5. BAĞLANTI TÜRLERİ

Birlikte Düşünelim

Bilişim sistemlerinde her olgu bir ID niteliğiyle ifade edilmek zorunda mıdır? Birincil anahtar olarak seçilebilecek daha iyi bir nitelik bulunamaz mı?

İki veri tabanı tablosu hiçbir ilişki ile birbirine bağlı değilse bu yapı için hala veri bütünlüğünden bahsedilebilir mi?

Birden fazla ilişkisel veri tabanı, çapraz bir bağlantı ile anahtarları kullanarak veri bütünlüğünü sağlayabilirler mi?

Bir veri tabanı tablosu birden fazla birincil ve/veya ikincil anahtar bulundurabilir mi?

Hangi ilişki tipi daha önemlidir? Tablolar arasındaki mi yoksa nitelikler arasındaki mi?

Başlamadan Önce

Bir önceki bölümde ilişkisel veri tabanlarının iki tür ilişki barındırdığını vurgulamıştık. Bununla birlikte önceki bölüm, nitelikler arasındaki ilişkilerin belirlenmesi üzerineydi. Bu bölümde ise tablolar arasındaki ilişkilere yoğunlaşacağız.

Ayrık iki konu gibi gözükse de aslında birbirini tamamlayan iki süreçten bahsediyoruz. Önceki bölümde; eldeki bir ham veri kümesinin sahip olduğu nitelikleri, aralarındaki fonksiyonel bağımlılıklara göre gruplayarak tabloları elde ediyoruz. Şimdi de tablolar arasındaki ilişkileri tanımlayarak veri kümesi biçimindeki veri yapımızı veri tabanı biçimine dönüştürmüş oluyoruz. Böylece verimizi, veri bütünlüğünü kaybetmeden küçük parçalara bölerek ilişkisel veri tabanları sayesinde elde etmek istediğimiz avantajları yakalamış oluyoruz.

Bu bölümde öncelikle bağlantı türlerinin neler olduğunu inceleyeceğiz. İlişkisel veri tabanı tasarımında 3 çeşit ilişki türü bulunmaktadır. Bu ilişki türleri, iki veri tabanı tablosunun bağlanabileceği her türlü kombinasyonu içermektedir. Ayrıca bölüm içerisinde birincil anahtar seçimine yönelik çeşitli yaklaşımları da ele alacağız. Böylece ihtiyaca yönelik olarak hangi birincil anahtarın kullanılabileceğini belirleme becerisi kazandırılmış olacaktır.

İki veri tabanı tablosu, aralarında çoğa çok ilişki türü ile bağlantı sağladıysa bu durumda yeni bir varlık oluşturulması gerekebilir. Pivot ya da bağlantı tablosu adı verilen bu tablonun nasıl oluşturulacağıyla ilgili bilgileri de sağlayarak bu bölümde anlatılacakları tamamlayacağız.

5.1. Bağlantı Türleri

İlişkisel veri tabanlarının içerdiği ikinci tür ilişki olan; tablolar arasındaki ilişkinin sağlanması konusunu ele almaya başlıyoruz. Bu bölüm, bir önceki bölümle farklı gibi gözükse de birlikte ele alınması gerekir. Bir ilişkisel veri tabanı tasarlamadan önce elimizde en iyi ihtimalle örnek verinin ham veri kümesi hali bulunmaktadır. En kötü ihtimalle ise sadece gerçekleştirilecek yapının bir tarifi bulunur. Her durumda, eldeki veri kümesinin yapısı ya da ihtiyaçlar listesi gözetilerek hangi niteliklerin ya da veri alanlarının kullanılması gerektiği veri tabanı tasarımcısı tarafından belirlenebilir (Hoffer, Ramesh & Topi, 2016; Akadal, 2021).

Tabloların oluşturulması aşaması, fonksiyonel bağımlılıkların belirlenmesi aşamasında yaptığımız gibi hangi niteliklerin birbirleriyle ilişkili olduklarını belirleyerek ve birbirlerini temsil etme durumlarını irdeleyerek onları gruplandırmak üzerine kuruludur. Bu aşamada, ilk başta elde tüm nitelikleri içeren tek bir grup varken; ilişkili niteliklerden oluşan çeşitli gruplar elde edilmiş olmaktadır. Bu sayede tasarlamaya çalıştığımız veri

tabanı yapısı için tablolar elde edilmiş olmaktadır. Daha önce vurguladığımız üzere; veri tabanları, büyük veri kümelerini küçük tablolar halinde sunmakta ve bu sayede çeşitli avantajlar sağlamaktadır.

Görece büyük bir veri kümesinin küçük tablolar haline dönüştürülmesiyle birlikte veri yapısı ayrık ve bağımsız olacağı için veri bütünlüğünün kaybedilmesi söz konusudur. Örneği kişiler ve bu kişilerin yaptıkları satın alımları düşünelim. Kişi ve satın alımla ilgili iki tablo elde etmemiz ve bu iki tablonun bağımsız olması durumunda, hangi satın alımın kim tarafından gerçekleştirildiği bilgisini de kaybetmiş olacağız. İlişkisel veri tabanlarının veri bütünlüğünü kaybetmeden bu süreci gerçekleştirebildiğini biliyoruz. Bu da demek oluyor ki veri kümesi tablolara bölündükten sonra veri bütünlüğünü sağlamaya devam etmenin bir yolu mevcut. Bu da tabloların birbirleriyle ilişkiler üzerinden bağlanmasıdır.

Veri tabanı tabloları, belirlenen nitelikler üzerinden en uygun ilişki türü seçilerek bağlanırlar. Alt başlıklarda bu bağlantı türlerinin neler olduğu, hangi durumda hangi bağlantı türünün seçilmesi gerektiği konusuna değineceğiz.

Tablolar arasındaki ilişkiler, tablolarda yer alan nitelikler üzerinden sağlanırlar. Bu nitelikler de tablolar için birincil ve ikincil anahtar özelliği kazanırlar. Birincil ve ikincil anahtarları ayrıntılı olarak ele alacağız. Birbirleriyle bağlanmış iki tablo, uygulama anında iki tablo içerisinde yer alan birer satır kaydın aynı anda aynı satırda birleştirilmesi ihtiyacıyla gerçekleştirilmektedir. Örneklerle açıklayalım.

Ogrenci ve Ders tablolarını ele alalım. Bu iki tablo, hangi öğrencinin hangi dersi aldığının belirlendiği bir veri kümesinin ilişkisel veri tabanına dönüştürülmesiyle elde edilmiş olsun. Dolayısıyla eldeki kayıt gerçekte her bir satırda bir öğrencinin bir dersi alıyor olması durumudur. Veri tabanını elde etme süreci içerisinde biz bu veri kümesini Ogrenci ve Ders adında iki tabloya ayırabiliriz. Böylece veri kümesinde yer alan öğrenciyle ilgili nitelikler bir grup, ders ile ilgili nitelikler de başka bir grup oluşturacaktır. Fonksiyonel bağımlılık açısından ele aldığımızda niteliklerin 2 grup oluşturduğu sonucuna varacağız. Sonuç olarak elimizde bir Ogrenci tablosu yani öğrencilerin bir listesi ve bir Ders tablosu yani alınabilecek derslerin bir listesi oluşacaktır. Ancak burada bir veri kaybı var, fark ettiniz mi? Hangi öğrencinin hangi dersi aldığının bilgisini içeren bir veri kümemiz varken artık sadece öğrenci ve dersler listesine sahibiz. Halbuki bizim her bir satırda bir ders alma durumunu ifade eden, öğrenci ve ders kayıtlarının eşleşmesine ihtiyacımız var. Tablolar arasındaki ilişki bizim için bunu sağlamaktadır. Kayıtları, iki farklı tablo içerisinde yer alacak şekilde bölmüş olsak da ihtiyaç halinde veri bütünlüğünü sağlayacak yapıyı kurmamız gerekmektedir. Bu noktada ilişkiler ve anahtarlar bizim aradığımız veri bütünlüğünü sağlamamız konusunda yardımcı olacaktır. İlişkiler, bir tablonun diğer tabloya bağlantı kurma aşamasında o tablo üzerinde kaç kayıt ile eşleşebileceğini gösteren türlere sahiptir. Bu özelliklere göre ilişkileri birbirlerinden ayırmaktayız.

Tablolar arasındaki ilişkiyi belirlerken, bu ilişkileri yönlü kabul ederek inceleme yaparsak ilişki türünü belirlemek çok daha kolay olacaktır. Örneğin A tablosunun B tablosuyla olan bağlantısının özelliklerini incelediğimizde çıkan sonuç ile B tablosunun A tablosuyla olan bağlantısının özelliklerini incelediğimizde çıkan sonuç ilişki türünü iki adımda elde etmemiz sağlayacaktır. Bu durum örneklerle çok daha iyi anlaşılacaktır.

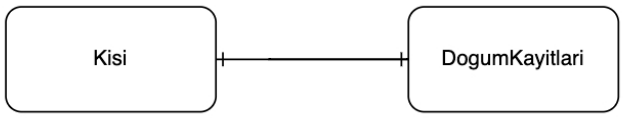
Şimdi sırasıyla bu ilişki türlerini inceleyelim.

5.1.1. Bire Bir (1-1)

İlk inceleyeceğimiz ilişki türü bire bir olarak adlandırılan ilişki türüdür. Ele alınan iki tablo, karşılıklı olarak yalnızca bir kaydı işaret edecek şekilde bir bağlantı sağladıysa bire bir ilişki türü ele alınır. Anlatımı kolaylaştırmak için A ve B tablolarını ele aldığımızı düşünelim. A tablosunda yer alan herhangi bir kayda karşılık olarak B tablosunda yalnızca tek bir kayıt elde ediyorsak ilişkinin A'dan B'ye yönü "tek" özelliğine sahip olacaktır. B tablosu içerisindeki herhangi bir kayıt A tablosunda yalnızca tek bir kayıt ile eşleşiyorsa bu durumda ilişkinin B'den A'ya olan yönü de "tek" özelliğine sahip olur. Bu durumda A ve B tablolarının arasındaki ilişki türünü bire bir olarak belirtebiliriz.

Örnek vererek bu ilişki türünü açıklayalım. Kisi ve DogumKayitlari tabloları arasındaki ilişkiyi ele alalım. İlişki türünü iki yön için de incelemek en faydalı yaklaşım olacaktır. Kisi tablosunda yer alan bir kayıt DogumKayitlari tablosunda yalnızca tek bir kayda denk gelebilecektir. Çünkü bir kişinin ancak tek bir doğum kaydı bulunabilir. Bu durumda Kisi ve DogumKayitlari tabloları arasındaki ilişkinin Kisi tablosundan

DogumKayitlari tablosu yönüne; yani ilişki gösteriminin DogumKayitlari tablosuyla birleştiği noktaya tek işareti koyulmalıdır. Diğer ilişki yönünü de incelersek eğer; DogumKayitlari tablosunda yer alan herhangi bir kayıt, Kisi tablosunda yalnızca tek bir kayıt ile eşleşebilecektir. Çünkü bir doğum kaydı ancak tek bir kişiyle ilgili olabilir. Birden fazla kayıtla eşleşmesi mümkün değildir. Bu durumda ilişkinin diğer yönü yani Kisi tablosuyla birleştiği nokta da tek olarak işaretlenmelidir. İlişkinin iki yönünün de tek türünde belirlenmesi, bu iki tablo arasındaki ilişki türünün bire bir olmasına sebep olmaktadır. Şekildeki özet gösterimi inceleyiniz.

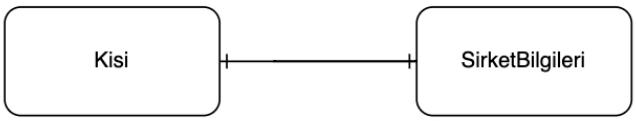


Hatırlayacağınız üzere tablo gösterimlerinde nitelikleri ve anahtarları da ifade ediyorduk ancak şu anda niteliklerle ve anahtarlarla ilgilenmiyoruz. Bu sebeple özet bir gösterime başvurduk. Soldaki kutu Kisi tablosunu, sağdaki kutu ise DogumKayitlari tablosunu ifade etmektedir. Aradaki ilişkinin iki tabloyla birleştiği yerde de tek işaretine sahip olduğuna dikkat etmelisiniz.

Bir şirket veri tabanı içerisinde tüm çalışanlara ait hem kişisel hem de şirketle ilgili bilgilerin kayıt altında olduğunu varsayalım. Bu kayıtlar içerisinde kişinin adı, doğum tarihi ve yeri bilgileri, adresi, vergi kimlik numarası gibi kişisel bilgilerin yanında şirketin hangi bölümünde çalıştığı, ofisinin hangi odada olduğu, hangi pozisyonda görev yaptığı gibi şirket dahilinde geçerli bilgileri de bulunabilir. Tüm bu sayılan özellikler temelde tek bir kişiye ait bilgiler gibi gözükebilir. Ancak biri kişinin kişisel bilgileri, diğerleri ise çalıştığı şirket dahilinde geçerli olan bilgileridir. Bu sebeple bu nitelikler iki gruba ayrılır ve iki tablo ile karşı karşıya kalırız.

Burada tartışmalı bir noktayla karşılaşabiliriz. Sayılan niteliklerin tümü tek bir tablo içerisinde yer alabilir. Doğrudur, bire bir ilişki türü olan tüm tablolar için bu durum geçerlidir. Bu ilişki türüne sahip iki tablo birleştirilerek tek tablo olarak da sunulabilir. Burada tek risk, bazı niteliklerin çok sayıda boş değere sahip olacak olmasıdır. Aynı örnekte şirketin eski çalışanların kayıtlarının da bulunduğunu düşünelim. Kişiyle ilgili kayıtlar varlığını sürdürmeye devam edecektir. Ancak bu kişi şirketten ayrıldığı için şirket içindeki pozisyonu, ofisi ve benzer nitelikler herhangi bir değere sahip olmayacaktır. İki farklı tablo olması durumunda şirket bilgilerinin bulunduğu tabloda eğer kişi hala çalışmıyorsa kayıt eklenmeyebilir. Ancak iki tablonun birleştirilmesi durumunda bunu yapmak mümkün olmayacak, kaydın şirketle ilgili niteliklere denk gelen hücreleri boş değer alacaktır. Bu sebeple bire bir ilişkili tabloları da ikiye ayırarak kullanmak fayda sağlayacaktır.

İlişki türünü araştırmaya geçelim. Kişisel bilgilerin bulunduğu tabloyu Kisi, şirket bilgilerini içeren tabloyu ise SirketBilgileri olarak adlandıralım. Kisi tablosunda yer alan bir kayıt, SirketBilgileri tablosunda ancak tek bir kayda denk gelebilir. Bu sebeple ilişkinin bu yönünde tek işareti kullanılmalıdır. SirketBilgileri tablosunda yer alan herhangi bir kayıt ise yalnızca tek bir kişi kaydıyla eşleşmelidir. Bu durumda ilişkinin bu yönde de tek işaretini alması beklenir. Sonuç olarak bire bir ilişki türünü elde ederiz.



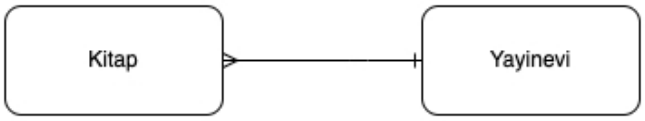
Şekil incelendiğinde bu örnekte de bire bir ilişki türüyle karşılaştığımızı görmüş oluyoruz.

5.1.2. Bire Çok (1-m) / Çoğa Bir (m-1)

İkinci ilişki türümüz bire çok ya da çoğa bir olarak adlandırılan ilişki türüdür. Bu ilişki türü ile şu durumda karşılaşmaktayız: A tablosunda yer alan herhangi bir kayıt, B tablosunda yer alan birden fazla kayıt ile eşleşiyorsa bu ilişki türüne çok adı verilmektedir. Burada dikkat edilmesi gereken, bu durumun her zaman gerçekleşme zorunluluğu olmamasıdır. Yalnızca bir kez bile A tablosundaki bir kayıt B tablosundaki çok kayıtla eşleşse; hatta eşleşmese ancak eşleşme olasılığı olsa çok türünü kullanmak zorundayız. Bir tablodan

diğerine çok, ancak tersi yönde tek türünü belirlememiz halinde bu ilişki türü bire çok ya da çoğa bir adını almaktadır. Bunlardan hangisinin seçileceği ise sadece tabloların dizilişiyle ilgilidir. A tablosundan B tablosu yönüne çok, B tablosundan A tablosu yönüne tek türünde ilişki bire çok; iki tablo sadece yer değiştirdiğinde ilişki türü çoğa bir olarak anılır. Örnek ile devam edelim.

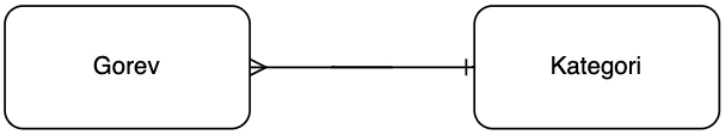
Kitap ve kitabın yayınevine ait bilgilerin yer aldığı iki tabloyu düşünelim. Her bir kitap yalnızca tek bir baskı yapmış olsun. Dolayısıyla bir kitap ancak bir yayınevinden dağıtılabilir. İlişki türünü inceleyelim. Kitap tablosunda yer alan bir kayıt, Yayinevi tablosunda birden fazla kayıtla eşleşebilir mi? Hayır. Bir kitap yalnızca bir yayın evi ile eşleşecektir. Peki Yayinevi tablosunda yer alan bir kayıt, Kitap tablosunda birden fazla kayıt ile eşleşebilir mi? Elbette. Yayınevleri çok sayıda kitaba ev sahipliği yapmaktadır. Dolayısıyla Yayinevi tablosundaki bir kayıt, Kitap tablosundaki çok sayıda kayıtla eşleşme olasılığına sahiptir. Bu incelemeye göre ilişkinin Kitap'tan Yayinevi'ne olan yönü tek, Yayinevi'nden Kitap'a olan yönü ise çok olarak belirlenir. Şemayı inceleyerek devam edelim.



Şema çizilirken önce Kitap, sonrasında Yayinevi tablosunun çizilmesi sebebiyle ilişkiler çok ve tek olarak sıralanmıştır. Bu durumda ilişki türünü çoğa bir olarak adlandırmaktayız. Hiçbir değişiklik yapmadan sadece şemada Kitap ve Yayinevi tablolarının yerlerini değiştirsek ilişki türüne bire çok dememiz gerekecekti. Yani bu iki tür arasındaki fark yalnızca tabloları hangi sırada ele aldığımızla alakalıdır.

Bir örnek daha ele alalım. Yapılacak işleri kaydetmeyle ilgili çeşitli uygulamalar var, gündelik hayatta denk gelmişsinizdir. Daha çok "to-do" adıyla geçen bu uygulama türünde yapılacak işleri unutmamak için maddeler halinde kayıt altına alabiliyoruz. Dilersek bunları kategoriler altında gruplayarak daha düzenli bir yapı elde edebiliyoruz. Bu yapı için elimizde iki veri tabanı tablosu olduğunu düşünelim: Gorev ve Kategori. Gorev tablosu eklenen yapılacak işler maddelerini, Kategori ise görevlerin gruplanacağı kategorileri kayıt altına almak için hazırlanan bir tablo olsun. Bu iki tablo arasındaki ilişkiyi inceleyelim.

Gorev tablosunda her alan herhangi bir kayıt, Kategori tablosunda birden fazla kayıt ile eşleşebilir mi? Diğer bir deyişle bir görev birden fazla kategori ile ilişkili olabilir mi? Aslında burada geliştirilen uygulamanın yapısına göre olabilir de olmayabilir de. Ancak biz klasik yaklaşımla yanıtlarsak bir görev yalnızca bir kategorinin altına eklenebileceğine göre Gorev tablosunda yer alan kayıtlar Kategori tablosunda yalnızca bir kayıt ile eşleşebilecektir. İki tablo arasındaki ilişkinin Kategori tablosuna bağlanan ucunda tek işaretinin yer alması gerekmektedir. Peki Kategori tablosunda yer alan herhangi bir kayıt, Gorev tablosunda birden fazla kayıtla eşleşebilir mi? Evet. Bir kategori altında birden fazla görev yer alabilir, dolayısıyla Kategori tablosunda yer alan bir kayıt, Gorev tablosunda birçok kayıtla eşleşebilir. Bu durumda çoğa bir ilişki türüyle karşı karşıyayız. Şema üzerinde inceleyelim.



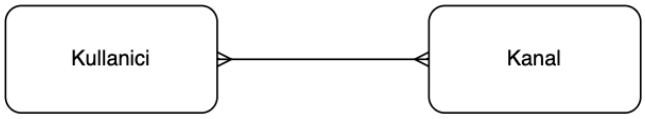
Gorev ve Kategori tabloları çoğa bir ilişki türüyle bağlantılıdır. Bu sebeple ilişki Gorev tablosuna çok, Kategori tablosuna tek işaretiyle bağlanmıştır.

5.1.3. Çoğa çok (m-n)

Ele alacağımız üçüncü ve son bağlantı türü de çoğa çok bağlantı türüdür. Önceki başlık ve örnekleri inceledikten sonra bu ilişki türünün nasıl belirlendiğini kolaylıkla kavrayabilirsiniz. Özetle yine iki varlık alacağız ve bu iki varlık arasındaki ilişki iki yönde de çok bağlantı türüne sahip olacak. Şimdi açıklayalım. Bir A tablosu içerisinde yer alan herhangi bir kayıt, B tablosu içerisinde birden fazla kayıt ile eşleşebiliyorsa

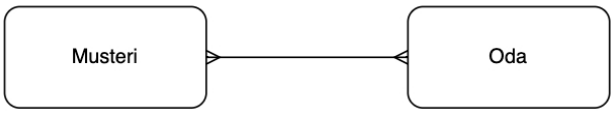
ve B tablosu içerisinde yer alan herhangi bir kayıt A tablosu içerisinde birden fazla kayıt ile eşleşebiliyorsa burada çoğa çok ilişki türünden bahsedebiliriz.

Bir örnek ile devam edelim. Bir video paylaşım sitesinde kullanıcıların kanallara abonelik durumlarını içeren bir veri kümesini ele alalım. Kullanıcılarla ilgili bilgiler Kullanici, sitede yer alan kanalların bilgileri Kanal tablosunda bulunuyor olsun. Kullanici ve Kanal tabloları arasında yer alan ilişki türünü belirleyeceğiz. Kullanici tablosunda yer alan herhangi bir kayıt, Kanal tablosunda birden fazla kayıt ile eşleşebilir mi? Burada eşleşmekten kastımız kanala abone olmak. Dolayısıyla soru diğer bir deyişle bir kullanıcı birden fazla kanala abone olabilir mi? Yanıtımız evet. Böylece iki tablo arasındaki ilişkinin Kanal tablosuyla birleşen ucunun çok türünde olduğunu biliyoruz. Peki Kanal tablosundaki herhangi bir kaydın Kullanici tablosunda birden fazla kayıtla eşleşmesi mümkün müdür? Bu soruyu da diğerine benzer şekilde sorarsak; bir kanala birden fazla kullanıcı abone olabilir mi? Evet. Bu durumda iki tablo arasındaki ilişkinin Kullanici tablosuyla birleştiği yerde de çok ilişki türünü seçmemiz gerekecektir. Sonucu şema ile görelim.



Burada ilişki işaretinin hem Kullanici hem de Kanal tablolarına çok işaretiyle bağlandığını görebiliyoruz.

Bir başka örneği ele alalım. Bir otel içerisinde gelen misafir kayıtları; konaklayan kişi ve bu kişinin hangi odada konakladığı bilgisini içeriyor olsun. Eldeki nitelikler incelendiğinde de basitçe iki tablo elde ettiğimizi varsayalım: Musteri ve Oda. Musteri tablosu, konaklama yapan kişiyle ilgili kişisel nitelikleri içerirken, Oda tablosu, müşterilerin konakladığı odaya dair özelliklerin bulunduğu nitelikleri barındırıyor olsun. Şimdi bu iki tablo arasındaki ilişkiyi inceleyelim. Musteri tablosunda yer alan bir kayıt, Oda tablosunda birden fazla kayıt ile eşleşebilir mi? Yanıt verebilmek için durumu biraz incelememiz gerekecektir. Elbette bir müşteri yalnızca bir odada konaklayabilir. Ancak ele alınan bu kayıtlar uzunca süredir tutulmaktaysa ve kişi bu oteli daha önce de ziyaret ederek farklı bir odada konakladıysa birden fazla oda kaydı ile eşleşmesi mümkün olacaktır. Biz veri tabanlarında geçmişe yönelik kayıtları da bulundurduğumuz için ikinci yanıtı kabul ederek devam edebiliriz: Evet, Musteri tablosunda yer alan bir kayıt, Oda tablosunda birden fazla kayıt ile eşleşebilir. Şimdi Oda tablosu açısından ele alalım. Oda tablosu içerisinde yer alan bir kayıt, Musteri tablosunda birden fazla kayıt ile eşleşebilir mi? Yine benzer bir inceleme yapmamız gerekmektedir. Burada, bir odada birden fazla müşteri kalabileceği gibi, zaman içerisinde ilgili oda birden fazla müşteri tarafından kullanılmış olabilir. Bu sebeple iki tablo arasındaki ilişkinin Oda tablosundan Musteri tablosuna olan yönünde de çok türünde bağlantı kurulması gerekecektir. Yanıtı şema üzerinde inceleyelim.

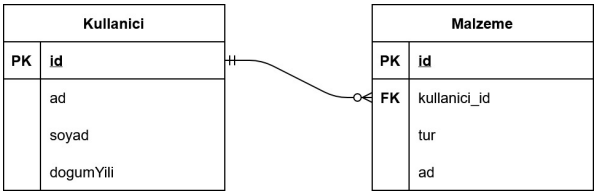


Şemada da görüldüğü üzere Musteri ve Oda tabloları iki yönde de çok işaretiyle birbirlerine bağlanmıştır. Bu sebeple bu örnekteki bağlantı türü de çoğa çok olarak ifade edilebilir.

5.1.4. Zorunlu Kayıt Durumu

İki tablo arasındaki ilişki türü gösterilirken her bir yönde iki işaret kullanılır. Bunlardan birincisinin ne olduğunu önceki başlık altında sunduk. İlişki türünün tek ya da çok olmasına bağlı olarak bir işaretleme gerçekleştirmekteyiz. Bu işaret, bir tabloda yer alan herhangi bir kaydın diğer tabloda birden fazla kayıtla eşleşmesine göre belirlenmektedir. Yani aslında olası en fazla durumu irdelemekteyiz. Bir de olası en az durumu irdelediğimiz bir durum mevcut. Tarif ederek ilerleyelim.

Ele aldığımız bir tablo içerisinde yer alan herhangi bir kayıt, diğer tabloda hiçbir kayıt ile eşleşmeyebilir mi? Bu bizim temel soru cümlemiz. Fark edeceğiniz üzere bir önceki ilişki türünde birden fazla kayıtla eşleşme olasılığını sorgularken, bu ilişki türünde hiçbir kayıtla eşleşme olmaması durumunu irdelemekteyiz. Bu sorunun yanıtına göre; eğer hiçbir kayıtla eşleşmeme ihtimali varsa 0, en az bir kayıtla eşleşme zorunluluğu varsa da 1 işareti koyarak ilişki türünü tamamlıyoruz. Şema ile inceleyelim.



Örnekte Kullanici ve Malzeme tablolarını ve bunlar arasındaki ilişkiyi görmekteyiz. Bu ilişki türünün bire çok olduğunu artık biliyoruz. Ancak bununla birlikte zorunluluk ifadesinin de eklenmiş olduğunu görüyoruz. İlişkiyi iki yönlü de ayrıntılı olarak ele alalım. İlişkinin Kullanici tablosundan Malzeme tablosuna olan yönünde, yani Malzeme tablosuna bitişik yerde çok ve 0 işaretlerini birlikte görüyoruz. Çok işareti; Kullanici tablosunda yer alan bir kaydın, Malzeme tablosunda birden fazla kayıt ile eşleşebileceğini göstermektedir. Yani bir kullanıcı birden fazla malzemeye sahip olabilecektir. Peki, Kullanici tablosunda yer alan bir kayıt, Malzeme tablosunda hiçbir kayıt ile eşleşmeyebilir mi? Yanıtımız evet. Çünkü bir kullanıcı henüz hiçbir malzemeye sahip olmayabilir. Bu durumda kullanıcı kaydı mevcuttur ancak bu kullanıcıya ait herhangi bir malzeme yoktur. Bu yorumumuz neticesinde ilgili ilişki türü için 0 gösterimini kullanmaktayız. Böylece ilişkinin bu yönündeki tür belirlemesi sürecini tamamlamış oluyoruz. Diğer yönü incelersek iki tane 1 işaretinin yer aldığını göreceksiniz. Bunu yorumlamaya çalışalım. Malzeme tablosunda yer alan bir kayıt, Kullanici tablosunda birden fazla kayıt ile eşleşebilir mi? Yanıtımız hayır. Bir malzeme ancak tek bir kişiye ait olabilir. Dolayısıyla tek işaretini kullanıyoruz. En çok durumunu inceledik, şimdi en az durumunu inceleyelim. Malzeme tablosunda yer alan bir kayıt, Kullanici tablosunda hiçbir kayıt ile eşleşmeyebilir mi? Bunun yanıtı da hayır. Bir malzeme sisteme kaydedildiyse bir kullanıcı ile eşleştirilmesi gerekmektedir. Bu sebeple Malzeme tablosunda yer alan kayıtlar Kullanici tablosunda en az bir kayıt ile eşleşmek zorundadır. Hem en az bir kayıtla eşleşme hem de en fazla bir kayıtla eşleşme durumu birlikte gerçekleştiği için ilişki türü iki tane 1 kullanılarak ifade edilmiştir.

Böylece ilişki türlerinin tüm gösterimlerini ele almış olduk. Bir ilişki türünü tanımlamak için hem en az hem de en çok durumlarını inceleyerek bunu şema üzerinde gösterdik. Şimdi bu ilişkilerin veri bütünlüğünü nasıl sağlayacağı üzerine gideceğiz. Bunu yapmanın yolu da, anahtarlardır.

5.2. Anahtarlar

Önceki başlıkta ilişki türlerini ve zorunluluk ifadesini, nasıl tanımlanması gerektiğini ve şema üzerinde gösterim şeklini gördük. Tabloların birbirleriyle nasıl bağlantılı olduklarını artık biliyoruz. Ancak bir ayrıntı var. Daha önceki başlıklarda da ele aldığımız üzere, veri tabanı tasarımı sürecinde biz en başta elimizde tek bir elektronik tablo olarak bulunan veri kaydını birden fazla tabloya böldük. Böylece elimizde birden fazla bağımsız tablo oldu. Ancak bu tek başına etkin bir çözüm olamaz. Çünkü bağımsız olarak ele alınan veri tabanı tabloları içerisinde yer alan veri, anlam kaybına uğramıştır. Bunun da ötesinde tekrar birleştirmenin de bir yolu kalmayacaktır. Veri tabanının avantajlarından bahsederken veri bütünlüğünün korunuyor olduğunu söylemiştik. Dolayısıyla bu problemi aşacak bir şey yapmamız gerektiği açıktır. Bunun da yolu anahtarları kullanmaktır.

Anahtar kelimesi, bir tabloda yer alan niteliğin temsil edicilik özelliğini göstermektedir. Fonksiyonel bağımlılık konusunda bu durumu detaylı şekilde ele almıştık. Bir tabloda yer alan hangi niteliklerin temsil edici özellikte olduğunu, dolayısıyla hangi niteliklerin diğerlerine fonksiyonel bağımlı olduğunu tespit etmeye çalışıyorduk. Ancak fonksiyonel bağımlılıkların belirlenmesi, anahtar nitelik belirlemekten öte nitelikler arasındaki ilişkileri fark edebilme ve nitelikleri gruplayabilme açısından faydalıdır. “En düşük riskli” niteliğin seçilip anahtar olarak kullanılması, pratikte uygun olmayan bir çözümdür. Bir anahtar nitelik her durumda tekil olarak bir kaydı işaret etmek zorundadır. Bunun istisnası yoktur. Bu sebeple veri tabanı tasarımcıları genellikle anahtar olarak mevcut bir niteliği seçmek yerine yeni bir nitelik tanımlayarak ve bu niteliği yöneterek kendi anahtar niteliklerini oluşturma yolunu tercih ederler. Tablo içerisinde anahtar olmaya müsait bir nitelik olsa bile bu yöntem sıklıkla tercih edilir. Örneğin kişilerle ilgili kayıtlarda kimlik numarası bilgisini anahtar olarak kullanmak oldukça iyi bir çözümdür. Ancak yine de id adında bir nitelik tanımlayarak kullanıcıyı id niteliğine bağlı olarak yönetmek tercih edilmektedir.

İki ayrı tablo, bir ilişki türü ile birbirine bağlanırken tablolarda yer alan nitelikler üzerinden bu süreç gerçekleştirilir. Tablolar arasındaki ilişkiyi gösteren şema, iki tablo üzerinde de bir niteliği işaret eder. Böylece iki tablonun hangi nitelikler üzerinden bağlandığı da tanımlanmış olur. Niteliğin tablo içerisindeki

görevi, hangi tür anahtar olduğunu gösterir. Az önce örneğini verdiğimiz, temsil edici nitelikteki anahtarlar birincil anahtar olarak adlandırılırlar. Veri bütünlüğünü korumak için bir birincil anahtarın başka bir tablo içerisinde sunulması durumunda elde edilen niteliğe ise ikincil anahtar adı verilmektedir. Şimdi anahtarlara dair ayrıntıları daha detaylı inceleyelim.

5.2.1. Birincil Anahtar

Birincil anahtar, varlık içerisinde her bir kayıt için benzersiz olan, dolayısıyla her bir değeri tek bir kaydı işaret eden niteliktir (Chen, 1976). Bir diğer deyişle tüm niteliklerin fonksiyonel bağımlı olduğu nitelik denebilir. Sıkça karşılaştığımız ID'ler birincil anahtarların en yaygın örneğidir. Mevcut bir niteliği birincil anahtar olarak kullanmak her zaman bir riske sahiptir. Ancak veri tabanlarında birincil anahtarların asla tekrar etmemesi gerekmektedir. Bu sebeple genellikle yeni bir nitelik birincil anahtar görevini üstlenmek üzere oluşturulur.

Birincil anahtarlar her ne kadar varlıklar arası bağlantıların oluşturulması için kullanılıyor olsa da genellikle tüm varlıklara (bir bağlantıya sahip olup olmadığına bakmaksızın) eklenirler. Dolayısıyla birincil anahtarın ne ve nasıl seçileceği varlığın sahip olduğu bağlantı türlerinden bağımsızdır. Bir kitap için ISBN, bir kişi için kimlikNumarasi ve bir film için IMDB ID'si, veri tabanı yapısı ve varlığın bağlantılarından bağımsız olarak tanımlanabilir.

Birincil anahtar olarak kullanılmak üzere oluşturulacak niteliklerin çeşitli biçimleriyle karşılaşıyoruz. Alt başlıklarda bu türler ele alınacaktır.

Sıralı Sayısal Birincil Anahtar

En sık karşılaşılan bu anahtar türünde kayıtlar 1'den başlayarak ardışık değerler alırlar. Varlığa eklenen her yeni kayıt için en son atanan sayının bir fazlası atanır. Silinen kayıtlar için kullanılan sayılar tekrar kullanılmazlar. Böylece her kaydın benzersiz bir birincil anahtara sahip olması garanti altına alınmış olur. Bunu sağlamak genellikle veri tabanı yönetim sistemi içerisinde ilgili nitelik tanımlanırken birincil anahtar ve AUTO_INCREMENT özelliği verilerek gerçekleştirilir.

Birçok sistem içerisindeki birçok varlık için bu yaklaşım, uygulanmasının da oldukça kolay olması sebebiyle sıklıkla tercih edilir. Ancak hassas veri içeren varlıklar için bu yöntem bir güvenlik açığı oluşturabilir. Örneğin laboratuvar sonuçlarını aldığımız kodlar sıralı sayılar ile kodlanmış olsaydı, dileyen herkes sırayla tüm sayıları deneyerek diğer hastaların kayıtlarına da ulaşabilirdi. Bir diğer örnek olarak; çok fazla içeriğe sahip bir websitesinin her bir içeriğini doğrudan sıralı sayılar ile erişime açması, kötü niyetli birinin kolayca tüm içeriği kolayca ele geçirmesine sebep olabilir. Yazılımsal ve donanımsal yükün çok az olması sebebiyle akla gelecek ilk yöntem olması gereken sıralı sayısal birincil anahtarların bir güvenlik açığı yaratıp yaratmayacağı göz önünde bulundurulmalıdır.

Rastgele Sayısal Birincil Anahtar

Bu yöntem ile birincil anahtar, belirlenen aralıktaki rastgele bir sayı seçilerek oluşturulur. Sıralı olmaması, sıralı olması durumunda karşılaşılacak riski ortadan kaldıracaktır. Bu yöntemin dezavantajı ise her yeni kayıt için yeni bir sayı üretme, bu sayının varlık içerisinde olup olmadığını kontrol etme ve eğer varsa yeni bir tane üretme gibi süreçler sebebiyle zaman maliyetli olmasıdır. Ayrıca belirli bir aralık tanımlanacağı için bu aralığın gelecekte de yeterli olacağından emin olunmalıdır.

Alfanumerik Birincil Anahtar

Sayılar çoğu durumda anahtar görevini çok iyi üstlenmektedirler. Ancak 10 sayı tabanını kullanmamız sebebiyle her bir rakam elde edebileceğimiz farklı durum sayısı 10'dur. Bu da durum sayısı arttıkça sayıyı oluşturan rakamların sayısının da artması anlamına gelmektedir.

Alfanumerik anahtarlar hem sayı hem de karakterleri birlikte kullanmaktadırlar. Burada karakterleri de kullanma amacı kullandığımız sayı tabanını artırmak, bu sayede tek bir karakter ile çok daha fazla durum ifade edebilmektir. Sadece sayıları kullanarak tek bir karakter ile 10 durum ifade edebilirken, sayılar ve

İngiliz alfabesini kullanarak 36 durum ifade edebiliriz. Bir diğer deyişle 10 sayı tabanından 36 tabanına geçmiş oluruz.

Örneğin; 8 haneli sayısal bir içerik en fazla $10^8 = 100.000.000$ durum ifade edebilirken, alfanumerik içerik $36^8 = 2.821.109.907.456$ durum ifade edebilir. Bu da yaklaşık 28.000 kat daha fazla seçenek demek. İki durumda da aynı sayıda (8) karakter kullandığımızı hatırlatmak isterim.

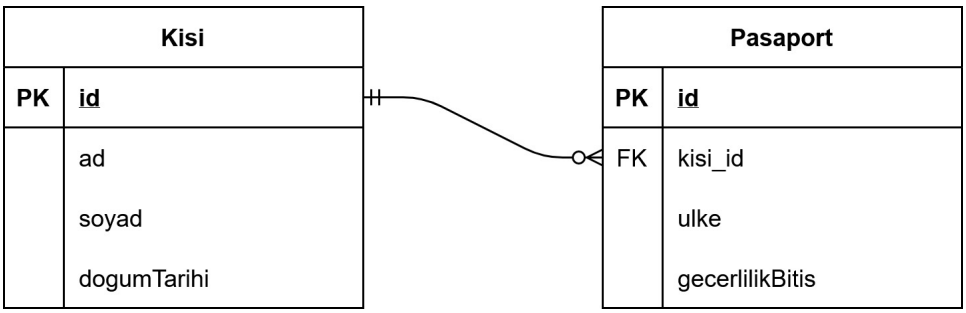
Bu sebeplerle eğer kayıtlarımızın sayısı çok fazlaysa ve biz bu kayıtları daha az karakterle ifade etmek istiyorsak, alfanumerik anahtarlar doğru bir tercih olabilir. Gerçek hayatta doktordan alınan reçete kodları, elektronik imza ispatları ya da renk kodları (16 sayı tabanı kullanırlar) alfanumerik anahtar kullanımına örnek teşkil ederler.

Kontrollü Birincil Anahtar

Kontrollü anahtarlar, anahtarın belirlenen bir kural ya da daha fazla kurala uymasıyla oluşturulurlar. Böylece anahtar rastgele oluşturulamaz. Her bir anahtarın geçerliliği bu kurallara uygunluğu kontrol edilerek doğrulanabilir. Kimlik numaraları bu anahtarlara uygun bir örnektir. 11 haneli olan kimlik numarasının ilk 10 hanesinin toplamının 10'a bölümünden kalanı son rakamı vermelidir. Ayrıca kimlik numarası doğrulamak için daha karmaşık kontroller de vardır. Bu sayede beyan edilmiş bir kimlik numarasının rastgele ya da gerçek olup olmadığı kolayca tespit edilebilir.

5.2.2. İkincil anahtar

Tablolar arası bağlantılar, birbirleriyle ilişkili iki tablodan birinin birincil anahtarının bir başka tablo içerisinde nitelik olarak yer almasıyla sağlanır. Bir başka tablo içerisinde tekrar eden bu anahtar değerlere ikincil anahtar adı verilmektedir. Ayrıntılardan bir örnek üzerinde bahsetmekte fayda var.



Şekil, bire çok ilişkili Kisi ve Pasaport tablolarını içeren bir veri tabanı tasarımı sunmaktadır. Bu tasarımda iki tablo da id adlı birer birincil anahtara sahiptirler. Ayrıca Pasaport tablosu içerisinde kisi_id adlı nitelik, ikincil anahtar özelliğine sahiptir. Bu nitelik, eşleşen kayıtlar için Kisi tablosundaki kayıtların birincil anahtar değerini taşır. Bu sayede Kisi ve Pasaport tablolarındaki kayıtlar olması gerektiği şekilde eşleşebilirler.

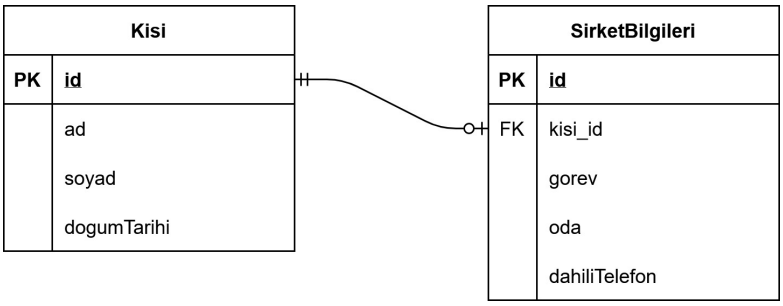
Eğer tablolar özet olarak (yalnızca tablo adını içeren şekilde) gösterilmiyorsa, ilişkiler tablolar arası değil nitelikler arasında tanımlanırlar. Bu sayede hem tablolar arasındaki ilişki tespit edilebilir, hem de hangi birincil ve ikincil anahtarlar kullanılarak verinin birleştirilebileceği rahatça anlaşılabilir.

İkincil anahtarlar bire çok ilişkili tablolar söz konusu iken çok bağlantının sağlandığı tabloya eklenirler. Bunun sebebini anlamak ve ezberlemek zorunda kalmamak için aykırı durumu inceleyelim. Şekil ile verilen örnekte ikincil anahtarın Pasaport değil de Kisi tablosunda olduğunu ve pasaport_id adlı bir nitelik olduğunu varsayalım. Bu durumda Kisi tablosunu incelemek istediğimizde, birkaç örnek kayıt sayesinde Tablodaki gibi bir görüntü elde edebiliriz.

id	pasaport_id	ad	soyad	dogumTarihi
1	1,5,9	Evangeline	Taylor	04.08.2001
2	2,3	Herbert	Todd	01.09.1987
3	6	Alvin	Miles	24.02.1995

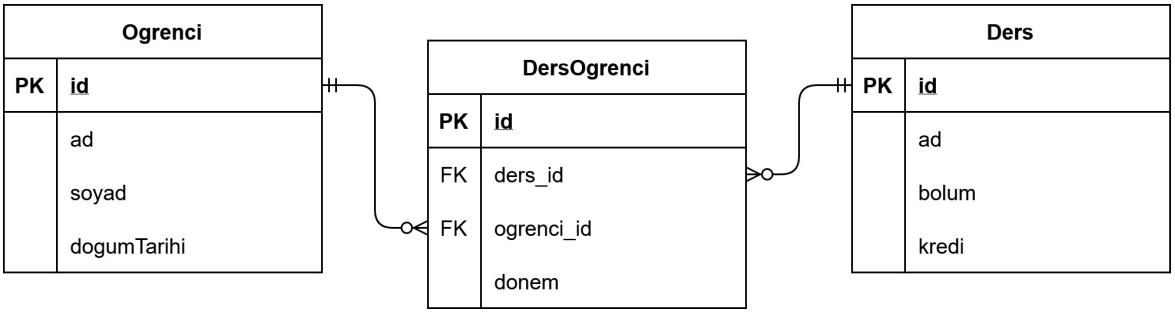
Bire çok ilişkinin; bir tablo içerisindeki tek bir kaydın birden fazla kayda denk gelme durumunda kullanıldığından bahsetmiştik. İkincil anahtar iki tablodan birinde yer almak zorundadır. Eğer Tablo'daki gibi Kisi tablosu içerisinde yer alırsa, her bir kişinin birden fazla pasaporta sahip olabilmesi sebebiyle pasaport_id niteliği birden fazla değer alabilir. Bu da iyi bir veri tabanı tasarımı oluşturmanın öncüllerinden biri olan, bir hücre yalnızca tek bir (atomik) kayda sahip olmalıdır kuralına aykırıdır. Eğer ikincil anahtar Pasaport tablosu içerisinde olursa, her bir pasaport yalnızca tek bir kişiye ait olacağı için her zaman tek bir değer alacaktır. Bu sebeple bire çok ilişkili tablolar için ikincil anahtar her zaman tek bir kayda eşlenen tablo içerisinde yer almalıdır.

Bire bir ilişkili tablolarda eğer ilişki türünde zorunluluk tanımı yoksa ikincil anahtarın hangi tablo içerisinde yer aldığı önemli değildir. Ancak zorunluluğun tanımlandığı durumlarda zorunlu olmayan taraftaki tabloda ikincil anahtarı bulundurmak, toplam veri hacmini azaltan; dolayısıyla en etkili çözüm olacaktır. Şekil buna bir örnek sunmaktadır.



Şekildeki örnekte kişiler ve bu kişilerin şirket bilgileri saklanmaktadır. İlişki türündeki zorunluluk tanımı göz önünde bulundurulduğunda; bir kişinin mutlaka şirket kaydının bulunması gerekmediği; ancak bir şirket bilgisinin mutlaka bir kişiye ait olması gerektiği görülmektedir. İkincil anahtarı Kisi tablosuna yerleştirmemiz durumunda, SirketBilgileri kaydı bulunmayan kişi kayıtları için bu nitelik boş değer alacaktır. Ancak anahtarı SirketBilgileri tablosuna yerleştirirsek mutlaka bir kişiye bağlı olduğu için her zaman bir değeri olacaktır. Ayrıca yine zorunluluk tanımı sebebiyle Kisi varlığındaki kayıt sayısının SirketBilgileri tablosundakinden fazla olması beklenmektedir. Bu durumda az kaydolan tabloya ekleyeceğimiz yeni nitelik, veri tabanının toplam hacminde diğer varlığa eklemeye kıyasla azalma sağlayacaktır.

Son olarak çoğa çok bağlantı türü için ikincil anahtarları ele alalım. Bu ilişki türünden bahsederken üçüncü bir tablo oluşturularak bunun üzerinden bağlantı kurulduğundan bahsetmiştik. Çoğa çok ilişkide her iki tablo da diğer tabloda birden fazla kayda denk geldiğinden ikincil anahtarı bu iki tablodan birine yerleştirerek elde edeceğimiz tasarım, veri tabanı ilkelerine aykırı olacaktır. Bu sebeple oluşturduğumuz üçüncü tablo içerisinde iki tabloyu da temsil etmek üzere iki tane ikincil anahtar tanımlanır. Şekil bunun bir örneğini sunmaktadır.

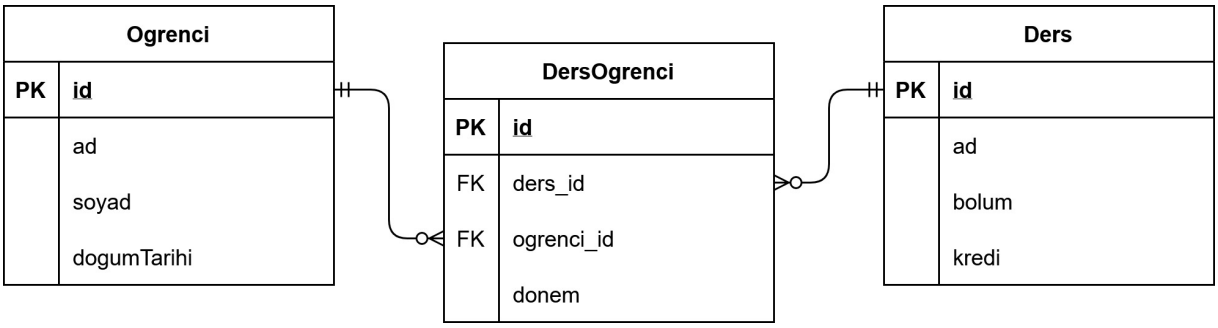


Örnekte verilen veri tabanı tasarımı; bir öğrenci birden fazla ders alabilir, bir ders de birden fazla öğrenci tarafından tercih edilebilir durumuna uygun olarak hazırlanmıştır. Bir öğrencinin aldığı birden fazla dersi Ogrenci tablosunda tutmak ya da bir dersi alan öğrencileri Ders tablosunda tutmak etkin bir yöntem olmayacaktır. DersOgrenci adlı bağlantı tablosu, bu iki tablonun tüm kesişim kayıtlarını tutmaktadır. Her bir ders-öğrenci ilişkisi tek bir öğrenci ve tek bir ders kaydına denk geleceği için ikincil anahtarların bu tabloya yerleştirilmesi gerekmektedir. Bunun yanında bağlantı tablosu, yapılan eşleştirmeyle ilgili farklı kayıtları da barındırabilir. Bu örnekte öğrencinin dersi aldığı dönem bilgisi bağlantı tablosu üzerindedir.

5.3. Bağlantı (Pivot) Tablosu

İki tablo arasında çoğa çok bağlantı kurulduğunda, bu bağlantıyı tablolardan birine ikincil anahtar koyarak gerçekleştiremeyiz. Bunun sebebi, iki tablo için de herhangi bir kaydın karşı tabloda birden fazla kayıtla eşleşme olasılığının olmasıdır. Bir kaydın karşıdaki tabloda çok kaydı işaret etmesinin tek yolu, karşıda yer alan kayıtların, gösterici kaydın birincil anahtarını ikincil anahtar olarak kullanmasıdır. Biraz daha açalım. Kullanıcı ve Kanal tablolarını düşünelim. Bir kullanıcı birden fazla kanalı takip edebilir. Bunu veri tabanına işlemek için Kanal tablosu içerisinde kullanici_id adında bir ikincil anahtar belirlenir ve bu anahtar, Kullanici tablosundaki birincil anahtarın değerini alır. Kanal tablosunda yer alan herhangi bir kayıt, hangi kullanici_id değerini içeriyorsa aslında o kayıt ile eşleşmektedir. Ancak ilişkinin tersi yönünde de çok kayıt ile eşleşme ihtimali vardır. Bir kanal birden fazla kullanıcı tarafından takip edilebilir. Bunun gösterimi için de Kullanici tablosunda ikincil anahtar bulunması ve Kanal tablosunun birincil anahtarının kullanılması gerekir. Pratik olarak bunu gerçekleştirmek mümkün değildir. Bu sebeple çoğa çok ilişki olduğu durumlarda bir tablo daha oluşturulur ve ikincil anahtarlar bu tablo içerisine yerleştirilir. Böylece Kullanici ve Kanal tablolarının kayıtlarının bir araya geldiği her türlü kombinasyon, bu tablo içerisinde tutulmuş olur.

Elde edilen bu tabloya bağlantı tablosu ya da pivot tablo adı verilir. Bir örnek ile en basit halini inceleyelim ve sonrasında bu tablonun yetenekleri üzerine tartışalım.



Örnekte Ogrenci ve Ders tablolarını görmekteyiz. Bu tablolar arasındaki ilişki türü çoğa çoktur. Bir öğrencinin çok sayıda dersi olabilir ve bir ders çok sayıda öğrenci tarafından tercih edilebilir. Çoğa çok ilişki türüne sahip iki tablo gördüğümüzde hiç tereddüt etmeden bağlantı tablosunu ekleyebiliriz. Çünkü ikincil anahtarları yerleştirmenin tek yolu budur. Eğer bağlantı tablosuna bir ad vermekte zorlanıyorsak alfabetik olarak bağlı tabloların adını kullanmak yaygın bir yöntemdir. Bağlantı tablosunda yer alan her bir kayıt, iki ana tablonun kesiştiği kayıtları ifade etmektedir. Ancak burada bir ayrıntı daha var. Gördüğünüz üzere, bağlantı tablosu içerisinde bir de donem niteliği mevcut. Bunun sebebini incelemeden önce bir ayrıntıyı vurgulamak iyi olacaktır.

Bağlantı tabloları gerçek hayatta genellikle bir işlevi sembolize ederler. İki temel olgu arasında kayda alınması gereken bir işlev olduğunda bu bağlantı tablosu ile kayıt altına alınır. Örnek vermek gerekirse; Ogrenci ve Ders birer varlıktır. Ancak bu ikisinin ilişkisi bir faaliyettir. Öğrencinin dersi alma durumu, faaliyet olarak kayda geçmelidir. Bu sebeple örnekteki bağlantı tablosunun adını DersAlma olarak yazmamız da mümkündü. Şimdi donem niteliğine dönersek eğer; bir bağlantı tablosu, kayıt altına aldığı işlevle birlikte bu işleve dair ayrıntıları da barındırabilir. Eğer Ogrenci ve Ders tabloları DersAlma işlevi altında bir araya geliyorlarsa bu ders alma eylemiyle ilgili diğer ayrıntılar da bu tablo içerisinde yer alabilir.

Sosyal medya siteleri buna güzel bir örnektir. Video paylaşım sitesinden örnek verelim. Kullanici ve Video tablolarının ilişki türünü incelerseniz çoğa çok olduğunu göreceksiniz. Aynı zamanda bir kullanıcının bir video ile çok farklı şekillerde etkileşime geçmesi de mümkündür. Kullanıcı videoyu izleyebilir, beğenebilir, beğenmeyebilir, yorum yapabilir, şikayet edebilir ya da kendisi oluşturmuş olabilir. Tüm bu işlevler için iki tablo arasında bağlantı tabloları kurulabilir. Burada önemli olan ayrıntı şudur ki, her bir işlev için yeni bir bağlantı tablosu kurulması gerekmektedir. Yani çoğa çok bağlantılı iki tablo arasındaki bağlantı tablosu her zaman bir tane olmak zorunda değildir. Bununla birlikte bağlantı tablosu içerisindeki nitelikler de o eylemin ayrıntılarına bağlı olarak belirlenebilir.

Bağlantı tabloları fark edeceğiniz üzere yalnızca ikincil anahtarları yerleştirmek için değil, aynı zamanda faaliyetlerin kayda alınması için kullanılan önemli tablolardır. Birçok sistemde kayıtsal veri dediğimiz veri türleri bu tür tablolarda yer alırlar. Örneğin geçiş bilgileri, satış kayıtları, sürekli tekrarlanan kayıtsal işlemler bu tablolar üzerinden yaparlar. Bu sebeple bağlantı tabloları genellikle çok fazla kayda sahip olur. Yıldız şema başlığı altında bu konuya daha detaylı değineceğiz. Bu başlık itibariyle kesin olan konu şudur ki; çoğa çok bağlantılı iki tablo arasında mutlaka bir bağlantı tablosu tanımlanmalıdır. Bununla birlikte bu bağlantı

tablosunun hangi faaliyeti kaydetmek için tasarlandığı da tespit edilmeye çalışılmalı, mümkünse bu faaliyete uygun bir tablo adı verilmelidir. Faaliyetin araştırılması, iki tablo arasında başka faaliyetler bulunmasının da mümkün olduğu durumlarda diğer olası bağlantı tablolarının da belirlenmesini ve uygulanmasını sağlayacaktır. Bu sayede olası bir hatanın önüne geçilebilir.

Bölüm Özeti

İlişkisel veri tabanları iki tür ilişki üzerine kuruludur. Birincisi nitelikler arasındaki ilişkidir. Bu tür ilişkiyi bir önceki başlık altında, fonksiyonel bağımlılıkla birlikte incelemiştik. İkinci ilişki türü ise tablolar arasındaki ilişki türüdür. Bu bölümde, tablolar arasındaki ilişkileri ayrıntılı şekilde ele aldık.

Tablolar arasında üç tür ilişki türü bulunması mümkündür. Bunlar: Bire bir, bire çok ve çoğa çok ilişki türleridir. Tablolar arasındaki ilişki türleri yönlü olarak kabul edilebilir. İlişki türü belirlenirken de bu yönden faydalanılmaktadır. Bir ilişki türü, ele alınan tabloda yer alan herhangi bir kaydın, diğer tabloda birden fazla kayıt ile eşleşip eşleşmeme durumuna göre belirlenir. Birden fazla kayıtla eşleşmesi çok, eşleşmemesi ise tek işaretiyle gösterilir. Aynı tayin işlemi ilişkinin ters yönünde de ele alınarak ilişkinin diğer ucunda yer alması gereken işaret de belirlenir. Nihayetinde ele alınan tablolara bağlanan ilişki türleri sırasıyla okunarak ilişki türü belirlenmiş olur. A ve B tablolarını düşünelim. İkisi arasındaki ilişkiyi belirlerken, bu ilişki türünün B tablosuna bağlanan ucu çok, A tablosuna bağlanan ucu tek işareti alsın. Bu durumda A ve B tabloları arasındaki ilişki türü bire çoktur. Tabloların yerlerini değiştirdiğinizde ilişki türleri de ters döneceği için ilişki türü çoğa bir olarak anılır.

İlişki türü belirlemesinde bir de zorunluluk ifadesi araştırılır. İlişki türü aranırken en çok eşleşme kontrol edilirken, bu araştırmada en az durum araştırılır. Bir tabloda yer alan kaydın, diğer tabloda herhangi bir kayıtla eşleşmemesinin mümkün olup olmadığı kontrol edilir. Eğer mümkün ise 0, değil ise 1 işareti kullanılır. Araştırma iki yönlü de gerçekleştirilerek 1 ve 0 işaretlerinden uygun olan kullanılır.

Bir veri kümesinin birden fazla alt parçaya bölünmesi, veri bütünlüğünün kaybolmasına sebep olabilir. Bu bütünlüğün sağlanması, yalnızca ilişki türlerini belirlemekle mümkün olmamaktadır. Belirlenen ilişki türleri, aynı zamanda anahtar özelliği verilmiş nitelikler üzerinden gerçekleştirilmektedir. İki tür anahtar bulunmaktadır: Birincil ve ikincil anahtarlar.

Birincil anahtar, bir tabloda yer alan temsil edici niteliktir. Bu niteliğe ait herhangi bir değer yalnızca tek bir kaydı işaret etmek zorundadır. Her tabloda mutlaka bir birincil anahtar bulunmaktadır. Mevcut niteliklerden birisi birincil anahtar olarak atanabilir ancak pratikte bu tercih edilen bir yöntem değildir. Herhangi bir risk ile karşı karşıya kalmamak için her tabloda bağımsız bir birincil anahtar niteliği tanımlanır. Çeşitli birincil anahtar belirleme yöntemleri bulunmaktadır. En sık kullanılanı sıralı sayısal birincil anahtardır. Bu anahtar aynı zamanda karşımıza genellikle “id” adıyla çıkar. “id” niteliği, 1’den başlayarak her yeni kayıt için bir arttırılarak değer alan sayısal bir niteliktir ve oldukça kullanışlıdır. Ancak toplam kayıt sayısı ve belirli bir sürede gerçekleştirilen yeni kayıt eklemeleri gibi örtük bilgileri içerisinde barındırdığı için güvenlik öncelikli bazı durumlarda tercih edilmezler.

İkincil anahtarlar ise bir birincil anahtarın bağlantılı olduğu tablo içerisinde veri bütünlüğünü sağlamak için tutulmasıyla elde edilen niteliktir. Bire çok bağlantılı tablolarda ikincil anahtarlar çok bağlantısı kurulan tabloda yer alırlar. Bunun sebebi, bu tabloda bulunan çok sayıda kaydın bağlı olduğu asıl kaydı işaret etmesidir. Bağlantı türü tek olan tabloda ikincil anahtar bulunması, çoklu bağlantı sebebiyle birden fazla değer almasını gerektirir. Bu sebeple tercih edilmez. İkincil anahtarlar her zaman çok türünde ilişkinin bağlı olduğu tabloda bulundurulurlar.

Çoğa çok bağlantıya sahip tablolar, çapraz olarak birçok birleşim kombinasyonuna sahip oldukları için ikincil anahtar ile doğrudan bağlanamazlar. Bu sebeple bu ilişki türünde iki tablo arasına bağlantı tablosu adı verilen, ikincil anahtarların ve dolayısıyla iki tablo arasındaki eşleşmelerin tutulduğu bir tablo oluşturulur. Bağlantı tabloları genelde bir eylemi ifade ederler. Bu sebeple çoğa çok bağlantılı iki tablo arasında mümkün olan her türlü eylem için birer bağlantı tablosu oluşturulabilir. Bir sosyal medya ağında kullanıcı ile içerik arasında beğenme, takip etme, yorum yapma ve benzeri eylemler gerçekleştirilebilmesi buna örnektir.

Kaynakça

Akadal, E. 2020. Veritabanı Tasarlama Atölyesi. Türkmen Kitabevi, İstanbul.

Hoffer, A. J., Ramesh, V. & Topi, H. 2016. Modern database management. Pearson.

Chen, P. 1976, The entity-relationship model-toward a unified view of data. ACM transactions on database systems (TODS).