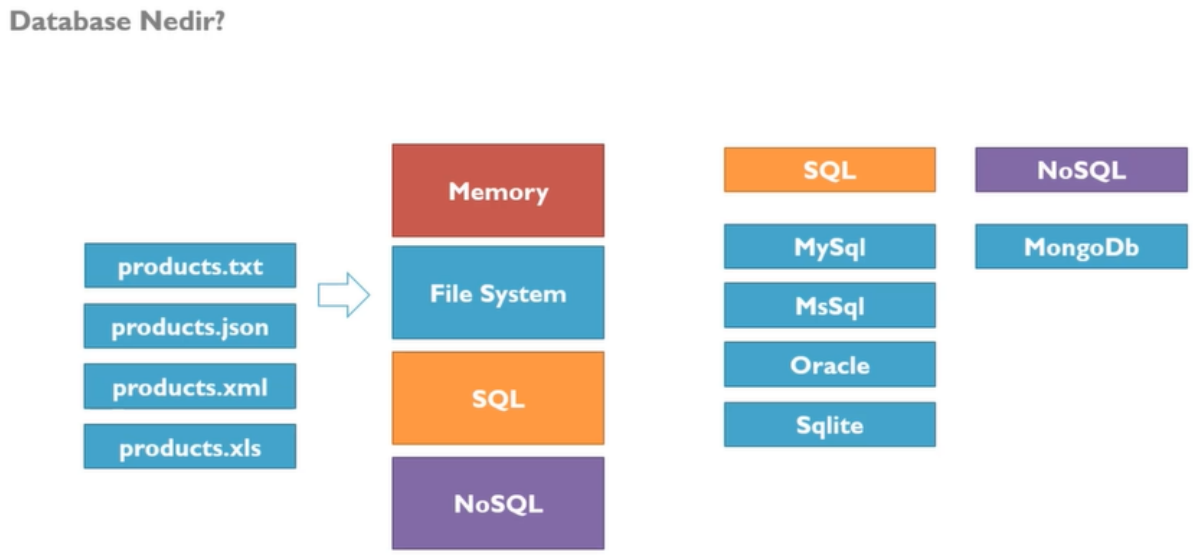
1. **Giriş**

Hazırlanan programlardaki bilgiler ram üzerinde tutulur, dolayısıyla program yeniden çalıştırıldığında bigliler kaybolur.

Bilgilerin kaybolmaması için bunları biryerde saklamamız gerekiyor, bunun için en basit opsiyon, bilgileri .txt,.json,.xml veya .xls gibi formatlarda saklamaktır. Ki küçük çaplı uygulamalar için bu işe yarayacaktır.

Ancak bu yöntemlerin problemi şu, örneğin binlerce ürünün olduğu bir .txt dosyasından tek bir id'ye göre bulmaya çalışıyoruz, işte bunu yapmak için tüm .txt file'ının yüklenmesi gerekecek, scale arttıkça bu yöntem pratik bir yöntem olmaktan çıkıyor.



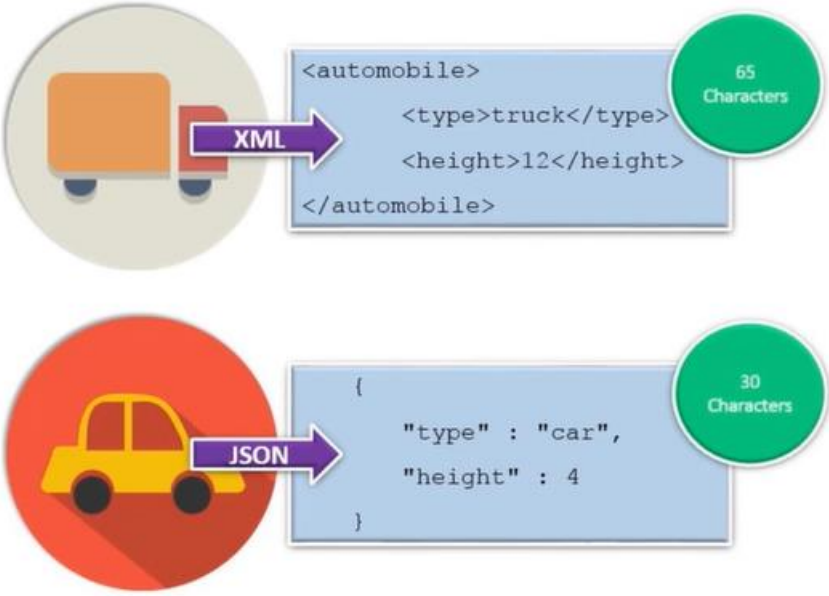
Veri tabanı kullandığımız zaman işte bu problemi çözmüş oluyoruz, kaydedilen ürünlere SQL ile query yollayıp sadece istediğimiz nitelikteki ürünleri sayfamıza çağırabiliriz.

Veri tabanı olarak SQL ve NoSQL veritabanı şekilleri mevcut, SQL için yukarıdaki görüldüğü gibi farklı servisler mevcut, bunların hepsinin kullanımı birbirine çok benziyor.

SQL veritabanı altındaki veritablarına ilişkisel veritabanları da denir.

Bunun dışında bir de NoSQL veritabanından bahsedilebilir. SQL ve NoSQL veritabanlarının farklarını anlamak için JSON veya XML'i anlayalım:

- Diyelim ki bir bir araç bilgisini xml veya json dosyası olarak saklamak istiyoruz.
- Bunu aşağıdaki gibi yaparız:



İşte bu tipte saklanan verileri, NoSQL database yapısında saklarız.

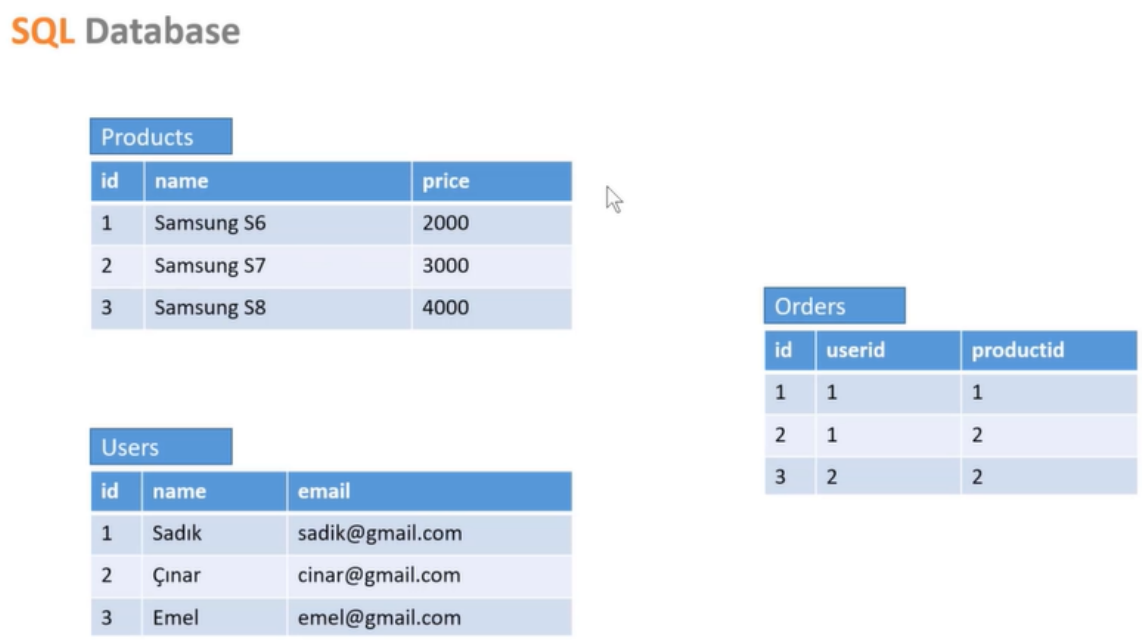
Buna karşılık SQL veritabanı içerisinde tutulan veriler ise bir excel tablosuna benzer.

	A	B	C
1	Name	Price	Image
2	Samsung s6	2000	1.jpg
3	Samsung s7	3000	2.jpg
4			

Yani verilerin saklanış biçimine göre SQL veya NoSQL grubu içinde yer alır.

2. SQL Database Yapısı

Kursta database'leri SQL ve NoSQL olarak iki farklı grupta inceleyeceğiz. Bu word dosyası boyunca SQL Database yapısından bahsedelim.

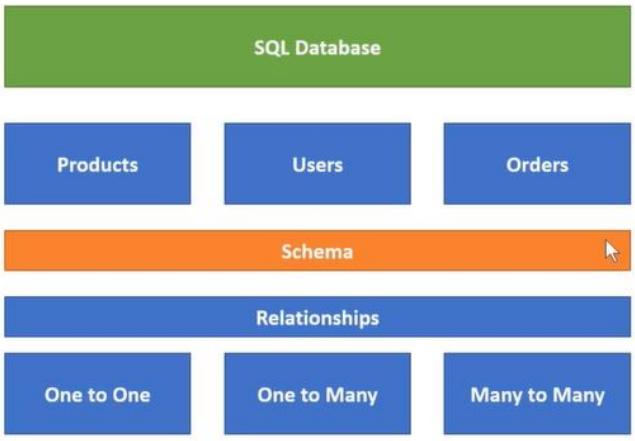


SQL Database için yukarıdaki gibi tablolar oluşturulur.

Yukarıda 3 farklı konu başlığında özelleşmiş 3 farklı tablo var, birtanesi ürünleri tutarken, diğeri kullanıcıları tutuyor ve sonuncusu ise siparişleri tutuyor. Bu şekilde farklı veriler özelleşmiş olarak farklı tablolarda tutulur ve gerektiğinde tablolar arası ilişkiler kurulur.

Burada her tablonun ilk kolonu id kolonudur, her bir girdinin unique olduğunu gösterir ve bu id kolonu primary key olarak adlandırılır.

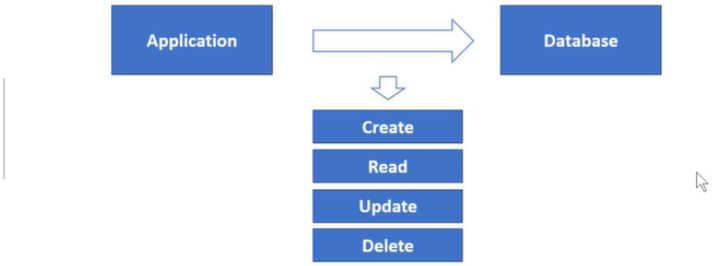
Her grup bilginin farklı tablolarda saklanması bizi gereksiz bilgi tekrarı yapılmasından korur.



Yukarıdaki gibi SQL Database altında 3 adet tablo oluşturuldu, bunların herbirinin hangi bilgileri tutacağı yani hangi kolonlara sahip olacağı önceden hazırlanan şema ile belirlenir.

Tablolar arasında 3 farklı ilişkiden söz edilebilir. One to One, One to Many, Many to Many. Bunlar daha sonra açıklanacak.

SQL Query



Peki diyelim ki bir SQL database'i oluşturuldu biz bu database ile nasıl interact edeceğiz?

- Bunu SQL Queries ile yapacağız.
- Bu şekilde tablolara yeni satırlar ekleyebilir veya onları istediğimiz filtrelerle okuyabiliriz, update ya da delete işlemleri gerçekleştirebiliriz.
- Örneğin bir read sorgusu SQL ile aşağıdaki gibi yapılabilir:

```
Select * from Products Where Price > 100
```

- Products tablosunun tüm sütunlarını seç, bunların arasından price'I 100'den büyük olan satırların hepsini getir.

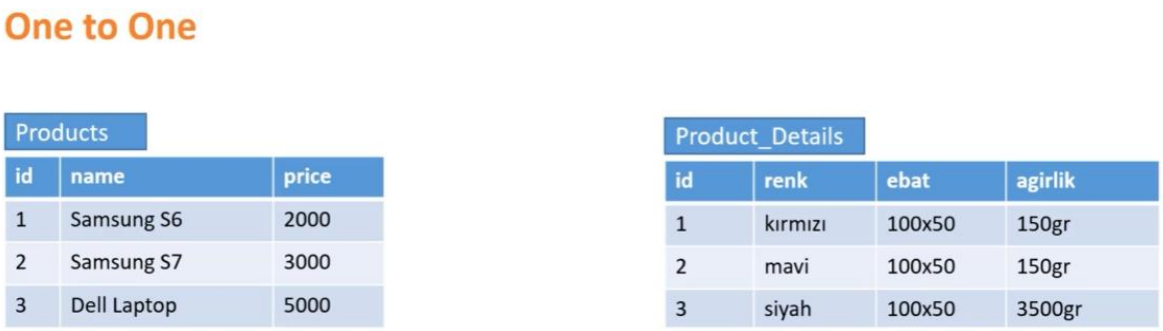
3. SQL Tablo İlişki Çeşitlerini Anlayalım

Önceki kısımda söylendiği gibi 3 farklı tablo ilişkisinden söz edilebilir:

1. One to One, 2. One to Many 3. Many to Many.

Birincisiyle başlayıp bu ilişkileri daha iyi anlamaya çalışalım.

One to One



Bu en kolay olan ilişki aynı tablo dikeylemesine kesilmişse aralarındaki ilişki one to one. İlk tablodaki her bir girdiye karşılık ikinci tabloda tek bir girdi var ve vice versa!

One to Many

İkinci olarak one to many ilişkisini anlayalım.

Aşağıdaki iki tablo arasında bir ilişki yoktur, çünkü ilk tablodaki ürünleri category tablosuna bağlayan bir bilgi yoktur!

Bu iki tabloyu ilişkilendirmenin bir yolu ilk tabloya bir category sütunu eklemektir.

Products

id	name	price	category
1	Samsung S6	2000	Telefon
2	Samsung S7	3000	Telefon
3	Dell Laptop	5000	Bilgisayar

Category

id	name
1	Telefon
2	Bilgisayar
3	Tablet

Ancak bu pek iyi bir practice değildir, diyelim ki telefon kategorisinin ismini değiştirmek istedim, diğer tablodan tek tek isimler değiştirilmeli falan verimsiz.

Bunun yerine tabloya bir categoryId sütunu eklemek daha mantıklı:

Products

id	name	price	catid
1	Samsung S6	2000	1
2	Samsung S7	3000	1
3	Dell Laptop	5000	2

Category

id	name
1	Telefon
2	Bilgisayar
3	Tablet

Şimdi daha mantıklı. İşte bu tablolar arasındaki ililki One to Many'dir.

İlk tablodaki bir girdiye yani bir ürüne karşılık ikinci tabloda tek bir girdi yani kategori var. Yani bir ürünün tek bir kategorisi var!

Buna karşın bir kategoriye karşılık bir çok ürün var.

Yani bir ürünün tek bir kategorisi olabilirken, bir kategori birden fazla ürüne atanabilir.

Şimdilik şöyle formülüze edelim, diyelim ki iki tabloyu da genişleteceğim ve iki tablo da tüm bilgileri içerecek, yani products tablosunda kategoriler de yazacak buna karşılık, category tablosunda da o category'e sahip tüm ürünler yazılacak.

İlk tabloda her girdiye karşılık ikinci tablodan tek bir girdi kullanılır. Buna karşılık ikinci tablo için tek bir girdiye karşılık ilk tablodan birden fazla girdi yazılmalıdır.

Bir ürünün yalnız bir kategorisi olabilirken, bir kategorinin birden fazla ürünü olabilir!

Many to Many

Ancak eğer ki yukarıdaki örnekte her bir ürün birden fazla kategoriye sahip olabilseydi işte bu durumda tablolar arasında many to many ilişkisinden söz edilecekti.

Mesela kategoriler telefon, bilgisayar ve elektronik olsaydı o halde bir ürün hem telefon hem de elektronik kategorisine sahip olabilirdi.

Bu durumda 2 tablo yerine 3 tablo kullanmak daha verimli olacaktır. Bir ürünün kategori sütununa birden fazla değer girmek istemiyorum, o halde diğer opsiyon her aynı ürünü sahip olduğu her kategori için tekrar etmek bu da gereksiz yere bir çok elemanın tekrarlanması anlamına geliyor onun için 3 tablo kullanarak olayı şöyle çözebiliriz:

Products

id	name	price
1	Samsung S6	2000
2	Samsung S7	3000
3	Dell Laptop	5000

Category

id	name
1	Telefon
2	Bilgisayar
3	Elektronik

ProductCategory

productid	categoryid
1	1
1	3
2	1

Göründüğü gibi bir ürün'ün birden fazla kategorisi olabilir. Buna karşın bir kategorinin de birden fazla ürünü olabilir. O halde many to many ilişkisi var.

Product satırlarını tekrarlamamak için 3. bir tabloda ilişkiler tutulmuş.

4. Uygulama: E-ticaret Veritabanı Tasarımı

Şimdi excel üzerinde bir veritabanı tasarım örneği yapacağız. Burada amacımız veritabanlarının nasıl tasarlandığını genel hatlarıyla kavramak.

Aşağıdaki bilgilerin tutulduğu bir tablo isteyebiliriz:

Id	Ürün Adı	Kategori	Müşteri	Sipariş Tarihi
1	Samsung S6	Telefon	Ahmet	1.01.2019
2	Samsung S7	Telefon	Ali	1.01.2020
3	Lenovo Laptop	Bilgisayar	Zeynep	1.11.2020
4	Samsung S6	Telefon	Hasan	1.01.2020

Eğer uygulamamızda az sayıda ürün kullanacaksak böyle bir tablo da kullanılabilir, örneğin tüm kategorileri bir div içinde kullanıcıya göstermek istersem yukarıdaki tablodaki tüm kategorileri çekerim, tekrarlayanları elimine ederim ve kullanıcıya gösterebilirim.

Ancak eğer onbinlerce ürünle başa çıkmaya çalışıyorsak yukarıda kurulan tablo biraz verimsiz oluyor, bir çok gereksiz bilgi tekrarı yapılmasının yanında sürekli string değerler ile atama yapıldığı için örneğin samsung s6 ürününün sonuna garantili yazısını eklemek istesek tek tek tüm samsung s6 değerlerini değiştirmemiz gerekecek bunun yerine bu atamaları foreign keys kullanarak yapmak daha verimli ve sade olacaktır.

Siparişler

Id	Ürün Id	Kategori	Müşteri	Sipariş Tarihi
1	1	Telefon	Ahmet	1.01.2019
2	2	Telefon	Ali	1.01.2020
3	3	Bilgisayar	Zeynep	1.11.2020
4	1	Telefon	Hasan	1.01.2020

Ürünler

Id	Ürün adı
1	Samsung S5
2	Samsung S7
3	Lenovo Laptop

Yukarıdaki artık database'e yeni bir tablo ekleyerek, siparişler tablosunda doğrudan ürün adları kullanmak yerine foreign key kullandık. Artık Samsung S6 ürününün ismini değiştirmek istersek bunu tek bir cell'i değiştirerek yapabiliyoruz.

Şimdi ürünler tablosuna bir iki kolon daha ekleyelim, ayrıca siparişler tablosundaki kategori isimlerini temsil etmek için de bir foreign key kullanalım, bunun için de yeni bir kategori tablosu oluşturalım:

Siparişler

Id	Ürün Id	Kategori Id	Müşteri	Sipariş Tarihi
1	1	1	Ahmet	1.01.2019
2	2	1	Ali	1.01.2020
3	3	2	Zeynep	1.11.2020
4	1	1	Hasan	1.01.2020

Ürünler

Id	Ürün adı	Price	Açıklama
1	Samsung S5	2000	idare eder
2	Samsung S7	3000	iyi
3	Lenovo Laptop	6000	iyi

Kategori

Id	Kategori Ad
1	Telefon
2	Bilgisayar

Benzer şekilde bir müşteri tablosunu da database'e ekleyebiliriz, böylece siparişler tablosunda foreign keys kullanılarak sipariş bilgileri başka tablolara bağlantılarla tutulmuş oluyor.

Siparişler

Id	Ürün Id	Kategori Id	Müşteri	Sipariş Tarihi
1	1	1	1	1.01.2019
2	2	1	2	1.01.2020
3	3	2	3	1.11.2020
4	1	1	1	1.01.2020

Müşteri

Id	Müşteri Adı
1	Ahmet
2	Ali
3	Zeynep

Ürünler

Id	Ürün adı	Price	Açıklama
1	Samsung S5	2000	idare eder
2	Samsung S7	3000	iyi
3	Lenovo Laptop	6000	iyi

Kategori

Id	Kategori Ad
1	Telefonlar
2	Bilgisayar

Bu yapıyla farklı başlıklar farklı tablolarda tutuluyor ve bize gerektiğinde istediğimiz tablodan istediğimiz veriyi çekebiliyoruz ve bu tablolar arasındaki ilişkileri kullanarak daha detaylı verileri de elde edebiliyoruz.

Örneğin kategori tablosu tamamen kategori bilgisi tutarken, müşteri tablosunda sadece müşleriler var, ürünler tablosunda ürün adı, fiyatı, açıklaması gibi bilgiler tutuluyor, buna karşın siparişler tablosu ise aslında diğer tabloların foreign keyler ile birbirine bağlanmış hali gibi, elbette her bir unique id bir siparişe ait yani siparişi tutuyor.

Burada baktığın zaman siparişler tablosunda ürün id verildiği zaman aslında kategori id'yi vermeye gerek olmamalı, kategori zaten ürünle ilgili bir şey o halde kategori id kısmını ürünler tablosuna taşıyabiliriz.

Siparişler

Id	Ürün Id	Müşteri	Sipariş Tarihi
1	1	1	1.01.2019
2	2	2	1.01.2020
3	3	3	1.11.2020
4	1	1	1.01.2020

Müşteri

Id	Müşteri Adı
1	Ahmet
2	Ali
3	Zeynep

Ürünler

Id	Ürün adı	Price	Açıklama	Kategori Id
1	Samsung S5	2000	idare eder	1
2	Samsung S7	3000	iyi	1
3	Lenovo Laptop	6000	iyi	2

Kategori

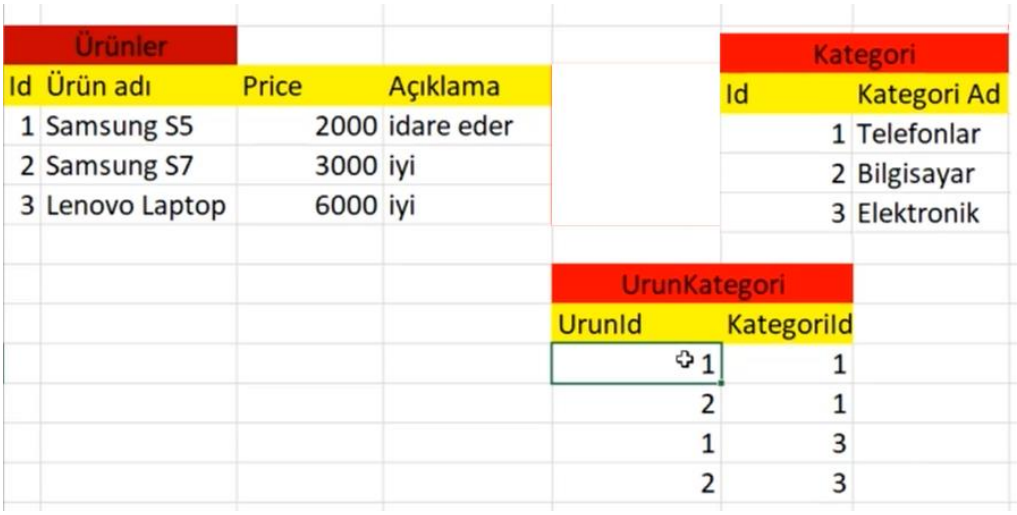
Id	Kategori Ad
1	Telefonlar
2	Bilgisayar

Burada foreign key'lerin yaptığı iş iki tablo arasında ilişki kurmak! Bu ilişkinin olabilecek tiplerinden geçen kısımda bahsettik.

Örneğin ürünler listesindeki foreign key: kategori id kısmı ve böylece iki tablo arasında bir ilişki kurulmuş oluyor, buradaki ilişki de one to many ilişkisi. Çünkü her bir ürünün tek bir kategorisi olabilirken bir kategorinin birden fazla ürünü olabilir.

Önemli bir detay hakkında tablo oluşturulacak başlıkların CATEGORICAL VARIABLE olduğuna dikkat et, gidip price veya tarih için veya açıklama için ayrı bir tablo oluşturmak verimsiz olacak çünkü bunlar kategorical variable değiller. Bu yüzden gidip price tablosu oluşturursak sınırsız id olacak, çünkü price'ın sonu yok. He eğer benim ürünlerim 10 farklı price'dan oluşuyorsa belki ayrı bir price tablosu mantıklı olabilir böylece tek bir price'ı değiştirince diğer yerlerde de değiştirmiş olurum.

Eğer ürünler ile kategori tabloları arasında many to many bir ilişki olsaydı, yani bir ürüne birden fazla kategori atayabiliyor olmak isteseydik bunun için 3 tabloya ihtiyaç duyacaktık. Ürünler tablosundan kategori id foreign key'i silinecekti, ve yeni bir tabloda 2 foreign key'in ilişkisi verilecekti:



Ürünler			
Id	Ürün adı	Price	Açıklama
1	Samsung S5	2000	idare eder
2	Samsung S7	3000	iyi
3	Lenovo Laptop	6000	iyi

Kategori	
Id	Kategori Ad
1	Telefonlar
2	Bilgisayar
3	Elektronik

UrunKategori	
UrunId	KategoriId
1	1
2	1
1	3
2	3

5. Uygulama: Okul Veritabanı Tasarımı

Önce kendim uğraşacağım daha sonra kursla karşılaştıracağım.

İsterler:

** Öğrenci bilgileri.
** Öğretmen bilgileri
** Her öğrencinin tek şubesi.
** Öğretmenlerin branş bilgileri.
** Her öğretmenin tek rehberlik sınıfı olsun.
** Her öğretmen birden fazla sınıfa derse girebilir.
** Her öğretmen farklı derslere girebilir.

Kendi yaptığım taslak:

ÖĞRENCİ

Id	İsim	Soyad	Numara	Şube
1	Ali	Öz	123	1
2	Veli	Kayın	124	2
3	Cemre	Derin	125	3
4	Fatma	Çalış	126	1
5	Kerim	Renk	127	3

ÖĞRETMEN

Id	İsim	Soyad	Branş	Rehberlik Sınıfı
1	Hasan	Tahsin	6	2
2	Yüksel	Örnek	2	3
3	Figen	Deynek	3	1
4	Kadriye	Zühre	4	4
5	Cingöz	Recai	1	5

ŞUBE

Id	Şube
1	10A
2	10B
3	10C
4	10D
5	10E

BRANŞ

Id	Şube
1	Tarih
2	Coğrafya
3	Matematik
4	Edebiyat
5	Kimya
6	Beden
7	Resim

ÖĞRETMEN-ŞUBE

Id	ÖğrId	ŞubeId
1	1	1
2	1	3
3	2	1
4	2	5
5	3	5
6	3	2
7	4	3
8	4	4
9	5	2
10	5	4

- Her öğrencinin tek şubesi var.
- Her öğretmenin tek rehberlik sınıfı var.
- Öğretmen birden fazla şubeye derse girebiliyor. (Öğr-Şube tablosu)
- Her öğretmenin birden farklı derslere girebiliyor olmasını ben belirtmedim ancak aynı mantık olacak, 3. Bir tablo ile bunu sağlayabiliriz (Öğr-Branş) veya aşağıdaki gibi tek tabloda hem şube-öğretmen hem de öğretmen-branş bilgisi verilir

Şube-Öğretmen

ŞubeId	ÖğretmenId	DersId	
1	1	1	
1	1	2	
1	2	2	
2	1	1	

6. Uygulama: Sinema Veritabanı Tasarımı

Kriterler:

** Film bilgileri.
** Kategori bilgileri
** Yönetmen bilgileri
** Oyuncu bilgileri
** Her film bir ya da birden fazla kategoride olabilir.
** Her filmin bir yönetmeni olsun.
** Her oyuncu bir ya da birden fazla filmde oynayabilir.

Kendi taslağım:

FİLM

Id	İsim	YönetmenId	Tarih
1	Matrix	3	1999
2	Babam	2	2003
3	Old town	3	2007
4	Back	1	1993
5	Pulp	2	1995

KATEGORİ

Id	Kategori
1	Korku
2	Gerilim
3	Romantik
4	Komedi
5	Gizem
6	Bilim Kurgu

YÖNETMEN

Id	İsim	Yaş
1	Steph	35
2	Joels	40
3	Karen	36

OYUNCU

Id	İsim	Yaş
1	Gared	25
2	Leo	35
3	Margot	30
4	Chris	33
5	Dan	45

Oyuncu-Film

Id	OyuncuId	FilmId
1	1	1
2	1	5
3	2	1
4	3	4
5	4	2
6	4	3
7	5	1

Film-Kategori

Id	FilmId	KategoriId
1	1	1
2	1	2
3	2	1
4	3	6
5	3	3
6	4	1
7	5	3

Burada her filmin bir yöntemeni olsun dendiğinde problem yok, direkt film tablosuna yönetmen id'yi ekledik. Burada one to many ilişkisi var.

Ne zaman ki her film birden fazla kategoride olabilir dedi o zaman film tablosuna kategori column'u ekleyemeyeceğimi anlarım çünkü her filme birden fazla kategori düşebilir. E her kategoriye de birden fazla film düşeceği için burada many to many ilişkisi söz konusu iki tabloyu bağlamak için yeni bir tablo oluşturuyoruz mesaja Film-Kategori tablosu gibi.

Benzer şekilde her oyuncu birden fazla filmde oynayabilirse e bir filmde birden fazla oyuncunun oynayabileceği de zaten bariz, demekki many to many ilişkisi var o halde ekstra bir Oyuncu-Film tablosu kullanılıyor.

Burada ekstra olan şu Many to Many ilişkileri için yaratılan tablolarda genelde id kullanılmıyormuş çünkü zaten her girdi unique olacak bu yüzden mesela OyuncuId de FilmId de primary key olarak tanımlanacak.

7. **Veri Tabanı Sunucusu Nedir?**

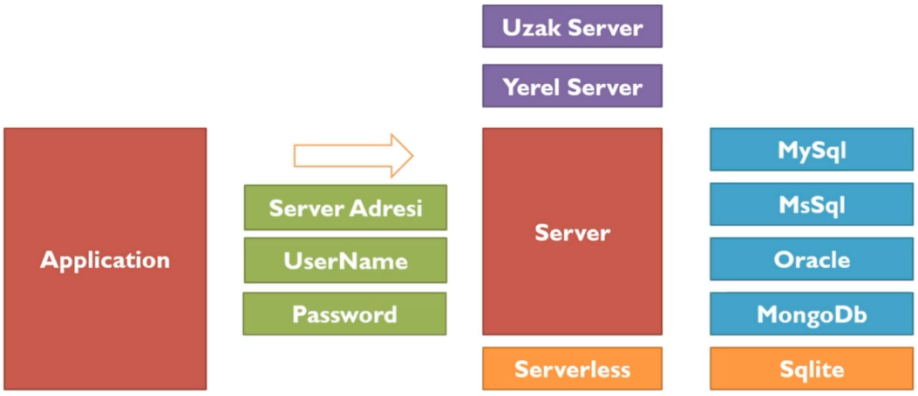
Diyelim ki bir uygulama yaptık, mobil veya web, masaüstü farketmez. Sonuçta verilerimizi bir database'de saklama ihtiyacı duyuyoruz.

Bu amaçla SQL veya NoSQL veritabanı türlerinden birini seçeceğiz. Yani aşağıda en sağda görülen MySql, MsSql, …, Sqlite gibi türlerinden birini seçeceğiz.

Aşağıda mavi olanlar server tabanlı database türleri iken turuncu olan Sqlite server tabanlı olmayan bir database türüdür.

Server tabanlı olanlarda veri tabloları bir sunucuda tutuluyor ve uygulamamızdan bu sunucuya ulaşıp sorgular gönderebiliyoruz.

Buna karşın serverless olan Sqlite database için bir server'a gerek yoktur. Burada tablolar aynı bir excel dosyası gibi yerelde tutulur ancak biz yine SQL ile bu database'e sorgular gönderebiliriz ve böylece verimli bir şekilde database ile bilgi alışverişi yapılabilir. Zira verileri excel dosyasında kaydediyor olsaydık, tek bir datayı çekmek istesek bile önce tüm veriyi uygulamamıza almamız gerekecekti.



Server gereken database'ler için deploy aşamasında uzak serverler kullanılır ancak geliştirme aşamasında veya şuanda bizim bilgisayarımızı localhost olarak kullanabiliriz. Yani hazırlanan database hem yerel server'da hem de uzak bir serverda çalışabilir, uygulamamız çalışmaya başladığında bizim bir hosting kiralayıp database'i buraya gömmemiz gerekecek.

Uzak serverlardaki database'lere erişmek için server adresi, username ve parola kullanılır, yerelde buna gerek kalmayacak.

Peki hangi durumda server database'leri kullanılır hangi durumlarda serverless database'ler kullanılır.

Örneğin bir mobil uygulama yaptık, kullanıcının kendisiyle ilgili olan verileri Sqlite database'inde saklayabiliriz, dediğimiz gibi bu aynı verileri proje içerisinde bir excel dosyası içinde tutmak gibi olacak sadece uygulama veriye yine sql dili ile verimli bir şekilde ulaşabilecek.

Server tabanlı database ise kullanıcıların hepsinin aynı bilgiye ulaşması gerektiği durumlarda kullanılır. Örneğin ürünleri server database'inde tutarız, ürünlerde bir değişiklik olduğunda tüm kullanıcılar bu database'I kullandığı için değişiklikten haberdar olur.

8. **Veri Tabanı Sunucu Kurulumu – MySQL**

Şimdi server tabanlı database software'lerinden biri olan MySQL'I kullanacağız.

Bu database software'i kullanabilmek için bunun servisini bilgisayarımıza kurmalıyız. Bilgisayarımıza bir yerel server kuracağız, çünkü MySQL server tabanlı bir database management system.

Daha sonra uygulama yaparken, yerel serverda yapılan işlemlerin aynısı kiralanan uzak serverda da yapılabilir ve böylece uygulamamızı çalışır hale getiririz.

Öncelikle bilgisayarımızı MySql server kurcağız. Bu server kurulduktan sonra server'a ulaşıp server üzerinde bir veritabanı oluşturmak, bu veritabanı üzerinde bir kayıt sorgulamak, veya yeni kayıt eklemek, güncellemek, silmek için biz bir IDE lazım.

Her server için oluşturulan bir IDE var, MySQL server'I için Workbench isminde bir IDE kullanılacak, bunu aynı bir editör gibi düşünfatkun. Databaser server'ın altında hizmet veriyor ve biz bu hizmet ile iletişimimizi IDE üzerinden yaparak database ile interact edebiliyoruz.

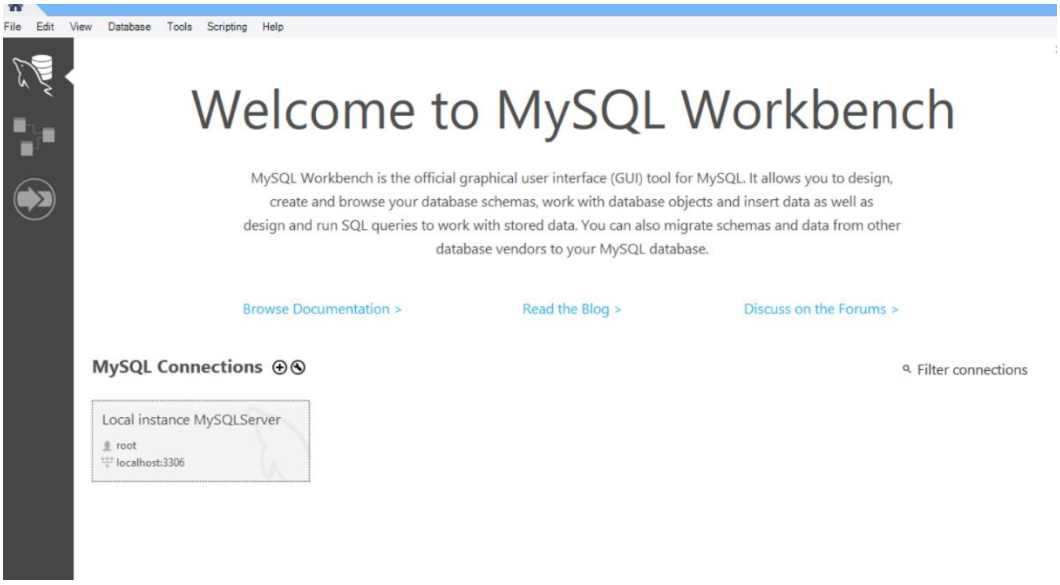
SQL veritabanlarının hepsinde kullanılan dil Structured Query Langugae yani SQL dir. Dolayısıyla hangi database software'in kullanıldığı çok önemli değil.

Aşağıdaki linkten MySQL Server'ını ve Workbench ide'sinin installer'ını indirebiliriz.

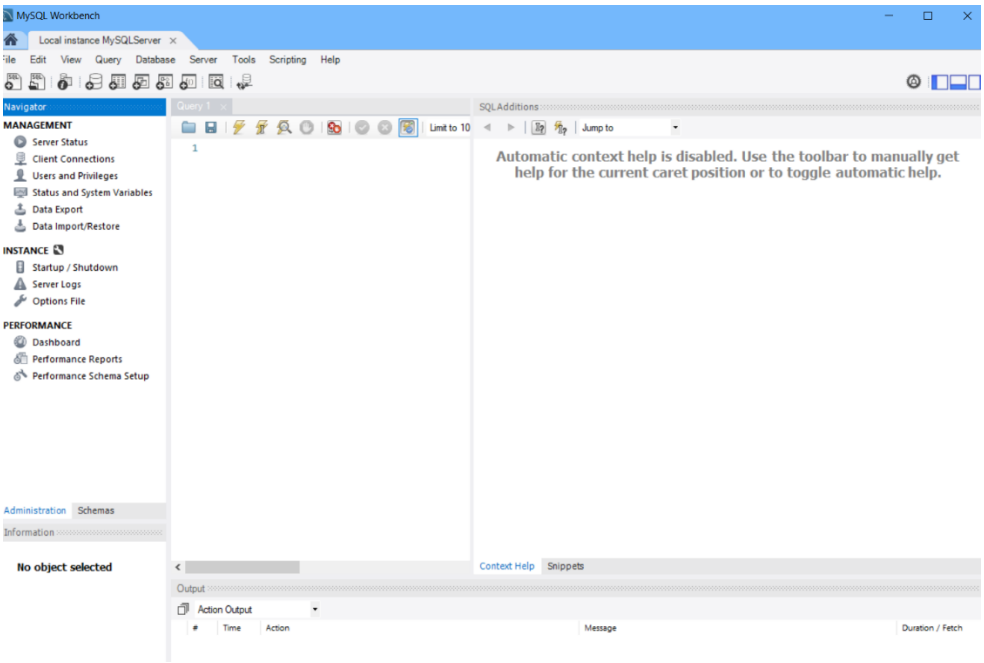
https://dev.mysql.com/downloads/file/?id=497105

Bunun yanında kurulmda Server ve Workbench dışında farklı connector'lar da yüklenebilir connector'ların işlevi Workbench ide'sini devre dışı bırakıp uygulamamız içinden MySQl Server'ına bağlanmak, bu amaçla farklı diller için farklı connector'lar install edilebilir.

Kurulum yaparken server ismini ve parolamızı belirledik. Sonuçta serverimiz ve workbench idemiz kuruldu.

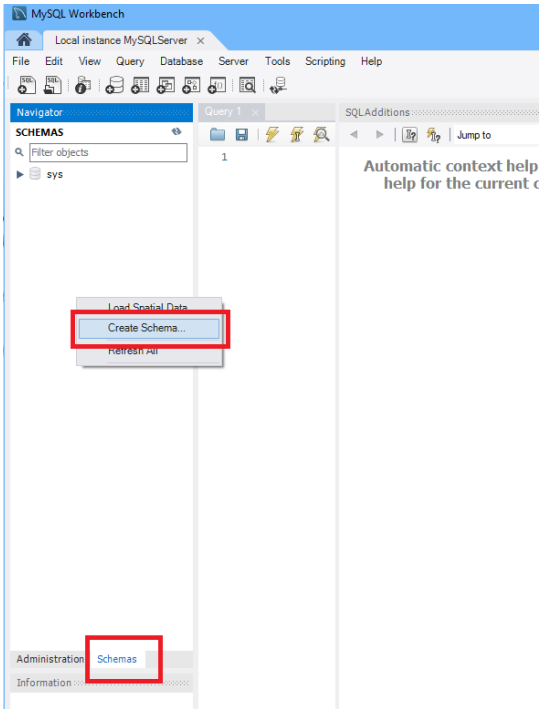


Sol alttaki Local Instance'a tıklayıp local MySQL serverine WorkBench'I kullanarak bağlanabiliyoruz, servere ulaşmak için kullanıcı adımız root şifremiz de daha önce belirlediğimiz şifre.

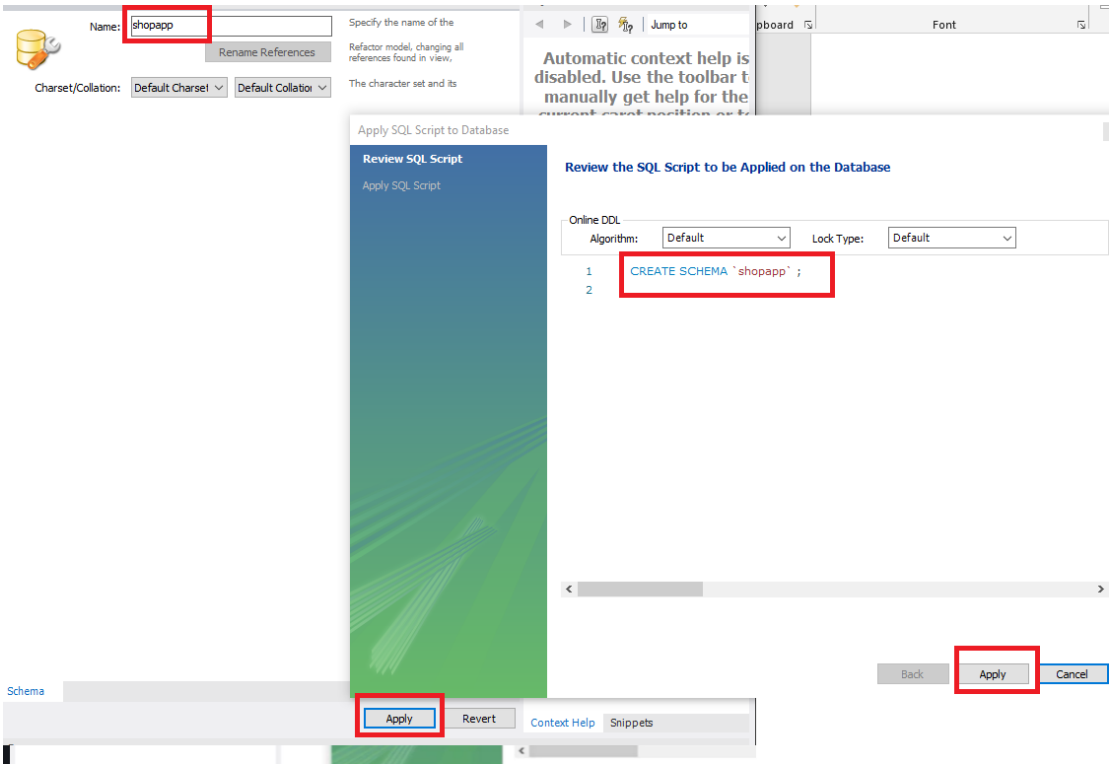


9. Veri Tabanına Tablo Ekleme

Workbench'de aşağıdan schemas sekmesi seçiliyken schemas kısmına sağ tılıyorum ve create schema diyorum.



Buradan shopapp isminde bir schema yani database oluşturuyoruz:



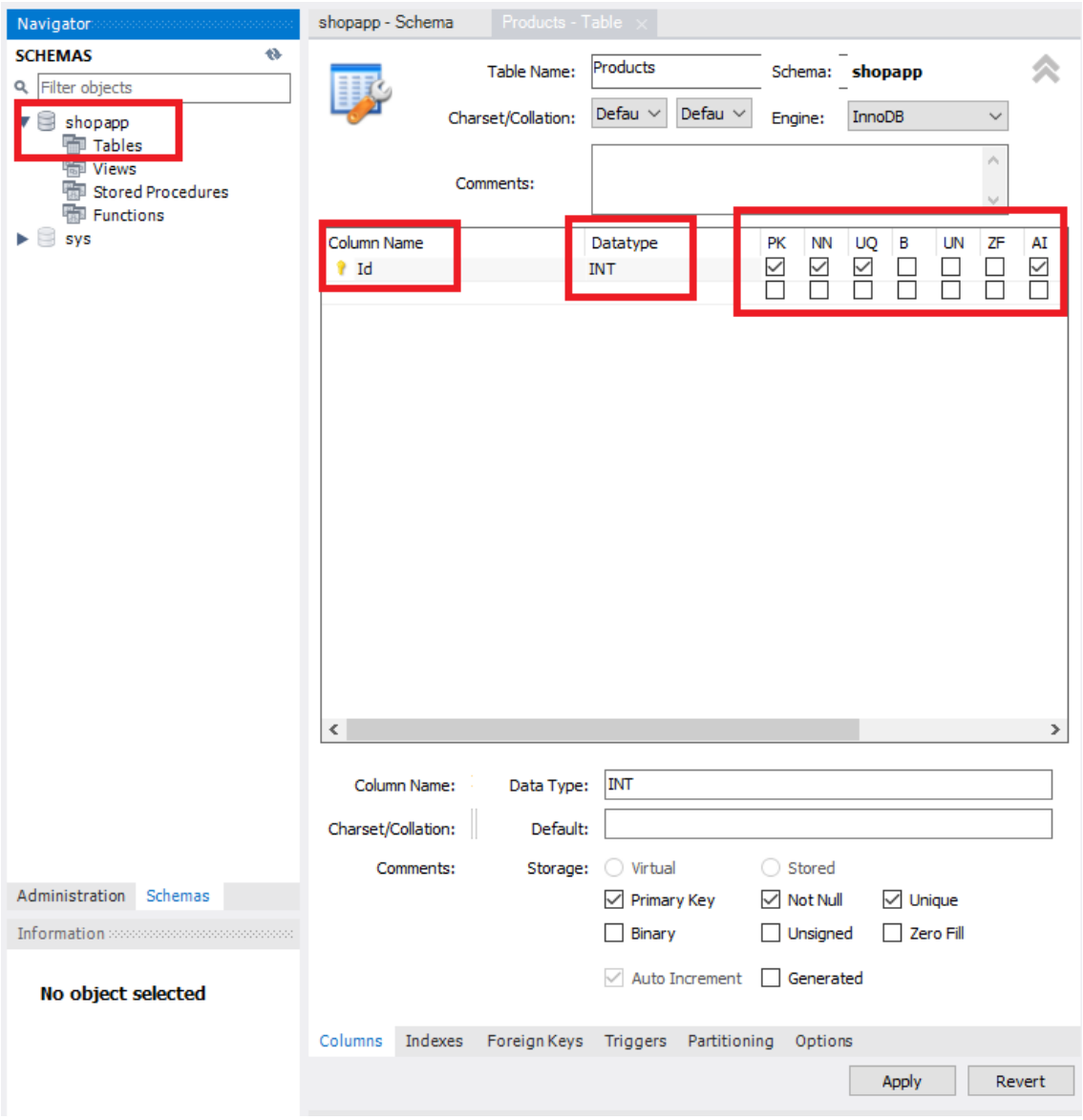
Apply dediğimizde açılan ekranda, ide üzerinden GUI ile yaptığımız işlemin servera gönderilecek hangi koda karşılık geldiğini görüyoruz.

Yani servere **CREATE SCHEMA 'shopapp';** komutu gönderilecek ve böylece shopapp database'i oluşturulacak.

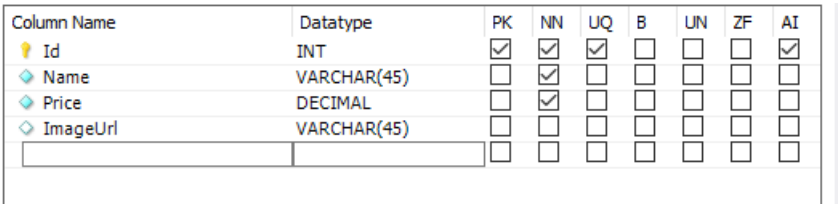
Shopapp database'i server üzerinde oluşturulduktan sonra aşağıdaki gibi solda görünecektir. Burada shopapp database'ine bir tablo eklemek için altındaki tables'a sağ tıklarız ve create new table deriz.

Aşağıdaki gibi Table Name'I girdikten sonra, tabloya Column Name gireriz ayrıca column'un datatype'ını dropdown menu'den seçebiliriz.

Bunun yanında column'u primary key olarak belirledik, not null yani boş geçilemez dedik, bu columndaki her değer unique olacak dedik, auto increment ile de yeni eklenen değeri otomatik olarak ata dedik.

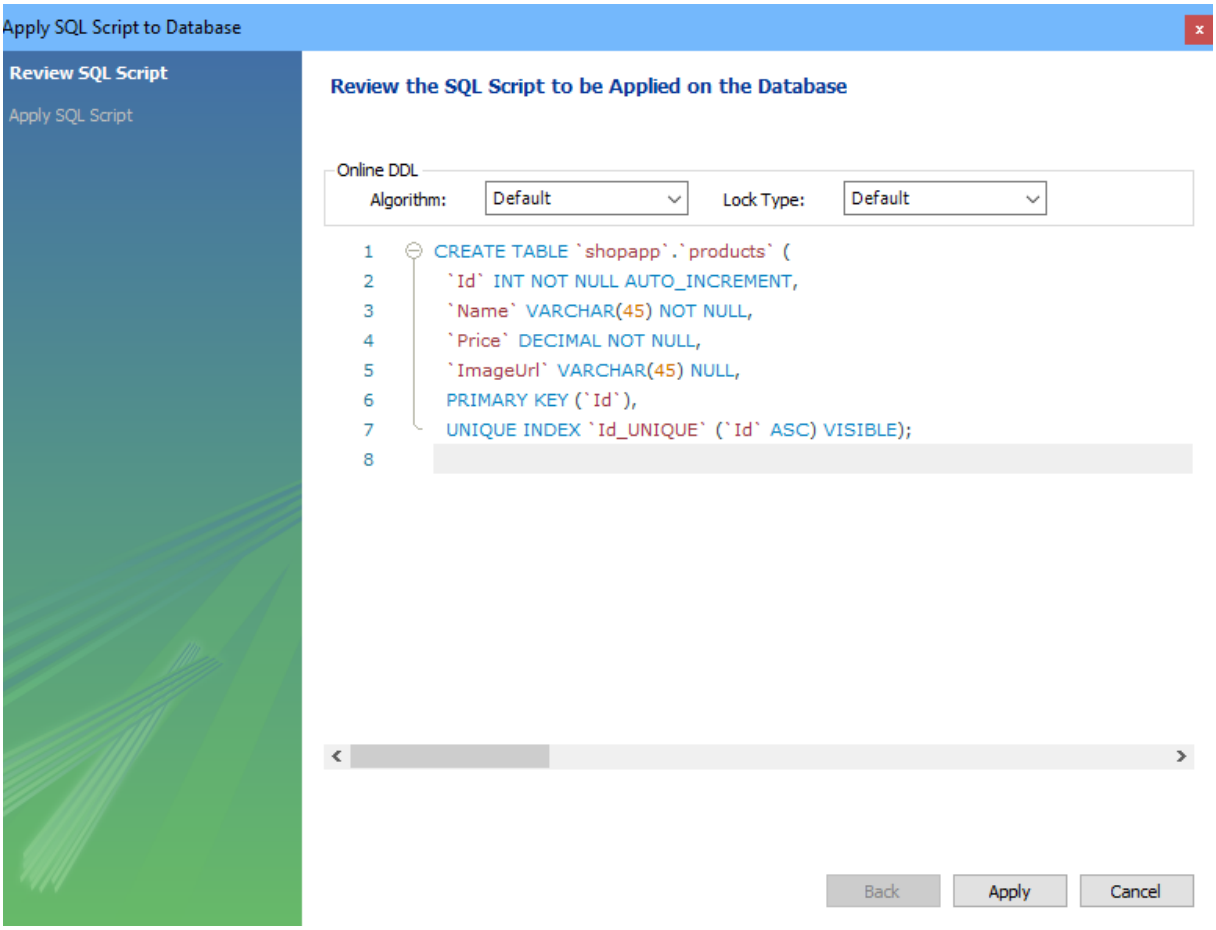


Elbette Products tablosuna Id'nin yanında Name, Price, ImageUrl gibi başka column'lar da ekleyebiliriz.



Yukarıda görüldüğü gibi bu column'larıın datatype'larını belirtiyoruz ayrıca primary key olup olmadığını, boş geçilip geçilemediğini unique olup olmadığını vesaire de biz belirtiyoruz. Mesela ImageUrl boş geçilebilir dedik.

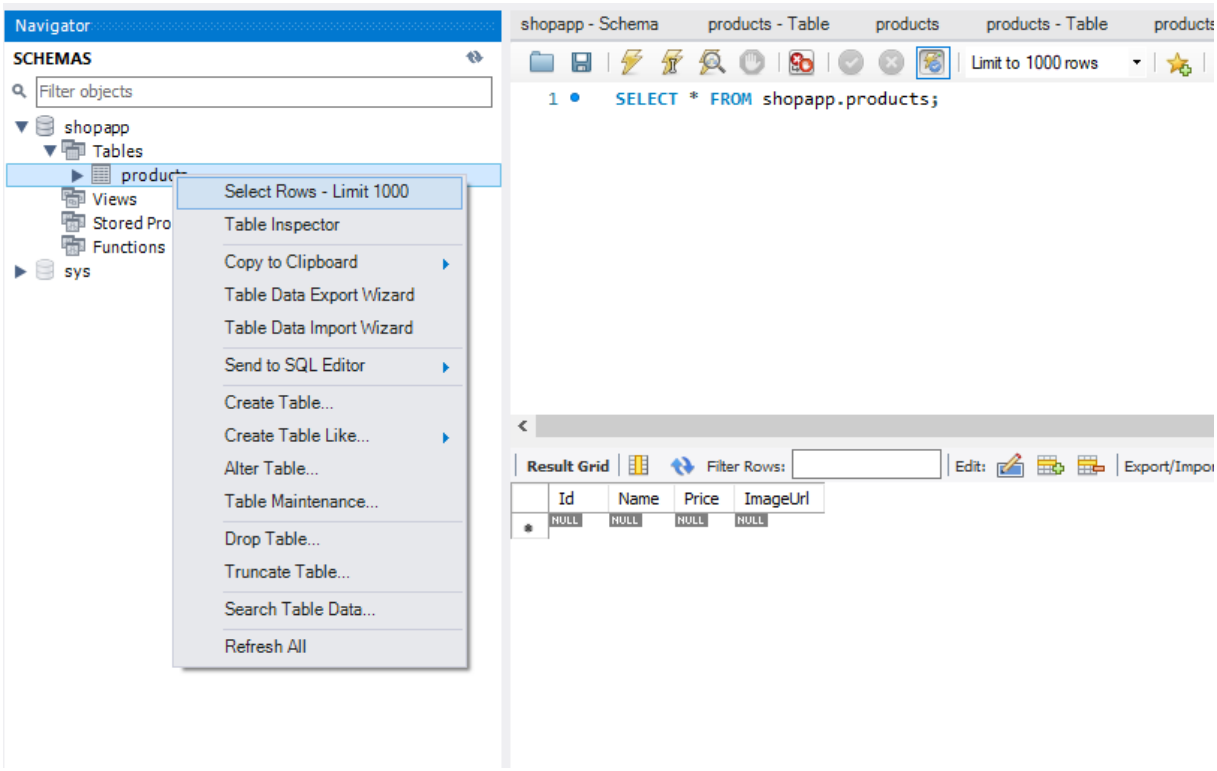
Bu tabloyu eklemek için apply dediğimizde karşımıza aşağıdaki gibi bir window daha geliyor:



Yani anlıyoruz ki bizim deminki GUI üzerinden elimizde doldurduğumuz işleri IDE'miz koda çevirdi ve onaylarsak server'a yollayacak böylece shopapp database'inde bir products table'I istediğimiz özelliklerle yaratılacak.

Bu sorguyu biz IDE GUI'si ile yapmak yerine kendimiz elle de yazabilirdik yine tabloyu yaratacaktık.

Porudct'a sağ tıklayıp Select Rows dersem tabloyu görebilirim, sanırım bunun yaptığı da sağdaki sorguyu oluşturmak, yani tüm satırları göster demek.



Bunun yanında tablonun ayarlarını değiştirmek istersek tablonun yanındaki anahtar işaretine tıklayabiliriz böylece aşağıdaki ekrandan değişiklik yapabiliriz.