**BAŞTAN SONA SQL VERİTABANI OLUŞTURMAK**

Teknolojinin neredeyse son safhalarda kullanıldığı günümüzde bir çok eşyamız lüks olma modelinden uzaklaştı. İlk çıktığı zamanlar cep telefonları insanlar için lüks bir eşya gibi gelebiliyordu. Fakat günümüze baktığımızda cep telefonların bile artık lükslüğünü yitirdiğini ve aşırı gereksinimlerden dolayı neredeyse normal bir eşya statüsünü aldı. Tabi ki bu gelişmeler tek cep telefonlarında yaşanmadı. Günümüzde belki de halen lüks sayılan ama özellikle yazılımcı ve gününün yarısını bilgisayarda geçiren kullanıcılar için, bilgisayar lüks olma statüsünden çıkmıştır. Bilgisayarlarımızın en önemli özelliği istediğimiz an istediğimiz verilere ulaşabilmemiz çok büyük ansiklopediler tutabilecek bilgileri veya arşivleri depolayabilmesi ve bize neredeyse sınırsız imkanlar tanıyıp günlük hayatımızda ki işlemlerimizi kolaylaştırması. Günümüzde bir firmaya baktığınızda mutlaka günlük, aylık ve yıllık bilgilerini tutması ve değerlendirmesi gerektiğini görüyoruz. Eskilerden bu işlemler belirli kağıtlarla, belgelerle yapıldığı için denetleme zor olabiliyordu ama artık bilgisayarların yardımıyla bu tür veriler kolayca depolanabiliyor, hesaplanabiliyor ve yorumlanabiliyor.

Bu verileri bilgisayarda tutmak için bir çok yöntem geliştirilmiştir. En basit ve en popüler yöntem olan veritabanına saklama olaylarını sizlere şöyle açıklamak isterim.

Günümüzde pek çok firma, verilerini hep veritabanlarında saklamayı uygun görür bunun nedeni Veritabanlarının daha çok sistematik yapıda oluşması ve daha sistemli oluşturulabilmesidir, extra dan kullanım kolaylığı diyebiliriz. Veritabanları günümüzde neredeyse her yerde kullanılıyor. Günümüzde Bir web sitesine baktığınızda bile o sitede %90 oranlarında bir veritabanının olduğunu ve belirli şekilde bilgilerin tutulduğunu fark edecek siniz.

Veritabanları denildiğinde günümüzdeki en popülerlerini saymak gerekirse Microsoft SQL Server, Oracle diyebiliriz. Kurulacak veritabanı küçük veya orta ölçekli iş yerleri için kuruluyorsa genellikle tercih edilen Microsoft Access veritabanı veya daha benzer veritabanları olur (tercihe göre). Ben bu makalemde sizlere Microsoft SQL Server programından bahsedeceğim. Öncelikle SQL Serverin ne olduğunu ve neden SQL Serveri tercih ettiğimi tanımlasak daha iyi olacağı kanısındayım.

Microsoft SQL Server, adından da anlaşılacağı gibi microsoft firmasının çıkardığı bir veritabanı sistem programıdır. SQL Serverin en büyük avantajı, bir veritabanı sistemi olması, ip adresi bazlı çalışıyor olabilmesi ve veritabanına neredeyse her yerden ulaşılabilir olması ayrıca veritabanı büyüklüğü sürümüne göre 1000GB büyüklüğüne kadar çıkabilmesidir.

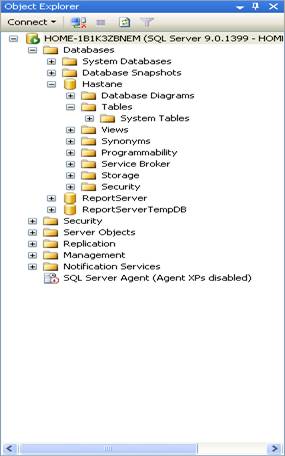
SQL Serverin günümüzde kullanılan pek çok sürümü bulunmaktadır. Office cd sini aldığınızda yanında ücretsiz sürümü olan MSDN veya genellikle yazılımcıların tercih ettiği Microsoft SQL Server Enterprise veya daha alt sürümleri kullanılmaktadır. SQL Serverin en büyük avantajlarından birisi,

Bir veritabanı sistemi olduğundan ve ip bazlı çalışabildiğinden aynı anda bir çok makinenin sorgu gönderebilmesidir ve hızlı çalışabilmesidir. SQL Serverin gelişmiş sürümleri daha öncedende söylediğim gibi yaklaşık 1000GB yakın veriyi depolayabilecek kapasiteye sahiptir. Günümüzdede neredeyse her programlama dilinin altyapısında microsoft sql servere destek sağlanmaktadır. Bunun anlamı hangi dille çalışıyorsanız çalışın sql servere anında bağlantı kurabilir ve istediğiniz herşeyi yapabilirsiniz. Özellikle yeni nesil Microsoftun çıkarmış olduğu Visual Studio 2005 ve alt sürümleri sadece SQL Servere ait olan kütüphaneler geliştirmiş ve performansı daha da arttırmışlardır.

Temel bilgilerimizi verdiğimize göre sizinle sıfırdan bir veritabanı oluşturmak istiyorum. Oluşturacağımız veritabanı küçük bir hastane üzerine olsun. Hastanenin hastaları, doktorları olsun ve bunların kayıtları tutulsun. Extradan da Muayene tablosu olsun. Veritabanı sistemi olarak SQL Server 2005 sürümünü tercih ettim. SQL Serverime Windows Authentication üzerinden bağlanıyorum. Bilmeyen arkadaşlarım için Authentication terimini açıklamak istiyorum. Kısaca SQL Servere bağlantı kurmamız için tanımlanan yolun yapısıdır. Ben kendi bilgisayarım üzerinden bağlantı kuracağım için Windows Authentication aracılığıyla bağlantı kuruyorum. Eğer bir çok makineyle iş yapmayı düşünseydim SQL Authentication alt yapısını kullanacaktım. Bu yapıyı kullanarak bana bir kullanıcı adı ve şifre belirecekti. Bu bilgiler doğrultusunda veritabanıma ağ bağlantısında tüm bilgisayarlardan bağlantı kurup istediğim verileri çekebilecektim. Veritabanıma Windows Authentication yapısıyla bağlanıyorum. Kolaylık olması açısında resimi aşağıdaki gibi olacaktır.



Resimde gördüğünüz gibi kişisel bilgisayarımda Sunucumun adı, eğer makinanız da sql server düzgün olarak kurulmuşsa connect butonuna bastığınızda sql servere bağlantı kurulacaktır. Bağlantı kurduktan sonra yapmamız gereken ilk iş herhangi bir veritabanı olmadığından ötürü bağlantısını kurduğumuz SQL Sunucusunda bir veritabanı oluşturmak. Oluşturmak istediğim veritabanı bir hastanenin veritabanı olmasından ötürü ismini Hastane olarak kaydediyorum. Seçmekte özgürsünüz :) Veritabanımızı oluşturduğumuzda aşağıdaki gibi bir resim bilgisayarınızda belirecektir.

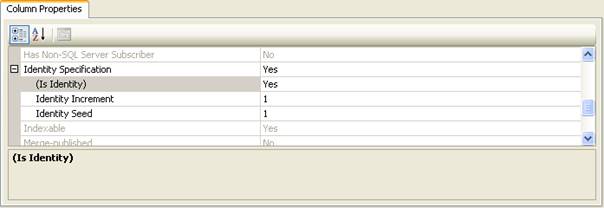


Hastane adlı veritabanımızı açmış olduk sıra geldi konumuza göre tablolar oluşturmaya.

Hastaneye gittiğimizde ilk dikkatinizi çeken görevli bir personelin sizi ilk olarak kayıt merkezine yönlendireceği olmasıdır. Kayıt merkezinde adınız, soyadınız,doğum tarihi ve benzeri bilgiler sorulacak ve böylece adınıza ait bir kayıd oluşturulacak. Muayene esnasında da sizin var olan kayıdınızdan yola çıkılarak hastanede o zamandan gelecek zamanlara, o hastanede ne işlem ne muayene yaptıysanız veritabanında gözükecektir. Böylece bilgilere veritabanından sorgulanarak ulaşılacaktır. Veritabanımızda HASTALAR adlı yeni bir tablo oluşturalım. Tablomuzu oluştururken bazı hususlara dikkat etmemiz gerekiyor aksi takdirde ileride muhtemel veritabanı problemleri yaşayabiliriz. Bir tablo oluştururken dikkat etmemiz gereken ilk husus ihtiyaçlarımıza yönelik veri tipleriyle ayarlamak ve tablomuza baş anahtar (primary key) yada foreign key belirtmek. Baş Anahtar, tablolarımızda veri bütünlüğünün sağlanmasında gereken en önemli kısıtlayıcı dersek sanırım yanlış olmaz. Bir tabloda baş anahtar neden tanımlarız şöyle açıklayalım.

Günün birinde hastaneye bir hasta geldi diyelim. Adı ragıp elibol olsun. Bu hasta veritabanına kaydedildi herhangi bir problem söz konusu değil. Ertesi gün tesadüfler çoğu zaman olur aynı isim ve soyisimde olan fakat farklı olan ikinci bir ragıp elibol geldi. Bunuda veritabanına kaydettik. Şimdi size şöyle bir soru sormalıyım. Ragıp elibol adlı kişinin bana 1 ay içerisinde yapmış olduğu tüm muayeneleri getir. Böyle bir sorguya ne yapardınız :) ya iki ragıp elibol kişisinide getirip aile ve benzeri bilgilerden kimin kim olduğuna bakacaksınız (uzun iş) yada allah kolaylık versin başka yollar düşüneceksiniz. Bilgisayar extradan farklı yollar düşünemeyeceği için size ilk gördüğü ragıp elibol adlı vatandaşı getirecektir. Ama işin gerçeği muayenesi istenen kişi ikinci kişidir böylece veriler karışacaktır :) Böyle bir durumda baş anahtar bize yardımcı oluyor. Baş anahtar eğer numeric olarak atanırsa oto identity özelliği arttırılarak yeni gelen kayıtların nolarını birer birer arttırırız böylece verilerde karışıklık olmaz. Sorgu yapılmak istendiği anda. HastaNosu şu olan ragıp elibolu bana getir ve artık iş bitti. O an bilgisayar tarafından atanan sadece 1 numaradır ve o numara bir başka kişiye devredilemeyecek böylece veri çakışması durumu olmayacak. Diyelimki veritabanımızda bulunan kolon numeric değilde string yani alfabetik bir katakter türünde olsun. Böyle bir durumda o kolonu baş anahtar seçtiğinizde o kolona yazılan kişinin aynısından bir daha kaydedilemeyecek. Kaydettiğiniz anda SQL sizi uyaracak ve hata yaptığınızı söyleyecek. :) Baş anahtar seçmenin daha bir çok avantajı bulunmaktadır ilerleyen açıklamalarda bunu kendiniz farkedeceksiniz. Foreign keyden şuan bahsetmenin yanlış olacağını düşünüyorum. Veritabanlarımızı oluşturup ilişkilendirdiğimizde detaylı olarak göreceğiz.

HASTALAR adlı veritabanımızın ilk kolonuna bir baş anahtar olarak tanımlayacağımız bir isim veriyoruz. Ben HastaNo adını verdim. HastaNo adlı kolonumuzun veri türü int tipinde olacaktır. Burada dikkat etmeniz gereken husus HastaNo adlı kolonumuzun daha sonradan otomatik artması için Identity Specification özelliğini yes yapıp identity increment seçeneğinide 1 1 şeklinden yapmalıyız bu durumda kolonumuz yeni kayıtlar eklendiğinde bir bir artacak böylece veri bütünlüğü sağlanmış olacak.



İkinci kolonumuza Adi veri tipi NVARCHAR(30), üçüncü kolonumuza Soyadi NVARCHAR(30), dördüncüye DogumTarihi veri tipine DATETIME, beşinciye Cinsiyet NVARCHAR(7), altıncıya Adres NVARCHAR(100) ve sonuncuyada Tel NVARCHAR(20) diyelim ve son olarakta HASTALAR adıyla kaydedelim.

Şimdi ikinci tablomuzu oluşturalım. İkinci tablomuz neredeyse birinci tabloyla aynı olacak adı DOKTORLAR tablosu olsun. Diğer tabloyla tamamen aynı değerleri kuralım, sadece fazladan en alt kolon olarak Uzmanlik NVARCHAR(50) kolonunu ekleyelim. Tüm bu işlemlerimiz olduğunda karşımıza şu diyagram çıkacaktır.



Sanırım veritabanımız yavaş yavaş oluşmaya başlıyor. Hastalarımızı ve doktorlarımızı kaydedeceğimiz tablolar oluştu. İsterseniz muayene tablomuza başlayalım.

Tablomuzun ilk kolonuna MuayeneNo veri türü olarakta INT veri türünü verelim ve kesinlikle bu kolonu baş anahtar olarak seçmeyi unutmayalım. İkinci gireceğimiz kolona HastaNo veri türüne INT girelim. Fakat bu kolonumuzu baş anahtarlardan seçmiyoruz. Bunun nedenini tablomuzu bitirdiğimizde açıklayacağım. Üçüncü kolonumuzada HastaNo diyelim veri tipi olarakta aynı şekil INT veri türünü ayarlayalım. Geriye kaldı muayenemizin kayıt tarihi. Tarih adıyla kolonumuzu girelim veri tipi olarakta DATETIME veri türünü ayarlayalım. Yeri gelmişken burada bir uyarı yapmak istiyorum. Tarih kolonlarında mümkün olduğunda date veya benzeri veri tiplerini girin, text veri tiplerinden kullanacak olursanız bu size ileride extradan kod yazmanıza veya uğraşmanıza neden olabilir. Kolonlarımızı oluşturduktan sonra tablomuzu MUAYENELER olarak kaydedelim. Şimdi neden HastaNo ve DoktorNo sunu baş anahtar olarak ayarlamadık isterseniz bunlara bir açıklık getirelim. Bir hastaneye, bir hasta bir yılda istediği kadar gelebilir fakat o hastanın şahsından dünyada sadece bir tane vardır. Sanırım biraz karışık oldu :). Eğer baş anahtarlardan olarak bu ikisini seçseydik daha sonra aynı hasta ikinci bir muayeneyi olmak isterse aynı hasta olduğundan ötürü veritabanımız hata verecekti ve bize engel olacaktı. Bunun nedeni size daha öncedende bahsettiğim baş anahtar (Primary key) özelliği bu sayede o kişiye ait sadece bir veri bulunabilir. Fakat biz burada o hastanın bir çok kez muayene olabileceğinibildiğimiz için bunu foreign key yane referans anahtar tipi olarak belirleyeceğiz. Böylece o hasta isterse on bin kerede gelse veritabanında herhangi bir hata olmayacak. Fakat dikkat etmenizi istediğim husus MuayeneNo bölümünü baş anahtar olarak seçmemiz. Bir hasta muayene olurken o muayene tekrarlamayacağı için bu kolonu mutlaka baş anahtar olarak seçmeliyiz. Çünkü aynı muayenenosu veritabanına eklenirse daha sonra o muayenenin detaylarına ulaşamayız buda bize büyük bir problem çıkartır. Tablomuzu oluşturduktan sonra sıra geldi MUAYENEDETAY tablomuzun oluşturulmasına. Bu nedir diyebilirsiniz. Bu sorunun açıklamasına şöyle bir açıklama yapabilirim. Bir hasta, hastaneye ilk geldiğinde öncelikle randevu ve randevu tarihi alır. Herhangi bir muayene olmadığından ötürü bir karar yoktur. Bu bilgileri MUAYENELER tablosuna kaydederiz. Hastamızın tarihi geldi ve hastamız tam zamanında hastanede bulundu. Doktor muayenesini yaptı ve hastanın teşhisini koydu. Devreye MUAYENEDETAY tablosu giriyor. Burada ÇOK DİKKATLE takip etmenizi istiyorum.

Tablomuzu oluştururken ilk kolonumuzun adına MuayeneNo diyelim veri tipi INT türünden olmalı. İkinci kolonumuzada Karar olarak girelim Veri tipi NVARCHAR(50) diyelim. Tablomuzu kaydedelim.

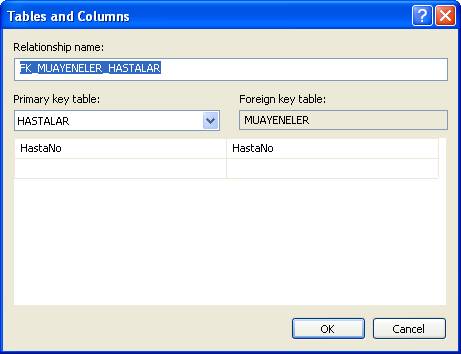
Bu bölüme baş anahtar seçmekte veya seçmemekte özgürsünüz. Seçmek istiyorsanız MuayeneNo adlı bölümü baş anahtar olarak seçebilirsiniz fakat kesinlikle Identity özelliğini kapatın ve sayısal olarak tekrar etmesini önleyin. Aksi takdirde hata verecektir. Daha önce MUAYENELER tablosuna kaydedilen veya kaydedeceğiniz bilgilerin ortaya çıkmasıyla MuayeneNo ortaya çıkaracaktır. Diyelimki otomatik sayı olarak 100 geldi. Bu sayı hastanın o sene içerisindeki o an kaydettirği muayene nosudur. Bu No sayesinde her türlü bilgiye ulaşabilirsiniz. Doktor kararı verdi. Bilgisayardan kararı girdi. Bu karar MUAYENEDETAY tablosunda MuayeneNo kolonuna 100 olarak girildi ve kararda grip olarak girildi. Olay bitti. Sorgulama yapıldığı zaman bu muayeneno sayesinde hastanın ne zaman muayene olduğunu öğrenebileceğiz. olucaz.

Son aşamalardan sonra diyagramımız aşağıdaki şekildeki gibi olacaktır.

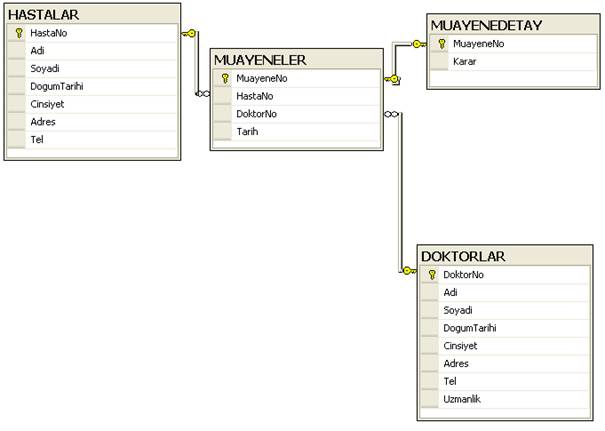


Böylece Hastalarımızı, doktorlarımızı ve muayeneler tablomuzu oluşturmuş olduk. Sıra geldi veritabanımızı ilişkilendirmeye ve sorgulama, kayıtlar yaparak veritabanımızı denemeye. Öncelikle veritabanları arasındaki ilişkilendirme olaylarına açıklama yapmak istiyorum.

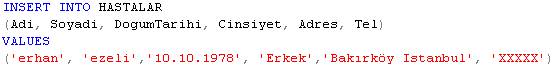
Veritabanı oluştuğunda ve kullanıldığında herşey normal bir görünümde çalışabilir. İstediğimiz hastayı kaydedebilir ve muayenelerini ekleyebiliriz. Fakat silme işlemi yaptığımızda veya benzeri işlemlerde, ilişkilendirme yapmassak problemler yaşayabiliriz. Bunu özellikle büyük projelerde yapmamız gerekir. Bu açıklamalarıma şöyle bir örnekle destek vermek istiyorum. HASTALAR tablosunda var olan bir hastanın nosunu kullanarak MUAYENELER tablosuna ver girişinde bulunduk. Daha sonra işlemler yapıldı ve HASTALAR tablosundan bir nedenden ötürü o hastayı silmek zorunda kaldık. Gerekli işlemleri yaptık ve sildik sizce unuttuğumuz birşey varmı! Silinen kayıt üzerinden daha önce işlem yapılmış ve MUAYENELER tablosunda silinen hastanın nosunun kaydedildiği bir veri girilmiştir. Bu durum daha sonra karşımıza çıkacak ve büyük karışıklıklara neden olabilecek. Düşünün hastalar tablosunda var olmayan bir hastanın muayenesi yapılmış iyi ama bu hasta kimdi!. Böyle durumlarda ilişkilendirme yaparak veritabanımızı belirli bir sisteme entegre edebiliriz. Aşağıda yapacağımız ilişkiyi tanımlayacağımız tablo durmaktadır. Bu tabloya diyagramlar bölümünden, tablolar arası ilişki yaparak çıkartabiliriz.



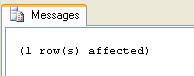
İlişkilerimizi şöyle açıklayalım. HASTALAR tablomuzdaki baş anahtar olan HastaNo ile MUAYENELER tablosundaki foreign key olan HastaNo arasında bire sonsuz bir ilişki kuruyoruz. Yani O şahısta sadece bir hasta olabilir ama o hasta her zaman istediği kadar muayene olabilir. Aynı şekilde DOKTORLAR tablosunda DoktorNo kolonu ile MUAYENELER tablosunda DoktorNo kolonunu bire sonsuz olarak ilişki kuruyoruz. Bunun anlamıda o şahısta sadece bir doktor olabilir ama o doktor istediği kadar hastayı muayene edebilir. Son olarakta MUAYENELER tablosundaki MuayeneNo kolonu ile MUAYENEDETAY tablosundaki MuayeneNo kolonlarını bire bire olarak ilişkilendiriyoruz. Daha önce MUAYENEDETAY tablosundaki MuayeneNo kolonunu başanahtar olarak seçmek veya seçmemek size kalmıştır demiştim. Şöyle bir düzeltmeyle seçmeniz mükemmel olurdu desek daha iyi yapmış olurdu. Çünkü arada ki ilişki bire sonsuz değil bire bir ilişki. Yani O an yapılan muayeneden sadece bir tane olabilir ve o yapılan muayenenin detayları sadece bir numarada toplanabilir. Yani sonsuz ilişkisi söz konusu değil. Bu durumda bire bir ilişki karşımıza çıkıyor. İlişkilendirmelerimizden sonra diyagram aşağıdaki gibi olacaktır.



İlişkilendirme işlemimiz gördüğünüz gibi oldu. Artık veritabanımız oluştu :) . Sıra geldi kayıt ekleme sorgulama ve güncelleme işlemlerine.

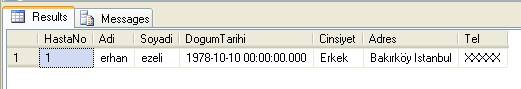


İlk sorgumuzla adı erhan soyadı ezeli olan, tarihlerinde doğan erkek, adres ve numara ile ilk sorgumuzu kaydediyoruz.



Mesajımızı aldık. Artık sorgulamamızı yapalım ve bakalımki kayıdımız HASTALAR tablosunda mevcutmu.

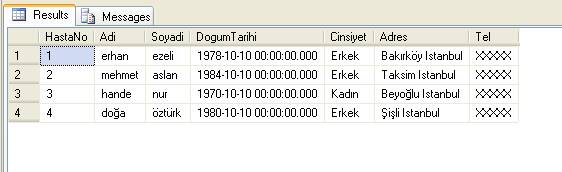
http://www.yazgelistir.com/Makaleler/Resimler/1000001823_image010.jpg



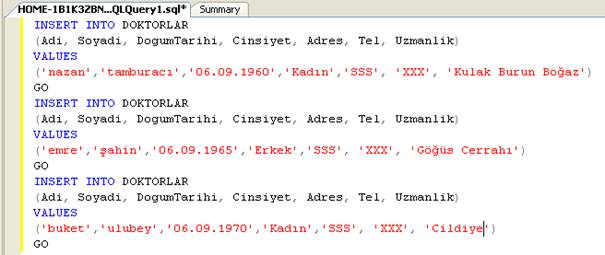
Gördüğünüz gibi veritabanımıza kayıt eklendi. İsterseniz hasta sayımızı arttıralım ve dörde çıkartalım.



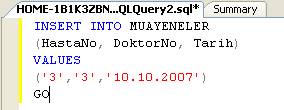
SQL sorgularımızı kabaca yazdık diyelim :) . Kayıtlarımızı girdik Burada birşey aklınıza takılmış olabilir farkındaysanız kayıt kodlarımıza HastaNoları girmedik ama nedense SQL bizim yerimize otomatik olarak sayılar vermiş :). Konunun en başında anlattığımı burada anlamış olmanız gerek.



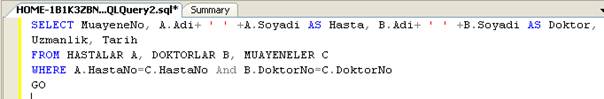
Son hal gördüğünüz gibi. Şimdi DOKTORLAR tablomuza aynı yollarla bir üç tane farklı uzmanlık alanları olan doktorlar girelim.



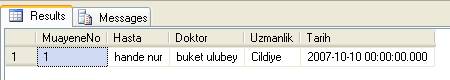
Doktorlarımızıda tanımladık. Geldik en heyecanlı yere. Diyelimki HastaNosu 3 olan hasta, DoktorNosu 3 olan buket hanımdan 10.10.2007 tarihi için randevu alacak



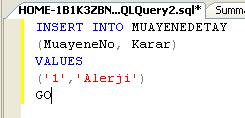
Kayıdımızı girmiş olduk. Artık önemli sorgulara gelelim.



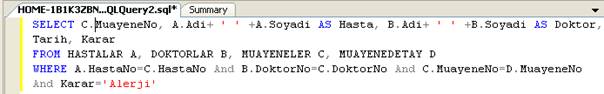
SQL cümleciğimizle veritabanlarının nolarını kullanarak birbirlerine bağladık ve istediğimiz sorgularımız çektik. Bu komut aşağıdaki gibi bir sonuç verecektir.



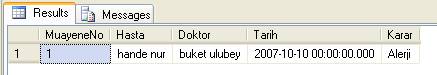
Aynı şekil MuayeneNoyu kleyerek MUAYENEDETAY bölümüne kayıdımızı girelim.



Sıra geldi en kritik koda. Sadece kriter olarak Alerji kararı verilmiş olan hastayı veya hastaları bulmak ve tüm muayene bilgilerinin gözükmesini isteyelim. Kodlar aşağıdaki gibidir.



Kodları işlediğinizde netice aşağıdaki gibi olacaktır.



Makelenin sonuna geldik. Umarım makele size az da olsa bilgi vermiş olur. Hastane veri tabanı çok daha ileri seviyelere geliştirilebilir.