Kalıcı olarak bir bilgi topluluğu oluşturmak istiyorsak database kullanmamız gerekir, önceki örneklerde yazdığımız bilgiler hep siliniyordu fakat şuandan itibaren database kullanarak bu bilgileri kayıt edecek, ve sonradan kullanacağız.

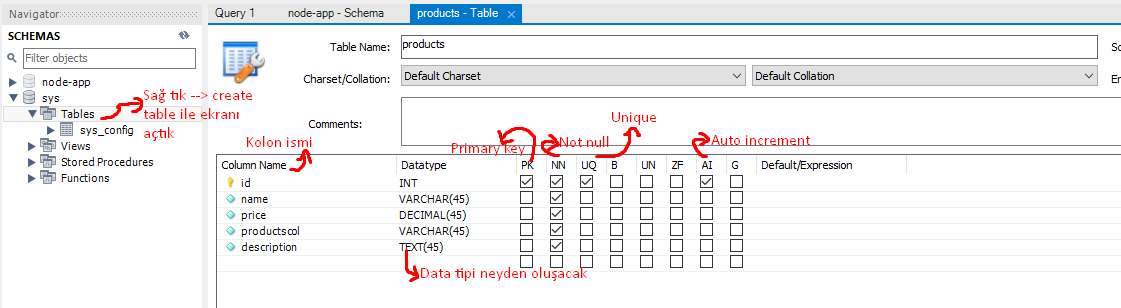
Database’ler SQL ve NoSQL olarak iki alanda incelenir: SQL 🡪 MySql, MsSql, SQLite, NoSQL 🡪 MongoDb. SQLite haricindeki database programları bir server tarafından yönetilmektedir. Server tabanlı programlarda kullanıcının ad/şifre gibi bilgilerini kullanarak internet üzerinden akış gerçekleşir.

Id kolonu primary key olarak adlandırılır.

SQL içerisinde üç farklı ilişki tipi vardır; 1-One to Many, 2-One to One, 3-Many to Many.

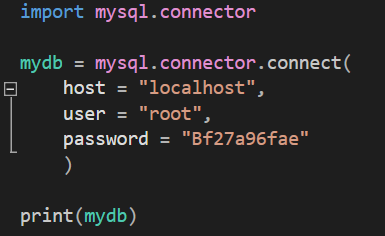
1. Her bir ürün maksimum bir kategoriye sahip olabilir fakat bir kategori sınırsız ürüne sahip olabilir. Ürün için bir kategoriden başka kategori kullanamayız. Bunun için Many to many kullanmamız gerekir.
2. One to one ilişki türünde ise bir ürün yalnızca bir kategoriye sahip olabilir, aynı zamanda o kategori de yalnızca bir ürüne sahip olmalıdır. Buna da ürün olarak bir telefonun kendisine ve kategori olarak da o telefonun spesifik özelliklerine yer verebiliriz (renk, ebat, ağırlık vs).
3. Her bir ürün birden fazla kategoriye sahip olabilir ve her bir kategori birden fazla ürüne sahip olabilir. Örneğin rastgele ikili bir tabloda bir cep telefonun hem elektronik hem de telefon kategorisinde olması gibi aynı zamanda elektronik kategorisi ürün olarak hem telefonu hem de bilgisayarı kapsar.

Şu ana kadar yapılan database derslerini özetlemek gerekirse bir excel dosyası üzerine oluşturulmuş farklı tabloların birbiri ile ilişkilendirilmesi üzerinde durulmuştur. Bu ilişkilendirmeler id atama yolu ile yapılır, örnek vermek gerekirse bir lise database’inde öğrenciler ve öğretmenler olsun. Bu öğretmenlerin sınıf öğretmeni olduğu sınıf ile bir öğrencinin sınıfı id haline getirilerek ortak olarak alınabilir, bu şekilde tüm verileri ayrı tabloda id haline getirerek ana tabloda göstermek veri açısından tasarruf sağlamakla birlikte diğer bilgilerle (isim gibi) ilişkilendirmeyi kolaylaştırır.

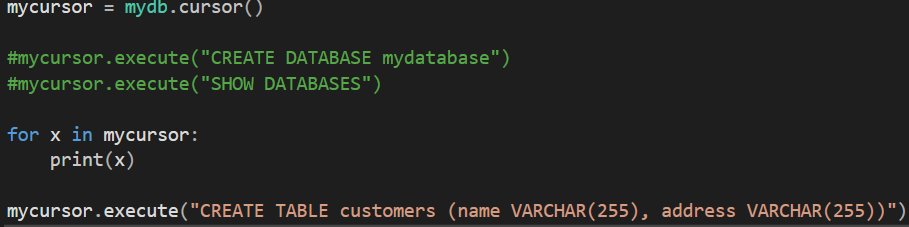


MySQL’de yukarıdaki şema yardımı ile table oluşturulabilir. Schematic oluşturmak için “SCHEMAS” olarak gözüken yere sağ tıklayarak şema oluşturuyoruz.

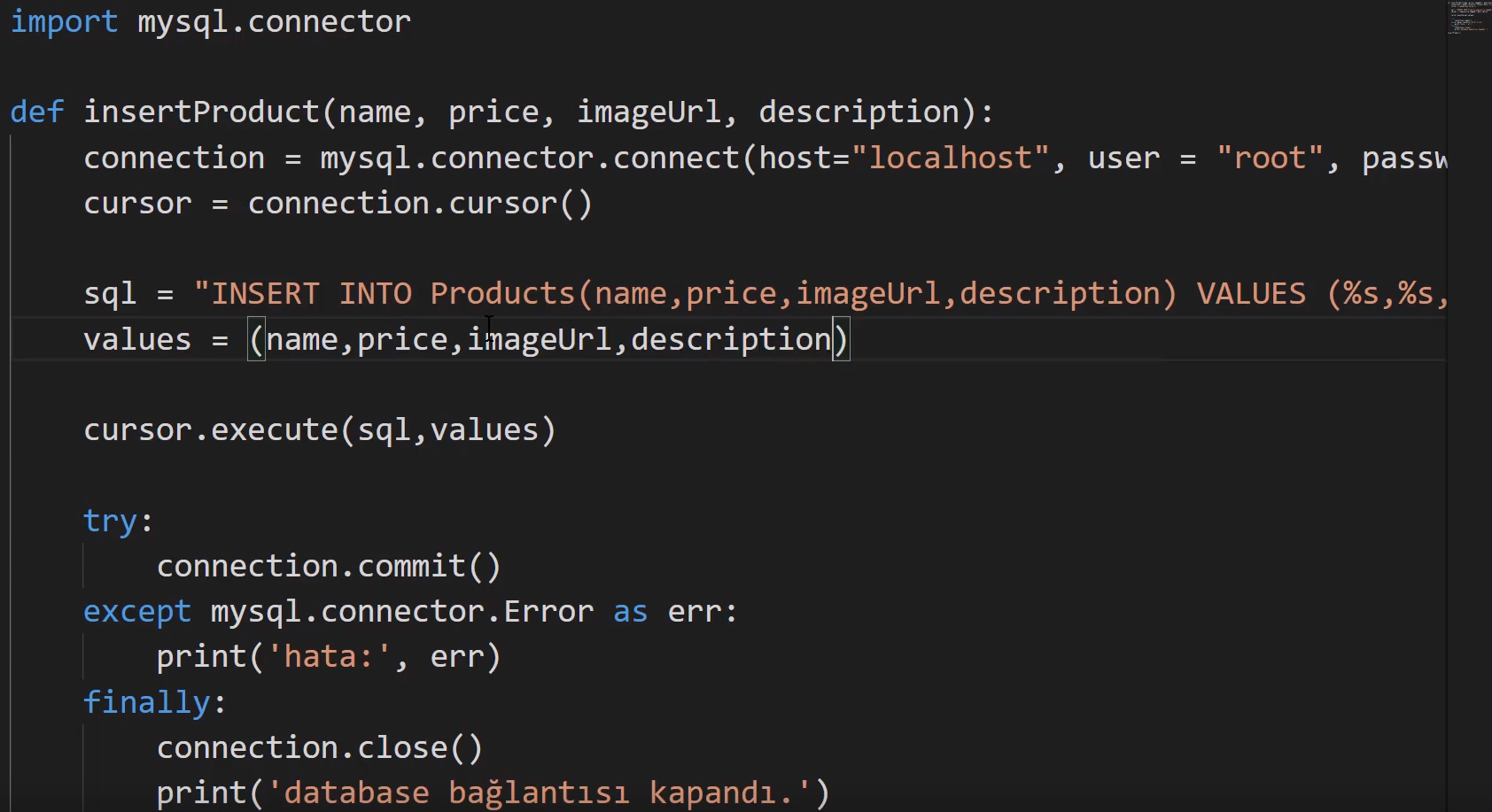
Tables sekmesinden ürettiğimiz tabloya sağ tık ile “select row” dediğimizde karşımıza grid çıkacak, o grid ile tablo oluşturmamız mümkün, tablo oluşturuyorken öncesinde tabloyu belirlediğimiz kriterlere göre oluşturmamız gerekir aksi takdirde uygulama kısmında sistem error verir.



Örnekte olduğu gibi user ve password bilgilerini “connect()” komutuna girerek mySQL server’ındaki hesabımıza ulaşabiliriz. IDE üzerinden database oluşturma işlemleri yapıldığı gibi direkt python üzerinden de yapılabilir. “database = “something”” olarak database ismi connect içerisinde belirtilir.



“.cursor()” komutunu connectimize entegre ettikten sonra “.execute()” komutuyla manuel olarak mySQL’e komut gönderebiliriz, örnekte olduğu gibi “CREATE DATABASE”, “SHOW DATABASES” komutlarıyla database oluşturabilir veya olanları görüntüleyebiliriz. Üçüncü execute komutuyla ise database içerisinde customers adlı table oluşturup name ve addres parametrelerini “VARCHAR” şeklinde atadık.

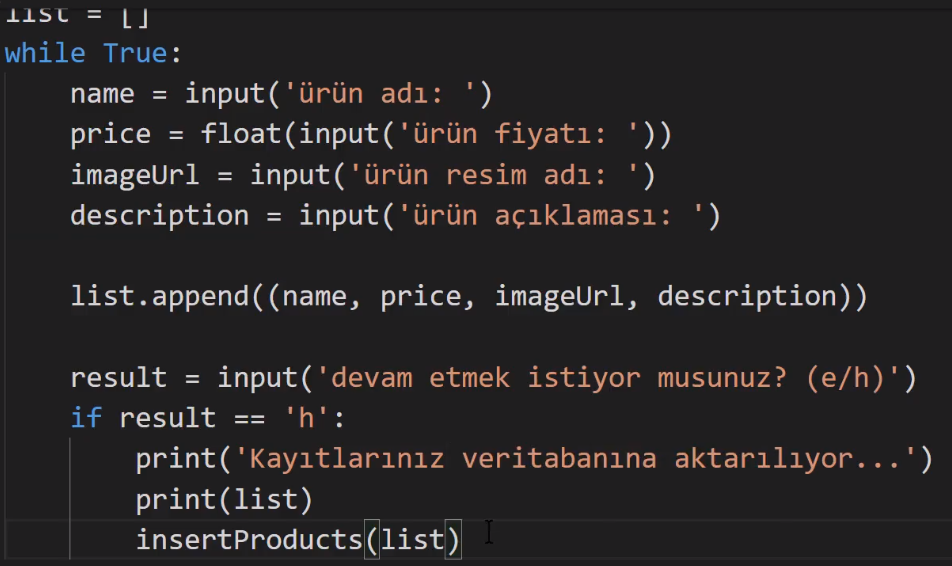


mySQL yapısını import ettikten sonra bir fonksiyon oluşturarak connect ile gerekli SQL hesabımızı koda bağladık, bunun ardından cursor komutunu bir değişkene entegre etmemizin ardından sql ve values adında iki değişkene üzerinde uğraşacağımız table’ın kolonlarını ve içerisine girelecek değeri yerleştirdik, sonrasında bu değişimi cursor’ımıza entegre ettik. Son olarak hata oluşması halinde bir try except fonksiyonu kurarak hata olması halinde bize gösterecek bir kod yazdık ve finally komutuyla sistemi kapatarak bize kapanma mesajı printleyecek bir kod meydana gelmiş oldu.

Üstteki paragraf “insertProduct()” fonksiyonunun içi boş biçimde table içi değişkenlerin fonksiyon içerisinde tek sefer tanımlandığı yapıya ait, resimde ise fonksiyona vereceğimiz dıştan değişkenler ile SQL table üzerinde eklemeler yapabiliriz. Bu eklemeleri kod içerisine yazmak yerine input olarak entegre ederek veriyi direkt kullanıcıdan alabiliriz.

“cursor.rowcount” kaç adet sıra olduğunu (örneğimizde kayıt oluyor) table’dan çekebiliriz. “cursor.lastrowid” komutu ile de son eklenen sıranın id numarasını çekebiliriz.

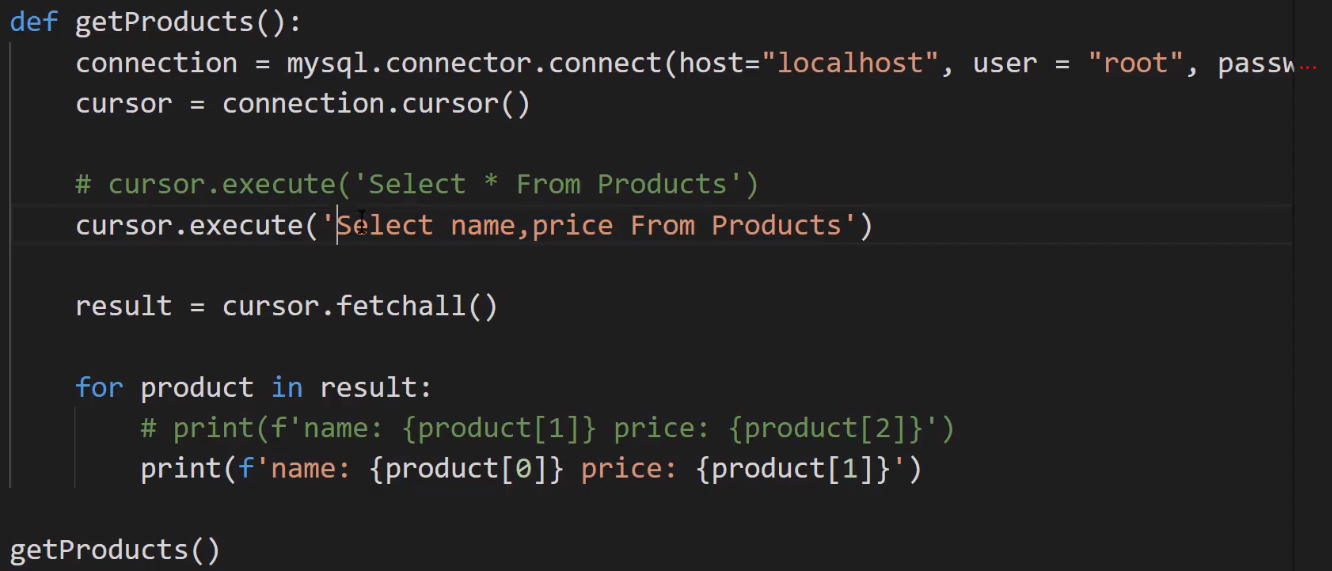
Eğer girdi işlemini tekli değil de çoklu yapmak istersek o durumda “cursor.execure” komutunu “cursor.executemany” komutu ile değiştirmemiz gerekir, ayrıca name vb parametreleri ayrı bir fonksiyonda tanımlarız ve insertProduct(list) olarak fonksiyonumuzu değiştiririz (normalde name, price gibi parametreleri alıyordu).



Yukarıda bahsettiğimiz gibi bir liste oluşturarak “.append” metoduyla liste içerisine input değerlerini gönderdik, result’ın sorusuna “h” denmesi halinde girdiler kullanıcıya yansıtıldı ve SQL içerisine eklendi, bunun yerine “.append” komutunu kullanmayarak direkt list aşamasında SQL’e ekleme yapabiliriz fakat bu durumda girdiler sürekli olarak SQL’e giriş/çıkış yapacağından işlem fazlalığının önüne geçmek adına listeye ekleme metodu bu anlamda oldukça işlevli oluyor.

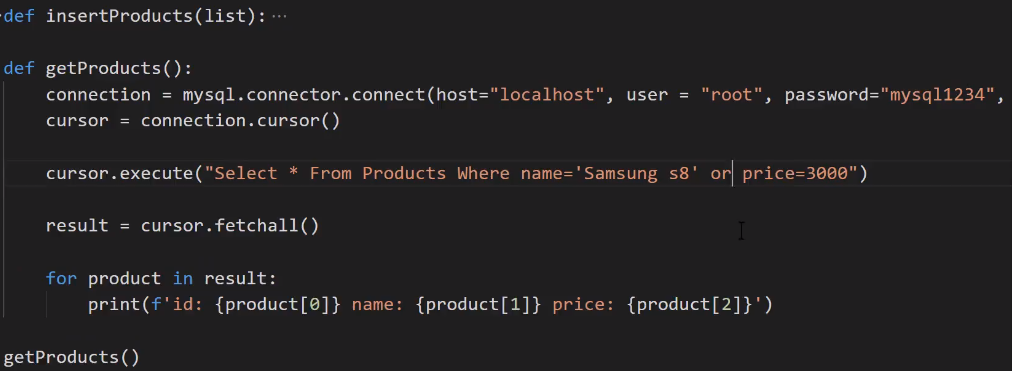
“import datetime” komutu SQL ile sıkça kullanılır, bu komut sayesinde zaman ile ilgili komutlara izin vermiş oluruz. Örneğin database’imize bir kullanıcının doğum bilgisini girecekken “datetime.datetime(year, month, day)” komutunu kullanarak tarihi sisteme başarılı bir şekilde aktarmış oluruz.

*Nesne tabanlı programlama ile mySQL’e veri aktarımı 17. Dakikadan sonrasını pek anlayamadım videonun geneline bak* 🡪 [**Video**](https://www.btkakademi.gov.tr/portal/course/player/deliver/s-f-rdan-ileri-seviye-python-programlama-5877), [**Kod**](https://github.com/metinoguzhan/python_bastan_sona_sadik_turan/blob/master/21_Python_MySql_SqLite/Uygulama_mysql-demo-students.py)

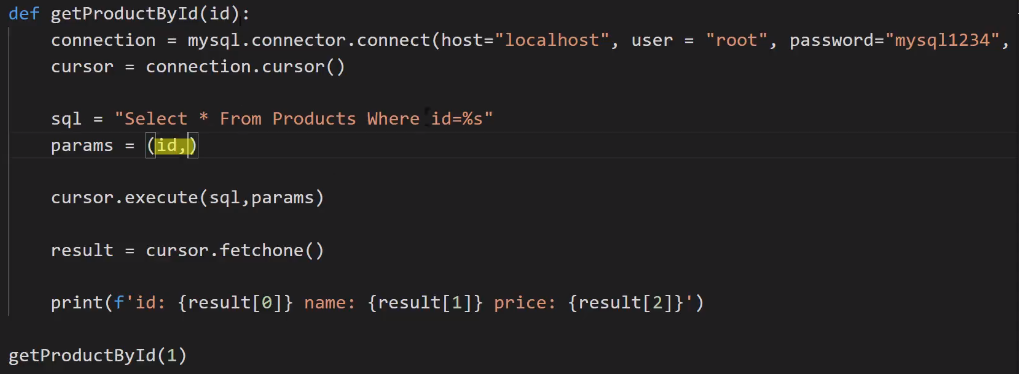


Önceki metotlarla database’e işlemiş olduğumuz verileri kod içerisine çekmek istersek yukarıdaki gibi bir fonksiyon oluşturabiliriz. Öncelikle getProducts() fonksiyonu oluşturarak connection değişkeni ile mySQL’e giriş yaptık. Bunun ardından cursor’ümüzü connection değişkenine entegre ettikten sonra çekmek istediğimiz table’ı belirledik (products), burada “\*” komutu tüm verileri çek anlamına gelmektedir. Fakat bu şekilde database’den bütün değerleri çekmiş olduk, bu durum performansı olumsuz etkileyeceğinden kodumuz için yalnızca “name, price” parametrelerini çekmek daha iyi olacaktır. “cursor.fetchall()” komutu ile execute içerisinde belirlediğimiz “name, price” parametrelerini getirdik ve for döngüsü içerisinde name ve price parametrelerini yazdırdık. “cursor.fetchone()” komutu ile yalnızca koşula uyan ilk satırın verilerini çekebiliriz, “fetchall” ile arasındaki fark diğerinin list halinde çağırılmasıdır ve for döngüsüne ihtiyaç duyarız. Bu fonksiyon filtreleme yaptığımızda benzersiz girdilerin elde edilmesi için idealdir.

“execute()” içerisindeki “Products” database’inin ardından aynı string içerisine ekleyeceğimiz “Where” komutu ile databate içindeki bir parametreyi değere eşitleyerek belirli satıra ulaşabiliriz.



“Where” komutu and/or komutları ile de kullanıma uygundur. Bunun yanında örneğin price kısmında “=” yerine “<=” gibi işaretleri de kullanabiliriz.



Örnekteki fonksiyon ile yalnızca id numarası girerek satır verilerini çeken bir fonksiyon oluşturduk, işaretli yedeki id’ye neden “,” koyulduğunu anlayamadım.



Örnekteki gibi “Order By some” komutunu kullanarak (bu some birden fazla parametre olabilir) verilerimize sıralama işlemi yaptırabiliriz. Alfabetik veya sayısal ifadelerde azalana göre sıralama yapmamız da mümkün, parametrenin ardından yazacağımız “DESC” komutu ile (descending) azalana göre verileri sıralayabiliriz (ASC komutu artana göre demektir ve default olarak ayarlıdır. Örnekteki gibi bir düzen kurarsak kodumuz önce “name” parametresine göre alfabetik sıralama yapacak, eğer aynı isme sahip veriler var ise “price” parametresi devreye girerek küçükten büyüğe sıralamayı yapacaktır.



“COUNT()” metodu ile database içerisinde kaç adet satır olduğunu hesaplayabiliriz, “\*” yerine “name” gibi başka parametreler koyduğumuz takdirde bu parametreye sahip satırlar içerisinde işlem yapacaktır.

“AVG()” komutu ile de sayısal değere sahip parametrelerin ortalamasını alabiliriz (örneğin Price).

“SUM()” komutu ile sayısal değerlerin toplam değerine ulaşabiliriz. “MIN()” komutu ile minimum değere, “MAX()” komutu ile maksimum değere sahip veriye ulaşabiliriz.



Yukarıdaki örnekte product table’ındaki fiyatı en yüksek olan ürünün ismini çektik. İfadeyi parantez içerisine zorunlu değil fakat okunabilirliği arttırıyor.

mySQL’de table içerisinde sütun atarken isimlerde boşluk bırakmamaya dikkat et, değişkenlerin teş parçadan oluşması gerekiyor.



Id’si 5’e eşit olan satırdaki “name” değişkenini “Samsung S10” olarak güncelledik. Set’ten sonra gelen içerikler üzerinde değişiklik yapılacak parametrelerdir, where’den sonraki parametreler ise bu değişimin hangi kısımda meydana geleceğini gösterir.



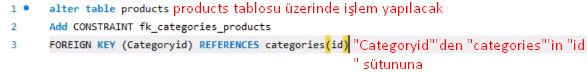
Şeklinde ise parametreleri dışarıda tanımlayıp değerleri dışarıdan (kullanıcıdan vs) çekebiliriz.



Yukarıdaki komut ile where komutu ile belirlediğimiz satırdan veri silme işlemi yapabiliriz.

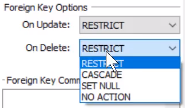


mySQL’de iki farklı tabloyu birbirine bağlamak için öncelikle ayrı bir tablo oluşturulur, sonra var olan tablomuza yeni bir kolon eklenir ve bu kolon ile diğer tabloyu birbirine bağlamak için aşağıdaki metot uygulanır.



Yazılan kodu aktive etmek için kod kısmını seçip çalıştırmamız (yıldırım işareti) gerekmektedir.

Önemli olan bir diğer nokta bağlamak istediğimiz sütunların datatype değerleri aynı olmalıdır. Örneğin bir sütun “INT” ise diğeri de aynı şekilde olması gerekir.



Tables bölümünde ayarlar kısmından bu kısma ulaşabiliriz, bir silme işleminde restrict dersek silmeye izin verilmeyecektir, set null dersek silmek istediğimiz veriler null (0) olur fakat bunun için başlangıçta NN fonksiyonunu kapalı tutmuş olmamız gerekiyor. Cascade ise sütunu silmemizi sağlayan komuttur.



“Select” yapısına “iner join table2 on table2.id=table1.table2id” komutunu ekleyerek (buradaki id’leri biz belirledik kodun kendisinde yok, genel kullanımda böyle yapılır) iki tabloyu birleştirebiliriz.



Table’ların tamamını değil de belirli kısımlarını çekmek için örnekteki “table.name,price” yapısını kullanabiliriz.



“sql” değişkeninin sonuna ekleyeceğimiz where yapısı ile de hangi değerden olanları çekeceğimize karar verebiliriz. Bu yapının aktif olabilmesi için select yapısı ile kullanılmalıdır.

İlk örnekteki “From Products” ve “join Categories” kısmına denk gelen bu iki tabloyu “as char” (char burada belirlediğimiz herhangi bir harf/sözcük) ifadesiyle düzenleyerek tablolarımızı “char” olarak adlandırabiliriz.

İlk örnekteki “inner join” ifadesi iki tablonun kesişiminden meydana gelen bir ifadedir ve “left/right join” olarak da yerine göre kullanılır. Resimlerdeki örnekten gidersek “left join” olması durumunda “Product” tablosundaki değerleri, kategorisi olup olmaması farketmeden alır, “right join” ifadesi ise “Category”’nin bir ürüne karşılık gelip gelmemesine bakmaksızın alınır.

SQLite

Database’in offline olacağı durumlarda SQLite dosyası kullanılabilir. Örnek olarak vermek gerekirse hepsiburada gibi alışveriş sitesinde kullanıcıların hesap bilgileri, ürün bilgileri ve diğer bilgilerin online olarak etkileşimde olduğu bir taban için MySQL tarzı online database’ler gerekirken, offline oynanabilecek bir oyun için veya internete dosya olarak yükleyeceğimiz bir veri tabanı sistemi için SQLite’ı kullanabiliriz. MySQL server tabanlı bir database’ken SQLite uygulama tabanlı bir database’dir.

“import sqlite3” ile koda SQLite’ı entegre edebiliriz.



Veritabanı dosyasına bağlanmak için yukarıdaki komuttaki gibi connect ile dosyanın sonuna “.db” ekleyerek ulaşabiliriz.



Kod yapısı açısından MySQL ile arasında pek bir fark yok, farklı yazılım kullanıyorlar ve veritabanını kendi programları aracılığıyla oluşturma şekilleri farklılık gösteriyor. Bunun haricinde online bir yapıya sahip olmadığını için kullanıcıdan bağlanma adına isim/şifre gibi kavramlara ihtiyaç duymuyor.

Table içerisinde bir tablonun bazı özelliklerini görüntülemek için tabloya sağ tık yapıp “table inspector” dememiz gerekir, bu alandan “foreign key”’e tıkladığımızda referans ettiğimiz tabloların durumunu görüntüleriz.