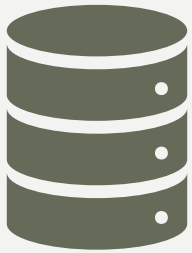
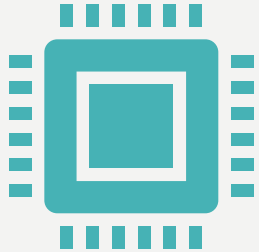
SQLİTE VERİTABANI VE TABLO OLUŞTURMA

GÖZDE MİHRAN ALTINSOY

VERİTABANI NEDİR ?



Veritabanı; uygulamalarımızda , web sitelerimizde veya en genel anlamda programlarımızda gerekli olan bilgileri depoladığımız bir yapıdır.



Örneğin , üye olduğumuz bir platformun kullanıcıları, gönderileri tuttuğu veritabanları gibi. Günümüzde kullanılan bazı veritabanı türleri şunlardır;

Relational Database (İlişkisel Veritabanları) :

Tablolardan oluşur. Mysql, Sqlite vs.

DocumentBased Database (Doküman Veritabanları) :

Dokümanlardan oluşur. MongoDb, Azure DocumentDb

SQLITE VERİTABANI

Sqlite ilişkisel bir veritabanıdır ve bu veritabanı tablolardan oluşur. Her bir tablo veritabanında belli gruplanmış verileri tutar.

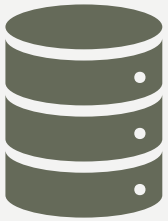
Örnek bir tablo yapısı şu şekildedir;

```
In [1]: from IPython.display import Image
        Image(filename='Tablo.png')
```

Out[1]:

İsim	Yazar	Yayınevi	Sayfa Sayısı
İstanbul Hatırası	Ahmet Ümit	Everest	561
Cerrah	Tess Gerritsen	Doğan Kitap	270
Budala	Dostoyevski	Sonsuz Kitap	712

SQLİTE VERİTABANI



Sqlite veritabanı diğer veritabanları gibi sunucu kurmamızı gerektirmez, yani sunucusuz bir veritabanıdır. Bu anlamda herhangi bir programımızın yanına direk olarak kurulabilir.

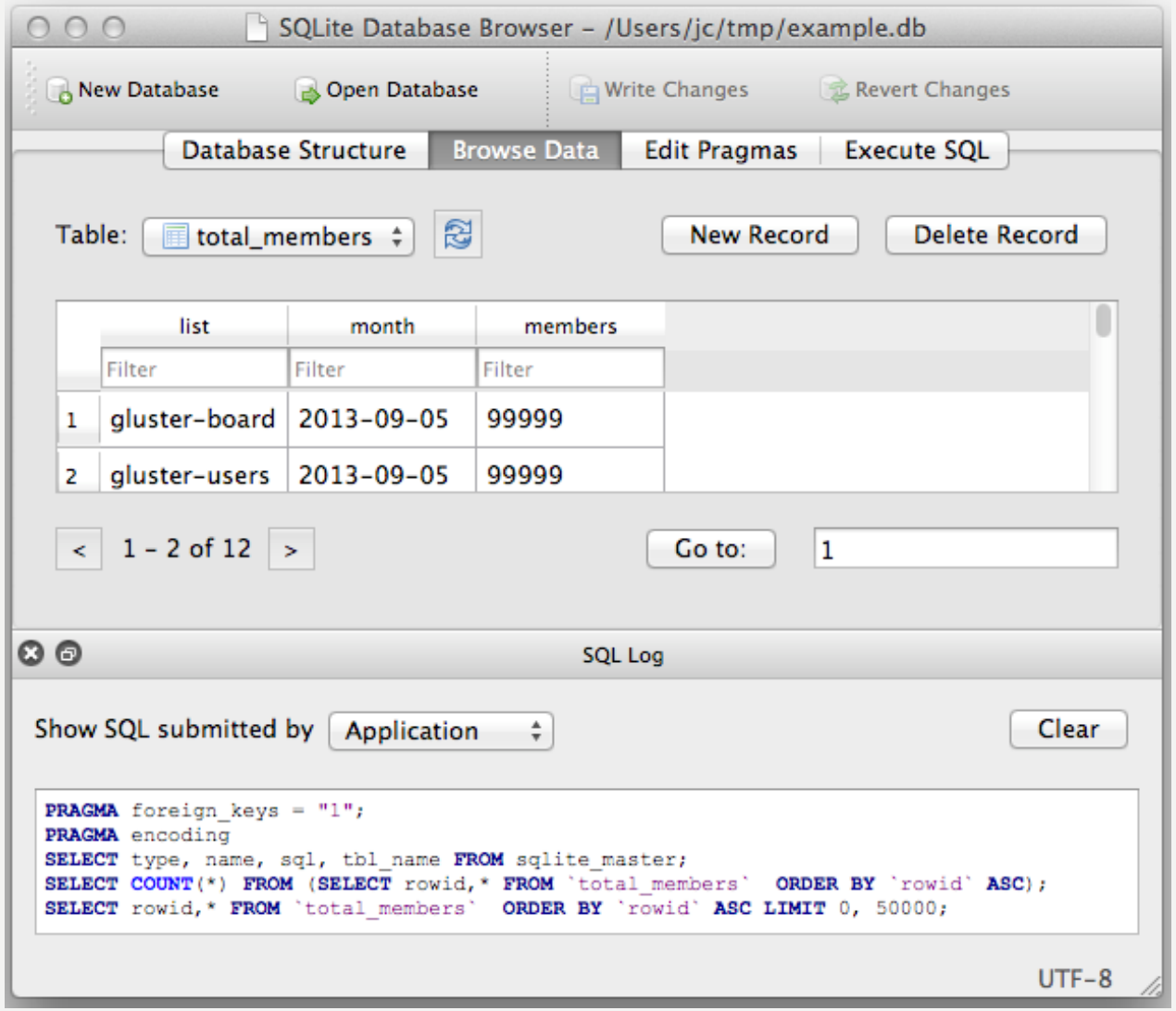
İlişkisel veritabanları tablo işlemlerini **SQL(Structured Query Language)** adında bir sorgulama diliyle gerçekleştirir.

Sql dilini incelemek isterseniz şu siteye bakabilirsiniz :https://www.w3schools.com/SQL/deFault.asp

YAZILIM

Sqlite veritabanında işlemler yaptıktan sonra veritabanını incelemek için kullanılabilecek yazılımını aşağıdaki linkten indirebilirsiniz:

http://sqlitebrowser.org/



SQLİTE VERİTABANI OLUŞTURMA

1.import sqlite3 - Modülümüzü dahil ediyoruz.
2.con = sqlite3.connect("kütüphane.db") - kütüphane.db veritabanını oluşturup bağlanıyoruz.
Eğer öyle bir veritabanı varsa bağlanıyoruz. Bağlantıyı con isimli bir değişkene atıyoruz.
3. Database üzerinde Sql sorgularını çalıştırmak için cursor bir tane imleç oluşturuyoruz.
cursor = con.cursor()
4. Veritabanı işlemlerinin sonunda con.close() ile bağlantımızı koparıyoruz.

SQLİTE VERİTABANI OLUŞTURMA

In [2]:
```
import sqlite3 # Sqlite'yı dahil ediyoruz

con = sqlite3.connect("kütüphane.db") # Tabloya bağlanıyoruz.

cursor = con.cursor() # cursor isimli değişken veritabanı üzerinde işlem yapmak için
#kullanacağımız imleç olacak.


con.close() # Bağlantıyı koparıyoruz.
```

KİTAPLIK TABLOSU OLUŞTURMA

Veritabanında ***kitaplık*** isimli bir tablo oluşturmak için şu 2 sorgudan birini kullanabiliriz.

*** CREATE TABLE kitaplık (İsim TEXT, Yazar TEXT, Yayınevi TEXT , Sayfa_Sayısı INT) *** - Bu sorgu kitaplık isimli bir tablo oluşturacak ve bu tablonun özellikleri İsim (TEXT --> String), Yazar(TEXT --> String), Yayınevi (TEXT ---> String), Sayfa_Sayısı(INT --- int) olacak. Ancak bu sorguyu arda arda çalıştırırsak "Tablo Already Exists" hatası alacağız.

*** CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT, Yazar TEXT, Yayınevi TEXT , Sayfa_Sayısı INT) *** - Bu sorgu tablo yoksa oluşturacak, tablo varsa hata vermeyecektir.

```
In [ ]: import sqlite3 # Sqlite'yı dahil ediyoruz

con = sqlite3.connect("kütüphane.db") # Tabloya bağlanıyoruz.

cursor = con.cursor()
# cursor isimli değişken veritabanı üzerinde işlem yapmak için kullanacağımız imleç olacak.

def tablo_oluştur():
    cursor.execute("CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT, Yazar TEXT, Yayınevi TEXT, Sayfa_Sayısı INT)")
    # Sorguyu çalıştırıyoruz.
    con.commit() # Sorgunun veritabanı üzerinde geçerli olması için commit işlemi gerekli.
tablo_oluştur()

con.close() # Bağlantıyı koparıyoruz.
```

KODLARIMIZ

Şu anda veritabanımıza bağlandık ve kitaplık isimli bir tablonun oluşturduk.

KİTAPLIK TABLOSU OLUŞTURMA

In [ ]:
```
import sqlite3 # Sqlite'yı dahil ediyoruz

con = sqlite3.connect("kütüphane.db") # Tabloya bağlanıyoruz.

cursor = con.cursor() # cursor isimli değişken veritabanı üzerinde işlem yapmak için kullanacağımız imleç olacak.

def tablo_oluştur():
    cursor.execute("CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT, Yazar TEXT, Yayınevi TEXT, Sayfa_Sayısı INT)")
    # Sorguyu çalıştırıyoruz.
    con.commit() # Sorgunun veritabanı üzerinde geçerli olması için commit işlemi gerekli.
tablo_oluştur()
con.close() # Bağlantıyı koparıyoruz.
```

KİTAPLIK TABLOMUZA VERİ EKLEMEK

INSERT INTO kitaplık VALUES('İstanbul Hatırası','Ahmet Ümit','Everest',561)

Not : SQL Sorguları büyük veya küçük harfle de yazılabilir. Örnek olarak

insert into kitaplık values ('İstanbul Hatırası','Ahmet Ümit','Everest',561)

Insert Into kitaplık Values ('İstanbul Hatırası','Ahmet Ümit','Everest',561)

KODLARIMIZ

In [ ]:

```
import sqlite3

con = sqlite3.connect("kütüphane.db")

cursor = con.cursor()

def tablo_oluştur():
    cursor.execute("CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT, Yazar TEXT, Yayınevi TEXT, Sayfa_Sayısı INT)")
    con.commit()
def deger_ekle(isim,yazar,yayınevi,sayfa_sayısı):
    cursor.execute("INSERT INTO kitaplık VALUES('İstanbul Hatırası','Ahmet Ümit','Everest',261)")
    con.commit()
deger_ekle()
con.close()
```

KULLANICIDAN ALDIĞIMIZ DEĞERLERİ TABLOMUZA EKLEMEK

In [ ]:
```
import sqlite3

con = sqlite3.connect("kütüphane.db")

cursor = con.cursor()

def tablo_oluştur():
    cursor.execute("CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT, Yazar TEXT, Yayınevi TEXT, Sayfa_Sayısı INT)")
    con.commit()
def deger_ekle(isim,yazar,yayınevi,sayfa_sayısı):
    cursor.execute("INSERT INTO kitaplık VALUES(?,?,?,?)",(isim,yazar,yayınevi,sayfa_sayısı))
    con.commit()
isim = input("İsim:")
yazar = input("Yazar:")
yayınevi = input("Yayınevi:")
sayfa_sayısı =  int(input("Sayfa Sayısı:"))


deger_ekle(isim,yazar,yayınevi,sayfa_sayısı)

con.close()
```

? işaretlerinin her birinin yerine fonksiyona değer olarak gönderdiğimiz isim , yazar, yayınevi ve sayfa sayısı bilgileri gidiyor ve tablomuza bu şekilde veri ekleyebiliyoruz.

TABLODAKİ VERİLERİ ÇEKME

Tablodan veri çekmek için şu **SQL** sorgularını kullanacağız.

Select * From kitaplık - Tablodaki tüm bilgileri almamızı sağlar.

Select İsim, Yazar From kitaplık Tablodaki tüm bilgileri sadece İsim ve Yazar özelliklerini almamızı sağlar.

Select * From kitaplık where Yayınevi = 'Everest' Sadece Yayınevi özelliği Everest olanları alır.

KODLARIMIZ
1.SORGUMUZ

In [ ]:
```
import sqlite3

con = sqlite3.connect("kütüphane.db")

cursor = con.cursor()
def tablo_olustur():
    cursor.execute("CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT,Yazar TEXT,Yayınevi TEXT,Sayfa_Sayısı INT)")
    con.commit()
def deger_ekle(isim,yazar,yayınevi,sayfa_sayısı):
    cursor.execute("Insert into kitaplık Values(?,?,?,?)",(isim,yazar,yayınevi,sayfa_sayısı))
    con.commit()
def verileri_al():
    cursor.execute("Select * From kitaplık") # Bütün bilgileri alıyoruz.
    data = cursor.fetchall() # Veritabanından bilgileri çekmek için fetchall() kullanıyoruz.
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
    # con.commit() işlemine gerek yok. Çünkü tabloda herhangi bir güncelleme yapmıyoruz.
verileri_al()
con.close()
```

KODLARIMIZ
2.SORGUMUZ

In [ ]:

```
import sqlite3

con = sqlite3.connect("kütüphane.db")

cursor = con.cursor()
def tablo_olustur():
    cursor.execute("CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT,Yazar TEXT,Yayınevi TEXT,Sayfa_Sayısı INT)")
    con.commit()
def deger_ekle(isim,yazar,yayınevi,sayfa_sayısı):
    cursor.execute("Insert into kitaplık Values(?,?,?,?)",(isim,yazar,yayınevi,sayfa_sayısı))
    con.commit()
def verileri_al():
    cursor.execute("Select * From kitaplık")
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
def verileri_al2():
    cursor.execute("Select İsim,Yazar From kitaplık") # Sadece İsim ve Yazar özelliklerini alıyoruz.
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
verileri_al2()
con.close()
```

KODLARIMIZ
3.SORGUMUZ

In [ ]:

```
import sqlite3

con = sqlite3.connect("kütüphane.db")

cursor = con.cursor()
def tablo_olustur():
    cursor.execute("CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT,Yazar TEXT,Yayınevi TEXT,Sayfa_Sayısı INT)")
    con.commit()
def deger_ekle(isim,yazar,yayınevi,sayfa_sayısı):
    cursor.execute("Insert into kitaplık Values(?,?,?,?)",(isim,yazar,yayınevi,sayfa_sayısı))
    con.commit()
def verileri_al():
    cursor.execute("Select * From kitaplık")
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
def verileri_al2():
    cursor.execute("Select İsim,Yazar From kitaplık")
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
def verileri_al3(yayınevi):
    cursor.execute("Select * From kitaplık where Yayınevi = ?",(yayınevi,)) # Sadece yayınevi ,Everest olan kitapları alıyoruz.
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
verileri_al3("Everest")
con.close()
```

TABLODAKİ VERİLERİ GÜNCELLEME VE SİLME

- **Verileri Güncelleme**

Tablodaki verileri güncelleme için şöyle bir sorgu kullanabiliriz.

```
Update kitaplık set  Yayınevi = 'Everest' where Yayınevi = 'Doğan Kitap'
-- Yayınevi 'Doğan Kitap' olan kitapların Yayınevi bilgilerini 'Everest' e günceller.
```

VERİLERİ GÜNCELLEME KODUMUZ

In [ ]:

```
import sqlite3

con = sqlite3.connect("kütüphane.db")

cursor = con.cursor()
def tablo_olustur():
    cursor.execute("CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT,Yazar TEXT,Yayınevi TEXT,Sayfa_Sayısı INT)")
    con.commit()
def deger_ekle(isim,yazar,yayınevi,sayfa_sayısı):
    cursor.execute("Insert into kitaplık Values(?,?,?,?)",(isim,yazar,yayınevi,sayfa_sayısı))
    con.commit()
def verileri_al():
    cursor.execute("Select * From kitaplık")
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
def verileri_al2():
    cursor.execute("Select İsim,Yazar From kitaplık")
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
def verileri_al3(yayınevi):
    cursor.execute("Select * From kitaplık where Yayınevi = ?",(yayınevi,))
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
def verigüncelle(yayınevi):
    cursor.execute("Update kitaplık set Yayınevi = ? where Yayınevi =  ?",("Everest",yayınevi))
    con.commit()

verigüncelle("Doğan Kitap")
con.close()
```

TABLODAKİ VERİLERİ GÜNCELLEME VE SİLME

- **Verileri Silme**

Tablodaki verileri silme için şöyle bir sorgu kullanabiliriz.

```
Delete From kitaplık where Yazar = 'Ahmet Ümit’
--Yazar özelliği 'Ahmet Ümit' olan kitapları tablodan siler.
```

VERİLERİ SİLME KODUMUZ

In [ ]:

```
import sqlite3

con = sqlite3.connect("kütüphane.db")

cursor = con.cursor()
def tablo_olustur():
    cursor.execute("CREATE TABLE IF NOT EXISTS kitaplık (İsim TEXT,Yazar TEXT,Yayınevi TEXT,Sayfa_Sayısı INT)")
    con.commit()
def deger_ekle(isim,yazar,yayınevi,sayfa_sayısı):
    cursor.execute("Insert into kitaplık Values(?,?,?,?)",(isim,yazar,yayınevi,sayfa_sayısı))
    con.commit()
def verileri_al():
    cursor.execute("Select * From kitaplık")
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
def verileri_al2():
    cursor.execute("Select İsim,Yazar From kitaplık")
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
def verileri_al3(yayınevi):
    cursor.execute("Select * From kitaplık where Yayınevi = ?",(yayınevi,))
    data = cursor.fetchall()
    print("Kitaplık Tablosunun bilgileri.....")
    for i in data:
        print(i)
def verigüncelle(yayınevi):
    cursor.execute("Update kitaplık set Yayınevi = ? where Yayınevi =  ?",("Everest",yayınevi))
    con.commit()
def verilerisil(yazar):
    cursor.execute("Delete  From kitaplık where Yazar = ?",(yazar,))
    con.commit()

verilerisil("Ahmet Ümit")
con.close()
```

PROJE 1 - ŞARKI PROJESİ GELİŞTİRMEYE ÇALIŞALIM.
ÖRNEK SQLİTE VERİTABANI

Örnek özellikler;
1.Şarkı İsmi
2.Sanatçı
3.Albüm
4.Prodüksiyon Şirketi
5.Şarkı Süresi (saniye olarak saklanabilir)
6.Çıkış Tarihi (yıl olarak saklanabilir)

Örnek Metodlar;
1.Veritabanındaki Toplam Şarkı Süresini Hesaplayan Metod
2.Şarkı Ekle
3.Şarkı Sil
4.Şarkı Güncelleme
5.Şarkı Adetini Yazdır
6.Bütün Değerleri Listele

Txt dosyasında aşağıdaki gibi kayıtlı olan verileri veritabanına atayalım. Aynı şekilde veritabanından aldığımız verileri dosyaya bu şekilde atamaya çalışalım.
Ör;

Selvi Boylum Al Yazmalım,Jehan Barbur,Evim Neresi,Ada Müzik,235,2017
Dön Bak Dünyaya,Pinhani,İnandığın Masallar,394,2006

PROJE 2 - SÜPERMARKET PROJESİ GELİŞTİRMEYE ÇALIŞALIM.
ÖRNEK SQLİTE VERİTABANI

Örnek özellikler;

1. Barkod
2. Ürün Adı
3. Ürün Türü
4. Miktar
5. Fiyat

Örnek Metodlar;

1. Ürün Ekle
2. Ürün Güncelle
3. Ürün Sil
4. Listele

Txt dosyasında aşağıdaki gibi kayıtlı olan verileri veritabanına atayalım. Aynı şekilde veritabanından aldığımız verileri dosyaya bu şekilde atamaya çalışalım.

Ör;

```
11111,Su,İçecek,1000,2
11112,Çikolata,Yiyecek,500,3
```