Tablolar:

Tablolar, günlük hayattaki listelere karşılık gelir. Öğrencii listesi yerine öğrenci tablosu, öğretmen listesi yerine öğretmen tablosu, veli listesi yerine veli tablosu şeklinde tablolar tanımlayaibliriz.

Günlük hayatta listelere karşılık, ilişkisel veritabanında tablolar yer alır. Bir tablo, normalizasyon kuralları çerçevesinde belirlenmiş bir varlığa ait bilgileri tutmak üzere tasarlanmalıdır. Daha sonra bu tablolar arasında bir ilişki tanımlanabileceği için, bu türden veritabanı yaklaşımlarına "İlişkisel Veritabanı Yönetim Sistemleri" denir.

Tablo Oluşturma için genel ifade şu şekildedir:

CREATE TABLE tablo_adi
(kolon_adi1 veri_tipi[NOT NULL][,
kolon_adi2 veri_tipi[NOT NULL],.....])

Örnek-3:

```
CREATE TABLE Kitap1(
kitapNo INTEGER IDENTITY(1,1) NOT NULL CONSTRAINT cnsKitapNo PRIMARY KEY,
kitapAdi VARCHAR(63) NOT NULL,
ISBNNo VARCHAR(15),
Turu VARCHAR(20),
sayfaSayisi INTEGER,
kitapOzeti VARCHAR(255)
);
```

ile örneğimizdeki tabloyu oluşturabiliriz.

Kısıtlar(Constraint) oluşturma

Kısıtlayıcılar, tabloların oluşturulması sırasında tabloların bir parçası olarak tanılanır. MS Access ve SQLServer'de kullanılır. Aşağıdaki işlevleri yerine getirebilirler:

1. INDEX gibi, birincil anahtar alan tanımlayabilirler (PRIMARY KEY): Bu durumda standart olarak Clustred Index gibi davranırlar.(bkz. INDEX'ler)
2. INDEX gibi tekil alan belirleyebilirler. (UNIQUE KEY)
3. Tablo oluşturma esnasında da yapılabilen, (NOT NULL) alan belirleyebilirler
4. Yabancı Anahtar Belirleyebilirler.(FOREIGN KEY)

Primary Key: Bir tablodaki, her bir satırın yerine vekil olabilecek bir anahtar veridir. Tekrarlamaması gerekir. Standart olarak bir tabloda verilerin, fiziksel hafıza üstünde de hangi alana göre dizileceğini de primary key belirler. Bu, bazen bir tek alan olabileceği gibi, bazen birden fazla alan da birleşerek bir birincil anahtar oluşturabilir.

Unique Key: Unique key olarak tanımlanan alan için bir değer sadece bir kere girilebilir. Bir başka satıra daha aynı verinin girilmesine izin verilmez. Primary Key'den farklı olarak Unique Key, NULL değerini alabilir.

Foreign Key: Bir tabloya girilebilecek değerleri başka bir tablonun bir belli alanında yer alabilecek veri grubu ile sınırlandırmaya ve en önemlisi de ilişkilendirmeye yarar.

Örneğin, olmayan bir kitabın ödünç tablosuna eklenememesi ve ödünç tablosuna eklene bir kitabın numarası aracılığıyla detay bilgilerine erişilmesi. Burada Kitap.KitaNo Brinicil anahtar alan; Odunc.kitapNo ise yabancı anahtardır. (bkz. Şekil1.6)

Genel yapısı şu şekildedir.

...
CONSTRAINT constraint_name PRIMARY KEY | UNIQUE | NOT NULL | REFERENCES foreign_table [(foreign_field1[,foreign_field2,..])]

Tablolar oluşturulurken, bazı alanlara girilebilecek değerler ile ilgili kısıtlar koymak zorunda kalabiliriz. Böylesi durumlarda CONSTRAINT'ler kullanılır.

CONSTRAINT'ler aslında INDEX'lere benzerler ama indekslerden farklı olarak bir tek tablo üstünde etkili olmayabilirler. Özellikle yabancı anahtar zorlayıcısı ilişkisel veri girişi için oldukça etkili bir zorlayıcıdır. Ancak bir Foreign Key tanımı yapabilmek için, FOREIGN KEY yabancı anahtarının asıl tablosunda birincil anahtar olması gerekir.

Örnek-4:

```
CREATE TABLE Kitap (
        kitapNo int NOT NULL ,
        kitapAdi varchar63),
        ISBNNo varchar (15),
        sayfaSayisi int ,
        kitapOzeti varchar (255)
);
```

veya

```
CREATE TABLE odunc(
        oduncNo int NOT NULL ,
        kitapNo int NOT NULL ,
        uyeNo int NOT NULL ,
        vermeTarihi datetime NOT NULL,
        vermeSuresi int NOT NULL ,
        geldiMi bit
);
```

Tablo silme:

Bir tabloyu kullanmaktan vazgeçersek:

DROP TABLE tablo_adi

Tabloda Değişiklik Yapma:
Bir tabloya sütun eklemek veya tablodan sütun silmek için ALTER TABLE deyimi kullanılır.

ALTER TABLE tablo_adi
{ADD{COLUMN alan alan_tipi [(boyut)][NOT NULL][CONSTRAINT indeks]
CONSTRAINT coklu_indeks}
DROP{COLUMN alan | CONSTRAINT constraint_adi}
}

Örnek-5:
Kitap tablosuna kitap bedeli adında yeni bir tamsayı sütun ekleyelim.

```
ALTER TABLE Kitap ADD kitapBedeli INTEGER;
```

Örnek-6:
Bu sütunun boş geçilememesini istese idik:

```
ALTER TABLE Kitap ALTER COLUMN  kitapBedeli INTEGER NOT NULL;
```

dememiz gerekirdi.

Örnek-7:
Bu sütunu silmek için,

```
ALTER TABLE Kitap DROP COLUMN kitapBedeli;
```