PHP ile İnternet Programlama

Prof.Dr. Tolga GÜYER

Gazi Üniversitesi
Gazi Eğitim Fakültesi
Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü

4. BÖLÜM: MySQL ile Veritabanı Uygulamaları

Neden Veritabanı Kullanırız?

Çünkü verilerimizi düzenli ve kalıcı olarak saklamanın başka yolu yoktur.

Bildiğimiz gibi değişkenler ve sabitler, taşıdıkları değerleri programımız çalıştığı sürece saklayabilirler. Çünkü değişken ve sabitlerin veri saklamak için kullandıkları fiziksel ortam, bilgisayarın belleğidir.

Oysa verilerin kalıcı olarak saklandıkları ortamlar, öncelikle sabit disk ortamlarıdır. Bu donanım birimlerde veri saklayabilmek için, verilerimizi “dosya” adı verilen yazılım öğelerine aktarmamız gerekir.

Neden Veritabanı Kullanırız?

Verilerimizi saklayabileceğimiz dosyaları üç kategoride gruplandırabiliriz. Bunlardan ilki, yazılımların kullandıkları kendilerine özgü formatlardaki dosyalardır. İkili (binary) türdeki bu dosyalara örnek olarak doc, pdf, xls gibi dosya biçimleri verilebilir.

İkinci grupta ise metin-tabanlı (text-based) dosyalar yer alır ki veritabanlarının alternatifi olarak bu dosya türünü düşünebiliriz. Ancak bu tür dosyalarda saklanacak büyük veri yığınlarında istenilen bilgiye ulaşmanın oldukça güçleştiğini göz önünde bulundurursak, veri saklamak için çok da uygun olmadıkları anlaşılacaktır.

Üçüncü grupta ise veritabanları yer almaktadır.

Neden Veritabanı Kullanırız?

Veritabanları çok büyük veri yığınlarını, istenildiğinde aranan bilgi ya da bilgilere en kısa sürede ulaşılabilecek şekilde tasnif ederler. Bunun için kullandıkları temel veri saklama birimleri, tablolardır.

Tablolar, bildiğimiz anlamdaki çizelgelerden çok farklı değildir. Örneğin aşağıdaki çizelge, bir veritabanı tablosu olabilir:

Sıra No	Adı	Soyadı	Mesleği	Cinsiyeti
1	Cüneyt	Sancak	Yıllık izninin bir kısmını kullanan köşe yazarı	Erkek
2	Nilay	Ulusoy	Son ütücü	Kadın
3	Hüseyin	Korkmaz	Kaplumbağa terbiyecisi	Erkek

Neden Veritabanı Kullanırız?

Bir veritabanı bir çok tabloyu içerebilir. Her tablonun bir adı vardır. Tablodaki her bir alanın da (sütun) bir adı ve içersine yerleşecek verinin türü gibi özellikleri vardır.

Bütün bu yapının oluşturulmasına veritabanının tasarımı adı verilir. Bu süreç, doğal olarak, boş bir veritabanının oluşturulması ile başlar. Ardından tablolar ve bu tablolarda yer alacak alanlar ve bu alanların özelliklerinin atanması, dizin oluşturma işlemleri, anahtarların belirlenmesi gibi işlemler gelecektir.

Neden Veritabanı Kullanırız?

Veritabanları sadece verilerin tasnif edilmesi için gereken altyapının oluşturulduğu yazılımlar değildir. Veriye en kısa sürede ulaşmak için kullandıkları çeşitli algoritmik yöntemler, veri kapasitesi, veri güvenliği, programlama desteği gibi daha bir çok özellik, bu yazılımlarda standart olarak yer almaktadır.

Farklı veritabanı yazılımlarını birbirlerinden ayıran farklar da yukarıda sayılan özelliklerinden kaynaklanmaktadır. Örneğin Microsoft Access yazılımı da bütün standart özelliklere sahip bir veritabanı yazılımı olmasına karşın, veri kapasitesi ve güvenliği konularındaki sınırlılıkları nedeniyle çok fazla verinin işlenmesi gerektiği profesyonel uygulamalarda tercih edilmemektedir.

Bilinen bazı kalburüstü veritabanı yazılımları olarak Oracle, MySQL, Microsoft SQL Server, IBM DB2, dBase ve Paradox sayılabilir.

MySQL Veritabanının Yönetimi

XAMPP yazılımı bilgisayarınıza birinci bölümde anlatıldığı gibi kurulduysa, MySQL veritabanını yazılımın kontrol panelini kullanarak başlatabilirsiniz.

Veritabanına ait yönetim paneline ulaşmak için ise, herhangi bir internet tarayıcısının adres çubuğuna

http://localhost/phpmyadmin/

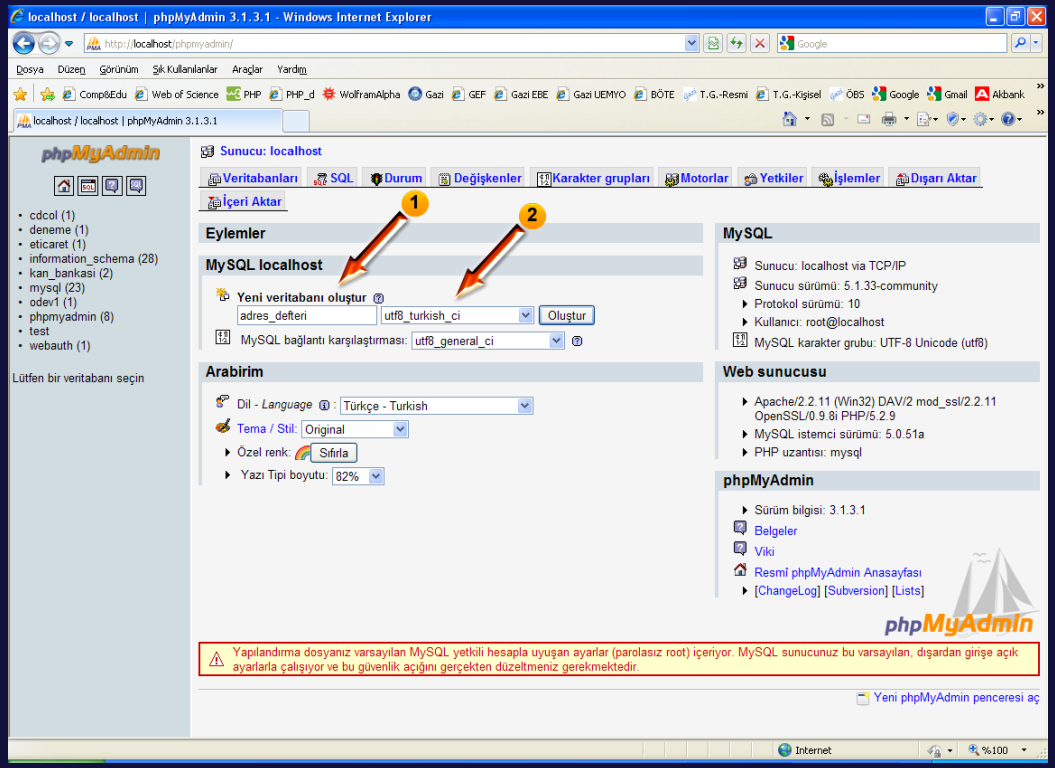
yazmak yeterli olacaktır. Ancak öncelikle localhost dizinine ulaşabilmemiz için, Apache sunucusunun da yine XAMPP kontrol panelinden başlatılmış olması gerektiğini hatırlatalım.

Karşımıza MySQL localhost adlı, veritabanı yönetimiyle ilgili her türlü işlemi gerçekleştirebileceğimiz ekran gelecektir.

İlk olarak bir veritabanı oluşturmakla başlayalım.

MySQL Veritabanının Yönetimi

Veritabanı Tanımlama



MySQL Veritabanının Yönetimi

Veritabanı Tanımlama

(1) numaralı alana oluşturacağımız veritabanının ismini yazalım. Bu isim belirlenirken, standart değişken isimlendirme kuralları geçerli olacaktır.

(2) numaralı alandan ise oluşturacağımız veritabanında yer alacak bilgilere uygun bir karakter kodlama tablosu seçilir. Türkçe için seçebileceğimiz en uygun karakter tablosu

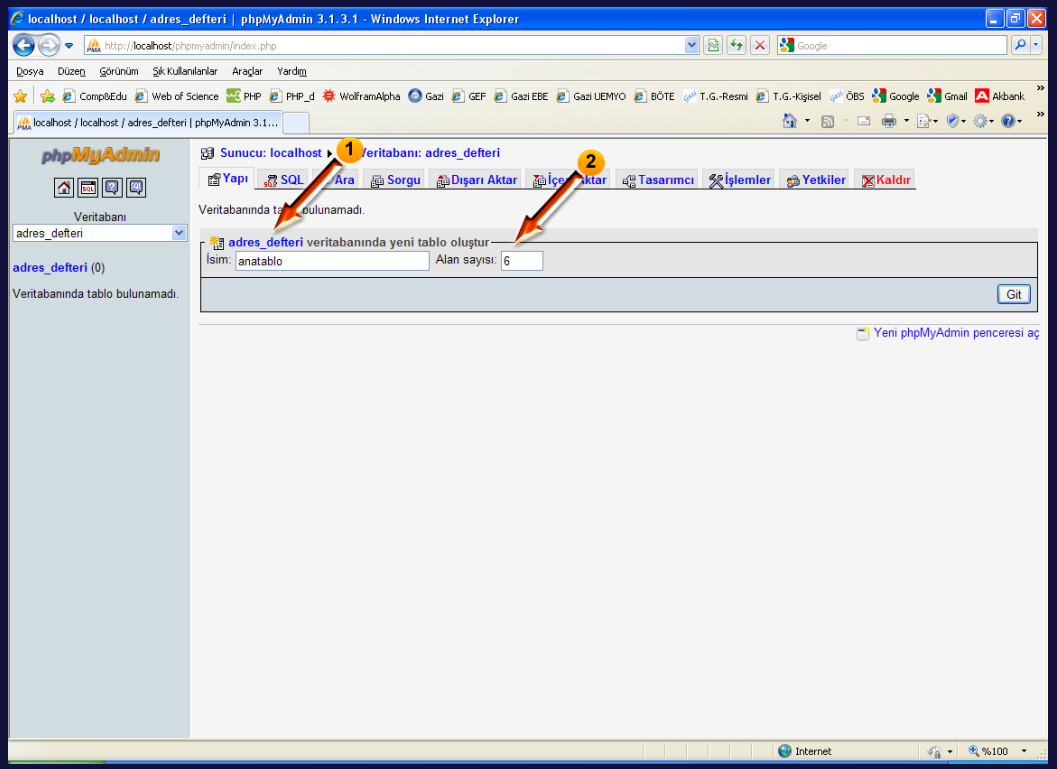
utf8_turkish_ci

olacaktır.

Oluştur butonuna bastığımızda veritabanı oluşturulur. İkinci aşama olarak ilk tablomuzu oluşturmalıyız. Her veritabanı, en az bir tablo içermek zorundadır.

MySQL Veritabanının Yönetimi

Tablo Oluşturma



MySQL Veritabanının Yönetimi

Tablo Oluşturma

(1) numaralı alana oluşturacağımız tablonun ismini yazalım. Bu isim belirlenirken de standart değişken isimlendirme kuralları geçerli olacaktır.

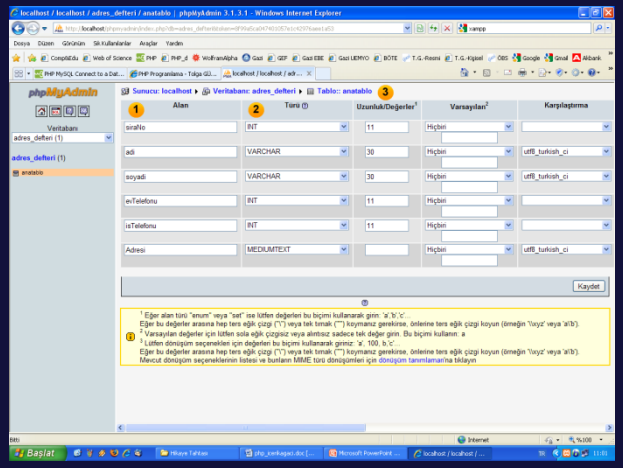
(2) numaralı alana ise oluşturacağımız tabloda yer alacak alanların sayısını yazalım.

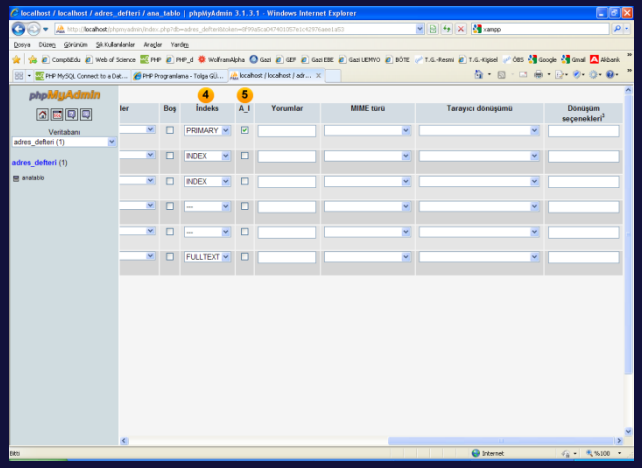
Git butonuna bastığımızda tablomuz oluşturulur ve her bir alanla ilgili özellikleri belirleyebileceğimiz ekranla karşılaşırız.

Bu özelliklerden şu an için gereksinim duyduğumuz kadarını belirleyelim.

MySQL Veritabanının Yönetimi

Alanların Özellikleri





(1) numaralı bölüme tanımladığımız alanın ismi yazılır. Bu isimler yine değişken isimlendirme kurallarına uyularak verilir.

(2) numaralı bölümde tanımlanan alana ait veri türü seçilir.

(3) numaralı bölümde ise seçilen veri türü için maksimum uzunluk değeri girilir.

MySQL Veritabanının Yönetimi

Alanların Özellikleri

(4) numaralı bölüm, alana ait indeks türünün seçileceği bölümdür. Her tabloda en az bir “birincil anahtar” (primary key) alanı bulunmak zorundadır. Biz bu alanı sıra numarasının saklandığı alan olarak belirledik. “Index” özelliği alanı arama işlemlerinde hızlı ulaşım için indeksler. “Full Text” özelliği ise alana tam metin indeksi özelliği vermek için kullanılır. Yani bu alanda yer alan verinin tamamı, anahtar kelime ile arama yapmaya uygun hale getirilmiş olur.

Kaydet butonuna basıldığında, artık veritabanımızda bir tablo oluşmuştur.

MySQL Veritabanının Yönetimi

Daha sonra veritabanımızla ilgili herhangi bir işlem yapmamız gerektiğinde, phpmyadmin panelinin sol tarafından veritabanımız seçmemiz yeterli olacaktır. Bu yapıldığında, sağ bölümde veritabanımızla ilgili olarak gerçekleştirebileceğimiz bütün işlemleri içeren görsel tasarımlı bir ekran gelecektir.

Benzer şekilde tablolarımızdaki her türlü güncellemeyi yine panelin sol tarafında veritabanımız seçili iken veritabanı adının altında yer alan tablo isimlerine tıklayarak yapabiliriz.

Genel olarak,

Veritabanı => Tablo => Alan

hiyerarşik yapısını düşündüğümüzde, phpmyadmin panelini kullanarak ayarlarını güncellemek istediğimiz nesnelere daha rahat ulaşabiliriz.

PHP & MySQL

MySQL Veritabanı sistemi bir çok programlama dili ile sorunsuz olarak çalışabilmesine karşın, genellikle PHP programcılarının tercih ettikleri bir veritabanı yönetim sistemi olmuştur. Bunda iki yazılımın mükemmele yakın bir şekilde uyumlu çalışabilmelerinin yanı sıra, her ikisinin de açık kaynak kod lisansına sahip olmaları da etkili olmaktadır.

Ayrıca bütün dünyada internet üzerinden gerek PHP, gerek MySQL için resmi web sitelerinin yanı sıra binlerce kaynak ve tartışma ortamları içeren web sitelerine ve günlüklerine (blog) ulaşmak mümkündür.

Bu bölümde, ilk olarak PHP ve MySQL arasındaki bağlantıdan başlayarak temel veritabanı işlemlerinin PHP kullanılarak web üzerinden nasıl gerçekleştirileceğine kadar olan bilgi ve tekniklere yer verilmiştir.

PHP & MySQL

MySQL ile Bağlantı Kurma

PHP programı içersinden, daha önceden oluşturduğumuz bir MySQL veritabanına, geçerli kullanıcı olan root kullanıcısı ile şifresiz olarak erişim aşağıdaki gibi sağlanabilir:

```
<?php
   $snc1 = mysqli_connect("localhost","root","");
   $snc2 = mysqli_select_db($snc1,"adres_defteri");
    if ($snc1 and $snc2)
    {
          echo "Bağlantı kuruldu ve veritabanı seçildi.";
    }
    else
    {
          echo "Sorun var!";
    }
?>
```

PHP & MySQL

MySQL ile Bağlantı Kurma

Burada $snc1 ve $snc1 değişkenlerinin her ikisinin de True değerini aldığı durumda, MySQL ile bağlantı kurulabilmiş ve adres_defteri adlı veritabanı seçilebilmiş demektir.

Aksi durum bu iki işlemden birisinde problem var anlamına gelecektir.

PHP & MySQL

MySQL ile Bağlantı Kurma

Biraz daha profesyonel tarzda bir bağlantı kurma kodu yazalım:

```
<?php
     define("MyServer","localhost");
     define("MyDatabase","adres_defteri");
     define("MyUser","root");
     define("MyPassword","");
function sqlConnect()
{
   $snc1 =
            mysqli_connect(MyServer,MyUser,MyPassword);
   $snc2 = mysqli_select_db($snc1,MyDatabase);
   $sncDeger = $snc1 and $snc2;
   return $sncDeger;}
     if (sqlConnect())
     {echo "Bağlantı kuruldu ve veritabanı seçildi.";}
     else {echo "Sorun var!";}
?>
```

PHP & MySQL

MySQL ile Bağlantı Kurma

Büyük programlarda genellikle sabit ve fonksiyon tanımlamaları ayrı dosyalarda tutulur. Böylece bu kodların her programda tekrarlanmasına da gerek kalmamış olur.

Bu dosyaların içerikleri include ya da require fonksiyonları kullanılarak okunur. Örneğin,

```
<?php
   require("tanimlar.php");
   require("fonksiyonlar.php");
   sqlConnect();
?>
```

kodunun çalışabilmesi için sistemde,

PHP & MySQL

MySQL ile Bağlantı Kurma

```
<?php
      define("MyServer","localhost");
      define("MyDatabase","basvuru");
      define("MyUser","root");
      define("MyPassword","");
?>
```

```
<?php
      function sqlConnect()
      {
            $snc1= mysqli_connect(MyServer,MyUser,MyPassword);
            $snc2 = mysqli,_select_db($snc1,MyDatabase);
            mysql_query("SET NAMES 'latin5'");
            $sncDeger = $snc1 and $snc2;
            return $sncDeger;
      }
?>
```

dosyalarının da bulunması gerekir.

PHP & MySQL

Temel Veritabanı İşlemleri

Veritabanlarının kullanım amacı, veri saklamak ve istenildiğinde bunlar üzerinde işlem yapmaktır. Bu işlemler ayrıntıya inildikçe çok fazla çeşitlilik gösterse de, temel olarak verilerimiz üzerinde gerçekleştirdiğimiz ortak işlemler,

- Kaydetme
- Arama
- Güncelleme
- Silme

şeklinde sıralanabilir.

PHP & MySQL

Temel Veritabanı İşlemleri

Bu tür işlemlerin kullanılan veritabanı ya da programlama dilinden bağımsız olarak standart yöntemlerle gerçekleştirilebilmeleri amacıyla SQL (Structured Query Language) adlı dil geliştirilmiştir. Diğer bir deyişle SQL dilinin amacı, bütün veritabanlarının bütün programlama dilleri ile aynı dili konuşmalarını sağlamaktır.

Bu bölümde gerçekleştireceğimiz PHP ve MySQL arasındaki iletişimlerde de SQL cümlelerini kullanacağız.

PHP & MySQL

Temel Veritabanı İşlemleri – Insert (Kayıt Girme)

Veritabanının bir tablosuna veri eklemek için kullandığımız SQL cümlesinin genel yapısı aşağıdaki gibidir:

INSERT INTO <tablo adı> (<alan adları> VALUES (<değerler>)

Alan adları aynen veritabanında tanımlandığı biçimleri ile yazılmalıdır ve birbirlerinden virgül işreti ile ayrılırlar.

Değerler ise karşılık gelen alan adları ile aynı sırada yazılırlar ve yine birbirlerinden virgül işreti ile ayrılırlar. Değerler yazılırken dikkat edilmesi gereken diğer bir husus da, veritabanında yerleştikleri alanın veri türü ile uyuşmalarıdır.

PHP & MySQL

Temel Veritabanı İşlemleri – Insert (Kayıt Girme)

"ogrenci1" adlı, aşağıdaki yapıda tanımlanmış bir veritabanımızın bulunduğunu varsayalım:

siraNo	ad	soyad	eposta
1	...	...	...
2	...	...	...
...	...	...	...

Sonraki slaytta yer alan program, bu veritabanına bir kayıt girdisi yapacaktır.

PHP & MySQL

Temel Veritabanı İşlemleri – Insert (Kayıt Girme)

```
<?php
    $baglanti=mysqli_connect("localhost","root" ,
"");
    mysqli_select_db($baglanti,"ogrenci1");
    mysqli_query($baglanti,"INSERT INTO anatablo
(ad , soyad ,
         eposta) VALUES ('Tolga' , 'Güyer' ,
         'guyer@gazi.edu.tr')");
?>
```

PHP & MySQL

Temel Veritabanı İşlemleri – Insert (Kayıt Girme)

Şimdi biraz daha karmaşık bir örnek geliştirelim. Bu defa veritabanımızın aşağıdaki gibi olsun:

numara	ad	soyad	vize	final	gecme	harf	basari
...	...	...	...	...	...	...	...

Temel Veritabanı İşlemleri – Insert (Kayıt Girme)

Bu durumda öğrenciye ait numara, ad, soyad, vize ve final notları kullanıcı tarafından giriliyor. Geçme notu, vize notunun %40'ı ve final notunun %60'ı toplanarak elde ediliyor. Harf notu ise referans değer tablosuna göre yine program tarafından atanıyor. Başarı durumu, geçme notuna göre hesaplanmaktadır. Eğer geçme notu 50 ve üzerinde ise bu alana geçti, aksi halde kaldı bilgisi otomatik olarak girilmektedir.

Sonuç olarak, kullanıcı tarafından girilen beş değer, programımız tarafından hesaplanan üç değerle birlikte veritabanımızın tablosuna bir kayıt girdisi olarak eklenmektedir.

PHP & MySQL

Temel Veritabanı İşlemleri – Insert (Kayıt Girme)

```
<form action="" method="post">
<table width="503" border="1">
<tr>
<td width="169" bgcolor="#FFFFCC">Numara :</td>
<td width="318" bgcolor="#33FFFF"><input size="10" type="text"
name="numara"></td></tr>
<tr>
<td bgcolor="#FFFFCC">Ad :</td>
<td bgcolor="#33FFFF"><input size="20" type="text" name="ad"></td></tr>
<tr>
<td bgcolor="#FFFFCC">Soyad :</td>
<td bgcolor="#33FFFF"><input size="20" type="text" name="soyad"></td></tr>
<tr>
<td bgcolor="#FFFFCC">Vize :</td>
<td bgcolor="#33FFFF"><input size="3" type="text" name="vize"></td></tr>
<tr>
<td bgcolor="#FFFFCC">Final :</td>
<td bgcolor="#33FFFF"><input size="3" type="text" name="final"></td></tr>
<tr>
<td colspan="2" bgcolor="#FFCC99"><input name="dugme" type="submit"
value="Hesapla &amp;amp; Kaydet"></td></tr>
</table>
</form>
```

PHP & MySQL

Temel Veritabanı İşlemleri – Insert (Kayıt Girme)

```
<?php
function sqlConnect() {
   $snc1 = mysqli_connect("localhost","root" , "");
   $snc2 = mysqli_select_db($snc1,"ogrenci2");
   $sncDeger = $snc1 and $snc2;
   return $sncDeger;}
function gecmeNotuHesapla($v, $f){
   return (2*$v+3*$f)/5;}
function harfNotuHesapla($Not){
   if ($Not<=39) {$harf = "FF"; $basari=0;}
   elseif ($Not<=49){$harf = "DC"; $basari=0;}
   elseif ($Not<=59){$harf = "CC"; $basari=1;}
   elseif ($Not<=69){$harf = "CB"; $basari=1;}
   elseif ($Not<=79){$harf = "BB"; $basari=1;}
   elseif ($Not<=89){$harf = "BA"; $basari=1;}
   elseif ($Not<=100){$harf = "AA"; $basari=1;}
   return $harf;}
function basariDurumu($Not){
   if ($Not<=49) {$basari=0;}
     else {$basari=1;}
   return $basari;}
```

PHP & MySQL

Temel Veritabanı İşlemleri – Insert (Kayıt Girme)

```
if(isset($_POST["dugme"])){
   if (sqlConnect()){
     echo "<p>Bağlantı başarılı...";
     $numara = $_POST['numara'];
     $ad = $_POST['ad'];
     $soyad = $_POST['soyad'];
     $vize = (int)$_POST['vize'];
     $final = (int)$_POST['final'];
     $gecme = number_format(gecmeNotuHesapla($vize, $final),2);
     $harf = harfNotuHesapla($gecme);
     if (basariDurumu($gecme)==1) {$basari_durumu="Geçti";}
       else {$basari_durumu="Kaldi";}
     $sonuc=mysqli_query($snc1,"INSERT INTO anatablo (numara, ad , soyad , vize,
final, gecme, harf, basari) VALUES ('$numara', '$ad' , '$soyad' , '$vize' ,
'$final' , '$gecme' , '$harf' , '$basari_durumu')");
     if ($sonuc){
        echo "<p>Geçme Notu.......", $gecme;
        echo "<p>Harf Notu........", $harf;
        echo "<p>Başarı Durumu....", $basari_durumu;
        echo "<p>Bilgiler veritabanına eklenmiştir...";}
      else
         {echo "Bilgiler eklenemedi...";}
    }
    else{ echo "Bağlantıda sorun var...";   } } ?>
```

Temel Veritabanı İşlemleri – Select (Kayıt Okuma)

Veritabanının bir tablosunda yer alan verileri okumak için kullandığımız SQL cümlesinin genel yapısı aşağıdaki gibidir:

SELECT <alan adları> FROM <tablo adı>

Alan adları aynen veritabanında tanımlandığı biçimleri ile yazılmalıdır ve birbirlerinden virgül işreti ile ayrılırlar. Eğer veritabanından bütün alanların içerdiği değerleri çekmek istiyorsak alan adları yerine ALL ya da * ifadesi kullanılır.

PHP & MySQL

Temel Veritabanı İşlemleri – Select (Kayıt Okuma)

Bir önceki örneğimizde kullandığımız “ogrenci2” veritabanındaki kayıtları okuyarak ekranda listeleyecek örneğimiz aşağıdaki gibidir:

```
<?php
function sqlConnect(){
   $snc1 = mysqli_connect("localhost","root" , "");
   $snc2 = mysqli_select_db($snc1,"ogrenci2");
   $sncDeger = $snc1 and $snc2;
   return $sncDeger;}
if (sqlConnect()){
     $sonuclar = mysqli_query($snc1,"SELECT numara,ad,soyad FROM
anatablo");
     while ($row=mysqli_fetch_array($sonuclar)){
     echo $row['numara']," ",$row['ad']," ",
                    $row['soyad'],"<br>";
     }
}
?>
```

Temel Veritabanı İşlemleri – Select (Kayıt Okuma)

Burada anatablo'dan her bir öğrenciye ait numara, ad ve soyad bilgileri çekilmiştir. Sorgulama sonucunda elde edilen kayıt kümesi (record set), $sonuçlar adlı bir değişkende saklanmıştır.

While döngüsü bu veri kümesini tarayarak her bir kaydın ekrana yazılmasını sağlamaktadır.

Temel Veritabanı İşlemleri – Where (Kayıt Süzme)

Bu kesimde SELECT cümlesinin koşullu kullanımını göreceğiz. Bu kullanımda, belirli bir koşulu sağlayan verilerin süzülmesi sağlanmaktadır. Genel yapısı aşağıdaki gibidir:

SELECT <alan adları> FROM <tablo adı> WHERE <koşul>

Koşul, alanlar üzerine tanımlı Boolean bir ifade şeklinde belirlenir.

PHP & MySQL

Temel Veritabanı İşlemleri – Where (Kayıt Süzme)

Yine önceki örneğimizde kullandığımız “ogrenci2” veritabanını kullanan bir örnek geliştirelim. Ancak bu defa, çektiğimiz verileri kullanıcının tercihine göre geçenler ya da kalanlar biçiminde süzecek bir tasarım yapalım.

PHP & MySQL

Temel Veritabanı İşlemleri – Where (Kayıt Süzme)

```
<?php
function sqlConnect(){
   $snc1 = mysqli_connect("localhost","root" , "");
   $snc2 = mysqli_select_db($snc1,"ogrenci2");
   $sncDeger = $snc1 and $snc2;
   return $sncDeger;}
if (sqlConnect()){
    $basari_durumu = $_POST['filtre'];
    $sonuclar = mysqli_query($snc1,"SELECT
numara,ad,soyad FROM anatablo WHERE
basari='$basari_durumu'");
    while ($row=mysqli_fetch_array($sonuclar)){
    echo $row['numara']," ",$row['ad']," ",
                $row['soyad'],"<br>";
    }
}
?>
```

Temel Veritabanı İşlemleri – Where (Kayıt Süzme)

SELECT cümlesinin bir diğer kullanım da, çekilen kayıt kümesini istenilen alan ya da alanlara göre sıralı olarak oluşturur. Genel yapısı aşağıdaki gibidir:

SELECT <alan adları> FROM <tablo adı> ORDER BY <alan(lar)> ASC|DSC

Artan sıralama isteniyorsa ASC, azalan sıralama için ise DSC ifadesi kullanılır.

PHP & MySQL

Temel Veritabanı İşlemleri – Where (Kayıt Süzme)

Verileri öğrenci adına göre artan sırada listeleyelim:

```
<?php
function sqlConnect(){
   $snc1 = mysqli_connect("localhost","root" , "");
   $snc2 = mysqli_select_db($snc1,"ogrenci2");
   $sncDeger = $snc1 and $snc2;
   return $sncDeger;}
if (sqlConnect()){
    $sonuclar = mysqli_query($snc1,"SELECT
numara,ad,soyad FROM anatablo ORDER BY ad ASC");
    while ($row=mysqli_fetch_array($sonuclar)){
    echo $row['numara']," ",$row['ad']," ",
                $row['soyad'],"<br>";
    } }
?>
```

PHP & MySQL

Temel Veritabanı İşlemleri – Update (Kayıt Güncelleme)

Veri güncellemek için kullandığımız SQL cümlesinin genel yapısı aşağıdaki gibidir:

UPDATE <tablo adı> SET <alan1>=<değer>, <alan2>=<değer>, …
WHERE <koşul>

Koşul cümlemizi sağlayan kayıt kümesindeki alanlar üzerinde güncelleme işlemi gerçekleştirilecektir.

PHP & MySQL

Temel Veritabanı İşlemleri – Update (Kayıt Güncelleme)

```
<?php
    $baglanti =
    mysqli_connect("localhost","root","");
    if (!$baglanti)
    {    die('Bağlanılamadı: ' . mysql_error());}
    mysqli_select_db($baglanti,"veritabanim");
    mysqli_query($baglanti,"UPDATE Kisiler SET Yas =
'36'
    WHERE Adi = 'Murtaza' AND Soyadi =
    'Dalkılıç'");
    mysqli_close($baglanti);
?>
```

Bu örnekte, “veritabanim” adlı veritabanının “Kisiler” tablosunda yer alan Murtaza Dalkılıç adlı yaş bilgisi 36 olarak güncellenmiştir.

PHP & MySQL

Temel Veritabanı İşlemleri – Delete (Kayıt Silme)

Veri silmek için kullandığımız SQL cümlesinin genel yapısı aşağıdaki gibidir:

DELETE FROM <tablo adı> WHERE <koşul>

Koşul cümlemizi sağlayan kayıt kümesindeki alanlar üzerinde silme işlemi gerçekleştirilecektir.

PHP & MySQL

Temel Veritabanı İşlemleri – Delete (Kayıt Silme)

```
<?php
    $baglanti =
    mysqli_connect("localhost","root","");
    if (!$baglanti)
    {die('Bağlanılamadı: ' . mysql_error());}
    mysqli_select_db($baglanti,"veritabanim");
    mysqli_query($baglanti,"DELETE FROM Kisiler
                    WHERE Soyadi = 'Dalkılıç'");
    mysqli_close($baglanti);
?>
```

Bu örnekte, “veritabanim” adlı veritabanının “Kisiler” tablosunda yer alan ve soyadı Dalkılıç olan bütün kayıtlar silinecektir.

Veritabanı-bağlantılı Sayfa Tazeleme

Sayfamız üzerinde yer alan nesnelerin, veritabanından gelen bilgiler kullanılarak doldurulması işlemine verilen addır. Bu nesneler açılan kutular (combobox) ya da liste kutuları (listbox) olabileceği gibi, gazetelerin web sayfalarında olduğu gibi resim alanları bile bu şekilde güncel tutulabilir. Böylelikle oluşturulan web sitesinin dinamik bir yapıda olması sağlanmış olur.

Bu kesimde, sayfa tazeleme için bir Javascript kodunun kullanılacağı sayfa-veritabanı bağlantısı tekniği, bir örnek üzerinden anlatılacaktır.

Veritabanı-bağlantılı Sayfa Tazeleme

Öncelikle, sayfa tazeleme işlemimin gerçekleştirecek Javascript kodumuzu yazalım.

```
<SCRIPT LANGUAGE="JavaScript">
   function frmYenile()
   {
        document.form1.method='POST';
        document.form1.submit();
   }
</SCRIPT>
```

Burada kırmızı ile işaretlenen form1 ifadesi, kullandığımız HTML formunun adı olacaktır.

PHP & MySQL

Veritabanı-bağlantılı Sayfa Tazeleme

Şimdi HTML formumuzu tasarlayalım.

```
<form name="form1" action="" method="post">
 <p>Adı          : <?php echo "<input size=\"20\" type=\"text\"
name=\"ad\" value=\"$_POST[ad]\"></p>"; ?>
 <p>Soyadı       : <?php echo "<input size=\"20\" type=\"text\"
name=\"soyad\" value=\"$_POST[soyad]\"></p>"; ?>
 <p>Personel No  : <select size="1" name="per_no" onchange="frmYenile();">
    <?php
      sqlConnect();
      $sonuclar = mysqli_query($baglanti,"SELECT pernum FROM personel");
      $secenekler="<option value=\"-1\"
selected=\"selected\">Seçiniz</option>";
      while ($row=mysqli_fetch_array($sonuclar)){
             $secenekler.="<option value=".$row['pernum']."></option>\n";
      }
      echo ($secenekler);
    ?>
 </select>
  <p><input name="dugme" type="submit" value="Veritabanına Ekle"></p>
</form>
```

Veritabanı-bağlantılı Sayfa Tazeleme

Örneğimizde dikkat edilmesi gereken noktalardan ilki, ad ve soyad alanlarında value parametresinin kullanımdır. Bunun amacı, sayfa tazelendiğinde bu alanların kullanıcı tarafından önceden girilmiş olan değerlerini kaybetmemelerini sağlamaktır.

İkinci nokta ise, bir açılan kutu olarak tasarlanan personel numarası alanıdır. Bu alana ait seçenekler, doğrudan veritabanında yer alan personel tablosundan çekilmektedir. Alandan herhangi bir seçim yapıldığı anda frmYenile() kodu, sayfayı tazelemektedir. Bunun sebebi, örnek uygulamalarımızda da olduğu gibi, bu alana bağlı olarak başka bir alanda gerçekleştirilebilecek herhangi bir değişikliğin geçerli olmasını sağlamaktır.