**Normalizasyon Nedir? *Normalizasyona neden ihtiyaç duyarız? Neden kullanmalıyız?***

Normalizasyon; veri tabanı tasarım aşamasında veri tekrarını, veri kaybını veya **veri yetersizliğini** önlemek için gerçekleştirilen işlemlerdir. Normalizasyon tanım olarak ise ; (Ayrıştırma), veri tabanlarında çok fazla sütun ve satırdan oluşan bir tabloyu tekrarlardan arındırmak için daha az satır ve sütun içeren alt kümelerine ayrıştırma işlemidir.

Oluşturulacak veritabanını doğru tasarlamak ve belirli kurallara bağlı kalmak, sonrasında ilgili veritabanından veri sorgulamayı ve o veritabanını verimli kullanmayı daha kolay ve daha esnek hale getirecektir. Her şeyden önemlisi iyi bir normalizasyon, veritabanı yönetim sistemini kolaylaştıran önemli bir unsurdur.

Normalizasyon veritabanını sunucu üzerinde etkili çalıştırır ve o veritabanından iyi performans alınmasını sağlar. Bu performans, hem veritabanının sunucu üzerinde hızlı çalışması ve/veya bellek tasarrufu sağlaması hem de kullanıcıların o veritabanından yapacağı sorgulara en hızlı ve kolay cevap vermesi ile ilgilidir.

Normalizasyonun temelinde yatan asıl hedef, **veritabanı tablolarını küçük parçalara ayırıp, onlar arasında ilişki kurarak, verileri daha kolay yönetme mantalitesi yer almaktadır**.

Normalizasyon, veri tabanlarına seviyelerle (normal formlar) uygulanır. Bir veri tabanının bu normal formlardan herhangi birine uygun olduğunu söyleyebilmek için, söz konusu normal formun tüm kriterlerini eksiksiz yerine getiriyor olması şarttır.

Başarılı bir şekilde uygulandığında normalizasyon işlemi veri tabanının süratini büyük oranda artırır. Veri tabanının sabit diskteki boyutunu azaltır. Ayrıca veri tutarlılığını artırarak veri tekrarlarını engeller.

***Normalizasyonun Amacı***

Normalizasyonun amacı, gereksiz veri tekrarını ortadan kaldırarak, veri fazlalığını en aza indirmektir. Veri tekrarı, veri anomalisine(data anomalies) yol açar. Bu da veri bütünlüğünün(data integrity) bozulmasına neden olur.

***Veri anormalliği 3 şekilde olabilir.***

Güncelleme (Update Anomalies), Ekleme (Insert Anomalies), Silme (Delete Anomalies)

İyi tasarlanmış veritabanı, güncelleme, silme, ekleme problemlerine yol açmamalıdır.

***Normalizasyonun Avantajları***

* Veri bütünlüğünü sağlar.
* Verimli bir veri yapısı sunar.
* Gereksiz veri tekrarını engeller, minimum alan kullanılır ve yerden tasarruf sağlar.
* Saklanan veri daha anlaşılır hale gelir.
* Hızlı sorgulama imkânı verir.

***Normal Formlar***

Basitçe tanımlamak gerekirse, normal formlar normalizasyon seviyeleridir. Bu seviyeler gereksiz veri tekrarlarını ne derecede engellediği ve tutarlılığı ne kadar sağladığına bağlı olarak derecelendirilir. Seviye yükseldikçe veri tutarlılığı artar, veri tekrarı düşer.

Normalizasyon seviyeleri 1NF (Birinci Normal Form), 2NF, 3NF, BCNF(Boyce-Codd Normal Form, 3.5NF'de denir), 4NF, 5NF şeklinde adlandırılır ve yukarı doğru devam eder. Ancak daha yukarı normalizasyon seviyeleri çok nadiren kullanılır çünkü çoğu zaman uygulanması mümkün olmayabilir.

Konuyu detaylandırabilmek için bir veri tabanı oluşturalım ve normalizasyonunu yapalım. Tabloda bir teknik destek firmasının çalışanları, servis araçları, servis şoförleri ve servis verilen semtler bulunsun. Her bir şoför tek araç ile semt bazında servis yapmaktadır. Örneğin şoför "Ahmet", teknik elemanları (çalışanları) "Toyota" araçla, "Levent", "Etiler" ve "Ulus" semtlerindeki destek çağrılarına götürmektedir.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Calisan** | **Soyad** | **Sofor** | **Arac** | **Semt** |
| Orçun | Yılmaz | Ahmet | Toyota | Levent, Etiler, Ulus |
| Metin | Seyyar | Mehmet | Honda | Bakırköy, Ataköy, Yeşilköy |
| Metin | Seyyar | Tolga | Ford | Kandilli, Beylerbeyi, Kuzguncuk |

***1NF (1. Normal Form)***

Bir veri tabanının 1NF olabilmesi için aşağıdaki özellikleri karşılayabilmesi gerekir:

* Aynı tablo içinde tekrarlayan kolonlar bulunamaz,
* Her kolonda yalnızca bir değer bulunabilir. (bkz. "Semt" kolonu)

Veri tabanımızda ikinci kurala açıkça uymayan bir kolon var: Semt. Bu durumu düzeltmek için tekrar düzenleyelim:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Ana Tablo** |  |  |  |  |  |  |
| **Calisan** | **Soyad** | **Sofor** | **Arac** | **Semt 1** | **Semt 2** | **Semt 3** |
| Orçun | Yılmaz | Ahmet | Toyota | Levent | Etiler | Ulus |
| Metin | Seyyar | Mehmet | Honda | Bakırköy | Ataköy | Yeşilköy |
| Metin | Seyyar | Tolga | Ford | Kandilli | Beylerbeyi | Kuzguncuk |

Bir sorun var. Tablo bu şekliyle birinci kuralla çelişti. Semt 1, Semt 2, Semt 3 tekrarlayan kolonlar. Bir daha deneyelim:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ana Tablo** |  |  |  |  |
| **Calisan** | **Soyad** | **Sofor** | **Arac** | **Semt** |
| Orçun | Yılmaz | Ahmet | Toyota | Levent |
| Orçun | Yılmaz | Ahmet | Toyota | Etiler |
| Orçun | Yılmaz | Ahmet | Toyota | Ulus |
| Metin | Seyyar | Mehmet | Honda | Bakırköy |
| Metin | Seyyar | Mehmet | Honda | Ataköy |
| Metin | Seyyar | Mehmet | Honda | Yeşilköy |
| Metin | Seyyar | Tolga | Ford | Kandilli |
| Metin | Seyyar | Tolga | Ford | Beylerbeyi |
| Metin | Seyyar | Tolga | Ford | Kuzguncuk |

1NF kurallarına harfiyen uyduk.

***2NF (2. Normal Form)***

Bir veri tabanının 2NF olabilmesi için aşağıdaki özellikleri karşılayabilmesi gerekir:

* Tablo 1NF olmalıdır,
* Her satır bir eşsiz anahtarla tanımlanmalıdır (Unique Key - Primary Key)
* Anahtar olmayan değerler ile kompozit (bileşik) anahtarlar arasında kısmi (partial) bağımlılık durumu oluşmamalıdır. Kısmi bağımlılık durumu, anahtar olmayan herhangi bir değer kompozit bir anahtarın yalnızca bir kısmına bağıl ise oluşur.
* Herhangi bir veri alt kümesi birden çok satırda tekrarlanmamalıdır. Bu tür veri alt kümeleri için yeni tablolar oluşturulmalıdır.
* Ana tablo ile yeni tablolar arasında, dış anahtarlar (foreign key) kullanılarak ilişkiler tanımlanmalıdır.

Eğitmene Not: Aday anahtar ve eşsiz anahtarın bilinmesi gerekebilir.

**Aday Anahtar (Candidate Key):** Bir ya da daha fazla kolondan meydana gelir. Tablonun her bir veri satırını eşsiz olarak tanımlar, başka bir deyişle tabloda kaç satır olursa olsun bu kombinasyonu bulunduran birden fazla satır asla olamaz. Örneğin, "Çalışan - Soyad" kombinasyonu bir aday anahtar değildir çünkü 1, 2 ve 3üncü satırlar ve 4, 5, 6, 7, 8 ve 9uncu satırlarda değerler tekrar etmektedir. Öte yandan "Çalışan - Semt" kombinasyonu hiç bir şekilde tekrar etmiyor. Öyleyse "Çalışan - Semt" kombinasyonu bir aday anahtardır.

**Eşsiz Anahtar (Primary Key):** Tablodaki aday anahtarlardan herhangi birini eşsiz anahtar olarak atayabiliriz. Bu anahtar tablodaki satırları tanımlamak için kullanılır ve bir tabloda yalnızca 1 tane eşsiz anahtar bulunabilir.

Tablomuza dönecek olursak, "Çalışan - Semt" kombinasyonunu eşsiz anahtar olarak atayabiliriz. Elimizdeki örnek son derece basit bir tablo olduğundan sorun yok ama daha karmaşık işlerde eşsiz anahtar için kolon kombinasyonlarını pek tercih etmiyoruz.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ana Tablo** |  |  |  |  |  |
| **Id** | **Calisan** | **Soyad** | **Sofor** | **Arac** | **Semt** |
| 1 | Orçun | Yılmaz | Ahmet | Toyota | Levent |
| 2 | Orçun | Yılmaz | Ahmet | Toyota | Etiler |
| 3 | Orçun | Yılmaz | Ahmet | Toyota | Ulus |
| 4 | Metin | Seyyar | Mehmet | Honda | Bakırköy |
| 5 | Metin | Seyyar | Mehmet | Honda | Ataköy |
| 6 | Metin | Seyyar | Mehmet | Honda | Yeşilköy |
| 7 | Metin | Seyyar | Tolga | Ford | Kandilli |
| 8 | Metin | Seyyar | Tolga | Ford | Beylerbeyi |
| 9 | Metin | Seyyar | Tolga | Ford | Kuzguncuk |

İşte şimdi oldu. "Id" sütununu eşsiz anahtar olarak atadık ve 2NF in 1. Maddesini uygulamış olduk.

Aslında üç ve dördüncü maddeler ikinci maddenin sonuçlarıdır. Eğer anahtar olmayan bir kolonla herhangi bir komposit anahtar arasında kısmi bağımlılık varsa her zaman tekrarlayan veri alt kümeleri oluşur. Bu durumu düzeltmek için bahis konusu alt kümeleri farklı bir tablo haline getirmeli ve elde ettiğimiz tablolar ile ana tablomuz arasındaki ilişkiyi tanımlamalıyız.

Tablomuzu bir gözden geçirelim: "Çalışan - Soyad" kombinasyonuna bakın. Çok tekrarlanıyor çünkü eşsiz anahtara verimli bir şekilde bağlayamamışız. Bunu düzeltmek için tablomuzu aşağıdaki gibi ikiye bölelim ve aralarında bir ilişki oluşturalım:

|  |  |  |
| --- | --- | --- |
| **Ana Tablo (Calisan Tablomuz oldu)** | | |
| **Id** | **Calisan** | **Soyad** |
| 1 | Orçun | Yılmaz |
| 2 | Metin | Seyyar |

|  |  |  |  |
| --- | --- | --- | --- |
| **Servis Tablosu** |  |  |  |
| **Cid** | **Sofor** | **Arac** | **Semt** |
| 1 | Ahmet | Toyota | Levent |
| 1 | Ahmet | Toyota | Etiler |
| 1 | Ahmet | Toyota | Ulus |
| 2 | Mehmet | Honda | Bakırköy |
| 2 | Mehmet | Honda | Ataköy |
| 2 | Mehmet | Honda | Yeşilköy |
| 2 | Tolga | Ford | Kandilli |
| 2 | Tolga | Ford | Beylerbeyi |
| 2 | Tolga | Ford | Kuzguncuk |

Yeni tablomuz ile ana tablomuzu ilişkilendirmek için "Cid" (Çalışan ID) isimli bir kolon yarattık. Dikkat ederseniz bu kolonun aldığı değer ana tablomuzdaki eşsiz anahtarı işaret ediyor. Bu alana Foreign Key diyoruz.

***3NF (3. Normal Form)***

Bir veri tabanının 3NF olabilmesi için aşağıdaki özellikleri karşılayabilmesi gerekir:

* Veri tabanı 2NF olmalıdır,
* Anahtar olmayan hiç bir kolon bir diğerine (anahtar olmayan başka bir kolona) bağıl olmamalı ya da geçişken fonksiyonel bir bağımlılığı (transitional functional dependency) olmamalıdır. Başka bir deyişle her kolon eşsiz anahtara tam bağımlı olmak zorundadır.

Veri tabanımızı 3NF şartlarına uydurabilmek için anahtar olmayan ve eşsiz anahtara tam bağımlı olmayan tüm kolonları kaldırmalıyız. Dikkat ederseniz bizim tablomuzda "Araç" kolonu eşsiz anahtarımıza değil "Şoför" kolonuna bağımlı. Birbirine bağlı olan bu iki kolonu (Şoför - Araç) ayrı bir tabloya ayırmamız ve tablomuzla aralarında bir ilişki yaratmamız gerekiyor.

|  |  |  |
| --- | --- | --- |
| **Ana Tablo** |  |  |
| **Id** | **Calisan** | **Soyad** |
| 1 | Orçun | Yılmaz |
| 2 | Metin | Seyyar |

|  |  |  |
| --- | --- | --- |
| **Servis Tablosu** |  |  |
| **Cid** | **Sid** | **Semt** |
| 1 | 1 | Levent |
| 1 | 1 | Etiler |
| 1 | 1 | Ulus |
| 2 | 2 | Bakırköy |
| 2 | 2 | Ataköy |
| 2 | 2 | Yeşilköy |
| 2 | 3 | Kandilli |
| 2 | 3 | Beylerbeyi |
| 2 | 3 | Kuzguncuk |

|  |  |  |
| --- | --- | --- |
| **Şoför Tablosu** |  |  |
| **Sid** | **Sofor** | **Arac** |
| 1 | Ahmet | Toyota |
| 2 | Mehmet | Honda |
| 3 | Tolga | Ford |

Öncelikle şoför tablosu adında yeni bir tablo oluşturduk. Bu tabloda Sid (Şoför ID) adıyla bir eşsiz anahtar yarattık ve Servis tablomuzdaki Sid kolonundan bu eşsiz anahtara referans vererek foreign key oluşturduk.

***İlişkiler***

Tabloların birbirleri ile haberleşmesi sırasında kullanılan 3 ayrı ilişki tipi vardır.

**Birebir İlişki:** Bu ilişki adından da anlaşılacağı gibi bir tablodaki kayıtların başka bir tablodaki eşleşen kayıt arasında ilişki vardır.

**Bire Çok İlişki:** Bu ilişki örnekle daha rahat açıklanabilir. Örneğin; bir makalenin birden çok yorumu olabilir, bir konunun birden çok alt konusu olabilir. Bu tekten çoğa giden bir ilişkidir.

**Çoka Çok İlişki:** Bu tür ilişkilerde her iki tarafında birden fazla kayıtla eşleşmesi söz konusudur. En iyi örneklerden biri Ürün ve Siparişler olabilir. Bir ürün birden fazla siparişte bulunabilir ve aynı şekilde bir siparişte birden fazla ürün bulunabilir.

***Eğitmene Not:***

Normalizasyon kuralları toplamda 6 adettir. Fakat ilk 3 tanesi yapıldığı takdirde database normalize edilmiş demektir(Buraya kadar olan kısım derste anlatılacak olan kısım). Diğer Normalizasyon kuralları aşağıda mevcuttur.

***BCNF / 3.5NF (Boyce-Codd Normal Form)***

Bir veri tabanının 3.5NF olabilmesi için aşağıdaki özellikleri karşılayabilmesi gerekir:

* Veri Tabanı 3NF olmalıdır,
* Her determinant (belirleyici kolon) aynı zamanda bir aday anahtar olmalıdır.

**Determinant:** Aynı satırdaki diğer kolon değerlerini belirlemek için kullanılan kolon kümesi determinant olarak adlandırılır.

Servis tablomuza dikkatle baktığımızda iki tane determinant olduğunu görebiliriz. Semt kolonu, Cid - Sid kombinasyonunun; Sid ise Cid kolonunun determinantıdır.

Bu noktada Semt kolonunun hali hazırda bir aday anahtar olduğunu görebiliyoruz çünkü her bir değer tekrar oluşturmaksızın tüm kayıt satırını tanımlayabilmekte. Öte yandan Sid için aynı şeyi söylemek mümkün değil çünkü tekrarlanıyor.

Elbette bu durumu düzeltmek için tabloyu ikiye ayıracak semt tablosunun değerini tabloları ilişkilendirmek için foreign key olarak kullanacağız.

|  |  |  |
| --- | --- | --- |
| **Ana Tablo** |  |  |
| **Id** | **Calisan** | **Soyad** |
| 1 | Orçun | Yılmaz |
| 2 | Metin | Seyyar |

|  |  |
| --- | --- |
| **Servis Tablosu** |  |
| **Cid** | **Semt** |
| 1 | Levent |
| 1 | Etiler |
| 1 | Ulus |
| 2 | Bakırköy |
| 2 | Ataköy |
| 2 | Yeşilköy |
| 2 | Kandilli |
| 2 | Beylerbeyi |
| 2 | Kuzguncuk |
|  |  |
| **Semt Tablosu** |  |
| **Semt** | **Sid** |
| Levent | 1 |
| Etiler | 1 |
| Ulus | 1 |
| Bakırköy | 2 |
| Ataköy | 2 |
| Yeşilköy | 2 |
| Kandilli | 3 |
| Beylerbeyi | 3 |
| Kuzguncuk | 3 |

|  |  |  |
| --- | --- | --- |
| **Şoför Tablosu** |  |  |
| **Sid** | **Sofor** | **Arac** |
| 1 | Ahmet | Toyota |
| 2 | Mehmet | Honda |
| 3 | Tolga | Ford |

***4NF (4. Normal Form)***

Bir veri tabanının 4NF olabilmesi için aşağıdaki özellikleri karşılayabilmesi gerekir:

* Veri Tabanı 3NF olmalıdır,
* Çok-değerli bağımlılıkları (Multli-Valued dependency) olmamalıdır.

***Özet:***

***Bir tabloda birincil anahtar alanlar ile anahtarı olmayan alanlar arasında, birden fazla bağımsız bire-çok ilişkisine izin verilmez.***

***Her bağımsız bire çok ilişki için ayrı bir tablo oluşturulması gerekir.***

**Multi-Valued Dependency:**

Bu durum bir ya da daha çok veri satırının var olması, aynı tabloda başka bir (ya da daha çok) veri satırının bulunmasını gerektirdiğinde ortaya çıkar. Örneğin, bir yazılım firması düşünelim. Geliştirdikleri yazılımların masaüstü bilgisayarlar için olanlarını tek-kullanıcılı ve çok-kullanıcılı olarak iki versiyonla piyasaya sunuyor olsunlar. Diyelim ki bu firmanın geliştirdiği tüm yazılımları barındıran bir veri tabanı oluşturuyoruz. Bu veri tabanında bir masaüstü yazılımın tek-kullanıcılı versiyonunu eklediysek mutlaka bir başka satırda aynı yazılımın çok-kullanıcılı versiyonu için de bir kayıt açılmış olmak durumundadır...

***5NF (5. Normal Form)***

Tekrarlamaları ortadan kaldırmak için her bir tablonun mümkün olduğunca küçük parçalara bölünmesi gerekir.

Aslında ilk 4 kural sonuçta bu işe yarar ancak, bu kurallar kapsamında olmayan tekrarlamalar da 5. normalizasyon kuralı ile giderilebilir.