Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi
Pamukkale University Journal of Engineering Sciences

ÇİFT ZAMANLI İLİŞKİSEL VERİTABANI YÖNETİMİ SİSTEMLERİ ÜZERİNE BİR UYGULAMA

AN IMPLEMANTATION OF BITEMPORAL RELATIONAL DATABASE MANAGEMENT SYSTEMS

Canan Eren ATAY[1]

[1]Bilgisayar Mühendisliği Bölümü, Mühendislik Fakültesi, Dokuz Eylül Üniversitesi, 35390, İzmir.
canan@cs.deu.edu.tr

Geliş Tarihi/Received: 03.05.2013, Kabul Tarihi/Accepted: 09.10.2013 doi: 10.5505/pajes.2014.25743

Özet

Bu makale birden fazla seviyede içiçe geçmişliği mümkün kılan çift zamanlı ilişkisel veri modelinin uygulanma araştırmasını içerir. Zamansal veriyi betimlemenin temel yapısı olan Çift Zamanlı Atom beş bölümden oluşur; olay zamanı periyodu, bu verinin veri tabanına işlenme zaman dönemlerinin alt ve üst sınırları ve yeni veri değeri. Çift zamanlı veri özniteliklerde bulunmaktadır ve soyut veri tipi olarak uygulanmıştır. İçiçe geçmiş çift zamanlı veri modeli yönetimi için örnekler, modelin özelliklerini göstermek için sorgulamalar sunulmuştur. Örnek uygulama modelin yapılabilirliğini göstermek için nesnesel-ilişkisel veri tabanı sisteminde tamamlanmıştır.

Anahtar kelimeler: Zamansal veri tabanı, Çift zamanlı veri tabanı, Sütun zaman etiketleme, Satır zaman etiketleme, Çift zamanlı sorgulama.

Abstract

This paper presents an implementation of a nested bitemporal relational data model that allows more than one level of nesting. The fundamental construct for representing temporal data is a Bitemporal Atom (BTA) that consists of five parts: valid time period, the time periods' lower and upper bounds this data is recorded in the database and a value. Bitemporal data is attached to attributes and applied as an abstract data type. A number of examples of the management of nested bitemporal data model and queries to demonstrate the features of the model are provided. A prototype implementation has been completed in an object-relational database system to demonstrate the feasibility of the model.

Keywords: Temporal database, Bitemporal database, Attribute time stamping, Tuple time stamping, Bitemporal query.

1 Giriş

Tüm bilgisayar uygulamaları zamana bağlı olarak değişiklik gösterir. Bu değişiklik genellikle depolanan verinin özelliklerinin değişimi sonucunda ortaya çıkan gereksinimler ile gerçekleşir. Örneğin, bir üniversitede çalışanların maaş, bölüm, kadro gibi bilgilerinden; öğrencilerin kayıt bilgileri, geçmişte aldıkları, halen almakta oldukları ve gelecekte alacakları dersler, bu derslerden aldıkları not ile öğrenim gördükleri bölüm bilgilerinden oluşan öznitelikler zaman içinde değişim gösterirler.

Özellikle finansal piyasalarda, işletmeler her bir müşterileri için zamana bağlı olarak nakit akışı veya hesap bakiye bilgilerini tutmak zorundadır. Borsa verileri, tanı konulmuş hastaların radyoloji ve laboratuvar sonuçları, havayollarında bilet rezervasyon bilgileri, oto kiralama ve otel rezervasyon verileri, veri ambarı verileri ve uzaysal veri tabanları yine zaman içinde değişen veriler olarak günlük yaşamda karşılaşılan örneklerdir. Zaman yönetimi desteği, veri tabanı uygulamalarının işlevselliğini arttırır. Bir veri tabanı sisteminin geçmişteki, bugünkü ve gelecekteki veri değerlerini barındırması tercih edilir. Böyle veri tabanlarına "zamansal veri tabanları" denir [1].

Zamansal veri desteği olmayan veri tabanları yalnızca en son değeri barındırır. Verinin eski değerleri, ya yeni değerler üzerine yazılır ya da tamamen silinir. Eski veriye ulaşmak gibi bir seçenek kalmaz. Ticari veri tabanları zamansal veri işleme yeteneğine sahip olmadığından, zaman desteğini sağlamak amacıyla her durum için o duruma özgü çözümler geliştirilmiştir. Ancak, geliştirilen özel çözümlerin uygulaması olmadan, gerçek dünya olaylarının zaman içinde gelişimi veri tabanlarında saklanamamakta ve yalnızca veri tabanındaki mevcut duruma ait sorgulamalara yanıt verile bilinmektedir.

Zamansal veri tabanı sistemleri nesnelerin ya geçmiş değerlerini (olay zamanı) ya da veri tabanına yazılma geçmişlerini (işlem zamanını) saklayabilmesine karşın, çift zamanlı veri tabanı sistemleri, birbirinden bağımsız olan, olay zamanı ve işlem zamanının her ikisini birden desteklemektedir [2]. Zamansal veri tabanı sistemlerinde yalnızca olay zamanı'nın saklanması durumunda, nesnelerin geçmiş değerleri saklanmakta, ancak geriye ve ileriye dönük değişikliklerin değişim aşamaları saklanamamaktadır. Yalnızca işlem zamanı'nın saklanması durumunda ise veri değerlerinin geçerlilik süresi gibi geçmiş veya gelecek değerlerin barındırabilmesi söz konusu olamamaktadır.

Çift zamanlı veri tabanları, değişen dünyamızın değişen bilgilerini modelleyerek veri değerlerini eylemler ile birleştirir ve ayrıca eylemlerin ne zaman geçerli olduğunu belirtir. Dolayısıyla veri değerlerinin geçmişini ve değişimlerinin bütün dökümünü sağlar. Gerçek dünyayı eksiksiz ve düzgün modelleyebilmek için her iki zaman boyutu da gereklidir. Özellikle denetlemenin önemli olduğu (vergi, finans ve sigorta uygulamaları vb.) pek çok uygulama alanı, hem olay hem de işlem zamanının desteklenmesinden yararlanır.

Bu makalede olay ve işlem zamanlarının sütunlara iliştirildiği, varlıkların geçmişleri ve aralarındaki ilişkileri göstermek için birden fazla seviyede içiçe geçmişliğin izin verildiği [3]'de önerilen İçiçe Geçmiş Çift Zamanlı İlişkisel Model (NBRM)'in uygulamasını sunulacaktır. NBRM soyut veri tipleri ve içiçe geçmiş ilişkileri [4] destekleyen, aynı zamanda [5] SQL99'da bulunan, nesnesel ilişkisel veri tabanı yönetim sistemleri

üzerine kurulmuştur. Bu çalışma çift zamanlı sütun zaman etiketleme ve nesnesel ilişkisel veri tabanı yönetim sisteminin birlikte kullanılmasına örnek oluşturacaktır.

Son 30 yılda zamansal veri tabanlarına ait geniş bir araştırma çabası olmasına rağmen, hiçbir standart zamansal veri tabanı modeli veya sorgu dili ISO gibi büyük standardizasyon kuruluşu tarafından tanımlanmamıştır. Zamansal veri yönetiminin pek çok uygulama alanında ve bilgi keşif sistemlerinde tümlevsel bir parça olacağı öngörülmektedir. Bu araştırmada önerilen model uygulaması, çift zamanlı veri tabanlarının ihtiyacı olan gerekli yapıları desteklediğinden, pek çok uygulama alanında çift zamanlı veri tabanlarının yapılabilirliğini kanıtlamak için sınama ortamı olarak kullanılabilecektir.

Makalenin kalanı şu şekildedir. Bir sonraki bölüm çift zamanlı veri tabanları hakkında yapılan çalışmaları sunar. 3. bölüm zamansal veri tabanı için gerekli tanımları açıklar. Satır ve sütun zaman etiketleme yöntemleri 4. bölümde anlatılırken, 5. bölüm çift zamanlı veri tabanı uygulaması ve sorgulamaları içerir. 6. Bölüm makalenin sonuçlarını ve ileride yapılabilecek araştırmalarını kapsar.

2 Literatüre Taraması

Bilgi sistemlerinin zamansal yönlerine odaklı önemli araştırmalar yapılmıştır [6, 7]. İlişkisel veri modelinin geliştirilmesinde yaygın iki genel yaklaşım vardır: Satır Zaman Etiketleme ve Sütun Zaman Etiketleme. Satır zaman etiketleme yaklaşımı, birinci normal formdaki (1NF) tabloya iki yeni zaman sütunu (zaman aralığının başlangıç ve bitiş noktalarını) ilave eder [8, 9]. Bu yaklaşımda geleneksel ilişkisel veri tabanlarının bütün avantajları vardır. Buna rağmen gereksiz veri tekrarı kaçınılmazdır. Birinci normal form'da (N1NF) [10] olmayan veri tabanlarında kullanılan sütun zaman etiketleme yaklaşımı daha karmaşık olmasına karşın veri tekrarlanmasını engeller, nesneye ait bütün geçmişi pek çok satıra bölmektense bir tek satırda tutarak [11, 12, 13, 14, 15, 16] satır zaman etiketleme yönteminde olan yatay ve dikey veri tekrarını engeller [11].

Ben-Zvi, çift zamanlı veri tabanları için ilk veri modelini, zamansal sorgulama dilini, depolama mimarisini, indeksleme, kurtarma, eşzamanlılık, eşleme ve uygulanmasını önermiştir [8]. Snodgrass her bir zamanla değişen tablo için dört sütun eklemiş [9] ve TQuel zamansal sorgulama dilini sunmuştur. Gadia ve Bhargava, olay ve işlem zamanı olarak iki boyutlu zamansal bileşenleri sütunlara eklemişlerdir. Geliştirdikleri bu model, güncelleştirmeler ve hatalar için geniş kapsamlı biçimcilik olarak tasarlanmıştır. Modellerine özgü özel anlamları ve tanımladıkları farklı yapıdaki kullanıcıları göz önündeki bulundurarak, güncellenmiş veriye ulaşmak ve tabloları sorgulamak için yeni işlemler tanımlamışlardır [13].

SpyTime, [17] casusların şehirlerdeki hareketlerini raporlayan satır zaman etiketleme yöntemine dayalı çift zamanlı veri tabanıdır. SpyTime zamansal veri tabanlarında ölçüt olarak kullanılacak sorgu örnekleri bulunan ilk veri tabanı modelidir. T4SQL Combi, Montanari ve Pozzi tarafından satır zaman etiketleme yöntemine dayalı çok boyutlu zamansal ilişkiler üzerinde çalışan yeni bir sorgulama dili olarak önerilmiştir [18]. Kullanıcı zamansal boyutlardan olay zamanı, işlem zamanı, olay anı gibi farklı zamansal boyutların anlamlarına göre zamansal ilişkileri sorgulayabilir. T4SQL sorguları eşdeğer SQL sorgulara çevrilebilir fakat karşılık SQL sorgusu daha karmaşık, boyutu daha büyük ve çalışması genellikle daha verimsizdir.

İlişkisel veya nesnesel ilişkisel veri tabanı yönetim sistemleri üzerinde yapılan başka zamansal veri tabanı uygulamaları vardır ve bunların bir kısmı yeni sistem uygulamalarıyla Dumas, Fauvet ve Scholl'un [16] çalışmasında bulunmaktadır. Bu uygulamalar genellikle satır zaman etiketleme yöntemini kullanırken, Chau ve Chittayasothorn zamansal bileşenler ve sütun zaman etiketleme yöntemini kullanmıştır [15], fakat yalnızca olay zamanı modellenmiştir. Yang, Ying ve Widom'un nesnesel ilişkisel veri tabanı sisteminin genişletilmesiyle zamansal destek sağlayan çalışması [19]'da bulunmaktadır.

XML (Genişletilebilir İşaretleme Dili) dili, Web üzerinden yapılanma ve veri alışverişi için standart olarak ortaya çıkmıştır. XML, bir Web sayfasında görüntülenen bazı bileşenlerin yapısı ve verilerin anlamı hakkında bilgi sağlamak için kullanılabilir. Bunlara ek olarak XQuery ile karmaşık sorgulamalar yazılabilir. XML bilgi işlem sistemleri tarafından kolayca okunabilecek dokümanlar oluşturmaya yaradığından, hiyerarşik veri depolama yanında farklı sistemler arasında veri alışverişi yapmaya da elverişli yeni veri tabanı modelidir. XML'in hiyerarşik veri yapısı ile sağladığı destek zamansal gruplandırılmış [20] veya sütun zaman etiketleme yapısına benzemektedir. Zamansal ve çift zamanlı atomik veri yapısı, dolayısı ile zamansal veri tabanları sütun zaman etiketleme kullanarak XML ile gösterilebilir. Wang ve Zaniolo olay zamanı, işlem zamanı veya çift zamanlı zamansal veri tabanlarının geçerli standartlara uzantılar gerektirmeden XML ile belirtileceğini ve XQuery ile sorgulanacağını göstermiştir [21]. Ayrıca, satır zaman etiketleme yöntemini desteklemek için XML, karmaşık zamansal sorguları ifade etmek için XQuery kullanan, ilişkisel veri tabanlarında geçmişe yönelik gerçek verileri yönetebilmek için sıralama ve zamansal kümeleme teknikleri kullanan ArchIS sistemi sunulmuştur [22]. Literatürde bulunan farklı zamansal XML veri modellerinin karşılaştırılması [23]'de bulunmaktadır.

3 Zamansal Veri Modellenmesi

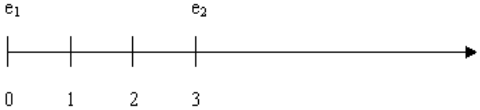
3.1 Zamanın Modellenmesi

Zamanın modellenmesinde kullanılan kavramlar genel olarak Atom, U ve T kümeleridir. Burada; **Atom** özyineli tanımların kurulduğu tanımlanmamış en temel terimdir. **U** tam sayılar, doğal sayılar, tek karakterler ve boş (null) gibi atomik değerlerden oluşan evrensel bir kümedir. **T** zamanı betimleyen ve U kümesinin alt kümesi olan kümedir. Bu küme küçüktür veya eşittir (≤) ilişkisi altında ayrık, doğrusal sıralanmıştır ve doğal sayılara eş yapılıdır.

Zaman değerlerinin "0, 1, 2, …, *şimdi*" olarak tam sayılardaki gibi sıralandığı kabul edilmektedir. Burada, "0" sembolü zamanın başlangıcını betimler ve "*şimdi*"de bulunduğumuz andaki zamanı simgeleyen özel değişkendir.

3.1.1. Zamansal Veri

Zaman anı zaman ekseninde bir zaman noktasıdır ve olay gerçek dünyadaki bazı nesnelerin durumlarını değiştirmeye sebep olabilecek eylemlerle sonuçlanan zamanda ayrıştırılmış andır. Olay anlıktır ve durum ise zamanla değişendir. Şekil 1'de olay e1 zaman anı 0'da olmuştur ve zaman anı 3'e kadar geçerli olan veri değerlerini oluşturmuştur. Olay e2, zaman anı 3'te olup e1'e ait bilgilerin yerine yazılmıştır.



Şekil 1: Zaman noktaları ve olaylar.

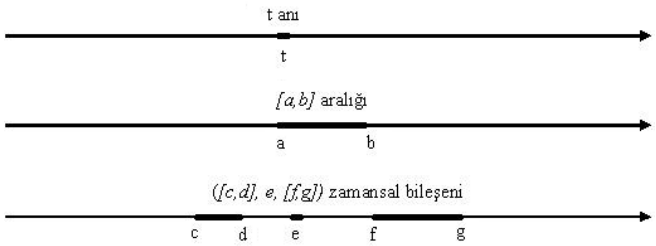
3.1.2 Zaman Aralıkları

Zaman aralığı iki olay arasındaki zamandır. Birbirini izleyen zaman birimleri kümesi, t_a başlangıç zaman anı (alt sınır) ve $t_ü$ bitiş zaman anı (üst sınır) [t_a , $t_ü$] olarak belirtilebilir. Zaman aralıkları T'nin doğal alt kümesidir. Zamansal küme herhangi zaman noktalarının kümesidir. Süreli zamansal küme, ya kapalı aralık [t_i, t_{i+n}] ya da yarı-açık aralık [t_i, t_{i+n+1}) olarak gösterilen, birbirini izleyen zaman noktalarını {t_i, t_{i+1}, …, t_{i+n}} içerir [24]. Küme işlemleri kullanılarak birleşme, kesişim ve fark, aralıklar için Allen tarafından tanımlanmıştır [25]. Fakat T'deki aralıklar kümesi bu işlemler için kapalı değildir.

3.1.3 Zamansal Bileşenler

Zamansal bileşen, ayrışık zaman aralıklarının sonlu birleşimidir. Örneğin, {[t_1, t_2) ∪ [t_5, t_6] ∪ [t_{12}, t_{13}]} zamansal bileşendir. Mantıksal cebir oluşturmak üzere zaman anları kümesi T'nin en büyük elemanı ve ∅ 'nin en küçük elemanı olarak zamansal bileşenler kümesi birleşim, kesişim ve bütünlük küme işlemleri için kapalıdır. Zaman noktaları, aralıkları ve zamansal bileşenlerin zamansal modelleme ve zamansal veri sorgulama için ne kadar gerekli olduğunun göstergesidir.

Örnek 1: Şekil 2 zaman noktası, zaman aralığı ve zamansal bileşenini gösterir.



Şekil 2: Zaman ekseninde zaman noktası, zaman aralığı ve zamansal bileşen.

3.2 Zamansal Verinin Gösterimi

Zamansal veri, verinin zamanla ilgili bilgisinin ilişkilendirilerek tanımlanması demektir. Zaman etiketi zaman değerinin (zaman noktası, zaman aralığı veya zamansal bileşeni) veri değeri ile ilişkilendirilmesidir. Zamanın birbirinden bağımsız olabilecek farklı kavramları vardır. Zamansal gerçekçiliğin farklı taraflarını yakalayabilmek için zaman boyutuna bağlı olarak veri değerleri farklı anlamlar taşıyabileceğinden, zamansal veri tabanları için birden fazla zaman boyutu önerilmiştir.

3.2.1 Olay Zamanı

Olay zamanı, modellenmiş ortamda eylemin (veri değerinin) ne zaman doğru olduğunu/olacağını gösterir. Doğruluk süresi geçmişte, şimdi veya belki de gelecekte olabilir. Zaman etiketleri veriyi eklerken veya değiştirirken kullanıcı tarafından verilir veya kullanıcı işlemleri ile belirlenir.

3.2.2 İşlem Zamanı

İşlem zamanı, değerlerin veri tabanına kayıtlanma zamanı olarak belirtilir. Veri tabanı yönetim sistemi işlem zamanını belirtir ve şimdiki zamandan daha sonra ki bir değer olamaz. Sistem tarafından oluşturulan değerler veri tabanı evreleri değiştikçe tekdüze bir şekilde artar. Olay ve işlem zamanları birbirinden bağımsız zaman eksenleridir.

3.2.3 Kullanıcı-Tanımlanmış Zaman

Tablodaki veri tipi "zaman" olan sütun, veri tabanı yönetim sistemi tarafından yorumlanmaz. Bu tür öznitelik *kullanıcı-tanımlanmış zaman* [2] olarak adlandırılır çünkü veri tabanı yönetim sistemi bu zamansal veriyi herhangi bir veri gibi değerlendirir. Doğum-Tarihi özniteliğindeki değer kullanıcı-tanımlanmış zamana bir örnektir. Zaman verisi değerleri, kullanıcının belirlediği ve gerekirse yenilendiği için yalnızca kullanıcıya anlamlıdır. DATE, TIME ve DATE-TIME gibi veri tipleri kullanıcı-tanımlanmış zamanın alanı olarak kullanılır.

3.3 Zamansal Veri Tabanları

Zamansal veri tabanları modellenen nesneye ait geçmişi saklamak için zamanın farklı yönlerini yakalar. Snodgrass ve Ahn, [2] tarafından sunulan zamansal veri tabanları gruplandırılmasına göre, olay ve işlem zamanına bağlı olarak dört zamansal veri tabanı sınıfı tanımlanmıştır. Bunlar anlık (şipşak), tarihsel, geri çağırma (işlemsel) ve çift zamanlı veri tabanlarıdır.

3.3.1 Anlık Veri Tabanları

Günümüzde kullanılan veri tabanları değişmekte olan dünyayı modeller. Bu veri tabanları işletmenin belirli bir zamandaki durumunu (genellikle de şimdiki durumunu) belirler. Yeni veri değerleri eskileri ile değiştirildiğinde, geçmişteki durumu belirleyen veriler yok olur. Bu çeşit veri tabanlarına anlık veri tabanları denir, çünkü gerçeğin yalnızca anlık halini alırlar. Anlık veri tabanları yalnızca kullanıcı tarafından tanımlanmış veri tipini destekler.

3.3.2 Tarihsel Veri Tabanları

Tarihsel veri tabanları veri tabanı durumlarını olay zamanı ekseninde saklar. Tarihsel veri tabanlarının geçmişteki değerleri gerçek dünyaya göre değişir, geçmişi şimdi olarak bilinen değere göre saklar. Veri tabanı durumları olay zamanı ile geçmiş, şimdi veya gelecek olarak saklanır. Veri tabanı kullanıcısı olay zamanı değerlerini sağlar. Bir hata bulunursa, bu hatayı düzeltmek için yenileme işlemi gerekmektedir bu da veri tabanının durumunun değişmesine neden olur. Bir önceki değer silinmiştir ve hatalı değer kayıp olmuştur. Bundan dolayı veri tabanı geçmişteki gibi görülemez. Bu çerçevede tarihsel veri tabanları anlık veri tabanlarına benzer.

3.3.3 Geri Çağırma (İşlemsel) Veri Tabanları

Geri çağırma (işlemsel) veri tabanları, verileri işlem zamanı ekseninde veya işlem zamanıyla sıralanmış anlık veri tabanı olarak saklar. Veri tabanını geçmişteki bir zamandaki duruma getirmek, geri çağırma veri tabanları ile mümkündür. İşlemsel zaman sistem tarafından sağlandığı için geri çağırma veri tabanları ileri veri tabanı durumlarını saklayamaz.

3.3.4 Çift Zamanlı Veri Tabanları

Tarihsel veri tabanları yalnızca nesnenin tarihçesini modeller, sistemdeki değişiklikleri saklamaz. Geri çağırma veri tabanları da modellenen nesnedeki gerçek değişimleri belirlemez. Eğer gerçek doğru bir şekilde modellenecek ise, hem tarihsel hem de geri çağırma veri tabanları birleştirilmeli ve sistem hem olay hem de işlemsel zamanı desteklemelidir. Çift zamanlı veri tabanı sistemleri hem olay hem de işlem zamanını destekler. Bunu yaparak, sistem aslında değişen gerçek dünyadaki değişen bilgileri modeller. Bundan dolayı, değerleri olaylar ile ilişkilendirir ve bu olayların ne zaman veri tabanına yazıldığını belirtir. Yani, sistem geçmişe ve geleceğe yönelik değişikliklerin saklanmasına izin verir.

4 Zaman Etiketleme Yöntemleri

Bu bölümde zaman aralığı kavramını kullanarak, ilişkisel veri tabanı modelinde zamansal etiketleme yöntemleri açıklanacaktır, fakat aynı yöntem nesneye yönelik veya öğeler arası ilişkisel veri modelleri gibi diğer modellere genişletilebilir. Önerilen zamansal ilişkisel veri tabanı modelleri pek çok açıdan farklı olsa bile, zaman etiketinin nereye iliştirildiğine göre genelde iki yaklaşım vardır: Birincil normal şekil (1NF) ilişkileri kullanan *satır zaman-etiketleme*; Birincil olmayan normal şekil (N1NF) ilişkileri kullanan *sütun zaman-etiketleme.* Satır ve sütun zaman-etiketleme arasındaki farklar, Clifford, Croker ve Tuzhilin tarafından *zamansal gruplanmamış* ve *gruplanmış* veri modelleri olarak tanımlanmıştır [20].

4.1 Satır Zaman-Etiketleme Yöntemi

Satır zaman-etiketleme yaklaşımındaki zamansal veri modelleri (1NF) birincil normal şekildedir ve tablodaki her bir satıra zaman etiketi ilave edilir. Yenileme işlemi veri tekrarı ile sonuçlanan yeni satır eklemeyi gerektirir. Birden fazla zamansal sütun olması durumunda, her bir sütundaki yeni değerler veri tekrarını önemli şekilde arttırır. Zamansal sütunları farklı tablolara ayrıştırmak, satır zaman-etiketleme yönteminde veri tekrarını azaltır fakat pek çok küçük tablolarla sonuçlanır.

Tablo 1: Zaman aralığı ile ÜCRET sütununun satır zaman-etiketlenmesi.

Numara	İsim	Ücret	Başlangıç	Bitiş
101	Ahmet	2500	01.06.2003	01.12.2005
101	Ahmet	3000	02.12.2005	*Şimdi*
102	Ayşe	3500	01.06.2011	*Şimdi*
103	Ali	3500	01.06.2003	01.06.2004
103	Ali	4200	02.06.2004	31.12.2004
103	Ali	4500	01.01.2006	*Şimdi*

Zaman aralığı ile satır zaman-etiketleme yönteminde, satırların zaman referansını gösteren iki tane zaman sütunu 1NF tabloya ilave edilir. Bunlardan birincisi (BAŞLANGIÇ) öznitelik değerinin gerçekleştiği anı, ikincisi (BİTİŞ) güncelleme işlemi sonucunda, öznitelik değerinin yeni bir değer ile değiştirileceği zamanı belirtir.

Örnek 2: Tablo 1'deki ÇALIŞAN tablosu olay zamanı aralığının başlangıç ve bitiş zaman anlarını saklamak için iki zaman sütunu ile genişletilmiştir.

4.2 Sütun Zaman-Etiketleme Yöntemi

Sütun zaman-etiketleme yönteminde zaman etiketi (zaman noktası, aralığı veya zamansal bileşen olmasından bağımsız olarak) değeri zamansal atom olan sütunlara eklenir. Atom, değerlerini U kümesinden alır. *Zamansal atom*, *z*'nin zamanı, *d*'nin de atomik değeri belirttiği <*z, d*> çiftidir. Zamansal atom ***d*** değerinin ***z*** zamanı boyunca doğru olduğunu belirtir. Değerlerin tarihçesi her bir öznitelik için ayrı ayrı ve nesnenin bütün tarihçesi tek bir sırada saklanmış olur. Bundan dolayı, yenilenmeden etkilenmeyen sıra değerlerin tekrarlanması gerekmez. Bu yaklaşım genel olarak bu alandaki araştırmacılar tarafından kabul edilen zamansal verinin üç-boyutlu görünümünü betimler.

Sütun zaman-etiketleme, N1NF ilişkileri destekleyen veri modeli üzerine kurulmasını gerektirir, çünkü zaman etiketlenmiş sütunlar öznitelik değeri ile birlikte zaman etiketini de saklayabilir. Zamanla değişen bütün öznitelikler tek bir tabloya dahil edilir. Bu tablo zamanla değişen özniteliklerle birlikte zamanla değişmeyenleri de kapsayabilir.

Zaman aralığı ile sütun zaman-etiketleme yönteminde, zamansal atom'un zaman etiketi bölümü değerin doğru olduğu aralıktır, örneğin < [t_a, $t_ü$], d>'de aralığın başlangıç ve bitiş sınırları t_a ve $t_ü$ olmaktadır.

Örnek 3: Tablo 2'deki zaman aralıklarının kullanıldığı ÇALIŞAN tablosunu ele alalım. Ahmet'in ücret ve bölüm geçmiş bilgileri aynı sıraya dâhil edilmiştir. Satır zaman-etiketlemenin aksine, Numara ve İsim sütunlarındaki değerler tekrar edilmemiştir.

Tablo 2: Zaman aralığı ile sütun zaman-etiketleme.

Numara	İsim	Ücret	Bölüm adı
101	Ahmet	{<[01.06.2003, 01.12.2005), 2500>, <[02.12.2005, *şimdi*], 3000>}	{<[01.06.2003, 01.12.2006), Satış>, <[02.12.2006, *şimdi*], Pazarlama>}
102	Ayşe	<[01.06.2011, *şimdi*], 3500>	<[01.06.2011, *şimdi*], Satış>
103	Ali	{<[01.06.2003, 01.06.2004), 3500>, <[02.06.2004, 31.12.2004), 4200>, <[01.01.2006, *şimdi*], 4500>}	{<[01.06.2003, 31.12.2004), Oyuncak>, <[01.01.2006, 01.04.2006), Pazarlama >, <[02.04.2006, *şimdi*], Oyuncak>}

Tablo 3: Çift Zamanlı İlişkisel Modeli kullanarak tanımlanan tablo örneği.

Numara	İsim-B	Adres-B	Doğum Tarihi	Bölüm		Ücret-B
	İsim	Adres		Bölüm_Adı-B	Şef-B	Ücret
				Bölüm_Adı	Şef	

5 Çift Zamanlı İlişkisel Veri Tabanı Uygulaması

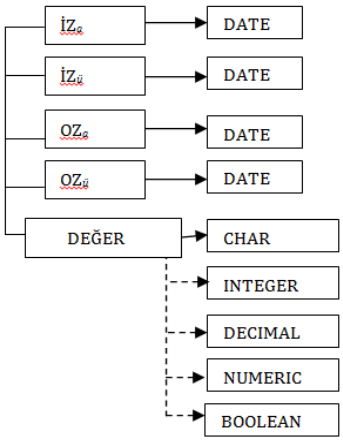
Tansel ve Atay, [3] tarafından sunulan İçiçe Geçmiş Çift Zamanlı İlişkisel Model'in uygulamasının yapılabilmesi amacıyla, Tablo 3'de gösterildiği gibi bir yapı oluşturulmuş ve bu yapının oluşturulma aşamaları, sorguların ve yenilenme işlemlerinin yapılması sırasında kullanılan yapılar ayrıntılı olarak belirtilmiştir.

5.1 Çift Zamanlı Atom Tipi

Çift zamanlı atom (<işlem zamanı, olay zamanı, değer>) olarak gösterilen üçlüdür. İşlem ve olay zamanları için zaman noktası, aralığı veya zamansal bileşen kullanılabilir. Eğer zaman aralığı kullanılırsa, <[$İZ_a$, $İZ_ü$), [OZ_a, $OZ_ü$), D> formundaki çift zamanlı atomda, $İZ_a$ işlem zamanı alt sınırı, $İZ_ü$ işlem zamanı üst sınırı, OZ_a olay zamanı alt sınırı, $OZ_ü$ olay zamanı üst sınırı ve D'de veri değeri olur. [$İZ_a$, *şimdi*] veya [OZ_a, *şimdi*] gibi üst sınırı

şimdi olan aralık, şimdi'nin değeri zaman ilerledikçe değişeceğinden, kapalı ve genişleyen bir aralıktır.

Örnek uygulamada, çift zamanlı atomun işlem ve olay zamanı alt ve üst sınırları için veri tipi DATE (GG.AA.YYYY formatında) kullanıldı. Uygulama alanına bağlı olarak, zaman aralığı için TIMESTAMP gibi farklı bir zaman birimi de kullanılabilir. Değer alanı CHARACTER, CHARACTER VARYING, CHARACTER LARGE OBJECT, NUMERIC, DECIMAL, INTEGER, SMALLINT, BIGINT veya BOOLEAN olabilir. Şekil 3'de görüldüğü gibi çift zamanlı atom'u (Bitemporal Atom, BTA) soyut veri tipi olarak kullanacağız.



Şekil 3: Çift zamanlı atom'un (BTA) soyut veri tipi ile betimlenmesi.

Nesnesel-ilişkisel veri tabanı sistemlerinin yeni veri tanımlama kullanılmasına olanak sağlamasıyla, Şekil 4'te görüldüğü gibi çift zamanlı zamansal atom yapılı soyut veri tipi olarak tanımlanmıştır. İşlem zamanı alt ve üst sınırları gibi bileşenlerin bulunup getirilmesine veya değiştilmesine olanak sağlanmıştır ve bunlar sorgulamalarda kullanılır. Soyut veri tipi bir kez tanımlandığında, diğer yerleşik veri tiplerinin kullanıldığı SQL komutlarında kullanılabilir.

```
CREATE TYPE BTA AS (
        İŞLEM_ZMN_ALT_SINIR DATE,
        İŞLEM_ZMN_ÜST_SINIR DATE,
        OLAY_ZMN_ALT_SINIR DATE,
        OLAY_ZMN_ÜST_SINIR DATE,
        DEĞER CHARACTER VARYING(50));
```

Şekil 4: Çift zamanlı atom'un soyut veri tipi ile tanımlanması.

Soyut veri tipleri bütün tablonun "veri tipi" olarak bildirilir, böylece tablo'nun sütunları bu soyut veri tipi ile tanımlanır. Ekleme, silme ve çekme gibi verilerin değişiklik işlemleri normal tablolardaki gibi uygulanabilir. Şekil 5'deki SQL komutu BTA_ADRES'in sütun tipi BTA olan çift zamanlı tablo olduğunu gösterir. Her bir tablo çift zamanlı atomları betimlemek için gerektiği kadar sıradan oluşabilir.

```
CREATE TYPE BTA_ADRES AS TABLE OF BTA;
```

Şekil 5: BTA_ADRES'in sıralarının içiçe geçmiş tablo olarak tanımlanması.

5.2 İçiçe Geçmiş Çift Zamanlı İlişkisel Tablo

İçiçe geçmiş tablo aynı tipte elemanların sıralanmamış yığınıdır. Eleman sayısı herhangi bir sayı olabilir, tablonun tanımında maksimum sayı belirtilmez. İçiçe geçmiş tablodaki elemanlar aslında bağlı bulunduğu tabloyu belirleyen bir özniteliği içeren farklı bir tabloda saklanır.

İçiçe geçmiş çift zamanlı tablodaki bir sıra, tablonun tanımlandığı yapılandırılmış bir veri tipinin somutlanan örneğidir. Bu sıra somutlanmış örneğe tek özdeşlik verir. Tek bir sıra içinde birbirine özdeş soyut veri tipi kümesinin varlığı aslında (her bir nesnenin zamansal özniteliğinin bir sütun ile) sütun zaman etiketleme yaklaşımın benzetilmemesidir. Bunlar zamansal gruplanmış tablolardır. Örneğin, ADRES tablo tipi BTA_ADRES çift zamanlı atomların içiçe geçmiş çift zamanlı tablo ADRES_TABLE'da olası kullanımını sağlamak için tanımlanmıştır. Şekil 6 ÇALIŞAN tablosunun içiçe geçmiş veri kümesi tablolarla tanımını verir.

```
CREATE TABLE ÇALIŞAN (
  NUMARA          NUMBER Primary Key,
  İSİM            BTA_İSİM,
  ADRES           BTA_ADRES,
  DOĞUMTARİHİ     DATE,
  BÖLÜM           BÖLÜM_ŞEF,
  ÜCRET           BTA_ÜCRET
)
NESTED TABLE İSİM STORE AS İSİM_TABLE,
NESTED TABLE ADRES STORE AS ADRES_TABLE,
NESTED TABLE BÖLÜM_ŞEF STORE AS BÖLÜM_TABLE
  (NESTED TABLE BÖLÜM_GEÇMİŞİ STORE AS
BÖLÜM_TABLE,
    NESTED TABLE ŞEF_GEÇMİŞİ STORE AS ŞEF_TABLE),
NESTED TABLE ÜCRET STORE AS ÜCRET_TABLE;
```

Şekil 6: ÇALIŞAN tablosunun tanımlanması.

5.3 Çift Zamanlı İlişkisel Veri Tabanı Değişiklik İşlemleri

Tablo 3'deki ÇALIŞAN tablosunun altı tane sütunu vardır. Veri tipi tam sayı (INTEGER) ve birincil anahtar olan NUMARA, veri tipi DATE olan DOĞUM TARİHİ zamansal olmayan iki sütundur. İSİM, ADRES, ÜCRET zamansal sütunlardır ve içiçe geçmiş soyut veri tipi kümesini çift zamanlı atom ile kullanırlar. BÖLÜM iki adet sütunu olan (BÖLÜM_ADI ve ŞEF) içiçe geçmiş tablodur. BÖLÜM_ADI çalışanın bağlı bulunduğu bölüm adını, ŞEF çalışanın yöneticisi bilgilerini ikinci seviyede içiçe geçmiş tablo kümesinde çift zamanlı atom olarak tutar.

5.3.1 Son_değer ve Sysdate

son_değer 'şimdi' ve/veya sonsuz üst limiti betimleyen özel bir sabittir ('31.12.9999' kullanılmıştır). Bu diğer zamansal/çift zamanlı uygulamalarda da yaygın olarak kullanılan bir değerdir. Sysdate şimdiki zamanı veren bir SQL fonksiyonudur. Bu uygulamada eğer özel bir zaman belirtilmemiş ise hem olay hem de işlem zamanı üst limitlerinin varsayılan değeri 'şimdi'dir.

5.3.2 Yeni Veri Ekleme

Şekil 7 ÇALIŞAN tablosuna çalışan numarası 20001, adı 'HAKAN ÖZTÜRK', doğum tarihi '10.3.1980', adresi '1219 Sokak Menekşe Apt. No: 3 D: 9 Bostanlı İZMİR', bölümü DEP_ID23 olan bilgilerin ekleme işlemini gösterir. Şefi 'ALİ CAN YILMAZ', ücreti 2500 ve işe başlama tarihi 01.01.2007 olarak belirlenmiştir.

5.3.3 Veri Güncellenmesi

Güncelleme işlemi eski bilgiyi korurken yeni bir çift zamanlı atom "ekler". Örneğin, Şekil 8 çalışan numarası 12345 olan çalışanın ücretinin 1 Şubat 2007'den itibaren geçerli olmak üzere 5000 TL olarak yenilendiğini gösteriyor. Eğer yenilenen

veride bir hata bulunursa, bu hatayı düzeltmek için farklı bir yenileme işleminin gerçektirilmesi gerekmektedir. Hatalı veri tutulur, doğru değer ve olay zamanı doğru tarih ile yenilenir.

```
INSERT INTO ÇALIŞAN VALUES
(20001,
İSİM(BTA(sysdate, son_değer, '01.01.2007', son_değer,
'HAKAN ÖZTÜRK')),
ADRES(BTA(sysdate, son_değer, '01.01.2007', son_değer,
     '1219 Sk Menekşe Apt. No:3 D:9 Bostanlı İZMİR
35120')),
'03.10.1980',
BÖLÜM_ŞEF(TYPE_BÖLÜM_ŞEF(
BÖLÜM_ADI(BTA(sysdate, son_değer, '01.01.2007',
son_değer, 'DEPID_23')),
ŞEF_ADI(BTA(sysdate, son_değer, '01.01.2007', son_değer,
'ALİ CAN YILMAZ')))),
ÜCRET(BTA(sysdate, son_değer, '01.01.2007', son_değer,
2500)));
```

Şekil 7: SQL dilinde BTA ile yeni bir veri eklenmesi.

```
UPDATE TABLE ( SELECT Ç.ÜCRET
                    FROM  ÇALIŞAN Ç
                    WHERE  Ç.NUMARA=12345) ÜCR
SET ÜCR.OLAY_ZMN_ÜST_SINIR = '01.02.2007',
  ÜCR.İŞLEM_ZMN_ÜST_SINIR = sysdate
WHERE ÜCR.OLAY_ZMN_ÜST_SINIR = son_değer;

INSERT INTO THE (SELECT Ç.ÜCRET
                    FROM  ÇALIŞAN Ç
                    WHERE  Ç.NUMARA=12345)
VALUES(BTA(sysdate, son_değer, '01.02.2007', son_değer,
5000));
```

Şekil 8: SQL dilinde BTA tipi ile ÜCRET sütun 'unundaki değerin yenilenmesi.

5.3.4 Veri Silinmesi

Zamansal/çift zamanlı veri tabanlarından kayıtlı bir sıra pek çok nedenden dolayı asla silinmez. Uygulama örneği şirket veri tabanı olduğundan, eğer bir çalışan şirketten ayrılırsa bilgileri veri tabanından silinmez. Çift zamanlı özniteliklerin olay zamanı üst limitine işten ayrılacağı tarih, işlem zamanı üst limitine de sistem zamanı yazılır. Her iki zamanın da üst limitleri 'şimdi' olmadığından, bu sıradaki bilgiler artık "geçmiş zaman"ki bilgiler olarak veri tabanında saklanır. Şekil 9'daki kod, numarası 13456 olan çalışanın 01.15.2007'den sonra bu şirketten ayrılacağını belirtir.

5.3.5 Verilerin anlık değerlerinin sorgulanması

Şekil 10, halen 22'nolu bölümde çalışıp aylık 2500 TL üzerinde ücreti olanların listesi için SQL kodlanmasını gösterir. Bu anlık sorgulama, BÖLÜM= 'DEP_ID22', ÜCRET > 2500 ve çift zamanlı bitemporal atomlarının olay zamanı üst limitleri 'şimdi'ye karşılık gelen son değer olan çalışanların numaralarını ve isimlerini seçer.

5.4 Çift Zamanlı İlişkisel Tablo Sorgulamaları

Çift zamanlı ilişkisel cebir işlemlerinin kullanımının açıklanması için kullanılacak sorgulama örneklerinde, bir sonraki sayfada bulunan Tablo 4'deki veriler kullanılacaktır.

Sorgulama 1: 01.06.2008 ve 01.06.2009 tarihleri arasında hangi ücret bilgileri sisteme kayıtlıdır?

Bu sorgu veri tabanındaki bütün verileri ve işlem zaman aralığını kullanır. Çalışan tablosunda bulunan her bir sıranın (nesnenin) ücret sütundaki işlem zaman aralığı, alt sınırı '01.06.2008' ve üst sınırı '01.06.2009' ile karşılaştırıp seçilir, çalışan numarası ile birlikte yazdırılır. SQL dilinde yazılımı Şekil 11'de, sorgulama sonucu Tablo 5'de gösterilmiştir.

```
UPDATE TABLE (SELECT Ç.İSİM
                    FROM  ÇALIŞAN Ç
                    WHERE  Ç.NUMARA=13456) İS
SET İS.OLAY_ZMN_ÜST_SINIR = '15.01.2007',
  İS.İŞLEM_ZMN_ÜST_SINIR = sysdate
WHERE İS.OLAY_ZMN_ÜST_SINIR = son_değer;
UPDATE TABLE (SELECT Ç.ADRES
                    FROM  ÇALIŞAN Ç
                    WHERE  Ç.NUMARA=13456) ADR
SET ADR.OLAY_ZMN_ÜST_SINIR = '15.01.2007',
  ADR.İŞLEM_ZMN_ÜST_SINIR = sysdate
WHERE ADR.OLAY_ZMN_ÜST_SINIR = son_değer;
UPDATE TABLE (SELECT BÖLÜM_GEÇMİŞİ
          FROM TABLE (SELECT Ç.BÖLÜM_ŞEF
                         FROM ÇALIŞAN Ç
                         WHERE Ç.NUMARA = 13456))BÖL
SET BÖL.OLAY_ZMN_ÜST_SINIR = '15.01.2007',
  BÖL.İŞLEM_ZMN_ÜST_SINIR = sysdate
WHERE BÖL.OLAY_ZMN_ÜST_SINIR = son_değer;
UPDATE TABLE (SELECT ŞEF_GEÇMİŞİ
                FROM TABLE (SELECT E.BÖLÜM_ŞEF
                               FROM EMPLOYEE
                               WHERE Ç.NUMARA=13456)) ŞF
SET ŞF.OLAY_ZMN_ÜST_SINIR = '15.01.2007',
  ŞF.İŞLEM_ZMN_ÜST_SINIR = sysdate
WHERE ŞF.OLAY_ZMN_ÜST_SINIR = son_değer;
UPDATE TABLE (SELECT Ç.ÜCRET
                    FROM  ÇALIŞAN Ç
                    WHERE  Ç.NUMARA=13456) ÜCR
SET ÜCR.OLAY_ZMN_ÜST_SINIR = '15.01.2007',
  ÜCR.İŞLEM_ZMN_ÜST_SINIR = sysdate
WHERE ÜCR.OLAY_ZMN_ÜST_SINIR = son_değer;
```

Şekil 9: SQL dilinde BTA ile bir sıra silinmesi.

```
SELECT Ç.NUMARA, İS.DEĞER AS İSİM,
FROM ÇALIŞAN Ç,
          TABLE(Ç.İSİM) İS,
          TABLE(Ç.BÖLÜM_ŞEF) BÖL_ŞEF,
                TABLE(BÖL_ŞEF.BÖLÜM_TARİHÇESİ) BÖL,
          TABLE(Ç.ÜCRET) ÜCR
WHERE DEP.DEĞER = 'DEP_ID22' AND ÜCR.DEĞER >2500
AND İS.OLAY_ZMN_ÜST_SINIR = son_değer
AND ÜCR.OLAY_ZMN_ÜST_SINIR = son_değer
AND BÖL.OLAY_ZMN_ÜST_SINIR = son_değer
```

Şekil 10: BTA veri tipi ile anlık veri tabanı sorgulaması.

```
SELECT E.EMP#,
SAL.VALUE AS SALARY,
SAL.TRAN_TIME_LOWER_BOUND AS TT_LOWERBOUND,
SAL.TRAN_TIME_UPPER_BOUND AS TT_UPPERBOUND,
SAL.VALID_TIME_LOWER_BOUND AS VT_LOWERBOUND,
SAL.VALID_TIME_UPPER_BOUND AS VT_UPPERBOUND
FROM EMPLOYEE E, TABLE(E.SALARY) SAL
WHERE SAL.VALID_TIME_LOWER_BOUND BETWEEN
                    '01.06.2008' AND '01.06.2009'
```

Şekil 11: Sorgulama 1'in SQL dilinde yazımı.

Tablo 4: Çift Zamanlı İlişkisel Modeli kullanarak tanımlanan ÇALIŞAN tablosu. (Burada, Numara, İsim-B, Adres-B ve Doğum_Tarihi sütunları ilk bölümünde, Bölüm ve Ücret-B sütunları da ikinci bölümünde verilmektedir.)

No.	İsim-B	Adres-B	Doğum Tarihi	Bölüm
	İsim	Adres		Bölüm_Adı-B
				Bölüm_Adı
101	<[01.01.2007 , *şimdi*], [01.01.2007 , *şimdi*], Barış Özer>	<[01.01.2007 , *şimdi*], [01.01.2007 , *şimdi*], A_1>	05.10.1980	
102	<[01.01.2007 , *şimdi*], [01.01.2007 , *şimdi*], Emre Alper>	<[01.01.2007 , *şimdi*], [01.01.2007 , *şimdi*], A_2>	10.10.1975	
103	{<[01.12.2007 , 01.03.2008), [01.03.2008 , 01.03.2009), İmran Çelik>, <[01.04.2009, 01.04.2010), [01.04.2009, 01.04.2010), İmran Alper>, <[01.05.2010, 01.09.2010), [01.05.2010, 01.09.2010), İmran Çelik>, <[01.05.2011, *şimdi*], [01.07.2011, *şimdi*], İmran Çelik>}	{<[01.12.2007 , 01.03.2009), [01.03.2008 , 01.03.2009), A_3>, <[01.04.2009, 01.04.2010), [01.04.2009, 01.04.2010), A_2>, <[01.05.2010, 01.09.2010), [01.05.2010, 01.09.2010), A_3>, <[01.05.2011,*şimdi*], [01.07.2011, *şimdi*], A_3>}	03.02.1990	
104	<[01.03.2008 , *şimdi*], [08.03.2008 , *şimdi*], Esin Aksoy >	{<[01.03.2008 , 01.08.2011), [08.03.2008 , 01.08.2011), A_4>, <[01.09.2011, *şimdi*], [01.09.2011, *şimdi*], A_4>}	06.08.1982	
105	<[01.10.2007 , *şimdi*], [01.10.2007 , *şimdi*], Yunus Can>	<[01.10.2007 , *şimdi*], [01.10.2007 , *şimdi*], A_5>	05.05.1985	

Bölüm		Ücret-B
Bölüm_Adı-B	Şef-B	Ücret
Bölüm_Adı	Şef	
{<[01.01.2007 , 01.08.2007), [01.01.2007 , 01.10.2007), Pazarlama>, <[01.09.2007 , *şimdi*], [01.11.2007 , *şimdi*], Planlama>}	<[01.01.2007 , *şimdi*], [01.01.2007 , *şimdi*], Emre Alper>	{<[01.01.2007 , 01.08.2007), [01.01.2007 , 01.10.2007), 2500>, <[01.09.2007 , 01.10.2008), [01.11.2008 , 01.12.2008), 3000>, <[01.11.2008 , 01.08.2009), [01.01.2009, 01.10.2009), 3200>, <[01.09.2009, *şimdi*], [01.11.2009, *şimdi*], 4000>}
{<[01.01.2007 , 01.08.2007), [01.01.2007 , 01.10.2007), Satış>, <[01.09.2007, *şimdi*], [01.11.2007 , *şimdi*], Planlama>}	<[01.01.2007 , *şimdi*], [01.01.2007 , *şimdi*], Barış Özer>	{<[01.01.2007 , 01.02.2008), [01.01.2007 , 01.03.2008), 2500>, <[01.02.2008 , 01.08.2009), [01.04.2008 , 01.10.2009), 3200> <[01.09.2009, *şimdi*], [01.11.2009, *şimdi*], 4000>}
{<[01.02.2008 , 01.09.2010), [01.03.2008 , 01.09.2010), Teknik Destek>, <[01.05.2011, *şimdi*], [01.07.2011, *şimdi*], Teknik Destek >}	{<[01.02.2008 , 01.09.2010), [01.03.2008 , 01.09.2010), Yunus Can>, <[01.05.2011, *şimdi*], [01.07.2011, *şimdi*], Emre Alper>}	{<[01.02.2008 , 01.11.2008), [01.03.2008 , 01.03.2009), 2010>, <[01.12.2008 , 01.09.2010), [01.02.2009, 01.09.2010), 2200>, <[01.05.2011, *şimdi*], [01.07.2011, *şimdi*], 2500>}
<[01.03.2008 , *şimdi*], [08.03.2008 , *şimdi*], Satış>	{<[01.03.2008 , 07.03.2008), [08.03.2008 , *şimdi*], Emre Alper >, <[08.03.2008 , *şimdi*], [08.03.2008 , *şimdi*], Yunus Can>}	{<[01.03.2008 , 01.03.2009), [08.03.2008 , 01.06.2009), 2200>, <[01.04.2009, 01.03.2010), [01.07.2009, 01.06.2010), 2400>, <[01.04.2010, *şimdi*], [01.07.2010, *şimdi*], 2600>}
<[10.01.2007 , *şimdi*], [10.01.2007 , *şimdi*], Satış>	<[01.10.2007 , *şimdi*], [01.10.2007 , *şimdi*], Barış Özer>	{<[01.10.2007 , 01.06.2009), [01.10.2007 , 01.09.2009), 2500>, <[01.07.2009, 01.08.2010), [01.10.2009, 01.12.2010), 2800>, <[01.09.2010, *şimdi*], [01.01.2011, *şimdi*], 3000>}

Tablo 5: Sorgulama 1 sonucunda çıkan tablo.

Numara	Ücret
101	< [01.06.2008, 01.10.2008), [01.06.2008, 01.10.2008), 3000> < [01.11.2008, 01.06.2009), [01.11.2008, 01.06.2009), 3200>
102	< [01.06.2008, 01.06.2009), [01.06.2008, 01.06.2009), 3200>
103	< [01.06.2008, 01.11.2008), [01.06.2008, 01.11.2008), 2010> < [01.12.2008, 01.06.2009), [01.12.2008, 01.06.2009), 2200>
104	< [01.06.2008, 01.03.2009), [01.06.2008, 01.03.2009), 2200> < [01.04.2009, 01.06.2009), [01.04.2009, 01.06.2009), 2400>
105	< [01.06.2008 , 01.06.2009), [01.06.2008 , 01.06.2009), 2500>

Sorgulama 2: Aynı adresi paylaşan çalışanların kimlik numaralarını ve hangi tarihler arasında aynı adresi paylaştıklarını bulunuz.

Bu sorgulama veri tabanını üzerinde "birleştirme" ve "zaman kesiti" işlemlerini kullanır. Çalışan tablosunun birleştirme işlemi kullanılarak kendisi ile Kartezyen çarpımı alınır, adres değerleri aynı olan sıralar seçilir. Zaman kesiti işlemi ADRES sütunundaki olay zamanı aralığı paralel olanları belirleyerek 'ne zaman' yanıtını uygulamış olur. Son olarak çalışan numaraları, adres değeri ve bu adresin hangi zaman aralığında paylaşıldığı bilgileri yazdırılır. SQL dilinde yazılımı Şekil 12'de, çıkan sonuç Tablo 6'da bulunmaktadır.

```
SELECT  E.EMP#, A.ADDRESS.VALUE,
        E1.EMP#, B.ADDRESS.VALUE
FROM EMPLOYEE E, TABLE(E.ADDRESS) A,
        EMPLOYEE E1, TABLE(E1.ADDRESS) B
WHERE A.VALUE = B.VALUE
AND E.EMP# > E1.EMP#
AND MY_PKG.TIME_SLICE(E.EMP#, E1.EMP#, A, B)
```

Şekil 12: Sorgulama 2'nin SQL dilinde yazımı.

Tablo 6: Sorgulama 2 sonucunda çıkan tablo.

N.	Adres	N.	Adres_A
101	<[01.01.2007, *şimdi*], [01.01.2007, *şimdi*], A_2>	103	<[01.04.2009, 01.04.2010), [01.04.2009, 01.04.2010), A_2>

Sorgulama 3: ['01.05.2007', '12.12.2009'] tarihleri arasında 'Emre Alper' adlı çalışanın veri tabanında kayıtlı çalıştığı bölümlerin tarihçesini bulunuz.

Bu sorgulama, Şekil 13'de görüldüğü gibi, veri tabanını tarihsel veri tabanı olarak değerlendirerek olay zaman aralığını kullanır. AS_OF işlemi BÖLÜM özniteliğindeki olay zamanını '01.05.2007', '12.12.2009' aralığına geri çağırır. İSİM özniteliğindeki çift zamanlı atomdan değeri 'Emre Alper' olanlar seçilir. BÖLÜM özniteliğindeki çift zamanlı atomdan değeri ve karşı gelen olay zamanları yazdırılır. Tablo 7 bu sorgulama sonucunu göstermektedir.

Sorgulama 4: Esin Aksoy'un '01.03.2008' ve '30.03.2008' zaman aralığında şefi, veri tabanında '01.03.2008' ve '20.03.2008' işlem zaman aralığında kim olarak biliniyordu?

Bu sorgulama çift zamanlı veri tabanı modelinin hata düzeltme varsa bunu listeleyeceğini gösterir, Şekil 14'de SQL dilinde yazılımı verilmiştir. Tablonun işlem zamanı 01.03.2008-20.03.2008 arası durumu alınır. ŞEF özniteliğinin çift zamanlı atomunun olay zamanı alt sınırı 01.03.2008 ile ve üst sınırı 30.03.2008 ile karşılaştırılır. İSİM özniteliği değeri 'Esin Aksoy' ve olay zamanı '01.03.2008' ve '30.03.2008 ' arasında olan sıra seçilir. Bu sıranın ŞEF özniteliğindeki bilgiler yazdırılır.

Tablo 7: Sorgulama 3 sonucunda çıkan tablo.

İsim	Bölüm
Emre Alper	< [01.05.2007, 01.08.2007),[01.05.2007, 1.10.2007), Satış>
	< [01.09.2007, 12.12.2009), [01.11.2007, 12.12.2009), Planlama>

```
SELECT E.EMP#,NAM.VALUE AS NAME,
DEP.VALUE AS DEPARTMENT,
DEP.VALID_TIME_LOWER_BOUND AS VT_LOWERBOUND,
DEP.VALID_TIME_UPPER_BOUND AS VT_UPPERBOUND
FROM EMPLOYEE E, TABLE(E.NAME) NAM,
        TABLE(E.DEPT_MNG) DEP_MAN,
        TABLE(DEP_MAN.DEPARTMENT_HISTORY) DEP
WHEREMY_PKG.AS_OF(VALUE(DEP),
        '01.01.2004','12.12.2005')=1 AND
        NAM.VALUE LIKE '%EMRE ALPER%'
```

Şekil 13: Sorgulama 3'ün SQL dilinde yazımı.

```
SELECT MAN.VALUE AS MANAGER,
MAN.VALID_TIME_LOWER_BOUND AS VT_LOWERBOUND,
MAN.VALID_TIME_UPPER_BOUND AS VT_UPPERBOUND,
MAN.TRANSACTION_TIME_LOWER_BOUND AS
TT_LOWERBOUND,
MAN.TRANSACTION_TIME_UPPER_BOUND AS
TT_UPPERBOUND
FROM EMPLOYEE E, TABLE(E.NAME) NAM,
        TABLE(E.DEPT_MNG) DEP_MAN,
        TABLE(DEP_MAN.MANAGER_HISTORY) MAN
WHERE NAM.VALUE = 'ESİN AKSOY'
AND MY_PKG.AS_OF(VALUE(MAN),'01.03.2008',
'20.03.2008')=1
AND MAN.VALID_TIME_UPPER_BOUND BETWEEN
'01.03.2008' AND '30.03.2008'
```

Şekil 14 : Sorgulama 4'ün SQL dilinde yazımı.

Tablo 8'de de görüldüğü gibi "Emre Alper" Esin Aksoy'un şefi olarak işlem zamanı 01.03.2008 'de atanmıştır. Fakat 07.03.2008 zamanında aslında Yunus Can'ın şef olarak atanması gerektiği fark edilmiştir. Bu düzeltme çift zamanlı veri tabanında saklanabilmektedir.

Tablo 8: Sorgulama 4 sonucunda çıkan tablo.

Şef
<[01.03.2008 , 07.03.2008), [08.03.2008 , 30.03.2008), Emre Alper>
<[08.03.2008 , 30.03.2008), [08.03.2008 , 30.03.2008), Yunus Can>

6 Sonuçlar ve Öneriler

Satır zaman etiketleme yönteminin avantajı, satır zaman-etiketlenmiş ilişkilerin endüstride kullanılan veri tabanı yönetim sistemlerinde kolayca uygulanabilmesidir. Uygulama geliştiricileri ve kullanıcılar bu yöntemi kullanarak zamansal veri tabanı yapılarını anlayabilir ve sorguları ifade edebilirler. Satır zaman etiketleme yaklaşımı, 1NF kullandığından, iyi anlaşılmış bellek örgütlenimi, sorgulama eniyileme ve sorgulama değerlendirme teknikleri kullanılabilir, geleneksel

fonksiyonel bağımlılık kuralları kolayca uygulanabilir. Buna karşılık bu yaklaşımın en büyük dezavantajı, veri tekrarlanmasıdır.

Satırlar bileşik değer kümeleri olup, içiçe geçmiş (N1NF) tablolar gerektirmesine rağmen, sütun zaman etiketleme yönteminin satır etiketleme yöntemine göre sağladığı üstünlükler bulunmaktadır. N1NF ilişkiler kullanarak veri tekrarı önlenmekte olup, bu durum kullanıcılar için kesinlikle çok daha anlamlıdır. Sütun zaman etiketleme modeli zamansal olarak hem farklı yapıda hem de türdeş olan verileri destekler. Bu yaklaşım nesnenin bütün geçmişini pek çok satıra dağıtmaktansa tek bir satırda modeller. Sütun zaman etiketleme modeli kullanıcının gerçeği algılayacağı görünüme daha yakın doğal görünüm sağlar ve bundan dolayı tasarımı ve sorgulaması daha kolay olabilir. Pek çok zamansal özniteliğin tek bir tabloda olması mümkündür. Zamansal öznitelikler aynı tablo içinde farklı zaman birimleri (yıl, ay, gün, dakika... gibi) ile de ifade edilebilir.

Sütun zaman etiketleme yaklaşımı pek çok araştırmacı tarafından karmaşık olarak eleştirilmiştir. Fakat nesnesel-ilişkisel veritabanı yönetim sistemleri, SQL3 ve veri tabanı teknolojilerindeki gelişmeler N1NF tablolar kullanarak sütun zaman etiketleme ile zamansal verinin modellenmesine olanak sağlamıştır.

Bu çalışma, zamansal veritabanı konusu ile ilgili temel kavramlar, yöntemler ve örnek uygulama içermektedir. Kullanıcıların zamansal veritabanını kolayca kullanabileceği ara yüz çalışması ve sorgulamalardaki SQL dil örnekleri için yer kalmadığından bir sonraki makaleye bırakılmıştır. Ayrıca, bu çalışmada önerilen sütun zaman etiketleme ile satır zaman etiketleme çift zamanlı veri tabanı modellerinin karşılaştırma çalışması planlanmaktadır. Bunun için aynı veriler kullanılarak satır ve sütun çift zaman etiketlenmiş iki veritabanı oluşturulacaktır. Özellikle SpyTime'da [17] belirtilen ölçüt sorgulamalar temel alınarak her iki yönteminde hafızada kapladığı alan, disket bloklarına erişim zamanları ve sorgulamalarda kullanılan CPU zamanlarının analizinin yapılması planlanmaktadır.

Zamansal veri tabanlarına paralel olarak uzaysal veri tabanları da 20 yıldan fazla bir süredir araştırılan bir konudur. İçiçe geçmiş çift zamanlı ilişkisel veri tabanı modelini uzaysal veri tabanı ile birleştirerek uzaysal-çift zamanlı veri tabanı oluşturmak diğer ilginç bir araştırma alanıdır. Böyle bir uzaysal-çift zamanlı veri tabanı hem mekân hem de zaman boyutları için gerekli veri tiplerini destekleyecek ve sonuç olarak yeni veri tabanı uygulamalarını mümkün kılacaktır.

7 Teşekkür

Pamukkale Üniversitesi, Mühendislik Bilimleri Dergisine katkı sağlayan tüm yazar ve hakemlere teşekkür ederim.

8 Kaynaklar

[1] Tansel, A. U., 'Modeling Temporal Data', Information and Software Technology, pp. 514-520, 1990.
[2] Snodgrass, R., Ahn, I., 'Performance Evaluation of a Temporal Database Management System', ACM 1986.
[3] Tansel, A. U., Atay, C. E., 'Nested Bitemporal Relational Algebra', International Symposium on Computer and Information Sciences, pp. 622-633, 2006.
[4] Stonebraker, M., Moore, D., Object-relational DBMSs: Tracking the Next Great Wave, San Francisco, The Morgan Kaufmann Series in Data Management Systems, 1999.
[5] Melton, J., Understanding Object-Relational and Other Advanced Features, Morgan Kaufmann Publishers, San Francisco, 2003.
[6] Tansel, A. U., Clifford, J., Gadia, S., Jajodia, S., Segev, A., Snodgrass, R., eds., Temporal Databases: Theory, Design, and Implementation, Benjamin/Cummings, 1993.
[7] Atay, C. E., Tansel, A. U., Bitemporal Databases: Modeling and Implementation, VDM Verlag, 2009.
[8] Gadia, S. K., 'Ben-Zvi's Pioneering Work in Relational Temporal Databases'. Chapter 8, pp. 202 – 207. In A. Tansel et al., editors, Temporal Databases, Benjamin/ Cummings, 1993.
[9] Snodgrass, R., 'The Temporal Query Language TQUEL', ACM Transactions on Database Systems, June 1987.
[10] Ozsoyoglu, G., Ozsoyoglu, M.Z., Matos, V., 'Extending Relational Algebra and Relational Calculus with Set-Valued Attributes and Aggregate Functions', ACM Trans. Database Systems, vol. 12, no. 4, pp. 566-592, 1987.
[11] Gadia, S. K., 'A Homogeneous Relational Model and Query Languages for Temporal Databases', ACM Transactions on Database Systems, December 1988.
[12] Clifford, J., Croker, A., 'The Historical Relational Data Model (HRDM) and Algebra Based on Lifespans', in Proceedings of the International Conference on Data Engineering. (Los Angeles, Calif.). IEEE Computer Society Press, 1987.
[13] Bhargava, G., Gadia, S., 'Relational Database Systems with Zero Information Loss', IEEE Transactions on Knowledge and Data engineering , 1993.
[14] Tansel, A. U., 'Temporal Relational Data Model', IEEE Transactions on Knowledge and Data Engineering, Vol. 9, No. 3, June 1997.
[15] Chau, V. T., Chittayasothorn, S., 'A Temporal Compatible Object Relational Database System', The IEEE Southeastcon, 2007.
[16] Dumas, M., Fauvet, M. C., Scholl, P. C., TEMPOS: A Platform for Developing Temporal Applications on Top of Object DBMS, TKDE, IEEE, 2004.
[17] Shasha, D., Zhu, Y., "SpyTime-a Performance Benchmark for Bitemporal Database", www.cs.nyu.edu/shasha/ spytime/spytime.html, Retrieved December 01, 2009.
[18] Combi, C., Montanari, A., Pozzi, G., "The T4SQL Temporal Query Language", CIKM'07, pp 193-202, 2007.
[19] Yang, J., Ying, H., Widom, J., 'TIP: A Temporal Extension to Informix', Proceedings of the SIGMOD, pp. 596-6, 2010.
[20] Clifford, J., Croker, A. and Tuzhilin, A. 'On Completeness of Historical Relational Query Languages', ACM Transactions on Database Systems, 19 (1), 64-116, 1994.
[21] Wang, F., Zaniolo, C., 'XBiT: An XML-Based Bitemporal Data Model.' Proceedings of 23 International Conference on Conceptual Modeling, pp. 810–824, 2004.
[22] Wang, F., Zhou, X., Zaniolo, C., 'Using XML to Build Efficient Transaction-Time Temporal Database Systems on Relational Databases', Proceedings of the 22nd International Conference on Data Engineering, pp: 131-135, 2006.
[23] Ali, K. A., Pokorny, J., 'A Comparison of XML-Based Temporal Models', Advanced Internet Based Systems and Applications, pp: 339-350, 2009.
[24] Tansel, A. U. 'Adding Time Dimension to Relational Model and Extending Relational Algebra', Information Systems, 11, No. 4, pp. 343–355, 1986.
[25] Allen, J., 'Maintaining Knowledge about Temporal Intervals', Communications of the ACM, 16 (11), pp: 832-843, 1983.