(SQL)

Structured Query Language (Yapılandırılmış Sorgu Dili)

İçindekiler

Önsöz

Bu dokümanı benim gibi sql öğrenmek isteyenler ve sql hakkında Türkçe kaynak bulamayanlar için MS Access'in yardım bölümünden yararlanarak hazırladım. İtiraf etmek gerekirse her ne kadar yazıları hazır alsamda bu yazıları belirli bir düzene sokmak için çok uğraştım. Bu dokümanda en çok kullanılan sql komutlarını açıklama ve örnekleriyle birlikte bulacaksanız. Yararlı olacağı inancındayım. Belgede bazı eksiklikler olabilir onlarıda mazur görün. Son olarakta bu belgeyi başka yerlerde kullanmak isteyen arkadaşlarımız (yazılar için değil çünkü ben yazmadım) bu belgeyi hazırlamak için sarf ettiğim emeği göz önünde bulundurup kaynak belirtirlerse sevinirim.

Saadettin POLAT

(SQL)

Structured Query Language (Yapılandırılmış Sorgu Dili)

Sql Nedir?

İlişkisel veritabanlarını sorgulamak, güncellemek için kullanılan bir dildir.

Sql Nerelerde Kullanılır?

Programcılıkta (Delphi, VB v.b.), veritabanı programlarında ve veritabanına bağlı web sayfalarında (Asp v.b.) kullanılır.

VERİ KULLANMA DİLİ

- **Deyimler**
  - **SELECT Deyimi**

Veritabanından bir kayıt kümesi şeklinde bilgi döndürmek Microsoft Jet veritabanı alt yapısına komut verir.

Sözdizimi

SELECT [*doğrulama*] { * | *tablo*.* | [*tablo*.]*alan1* [AS *diğerad1*] [, [*tablo*.]*alan2* [AS *diğerad2*] [, ...]]}
FROM *tabloifadesi* [, ...] [IN *dışveritabanı*]
[WHERE...]
[GROUP BY...]
[HAVING...]
[ORDER BY...]
[WITH OWNERACCESS OPTION]

SELECT deyiminin bölümleri şunlardır:

Bölüm	Açıklama
doğrulama	Şu doğrulamalardan biridir: ALL, DISTINCT, DISTINCTROW, or TOP. Doğrulamayı, döndürülecek kayıtların sayısını sınırlandırmak için kullanırsınız. Bunların hiçbiri belirtilmezse, varsayılan değer ALL'dur.
*	Belirtilen tablodan veya tablolarda tüm kayıtların seçileceğini belirtir.
tablo	İçinden kayıtların seçileceği alanları içeren tablonun adıdır.
alan1, *alan2*	Almak istediğiniz verileri içeren alanların adıdır. Birden fazla alan yazarsanız, alanlar listelendikleri sırada alınırlar.
diğerad1, *diğerad2*	*Tablodaki* özgün sütun adları yerine, sütun başlıkları olarak kullanılacak adlardır.
tabloifadesi	Almak istediğiniz verileri içeren tablo veya tabloların adıdır.
dışveritabanı	*Tabloifadesi* içindeki tablolardan, geçerli veritabanı içinde olmayanları içeren veritabanının adıdır.

Uyarılar

Bu işlemi gerçekleştirmek için, Microsoft® Jet veritabanı alt yapısı, belirtilen tablo veya tabloları arar, seçili sütunları tarar, ölçütlere uyan satırları seçer ve sonuç satırları belirtilen sırada sıralar veya gruplandırır.

SELECT deyimleri veritabanındaki verileri değiştirmezler.

SELECT genellikle bir SQL deyiminin ilk sözcüğüdür. Çoğu SQL deyimi SELECT veya SELECT...INTO deyimleridir.

SELECT deyiminin en kısa sözdizimi şöyledir:

SELECT *alanlar* FROM *tablo*

Tablodaki tüm dizinleri seçmek için bir yıldız (*) simgesi de kullanabilirsiniz. Aşağıdaki örnek, Çalışanlar tablosundaki tüm alanları seçer.

SELECT * FROM *Çalışanlar;*

FROM yan tümcesinde bir alan adı birden fazla tabloda yer alıyorsa, alan adından önce tablo adını ve ardından **.** (nokta) işaretini yazın. Aşağıdaki örnekte, Bölüm alanı hem Çalışanlar hem de Denetçiler tablosunda bulunmaktadır. SQL deyimi, Çalışanlar tablosundan bölümleri ve Denetçiler tablosundan denetçi adlarını seçer:

SELECT Çalışanlar.Bölüm, Denetçiler.DenetçiAdı

FROM Çalışanlar INNER JOIN Denetçiler

WHERE Çalışanlar.Bölüm = Denetçiler.Bölüm;

Bir **RecordSet** nesnesi oluşturulduğunda, Microsoft Jet veritabanı alt yapısı tablonun alan adını **recordset** nesnesi içinde **Field** nesnesi adı olarak kullanır. Farklı bir alan adı isterseniz veya alan oluşturmak üzere kullanılan ifadede bir ad belirtilmezse, **AS** saklı sözcüğünü kullanın. Aşağıdaki örnek, sonuç **recordset** nesnesinde döndürülen **field** nesnesini adlandırmak için Doğum başlığını kullanır:

SELECT DoğumTarihi

AS Doğum FROM Çalışanlar;

Tanımsız veya yinelenen **Alan** nesne adları döndüren toplam işlevleri veya sorgular kullanırken, **Field** nesnesi için farklı bir ad sağlamak üzere AS yan tümcesini kullanmalısınız. Aşağıdaki örnek, sonuç **RecordSet** nesnesinde döndürülen **Field** nesnesini adlandırmak için Merkez başlığını kullanır:

SELECT COUNT(ÇalışanNo)

AS Merkez FROM Çalışanlar;

Döndürülecek verileri daha fazla kısıtlamak veya düzenlemek üzere, SELECT deyiminde diğer yan tümceleri de kullanabilirsiniz.

SELECT Deyimi, FROM Yan Tümcesi Örneği

Aşağıdaki örneklerden bazıları, Çalışanlar tablosunda Maaş alanının bulunduğunu varsayar. Bu alan gerçekte Nortwind veritabanının Çalışanlar tablosunda bulunmaz.

Bu örnek, Çalışanlar tablosundaki tüm kayıtların Adı ve Soyadı alanlarını seçen bir SQL deyimini temel alan bir **Recordset** devingen küme türü oluşturur. **Recordset** nesnesinin içeriklerini **Hata Ayıklama** penceresine yazdıran EnumFields yordamını çağırır.

```
Sub SelectX1()

    Dim dbs As Database, rst As Recordset

    ' Bu satırı, bilgisayarınızdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' Çalışanlar tablosundaki tüm kayıtların adı ve soyadı
```

```
    ' değerlerini seçer.
    Set rst = dbs.OpenRecordset("SELECT Soyadı, " _
        & "Adı FROM Çalışanlar;")
    ' Yeni kayıt kümesini başlatır.
    rst.MoveLast
' Recordset içeriklerini yazdırmak üzere EnumFields
' yordamını çağırır.
    EnumFields rst,12
dbs.Close
End Sub
```

Bu örnek, Posta Kodu alanında girilmiş değeri olan kayıt sayısını sayar ve geri döndürülen alanı Sayaç olarak adlandırır.

```
Sub SelectX2()
    Dim dbs As Database, rst As Recordset
    ' Bu satırı, bilgisayarınızdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' PostaKodu değeri olan kayıt sayısını sayar ve
    ' sonucu Sayaç alanında verir.
    Set rst = dbs.OpenRecordset("SELECT Count " _
        & "(PostaKodu) AS Sayaç FROM Müşteriler;")
    ' Recordset'i başlatır.
    rst.MoveLast
    ' Recordset içeriklerini yazdırmak üzere EnumFields
    ' yordamını çağırır. Alan genişliğini 12 yapar.
    EnumFields rst, 12
    dbs.Close
End Sub
```

Bu örnek, çalışan sayısını ve maaşların ortalaması ile en yüksek değerini gösterir.

```
Sub SelectX3()
    Dim dbs As Database, rst As Recordset
    ' Bu satırı, bilgisayarınızdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
Set dbs = OpenDatabase("Northwind.mdb")
```

```
    ' Çalışan sayısını sayar, ortalama maaşı hesaplar
    ' ve en yüksek maaşı verir.
    Set rst = dbs.OpenRecordset("SELECT Count (*) " _
        & "AS ToplamÇalışan, Avg(Maaş) " _
        & "AS OrtalamaMaaş, Max(Maaş) " _
        & "AS EnYüksekMaaş FROM Çalışanlar;")
    ' Recordset'i başlatır.
    rst.MoveLast
    ' Recordset içeriklerini yazdırmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 17
    dbs.Close
End Sub
```

EnumFields **Sub** yordamı, çağıran yordamdan bir **Recordset** nesnesi aktarır. Daha sonra, **Recordset** alanlarını biçimlendirir ve **Hata Ayıklama** penceresine yazdırır. intFldLen değişkeni, istenen basılı alan genişliğidir. Bazı alanlar düzgün basılmadan kesilebilir.

```
Sub EnumFields(rst As Recordset, intFldLen As Integer)
    Dim lngRecords As Long, lngFields As Long
    Dim lngRecCount As Long, lngFldCount As Long
    Dim strTitle As String, strTemp As String
    ' lngRecords değişkenini Recordset içindeki
    ' kayıt sayısına eşitler.
    lngRecords = rst.RecordCount
    ' lngFields değişkenini Recordset içindeki
    ' alan sayısına eşitler.
    lngFields = rst.Fields.Count

    Debug.Print "Kayyt kümesinde " & lngRecords _
        & " kayyt " & lngFields _
        & " alan var."
    Debug.Print

    ' Sütun başlığını yazdırmak üzere bir dize oluşturur.
```

```
strTitle = "Kayyt "
For lngFldCount = 0 To lngFields - 1
    strTitle = strTitle _
    & Left(rst.Fields(lngFldCount).Name _
    & Space(intFldLen), intFldLen)
Next lngFldCount

' Sütun başlığını yazdırır.
Debug.Print strTitle
Debug.Print

' Recordset içinde dönerek kayyt sayysyny ve
' alan değerlerini yazdırır.
rst.MoveFirst
For lngRecCount = 0 To lngRecords - 1
    Debug.Print Right(Space(6) & _
        Str(lngRecCount), 6) & " ";
    For lngFldCount = 0 To lngFields - 1
        ' Null değerleri denetler.
        If IsNull(rst.Fields(lngFldCount)) Then
            strTemp = "<null>"
        Else
            ' strTemp içeriğini alan içeriği olarak ayarlar.
            Select Case _
                rst.Fields(lngFldCount).Type
                Case 11
                    strTemp = ""
                Case dbText, dbMemo
                    strTemp = _
                        rst.Fields(lngFldCount)
                Case Else
                    strTemp = _
                        str(rst.Fields(lngFldCount))
```

```
            End Select
        End If
        Debug.Print Left(strTemp _
            & Space(intFldLen), intFldLen);
    Next lngFldCount
    Debug.Print
    rst.MoveNext
  Next lngRecCount
End Sub
```

- **DELETE Deyimi**

WHERE yan tümcesini sağlayan FROM yan tümcesinde listelenen tablolardan bir veya daha fazlasından kayıt silen bir silme sorgusu oluşturur.

Sözdizimi

DELETE [*tablo*.*]
FROM *tablo*
WHERE *ölçüt*

DELETE deyiminin bölümleri şunlardır:

Bölüm	Açıklama
Tablo	İçinden kayıtların silineceği tablonun isteğe bağlı adıdır.
Tablo	İçinden kayıtların silineceği tablonun adıdır.
Ölçütler	Silinecek kayıtları belirleyen bir ifadedir.

Uyarılar

DELETE, özellikle çok sayıda kayıt silmek istiyorsanız kullanışlıdır.

Veritabanından tablonun tümünü silmek için, **Execute** yöntemini DROP deyimi ile birlikte kullanabilirsiniz. Ancak tabloyu silerseniz, tablonun yapısı da kaybolur. Buna karşılık DELETE'i kullanırsanız yalnızca veriler silinir, tablo yapısı ve alan özellikleri ve dizin gibi tablo özelliklerinin tümü bozulmadan kalır.

Diğer tablolarla bir-çok ilişkisi olan tablolardan kayıt silmek için DELETE'i kullanabilirsiniz. Ardarda silme işlemleri, ilişkinin "bir" tarafında olan tabloda bulunan bir kaydın silinmesi halinde, ilişkinin "çok" tarafında olan tablodaki ilgili kayıtların silinmesine neden olur. Örneğin, Müşteriler ve Siparişler tablolarındaki ilişkide, Müşteriler tablosu ilişkinin "bir" tarafında ve Siparişler tablosu da "çok" tarafındadır. Ardarda silme seçeneği belirtilmişse, Müşteriler tablosunda bir kayıt silindiğinde ilgili Siparişler tablosu kayıtları da silinecektir.

Bir silme sorgusu, yalnızca belirli alanlardaki verileri değil kayıtların tamamını siler. Yalnızca belirli bir alandaki verileri silmek istiyorsanız, değerleri **Null** olarak değiştiren bir güncelleştirme sorgusu oluşturun.

Önemli

- Silme sorgusunu kullanarak kayıt sildikten sonra, işlemi geri alamazsınız. Hangi kayıtların silindiğini bilmek istiyorsanız, sonuçları önce aynı ölçütü kullanan bir seçme sorgusu ile denetleyin ve sonra silme sorgusunu çalıştırın.

- Her zaman verilerinizin yedeklerini saklayın. Yanlış kayıtları silerseniz, kayıtları yedek kopyalardan alabilirsiniz.

DELETE Deyimi Örneği

Bu örnek, ünvanı Eğitmen olan tüm çalışanların kayıtlarını siler. FROM yan tümcesi yalnızca bir tablo içeriyorsa, DELETE deyiminde tablo adını listelemeniz gerekmez.

```
Sub DeleteX()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' Ünvanı Eğitmen olan çalışanların kayıtlarını siler.

    dbs.Execute "DELETE * FROM " _

        & "Çalışanlar WHERE Ünvan = 'Eğitmen';"

        dbs.Close

End Sub
```

INSERT INTO Deyimi

Bir tabloya bir veya daha çok sayıda kayıt ekler. Buna ekleme sorgusu da denir.

Sözdizimi

Çok sayıda kayıt ekleme sorgusu:

INSERT INTO *hedef* [(*alan1*[, *alan2*[, ...]])] [IN *dışveritabanı*]
SELECT [*kaynak*.]*alan1*[, *alan2*[, ...]
FROM *tabloifadesi*

Tek kayıt ekleme sorgusu:

INSERT INTO *hedef* [(*alan1*[, *alan2*[, ...]])]
VALUES (*değer1*[, *değer2*[, ...])

INSERT INTO deyiminin bölümleri şunlardır:

Bölüm	Açıklama
Hedef	İçine kayıt eklenecek tablo veya sorgunun adıdır.
alan1, *alan2*	Bir *hedef* değişkeninin ardından geliyorsa içine veri eklenecek alanların adı, bir *kaynak* değişkeninin ardından geliyorsa içinden veri alınacak alanların adıdır.
dışveritabanı	Bir dış veritabanına olan yoldur. Yolun açıklaması için IN yan tümcesine bakın.
Kaynak	İçinden kayıtların kopyalanacağı tablo veya sorgunun adıdır.
tabloifadesi	İçinden kayıtların eklendiği tablo veya tabloların adıdır. Bu değişken;

	tek bir tablo adı , kaydedilmiş bir sorgu adı veya INNER JOIN, LEFT JOIN veya RIGHT JOIN sonucu olan bir birleşim olabilir.
değer1, *değer2*	Yeni kaydın belirli alanlarına eklenecek olan değerlerdir. Her değer, listede değerin konumuna göre ilgili alanın içine eklenir: *değer1* yeni kaydın *alan1* alanına, *değer2* yeni kaydın *alan2* alanına şeklinde eklenir. Değerleri virgül ile ayırmalı ve metin alanlarını tırnak imleri (' ') içine almalısınız.

Uyarılar

INSERT INTO deyimini, yukarıda gösterildiği gibi tek kayıt ekleme sorgusu sözdizimini kullanarak tabloya tek bir kayıt eklemek üzere kullanabilirsiniz. Bu durumda kodlarınız, kaydın her alanının adını ve değerini belirtir. Üzerine bir değerin atanacağı kayıt alanlarını ve bu alanların değerlerini belirtmelisiniz. Her alanı tek tek belirtmezseniz, eksik sütunlar için varsayılan değer veya **Null** değeri eklenir. Kayıtlar tablonun sonuna eklenir.

Yukarıda çok sayıda kayıt ekleme sorgusu sözdiziminde gösterildiği gibi, SELECT ... FROM yan tümcesini kullanarak bir kayıt kümesini bir başka tablodan veya sorgudan eklemek üzere INSERT INTO'yu da kullanabilirsiniz.Bu durumda SELECT yan tümcesi, belirtilen *hedef* tabloya eklenecek alanları belirtir.

Kaynak veya *hedef* tablo, bir tablo veya sorguyu belirtebilir. Bir sorgu belirtilirse, Microsoft Jet veritabanı alt yapısı, sorgu tarafından belirtilen herhangi bir tabloya ve tüm tablolara kayıt ekler.

INSERT INTO isteğe bağlıdır ancak belirtilirse, SELECT deyiminden önce gelir.

Hedef tablonuzda bir birincil anahtar varsa, birincil anahtar alan veya alanlarına benzersiz, **Null** olmayan değerler eklediğinizden emin olun; aksi takdirde Microsoft Jet veritabanı alt yapısı kayıtları eklemez.

İçinde OtomatikSayı alanı olan bir tabloya kayıt ekler ve eklenen kayıtların numaralarını değiştirmek isterseniz, sorgunuzda OtomatikSayı alanını bulundurmayın. OtomatikSayı alanının özgün değerlerini korumak istiyorsanız, alanı sorgu içinde bulundurun.

Kayıtları başka bir veritabanındaki tabloya eklemek için IN yan tümcesini kullanın.

Yeni bir tablo oluşturmak üzere bir tablo oluşturma sorgusu hazırlamak yerine SELECT... INTO deyimini kullanın.

Ekleme sorgusunu çalıştırmadan önce hangi kayıtların ekleneceğini görmek için, önce, aynı seçim ölçütünü kullanan bir seçme sorgusu çalıştırın ve sonuçları görün.

Bir ekleme sorgusu, bir veya daha çok tablodaki kayıtları bir başkasına kopyalar. Eklediğiniz kayıtları içeren tablolar, ekleme sorgusundan etkilenmezler.

Bir başka tablodan var olan kayıtları eklemek yerine, VALUES yan tümcesini kullanarak, tek bir yeni kaydın tüm alanlarının değerlerini belirtebilirsiniz. Alan listesini yazmazsanız VALUES yan tümcesi tablodaki her alanın değerini içermelidir; aksi takdirde INSERT işlemi başarısız olur. Oluşturmak istediğiniz her ek kayıt için, VALUES yan tümcesi ile birlikte ek bir INSERT INTO deyimi kullanın.

INSERT INTO Deyimi Örneği

Bu örnek, bulunduğu varsayılan Yeni Müşteriler tablosundaki tüm müşterileri seçer ve bunları Müşteriler tablosuna ekler. Tek tek sütunlar belirtilmezse, SELECT tablosu sütun adları ile, INSERT INTO tablosundaki sütun adları tam olarak eşleşmelidir.

```
Sub InsertIntoX1()

    Dim dbs As Database

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.
```

```
    Set dbs = OpenDatabase("Northwind.mdb")

    ' Yeni Müşteriler tablosundaki tüm kayıtları seçer
    ' ve bunları Müşteriler tablosuna ekler.
    dbs.Execute " INSERT INTO Müşteriler " _
        & "SELECT * " _
        & "FROM [Yeni Müşteriler];"
        dbs.Close
End Sub
```

Bu örnek Çalışanlar tablosunda yeni bir kayıt oluşturur.

```
Sub InsertIntoX2()
    Dim dbs As Database
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")

    ' Çalışanlar tablosunda yeni bir kayıt oluşturur. Yeni
    ' kaydyn ady Gamze, soyady Etikan ve
    ' iş ünvanı Eğitmen'dir.
    dbs.Execute " INSERT INTO Çalışanlar " _
        & "(Ady,Soyady, Ünvan) VALUES " _
        & "('Gamze', 'Etikan', 'Eğitmen');"
         dbs.Close
End Sub
```

- **SELECT...INTO Deyimi**

Bir tablo oluşturma sorgusu oluşturur.

Sözdizimi

SELECT *alan1*[, *alan2*[, ...]] INTO *yenitablo* [IN *dışveritabanı*]
FROM *kaynak*

SELECT...INTO deyiminin bölümleri şunlardır:

Bölüm	**Açıklama**
alan1, *alan2*	Yeni tabloya kopyalanacak alanların adıdır.
Yenitablo	Oluşturulacak tablonun adıdır. Standart adlandırma kuralları na uymalıdır. *Yenitablo*, var olan bir tablonun adı ile aynıysa, algılanabilir

	bir hata oluşur.
dışveritabanı	Bir dış veritabanına olan yoldur. Yolun açıklaması için IN yan tümcesine bakın.
kaynak	İçinden kayıtların seçileceği var olan bir tablonun adıdır. Bu, tek veya çok tablo veya sorgu olabilir.

Uyarılar

Kayıtların arşivini oluşturmak, tabloları yedeklemek veya farklı bir veritabanına vermek üzere veya belirli bir zaman aralığındaki verileri görüntüleyen raporlarda temel olarak kullanmak üzere tabloların kopyalarını oluşturmak için tablo oluşturma sorgularını kullanabilirsiniz. Örneğin, aynı tablo oluşturma sorgusunu her ay çalıştırarak, Bölgelere Göre Aylık Satışlar raporunu oluşturabilirsiniz.

Notlar

- Yeni tablo için bir birincil anahtar tanımlamak isteyebilirsiniz. Tablo oluşturduğunuzda, yeni tablodaki alanlar, sorgunun temel aldığı tablolardaki her alanın veri türünde ve alan boyutunda olurlar; ancak diğer alan ve tablo özellikleri aktarılmaz.
- Var olan bir tabloya veri eklemek için, bir ekleme sorgusu oluşturmak yerine INSERT INTO deyimini kullanın.
- Tablo oluşturma sorgusunu çalıştırmadan önce hangi kayıtların seçileceğini görmek için, önce, aynı seçim ölçütünü kullanan bir SELECT deyimi çalıştırın ve sonuçları görün.

SELECT...INTO Deyimi Örneği

Bu örnek, Çalışanlar tablosundaki tüm kayıtları seçer ve bunları Çalışanlar Yedeği adlı yeni bir tabloya ekler.

```
Sub SelectIntoX()

    Dim dbs As Database

    Dim qdf As QueryDef

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' Çalışanlar tablosundaki tüm kayıtları seçer

    ' ve bunları yeni Çalışanlar Yedeği tablosuna kopyalar.

    dbs.Execute "SELECT Çalışanlar.* INTO " _

        & "[Çalışanlar Yedeği] FROM Çalışanlar;"


    ' Bu bir örnek olduğu için tabloyu siler.

    dbs.Execute "DROP TABLE [Çalışanlar Yedeği];"


    dbs.Close

End Sub
```

- **UPDATE Deyimi**

Belirtilen ölçütleri temel alan belirtilen tablodaki alanların değerlerini değiştiren bir güncelleştirme sorgusu oluşturur.

Sözdizimi

UPDATE *tablo*
SET *yenideğer*
WHERE *ölçüt;*

UPDATE deyiminin bölümleri şunlardır:

Bölüm	Açıklama
tablo	Değiştirmek istediğiniz verileri içeren tablonun adıdır.
yenideğer	Güncelleştirilen kayıtlardaki belirli bir alana eklenecek değeri belirten bir ifadedir.
ölçütler	Hangi kayıtların güncelleştirileceğini belirleyen bir ifadedir. Yalnızca ifadeyi sağlayan kayıtlar güncelleştirilir.

Uyarılar

UPDATE özellikle, çok sayıda kaydı değiştirmek istediğinizde veya değiştirmek istediğiniz kayıtlar çok sayıda tablo içinde iken kullanışlıdır.

Aynı anda çok sayıda alanı değiştirebilirsiniz. Aşağıdaki örnek Sipariş Miktarı değerlerini yüzde 10 arttırır ve Navlun değerlerini İngiltere'deki taşımacılar için yüzde 3 arttırır:

```
UPDATE Siparişler

SET SiparişMiktarı = SiparişMiktarı * 1.1,

Navlun = Navlun * 1.03

WHERE Ülke = 'İngiltere';
```

Önemli

- UPDATE, bir sonuç kümesi oluşturmaz. Ayrıca, güncelleştirme sorgusunu kullanarak kayıtları güncelleştirdikten sonra, işlemi geri alamazsınız. Hangi kayıtların güncelleştirildiğini bilmek istiyorsanız, sonuçları önce aynı ölçütü kullanan bir seçme sorgusu ile denetleyin ve sonra güncelleştirme sorgusunu çalıştırın.

- Her zaman verilerinizin yedeklerini saklayın. Yanlış kayıtları güncelleştirirseniz, kayıtları yedek kopyalardan alabilirsiniz.

UPDATE Deyimi Örneği

Bu örnek, Raporla alanının değeri 2 olan tüm çalışan kayıtlarının Raporla alanını 5 olarak değiştirir.

```
Sub UpdateX()

    Dim dbs As Database

    Dim qdf As QueryDef

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")
```

```
    ' Raporla alanındaki değeri 2 olan tüm çalışan
    ' kayıtlarının Raporla alanının değerini 5 olarak
    ' değiştirir.
    dbs.Execute "UPDATE Çalışanlar " _
        & "SET Raporla = 5 " _
        & "WHERE Raporla = 2;"

    dbs.Close
End Sub
```

- **TRANSFORM Deyimi**

Bir çapraz sorgu oluşturur.

Sözdizimi

TRANSFORM *toplamişlevi*
** *seçmedeyimi***
** PIVOT *özetalanı* [IN (*değer1*[, *değer2*[, ...]])]**

TRANSFORM deyiminin bölümleri şunlardır:

Bölüm	Açıklama
toplamişlevi	Seçili veriler üzerinde işlem yapan bir SQL toplam işlevidir.
seçmedeyimi	Bir SELECT deyimidir.
özetalanı	Sorgunun sonuç kümesinde sütun başlığı oluşturmakta kullanmak istediğiniz alan veya ifadedir.
değer1, *değer2*	Sütun başlıklarını oluşturmak üzere kullanılan sabit değerlerdir.

Uyarılar

Bir çapraz sorguyu kullanarak verileri özetlerken, belirtilen alanlardaki veya ifadelerdeki değerleri sütun başlıkları olarak seçersiniz; böylece verileri, seçme sorgusu ile olduğundan daha açık bir biçimde görüntüleyebilirsiniz.

TRANSFORM isteğe bağlıdır, ancak seçildiğinde, bir SQL dizesinin ilk deyimidir. Satır başlıkları olarak kullanılan alanları belirten SELECT deyiminden ve satır gruplandırmasını belirten GROUP BY yan tümcesinden önce gelir. İsterseniz, ek seçim veya sıralama ölçütleri belirten WHERE gibi diğer yan tümceleri de bulundurabilirsiniz. Bir çapraz sorguda alt sorguları doğrulama olarak da kullanabilirsiniz (özellikle WHERE yan tümcesinde bulunanları).

Özetalanın döndürdüğü değerler, sorgunun sonuç kümesinde sütun başlıkları olarak kullanılır. Örneğin, aylık satışların satış değerlerini özetlemek, 12 sütunu olan bir çapraz sorgu oluşturur. Başlıkları, isteğe bağlı IN yan tümcesinde listelenen sabit değerlerle (*değer1*, *değer2*) oluşturmak üzere *özetalanını* kısıtlayabilirsiniz. Ek sütunlar oluşturmak için, ilgili verileri olmayan sabit değerleri de bulundurabilirsiniz.

TRANSFORM Deyimi Örneği

Bu örnek, 1994 yılının her üç aylık dönemi için çalışanların verdikleri sipariş miktarını gösteren bir çapraz sorgu oluşturmak üzere bir SQLTRANSFORM yan tümcesi kullanır. SQLTRANSFORMOutput işlevi, bu yordamın çalışması için gereklidir.

```
Sub TransformX1()
    Dim dbs As Database
    Dim strSQL As String
    Dim qdfTRANSFORM As QueryDef
    strSQL = "PARAMETERS prmYear SHORT; TRANSFORM " _
       & "Count(SiparişNo) " _
       & "SELECT Ady & "" "" & Soyady AS " _
       & "TamAdı FROM Çalışanlar INNER JOIN Siparişler " _
       & "ON Çalışanlar.ÇalışanNo = " _
       & "Siparişler.ÇalışanNo WHERE TarihAralığı " _
       & ("""yyyy""", SiparişTarihi) = [prmYear] "

      strSQL = strSQL & "GROUP BY Ady & " _
       & """ "" & Soyady " _
       & "ORDER BY Ady & "" "" & Soyady " _
       & "PIVOT TarihAralığı(""q"", SiparişTarihi)"

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    Set qdfTRANSFORM = dbs.CreateQueryDef _
       ("", strSQL)

    SQLTRANSFORMOutput qdfTRANSFORM, 1994

    dbs.Close
End Sub
```

Bu örnek, 1994 yılının her üç aylık dönemi için çalışanların verdikleri siparişlerin toplam TL miktarını gösteren daha karmaşık bir çapraz sorgu oluşturmak üzere bir SQLTRANSFORM yan tümcesi kullanır. SQLTRANSFORMOutput işlevi, bu yordamın çalışması için gereklidir.

```
Sub TransformX2()
    Dim dbs As Database
```

```
    Dim strSQL As String
    Dim qdfTRANSFORM As QueryDef
    strSQL = "PARAMETERS prmYear SMALLINT; TRANSFORM " _
       & "Sum(AltToplam) SELECT Ady & "" """ _
       & "& Soyady AS TamAdy " _
       & "FROM Çalışanlar INNER JOIN " _
       & "(Siparişler INNER JOIN [Sipariş Alt Toplamları] " _
       & "ON Siparişler.SiparişNo = " _
       & "[Sipariş Alt Toplamları].SiparişNo) " _
       & "ON Çalışanlar.ÇalışanNo = " _
       & "Siparişler.ÇalışanNo WHERE TarihAralığı " _
       & ("yyyy"", SiparişTarihi) = [prmYear] "

      strSQL = strSQL & "GROUP BY Ady & "" """ _
       & "& Soyady " _
       & "ORDER BY Ady & "" "" & Soyady " _
       & "PIVOT TarihAralığı(""q"",SiparişTarihi)"

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    Set qdfTRANSFORM = dbs.CreateQueryDef _
       ("", strSQL)

    SQLTRANSFORMOutput qdfTRANSFORM, 1994

    dbs.Close
End Sub
Function SQLTRANSFORMOutput(qdfTemp As QueryDef, _
    intYear As Integer)

    Dim rstTRANSFORM As Recordset
    Dim fldLoop As Field
```

```
Dim booFirst As Boolean
qdfTemp.PARAMETERS!prmYear = intYear
Set rstTRANSFORM = qdfTemp.OpenRecordset()

Debug.Print qdfTemp.SQL
Debug.Print
Debug.Print , , "Üç Ay"
With rstTRANSFORM
   booFirst = True
   For Each fldLoop In .Fields
      If booFirst = True Then
         Debug.Print fldLoop.Name
         Debug.Print , ;
         booFirst = False
      Else
         Debug.Print , fldLoop.Name;
      End If
   Next fldLoop
   Debug.Print

   Do While Not .EOF
      booFirst = True
      For Each fldLoop In .Fields
         If booFirst = True Then
            Debug.Print fldLoop
            Debug.Print , ;
            booFirst = False
         Else
            Debug.Print , fldLoop;
         End If
      Next fldLoop
      Debug.Print
      .MoveNext
```

Loop

End With

End Function

- **EXECUTE Deyimi**

Bir yordamın yürütülmesini başlatmak için kullanılır.

Sözdizimi

EXECUTE *yordam* [*parametre1*[, *parametre2*[, ...]]

EXECUTE deyiminin bölümleri şunlardır:

Bölüm	**Açıklama**
yordam	Yürütülecek yordamın adıdır.
parametre1, parametre2	Yordam tarafından tanımlanan parametrelerin değerleridir.

- **TRANSACTION Deyimi**

Dolaysız hareketleri başlatmak ve sonuçlandırmak için kullanılır.

Sözdizimi

Yeni bir hareket başlatma:

BEGIN TRANSACTION

Hareket sırasında gerçekleştirilmiş tüm işleri kaydederek hareketi sonlandırma:

COMMIT [TRANSACTION | WORK]

Hareket sırasında gerçekleştirilmiş tüm işleri geri alarak hareketi sonlandırma:

ROLLBACK [TRANSACTION | WORK]

Uyarılar

Hareketler otomatik olarak başlatılmaz. Bir hareketi başlatmak için, bunu BEGIN TRANSACTION'i kullanarak dolaysız olarak yapmalısınız.

Hareketler içiçe en çok beş düzeyde olabilirler. Bir alt düzey hareketini başlatmak için, var olan bir hareketin içeriği içinde BEGIN TRANSACTION'i kullanın.

Hareketler, bağlı tablolarda desteklenmezler.

- **Yan Tümceler**
  - **FROM Yan Tümcesi**

SELECT deyiminde listelenen alanları içeren tabloları veya sorguları belirtir.

Sözdizimi

SELECT *alanlistesi*
FROM *tablodeyimi* [IN *dışveritabanı*]

FROM yan tümcesi içeren bir SELECT deyiminin bölümleri şunlardır:

Bölüm	Açıklama
alanlistesi	Herhangi bir alan diğeradı, SQL toplam işlevi, doğrulamalar (ALL, DISTINCT, DISTINCTROW veya TOP) veya diğer SELECT deyimi seçenekleri ile alınabilecek alan veya alanların adı.
tabloifadesi	İçinden verilerin alınacağı bir veya daha fazla tabloyu belirten bir ifadedir. İfade; tek bir tablo adı, kaydedilmiş bir sorgu adı veya INNER JOIN, LEFT JOIN veya RIGHT JOIN sonucu olan bir birleşim olabilir.
dışveritabanı	*Tabloifadesi* içindeki tüm tabloları içeren dış veritabanının tam yolu.

Uyarılar

FROM gereklidir ve her SELECT deyiminin ardından gelir.

Tablodeyimi içindeki tablo adlarının sırası önemli değildir.

Daha yüksek bir başarımı ve daha kolay kullanım sağlamak için, dış bir veritabanından veri almak üzere IN yan tümcesini kullanmak yerine bağlı tablo kullanmanız önerilir.

Aşağıdaki örnek, Çalışanlar tablosundan nasıl veri alabileceğinizi gösterir:

```
SELECT Soyadı, Adı

FROM Çalışanlar;
```

SELECT Deyimi, FROM Yan Tümcesi Örneği

Aşağıdaki örneklerden bazıları, Çalışanlar tablosunda Maaş alanının bulunduğunu varsayar. Bu alan gerçekte Nortwind veritabanının Çalışanlar tablosunda bulunmaz.

Bu örnek, Çalışanlar tablosundaki tüm kayıtların Adı ve Soyadı alanlarını seçen bir SQL deyimini temel alan bir **Recordset** devingen küme türü oluşturur. **Recordset** nesnesinin içeriklerini **Hata Ayıklama** penceresine yazdıran EnumFields yordamını çağırır.

```
Sub SelectX1()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' Çalışanlar tablosundaki tüm kayıtların adı ve soyadı

    ' değerlerini seçer.

    Set rst = dbs.OpenRecordset("SELECT Soyady, " _

        & "Adı FROM Çalışanlar;")

    ' Yeni kayıt kümesini başlatır.

    rst.MoveLast

' Recordset içeriklerini yazdyrmak üzere EnumFields

' yordamını çağırır.
```

```
    EnumFields rst,12
dbs.Close
End Sub
```

Bu örnek, PostaKodu alanında girilmiş değeri olan kayıt sayısını sayar ve geri döndürülen alanı Sayaç olarak adlandırır.

```
Sub SelectX2()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' PostaKodu değeri olan kayıt sayısını sayar ve
    ' sonucu Sayaç alanynda verir.
    Set rst = dbs.OpenRecordset("SELECT Count " _
        & "(PostaKodu) AS Sayaç FROM Müşteriler;")
    ' Recordset'i başlatır.
    rst.MoveLast
    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Alan genişliğini 12 yapar.
    EnumFields rst, 12
    dbs.Close
End Sub
```

Bu örnek, çalışan sayısını ve maaşların ortalaması ile en yüksek değerini gösterir.

```
Sub SelectX3()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
Set dbs = OpenDatabase("Northwind.mdb")
    ' Çalışan sayısını sayar, ortalama maaşı hesaplar
    ' ve en yüksek maaşı verir.
    Set rst = dbs.OpenRecordset("SELECT Count (*) " _
        & "AS ToplamÇalışan, Avg(Maaş) " _
        & "AS OrtalamaMaaş, Max(Maaş) " _
        & "AS EnYüksekMaaş FROM Çalışanlar;")
    ' Recordset'i başlatır.
```

```
    rst.MoveLast
    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 17
    dbs.Close
End Sub
```

EnumFields **Sub** yordamı, çağıran yordamdan bir **Recordset** nesnesi aktarır. Daha sonra, **Recordset** alanlarını biçimlendirir ve **Hata Ayıklama** penceresine yazdırır. intFldLen değişkeni, istenen basılı alan genişliğidir. Bazı alanlar düzgün basılmadan kesilebilir.

```
Sub EnumFields(rst As Recordset, intFldLen As Integer)
    Dim lngRecords As Long, lngFields As Long
    Dim lngRecCount As Long, lngFldCount As Long
    Dim strTitle As String, strTemp As String
    ' lngRecords değişkenini Recordset içindeki
    ' kayıt sayısına eşitler.
    lngRecords = rst.RecordCount
    ' lngFields değişkenini Recordset içindeki
    ' alan sayısına eşitler.
    lngFields = rst.Fields.Count

    Debug.Print "Kayyt kümesinde " & lngRecords _
        & " kayyt " & lngFields _
        & " alan var."
    Debug.Print

    ' Sütun başlığını yazdırmak üzere bir dize oluşturur.
    strTitle = "Kayyt "
    For lngFldCount = 0 To lngFields - 1
        strTitle = strTitle _
        & Left(rst.Fields(lngFldCount).Name _
        & Space(intFldLen), intFldLen)
    Next lngFldCount
```

```
' Sütun başlığını yazdırır.
Debug.Print strTitle
Debug.Print

' Recordset içinde dönerek kayyt sayysyny ve
' alan değerlerini yazdırır.
rst.MoveFirst
For lngRecCount = 0 To lngRecords - 1
    Debug.Print Right(Space(6) & _
        Str(lngRecCount), 6) & " ";
    For lngFldCount = 0 To lngFields - 1
        ' Null değerleri denetler.
        If IsNull(rst.Fields(lngFldCount)) Then
            strTemp = "<null>"
        Else
            ' strTemp içeriğini alan içeriği olarak ayarlar.
            Select Case _
                rst.Fields(lngFldCount).Type
                Case 11
                    strTemp = ""
                Case dbText, dbMemo
                    strTemp = _
                        rst.Fields(lngFldCount)
                Case Else
                    strTemp = _
                        str(rst.Fields(lngFldCount))
            End Select
        End If
        Debug.Print Left(strTemp _
            & Space(intFldLen), intFldLen);
    Next lngFldCount
    Debug.Print
    rst.MoveNext
```

```
    Next lngRecCount

End Sub
```

- **GROUP BY Yan Tümcesi**

Belirtilen alan listesindeki benzer değerlere sahip kayıtları tek bir kayıt olarak birleştirir. SELECT deyiminde **Sum** veya **Count** gibi bir SQL toplam işlevi bulundurursanız, her kayıt için bir özet değer oluşturulur.

Sözdizimi

SELECT *alanlistesi*
FROM *tablo*
WHERE *ölçüt*
[GROUP BY *grupalanlistesi*]

GROUP BY yan tümcesi içeren bir SELECT deyiminin bölümleri şunlardır:

Bölüm	Açıklama
alanlistesi	Herhangi bir alan diğeradı, SQL toplam işlevi, doğrulamalar (ALL, DISTINCT, DISTINCTROW veya TOP) veya diğer SELECT deyimi seçenekleri ile alınabilecek alan veya alanların adı.
tablo	İçinden kayıtların seçileceği tablonun adıdır. Ayrıntılı bilgi için FROM yan tümcesine bakın.
ölçütler	Seçim ölçütü. Deyim bir WHERE yan tümcesi içeriyorsa, Microsoft Jet veritabanı alt yapısı, kayıtlara WHERE koşullarını uyguladıktan sonra değerleri gruplandırır.
grupalanlistesi	Kayıtları gruplandırmak için kullanılan en çok 10 alanın adı. *Grupalanlistesi* içindeki alan adlarının sırası, grupta en üstten en alta doğru gruplandırma düzeylerini belirler.

Uyarılar

GROUP BY isteğe bağlıdır.

SELECT deyiminde SQL toplam işlevi yoksa özet değerler göz ardı edilir.

GROUP BY alanlarındaki **Null** değerleri gruplandırılır ve dikkate alınır. Ancak, **Null** değerleri herhangi bir SQL toplam işlevinde değerlendirilmez.

Gruplandırmak istemediğiniz satırları dışarıda tutmak için WHERE yan tümcesini kullanın ve kayıtları gruplandırma işleminden sonra süzmek için HAVING yan tümcesini kullanın.

GROUP BY alan listesindeki bir alan Not veya OLE Nesnesi verileri içermiyorsa, bu alan, SELECT deyiminde bulundurulmamış olsa bile SELECT deyimi en az bir SQL toplam işlevi içeriyorsa, FROM yan tümcesinde listelenen tablolardaki herhangi bir alana başvurabilir. Microsoft® Jet veritabanı alt yapısı Not veya OLE Nesnesi alanlarını gruplandıramaz.

SELECT alan listesindeki tüm alanlar, ya GROUP BY yan tümcesinde ya da bir SQL toplam işlevinin değişkeni olarak bulundurulmalıdır.

GROUP BY Yan Tümcesi Örneği

Bu örnek, benzersiz iş ünvanlarının ve bu ünvanlara sahip olan çalışan sayısının listesini verir.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub GroupByX1()

    Dim dbs As Database, rst As Recordset
```

```
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' Her ünvan için, bu ünvana sahip çalışanları
    ' sayar.
    Set rst = dbs.OpenRecordset("SELECT Ünvan, " _
        & "Count([Ünvan]) AS Sayaç " _
        & "FROM Çalışanlar GROUP BY Ünvan;")

    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 25
    dbs.Close
End Sub
```

Bu örnek benzersiz her iş ünvanı için, bu ünvana sahip İstanbul'daki çalışan sayısını hesaplar.

```
Sub GroupByX2()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")

    ' Her ünvan için, bu ünvana sahip çalışanları
    ' sayar. Yalnyzca Ystanbul bölgesindeki
    ' çalışanları bulundurur.
    Set rst = dbs.OpenRecordset("SELECT Ünvan, " _
        & "Count(Ünvan) AS Sayaç " _
        & "FROM Çalışanlar WHERE Bölge = 'İS' " _
        & "GROUP BY Ünvan;")
```

```
    ' Recordset'i başlatır.
    rst.MoveLast
    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 25
    dbs.Close
End Sub
```

- **HAVING Yan Tümcesi**

GROUP BY yan tümcesi olan bir SELECT deyiminde hangi gruplandırılmış kayıtların görüntüleneceğini belirler. GROUP BY kayıtları birleştirdikten sonra, HAVING, HAVING yan tümcesinin koşullarını sağlayan ve GROUP BY yan tümcesi ile gruplandırılmış kayıtları görüntüler.

Sözdizimi

SELECT *alanlistesi*
FROM *tablo*
WHERE *seçmeölçütü*
GROUP BY *grupalanlistesi*
[HAVING *grupölçütü*]

HAVING yan tümcesi içeren bir SELECT deyiminin bölümleri şunlardır:

Bölüm	Açıklama
alanlistesi	Herhangi bir alan diğeradı, SQL toplam işlevi, doğrulamalar (ALL, DISTINCT, DISTINCTROW veya TOP) veya diğer SELECT deyimi seçenekleri ile alınabilecek alan veya alanların adı.
tablo	İçinden kayıtların seçileceği tablonun adıdır. Ayrıntılı bilgi için FROM yan tümcesine bakın.
seçimölçütü	Seçim ölçütüdür. Deyim bir WHERE yan tümcesi içeriyorsa, Microsoft Jet veritabanı alt yapısı, kayıtlara WHERE koşullarını uyguladıktan sonra değerleri gruplandırır.
grupalanlistesi	Kayıtları gruplandırmak için kullanılan en çok 10 alanın adı. *Grupalanlistesi* içindeki alan adlarının sırası, grupta en üstten en alta doğru gruplandırma düzeylerini belirler.
grupölçütü	Hangi gruplandırılmış kayıtların görüntüleneceğini belirleyen bir ifadedir.

Uyarılar

HAVING isteğe bağlıdır.

HAVING, WHERE'e benzer ve hangi kayıtların seçileceğini belirler. Kayıtlar GROUP BY ile gruplandırıldıktan sonra HAVING de hangi kayıtların görüntüleneceğini belirtir.

```
SELECT KategoriNo,
Sum(StoktakiBirim)
FROM Ürünler
GROUP BY KategoriNo
```

HAVING Sum(StoktakiBirim) > 100 And Like "BOS*";

Bir HAVING yan tümcesi, **And** ve **Or** gibi mantıksal işleçlerle birbirine bağlı en çok 40 ifade içerebilir.

HAVING Yan Tümcesi Örneği

Bu örnek, İstanbul bölgesindeki birden çok çalışana atanmış iş ünvanlarını seçer.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub HavingX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' İstanbul bölgesindeki birden çok çalışana atanmış
    ' iş ünvanlarını seçer.
    Set rst = dbs.OpenRecordset("SELECT Ünvan, " _
        & "Count(Ünvan) as Toplam FROM Çalışanlar " _
        & "WHERE Bölge = 'YS' " _
        & "GROUP BY Ünvan HAVING Count(Ünvan) > 1;")

    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdırmak üzere EnumFields'i çağırır.
    EnumFields rst, 25
    dbs.Close
End Sub
```

- **IN Yan Tümcesi**

Microsoft Jet database alt yapısının bağlanabileceği, dBASE veya Paradox veritabanı veya bir dış Microsoft® Jet veritabanı gibi herhangi bir dış veritabanındaki tabloları belirler.

Sözdizimi

Hedef bir tablo belirlemek için:

[SELECT | INSERT] INTO *hedef* IN
{*yol* | ["*yol*" "*tür*"] | ["" [*tür*; DATABASE = *yol*]]}

Kaynak bir tablo belirlemek için:

FROM *tabloifadesi* IN
{*yol* | ["*yol*" "*tür*"] | ["" [*tür*; DATABASE = *yol*]]}

IN yan tümcesi içeren bir SELECT deyiminin bölümleri şunlardır:

Bölüm	Açıklama
Hedef	İçine verilerin ekleneceği dış tablonun adıdır.
Tabloifadesi	İçinden verilerin alınacağı tablo veya tabloların adıdır. Bu değişken; tek bir tablo adı, kaydedilmiş bir sorgu adı veya INNER JOIN, LEFT JOIN veya RIGHT JOIN sonucu olan bir birleşim olabilir.
Yol	*Tabloyu* içeren dizin veya dosyanın tam yoludur.
Tür	Veritabanı bir Microsoft Jet veritabanı (örneğin, dBASE III, dBASE IV, Paradox 3.x veya Paradox 4.x) değilse, *tablo* oluşturmak için kullanılan veritabanı türünün adıdır.

Uyarılar

IN yan tümcesini bir anda yalnızca tek bir dış veritabanına bağlanmak için kullanabilirsiniz.

Bazı durumlarda *yol* değişkeni, veritabanı dosyalarını içeren dizini gösterir. Örneğin dBASE, Microsoft FoxPro® veya Paradox veritabanı tabloları ile çalışırken, *yol* değişkeni .dbf veya .db dosyalarını içeren dizini belirtir. Tablonun dosya adı, *hedef* veya *tabloifadesi* değişkeninden türetilir.

Microsoft Jet veritabanı dışında bir veritabanı belirtmek için, ada bir noktalı virgül (;) ekleyin ve adı tek (' ') veya çift (" ") tırnak imi içine alın. Örneğin 'dBASE IV;' veya "dBASE IV;" olabilir.

Dış veritabanı belirlemek için DATABASE saklı sözcüğünü de kullanabilirsiniz. Örneğin, aşağıdaki satırlar aynı tabloyu belirtir:

... FROM Tablo IN "" [dBASE IV; DATABASE=C:\DBASE\DATA\SALES;];

... FROM Tablo IN "C:\DBASE\DATA\SALES" "dBASE IV;"

Notlar

Daha fazla başarım ve daha kolay kullanım sağlamak için, IN yerine bağlı tablo kullanın.

IN saklı sözcüğünü, bir ifade içinde bir karşılaştırma işleci olarak da kullanabilirsiniz. Ayrıntılı bilgi için **In** işlecine bakın.

IN Yan Tümcesi Örneği

Aşağıdaki tablo, dış bir veritabanından veri almak için IN yan tümcesini nasıl kullanabileceğinizi gösterir. Her örnekte, dış veritabanında Müşteriler tablosunun bulunduğunu varsayın.

Dış veritabanı	SQL deyimi
Microsoft® Jet veritabanı	SELECT MüşteriNo FROM Müşteriler IN DiğerVeritabanı.mdb WHERE MüşteriNo Like "A*";
dBASE III veya IV. Bir dBASE III tablosundan veri almak için, "dBASE IV;" yerine "dBASE III;" yazın.	SELECT MüşteriNo FROM Müşteriler IN "C:\DBASE\VERİ\SATŞLAR" "dBASE IV;" WHERE MüşteriNo Like "A*";
Veritabanı sözdizimi kullanan dBASE III veya IV.	SELECT MüşteriNo FROM Müşteriler IN "" [dBASE IV; Database=C:\DBASE\VERİ\SATIŞ;] WHERE MüşteriNo Like "A*";
Paradox 3.x veya 4.x.	SELECT MüşteriNo

Paradox sürüm 3.x tablosundan veri almak için, "Paradox 4.x;" yerine "Paradox 3.x;" yazın.	FROM Müşteriler IN "C:\PARADOX\VERİ\SATŞLAR" "Paradox 4.x;" WHERE MüşteriNo Like "A*";
Veritabanı sözdizimi kullanan Paradox 3.x veya 4.x.	SELECT MüşteriNo FROM Müşteriler IN "" [Paradox 4.x;Database=C:\PARADOX\VERİ\SATIŞLAR;] WHERE MüşteriNo Like "A*";
Microsoft Excel çalışma sayfası	SELECT MüşteriNo, ŞirketAdı FROM [Müşteriler$] IN "c:\belgeler\xlveri.xls" "EXCEL 5.0;" WHERE MüşteriNo Like "A*" ORDER BY MüşteriNo;
Çalışma sayfası içinde adlandırılmış bir aralık	SELECT MüşteriNo, ŞirketAdı FROM MüşterilerAralığı IN "c:\belgeler\xlveri.xls" "EXCEL 5.0;" WHERE MüşteriNo Like "A*" ORDER BY MüşteriNo;

- **ORDER BY Yan Tümcesi**

Sorgunun sonuç kayıtlarını belirtilen alan veya alanlara göre artan veya azalan sırada sıralar.

Sözdizimi

SELECT *alanlistesi*
FROM *tablo*
WHERE *seçmeölçütü*
[ORDER BY *alan1* [ASC | DESC][, *alan2* [ASC | DESC]][, ...]]

ORDER BY yan tümcesi içeren bir SELECT deyiminin bölümleri şunlardır:

Bölüm	Açıklama
alanlistesi	Herhangi bir alan diğeradı, SQL toplam işlevi, doğrulamalar (ALL, DISTINCT, DISTINCTROW veya TOP) veya diğer SELECT deyimi seçenekleri ile alınabilecek alan veya alanların adı.
tablo	İçinden kayıtların seçileceği tablonun adıdır. Ayrıntılı bilgi için FROM yan tümcesine bakın.
seçimölçütü	Seçim ölçütüdür. Deyim bir WHERE yan tümcesi içeriyorsa, Microsoft Jet veritabanı alt yapısı, kayıtlara WHERE koşullarını uyguladıktan sonra değerleri gruplandırır.
alan1, *alan2*	Kayıtları sıralamada temel alınacak alanların adıdır.

Uyarılar

ORDER BY isteğe bağlıdır. Ancak, kayıtların sıralanmış olarak görüntülenmesini istiyorsanız ORDER BY'ı kullanmalısınız.

Varsayılan sıralama düzeni artan düzendir (A'dan Z'ye, 0'dan 9'a). Aşağıdaki örneklerden her ikisi de, çalışan adlarını soyadına göre sıralar:

SELECT Soyady, Ady

FROM Çalışanlar

ORDER BY Soyady;

SELECT Soyady, Ady

```
FROM Çalışanlar

ORDER BY Soyady ASC;
```

Azalan sırada sıralamak için (Z'den A'ya, 9'dan 0'a), azalan sırada sıralamak istediğiniz her alanın sonuna DESC saklı sözcüğünü ekleyin. Aşağıdaki örnek maaşları seçer ve bunları azalan sırada sıralar:

```
SELECT Soyadı, Maaş

FROM Çalışanlar

ORDER BY Maaş DESC, Soyadı;
```

ORDER BY yan tümcesinde Not veya OLE Nesnesi verileri içeren bir alanı belirtirseniz bir hata oluşur. Microsoft Jet veritabanı alt yapısı bu bu türdeki alanları gruplandıramaz.

ORDER BY genellikle bir SQL deyiminin son sözcüğüdür.

ORDER BY yan tümcesinde ek alanlar da bulundurabilirsiniz. Kayıtlar, ORDER BY'dan sonra listelenen ilk alana göre sıralanır. Bu alanda aynı değere sahip olan kayıtlar, listelenen ikinci alana göre sıralanır ve bu böylece devam eder.

ORDER BY Yan Tümcesi Örneği

Aşağıdaki örnekte gösterilen SQL deyimi, soyadlarını azalan sırada (Z-A) sıralamak için ORDER BY yan tümcesini kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub OrderByX()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' Çalışanlar tablosundan adı ve soyadı değerlerini

    ' seçer ve bunlary azalan syrada syralar.

    '

    Set rst = dbs.OpenRecordset("SELECT Soyady, " _

        & "Adı FROM Çalışanlar " _

        & "ORDER BY Soyady DESC;")


    ' Recordset'i başlatır.

    rst.MoveLast


    ' Recordset içeriklerini yazdırmak üzere EnumFields'i çağırır.

    EnumFields rst, 12

    dbs.Close
```

End Sub

- **WHERE Yan Tümcesi**

FROM yan tümcesinde listelenen tablolardaki hangi kayıtların SELECT, UPDATE veya DELETE deyiminden etkilendiğini belirtir.

Sözdizimi

SELECT *alanlistesi*
FROM *tabloifadesi*
WHERE *ölçüt*

WHERE yan tümcesi içeren bir SELECT deyiminin bölümleri şunlardır:

Bölüm	**Açıklama**
Alanlistesi	Herhangi bir alan diğeradı, doğrulamalar (ALL, DISTINCT, DISTINCTROW veya TOP) veya diğer SELECT deyimi seçenekleri ile alınabilecek alan veya alanların adı.
Tabloifadesi	İçinden verilerin alınacağı tablo veya tabloların adıdır.
Ölçütler	Kayıtların, sorgu sonuçları içinde yer alabilmesi için sağlamaları gereken bir ifadedir.

Uyarılar

Microsoft Jet veritabanı alt yapısı, WHERE yan tümcesinde listelenen koşullara uyan kayıtları seçer. Bir WHERE yan tümcesi belirlemezseniz, sorgunuz tablodaki tüm satırları geri döndürür. Sorgunuzda birden fazla tablo belirtir ve WHERE ya da JOIN yan tümcesi bulundurmazsanız, sorgunuz tabloların bir Kartezyen çarpımını oluşturur.

WHERE isteğe bağlıdır ancak varsa FROM'un ardından gelir. Örneğin, satış bölümündeki tüm çalışanları seçebilir (WHERE Bölüm = 'Satış') veya yaşı 18 ile 30 arasındaki tüm müşterileri seçebilirsiniz (WHERE Yaş Between 18 And 30).

Çok sayıda tabloda SQL birleşim işlemini gerçekleştirmek için bir JOIN yan tümcesi kullanmazsanız, sonuç **Recordset** nesnesi güncelleştirilebilir durumda olmaz.

WHERE, HAVING yan tümcesine benzer. WHERE, hangi kayıtların seçileceğini belirler. Benzer biçimde, kayıtlar GROUP BY ile gruplandırıldıktan sonra HAVING de hangi kayıtların görüntüleneceğini belirtir.

GROUP BY yan tümcesi ile gruplandırmak istemediğiniz kayıtları elemek için WHERE yan tümcesini kullanın.

SQL deyiminin döndüreceği kayıtları belirlemek için değişik ifadeler kullanın. Örneğin aşağıdaki SQL deyimi, yıllık maaşları 600 Milyon TL'den yüksek olan çalışanları seçer.

SELECT Soyadı, Maaş

FROM Çalışanlar

WHERE Maaş > 21000;

Bir WHERE yan tümcesi, **And** ve **Or** gibi mantıksal işleçlerle birbirine bağlı en çok 40 ifade içerebilir.

Bir boşluk veya noktalama işareti içeren bir alan adı girerken, adı köşeli ayraçlar ([ ]) içine alın. Örneğin bir müşteri bilgileri tablosu, belirli müşteriler hakkında bilgiler içerebilir:

SELECT [Müşterinin En Sevdiği Lokanta]

Ölçüt değişkenini belirttiğinizde, tarih rakamları, Microsoft® Jet veritabanı alt yapısının A.B.D. sürümünü kullanmıyor olsanız bile A.B.D. biçiminde olmalıdır. Örneğin 10 Mayıs 1996, İngilizce biçiminde 10/5/96 şeklinde ve A.B.D. biçiminde 5/10/96 olarak yazılır. Tarih rakamlarını, aşağıdaki örnekte görüldüğü gibi diyez işareti (#) içine aldığınızdan emin olun.

Bir İngiltere veritabanında 10 Mayıs 1996 tarihli kayıtları bulmak için, aşağıdaki SQL deyimini kullanmalısınız.

SELECT *

FROM Siparişler

WHERE YüklemeTarihi = #5/10/96#;

Ayrıca, Microsoft Windows® tarafından belirlenen uluslararası ayarları algılayan **DateValue** işlevini de kullanabilirsiniz. Örneğin, A.B.D. için şu kodları kullanın:

SELECT *

FROM Siparişler

WHERE YüklemeTarihi = TarihDeğeri('5/10/96');

Ve İngiltere için şu kodları kullanın:

SELECT *

FROM Siparişler

WHERE YüklemeTarihi = TarihDeğeri('10/5/96');

Not Ölçüt dizesinde başvurulan sütun GUID türünde ise, ölçüt ifadesi biraz farklı bir sözdizimi kullanır:

WHERE YinelemeNo = {GUID {12345678-90AB-CDEF-1234-567890ABCDEF}}

Gösterildiği gibi içiçe parantezler ve tireler eklediğinizden emin olun.

WHERE Yan Tümcesi Örneği

Aşağıdaki örnek, Çalışanlar tablosunda Maaş alanının bulunduğunu varsayar. Bu alan gerçekte Nortwind veritabanının Çalışanlar tablosunda bulunmaz.

Bu örnek, soyadı Etikan olan kayıtların Adı ve Soyadı bilgilerini seçer.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub WhereX()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' Çalışanlar tablosundan, soyadı Etikan olan

    ' kayytlary seçer.

    Set rst = dbs.OpenRecordset("SELECT Soyady, " _

        & "Adı FROM Çalışanlar " _
```

```
        & "WHERE Soyady = 'Etikan';")


    ' Recordset'i başlatır.
    rst.MoveLast


    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır.
    EnumFields rst, 12
    dbs.Close
End Sub
```

CONSTRAINT Yan Tümcesi

Bir kısıtlama bir dizine benzer, ancak bir başka tablo ile ilişki kurmak için de kullanılabilir.

CONSTRAINT yan tümcesini ALTER TABLE ve CREATE TABLE deyimlerinde, kısıtlama oluşturmak veya silmek üzere kullanabilirsiniz. İki tür CONSTRAINT yan tümcesi vardır: tek bir alanda bir kısıtlama oluşturmak için bir tane ve birden çok alanda bir kısıtlama oluşturmak için bir tane.

Not **Microsoft Jet veritabanı alt yapısı**, Microsoft Jet veritabanları dışında ALTER TABLE kullanımını veya herhangi bir **veri tanımı dili (DDL)** deyimini desteklemez. Bunun yerine **DAO Create** yöntemlerini kullanın.

Sözdizimi

Tek alan kısıtlaması:

CONSTRAINT *adı* {PRIMARY KEY | UNIQUE | NOT NULL |
REFERENCES *yabancıtablo* [(*yabancıalan1, yabancıalan2*)]
[ON UPDATE CASCADE | SET NULL]
[ON DELETE CASCADE | SET NULL]}

Çok alan kısıtlaması:

CONSTRAINT *adı*
{PRIMARY KEY (*birincil1*[, *birincil2* [, ...]]) |
UNIQUE (*benzersiz1*[, *benzersiz2* [, ...]]) |
NOT NULL (*nulldeğil1*[, *nulldeğil2* [, ...]]) |
FOREIGN KEY [NO INDEX] (*başvuru1*[, *başvuru2* [, ...]]) REFERENCES *yabancıtablo*
[(*yabancıalan1* [, *yabancıalan2* [, ...]])]
[ON UPDATE CASCADE | SET NULL]
[ON DELETE CASCADE | SET NULL]}

CONSTRAINT yan tümcesinin bölümleri şunlardır:

Bölüm	Açıklama
adı	Oluşturulacak kısıtlamanın adı.
birincil1, *birincil2*	Birincil anahtar olarak belirlenecek alan veya alanların adı.
benzersiz1, *benzersiz2*	Benzersiz anahtar olarak belirlenecek alan veya alanların adı.

nulldeğil1, nulldeğil2	Null olmayacak şekilde kısıtlanan alan veya alanların adı.
başvuru1, *başvuru2*	Başka bir tablodaki alanlara başvuran yabancı anahtar alan veya alanlarının adıdır.
yabancıtablo	*Yabancıanahtar* tarafından belirtilen alan veya alanları içeren yabancı tablonun adıdır.
yabancıalan1, yabancıalan2	*Başvuru1* ve *başvuru2* tarafından belirtilen *yabancıtablo* içindeki alan veya alanların adıdır. Başvurulan alan *yabancıtablonun* birincil anahtarı ise bu yan tümceyi ihmal edebilirsiniz.

Uyarılar

Tek alan kısıtlaması sözdizimini, ALTER TABLE veya CREATE TABLE deyiminin alan tanımı yan tümcesinde alanın veri türünün belirtiminin hemen ardından kullanabilirsiniz.

Çok alan kısıtlaması sözdizimini, ALTER TABLE veya CREATE TABLE deyiminde alan tanımı yan tümcesinin dışında CONSTRAINT saklı sözcüğünü her kullandığınızda kullanırsınız.

CONSTRAINT yan tümcesini kullanarak, bir alanı aşağıdaki kısıtlama türlerinden biri olarak belirleyebilirsiniz:

- UNIQUE saklı sözcüğünü, bir alanı benzersiz anahtar olarak belirlemek üzere kullanabilirsiniz. Bu, tablodaki iki farklı kaydın bu alanda aynı değere sahip olamaması anlamına gelir. Herhangi bir alanı veya alan gurubunu, benzersiz olarak kısıtlayabilirsiniz. Çok alanlı bir kısıtlama bir benzersiz anahtar olarak belirlendiyse, dizindeki tüm alanların birleştirilmiş değerlerinin benzersiz olması gerekir; bu alanlardan yalnızca birindeki değer bir veya daha çok kayıtta aynı olabilir.

- PRIMARY KEY saklı sözcüklerini, bir alanı veya bir alan kümesini bir tabloda birincil anahtar olarak belirlemek üzere kullanabilirsiniz. Birincil anahtardaki tüm değerlerin benzersiz olması ve **Null** olmaması gerekir; ayrıca, bir tabloda yalnızca bir tek birincil anahtar olabilir.

  Not Zaten birincil anahtarı olan bir tabloda PRIMARY KEY kısıtlaması belirlemeyin, aksi takdirde hata oluşur.

- FOREIGN KEY saklı sözcüğünü, bir alanı yabancı anahtar olarak belirlemek üzere kullanabilirsiniz. Yabancı tablonun birincil anahtarı birden fazla alandan oluşuyorsa, başvuran tüm alanları, yabancı tablonun adını ve başvuran alanların listelendiği sırada yabancı tablonun başvurulan alanlarını listeleyen çok alanlı bir kısıtlama tanımı kullanmalısınız. Başvurulan alan veya alanlar yabancı tablonun birincil anahtarı ise, başvurulan alanları belirtmeniz gerekmez. Veritabanı alt yapısı varsayım olarak, yabancı tablonun birincil anahtarının başvurulan alanlar olduğunu varsayar.

  Yabancı anahtar kısıtlamaları, ilgili bir birincil anahtar değeri değiştiğinde gerçekleştirilecek olan belirli eylemleri tanımlar.

- CONTRAINT yan tümcesinin tanımlı olduğu tablodaki birincil anahtarda yürütülecek ilgili eylemi temel alan ve yabancı tabloda gerçekleştirilecek eylemleri belirleyebilirsiniz. Örneğin, Müşteriler tablosunda aşağıdaki tanımın yapıldığını düşünelim:

  CREATE TABLE Müşteriler (MüştNo INTEGER PRIMARY KEY, İstNo NCHAR VARYING (50))

  Siparişler tablosunda da, Müşteriler tablosunun birincil anahtarına başvuran yabancı anahtar ilişkisinin tanımlandığını düşünelim:

  CREATE TABLE Siparişler (SiparişNo INTEGER PRIMARY KEY, MüştNo INTEGER, SiparişNotları NCHAR VARYING (255), CONSTRAINT YASiparişlerMüştNo FOREIGN KEY (MüştNo) REFERENCES Müşteriler ON UPDATE CASCADE ON DELETE CASCADE

  Yabancı anahtarda hem ON UPDATE CASCADE hem de ON DELETE CASCADE yan tümcesi tanımlıdır. ON UPDATE CASCADE yan tümcesi, müşterinin numarası (MüştNo) Müşteriler

tablosunda güncelleştirildiğinde bu güncelleştirmenin Siparişler tablosunda da yapılmasını sağlar. İlgili müşteri numarası değeri içeren her sipariş, otomatik olarak yeni değer ile güncelleştirilir. ON DELETE CASCADE yan tümcesi, Müşteriler tablosunda bir müşteri silindiğinde Siparişler tablosundaki aynı müşteri numarası değerini içeren tüm satırların silinmesini sağlar.

Şimdi de, Siparişler tablosunda CASCADE eylemi yerine SET NULL eylemini kullanarak farklı bir tanım yapıldığını düşünelim:

CREATE TABLE Siparişler (SiparişNo INTEGER PRIMARY KEY, MüştNo INTEGER, SiparişNotları NCHAR VARYING (255), CONSTRAINT YASiparişlerMüştNo FOREIGN KEY (MüştNo) REFERENCES Müşteriler ON UPDATE SET NULL ON DELETE SET NULL

ON UPDATE SET NULL yan tümcesi, müşterinin numarası (MüştNo) Müşteriler tablosunda güncelleştirildiğinde, Siparişler tablosunda otomatik olarak ilgili yabancı anahtar değerlerinin NULL olarak belirlenmesini sağlar. Benzer biçimde, ON DELETE SET NULL yan tümcesi, müşteri Müşteriler tablosundan silindiğinde, Siparişler tablosunda otomatik olarak tüm ilgili yabancı anahtarların NULL olarak belirlenmesini sağlar.

Yabancı anahtarlar için dizinlerin otomatik olarak oluşturulmasını engellemek için, NO INDEX değiştiricisi kullanılabilir. Bu yabancı anahtar tanımı şekli, ancak sonuç dizin değerleri sık sık yinelenecekse kullanılmalıdır. Yabancı anahtar dizinindeki değerlerin sık sık yinelendiği durumlarda, bir dizin kullanmak, basit biçimde bir tablo tarama işlemi yapmaktan daha az verimli olabilir. Tablodan satırların silindiği ve tabloya satırların eklendiği bu tür bir dizini tutmak, başarımı düşürür ve herhangi bir yarar sağlamaz.

PROCEDURE Yan Tümcesi

Sorgu için bir ad ve isteğe bağlı parametreler tanımlar.

Not PROCEDURE yan tümcesi PROCEDURE deyiminden sonra gelir. PROCEDURE yan tümcesi desteklenir ancak; PROCEDURE deyimi PROCEDURE yan tümcesinin yeteneklerinin bir üst kümesini sağlar ve önerilen sözdizimidir.

Sözdizimi

PROCEDURE *adı* [*parametre1 veritürü*[, *parametre2 veritürü*[, ...]]

PROCEDURE yan tümcesinin bölümleri şunlardır:

Bölüm	**Açıklama**
Adı	Yordamın adıdır. Standart adlandırma kurallarına uymalıdır.
param1, *param2*	Bir veya daha fazla alan adı veya parametredir. Örneğin: PROCEDURE Ülkelere_Göre_Satışlar [Başlangıç Tarihi] TarihSaat, [Bitiş Tarihi] TarihSaat; Parametreler hakkında ayrıntılı bilgi için parametreler konusuna bakın.
Veritürü	Birincil Microsoft Jet SQL veri türleri veya onların eşanlamlılarından biridir.

Uyarılar

Bir SQL yordamı, yordamın adını belirten PROCEDURE yan tümcesinden, isteğe bağlı bir parametre tanımları listesinden ve tek bir SQL deyiminden oluşur. Örneğin, Parça_Numarasını_Al yordamı, belirtilen parça numarasını alan bir sorgu çalıştırabilir.

Notlar

- Yan tümce birden fazla alan tanımı içeriyorsa (yani *parametre-veritürü* çiftse), bunları virgüllerle ayırın.
- PROCEDURE yan tümcesinin ardından bir SQL deyimi (örneğin bir SELECT veya UPDATE statement) gelmelidir.

CREATE PROCEDURE Deyimi, PROCEDURE Yan Tümcesi Örneği

Bu örnek, sorguyu KategoriListesi olarak adlandırır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub ProcedureX()
    Dim dbs As Database, rst As Recordset
    Dim qdf As QueryDef, strSql As String

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")

    strSql = "PROCEDURE KategoriListesi; " _
        & "SELECT DISTINCTROW KategoriAdy, " _
        & "KategoriNo FROM Kategoriler " _
        & "ORDER BY KategoriAdy;"

    ' SQL deyimini temel alan bir adlandırılmış
    ' bir QueryDef oluşturur.
    Set qdf = dbs.CreateQueryDef("YeniSorgu", strSql)
    ' Geçici bir anlık görüntü türünde Recordset oluşturur.
    Set rst = qdf.OpenRecordset(dbOpenSnapshot)
    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 15
```

```
    ' Bu bir örnek olduğu için QueryDef'i
    ' siler.
    dbs.QueryDefs.Delete "YeniSorgu"

    dbs.Close
End Sub
```

- **İşlemler**
  - **UNION İşlemi**

İki veya daha fazla bağımsız sorgu veya tablonun sonuçlarını birleştiren bir birleşim sorgusu oluşturur.

Sözdizimi

[TABLE] *sorgu1* UNION [ALL] [TABLE] *sorgu2* [UNION [ALL] [TABLE] *sorgun* [...]]

UNION işleminin bölümleri şunlardır:

Bölüm	Açıklama
sorgu1-n	Bir SELECT deyimi, saklı bir sorgunun adı veya TABLE anahtar sözcüğünden sonra gelen saklı bir tablonun adıdır.

Uyarılar

İki veya daha fazla sorgunun, tablonun ve SELECT deyiminin sonuçlarını herhangi bir birleşimde tek bir UNION işleminde birleştirebilirsiniz. Aşağıdaki örnek, Yeni Hesaplar adlı var olan bir tablo ile bir SELECT deyimini birleştirir:

TABLE [Yeni Hesaplar] UNION ALL

SELECT *

FROM Müşteriler

WHERE SiparişMiktarı >= .1000);

Varsayım olarak, UNION işlemini kullandığınızda yinelenen kayıt döndürülmez; ancak, tüm kayıtların döndürüldüğünden emin olmak için ALL doğrulamasını bulundurabilirsiniz. Bu ayrıca, sorguyu daha hızlandırır.

UNION işlemindeki tüm sorgular aynı sayıda alan istemelidir; ancak, alanların aynı boyutta veya aynı veri türünde olması gerekmez.

Diğeradları yalnızca ilk SELECT deyiminde kullanın; bunlar diğer deyimlerde dikkate alınmaz. ORDER BY yan tümcesinde alanlara, ilk SELECT deyiminde adlandırıldıkları şekilde başvurun.

Notlar

- Bir GROUP BY veya HAVING yan tümcesini döndürülen verileri gruplandırmak üzere her *sorgu* değişkeninde kullanabilirsiniz.
- ORDER BY yan tümcesini, döndürülen verileri belirli bir sırada görüntülemek için son *sorgu* değişkeninin sonunda kullanabilirsiniz.

UNION İşlemi Örneği

Bu örnek, Brezilya'daki tüm sağlayıcıların şehirlerini ve adlarını alır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub UnionX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")

    ' Brezilya'daki tüm sağlayıcıların ve müşterilerin
    ' adlarını ve şehirlerini alır.
    Set rst = dbs.OpenRecordset("SELECT ŞirketAdı," _
        & " Şehir FROM Sağlayıcılar" _
        & " WHERE Ülke = 'Brezilya' UNION" _
        & " SELECT ŞirketAdı, Şehir FROM Müşteriler" _
        & " WHERE Ülke = 'Brazil';")

    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 12
    dbs.Close
End Sub
```

- **INNER JOIN İşlemi**

Ortak bir alanda eşleşen değerler olduğunda iki tablonun kayıtlarını birleştirir.

Sözdizimi

FROM *tablo1* INNER JOIN *tablo2* ON *tablo1.alan1 karşılaştırma tablo2.alan2*

INNER JOIN işleminin bölümleri şunlardır:

Bölüm	Açıklama
tablo1, *tablo2*	İçinden kayıtların birleştirileceği tabloların adıdır.
alan1, *alan2*	Birleştirilen alanların adıdır. Alanlar sayısal değilse, aynı veri türünde olmalı ve aynı türde veriler içermelidir, ancak alanların aynı ada sahip olmaları gerekmez.
karşılaştırma	Herhangi bir karşılaştırma işlecidir: "=," "<," ">," "<=," ">=," or "<>."

Uyarılar

INNER JOIN işlemini herhangi bir FROM yan tümcesinde kullanabilirsiniz. Bu, en sık kullanılan birleştirme türüdür. İç birleşimler, iki tabloda ortak olan bir alanda eşleşen değerler olduğunda her iki tablonun kayıtlarını birleştirirler.

INNER JOIN'i, Bölümler ve Çalışanlar tablolarında, bölümlerdeki tüm çalışanları seçmek için kullanabilirsiniz. Buna karşılık, tüm bölümleri (bölüme atanmış çalışan olmasa bile) veya tüm çalışanları (herhangi bir bölüme atanmamış olsa bile) seçmek için, bir dış birleşim oluşturmak üzere LEFT JOIN veya RIGHT JOIN işlemini kullanabilirsiniz.

Not veya OLE Nesnesi verileri içeren alanları birleştirmeyi denerseniz bir hata oluşur.

Benzer türdeki herhangi iki sayısal alanı birleştirebilirsiniz. Örneğin, benzer türlerde olduklarında OtomatikSayı ve Uzun alanlarını birleştirebilirsiniz. Ancak, Tek ve Çift türündeki alanları birleştiremezsiniz.

Aşağıdaki örnek, Kategoriler ve Ürünler tablolarını KategoriNo alanına göre nasıl birleştirebileceğinizi gösterir:

```
SELECT KategoriAdy, ÜrünAdy

FROM Kategoriler INNER JOIN Ürünler

ON Kategoriler.KategoriNo = Ürünler.KategoriNo;
```

Önceki örnekte KategoriNo, birleştirilen alandır ancak SELECT deyiminde yer almadığı için sorgu sonuçlarında bulunmaz. Birleştirilen alanı bulundurmak için, alan adını SELECT deyiminde yazın (örneğimizde Kategoriler.KategoriNo).

Aşağıdaki sözdizimini kullanarak, JOIN deyimi içinde çok sayıda ON yan tümcesini de bağlayabilirsiniz:

```
SELECT alanlar
FROM tablo1 INNER JOIN tablo2
ON tablo1.alan1 karşılaştırma tablo2.alan1 AND
ON tablo1.alan2 karşılaştırma tablo2.alan2) OR
ON tablo1.alan3 karşılaştırma tablo2.alan3)];
```

Aşağıdaki sözdizimini kullanarak içiçe JOIN deyimleri de oluşturabilirsiniz:

```
SELECT alanlar
FROM tablo1 INNER JOIN
(tablo2 INNER JOIN [( ]tablo3
[INNER JOIN [( ]tablox [INNER JOIN ...)]
ON tablo3.alan3 karşılaştırma tablox.alanx)]
ON tablo2.alan2 karşılaştırma tablo3.alan3)
ON tablo1.alan1 karşılaştırma tablo2.alan2;
```

Bir LEFT JOIN veya bir RIGHT JOIN bir INNER JOIN içinde içiçe yer alabilir ancak bir INNER JOIN bir LEFT JOIN veya RIGHT JOIN içinde yer alamaz.

INNER JOIN İşlemi Örneği

Bu örnek iki benzer-birleşim oluşturur: biri Sipariş Ayrıntıları ve Siparişler tabloları arasında, diğeri Siparişler ve Çalışanlar tabloları arasında. Bu, Çalışanlar tablosu satış verilerini içermediği ve

Sipariş Ayrıntıları tablosu da çalışan verilerini içermediği için gereklidir. Sorgu, bir çalışan listesi ve bu çalışanların toplam satışlarını verir.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub InnerJoinX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")

    ' Sipariş Ayrıntıları ve Siparişler tablosu arasında
    ' bir birleşim ve Siparişler ile Çalışanlar tabloları
    ' arasında da bir birleşim oluşturur. Çalışanların listesini
    ' ve onların toplam satış miktarlarını alır.
    Set rst = dbs.OpenRecordset("SELECT DISTINCTROW " _
        & "Sum(BirimFiyatı * Miktar) AS Satışlar, " _
        & "(Ady & Chr(32) & Soyady) AS Ad " _
        & "FROM Çalışanlar INNER JOIN(Siparişler " _
        & "INNER JOIN [Sipariş Ayrıntıları] " _
        & "ON [Sipariş Ayrıntıları].SiparişNo = " _
        & "Siparişler.SiparişNo ) " _
        & "ON Siparişler.ÇalışanNo = " _
        & "Çalışanlar.ÇalışanNo " _
        & "GROUP BY (Ady & Chr(32) & Soyady);")

    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 20
    dbs.Close
End Sub
```

- **LEFT JOIN, RIGHT JOIN İşlemleri**

Herhangi bir FROM yan tümcesinde kullanıldığında kaynak tablo kayıtlarını birleştirir.

Sözdizimi

FROM *tablo1* [LEFT | RIGHT] JOIN *tablo2*
ON *tablo1.alan1 karşılaştırma tablo2.alan2*

LEFT JOIN ve RIGHT JOIN işlemlerinin bölümleri şunlardır:

Bölüm	Açıklama
tablo1, *tablo2*	İçinden kayıtların birleştirileceği tabloların adıdır.
alan1, *alan2*	Birleştirilen alanların adıdır. Alanlar aynı veri türünde olmalı ve aynı türde veriler içermelidir, ancak alanların aynı ada sahip olmaları gerekmez.
karşılaştırma	Herhangi bir karşılaştırma işlecidir: "=," "<," ">," "<=," ">=," or "<>."

Uyarılar

Bir sol dış birleşim oluşturmak için LEFT JOIN işlemini kullanın. Sol dış birleşimler, ikinci tablonun (sağdaki) kayıtlarında eşleşen değer olmasa bile, iki tablodan ilkinin (soldaki) tüm kayıtlarını içerir.

Bir sağ dış birleşim oluşturmak için RIGHT JOIN işlemini kullanın. Sağ dış birleşimler, ilk tablonun (soldaki) kayıtlarında eşleşen değer olmasa bile, iki tablodan ikincisinin (sağdaki) tüm kayıtlarını içerir.

Örneğin, Bölümler (sol) ve Çalışanlar (sağ) tabloları ile, bölüme atanmış çalışan olmasa bile tüm bölümleri seçmek için LEFT JOIN işlemini kullanabilirsiniz. Herhangi bir bölüme atanmamış olanlar da dahil, tüm çalışanları seçmek için RIGHT JOIN'i kullanmalısınız.

Aşağıdaki örnek, Kategoriler ve Ürünler tablolarını KategoriNo alanına göre nasıl birleştirebileceğinizi gösterir. Sorgu, ürün içermeyenler de dahil olmak üzere tüm kategorilerin bir listesini oluşturur:

SELECT KategoriAdy,

ÜrünAdy

FROM Kategoriler LEFT JOIN Ürünler

ON Kategoriler.KategoriNo = Ürünler.KategoriNo;

Bu örnekte KategoriNo, birleştirilen alandır ancak SELECT deyiminde yer almadığı için sorgu sonuçlarında bulunmaz. Birleştirilen alanı bulundurmak için, alan adını SELECT deyiminde yazın (örneğimizde Kategoriler.KategoriAdı).

Notlar

Yalnızca verileri birleştirilmiş alanlardaki verilerle aynı olan kayıtları bulunduran bir sorgu oluşturmak için, bir INNER JOIN işlemi kullanın.

- Bir LEFT JOIN veya bir RIGHT JOIN bir INNER JOIN içinde içiçe yer alabilir ancak bir INNER JOIN bir LEFT JOIN veya RIGHT JOIN içinde yer alamaz. Birleşimlerin diğer birleşimlerin içine nasıl konulabileceği konusunda, INNER JOIN konusundaki içiçe yerleştirme tartışmasına bakın.

- Çok sayıda ON yan tümcesini bağlayabilirsiniz. Bunun nasıl yapıldığı konusunda, INNER JOIN konusundaki yan tümce bağlama tartışmasına bakın.

Not veya OLE Nesnesi verileri içeren alanları birleştirmeyi denerseniz bir hata oluşur.

LEFT JOIN, RIGHT JOIN İşlemleri Örneği

Bu örnek, Çalışanlar tablosunda Bölüm Adı ve Bölüm No alanlarının bulunduğunu varsayar. Bu alanlar gerçekte Nortwind veritabanının Çalışanlar tablosunda bulunmaz.

Bu örnek, içinde çalışan bulunmayanlar dahil tüm bölümleri seçer.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub LeftRightJoinX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")

    ' Çalışanı bulunmayanlar dahil tüm bölümleri
    ' seçer.
    Set rst = dbs.OpenRecordset _
        ("SELECT [Bölüm Ady], " _
        & "Ady & Chr(32) & Soyady AS Ad " _
        & "FROM Bölümler LEFT JOIN Çalışanlar " _
        & "ON Bölümler.[Bölüm No] = " _
        & "Çalışanlar.[Bölüm No] " _
        & "ORDER BY [Bölüm Ady];")

    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 20
    dbs.Close
End Sub
```

- **İşleçler**
  - **Like İşleci**

Bir dize ifadesi ile bir SQL ifadesindeki bir örnek ile karşılaştırır.

Sözdizimi

***ifade* Like "*örnek*"**

Like işlecinin bölümleri şunlardır:

Bölüm	Açıklama
ifade	WHERE yan tümcesinde kullanılan SQL ifadesidir.
örnek	*İfadenin* karşılaştırılacağı dize veya karakter metinleridir.

Uyarılar

Like işlecini, belirlediğiniz örnek ile eşleşen alan değerlerini bulmak için kullanabilirsiniz. *Örnek* için, kesin bir değer belirtebilir (örneğin, Like "gamze") veya bir değer aralığını bulmak için joker karakterlerini kullanabilirsiniz (örneğin, Like "Ga*").

Bir ifadede **Like** işlecini, bir alan değerini bir dize ifadesi ile karşılaştırmak için kullanabilirsiniz. Örneğin bir SQL sorgusunda Like "C*" yazarsanız, sorgu, C harfi ile başlayan tüm alan değerlerini verir. Bir parametre sorgusunda, kullanıcıya aranacak örneği sorabilirsiniz.

Aşağıdaki örnek, P harfi ile başlayan ve A ile F arasında bir harf ile devam eden ve toplam üç hane olan verileri verir:

Like "P[A-F]###"

Aşağıdaki tablo, ifadeleri farklı örneklerle sınamak için **Like** işlecini nasıl kullanacağınızı gösterir.

Eşleme türü	Örnek	Eşleme (True verir)	Eşleme yok (False verir)
Çok sayıda karakter	a*a	aa, aBa, aBBBa	aBC
	ab	abc, AABB, Xab	aZb, bac
Özel karakter	a[*]a	a*a	aaa
Çok sayıda karakter	ab*	abcdefg, abc	cab, aab
Tek karakter	a?a	aaa, a3a, aBa	aBBBa
Tek hane	a#a	a0a, a1a, a2a	aaa, a10a
Karakter aralığı	[a-z]	f, p, j	2, &
Aralık dışı	[!a-z]	9, &, %	b, a
Hane değil	[!0-9]	A, a, &, ~	0, 1, 9
Birleşik	a[!b-m]#	An9, az0, a99	abc, aj0

Like İşleci Örneği

Bu örnek, adları A ile D harfleri arasındaki harflerle başlayan çalışanların listesini verir.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub LikeX()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' Adları A ile D arasındaki harflerle başlayan

    ' çalışanların listesini verir.

    Set rst = dbs.OpenRecordset("SELECT Soyady," _
```

```
        & "Adı FROM Çalışanlar" _
        & " WHERE Soyady Like '[A-D]*';")
    ' Recordset'i başlatır.
    rst.MoveLast
    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 15

    dbs.Close
End Sub
```

- **Between...And İşleci**

Bir ifadenin değerinin, belirtilen değer aralığı içinde olup olmadığını belirler. Bu işleci SQL deyimleri içinde kullanabilirsiniz.

Sözdizimi

ifade* [Not] Between *değer1* And *değer2

Between...And işlecinin bölümleri şunlardır:

Bölüm	Açıklama
ifade	Değerlendirmek istediğiniz verileri içeren alanı belirten ifadedir.
değer1, *değer2*	*İfadeyi* karşılaştırmak istediğiniz değerlerdir.

Uyarılar

İfadenin değeri *değer1* ile *değer2* arasında ise (bu değerler dahil), **Between...And** işleci **True** değerini aksi takdirde **False** değerini verir. Karşıt koşulu değerlendirmek (*ifadenin, değer1* ve *değer2* ile tanımlanan aralığın dışında kalması durumu) için **Not** mantıksal işlecini bulundurabilirsiniz.

Bir alanın değerinin, belirtilen sayısal aralık içinde olup olmadığını belirlemek için **Between...And** işlecini kullanabilirsiniz. Aşağıdaki örnek, bir siparişin belirli bir posta kodu aralığındaki adrese gönderilip gönderilmediğini belirler. Posta kodu 98101 ve 98199 arasında ise, **IIf** işlevi "Yerel" değerini verir. Aksi takdirde, "Yerel Değil" değerini verir.

```
SELECT IIf(PostaKodu Between 98101 And 98199, “Yerel”, “Yerel Değil”)
FROM Yayyncylar
```

İfade, *değer1* veya *değer2* **Null** ise **Between...And**, **Null** değerini verir.

* gibi joker karakterleri bir düz metin olarak algılandıklarından, bunları **Between...And** işleci ile kullanamazsınız. Örneğin 980 ile 989 rakamları ile başlayan posta kodlarını bulmak için 980* ve 989* yazamazsınız. Bunun yerine, bu işlemi yapmak için iki seçeneğiniz vardır. Sorguya, metin alanının en soldaki üç karakterini alan bir ifade ekler ve **Between...And** işlecini bu karakterlerde kullanırsınız. Ya da, en yüksek ve en düşük değerleri ek karakterler ile belirtirsiniz; bu durumda 98000 ile 98999 veya uzun posta kodları kullanılıyorsa 98000 ile 98999 - 9999 arası. (Düşük

değerlerde – 0000 yazmamalısınız; aksi takdirde, bazı posta kodlarında uzun posta kodu varsa diğerlerinde ise yoksa 98000 bırakılır.)

- **In İşleci**

Bir ifadenin değerinin, belirtilen listedeki değerlerden herhangi birine eşit olup olmadığını belirler.

Sözdizimi

ifade **[Not] In(*değer1, değer2, . . .*)**

Uyarılar

In işlecinin bölümleri şunlardır:

Bölüm	**Açıklama**
İfade	Değerlendirmek istediğiniz verileri içeren alanı belirten ifadedir.
değer1, *değer2*	*İfadeyi* karşılaştırmak istediğiniz değer veya değer listesidir.

İfade değer listesinde bulunursa*,* **In** işleci **True** değerini aksi takdirde **False** yanlış değerini döndürür. Karşıt koşulu değerlendirmek (*ifadenin,* değer listesi içinde bulunmaması durumu) için **Not** mantıksal işlecini bulundurabilirsiniz.

Örneğin, belirtilen bölgeler topluluğuna hangi siparişlerin gönderilmiş olduğunu belirlemek için **In** işlecini kullanabilirsiniz:

SELECT *

FROM Siparişler

WHERE TeslimBölgesi In (‘Avon’,’Glos’,’Som’)

In İşleci Örneği

Aşağıdaki örnek, İstanbul ve Ankara'ya teslim edilen tüm siparişleri ve bunların teslim tarihlerini içeren bir sorgu oluşturmak için, Northwind veritabanındaki Siparişler tablosunu kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub InX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' Siparişler tablosundan, TeslimYeri İstanbul
    ' veya Ankara olan kayytlary seçer.
    Set rst = dbs.OpenRecordset("SELECT " _
        & "MüşteriNo, TeslimTarihi FROM Siparişler " _
        & "WHERE TeslimYeri In " _
        & "('Istanbul','Ankara');")
```

```
    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır.
    EnumFields rst, 12
    dbs.Close
End Sub
```

SQL TOPLAM İŞLEVLERİ

SQL toplam işlevlerini kullanarak, değer kümelerinde çeşitli istatistikler belirleyebilirsiniz. Bu işlevleri bir sorguda ve toplam ifadelerini bir **QueryDef** nesnesinin **SQL** özelliğinde veya bir SQL sorgusunu temel alan bir **Recordset** nesnesi oluştururken kullanabilirsiniz.

➢ Avg İşlevi

Sorguda belirtilen alanda bulunan değer kümesinin aritmetik ortalamasını hesaplar.

Sözdizimi

Avg(*ifade*)

İfade yertutucusu, ortalamasını almak istediğiniz sayısal verileri içeren alanı belirten bir dizi ifadesini veya bu alandaki verileri kullanarak bir hesaplama yapan bir ifadeyi temsil eder. *İfade* içindeki işleçler, tablo alanının adını, bir sabiti veya bir işlevi (bilinen veya kullanıcı tanımlı bir işlev olabilir ancak diğer SQL toplam işlevlerinden biri olamaz) içerebilir.

Uyarılar

Avg ile hesaplanan ortalama, aritmetik bir ortalamadır (değerlerin toplamının değer sayısına bölümüdür). **Avg**'yi örneğin, ortalama navlun maliyetlerini hesaplamak için kullanabilirsiniz.

Avg işlevi hesaplamada herhangi bir **Null** alanı içermez.

Avg işlevini bir sorgu ifadesinde ve bir **QueryDef** nesnesinin **SQL** özelliğinde veya bir SQL sorgusunu temel alan bir **Recordset** nesnesi oluştururken kullanabilirsiniz.

Avg İşlevi Örneği

Bu örnek, navlun bedeki 100 milyonun üzerindeki siparişlerin ortalama navlun bedellerini hesaplamak için Siparişler tablosunu kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub AvgX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
```

```
    ' Navlun bedeli 100 milyonun üzerindeki siparişlerin
    ' ortalama navlun bedellerini hesaplar.
    Set rst = dbs.OpenRecordset("SELECT Avg(Navlun)" _
       & " AS [Ortalama Navlun]" _
       & " FROM Siparişler WHERE Navlun > 100;")

    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 25
    dbs.Close
End Sub
```

- **Count İşlevi**

Sorgunun döndürdüğü kayıt sayısını hesaplar.

Sözdizimi

Count(*ifade*)

İfade yertutucusu, saymak istediğiniz sayısal verileri içeren alanı belirten bir dizi ifadesini veya bu alandaki verileri kullanarak bir hesaplama yapan bir ifadeyi temsil eder. *İfade* içindeki işleçler, tablo alanının adını veya bir işlevi (bilinen veya kullanıcı tanımlı bir işlev olabilir ancak diğer SQL toplam işlevlerinden biri olamaz) içerebilir. Metin dahil, herhangi bir türdeki verileri sayabilirsiniz.

Uyarılar

Temel alınan sorgudaki kayıt sayısını saymak için **Count**'u kullanabilirsiniz. Örneğin belirli bir ülkeye yapılmış olan siparişlerin sayısını saymak için **Count**'u kullanabilirsiniz.

Her ne kadar *ifade* de alan üzerinde hesap yapabilirse de, **Count**, basit biçimde kayıt sayısını hesaplar. Kayıtlarda hangi değerlerin saklandığı ile ilgilenmez.

İfade yıldız şeklinde (*) bir joker karakteri değilse, **Count** işlevi **Null** değerine sahip olan kayıtları saymaz. unless . Bir yıldız kullanırsanız, **Count**, **Null** alanları içeren kayıtlar da dahil olmak üzere toplam kayıt sayısını hesaplar. **Count(***)**, **Count(**[*Sütun Adı*]**)** işlevinden belirgin biçimde daha hızlıdır. Yıldız karakterini tırnak imleri (' ') içine almayın. Aşağıdaki örnek, Siparişler tablosundaki kayıt sayısını hesaplar.

SELECT Count(*)

AS ToplamSiparişler FROM Siparişler;

İfade çok sayıda alanı belirtiyorsa, **Count** işlevi yalnızca, alan değerlerinden en az biri **Null** olmayan kayıtları sayar. Belirtilen alanların tümü **Null** ise, kayıt sayılmaz. Alan adlarını & simgesi

ile ayırın. Aşağıdaki örnek, sayma işlemini, SevkTarihi veya Navlun değerlerinin **Null** olmaması durumu ile nasıl sınırlandırdığınızı gösterir:

SELECT

Count('SevkTarihi & Navlun')

AS [Not Null] FROM Siparişler;

Count'u sorgu ifadesinde kullanabilirsiniz. Bu ifadeyi ayrıca, bir **QueryDef** nesnesinin **SQL** özelliğinde veya bir SQL sorgusunu temel alan bir **Recordset** nesnesi oluştururken kullanabilirsiniz.

Count İşlevi Örneği

Bu örnek, İngiltere'ye teslim edilen sipariş sayısını hesaplamak için Siparişler tablosunu kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub CountX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")

    ' İngiltere'ye yapılmış teslimlerin sayısını
    ' hesaplar.
    Set rst = dbs.OpenRecordset("SELECT" _
        & " Count (TeslimÜlkesi)" _
        & " AS [İngiltere Siparişleri] FROM Siparişler" _
        & " WHERE TeslimÜlkesi = 'Yngiltere';")

    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 25
    dbs.Close
End Sub
```

- **First, Last İşlevleri**

Sorgunun döndürdüğü sonuç kümesindeki ilk veya son kaydın alan değerini döndürür.

Sözdizimi

First(*ifade*)

Last(*ifade*)

İfade yertutucusu, kullanmak istediğiniz sayısal verileri içeren alanı belirten bir dize ifadesini veya bu alandaki verileri kullanarak bir hesaplama yapan bir ifadeyi temsil eder. *İfade* içindeki işleçler, tablo alanının adını, bir sabiti veya bir işlevi (bilinen veya kullanıcı tanımlı bir işlev olabilir ancak diğer SQL toplam işlevlerinden biri olamaz) içerebilir.

Uyarılar

First ve **Last** işlevleri, DAO Recordset nesnesinin **MoveFirst** ve **MoveLast** yöntemlerine benzer. Basit biçimde, sorgunun döndürdüğü sonuç kümesindeki ilk veya son kaydın alan değerini döndürür. Kayıtlar genellikle belirli bir sırada döndürülmediklerinden (sorgu içinde ORDER BY yan tümcesi yoksa), bu işlevlerin döndürdüğü kayıtlar düzensizdir.

First, Last İşlevleri Örneği

Bu örnek, Çalışanlar tablosundaki ilk ve son kayıtların Soyadı alanlarının değerlerini vermek üzere Çalışanlar tablosunu kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub FirstLastX1()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")


    ' Tablonun ilk ve son kayytlarynyn Soyady

    ' alanının değerini verir.

    Set rst = dbs.OpenRecordset("SELECT " _

        & "First(Soyady) as Ylk, " _

        & "Last(Soyadı) as Son FROM Çalışanlar;")


    ' Recordset'i başlatır.

    rst.MoveLast


    ' Recordset içeriklerini yazdyrmak üzere EnumFields

    ' yordamını çağırır. Recordset nesnesini ve

    ' istenen alan genişliğini aktarır.

    EnumFields rst, 12

    dbs.Close
```

```
End Sub
```

Bir sonraki örnek, Çalışanların ilk ve son doğum tarihlerini bulmak üzere basit bir şekilde **Min** ve **Max** işlevlerini kullanmak ile, **First** ve **Last** işlevlerini kullanmayı karşılaştırır.

```
Sub FirstLastX2()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")


    ' Çalışanların ilk ve son doğum tarihlerini

    ' bulur.

    Set rst = dbs.OpenRecordset("SELECT " _

        & "First(DoğumTarihi) as İlkDT, " _

        & "Last(DoğumTarihi) as SonDT FROM Çalışanlar;")


    ' Recordset'i başlatır.

    rst.MoveLast


    ' Recordset içeriklerini yazdyrmak üzere EnumFields

    ' yordamını çağırır. Recordset nesnesini ve

    ' istenen alan genişliğini aktarır.

    EnumFields rst, 12


    Debug.Print

    ' Çalışanların ilk ve son doğum tarihlerini

    ' bulur.

    Set rst = dbs.OpenRecordset("SELECT " _
        & "Min(DoğumTarihi) as EnKüçükDT," _

        & "Max(DoğumTarihi) as EnBüyükDT FROM Çalışanlar;")


    ' Recordset'i başlatır.

    rst.MoveLast


    ' Recordset içeriklerini yazdyrmak üzere EnumFields
```

```
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 12
    dbs.Close
End Sub
```

- **Min, Max İşlevleri**

Sorguda belirtilen alanda bulunan değer kümelerinin en büyük veya en küçük değerlerini hesaplar.

Sözdizimi

Min(*ifade*)

Max(*ifade*)

İfade yertutucusu, kullanmak istediğiniz verileri içeren alanı belirten bir dize ifadesini veya bu alandaki verileri kullanarak bir hesaplama yapan bir ifadeyi temsil eder. *İfade* içindeki işleçler, tablo alanının adını, bir sabiti veya bir işlevi (bilinen veya kullanıcı tanımlı bir işlev olabilir ancak diğer SQL toplam işlevlerinden biri olamaz) içerebilir.

Uyarılar

Min ve **Max**'ı, belirtilen toplamı veya gruplandırmayı temel alan alandaki en küçük veya en büyük değerleri belirlemek için kullanabilirsiniz. Örneğin, bu işlevleri, en az ve az çok navlun bedellerini bulmak için kullanabilirsiniz. Herhangi bir toplam işlevi belirtilmezse, tablonun tamamı kullanılır.

Min ve **Max** işlevlerini bir sorgu ifadesinde ve bir **QueryDef** nesnesinin **SQL** özelliğinde veya bir SQL sorgusunu temel alan bir **Recordset** nesnesi oluştururken kullanabilirsiniz.

Min, Max İşlevleri Örneği

Bu örnek, İngiltere'ye teslim edilen siparişlerin en az ve en çok navlun bedellerini vermek için Siparişler tablosunu kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub MinMaxX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")

    ' İngiltere'ye teslim edilmiş olan siparişlerin en az
    ' ve en çok navlun bedelini verir.
    Set rst = dbs.OpenRecordset("SELECT " _
        & "Min(Navlun) AS [En Az Navlun], " _
        & "Max(Navlun)AS [En Çok Navlun] " _
        & "FROM Siparişler WHERE TeslimÜlkesi = 'İngiltere';"
```

```
    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 12
    dbs.Close
End Sub
```

- **StDev, StDevP İşlevleri**

Sorgunun belirtilen bir alanında bulunan değerler kümesi olarak temsil edilen bir grubun veya grup örneğinin yaklaşık standart sapmasını verir.

Sözdizimi

StDev(*ifade*)

StDevP(*ifade*)

İfade yertutucusu, kullanmak istediğiniz sayısal verileri içeren alanı belirten bir dize ifadesini veya bu alandaki verileri kullanarak bir hesaplama yapan bir ifadeyi temsil eder. *İfade* içindeki işleçler, tablo alanının adını, bir sabiti veya bir işlevi (bilinen veya kullanıcı tanımlı bir işlev olabilir ancak diğer SQL toplam işlevlerinden biri olamaz) içerebilir.

Uyarılar

StDevP işlevi bir grubu değerlendirir ve **StDev** işlevi bir grup örneğini değerlendirir.

Temel alınan sorgu ikiden az kayıt içeriyorsa (veya **StDevP** işlevi için kayıt içermiyorsa), bu işlevler bir **Null** değeri döndürürler (bu da, standart sapmanın hesaplanamayacağını belirtir).

StDev ve **StDevP** işlevlerini bir sorgu ifadesinde kullanabilirsiniz. Bu ifadeyi ayrıca, bir **QueryDef** nesnesinin **SQL** özelliğinde veya bir SQL sorgusunu temel alan bir **Recordset** nesnesi oluştururken kullanabilirsiniz.

StDev, StDevP İşlevleri Örneği

Bu örnek, İngiltere'ye teslim edilen siparişlerin navlun bedellerinin standart sapmanı tahmin etmek için Siparişler tablosunu kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub StDevX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' İngiltere'ye teslim edilmiş olan siparişlerin navlun
```

```
    ' bedellerinin standart sapmasyny hesaplar.
    Set rst = dbs.OpenRecordset("SELECT " _
        & "StDev(Navlun) " _
        & "AS [Navlun Dev] FROM Siparişler " _
        & "WHERE TeslimÜlkesi = 'Yngiltere';")
    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 15

    Debug.Print

    Set rst = dbs.OpenRecordset("SELECT " _
        & "StDevP(Navlun) " _
        & "AS [Navlun DevP] FROM Siparişler " _
        & "WHERE TeslimÜlkesi = 'Yngiltere';")
    ' Recordset'i başlatır.
    rst.MoveLast
    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 15
    dbs.Close
End Sub
```

- **Sum İşlevi**

Sorguda belirtilen alanda bulunan değer kümesinin toplamını hesaplar.

Sözdizimi

Sum(*ifade*)

İfade yertutucusu, eklemek istediğiniz sayısal verileri içeren alanı belirten bir dize ifadesini veya bu alandaki verileri kullanarak bir hesaplama yapan bir ifadeyi temsil eder. *İfade* içindeki işleçler, tablo

alanının adını, bir sabiti veya bir işlevi (bilinen veya kullanıcı tanımlı bir işlev olabilir ancak diğer SQL toplam işlevlerinden biri olamaz) içerebilir.

Uyarılar

Sum işlevi, bir alandaki değerleri toplar. Örneğin, **Sum** işlevini, navlun bedellerinin toplamını bulmak için kullanabilirsiniz.

Sum işlevi, **Null** alanlarını içeren kayıtları yoksayar. Aşağıdaki örnek, ürünlerin BirimFiyatı ve Miktar bilgilerinin toplamını nasıl toplayabileceğinizi gösterir:

SELECT

Sum(BirimFiyaty * Miktar)

AS [Toplam Gelir] FROM [Sipariş Ayrıntıları];

Sum işlevini bir sorgu ifadesinde kullanabilirsiniz. Bu ifadeyi ayrıca, bir **QueryDef** nesnesinin **SQL** özelliğinde veya bir SQL sorgusunu temel alan bir **Recordset** nesnesi oluştururken kullanabilirsiniz.

Sum İşlevi Örneği

Bu örnek, İngiltere'ye teslim edilen siparişlerin toplam satış miktarını hesaplamak için Siparişler tablosunu kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub SumX()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' İngiltere'ye teslim edilmiş siparişlerin toplam

    ' satış miktarını hesaplar.

    Set rst = dbs.OpenRecordset("SELECT" _

        & " Sum(BirimFiyaty*Miktar)" _

        & " AS [Toplam İngiltere Satışları] FROM Siparişler" _

        & " INNER JOIN [Sipariş Ayrıntıları] ON" _

        & " Siparişler.SiparişNo = [Sipariş Ayrıntıları].SiparişNo" _

        & " WHERE (TeslimÜlkesi = 'Yngiltere');")

    ' Recordset'i başlatır.

    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields

    ' yordamını çağırır. Recordset nesnesini ve

    ' istenen alan genişliğini aktarır.

    EnumFields rst, 15
```

```
    dbs.Close

End Sub
```

- **Var, VarP İşlevleri**

Sorgunun belirtilen bir alanında bulunan değerler kümesi olarak temsil edilen bir grubun veya grup örneğinin yaklaşık varyansını verir.

Sözdizimi

Var(*ifade*)

VarP(*ifade*)

İfade yertutucusu, kullanmak istediğiniz sayısal verileri içeren alanı belirten bir dize ifadesini veya bu alandaki verileri kullanarak bir hesaplama yapan bir ifadeyi temsil eder. *İfade* içindeki işleçler, tablo alanının adını, bir sabiti veya bir işlevi (bilinen veya kullanıcı tanımlı bir işlev olabilir ancak diğer SQL toplam işlevlerinden biri olamaz) içerebilir.

Uyarılar

VarP işlevi bir grubu değerlendirir ve **Var** işlevi bir grup örneğini değerlendirir.

Temel alınan sorgu ikiden az kayıt içeriyorsa, **Var** ve **VarP** işlevleri standart sapmanın hesaplanamayacağını belirten bir **Null** değeri döndürürler.

Var ve **VarP** işlevlerini bir sorgu ifadesinde veya bir SQL deyiminde kullanabilirsiniz.

Var, VarP İşlevleri

Bu örnek, İngiltere'ye teslim edilen siparişlerin navlun bedellerinin varyansını tahmin etmek için Siparişler tablosunu kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub VarX()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' İngiltere'ye teslim edilmiş olan siparişlerin navlun

    ' bedellerinin varyansyny hesaplar.

    Set rst = dbs.OpenRecordset("SELECT " _

        & "Var(Navlun) " _

        & "AS [Yngiltere Navlun Var] " _

        & "FROM Siparişler WHERE TeslimÜlkesi = 'İngiltere';")

    ' Recordset'i başlatır.

    rst.MoveLast
```

```
    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 20

    Debug.Print

    Set rst = dbs.OpenRecordset("SELECT " _
       & "VarP(Navlun) " _
       & "AS [Yngiltere Navlun VarP] " _
       & "FROM Siparişler WHERE TeslimÜlkesi = 'İngiltere';")
    ' Recordset'i başlatır.
    rst.MoveLast
    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 20
    dbs.Close
End Sub
```

VERİ TANIMI DİLİ

➢ CREATE TABLE Deyimi

Yeni bir tablo oluşturur.

Not Microsoft Jet veritabanı alt yapısı, Microsoft Jet veritabanları dışında CREATE TABLE kullanımını veya herhangi bir DDL deyimini desteklemez. Bunun yerine DAO Create yöntemlerini kullanın.

Sözdizimi

CREATE [TEMPORARY] TABLE *tablo* (*alan1 tür* [(*boyut*)] [NOT NULL] [WITH COMPRESSION | WITH COMP] [*dizin1*] [, *alan2 tür* [(*boyut*)] [NOT NULL] [*dizin2*] [, ...]] [, CONSTRAINT *çokalanlıdizin* [, ...]])

CREATE TABLE deyiminin bölümleri şunlardır:

Bölüm	Açıklama
tablo	Oluşturulacak tablonun adı.
alan1, *alan2*	Yeni tabloda oluşturulacak alan veya alanların adı. En az bir alan oluşturmalısınız.

tür	Yeni tablodaki *alanın* veri türüdür.
boyut	Karakter olarak alan boyutu (yalnızca Metin ve İkili alanlarında).
dizin1, *dizin2*	Tek alanlı dizini tanımlayan bir CONSTRAINT yan tümcesidir. Bu dizini oluşturma konusunda ayrıntılı bilgi için CONSTRAINT Yan Tümcesi konusuna bakın.
çokalanlıdizin	Çok alanlı dizini tanımlayan bir CONSTRAINT yan tümcesidir. Bu dizini oluşturma konusunda ayrıntılı bilgi için CONSTRAINT Yan Tümcesi konusuna bakın.

Uyarılar

Yeni bir tablo ile alanlarını ve alan kısıtlamalarını tanımlamak için CREATE TABLE deyimini kullanın. Bir alan için NOT NULL özelliğini belirlerseniz, bu alanda geçerli veriler bulundurmak üzere yeni kayıtlar gerekir.

Bir CONSTRAINT yan tümcesi, bir alana değişik kısıtlamalar koyar ve birincil anahtar oluşturmak üzere kullanılabilir. CREATE INDEX deyimini, var olan tablolarda birincil anahtar veya ek dizinler oluşturmak üzere kullanabilirsiniz.

Tek bir alanda veya CONSTRAINT adlı tek veya çok alana uygulanmış adlandırılmış bir CONSTRAINT yan tümcesinde NOT NULL kullanamazsınız. Bununla beraber, NOT NULL kısıtlamasını yalnızca bir kez bir alana uygulayabilirsiniz. Bu kısıtlamayı birden çok kere uygulamayı denerseniz, çalışma anı hatası alırsınız.

Oluşturulan TEMPORARY bir tablo, ancak içinde oluşturulduğu oturumda görünür durumda kalır. Oturum sonlandırıldığında tablo da otomatik olarak silinir. Geçici tablolara, birden çok kullanıcı erişebilir.

WITH COMPRESSION özelliği, yalnızca CHARACTER ve MEMO (TEXT olarak da bilinir) veri türleri ve onların eşanlamlıları ile kullanılabilir.

Unicode karakter sunu biçimindeki değişiklikten dolayı, CHARACTER sütunları için WITH COMPRESSION özelliği eklenmiştir. Her bir Unicode karakter iki bayt gerektirir. Daha çok karakter verilerine sahip var olan Microsoft® Jet veritabanlarında, veritabanı Microsoft Jet version 4.0 biçimine dönüştürüldüğünde veritabanı dosyasının boyutu yaklaşık olarak iki katına çıkacaktır. Ancak, Tek Baytlık Karakter Kümeleri (SBCS) olarak bilinen çoğu karakter kümesinin Unicode sunumu kolaylıkla tek bayta sıkıştırılabilir. Bu özellikte bir CHARACTER sütunu tanımlarsanız, veriler saklanırken otomatik olarak sıkıştırılacak ve sütundan alınırken de otomatik olarak genişletileceklerdir.

Verileri sıkıştırılmış biçimde saklamak için MEMO sütunları da tanımlanabilir. Ancak, bir sınırlama vardır. Yalnızca, sıkıştırıldığında 4096 bayt ya da daha az yer tutan MEMO sütunları sıkıştırılır. Diğer tüm MEMO sütunları, sıkıştırılmamış olarak kalır. Böylece, belirli bir tabloda belirli bir MEMO sütunu için, verilerin bir kısmı sıkıştırılmış diğerleri sıkıştırılmamış olabilir.

CREATE TABLE Deyimi, CONSRAINT Yan Tümcesi Örneği

Bu örnek, iki metin alanı olan BuTablo adlı yeni bir tablo oluşturur.

```
Sub CreateTableX1()

    Dim dbs As Database

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

' İki metin alanı olan bir tablo oluşturur.

    dbs.Execute "CREATE TABLE BuTablo " _

        & "(Ady CHAR, Soyady CHAR);"
```

```
    dbs.Close
End Sub
```

Bu örnek, iki metin alanı bir Tarih/Saat alanı ve bu üç alandan oluşan benzersiz bir dizini olan Tablom adlı yeni bir tablo oluşturur.

```
Sub CreateTableX2()
    Dim dbs As Database
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' Üç alanı ve bu üç alandan oluşan benzersiz
    ' bir dizini olan bir tablo oluşturur.
    dbs.Execute "CREATE TABLE Tablom " _
        & "(Ady CHAR, Soyady CHAR, " _
        & "DoğumTarihi DATETIME, " _
        & "CONSTRAINT TablomKysytlamasy UNIQUE " _
        & "(Adı, Soyadı, DoğumTarihi));"
    dbs.Close
End Sub
```

Bu örnek, iki metin alanı ve bir tamsayı alanı olan yeni bir tablo oluşturur. SSN alanı birincil anahtardır.

```
Sub CreateTableX3()
     Dim dbs As Database
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' Üç alany ve bir birincil anahtary olan bir tablo
    ' oluşturur.
    dbs.Execute "CREATE TABLE YeniTablo " _
        & "(Ady CHAR, Soyady CHAR, " _
        & "SSN INTEGER CONSTRAINT AlanymKysytlamasy " _
        & "PRIMARY KEY);"
    dbs.Close
End Sub
```

➢ CREATE INDEX Deyimi

Var olan bir tabloda yeni bir dizin oluşturur.

Not Microsoft Jet veritabanları dışında, Microsoft Jet veritabanı alt yapısı CREATE INDEX (ODBC bağlı tabloda geçici dizin oluşturma dışında) deyiminin veya diğer veri tanımı dili (DDL) deyimlerinin kullanımını desteklemez. Bunun yerine DAO Create yöntemlerini kullanın. Ayrıntılı bilgi için Uyarılar bölümüne bakın.

Sözdizimi

CREATE [UNIQUE] INDEX *dizin*
ON *tablo* (*alan* [ASC|DESC][, *alan* [ASC|DESC], ...])
[WITH { PRIMARY | DISALLOW NULL | IGNORE NULL }]

CREATE INDEX deyiminin bölümleri şunlardır:

Bölüm	Açıklama
dizin	Oluşturulacak dizinin adı.
tablo	Dizini içerecek var olan tablonun adı.
alan	Dizilecek alan veya alanların adı. Tek alanlı dizinin oluşturulması için, tablo adının ardından ayraçlar içinde alan adını listeleyin. Çok alanlı dizin oluşturmak için, dizin içinde bulundurulacak her alanın adını listeleyin. Azalan dizinler oluşturmak için DESC saklı sözcüğünü kullanın, aksi takdirde dizinlerin artan biçimde olduğu varsayılır.

Uyarılar

Farklı kayıtların dizili alan veya alanlarında değerlerin yinelenmesini engellemek için, UNIQUE saklı sözcüğünü kullanın.

İsterseniz kullanabileceğiniz WITH yan tümcesi ile, veri doğrulama kurallarını verebilirsiniz. Şunları yapabilirsiniz:

- DISALLOW NULL seçeneğini kullanarak yeni kayıtların dizili alan veya alanlarında Null girişleri engelleyebilirsiniz.
- IGNORE NULL seçeneğini kullanarak, **Null** değerler içeren dizili alan veya alanları olan kayıtların dizin içinde yer almasını önleyebilirsiniz.
- PRIMARY saklı sözcüğünü kullanarak, dizili alan veya alanları birincil anahtar olarak belirleyebilirsiniz. Bu, anahtarın benzersiz olmasını sağlar böylece UNIQUE saklı sözcüğünü eklemeyebilirsiniz.

Microsoft® SQL Server™ gibi bir ODBC veri kaynağı içinde bulunan ve bir dizini olmayan bağlı tabloda geçici dizin oluşturmak için CREATE INDEX'i kullanabilirsiniz. Geçici dizin oluşturmak için, uzaktaki sunucuda yetki veya erişim izninizin olması gerekmez; uzaktaki veritabanı geçici dizinden etkilenmez ve bu durumdan haberdar olmaz. Hem bağlı hem de yerel tablolarda aynı sözdizimini kullanabilirsiniz. Genellikle salt okunur olacak bir tabloda geçici dizin oluşturmak özellikle kullanışlı olabilir.

Bir tabloya tek veya çok alanlı dizin eklemek için ALTER TABLE deyimini de kullanabilirsiniz; ya da ALTER TABLE veya CREATE INDEX ile oluşturulmuş bir dizini kaldırmak için ALTER TABLE deyimini veya DROP deyimini kullanabilirsiniz.

Not Zaten birincil anahtarı olan bir tabloda yeni bir dizin oluşturmak için PRIMARY saklı sözcüğünü kullanmayın, aksi takdirde hata oluşur.

➢ CREATE PROCEDURE Deyimi

Saklı bir yordam oluşturur.

Not Microsoft Jet veritabanı alt yapısı, Microsoft Jet veritabanları dışında CREATE PROCEDURE kullanımını veya herhangi bir DDL deyimini desteklemez.

Sözdizimi

CREATE PROCEDURE *yordam*
[*param1 veritürü*[, *param2 veritürü*[, ...]] AS sqlstatement

CREATE PROCEDURE deyiminin bölümleri şunlardır:

Bölüm	Açıklama
yordam	Yordamın adı. Standart adlandırma kurallarına uymalıdır.
param1, *param2*	1 ile 255 arasında alan adı veya parametredir. Örneğin: CREATE PROCEDURE Ülkelere_Göre_Satışlar [Başlangıç Tarihi] DateTime, [Bitiş Tarihi] DateTime; Parametreler hakkında ayrıntılı bilgi için PARAMETERS konusuna bakın.
veritürü	Birincil Microsoft Jet SQL veri türleri veya onların eşanlamlılarından biridir.
sqldeyimi	SELECT, UPDATE, DELETE, INSERT, CREATE TABLE, DROP TABLE vs. gibi bir SQL deyimidir.

Uyarılar

Bir SQL yordamı, yordamın adını belirten PROCEDURE yan tümcesinden, isteğe bağlı bir parametre tanımları listesinden ve tek bir SQL deyiminden oluşur.

Bir yordam adı, var olan bir tablonun adı ile aynı olamaz.

CREATE PROCEDURE Deyimi, PROCEDURE Yan Tümcesi Örneği

Bu örnek, sorguyu KategoriListesi olarak adlandırır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub ProcedureX()

    Dim dbs As Database, rst As Recordset

    Dim qdf As QueryDef, strSql As String


    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")


    strSql = "PROCEDURE KategoriListesi; " _

        & "SELECT DISTINCTROW KategoriAdy, " _

        & "KategoriNo FROM Kategoriler " _

        & "ORDER BY KategoriAdy;"
```

```
    ' SQL deyimini temel alan bir adlandırılmış
    ' bir QueryDef oluşturur.
    Set qdf = dbs.CreateQueryDef("YeniSorgu", strSql)
    ' Geçici bir anlık görüntü türünde Recordset oluşturur.
    Set rst = qdf.OpenRecordset(dbOpenSnapshot)
    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 15

    ' Bu bir örnek olduğu için QueryDef'i
    ' siler.
    dbs.QueryDefs.Delete "YeniSorgu"

    dbs.Close
End Sub
```

- **CREATE USER veya GROUP Deyimi**

Bir veya daha çok kullanıcı veya grup oluşturur.

Sözdizimi

Kullanıcı oluşturma:

CREATE USER *kullanıcı parola kimlikno* [, *kullanıcı parola kimlikno*, ...]

Grup oluşturma:

CREATE GROUP *grup kimlikno*[, *grup kimlikno*, ...]

CREATE USER veya GROUP deyimlerinin bölümleri şunlardır:

Bölüm	**Açıklama**
kullanıcı	Çalışma grubu bilgileri dosyasına eklenecek kullanıcı adıdır.
grup	Çalışma grubu bilgileri dosyasına eklenecek grup adıdır.
parola	Belirtilen *kullanıcı* adı ile ilişkilendirilmiş paroladır.
kimlikno	Kişinin kimlik numarasıdır.

Uyarılar

Kullanıcı ve *grup* adları aynı olamaz.

Oluşturulacak her bir *kullanıcı* veya *grup* için bir *parola* gereklidir.

- **CREATE VIEW Deyimi**

Yeni bir görünüm oluşturur.

Not Microsoft Jet veritabanı alt yapısı, Microsoft Jet veritabanları dışında CREATE VIEW kullanımını veya herhangi bir DDL deyimini desteklemez.

Sözdizimi

CREATE VIEW *görünüm* [(*alan1*[, *alan2*[, ...]])] AS *seçmedeyimi*

CREATE VIEW deyiminin bölümleri şunlardır:

Bölüm	Açıklama
görünüm	Oluşturulacak görünümün adı.
alan1, *alan2*	*Seçmedeyimi* içinde belirtilen ilgili alanlar için alan veya alanların adı.
seçmedeyimi	Bir SQL SELECT deyimidir. Ayrıntılı bilgi için SELECT deyimi konusuna bakın.

Uyarılar

Görünümü tanımlayan SELECT deyimi, bir SELECT INTO deyimi olamaz.

Görünümü tanımlayan SELECT deyimi herhangi bir parametre içeremez.

Bir görünüm adı, var olan bir tablonun adı ile aynı olamaz.

SELECT deyimi ile tanımlanan sorgu güncelleştirilebilir ise, görünüm de güncelleştirilebilir. Aksi takdirde görünüm salt okunurdur.

SELECT deyimi ile tanımlanan sorgudaki herhangi iki alan aynı ada sahipse, görünüm tanımı, sorgudaki alanların benzersiz adlarını belirten bir alan listesi içermelidir.

- **ADD USER Deyimi**

Varolan bir *gruba* bir veya daha çok *kullanıcı* ekler.

Sözdizimi

ADD USER *kullanıcı*[, *kullanıcı*, ...] TO *grup*

ADD USER deyiminin bölümleri şunlardır:

Bölüm	Açıklama
kullanıcı	Çalışma grubu bilgileri dosyasına eklenecek kullanıcı adıdır.
grup	Çalışma grubudadefWorkgroupbilgileri dosyasına eklenecek grup adıdır.

Uyarılar

Kullanıcı bir *gruba* eklendiğinde, *kullanıcı* bu *gruba* verilmiş olan tüm izinlere sahip olur.

- **DROP USER veya GROUP Deyimi**

Varolan bir veya daha çok *kullanıcıyı* veya *grubu* siler veya varolan bir *gruptan* varolan bir veya daha çok *kullanıcıyı* kaldırır.

Sözdizimi

Bir veya daha çok *kullanıcıyı* siler veya bir *gruptan* bir veya daha çok *kullanıcıyı* kaldırır.

DROP USER *kullanıcı*[, *kullanıcı*, ...] [FROM *grup*]

Bir veya daha çok *grubu* silme:

DROP GROUP *grup*[, *grup*, ...]

DROP USER veya GROUP deyimlerinin bölümleri şunlardır:

Bölüm	Açıklama
kullanıcı	Çalışma grubu bilgileri dosyasından kaldırılacak kullanıcı adıdır.
grup	Çalışma grubu bilgileri dosyasından kaldırılacak grup adıdır.

Uyarılar

FROM anahtar sözcüğü DROP USER deyiminde kullanılırsa, deyimde listelenen her *kullanıcı* FROM anahtar sözcüğünden sonra belirtilen *gruptan* kaldırılacaktır. Ancak *kullanıcıların* kendiler silinmez.

DROP GROUP deyimi belirtilen *grupları* siler. *Grubun* üyesi olan kullanıcılar bundan etkilenmezler, ancak artık silinen *grupların* üyesi değillerdir.

- **ALTER TABLE Deyimi**

CREATE TABLE deyimi ile oluşturulan tablonun tasarımını değiştirir.

Not Microsoft Jet veritabanı tasarımı, Microsoft Jet veritabanları dışında ALTER TABLE kullanımını veya herhangi bir veri tanımı dili (DDL) deyimini desteklemez. Bunun yerine DAO Create yöntemlerini kullanın.

Sözdizimi

ALTER TABLE *tablo* {ADD {COLUMN *alan türü*[(*boyut*)] [NOT NULL] [CONSTRAINT *dizin*] |
ALTER COLUMN *alan türü*[(*boyut*)] |
CONSTRAINT *çokalanlıdizin*} |
DROP {COLUMN *alan* I CONSTRAINT *dizinadı*} }

ALTER TABLE deyiminin bölümleri şunlardır:

Bölüm	Açıklama
tablo	Değiştirilecek tablonun adı.
alan	*Tabloya* eklenecek veya *tablodan* silinecek alanın adı. Ya da, *tabloda* değiştirilecek alanın adı.
tür	*Alanın* veri türü.
boyut	Karakter olarak alan boyutu (yalnızca Metin ve İkili alanlarında).
dizin	*Alanın* dizini. Bu dizini oluşturma konusunda ayrıntılı bilgi için CONSTRAINT Yan Tümcesi konusuna bakın.
çokalanlıdizin	*Tabloya* eklenecek çok alanlı dizinin tanımı. Bu dizini oluşturma konusunda ayrıntılı bilgi için CONSTRAINT Yan Tümcesi konusuna bakın.
dizinadı	Kaldırılacak çok alanlı dizinin adı.

Uyarılar

ALTER TABLE deyimini kullanarak var olan bir tabloyu farklı yollarla değiştirebilirsiniz. Şunları yapabilirsiniz:

- Tabloya yeni bir alan eklemek için ADD COLUMN'ı kullanabilirsiniz. Alan adını, veri türünü ve isteğe bağlı olarak boyutu (Metin ve İkili alanlar için) belirtirsiniz. Örneğin, aşağıdaki deyim Çalışan tablosuna Notlar adlı 25 karakterlik bir Metin alanı ekler.

  ALTER TABLE Çalışan ADD COLUMN Notlar TEXT(25)

  Bu alana bir dizin de tanımlayabilirsiniz. Tek alanlı dizinler konusunda ayrıntılı bilgi için CONSTRAINT Yan Tümcesi konusuna bakın.

  Bir alan için NOT NULL özelliğini belirlerseniz, bu alanda geçerli veriler bulundurmak üzere yeni kayıtlar gerekir.

- Var olan bir alanın veri türünü değiştirmek için ALTER COLUMN'ı kullanabilirsiniz. Alan adını, yeni veri türünü ve isteğe bağlı olarak Metin ve İkili alan boyutunu belirtirsiniz. Örneğin aşağıdaki deyim, Çalışan tablosundaki özgün olarak Tamsayı şeklinde tanımlı PostaKodu alanının veri türünü 10 karakterlik Metin alanı olarak değiştirir:

  ALTER TABLE Çalışan ALTER COLUMN PostaKodu TEXT(25)

- Çok alanlı dizin eklemek için ADD CONSTRAINT'i kullanabilirsiniz. Çok alanlı dizinler konusunda ayrıntılı bilgi için CONSTRAINT Yan Tümcesi konusuna bakın.

- Alan silmek için DROP COLUMN'ı kullanabilirsiniz. Yalnızca alanın adını belirtirsiniz.

- Çok alanlı dizini silmek için DROP CONSTRAINT'i kullanabilirsiniz. CONSTRAINT saklı sözcüğünün ardından yalnızca dizin adını belirtirsiniz.

Notlar

- Bir anda birden çok alan veya dizin ekleyemez veya silemezsiniz.

- Bir tabloya tek veya çok alanlı dizin eklemek için CREATE INDEX deyimini kullanabilirsiniz; ya da ALTER TABLE veya CREATE INDEX ile oluşturulmuş bir dizini silmek için DROP deyimini kullanabilirsiniz.

- Tek bir alanda veya CONSTRAINT adlı tek veya çok alana uygulanmış adlandırılmış bir CONSTRAINT yan tümcesinde NOT NULL kullanamazsınız. Bununla beraber, NOT NULL kısıtlamasını yalnızca bir kez bir alana uygulayabilirsiniz. Bu kısıtlamayı birden çok kere uygulamayı denerseniz, çalışma anı hatası alırsınız.

ALTER TABLE Deyimi Örneği

Bu örnek, Çalışanlar tablosuna **Money** veri türünde bir Maaş alanı ekler.

```
Sub AlterTableX1()

    Dim dbs As Database

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.

    Set dbs = OpenDatabase("Northwind.mdb")

    ' Maaş alanını Çalışanlar tablosuna ekler ve

    ' alanyn veri türünü Money yapar.
```

```
    dbs.Execute "ALTER TABLE Çalışanlar " _
        & "ADD COLUMN Maaş MONEY;"
    dbs.Close
End Sub
```

Bu örnek, Maaş alanının veri türünü **Money** yerine **Char** olarak değiştirir.

```
Sub AlterTableX2()
    Dim dbs As Database
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' Maaş alanını Çalışanlar tablosuna ekler ve
    ' alanyn veri türünü Money yapar.
    dbs.Execute "ALTER TABLE Çalışanlar " _
        & "ALTER COLUMN Maaş CHAR(20);"
    dbs.Close
End Sub
```

Bu örnek Maaş alanını Çalışanlar tablosundan kaldırır.

```
Sub AlterTableX3()
    Dim dbs As Database
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' Maaş alanını Çalışanlar tablosundan siler.
    dbs.Execute "ALTER TABLE Çalışanlar " _
        & "DROP COLUMN Maaş;"
    dbs.Close
End Sub
```

Bu örnek Siparişler tablosuna bir yabancı anahtar ekler. Yabancı anahtar ÇalışanNo alanını temel alır ve Çalışanlar tablosundaki ÇalışanNo alanına başvurur. Bu örnekte, REFERENCES yan tümcesinde Çalışanlar tablosundan sonra ÇalışanNo alanını listelemeniz gerekmez çünkü ÇalışanNo, Çalışanlar tablosunun birincil anahtarıdır.

```
Sub AlterTableX4()
    Dim dbs As Database
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
```

```
    Set dbs = OpenDatabase("Northwind.mdb")
    ' Siparişler tablosuna bir yabancı anahtar ekler.
    dbs.Execute "ALTER TABLE Siparişler " _
        & "ADD CONSTRAINT SiparişlerİlişkisiX5 " _
        & "FOREIGN KEY (ÇalışanNo) " _
        & "REFERENCES Çalışanlar (ÇalışanNo);"
    dbs.Close
End Sub
```

Bu örnek Siparişler tablosundan yabancı anahtarı kaldırır.

```
Sub AlterTableX5()
    Dim dbs As Database
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' SiparişlerİlişkisiX5 yabancı anahtarını Siparişler
    ' tablosundan kaldyryr.
    dbs.Execute "ALTER TABLE Siparişler " _
        & "DROP CONSTRAINT SiparişlerİlişkisiX5 " _
    dbs.Close
End Sub
```

- **ALTER USER veya DATABASE Deyimi**

Varolan bir kullanıcının veya bir veritabanının parolasını değiştirir.

Sözdizimi

ALTER DATABASE PASSWORD *yeniparola eskiparola*

ALTER USER *kullanıcı* PASSWORD *yeniparola eskiparola*

ALTER USER veya DATABASE deyimlerinin bölümleri şunlardır:

Bölüm	**Açıklama**
kullanıcı	Çalışma grubu bilgileri dosyasına eklenecek kullanıcı adıdır.
yeniparola	Belirtilen *kullanıcı* veya veritabanı adı ile ilişkilendirilmiş yeni paroladır.
eskiparola	Belirtilen *kullanıcı* veya *grup* adı ile ilişkilendirilmiş varolan paroladır.

- **DROP Deyimi**

Varolan bir tabloyu, yordamı veya görünümü veritabanından siler; ya da var olan bir dizini tablodan siler.

Not Microsoft Jet veritabanı alt yapısı**,** Microsoft Jet veritabanları dışında DROP kullanımını veya herhangi bir DDL deyimini desteklemez. Bunun yerine **DAO Delete** yöntemlerini kullanın.

Sözdizimi

DROP {TABLE *tablo* | INDEX *dizin* ON *tablo* | PROCEDURE *yordam* | VIEW *görünüm*}

DROP deyiminin bölümleri şunlardır:

Bölüm	Açıklama
tablo	Silinecek tablonun adı veya içinden dizinin silineceği tablonun adı.
yordam	Silinecek yordamın adı.
görünüm	Silinecek görünümün adı.
dizin	*Tablodan* silinecek dizinin adı.

Uyarılar

Tabloyu silmeden veya dizinini kaldırmadan önce tabloyu kapatmalısınız.

Tablodan bir dizin silmek için ALTER TABLE de kullanılabilir.

Tablo oluşturmak için CREATE TABLE ve dizin oluşturmak için CREATE INDEX or ALTER TABLE komutlarını kullanabilirsiniz. Bir tabloyu değiştirmek için ALTER TABLE'i kullanın.

DROP Deyimi Örneği

Aşağıdaki örnek, Northwind veritabanındaki Çalışanlar tablosunda YeniDizin dizininin bulunduğunu varsayar.

Bu örnek, Dizinim dizinini Çalışanlar tablosundan siler.

```
Sub DropX1()
    Dim dbs As Database
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' YeniDizin dizinini Çalışanlar tablosundan siler.
    dbs.Execute "DROP INDEX YeniDizin ON Çalışanlar;"
    dbs.Close
End Sub
```

Bu örnek, Çalışanlar tablosunu veritabanından siler.

```
Sub DropX2()
    Dim dbs As Database
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")
    ' Çalışanlar tablosunu siler.
```

```
    dbs.Execute "DROP TABLE Çalışanlar;"

    dbs.Close

End Sub
```

GRANT Deyimi

Belitilen ayrıcalıkları varolan bir kullanıcıya veya gruba verir.

Sözdizimi

GRANT {*ayrıcalık*[, *ayrıcalık*, ...]} ON
{TABLE *tablo* |
OBJECT *nesne*|

CONTAINER *konteyner* } TO {*yetkisahibiadı*[, *yetkisahibiadı*, ...]}

GRANT deyiminin bölümleri şunlardır:

Bölüm	Açıklama
ayrıcalık	Verilecek olan ayrıcalık veya ayrıcalıklardır. Ayrıcalıklar aşağıdaki anahtar sözcükler kullanılarak belirtilir: SELECT, DELETE, INSERT, UPDATE, DROP, SELECTSECURITY, UPDATESECURITY, DBPASSWORD, UPDATEIDENTITY, CREATE, SELECTSCHEMA, SCHEMA ve UPDATEOWNER.
tablo	Geçerli bir tablo adı.
nesne	Bu tablo olmayan bir nesneyi içerebilir. Saklı bir sorgu (görünüm veya yordam) bir örnek olabilir.
konteyner	Geçerli bir konteynerin adıdır.
yetkisahibiadı	Kullanıcı veya grup adıdır.

DİĞER

- **REVOKE Deyimi**

Belitilen ayrıcalıkları varolan bir kullanıcıdan veya gruptan geri alır.

Sözdizimi

REVOKE {*ayrıcalık*[, *ayrıcalık*, ...]} ON
{TABLE *tablo* |
OBJECT *nesne*|

CONTAINER *konteyner* }
FROM {*yetkisahibiadı*[, *yetkisahibiadı*, ...]}

REVOKE deyiminin bölümleri şunlardır:

Bölüm	Açıklama
ayrıcalık	Geri alınacak olan ayrıcalık veya ayrıcalıklardır. Ayrıcalıklar aşağıdaki anahtar sözcükler kullanılarak belirtilir: SELECT, DELETE, INSERT, UPDATE, DROP, SELECTSECURITY, UPDATESECURITY, DBPASSWORD, UPDATEIDENTITY, CREATE,

	SELECTSCHEMA, SCHEMA ve UPDATEOWNER.
tablo	Geçerli bir tablo adı.
nesne	Bu tablo olmayan bir nesneyi içerebilir. Saklı bir sorgu (görünüm veya yordam) bir örnek olabilir.
konteyner	Geçerli bir konteynerin adıdır.
yetkisahibiadı	Kullanıcı veya grup adıdır.

- **ALL, DISTINCT, DISTINCTROW, TOP Doğrulamaları**

SQL sorguları ile seçilen kayıtları belirler.

Sözdizimi

SELECT [ALL | DISTINCT | DISTINCTROW | [TOP *n* [PERCENT]]]
FROM *tablo*

Bu doğrulamaları içeren bir SELECT deyiminin bölümleri şunlardır:

Bölüm	**Açıklama**
ALL	Bu doğrulamalardan herhangi birini bulundurmazsanız varsayılan değerdir. Microsoft Jet veritabanı alt yapısı, SQL deyimindeki koşullara uyan tüm kayıtları seçer. Aşağıdadaki iki örnek eşdeğerdir ve Çalışanlar tablosundaki tüm kayıtları döndürürler. SELECT ALL * FROM Çalışanlar ORDER BY ÇalışanNo; SELECT * FROM Çalışanlar ORDER BY ÇalışanNo;
DISTINCT	Seçili alanlardan yinelenen verileri içeren kayıtları gözardı eder. Sorgu sonuçlarında yer alabilmesi için, SELECT deyiminde listelenen her alanın değeri benzersiz olmalıdır. Örneğin, Çalışanlar tablosundaki çok sayıda çalışan aynı soyadına sahip olabilir. İki kayıt Soyadı alanında Etikan adını içeriyorsa, aşağıdaki SQL deyimi Etikan adını içeren yalnızca bir tek kayıt döndürür. SELECT DISTINCT Soyadı FROM Çalışanlar; DISTINCT'i yazmazsanız, bu sorgu her iki Etikan kaydını da döndürür. SELECT yan tümcesi birden fazla alan içeriyorsa, belirli bir kaydın sonuçlar içinde bulunması için, alanlardaki değerlerin birleşimi benzersiz olmalıdır. DISTINCT'i kullanan bir sorgunun çıktısı güncelleştirilebilir değildir ve diğer kullanıcıların sonradan yaptıkları değişiklikleri yansıtmaz.
DISTINCTROW	Alanların değil, kaydın tamamı yineleniyorsa verileri gözardı eder. Örneğin, Müşteriler ve Siparişler tablolarını MüşteriNo alanıyla birleştiren bir sorgu oluşturabilirsiniz. Müşteriler tablosunda yinelenen MüşteriNo alanı yoktur, ancak her müşterinin çok sayıda siparişi olduğundan Siparişler tablosunda yinelenen MüşteriNo alanı vardır. Aşağıdaki SQL deyimi, en az bir siparişi olan şirketlerin listesini bu siparişler hakkında herhangi bir ayrıntı vermeden oluşturmak üzere DISTINCTROW'u nasıl kullanacağınızı gösterir.

	SELECT DISTINCTROW ŞirketAdı FROM Müşteriler INNER JOIN Siparişler ON Müşteriler.MüşteriNo = Siparişler.MüşteriNo ORDER BY ŞirketAdı; DISTINCTROW'u yazmazsanız bu sorgu, birden fazla siparişi olan her şirket için çok sayıda satır oluşturur. Sorguda kullanılan tabloların tüm alanlarını değil yalnızca bazı alanlarını seçtiğinizde DISTINCTROW'un bir etkisi olur. Sorgunuz yalnızca tek bir tablo içeriyorsa ya da tüm tablolardaki alanları alıyorsanız DISTINCTROW yok sayılır.
TOP *n* [PERCENT]	ORDER BY yan tümcesi ile belirtilen aralığın üstüne veya altına düşen belirli sayıda kaydı döndürür. 1994 yılının en başarılı 25 öğrencisinin adlarını almak istediğinizi varsayalım: SELECT TOP 25 Adı, Soyadı FROM Öğrenciler WHERE MezuniyetYılı = 1994 ORDER BY MezuniyetDerecesi DESC; ORDER BY yan tümcesini bulundurmazsanız, sorgu, WHERE yan tümcesini sağlayan Öğrenciler tablosunun rasgele 25 kaydını döndürür. TOP seçimi, eşit değerler arasında seçim yapmaz. Önceki örnekte 25. ve 26. öğrenci aynı mezuniyet derecesine sahipse, sorgu 26 kayıt geri döndürür. ORDER BY yan tümcesi ile belirtilen aralığın üstüne veya altına düşen kayıtların belirli bir yüzdesini döndürmek için, PERCENT saklı sözcüğünü de kullanabilirsiniz. 25 öğrencisi yerine, sınıfın en düşük yüzde 10'unun adlarını almak istediğinizi varsayalım: SELECT TOP 10 PERCENT Adı, Soyadı FROM Öğrenciler WHERE MezuniyetYılı = 1994 ORDER BY MezuniyetDerecesi ASC; ASC doğrulaması, alttaki değerlerin döndürüleceğini belirtir. TOP'u sağlayan değer, işaretsiz bir **Tamsayı** olmalıdır. TOP, sorgunun güncelleştirilebilir olup olmamasını etkilemez.
tablo	İçinden kayıtların seçileceği tablonun adıdır.

ALL DISTINCT, DISTINCTROW, TOP Doğrulamaları Örneği

Bu örnek, Müşteriler ve Siparişler tablolarını MüşteriNo alanıyla birleştiren bir sorgu oluşturur. Müşteriler tablosunda yinelenen MüşteriNo alanı yoktur, ancak her müşterinin çok sayıda siparişi olduğundan Siparişler tablosunda yinelenen MüşteriNo alanı vardır. DISTINCTROW kullanılarak, en az bir siparişi olan şirketlerin listesi, bu siparişlerin ayrıntıları verilmeden oluşturulur.

```
Sub AllDistinctX()

    Dim dbs As Database, rst As Recordset

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu

    ' bulundurmak üzere değiştirin.
```

```
    Set dbs = OpenDatabase("Northwind.mdb")

    ' Müşteriler ve Siparişler tabloları MüşteriNo
    ' alanı ile birleştirilir. En az bir siparişi olan
    ' şirketler seçilir.
    Set rst = dbs.OpenRecordset("SELECT DISTINCTROW " _
        & "ŞirketAdı FROM Müşteriler " _
        & "INNER JOIN Siparişler " _
        & "ON Müşteriler.MüşteriNo = " _
        & "Siparişler.MüşteriNo " _
        & "ORDER BY ŞirketAdı;")

    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 25
    dbs.Close
End Sub
```

- **WITH OWNERACCESS OPTION Bildirimi**

Güvenli çalışma grubu şeklindeki çok kullanıcılı bir ortamda bu bildirimi bir sorgu ile birlikte, sorguyu çalıştıran kullanıcıya sorgunun sahibi ile aynı izinleri vermek için kullanın.

Sözdizimi

sqldeyimi
    WITH OWNERACCESS OPTION

Uyarılar

WITH OWNERACCESS OPTION bildirimi isteğe bağlıdır.

Aşağıdaki örnek, sorgu sahibinin maaş bilgilerini görüntüleme iznine sahip olması halinde, kullanıcının bu bilgileri görmesini (kullanıcının Bordro tablosunu görüntüleme izni olmasa bile) sağlar:

SELECT Soyady,

Adı, Maaş

FROM Çalışanlar

ORDER BY Soyady

WITH OWNERACCESS OPTION;

Kullanıcıya normalde tablo oluşturma veya tabloya ekleme yapma izni verilmemişse, kullanıcının tablo oluşturma veya ekleme sorgusu çalıştırmasını sağlamak için WITH OWNERACCESS OPTION'ı kullanabilirsiniz.

Çalışma grubu güvenlik ayarlarına ve kullanıcının izinlerine uymak istiyorsanız, WITH OWNERACCESS OPTION bildirimini kullanmayın.

Bu seçenek, bu veritabanı ile ilişkilendirilmiş System.mdw dosyasına erişiminiz olmasını gerektirir. Bu, ancak çok kullanıcılı ve güvenli uygulamalarda kullanışlıdır.

- **SQL İşlevlerinde Alanları Hesaplama**

Bir alandaki değerler üzerinde bir hesaplama yapmak için, bir SQL toplam işlevinde dize ifadesi değişkenini kullanabilirsiniz. Örneğin, bir alan değerini bir sabitle çarparak bir yüzdeyi (satış vergisi veya ek vergi gibi) hesaplayabilirsiniz.

Aşağıdaki tablo, Northwind.mdb veritabanındaki Siparişler ve Sipariş Ayrıntıları tablolarının alanlarında yapılan hesaplama örnekleri sunar.

Hesaplama	**Örnek**
Alana bir sayı ekler	Navlun + 5
Alandan bir sayı çıkartır	Navlun - 5
Alanı bir sayı ile çarpar	BirimFiyatı * 2
Alanı bir sayıya böler	Navlun / 2
Bir alanı diğeri ile toplar	StoktakiBirim + SipariştekiBirim
Bir alanı diğerinden çıkarır	SiparişDüzeyi - StoktakiBirim

Aşağıdaki örnek, Northwind.mdb veritabanındaki tüm siparişlerin ortalama indirim miktarını hesaplar. Her siparişin indirim miktarını hesaplamak için BirimFiyatı ile İndirim alanlarındaki değerleri çarpar ve sonra ortalamayı hesaplar. Bu ifadeyi bir Visual Basic kodunda SQL deyimi olarak kullanabilirsiniz:

SELECT Avg(BirimFiyatı * İndirim) AS [Ortalama İndirim] FROM [Sipariş Ayrıntıları];

- **PARAMETERS Bildirimi**

Parametre sorgusundaki her parametre için adı veya veri türünü bildirir.

Sözdizimi

PARAMETERS *adı veritürü* [, *adı veritürü* [, ...]]

PARAMETERS bildiriminin bölümleri şunlardır:

Bölüm	**Açıklama**
adı	Parametrenin adıdır. **Parameter** nesnesinin **Name** özelliğine atanmıştır ve bu parametreyi **Parameters** derlemesinde belirlemek için kullanılır. *Adı*, uygulamanız sorguyu çalıştırırken bir iletişim kutusunda görüntülenen bir dize olarak kullanılabilir. Boşluk veya noktalama işareti içeren metinleri

	belirtmek için köşeli ayraçlar([ ]) kullanın. Örneğin [Düşük fiyat] ve [Rapora hangi ay ile başlansın?] geçerli *adı* değişkenleridir.
veritürü	Birincil Microsoft Jet SQL veri türleri veya onların eşanlamlılarından biridir.

Uyarılar

Düzenli olarak çalıştırdığınız sorgularda, bir parametre sorgusu oluşturmak üzere PARAMETERS bildirimini kullanabilirsiniz. Bir parametre sorgusu, sorgu ölçütünü değiştirme işlemini otomatik hale getirmenizde yardımcı olabilir. Bir parametre sorgusu kullanılırsa, sorgu her çalıştığında kodlarını parametreleri soracaktır.

PARAMETERS bildirimi isteğe bağlıdır ancak belirtilirse, SELECT dahil tüm deyimlerden önce gelir.

Bildirimde birden fazla parametre varsa, bunları birbirlerinden virgüller ile ayırın. Aşağıdaki örnek iki parametre içerir:

PARAMETERS [Düşük fiyat] ParaBirimi, [Başlangıç tarihi] TarihSaat;

WHERE veya HAVING yan tümcesinde *adı* değişkenini kullanabilirsiniz ancak *veritürü* değişkenini kullanamazsınız. Aşağıdaki örnek, iki parametrenin girilmesini ister ve sonra ölçütü Siparişler tablosuna uygular:

PARAMETERS [Düşük fiyat] ParaBirimi,

[Başlangıç tarihi] TarihSaat;

SELECT SiparişNo, SiparişMiktarı

FROM Siparişler

WHERE SiparişMiktarı > [Düşük fiyat]

AND SiparişTarihi >= [Başlangıç tarihi];

PARAMETERS Bildirimi Örneği

Bu örnek, kullanıcının iş ünvanını sağlamasını ister ve bu iş ünvanını sorguda ölçüt olarak kullanır.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub ParametersX()
    Dim dbs As Database, qdf As QueryDef
    Dim rst As Recordset
    Dim strSql As String, strParm As String
    Dim strMessage As String
    Dim intCommand As Integer

    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("NorthWind.mdb")

    ' Parameters yan tümcesini tanymlar.
    strParm = "PARAMETERS [Çalışan Ünvanı] CHAR; "
```

```
' Parameters yan tümcesini içeren bir SQL
' deyimi tanymlar.
strSql = strParm & "SELECT Soyady, Ady, " _
    & "ÇalışanNo " _
    & "FROM Çalışanlar " _
    & "WHERE Ünvan =[Çalışan Ünvanı];"

' SQL deyimini temel alan bir QueryDef nesnesi
' oluşturur.
Set qdf = dbs.CreateQueryDef _
    ("Çalışanları Bul", strSql)

Do While True
    strMessage = "Çalışanları İşe Göre Bul " _
        & "ünvan:" & Chr(13) _
        & "  İş Ünvanını Seçin:" & Chr(13) _
        & "   1 - Satış Müdürü" & Chr(13) _
        & "   2 - Satış Temsilcisi" & Chr(13) _
        & "   3 - İç Satışlar Koordinatörü"

    intCommand = Val(InputBox(strMessage))

    Select Case intCommand
        Case 1
            qdf("Çalışan Ünvanı") = _
                "Satış Müdürü"
        Case 2
            qdf("Çalışan Ünvanı") = _
                "Satış Temsilcisi"
        Case 3
            qdf("Çalışan Ünvanı") = _
                "İç Satışlar Koordinatörü"
        Case Else
```

```
            Exit Do
        End Select

        ' Geçici bir anlık görüntü türünde Recordset oluşturur.
        Set rst = qdf.OpenRecordset(dbOpenSnapshot)
        ' Recordset'i başlatır.
        rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
        EnumFields rst, 12
    Loop

    ' Bu bir örnek olduğu için QueryDef'i
    ' siler.
    dbs.QueryDefs.Delete "Çalışanları Bul"

    dbs.Close
End Sub
```

- **SQL Alt Sorguları**

Bir alt sorgu, SELECT, SELECT...INTO, INSERT...INTO, DELETE veya UPDATE deyimi içinde veya bir başka alt sorgu içinde yer alan bir SELECT deyimidir.

Sözdizimi

Alt sorgu oluşturmak için üç tür sözdizimi kullanabilirsiniz:

***karşılaştırma* [ANY | ALL | SOME] (*sqldeyimi*)**

***ifade* [NOT] IN (*sqldeyimi*)**

[NOT] EXISTS (*sqldeyimi*)

Bir alt sorgunun bölümleri şunlardır:

Bölüm	**Açıklama**
karşılaştırma	İfadeyi alt sorgunun sonuçları ile karşılaştıran bir ifade ve bir karşılaştırma işlecidir.
ifade	Alt sorgu sonuç kümesini arayan ifadedir.
sqldeyimi	Herhangi bir SELECT deyimi ile aynı biçimde olan ve aynı kurallara sahip bir SELECT deyimidir. Ayraçlar içine alınmalıdır.

Uyarılar

SELECT deyiminin alan listesinde veya WHERE ile HAVING yan tümcesinde yazılacak bir ifade yerine, bir alt sorgu kullanabilirsiniz. Bir alt sorguda, WHERE veya HAVING yan tümcesi ifadelerinde değerlendirmek üzere bir veya daha fazla değer kümesini sağlamak için SELECT deyimi kullanırsınız.

Eşanlamlı olan ANY veya SOME doğrulamalarını, alt sorguda alınan kayıtlarla yapılan karşılaştırmaları sağlayan kayıtları ana sorguda almak için kullanın. Aşağıdaki örnek, birim fiyatı, yüzde 25 veya daha fazla indirimle satılmış ürünlerin birim fiyatından daha büyük olan tüm ürünleri verir.

SELECT * FROM Ürünler

WHERE BirimFiyaty > ANY

(SELECT BirimFiyatı FROM SiparişAyrıntıları

WHERE Yndirim >= .25);

ALL doğrulamasını, yalnızca alt sorguda alınan tüm kayıtlarla yapılan karşılaştırmaları sağlayan kayıtları ana sorguda almak için kullanın. Önceki örnekte ANY yerine ALL yazılırsa, sorgu sonucunda birim fiyatı, yüzde 25 veya daha fazla indirimle satılmış tüm ürünlerin birim fiyatından daha yüksek olan kayıtlar alınır. Bu daha kısıtlayıcıdır.

IN doğrulamasını, yalnızca, alt sorgudaki bazı kayıtların değerine eşit olan kayıtları ana sorguda almak için kullanın. Aşağıdaki örnek, yüzde 25 veya daha fazla indirimi olan tüm ürünleri verir:

SELECT * FROM Ürünler

WHERE ÜrünNo IN

(SELECT ÜrünNo FROM SiparişAyrıntıları

WHERE Yndirim >= .25);

Buna karşılık NOT IN deyimini, yalnızca, alt sorgudaki kayıtların değerine eşit olmayan kayıtları ana sorguda almak için kullanın.

EXISTS doğrulamasını, (isteğe bağlı NOT saklı sözcüğü olmadan) doğru/yanlış karşılaştırmalarında alt sorgunun herhangi bir kayıt geri döndürüp dördürmediğini belirlemek için kullanın.

Ayrıca, alt sorgunun dışındaki FROM yan tümcesinde listelenen tablolara başvurmak için, alt sorguda tablo diğer adlarını da kullanabilirsiniz. Aşağıdaki örnek, maaşları, aynı iş ünvanına sahip olan tüm çalışanların maaş ortalamasına eşit veya ondan daha yüksek olan çalışanların adlarını verir. Çalışanlar tablosuna "T1" diğeradı verilmiştir:

SELECT Soyady,

Adı, Ünvan, Maaş

FROM Çalışanlar AS T1

WHERE Maaş >=

(SELECT Avg(Maaş)

FROM Çalışanlar

WHERE T1.Ünvan = Çalışanlar.Ünvan) Order by Ünvan;

Önceki örnekte, AS saklı sözcüğü isteğe bağlıdır.

Bazı alt sorguların çapraz sorgular içinde (özellikle WHERE yan tümcesindeki doğrulamalar olarak) kullanılmasına izin verilir. Çapraz sorgularda alt sorguların çıktı olarak (SELECT listesindeki gibi) kullanılmasına izin verilmez.

SQL Alt Sorguları Örneği

Bu örnek, 1995'in ikinci üç aylık döneminde sipariş vermiş olan her müşterinin müşteri ve ilgili kişi adını listeler.

Bu örnek, SELECT deyimi örneğinde bulabileceğiniz EnumFields yordamını çağırır.

```
Sub SubQueryX()
    Dim dbs As Database, rst As Recordset
    ' Bu satyry, bilgisayarynyzdaki Northwind yolunu
    ' bulundurmak üzere değiştirin.
    Set dbs = OpenDatabase("Northwind.mdb")

    ' 1995'in ikinci üç aylık döneminde sipariş vermi
    ' olan her müşterinin müşteri ve ilgili kişi
    ' adyny listeler.
    Set rst = dbs.OpenRecordset("SELECT KişiAdı," _
        & " ŞirketAdı, KişiÜnvanı, Telefon" _
        & " FROM Müşteriler" _
        & " WHERE MüşteriNo" _
        & " IN (SELECT MüşteriNo FROM Siparişler" _
        & " WHERE SiparişTarihi Between #04/1/95#" _
        & " And #07/1/95#);")

    ' Recordset'i başlatır.
    rst.MoveLast

    ' Recordset içeriklerini yazdyrmak üzere EnumFields
    ' yordamını çağırır. Recordset nesnesini ve
    ' istenen alan genişliğini aktarır.
    EnumFields rst, 25
    dbs.Close
End Sub
```

- **SQL Ayrılmış Sözcükleri**

Aşağıdaki liste, SQL deyimlerinde kullanılan Microsoft Jet veritabanı alt yapısı tarafından kendine saklanmış tüm sözcükleri içerir. Listedeki sözcüklerden tamamı büyük harfli olmayanlar diğer uygulamalar için de saklı sözcük olanlardır. Bu nedenle, bu sözcüklerin Yardım konuları genel açıklamalar içerir ve SQL kullanımı üzerine odaklanmazlar.

Not Ardından yıldız işareti (*) gelen sözcükler saklı sözcüklerdir ancak bir Microsoft® Jet SQL deyiminin (örneğin, **Level** ve **TableID**) içeriğinde herhangi bir anlam taşımazlar. Altı çizili olmayan sözcüklerin kendilerine bağlı açıklamaları yoktur.

A

ABSOLUTE	ANY
ADD	ARE
8ADMINDB	AS
ALL	ASC
Alphanumeric — See TEXT	ASSERTION
ALTER	AUTHORIZATION
ALTER TABLE	AUTOINCREMENT — See COUNTER
And	**Avg**
AS	

B-C

BEGIN	COLLATION
Between	COLUMN
BINARY	COMMIT
BIT	COMP, COMPRESSION
BIT_LENGTH	CONNECT
BOOLEAN — Bkz. BIT	CONNECTION
BOTH	CONSTRAINT, CONSTRAINTS
BY	CONTAINER
BYTE	CONTAINS
CASCADE	CONVERT
CATALOG	**Count**
CHAR, CHARACTER — Bkz. TEXT	COUNTER
CHAR_LENGTH	CREATE
CHARACTER_LENGTH	CURRENCY
CHECK	CURRENT_DATE
CLOSE	CURRENT_TIME
CLUSTERED	CURRENT_TIMESTAMP
COALESCE	CURRENT_USER
COLLATE	CURSOR

D

DATABASE	DISALLOW
DATE — Bkz. DATETIME	DISCONNECT
DATETIME	DISTINCT
DAY	DISTINCTROW
DEC, DECIMAL	DOMAIN
DECLARE	DOUBLE
DELETE	DROP
DESC	

E-H

Eqv	FOREIGN
EXCLUSIVECONNECT	FROM
EXEC, EXECUTE	FROM Clause
EXISTS	GENERAL — Bkz. LONGBINARY
EXTRACT	GRANT
FALSE	GROUP
FETCH	GUID
FIRST	HAVING
FLOAT, FLOAT8 — Bkz. DOUBLE	HOUR
FLOAT4 — Bkz. SINGLE	

I

IDENTITY	INPUT
IEEEDOUBLE — Bkz. DOUBLE	INSENSITIVE
IEEESINGLE — Bkz. SINGLE	INSERT
IGNORE	INSERT INTO
IMAGE	INT, INTEGER, INTEGER4 — Bkz. LONG
Imp	INTEGER1 — Bkz. BYTE
In	INTEGER2 — Bkz. SHORT
IN	INTERVAL
INDEX	INTO
INDEXCREATEDB	**Is**
INNER	ISOLATION

J-M

JOIN	LONGTEXT
KEY	LOWER
LANGUAGE	MATCH
LAST	**Max**
LEFT	MEMO — Bkz. LONGTEXT
Level*	**Min**
Like	MINUTE
LOGICAL, LOGICAL1 — Bkz. BIT	**Mod**
LONG	MONEY — Bkz. CURRENCY
LONGBINARY	MONTH
LONGCHAR	

N-P

NATIONAL	**Outer***
NCHAR	OUTPUT
NONCLUSTERED	OWNERACCESS
Not	PAD
NTEXT	PARAMETERS
NULL	PARTIAL
NUMBER — Bkz. DOUBLE	PASSWORD
NUMERIC — Bkz. DECIMAL	PERCENT
NVARCHAR	PIVOT
OCTET_LENGTH	POSITION
OLEOBJECT — Bkz. LONGBINARY	PRECISION

ON	PREPARE
OPEN	PRIMARY
OPTION	PRIVILEGES
Or	PROC, PROCEDURE
ORDER	PUBLIC

Q-S

REAL — Bkz. SINGLE	SMALLDATETIME
REFERENCES	SMALLINT — Bkz. SHORT
RESTRICT	SMALLMONEY
REVOKE	SOME
RIGHT	SPACE
ROLLBACK	SQL
SCHEMA	SQLCODE, SQLERROR, SQLSTATE
SECOND	**StDev**
SELECT	**StDevP**
SELECTSCHEMA	STRING — Bkz. TEXT
SELECTSECURITY	SUBSTRING
SET	**Sum**
SHORT	SYSNAME
SINGLE	SYSTEM_USER
SIZE	

T-Z

TABLE	UPDATEOWNER
TableID*	UPDATESECURITY
TEMPORARY	UPPER
TEXT	USAGE
TIME — Bkz. DATETIME	USER
TIMESTAMP	USING
TIMEZONE_HOUR	VALUE
TIMEZONE_MINUTE	VALUES
TINYINT	**Var**
TO	VARBINARY — Bkz. BINARY
TOP	VARCHAR — Bkz. TEXT
TRAILING	**VarP**
TRANSACTION	VARYING
TRANSFORM	VIEW
TRANSLATE	WHEN
TRANSLATION	WHENEVER
TRIM	WHERE
TRUE	WITH
UNION	WORK
UNIQUE	**Xor**
UNIQUEIDENTIFIER	YEAR
UNKNOWN	YESNO — Bkz. BIT
UPDATE	ZONE
UPDATEIDENTITY	

➢ SQL Veri Türleri

Microsoft Jet veritabanı alt yapısı SQL veri türleri, Microsoft® Jet veritabanı alt yapısı ile tanımlanan 13 birincil veri türünden ve bu veri türlerinin tanıdığı çeşitli geçerli eş anlamlılardan oluşur.

Aşağıdaki tablo birincil veri türlerini listeler. Eş anlamlılar, Microsoft Jet Veritabanı Alt Yapısı SQL Saklı Sözcükleri içinde belirlenmiştir.

Veri türü	Depolama boyutu	Açıklama
BINARY	1 bayt / karakter	Herhangi bir türdeki veri, bu tür alanda saklanabilir. Veri üzerinde çeviri yapılmaz (örneğin metin şekline). Verilerin ikili alana nasıl girildiği, çıktı olarak nasıl görüneceğini belirler.
BIT	1 bayt	Yalnızca Evet ve Hayır değerlerinden birini içeren Evet ve Hayır değerleri ve alanları
TINYINT	1 bayt	0 ile 255 arasında bir tamsayı.
MONEY	8 bayt	– 922,337,203,685,477.5808 ve 922,337,203,685,477.5807 arasında ölçeklenmiş bir tamsayı.
DATETIME (See DOUBLE)	8 bayt	100 ve 9999 yılları arasında bir tarih veya saat değeri.
UNIQUEIDENTIFIER	128 bit	Uzak yordam çağrılarında kullanılan benzersiz bir kimlik numarası.
REAL	4 bayt	Negatif değerler için – 3.402823E38 ile – 1.401298E-45 arasında ve pozitif sayılar ile 0 için 1.401298E-45 ile 3.402823E38 arasında tek duyarlıklı kayan noktalı değerdir.
FLOAT	8 bayt	Negatif değerler için – 1.79769313486232E308 ile – 4.94065645841247E-324 arasında ve pozitif sayılar ile 0 için 4.94065645841247E-324 ile 1.79769313486232E308 arasında çift duyarlıklı kayan noktalı değerdir.
SMALLINT	2 bayt	– 32,768 ile 32,767 arasında kısa tamsayıdır. (Bkz. Notlar)
INTEGER	4 bayt	– 2,147,483,648 ile 2,147,483,647 arasında uzun tamsayıdır. (Bkz. Notlar)
DECIMAL	17 bayt	1028 - 1 ile - 1028 - 1 arasında değerleri tutan kesin bir sayısal veri türü. Hem duyarlılığı (1 - 28) hem de ölçeği (0 - tanımlı duyarlık) tanımlayabilirsiniz. Varsayılan duyarlık ve ölçek sırasıyla 18 ile 0'dır.
TEXT	2 bayt / karakter (Bkz. Notlar)	Sıfır ile en çok 2.14 gigabayt arasıdır.
IMAGE	Gerektiği kadar	Sıfır ile en çok 2.14 gigabayt arasıdır. OLE nesneleri için kullanılır.
CHARACTER	2 bayt / karakter (Bkz. Notlar)	Sıfır ile 255 karakter arasıdır.

Notlar

- Hem başlangıç hem de artış, ALTER TABLE deyimini kullanarak değiştirilebilir. Sütun için otomatik olarak oluşturulan ve tabloya eklenen yeni satırlar, yeni başlangıç ve artış değerlerini temel alan değerlere sahip olacaklardır. Yeni başlangıç ve artış değerleri, önceki

başlangıcı ve artışı temel alarak oluşturulmuş değerlerle eşleşen değerler oluştururursa, yinelemeler sözkonusu olacaktır. Sütun bir birincil anahtar ise, yinelenen değerler içeren yeni satırlar eklemek hatalara neden olabilir.

- Otomatik artışlı bir sütun için son kullanılan değeri bulmak için şu deyimi kullanabilirsiniz: SELECT @@IDENTITY. Bir tablo adı belirtemezsiniz. Döndürülen değer, otomatik artış sütunu içeren ve güncelleştirilmiş olan son tablodandır.

- TEXT (MEMO olarak da bilinir) veya CHAR (belirli bir uzunlukta TEXT(n) olarak da bilinir) olarak tanımlanmış alanlardaki karakterler, Unicode sunu biçiminde saklanırlar. Unicode karakterleri her karakteri saklamak için aynı şekilde iki bayt gerektirirler. Bu nedenle, daha çok karakter verilerine sahip var olan Microsoft® Jet veritabanlarında, veritabanı Microsoft Jet version 4.0 biçimine dönüştürüldüğünde veritabanı dosyasının boyutu yaklaşık olarak iki katına çıkacaktır. Ancak, Tek Baytlık Karakter Kümeleri (SBCS) olarak bilinen çoğu karakter kümesinin Unicode sunumu kolaylıkla tek bayta sıkıştırılabilir. Ayrıntılar için CREATE TABLE konusuna bakın. COMPRESSION özelliğinde bir CHAR sütunu tanımlarsanız, veriler saklanırken otomatik olarak sıkıştırılacak ve sütundan alınırken de otomatik olarak genişletileceklerdir.

➢ ODBC Tek Verili İşlevleri

Microsoft® Jet SQL, tek verili işlevlerde ODBC tanımlı sözdizimini kullanımı destekler. Örneğin, aşağıdaki sorgu:

```
SELECT DAILYCLOSE, DAILYCHANGE FROM DAILYQUOTE
WHERE {fn ABS(DAILYCHANGE)} > 5
```

stok fiyatındaki değişimin mutlak değeri beşten büyük olan tüm satırları verecektir.

ODBC tanımlı tek verili işlevlerinin alt kümesi desteklnemektedir. Aşağıdaki tablo desteklenen işlevleri listelemektedir.

Bağımsız değişkenlerin tanımı için ve SQL deyimindeki işlevlerin sözdizimlerinin tam bir açıklaması için ODBC kitaplarına bakın.

Dize İşlevleri

ASCII	LENGTH	RTRIM
CHAR	LOCATE	SPACE
CONCAT	LTRIM	SUBSTRING
LCASE	RIGHT	UCASE
LEFT		

Sayısal İşlevler

ABS	FLOOR	SIN
ATAN	LOG	SQRT
CEILING	POWER	TAN
COS	RAND	MOD
EXP	SIGN	

Saat ve Tarih İşlevleri

CURDATE	DAYOFYEAR	MONTH
CURTIME	YEAR	WEEK
NOW	HOUR	QUARTER
DAYOFMONTH	MINUTE	MONTHNAME
DAYOFWEEK	SECOND	DAYNAME

Veri Türü Dönüşümü

CONVERT	Dize metinleri aşağıdaki veri türlerine dönüştürülebilir: SQL_FLOAT, SQL_DOUBLE, SQL_NUMERIC, SQL_INTEGER, SQL_REAL, SQL_SMALLINT, SQL_VARCHAR and SQL_DATETIME.

➢ Microsoft Jet SQL ve ANSI SQL Karşılaştırması

Microsoft Jet veritabanı alt yapısı SQL genellikle ANSI-89 Düzey 1 uyumludur. Ancak bazı ANSI SQL özellikleri Microsoft® Jet SQL içinde uygulanmamıştır. Microsoft Jet sürüm 4.X'in çıkması ile, Jet için Microsoft OLE DB Sağlayıcısı, daha fazla ANSI-92 SQL sözdizimini sağlamıştır. Buna karşılık da, Microsoft Jet SQL, ANSI SQL içinde desteklenmeyen saklı sözcükler ve özellikler bulundurur.

Ana Farklılıklar

- Microsoft Jet SQL ve ANSI SQL'in her birinin farklı saklı sözcükleri ve veri türleri vardır. Ayrıntılı bilgi için, Microsoft Jet Veritabanı Alt Yapısı SQL Saklı Sözcükleri ve Eşdeğer ANSI SQL Veri Tüleri konularına bakın. Jet için Microsoft OLE DB Sağlayıcısı'nı Jet 4.X ile kullanırsanız, ek saklı sözcükler de olur.

- **Between...And** yapısında, aşağıdaki sözdizimi ile farklı kuralları uygulanır:

  ifade1 [NOT] **Between** *değer1* **And** *değer2*

  Microsoft Jet SQL'de, *değer1 değer2*'den büyük olabilir; ANSI SQL'de *değer1 değer2*'ye eşit veya ondan küçük olmalıdır.

- Microsoft Jet SQL, **Like** işleci ile kullanmak üzere hem ANSI SQL joker karakterlerini hem de Microsoft Jet'e özel joker karakterlerini destekler.. ANSI ve Microsoft Jet joker karakterlerinin kullanımı karşılıklı özeldir. Bunlardan birini ya da diğerini kullanmalı ancak ikisini karıştırmamalısınız. ANSI SQL joker karakterleri ancak Jet 4.X ve Jet için Microsoft OLE DB Sağlayıcısı kullanılırken geçerlidir. ANSI SQL joker karakterlerini Microsoft Access veya DAO ile kullanmayı denerseniz, bu karakterler düz metin karakterleri olarak kabul edileceklerdir. Jet 4.X ve Jet için Microsoft OLE DB Sağlayıcısı'nı kullanırken bunun tersi geçerlidir.

Eşleme karakteri	Microsoft Jet SQL	ANSI SQL
Herhangi tek bir karakter	?	_ (alt çizgi)
Sıfır veya daha fazla sayıda karakter	*	%

- Microsoft Jet SQL genellikle daha az sınırlayıcıdır. Örneğin, ifadelerde gruplandırmaya ve sıralamaya izin verir.

- Microsoft Jet SQL, daha güçlü ifadeleri destekler.

➢ Microsoft Jet SQL'in Gelişmiş Özellikleri

Microsoft Jet SQL, aşağıdaki gelişmiş özellikleri sağlar:

☐ Çapraz sorgu desteği sağlayan TRANSFORM deyimi

- **StDev** ve **VarP** gibi ek toplam işlevleri.

Parametre sorguları tanımlamak için PARAMETERS bildirimi

➢ Microsoft Jet SQL İçinde Desteklenmeyen ANSI SQL Özellikleri

Microsoft Jet SQL şu ANSI SQL özelliklerini desteklemez:

- DISTINCT toplam işlevi başvuruları. Örneğin, Microsoft Jet SQL SUM(DISTINCT *sütunadı*) sözdizimine izin vermez.
- LIMIT TO *nn* ROWS yan tümcesi, sorgunun döndüreceği satır sayısını sınırlandırmak için kullanılır. Sorgu kapsamını sınırlandırmak için yalnızca WHERE yan tümcesini kullanabilirsiniz.

➢ Eşdeğer ANSI SQL Veri Türleri

Aşağıdaki tablo ANSI SQL veri tülerini, onlara eşdeğer Microsoft Jet veritabanı alt yapısı SQL veri türlerini ve geçerli eş anlamlılarını listeler. Ayrıca, eşdeğer Microsoft® SQL Server™ veri türleri de listelenir.

ANSI SQL veri türü	**Microsoft Jet SQL veri türü**	**Eş anlamlısı**	**Microsoft SQL Server veri türü**
BIT, BIT VARYING	BINARY (Bkz. Notlar)	VARBINARY, BINARY VARYING BIT VARYING	BINARY, VARBINARY
Desteklenmez	BIT (Bkz. Notlar)	BOOLEAN, LOGICAL, LOGICAL1, YESNO	BIT
Desteklenmez	TINYINT	INTEGER1, BYTE	TINYINT
Desteklenmez	COUNTER (Bkz. Notlar)	AUTOINCREMENT	(Bkz. Notlar)
Desteklenmez	MONEY	CURRENCY	MONEY
DATE, TIME, TIMESTAMP	DATETIME	DATE, TIME (Bkz. Notlar)	DATETIME
Desteklenmez	UNIQUEIDENTIFIER	GUID	UNIQUEIDENTIFIER
DECIMAL	DECIMAL	NUMERIC, DEC	DECIMAL
REAL	REAL	SINGLE, FLOAT4, IEEESINGLE	REAL
DOUBLE PRECISION, FLOAT	FLOAT	DOUBLE, FLOAT8, IEEEDOUBLE, NUMBER (Bkz Notlar)	FLOAT
SMALLINT	SMALLINT	SHORT, INTEGER2	SMALLINT
INTEGER	INTEGER	LONG, INT, INTEGER4	INTEGER
INTERVAL	Desteklenmez		Desteklenmez
Desteklenmez	IMAGE	LONGBINARY, GENERAL, OLEOBJECT	IMAGE
Desteklenmez	TEXT (Bkz. Notlar)	LONGTEXT, LONGCHAR, MEMO, NOTE, NTEXT (Bkz. Notlar)	TEXT
CHARACTER, CHARACTER VARYING, NATIONAL CHARACTER, NATIONAL CHARACTER VARYING	CHAR (Bkz. Notlar)	TEXT(n), ALPHANUMERIC, CHARACTER, STRING, VARCHAR, CHARACTER VARYING, NCHAR, NATIONAL CHARACTER, NATIONAL CHAR, NATIONAL CHARACTER	CHAR, VARCHAR, NCHAR, NVARCHAR

		VARYING, NATIONAL CHAR VARYING (Bkz. Notlar)	

Notlar

- ANSI SQL BIT veri türü, Microsoft Jet SQL BIT veri türüne karşılık gelmez. Bunun yerine, BINARY veri türüne karşılık gelir. Microsoft Jet SQL BIT veri türünün ANSI SQL eşdeğeri yoktur.
- TIMESTAMP artık DATETIME eş anlamlısı olarak desteklenmez.
- NUMERIC artık FLOAT veya DOUBLE eş anlamlısı olarak desteklenmez. NUMERIC artık DECIMAL eş anlamlısı olarak kullanılır.
- Bir LONGTEXT alanı her zaman Unicode sunu biçiminde saklanır.
- TEXT veri türü adı, TEXT(25) gibi herhangi bir uzunluk belirtilmeden kullanılırsa, bir LONGTEXT alanı oluşturulur. Bu, veri türlerinin Microsoft SQL Server ile uyumlu olmasını sağlayacak CREATE TABLE deyimlerinin yazılmasını sağlar.
- Bir CHAR alanı her zaman, ANSI SQL NATIONAL CHAR veri türünün eşdeğeri olan Unicode sunu biçiminde saklanır.
- TEXT veri türü adı kullanılır ve TEXT(25) gibi isteğe bağlı bir uzunluk belirtilirse, alanın veri türü, CHAR veri türüne eşdeğerdir. Bu, TEXT veri türünün bir uzunluk belirtilmeden Microsoft SQL Server ile kullanılmasına izin verirken, çoğu Microsoft Jet uygulamasında geriye doğru uyumluluğu sağlar.

➢ Dize Karşılaştırmasında Joker Karakterlerini Kullanma

Yerleşik örnek eşleme, dize karşılaştırmaları yapmak için çok yönlü bir araç sağlar. Aşağıdaki tablo, **Like** işleci ile kullanabileceğiniz joker karakterlerini ve bunların dikkate aldıkları hane veya dize sayısını gösterir.

***Örnek* içindeki karakter sayısı**	***İfade* içindeki eşleme**
? veya _ (alt çizgi)	Herhangi tek bir karakter
* veya %	Sıfır veya daha fazla sayıda karakter
#	Herhangi tek bir hane (0 — 9)
[*karakterlistesi*]	*Karakterlistesi* içindeki herhangi tek bir karakter
[!*karakterlistesi*]	*Karakterlistesi* içinde bulunmayan herhangi tek bir karakter

İfade içindeki herhangi tek bir karakterle eşlemek üzere köşeli ayraçlar ([ ]) içinde yer alan bir veya daha fazla karakteri (*karakterlistesi*) kullanabilirsiniz; *karakterlistesi*, rakamlar dahil ANSI karakter kümesinde bulunan neredeyse tüm karakterleri içerebilir. Köşeli ayraçlar içinde yer alıyorsa, açma ayracı ([ ), soru işareti (?), rakam işareti (#) ve yıldız (*) karakteri gibi özel karakterleri, kendilerini eşlemek üzere kullanabilirsiniz. Bir grup içinde kapatma ayracını ( ]) kendini eşlemek üzere kullanamazsınız, ancak grubun dışında bağımsız bir karakter olarak kullanabilirsiniz.

Karakterlistesi, köşeli ayraçlar içinde bulunan basit karakter listelerine ek olarak, aralığın alt ve üst sınırlarını ayırmak üzere bir tire işareti (-) kullanarak bir karakter aralığı da belirtebilir. Örneğin *örnek* olarak [A-Z] yazarsanız, *ifade* içindeki ilgili karakterin konumu A ile Z arasında herhangi bir büyük harf içeriyorsa eşleme gerçekleşir. Köşeli ayraçlar içinde, aralıkları birbirinden ayırmadan çok sayıda aralık bulundurabilirsiniz. örneğin [a-zA-Z0-9] yazarsanız herhangi alfasayısal bir karakter eşlenir.

(%) ve (_) ANSI SQL joker karakterlerinin yalnızca Microsoft® Jet sürüm 4.X ve Microsoft OLE DB Provider for Jet için geçerli olduğunu unutmamak gerekir. Bunlar, Microsoft Access veya DAO ile kullanılırlarsa, düz metin olarak kabul edileceklerdir.

Örnek eşlemenin diğer önemli kuralları şunlardır:

- *Karakterlistesi* başına konan bir ünlem işareti (!), eşlemenin ancak, *karakterlistesi* içindekiler hariç herhangi bir karakterin *ifade* içinde bulunması halinde olacağını belirtir. Köşeli ayraçlar dışında kullanıldığında ünlem işareti kendisini belirtir.

- Tire işaretini (-), kendisini belirtmek üzere *karakterlistesinin* başında (varsa ünlem işaretinden sonra) veya sonunda kullanabilirsiniz. Başka herhangi bir konumda yer alan tire işareti, ANSI karakterleri aralığını belirtir.

- Bir karakter aralığı belirtilirken, karakterler artan sıralama düzeninde olmalıdır (A-Z veya 0-100). [A-Z] geçerli bir örnektir ancak [Z-A] değildir.

- [ ] Karakter sırası gözardı edilir ve sıfır uzunlukta dize (“”) olarak değerlendirilir.

SÖZLÜK

Başlangıç

Counter ya da **Identity** veri türleriyle kullanılan seçime bağlı parametredir. SQL DDL'in, yalnızca Microsoft® OLE DB Provider for Jet içinden kullanılması durumunda geçerlidir. Başlangıç değeri de denir. Başlangıç, **Counter** ya da **Identity** veri türlerine ilişkin seçime bağlı iki parametreden ilkidir.

Aşağıda, parametreli SQL deyimlerine bir örnek verilmiştir:

CREATE TABLE Müşteriler (CustId IDENTITY (100, 10) CONSTRAINT pkCustomers PRIMARY KEY, CFrstNm VARCHAR(10), CLstNm VARCHAR(15));

İlk satırdaki CustId alanının değeri 100, ikinci satırın değeri 110 olacaktır.

Seçime bağlı parametreler kullanılırken (1,1) değeri içerilmezse, Jet başlangıç değerini, tablodaki en üst değere getirmeyecektir. Bu davranış, Jet'in önceki sürümlerindekinden farklıdır. Veritabanı düzenlenirken, seçime bağlı parametreler aktarılmazsa ya da (1,1) değerinden başka bir değer kullanılırsa, başlangıç değeri, tablodaki **Counter** ya da **Identity** sütunlarının en yüksek değerine getirilir.

Aralık

Counter ya da **Identity** veri türleriyle kullanılan seçime bağlı parametredir. SQL DDL'in, yalnızca Microsoft® OLE DB Provider for Jet içinden kullanılması durumunda geçerlidir. Bu değer, bir tablodaki **Counter** ya da **Identity** veri türünü içeren yeni bir satıra, sıradaki değerin atanmasında kullanılan değerdir. Aralık, **Counter** ya da **Identity** veri türlerine ilişkin seçime bağlı iki parametreden ikincisidir.

Text Veri Türü (Uzunluk Belirtimi Olmaksızın)

Bir alan veri türüdür. Belirli bir uzunluk bildirmeksizin tanımlanmış olan metin alanları Memo alanıyla özdeştir ve 2,14 GB'a kadar karakter verileri içerebilirler.

Unicode Temsil Biçimi

Unicode Kongresi'nce oluşturulan ve dünyadaki tüm diller için gerekli olan tüm karakterleri içeren 16 bitlik bir karakter kümesidir. Yazılımların çok dilde işletimini kolaylaştırmak üzere Unicode kullanımı giderek yaygınlaşmaktadır.

Yordam

SQL sözdiziminin, parametrelerin aktarılmasında SELECT deyimlerine (satır veren sorgular), ayrıca, DML (satır vermeyen sorgular) UPDATE, INSERT, DELETE ve SELECT INTO deyimlerine izin vermesi dışında aynen bir görünüme benzer. DAO nesne modeli kullanılmaksızın SQL sözdiziminden oluşturulabilmesi dışında, querydef ile aynıdır.

Görünüm

Görünüm, sanal tablonun ANSI'deki tanımlamasıdır. Bir görünüm, Access'teki parametresiz kullanılan bir SELECT deyiminden oluşan saklı bir sorguyla eşanlamlıdır. Görünüm, uygulamanın çalıştırılması sırasında ortaya çıkan saklı bir tanımdır. Fiziksel anlamda herhangi bir veri içermemesiyle bir tablodan ayrılır. Yalnızca, başka bir kaynaktan aldığı verileri iletir.

Microsoft® OLE DB Provider for Jet yardımıyla kullanılan yeni ANSI SQL CREATE VIEW sözdizimi aracılığıyla bir görünüm oluşturulabilir. Ya da bunun yerine, SQL DML deyimlerine ve parametreli SELECT deyimlerine izin vermek amacıyla bir yordam oluşturulabilir. Microsoft OLE DB Provider for Jet yardımıyla kullanılan yeni CREATE PROCEDURE sözdizimiyle bunu yapabilirsiniz.

- Dosya Sonu -