign(MyBMP);
SaveToFile('c:\winnt\ACD Wallpaper.JPEG');
Free;
end;
finally
Free;
end;
end;
```

Sayi Yuvarlama

24.499999 gibi bir sayiyi 24.5'e asagidaki kodla yuvarlayabilirsiniz.

```
procedure TForm1.Button1Click(Sender: TObject);
var
getal : real ;
AfgerondGetal :real;
begin
Getal:=24.499999;
AfgerondGetal:=round(Getal*100)/100;
Edit1.Text:=floattostr(AfgerondGetal);
end;
```

Alt+F4 tus kombinasyonuyla programin kapanmamasi için

Formun OnCreate olayina;

```
KeyPreview := true;
```

Formun OnKeyDown olayina;

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

Hareketli Imleç(animated cursor)leri kullanma

```
procedure TForm1.Button1Click(Sender:TObject);
var
h : THandle;
begin
h := LoadImage(0,
'C:\TheWall\Magic.ani',
IMAGE_CURSOR,
0,
0,
LR_DEFAULTSIZE or
LR_LOADFROMFILE);
if h = 0 then ShowMessage('Cursor not loaded') else begin
Screen.Cursors[1] := h;
Form1.Cursor := 1;
end;
```

```
end;
```

Windows lisans bilgilerinin (isim ve sirket) bulunmasi

uses kismina Registry unitini ekleyin;

```
procedure TForm1.Button1Click(Sender:TObject);
var
reg: TRegIniFile;
begin
reg := TRegIniFile.create('SOFTWARE\MICROSOFT\MS SETUP (ACME)\');
Memo1.Lines.Add(reg.ReadString('USER INFO',
'DefName',
'Mustafa SIMSEK'));
Memo1.Lines.Add(reg.ReadString('USER INFO',
'DefCompany',
'Bilgisayar Bilimleri Müh.'));
reg.free;
end;
```

Çok Satirli Ipucu

```
procedure TForm1.FormCreate(Sender: TObject);
begin
SpeedButton1.Hint:='Çok satirli ipucunu '+chr(13)+
'mutlaka denemelisiniz '+chr(13)+
'çok güzel';
end;
```

Edit'e girilen metnin ilk harfini büyük harf yapma

Form'a bir Edit componenti yerlestirin ve OnKeyPress olayina asagidaki kodu ekleyin.

```
with Sender as TEdit do
if (SelStart = 0) or
(Text[SelStart] = ' ') then
if Key in ['a'..'z'] then
Key := UpCase(Key);
```

Bir klasörü ve onun altindaki tüm dosyalari ve klasörleri silme

Ancak salt okunur (read only) özelligi olan ve kullanimda olan dosyalar silinmez.

```
procedure TForm1.Button1Click(Sender: TObject);
var
DirInfo: TSearchRec;
r : Integer;
begin
r := FindFirst('C:\Download\Test\*.*', FaAnyfile, DirInfo);
while r = 0 do begin
```

```
if ((DirInfo.Attr and FaDirectory <> FaDirectory) and
(DirInfo.Attr and FaVolumeId <> FaVolumeID)) then
if DeleteFile(pChar('C:\Download\test\' + DirInfo.Name))
= false then
ShowMessage('C:\Download\test\'+DirInfo.Name+' silinemiyor!!!');
r := FindNext(DirInfo);
end;
SysUtils.FindClose(DirInfo);
if RemoveDirectory('C:\Download\Test') = false then
ShowMessage('C:\Download\test klasörü silinemiyor!!!');
end;
```

Baslat butonunu gizlemek veya kullanilmaz hale getirmek

```
procedure TForm1.Button1Click(Sender: TObject);
var
Rgn : hRgn;
begin
// Baslat butonunu gizle
Rgn := CreateRectRgn(0, 0, 0, 0);
SetWindowRgn(FindWindowEx(FindWindow('Shell_TrayWnd', nil),
0,
'Button',
nil),
Rgn,
true);
end;
procedure TForm1.Button2Click(Sender: TObject);
begin
```

Gizlenen Baslat butonunu eski haline döndürmek için

```
SetWindowRgn(FindWindowEx(FindWindow('Shell_TrayWnd', nil),
0,
'Button',
nil),
0,
true);
end;
procedure TForm1.Button3Click(Sender: TObject);
begin
//Baslat butonunu kullanilmaz yap
EnableWindow(FindWindowEx(FindWindow('Shell_TrayWnd', nil),
0,
'Button',
nil),
false);
end;
```

```
procedure TForm1.Button4Click(Sender: TObject);
begin
//Kullanilmaz yapilan Baslat butonunu eski haline getirmek için
EnableWindow(FindWindowEx(FindWindow('Shell_TrayWnd', nil),
0,
'Button',
nil),
true);
end;
```

Windows Gezginini istediginiz bir klasörle açma

uses kismina Shellapi unitini ekleyin.

```
ShellExecute(0,
'explore',
'C:\WINDOWS', //açmak istediginiz dizin
nil,
nil,
SW_SHOWNORMAL);
```

Duvar kagidini degistirmek

```
var
s: string;
begin
s := 'c:\windows\athena.bmp';
SystemParametersInfo(SPI_SETDESKWALLPAPER, 0, PChar(s),0);
```

Form'un arka kismina bir resmi dösemek

```
Bitmap: TBitmap;
procedure TForm1.FormCreate(Sender: TObject);
begin
Bitmap := TBitmap.Create;
Bitmap.LoadFromFile('C:\WINDOWS\cars.BMP');
end;
procedure TForm1.FormPaint(Sender: TObject);
var
X, Y, W, H: LongInt;
begin
with Bitmap do begin
W := Width;
H := Height;
end;
Y := 0;
while Y < Height do begin
X := 0;
while X < Width do begin
```

```
Canvas.Draw(X, Y, Bitmap);
Inc(X, W);
end;
Inc(Y, H);
end;
end;
```

Bir Denetim Masasi uygulamasini çalistirmak

Control Panel uygulamalari Windows\System klasörü altinda bulunur. *.CPL uzantili dosyalardir. Bu uygulamalari Control.Exe programi ile çalistirabilirsiniz. Bazi Control Panel uygulamalari Windows\System klasöründe bulunmaz. Bunlarin ismini vererek çalistirabilirsiniz.

```
WinExec('C:\WINDOWS\CONTROL.EXE TIMEDATE.CPL', sw_ShowNormal);
WinExec('C:\WINDOWS\CONTROL.EXE MOUSE', sw_ShowNormal);
WinExec('C:\WINDOWS\CONTROL.EXE PRINTERS', sw_ShowNormal);
```

Sayilari virgüllerle yazmak

Sayi windows'unuz ayarina göre 12.345.678 veya 12,345,678 seklinde gösterilir.

```
procedure TForm1.Button1Click(Sender: TObject);
var
i : integer;
begin
i := 12345678;
Memo1.Lines.Add(FormatFloat('#,', i));
```

Sistem Tarihini ve Saatini Degistirmek

Sistemin tarihini ve saatini degistirmek için SetLocalTime fonksiyonunu kullanabilirsiniz.

```
var
t:TSystemTime;
begin
t.wYear:=1998;
t.wMonth:=5;
t.wDay:=23;
t.wHour:=12;
t.wMinute:=34;
SetLocalTime(t);
end;
```

Fareyi mesgul sekilde göstermek

```
try
Screen.Cursor := crHourGlass;
{buraya kodunuzu yazin...}
finally
Screen.Cursor := crDefault;
```

```
end;
Application.ProcessMessages;
```

Ekran Görüntüsünü Aktarma

Belirttiginiz sinirlar dahilinde ekranin belli bir alanini formunuzun üzerine koymak isterseniz. Formunuza image1 adli bir resim objesi ekleyin ve daha sonra formunuzun create olayina su kodu yazin.

```
procedure TForm1.FormCreate(Sender: TObject);
var
DCDesk: HDC;
begin
DCDesk:=GetWindowDC(GetDesktopWindow);
BitBlt(Image1.Canvas.Handle, 0, 0, Screen.Width, Screen.Height,DCDesk, 0, 0,SRCCOPY);
ReleaseDC(GetDesktopWindow, DCDesk);
end;
```

Enter Tusuna Basilmis gibi Gösterme

Windows programlarinda bir alttaki alana geçmek için TAB tusu kullanilir.Ancak DOS programlarindan gelen aliskanlikla kullanicilar hep Enter ile alt alana geçmek ister ve bu bir tik olmustur.

Delphide Enter tusu ile bir alt alana geçmek için bir yöntem;

· Formun Keypreview olayini True yapilir.

· Form üzerinde herhangiki tüm bilesenlere Default false yaplir.

· formun onKeypres olayina asagidaki function ilave edilir.

```
procedure TAdresformu.FormKeyPress(Sender: TObject; var Key: Char);
begin
if Key = #13 then begin
Key := #0;
if (Sender is TDBGrid) then
TDBGrid(Sender).Perform(WM_KeyDown,VK_Tab,0)
else
Perform(Wm_NextDlgCtl,0,0);
end;
```

Geometrik Formlar Olusturma

Formumuzun OnShow Eventine assagidaki kodu yaziyoruz.

```
procedure TForm1.FormShow(Sender: TObject);
var
regionhandle:integer;
area:array[0..2] of tpoint;
begin
area[0].x := 0; area[0].y := 0;
area[1].x := 400; area[1].y := 0;
area[2].x := 200; area[2].y := 200;
regionhandle:=CreatePolygonRgn(area,3,ALTERNATE); // 3 polygonda kaç tane nokta
oldugunu belirtir
```

```
// area ise polygon koordinatlarinin bulundugu dizi.
setwindowrgn(form1.handle,RegionHandle,true);
end;
```
Area dizisinde verilen x,y koordinatlarina göre polgon hesaplanir. Hesaplanan Handle ile herhangi bir form'a bu polyon sekli verilebilir. Polgon disinda kalan grafikler yarim veya hiç gözükmez.

Imlecin o anda ekranin neresinde oldugunu bulan ufak bir kod parçasi.

```
procedure TForm1.Button1Click(Sender: TObject);
var Yer:TPoint;
begin
if Assigned(ActiveControl) then
begin
Yer:=Point(0,0); { burda 0,0 imleç'in ekrandaki yeri oluyor }
ActiveControl.ClientToScreen(Yer);
SetCursorPos(Yer.X,Yer.Y);
end;
end;
and ws_Caption)<>ws
```

Istenilen alanlari Combobox'a yazdirma...

```
procedure TForm1.FormCreate(Sender: TObject);
begin
with Table1 do begin
DisableControls;
try
First
while not EOF do begin
with MyCombo.Items do
Objects[Add(FieldByName('Terms').AsString)] := Pointer((FieldByName('EmpNo').AsInteger));
Next;
end;
finally
EnableControls;
end;
end;
end;

procedure TForm1.MyComboClick(Sender: TObject);
var EmpNo: Integer;
begin
with MyCombo do
EmpNo:=LongInt(Items.Objects[ ItemIndex ]);
ShowMessage('Emp.No.: ' + IntToStr(EmpNo));
End;
```

Query'de SQL kullanarak arama yapmak...

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Query1.Close;
Query1.Sql.Clear;
Query1.Sql.Add('Select * From Database Where Adi like "'+Edit1.text+'%"');
Query1.Open;
End;
```

Table'da istenilen alana göre arama yapmak

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Table1.Locate('AlanAdi',Edit1.Text,[]);
end;
```

Table'da her tusa basista girilen kadarinin uydugu kayidi bulma...

```
procedure TForm1.Edit1Change(Sender: TObject);
begin
Table1.FindNearst([Edit1.Text]);
end;
```

Table'da indexli alana göre arama yapmak...

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Table1.FindKey([Edit1.text]);
end;
```