Türkçe' den SQL Sorgularına Çeviri Yapan Bir Doğal Dil İşleme Uygulaması (NALAN-TS)

Turkish Natural Language Interface for Generating SQL Queries (NALAN-TS)

ÖZET

Bu makalede NALAN-TS adında doğal dil işleyen ve Türkçe'den SQL sorgularına çeviri yapan bir uygulama anlatılmıştır. NALAN-TS cümle çözümleyici, anlam yaratıcısı, anlam çözümleyicisi, sorgu yaratıcısı ve sorgu çalıştırıcısı bileşenlerinin bir araya getirilmesiyle oluşmuştur. NALAN-TS Türkçe ve veritabanı sözlüklerini içeren sözlük tabanlı bir uygulamadır. NALAN-TS veritabanı sözlüğünü kullanarak tabloların ve sütunların neleri temsil ettiğini inceler.

ÖZGEÇMİŞ

İbrahim Maden

1981 yılında İstanbul'da doğan İbrahim Maden 1998 yılında Şişli Çağlayan Lisesi' ni, 2003 yılında da Yeditepe Üniversitesi Bilgisayar Mühendisliği Bölümü'nü bitirdi. 2002-03 yılları arasında aynı bölümde öğrenci asistanı olarak görev almıştır.

Şeniz Demir

1979 yılında Artvin'de doğan Şeniz Demir, 2000 yılında Marmara Üniversitesi Bilgisayar Mühendisliği Bölümü'nü bitirdi. Halen Boğaziçi Üniversitesi Bilgisayar Mühendisliği Bölümü'nde yüksek lisans eğitimine devam etmekte ve Yeditepe Üniversitesi Bilgisayar Mühendisligi Bölümü'nde araştırma görevlisi olarak çalışmaktadır.

Ender Özcan

1991 yılında ODTÜ Elektrik ve Elektronik Mühendisliği Bölümü'nü bitirdi. Devlet bursuyla gittiği Syracuse Üniversitesi (ABD, New York), Bilgisayar ve Enformatik Bilimleri Bölümü'nde 1994 yılında yüksek lisansını, 1998 yılında da doktorasını tamamladı. Bu süre içerisinde araştırma görevliliğinin yanı sıra, BlackWatch Inc.'de yazılım uzmanı olarak çalıştı. 1998 yılı güz döneminden beri Yeditepe Üniversitesi Bilgisayar Mühendisliği Bölümü'nde öğretim üyesi olarak görevine devam etmektedir.

ABSTRACT

In this paper, a Turkish natural language interface, named as NALAN-TS, for generating SQL queries is described. NALAN-TS is composed of five components: syntactic parser, semantic analyzer, meaning extractor, SQL constructor and SQL executer. NALAN-TS is a dictionary based application, including a Turkish and a database dictionary. The database dictionary is used to detect the referrals to the tables and the columns.

TÜRKÇE' DEN SQL SORGULARINA ÇEVİRİ YAPAN BİR DOĞAL DİL İŞLEME UYGULAMASI (NALAN-TS)

GİRİŞ

Doğal dil işleme (DDI) günümüzde hızla gelişmekte olan yapay zeka konularından biridir. Bu konu ile ilgili teorik olarak pek çok çalışma yapılmasına karşın, sınırlı sayıda pratik uygulama mevcuttur. Bu eksikliklerden birini doldurmak amacıyla yazılmış olan NALAN-TS, Yapısal Sorgu Dili (SQL) sorguları için bir Türkçe arayüz uygulamasıdır. Amacımız SQL'i hiç bilmeyen insanların bile kolay bir şekilde sorgularını çalıştırabilmelerine olanak sağlamaktır. NALAN-TS'de bazı değişiklikler ve eklemeler yapılarak Boğaziçi Üniversitesi'nde geliştirilen TOY [1] projesinin biçimbilim ve sözdizimbilim altyapıları kullanılmıştır. Bu altyapıların üzerine NALAN-TS'in amacına yönelik bir anlambilim yapısı oluşturulmuştur.

Girilen cümle beş ayrı aşamadan geçer. Bunlar cümle çözümleme, anlam yaratma, anlam çözümleme, sorgu yaratma ve sorgu çalıştırma aşamalarıdır. Cümle çözümleme aşamasında cümle andaçlarına ayrılır. Bu işlemi takiben andaçlar kullanılarak bir anlam yaratılır. Daha sonra yaratılan anlam çözümlenir ve SQL anahtar sözcükleri bulunur. Bu anahtar sözcükler sayesinde sorgu yaratılır ve en son olarak sorgu çalıştırılır.

NALAN-TS sözlük tabanlı bir uygulamadır ve yukarıda değinilen beş aşamanın dışında, değişik amaçlara yönelik iki tane de sözlük mevcuttur. Bunlar Türkçe ve veritabanı sözlükleridir. Türkçe sözlük programda kullanılabilecek sözcükleri içerir. Diğer bir deyişle programda kullanılacak bir sözcük öncelikle Türkçe sözlüğe girilmelidir. Veritabanı sözlüğü ise kullanılan veritabanının adını, kullanıcı adını ve şifresini, tablo ve sütun isimleri ile bu tablolar üzerinde tanımlanmış kısıtları içerir.

Ayrıca SQL anahtar kelimelerinin ve işlemlerinin yeraldığı bir SQL tablosuda mevcuttur. Bu tablo sorgu anahtar kelimelerinin ve işlemlerinin Türkçe karşılıklarından oluşur.

Kabul edilecek cümlelerin doğal dile daha yakın olmasını sağlamak için, tabloların ve sütunların adlarını kullanmak yerine bu adların nelere karşılık geldikleri üzerinde durulmuştur. Bu sayede NALAN-TS, SQL sorguları için bir çeviriciden ziyade tam bir arayüz özelliği kazanmıştır.

NALAN-TS her tür veritabanı ve işletim sistemi üzerinde çalışabilecek şekilde tasarlanmıştır. Programlama dili olarak SWI - Prolog, kullanıcı arayüz birimi için de Java kullanılmıştır. Yaratılan SQL sorgusu ANSI ölçütlerine uygundur. Ancak SWI- Prolog'un şimdilik ODBC arayüzünün çalışmaması nedeniyle testler ORACLE veritabanı üzerinde yapılmıştır.

DOĞAL DİL İŞLEME (NLP)

Bilgisayarlarla doğal dil ile iletişim kurma yapay zekanın uzun süreden beri uğraştığı bir araştırma konusudur. NLP beş ana seviyede incelenebilir [2]:

- Sesbilim
- Biçimbilim
- Sözdizimbilim
- Anlambilim
- Kullanımbilim

Sesbilim harflerin seslerini ve bunların dil içinde nasıl kullanıldığını inceler. Tüm dillerin bir alfabesi vardır ve her harfin sesi diğerlerinden farklıdır. Biçimbilim sözcük kurumudur. İki tür sözcük oluşturma yöntemi mevcuttur. Bunlar türetme ve ses değişmesidir. Sözdizimbilim sözcüklerin cümle oluşturmak için ne şekilde sıralanmaları gerektiğini inceler. Ancak günlük hayatta bazen olması gereken sözdiziminin dışında da cümleler kullanılabilir. Anlambilim dilin gerçek dünyayla iletişim kurmasını sağlar. Cümle yapısının anlaşılması ve bunun sonucunda eyleme geçilmesi bu aşamada olur. Günümüzde anlambilim sonuçlandırılmaya yaklaşılmış bir çalışma durumundadır. Kullanımbilim dilin duruma göre değişimini inceler. Bir sözcük tek başınayken yada bir cümle içindeyken farklı anlamlar ifade edebilir.

NALAN-TS bu beş seviyeden biçimbilim, sözdizimbilim ve anlambilim seviyelerini kullanır.

TÜRKÇE DİLİ

Türkçe dili Ural - Altay Dil Ailesi'nin bir üyesidir. Latin karakterlerini kullanır. Türk alfabesinde yirmidokuz harf mevcuttur. Bunlardan yirmibiri sessiz, sekizi sesli harftir. Türkçe ile NLP uygulaması gerçekleştirirken bazı zorluklarla karşılaşılmaktadır. Türkçe sondan eklemeli bir dildir. Aynı ek farklı köklere farklı şekillerde eklenebilir. Bazen bu ekleme sürecinde ses değişmesi de olabilir. Bu nedenle biçimsel olarak bir sözcüğün çözümlenmesi oldukça zordur. Bu olayla ilgili örnekler Tablo 1'de verilmiştir. TOY [1] projesinin altyapısı oluşabilecek ses değişimleri dikkate alınarak geliştirilmiştir.

Kelime	Biçimsel Ekler
Görünürlerde	gör + ün + ür + ler +de
Ağaca	ağaç + a
Ağlıyor	ağla + yor

Tablo 1: Biçimsel ekler

Türkçe'de yedi sözdizimsel sınıf vardır. Bu sınıflar isim, özel isim, birleşik isim, sıfat, fiil, edat ve bağlaçtır. NALAN-TS'de bu sınıflardan birleşik isim ve bağlaç dışında kalanlar gerçeklenmiş ve kullanılmıştır. Türkçe'de karşılaşılan diğer bir zorluk da sözcük sırasıdır. Türkçe'de düzenli sözcük sırası özne nesne yüklem şeklindedir. Cümlenin dil bilgisi yapısı, isim tümleciği tarafından belirlenir ve bu sebeple sözcük sırası dil bilgisi yapısını değiştirmeden serbestçe değiştirilebilir. Tablo 2'de anlamları aynı söz dizimleri farklı örnek cümleler vardır.

Cümle
Çocuğa kitabı ben verdim
Çocuğa ben kitabı verdim
Ben çocuğa kitabı verdim

Tablo 2: Farklı sözdizimli cümleler

Bu cümlelerde ortak olan nokta yüklemin her üçünde de en sonda olmasıdır. Söz dizimi problemini ortadan kaldırmak için NALAN-TS sadece Türkçe'nin düzenli sözcük sırasını destekler.

YAPISAL SORGU DİLİ (SQL)

SQL ilişkisel veritabanlarında kullanılan ve ANSI standardlarında bir sorgulama dilidir. Bütün gelişmiş veritabanı uygulamalarında kullanılır. SQL, kullanılan veritabanının biçimlendiricisinden kodlanabileceği gibi diğer programlama dillerinin içinden veribanı bağlantısı sağlanarak da kodlanabilir. NALAN-TS de bu türden bir bağlantı ile sorguları çalıştırır.

SQL ile veritabanına bilgi girilebilir, varolan bilgi görüntülenebilir, değiştirilebilir yada silinebilir. Bu işlemlerin NALAN-TS projesinde gerçekleştirilen detayları Tablo 3'de gösterilmiştir.

	Tek Satır	Çok Satır	Tek Sütun	Çok Sütun	Sütun Değeri	Koşul	Koşul Değeri	İşlev	Bitişme
Gösterme	•	•	•	•		•	•	•	•
Silme	•	•				•	•		
Ekleme	•		•	•	•				
Değiştirme	•	•	•	•	•	•	•		

Tablo 3: SQL işlemlerinin detayları

Görüntüleme işlemi için birçok seçenek mevcuttur. Bunlar çok satır bütün sütunlar, çok satır çok sütun, çok satır tek sütun, tek satır bütün sütunlar, tek satır çok sütun, tek satır tek sütun görüntülemedir. Bu işlem sırasında işlevler de kullanılabilir. Ayrıca görüntüleme işleminde birden fazla tablodan bitişmeler de olabilir. Sorgunun gövdesinde yalnızca tablo adı ve sütun adları gerekir. Koşul bölümü içeren sorgularda bu bölüm için sütun adı ve değeri gereklidir.

Değiştirme işlemi için de görüntüleme işleminde olduğu gibi çok satır bütün sütunlar, çok satır çok sütun, çok satır tek sütun, tek satır bütün sütunlar, tek satır çok sütun, tek satır tek sütun değiştirme seçenekleri mevcuttur. Görüntüleme işleminden farklı olarak değiştirme işlemi bitişmelere izin vermez. Sorgunun gövdesinde tablo adı ve sütun adının yanısıra sütunun değeri de gerekir. Koşul bölümü içeren sorgularda bu bölüm için sütun adı ve değeri gereklidir.

Silme işlemi bütün tabloyu, çok satır silme yada tek satır silme şeklinde olabilir. Sorgunun gövdesinde sütun adı veya değeri gerekmez yalnızca tablo adı gerekir. Koşul bölümü içeren sorgularda bu bölüm için sütun adı ve değeri gereklidir.

Ekleme işlemi tek satır ekleme şeklinde yapılır. Satırın bütün sütunlarına ekleme yapılabileceği gibi bazı sütunlar boş da bırakılabilir. Sorgu yaratılırken tablo adı, sütun adı ve sütunun değeri gereklidir. Ekleme işleminde koşul bölümü bulunmaz.

NALAN-TS bu dört işlem için yukarıda belirtilen türlerin tümünü yapabilecek bir altyapıya sahiptir.

TASARIM

NALAN-TS için anlambilim, SQL sorguları için ve veritabanı bilgileri için tasarım çalışmaları yapılmıştır. Bunlardan ilki ve en önemlisi anlambilim tasarımıdır. Amaç cümleden elde edilen anlamın SQL sorgusuna kolayca çevrilebilmesidir. Cümlenin anlamı kendisini oluşturan kelimelerin anlamlarından elde edilir. NALAN-TS kelimelerin anlamlarını biçimbilim yapılarında yer anlan anlamsal formüllerinden çıkartır.

NALAN-TS sözlük tabanlı bir dil olduğu için kullanılacak bütün kelimelerin kullanılmadan önce sisteme girilmiş olması gerekir. NALAN-TS fiiller hariç diğer kelime türlerinde TOY' da [1] geliştirilen biçimbilim yapısını aynen kullanır. Çizim 1' de "peynir" kelimesi için hazırlanan örnek bir biçim bilim yapısı gösterilmiştir.

```
tr_morph_entry('AdKök',[[p,e,y,n,i,r],[type(noun)
,sem(X^cheese(X))]],_,_,_,i,r,ok).
```

Çizim 1: Biçimbilim yapı örneği-1

Bu yapıda sırasıyla kelimenin türü, adı, anlamsal formülü, son sesli harfi, son harfi ve kelimenin durumunu programa ileten bir değer bulunur. Daha detaylı bilgi [1]' den alınabilir.

TOY' da [1] yüklem için kullanılan anlamsal formül özne, nesne, zaman, süre gibi bilgileri içerir. Bu bilgileri SQL sorgusu yaratırken kullanmayız. Bu nedenle fiil türündeki kelimelerin biçimbilim yapısında yer alan anlamsal formülleri amaca yönelik olarak değiştirilmiştir Çizim2'de "göster" fiili için hazırlanan örnek bir biçimbilim yapısı gösterilmiştir.

```
tr_morph_entry('FiilKök',[[g,ö,s,t,e,r],[type(verb),
sem(Numara^TabloAdı^SütunAdı^Koşul^göster(Numara,
TabloAdı,SütunAdı,Koşul))]],_,_,_,e,r,ok).
```

Çizim 2: Biçimbilim yapı örneği-2

Çizim 2' deki yapıda, dört değişken bulunur. Bunlar işlem numarası, tablo adı, sütun adı(adları) ve koşul adıdır(adlarıdır). İşlem numarası şimdilik kullanılmamıştır. İleride işlemlerin sıralanmasında kullanılabilir. Diğer üç değişkenin değerlerini kullanarak SQL sorgusu yaratılır. Bir SQL sorgusunda yer alacak sütun adı ve koşul adı bir yada birden fazla olabilir. Her durumda çalışacak bir yapı yaratmak için sütun adı ve koşul adı değişkenleri kendi içinde birer liste olarak tutulmaktadır.

Sütun adı değişkeninin içerebileceği listenin yapısı
[sütun_adı-1,sütun_adı-2,...]
şeklinde tasarlanmıştır.
Koşul adı değişkenin içerebileceği listenin yapısı ise
[sütun_adı, işlem, değer, bağlaç]
şeklinde tasarlanmıştır.

Şu anda işlem bölümü sadece "=" değerini, bağlaç bölümü de sadece "ve" değerini içerebilir. İleride bu bölümlerin alabileceği değerler genişletildiğinde daha geniş bir sorgu grubu NALAN-TS bünyesinde gerçekleştirebilecektir.

Örneğin, "Adı Levent, yöneticisi 7566 olan işçilerin maaşını göster" cümlesi NALAN-TS'e girildiğinde yüklemin yapısındaki sütun adı değişkeni [maaş], koşul adı değişkeni [[ad,=,levent,ve], [yönetici,=,7566,_]] olacaktır.

Çizim 1 ve 2'de görüldüğü gibi, anlamsal formüller Lambda matematiği kullanılarak gösterilir. Lambda matematiğinde kelimelere özellik olarak yaklaşılır. Örneğin Ponpon^kedi(Ponpon) ifadesinde kedi Ponpon' un bir özelliğidir.

NALAN-TS TOY'da tanımlı kuralların yanısıra Tablo 4' deki DCG kurallarını kullanarak cümleyi çözümler.

Cümle çözümlemesi
Cümle → isim ibaresi, fiil ibaresi
isim ibaresi → isim | sıfat, isim
fiil ibaresi → isim ibaresi, fiil | fiil
Sıfat → numara, isim | özel isim, isim | edat, sıfat | sıfat
özel isim → Ender | Şeniz | İbrahim ... vb
numara → 1 | 2 | bir | iki ... vb
isim → uzunluk | işçi | maaş ...vb
edat →en
fiil → göster | yap | sil ... vb

Tablo 4: Dilbilgisi kuralları

Cümlenin anlamı FOPC kullanılarak ifade edilir. Bu ifadede, yüklem dışındaki cümle öğeleri tekil – çoğul ve belirli – belirsiz oluşlarına göre Tablo 5' de gösterilen bileşenlerin içine anlamsal formülleri yerleştirilerek kullanılır. Daha detaylı bilgi [1]' den alınabilir.

Öğe Türü	Karşılık
Tekil, belirli	Singthe(X,Res,Sco)
Tekil, belirsiz	Singsome(X,Res,Sco)
Çoğul, belirli	Plurthe(X,Res,Sco)
Çoğul, belirsiz	Plursome(X,Res,Sco)

Tablo 5: Anlam bileşenleri

Örneğin "işçileri göster" cümlesinin anlamı yaratılırken işçiler kelimesi çoğul ve belirli olduğundan onun için plurthe(X,işçi(X),Sco) bileşeni yaratılır.

Bunun dışında SQL sorgularını kavramak için de Tablo 6' da gösterilen kurallar tasarlanmıştır. Bu kurallar SQL tablosunu oluşturur.

SQL kuralları
query (select, göster)
function (max, yüksek)

Tablo 6: SQL tablosu

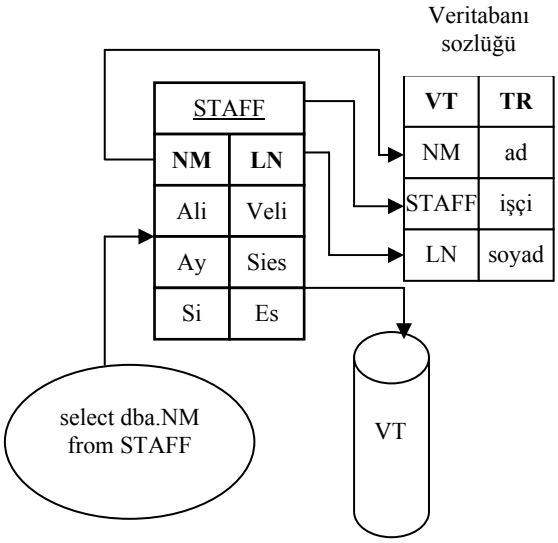
Son olarak veritabanı ile ilgili bilgileri düzenlemek için Tablo 7'deki kurallar tasarlanmıştır. Bu kurallar veritabanı sözlüğünü oluşturur.

Veritabanı kuralları
database ('NALAN-TSDB)
user ('scott', 'tiger')
table ('emp', 'işçi')
column ('emp', 'emp.empno', 'numara', 'number')
primarykey ('emp', 'emp.empno')
Foreignkey('emp','emp.deptno', 'dept', 'dept.deptno')

Tablo 7: Veritabanı sözlüğü

NALAN-TS

NALAN-TS'ı anlatmaya başlamadan önce önemli bir noktanın tekrar üzerinde durmamız gerekiyor. Çizim 3'de SQL sorgusu için Türkçe bir cümle düşünüldüğünde akla "STAFF tablosundan NM sütununu göster" gelebilir. Ancak bu cümlenin doğal dile ne kadar yakın olduğu tartışılır. Bu nedenle NALAN-TS, tablo ve sutun adlarının Türkçe'de nelere karşılık geldiklerini inceleyerek sonuçları veritabanı sözlüğüne kaydeder. Böylece örnekteki sorguya karşılık gelen cümlenin "işçilerin adını göster" şeklinde bir cümle olması sağlanır. Bu cümleyi ilk düşünülen cümle ile kıyasladığımız zaman doğal dile daha yakın olduğunu görürüz. Veritabanı sözlüğünü Prolog'da yaratmak bu dili hiç bilmeyen kullanıcılar açısından oldukça zordur. Ayrıca sözlükte oluşacak bir hata programın hatalı çalışmasına da yol açabilir. Bu iki nedenle veritabanı sözlüğünü yaratmak amacıyla Java dili kullanılarak bir kullanıcı arayüzü tasarlanmış ve gerçeklenmiştir. Böylece programın her seviyeden kullanıcıya hizmet verebilmesi sağlanmıştır.



Çizim 3: Örnek sorgu çizeneği

NALAN-TS iki ana bölümde incelenebilir. Birinci bölüm NLP bölümü ikinci bölüm de veritabanı bölümüdür. NLP bölümü girdi olarak düzenli bir Türkçe cümle kabul eder. Bu cümleyi andaçlarına ayırır. Sonra da bu andaçları kullanarak cümle için önceden tasarlanan şekilde bir anlam üretir. Veritabanı bölümü de NLP bölümünün ürettiği anlamı alır ve bu anlamı çözümleyerek sorgu yaratırken kullanılacak anahtar sözcükleri bulur. Daha sonra anahtar sözcüklerden yararlanarak sorguyu yaratır ve sorguyu veritabanında çalıştırır.

NALAN-TS'in girilen cümleyi kabul etmesi için önce kullanılacak bütün sözcükler Türkçe sözlüğe girilir. Kelimeleri sözlüğe girerken son harf ve son sesli harfe dikkat edilir. Böylece kelimenin alabileceği ekler belirlenmiş olur.

Daha sonra bu kelimeleri kullanarak cümle çözümlenmeye başlanır. Bu aşamada Tablo 4'deki dilbilgisi kuralları esas alınır.

Cümle çözümlenirken beraberinde cümlenin anlamı da çıkartılır. Aşağıda örnek bir cümle ve bu cümleden yaratılan anlam gösterilmiştir:

Cümle:
7902 numaralı işçiyi göster.
Anlamı:
singthe(_G282,(numara(7902),işçi(_G282)),göster(_G814,_G282,_G708,_G711))

Veritabanı bölümü yukarıda yaratılan anlamı alır ve bu anlamdan sorgu yaratmak için gerekli olan anahtar sözcükleri çıkarmaya çalışır.

Tasarım bölümünde de anlatıldığı gibi fiilin anlamsal yapısına bakılarak sorgunun yapısı belirlenir. Yukarıdaki örnekte tablo adını içeren değişkenin (ikinci değişken) değerinin _G282 olduğu görünüyor. Genel anlamda _G282' nin neye karşılık geldiği aranınca karşımıza _G282^işçi(_G282) çıkıyor ve tablo adının işçi olduğu anlaşılıyor. Benzer şekilde girilen cümleden varsa sütun adı(adları), koşul adı(adları) ve degeri(değerleri) elde edilir. Yukarıdaki örnekte olduğu gibi şayet bir değişkenin değeri anlamsal yapıda başka bir yerde kullanılmamışsa o değişkenin aslında cümlede kullanılmadığı anlaşılır.

Tasarım bölümünde de anlatıldığı gibi sütun adı ve koşul adı değişkenleri liste olarak tutulur. Bu listedeki bütün değerlerin hangi sözcüklere karşılık geldikleri sırayla bulunur. Örneğin "7902 numaralı işçinin adını, maaşını göster" cümlesi NALAN-TS'e girildiğinde aşağıdaki anlam elde edilir.

singthe(_G282,(numara(7902),işçi(_G282)),singthe(
_G1042,ad(_G1042),singthe(_G4687,maaş(_G4687),
göster(_G814,_G282,[_G1042,_G4687],_G711))))

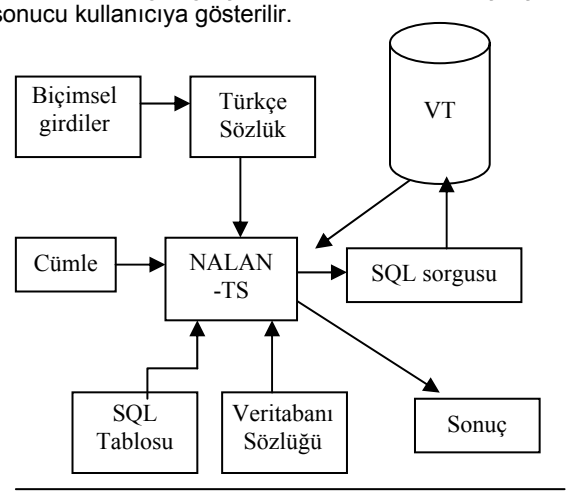
Sütun adı değişkenin içerdiği listedeki değerlerin (_G1042,_G4687) karşılık geldiği kelimeler sırasıyla bulunur.

Daha sonra elde edilen bu anahtar sözcükler veritabanı ve SQL sözlükleri kullanılarak veritabanındaki gerçek isimlerine çevrilirler. Sorgu yaratıcısı veritabanı isimlerine çevrilmiş sözcükleri alır ve sorguyu yaratır. Değişik sorgu türlerinin değişik sözdizimleri vardır. Sorgu yaratılırken bu hususa dikkat edilir. SQL' de atama işareti dizgileri ve tarihleri tek tırnak içinde, sayıları ise olduğu gibi ya da tek tırnak içinde kabul eder. Bu yüzden sorgu yaratılırken bütün değişkenlerin değerlerinin başına ve sonuna tek tırnak (') karakteri eklenir.

NALAN-TS ayrıca yaratacağı sorgularda bitişmelere de izin verir. Bu amaçla cümleden elde ettiği tablo adına öncelikli tablo adı verir. Sonra sütun adları ile ait oldukları tabloları arar. Eğer bir (yada çok) sütun için sütunun adı ile öncelikli tablo adı uyuşmazsa o sütunun bulunduğu tablonun adını bulmaya çalışır. Daha sonra bulduğu bu tablo ile öncelikli tablo arasında bağlantı kurmayı sağlayacak bir ikincil anahtar kısıtının tanımlı olup olmadığı kontrol eder. Eğer varsa sorguyu bitişmeli sorgu olarak kabul eder. Aksi halde yanlış cümle girildiği yada oluşturulacak sorgunu doğru çalışmayacağı için sorgu yaratmadan programdan çıkar. Bitişmeli sorgularda eğer aynı sütun ismi birden fazla tabloda geçiyorsa anlam karışıklığını gidermek için bu sütunlardan önce tablo adı yazılır (emp.deptno, dept.deptno). Ancak bu yapıyı başka tablolarda adları kullanılmayan sütunlar için de kullanabiliriz. Bu nedenle genel bir yapı oluşturmak için NALAN-TS'de bütün sütun adları tablo adıyla birlikte yazılır.

Son olarak NALAN-TS sorgularda küçük - büyük işlevlerini de destekler. Bu türdeki cümlelerde sütun adı işlev sütunu, koşul sütunun adı da işlev adı olarak kullanılır. Böyle bir cümle örneği Çizim 5'de verilmiştir.

Çizim 4'de NALAN-TS'deki genel iş akışını görebiliriz. Öncelikle biçimsel girdiler Türkçe sözlüğüne girilir. Sonra kullanılacak veritabanının adı, kullanıcı adı ve şifresi, tablo ve sütun adları ve bu tablolar üzerinde tanımlanmış kısıtlar varsa veritabanı sözlüğüne girilir. SQL tablosu önceden hazırlanmış haliyle sisteme dahil edilir ve program girdi almaya hazır hale gelir. Girilen cümle yukarıda değinilen işlemlerden geçirilerek SQL sorgusuna çevrilir. Daha sonra bu sorgu gerçek bir veritabanında çalıştırılarak sonucu kullanıcıya gösterilir.



Çizim 4: NALAN-TS iş akış çizeneği

KODLAMA ORTAMI

NALAN-TS'in hemen hemen tamamı SWI-Prolog kullanılarak yazılmıştır. Sadece veritabanı sözlüğünü yaratan kullanıcı arayüzü Java ile kodlanmıştır. Ayrıca XPCE ile yazılmış olan bir de kullanıcı arayüzü mevcuttur. NALAN-TS' in SWI-Prolog'da yazılmasının ana nedeni Prolog'un NLP için en uygun programlama dili [2] olmasından dolayıdır. Ayrıca SWI-Prolog NALAN-TS' in doğasına uygun olarak bütün işletim sistemleriyle (Windows, Linux) uyumlu çalışabilen bir Prolog sürümüdür. NALAN SWI-Prolog' un 5.2.2 sürümüyle derlenmiştir. Veritabanı arayüzü için Java'nın kullanılmasının sebebi de hem Java'nın işletim sistemlerinden bağımsız olması hem de arayüz tasarımının Java ile kolayca yapılabilmesidir. NALAN-TS'ı bilgisayarınızda kullanmak için bilgisayarınızda bazı programların kurulmuş olması gerkir. Bunlar SWI-Prolog derleyicisi [3], Java derleyicisi [4], ORACLE Veritabanı [6] ve SWI-Prolog ile ORACLE köprüsüdür [5].

DENEYLER

NALAN-TS ilk aşamadan itibaren her tür veritabanı ve işletim sistemi üzerinde çalışabilecek şekilde tasarlanmıştır. Deneyler Windows XP ortamında ORACLE veritabanı üzerinde yapılmıştır. Test veritabanı iki tablodan oluşur. Bunlar "emp" ve "dept" tablolarıdır.

Bu iki tablo arasında bire çok (dept – emp) türünden bir ilişki mevcuttur. Deneylere başlamadan önce bu tablolarla ilgili tüm gerekli bilgiler veritabanı sözlüğüne kaydedilmiştir.

Tablo 8 ve Tablo 9'da deney veritabanında kullanılan tabloların yapıları gösterilmiştir.

Sütun adı	Kısıt
empno	birincil anahtar
ename	yok
Sal	yok
hiredate	yok
Mgr	yok
comm	yok
deptno	ikincil anahtar
Job	yok

Tablo 8: Emp tablosunun içeriği

Sütun adı	Kısıt
Deptno	birincil anahtar
Dname	yok
Loc	yok

Tablo 9: Dept tablosunun içeriği

empno	Ename	mgr	sal
7369	Smith	7902	800
7499	Allen	7698	2000
7521	Ward	7698	1250
7566	Jones	7839	2975

Tablo 10: Emp tablosunun içeriğinden örnekler

Tablo 10'da deneylerde kullanılan "emp" tablosundan örnek satır ve sütunlar gösterilmiştir.

Deney aşamasında NALAN-TS'de gerçeklenen bütün cümle ve sorgu yapıları başarıyla test edilmiştir. Çizim 5' de NALAN-TS' in örnek bir çalışması gösterilmiştir. Satırların ilk değerleri Tablo 10 'de olduğu gibidir. Yapılan değişiklikler sonucunda "Emp" tablosunun oluşan son hali Tablo 11 'de gösterilmiştir.

Deneyler sırasında hız konusu göz önünde bulundurulduğunda Prolog' un özyinelenme ve geriye dönüklülük özelliklerinden kaynaklanan zaman kayıpları olduğu görülmüştür. Geriye dönüklülük Prolog' un önemli bir özelliği olmasına rağmen bazen istenmeyen sonuçlarında oluşmasına yol açabilir. Ek olarak Prolog yorumlanan bir dil olduğu için derlemeli dillere (örneğin; C / C++) kıyasla daha yavaş çalışır. Ayrıca başarımı etkileyen diğer bir faktör de veritabanına erişimden kaynaklanan gecikmelerdir.

SONUÇLAR

Bu proje ile NALAN-TS adında SQL sorguları için doğal dil ile (Türkçe) bir arayüz geliştirilmiştir. Bu arayüzde en temel SQL sorguları olan bilgi görüntüleme, değiştirme, silme ve ekleme işlemleri gerçeklenmiştir. Görüntüleme sorgularında çok satır çok sütün, tek satır çok sütun, çok satır tek sütun ve tek satır tek sütun işlemleri yapılabilmektedir. Ayrıca bu işlemler bitişmeli de olabilir ve işlev içerebilir. Silme sorguları tek satır ve çok satır olarak yapılabilir. Değiştirme sorgusu tek satır tek sütun, çok satır tek sütun şeklinde yapılabilir. Eklemeler ise tek satır tek sütun yada tek satır çok sütun şeklinde yapılır. Tasarlanan altyapı bütün sorgu türlerini kapsayabilecek şekildedir. NALAN-TS gelişmeye açık bir programdır. Gelecekte programa daha iyi bir kullanıcı ara yüzü yapılabilir. Ayrıca veritabanı sözlüğünü yaratan kısım da gene doğal dil kullanarak gerçeklenebilir. Son olarak, NALAN-TS'ın desteklemediği sorgu türleri de (tablo yaratma, sütun düşürme gibi) ileride programa eklenebilir.

Allen adlı işçiyi göster.
[7499,Allen,7698,2000]

7521 numaralı işçinin adını, maaşını göster.
[Ward,1250]

İşçilerin adını göster.
[Smith]
[Allen]
[Ward]
[Jones]

Jones adlı işçinin maaşını 5000 yap.

Jones adlı işçiyi göster.
[7566, Jones, 7839, 5000]

En yüksek işçi maaşını göster.
[5000]

Smith adlı işçiyi sil.

İşçileri göster.
[7499, Allen, 7698, 2000]
[7521, Ward, 7698, 1250]
[7566, Jones, 7839, 5000]

Adı John, maaşı 2500, numarası 7801, yöneticisi 7566 olan bir işçi vardır.

Bütün işçileri göster.
[7499, Allen, 7698, 2000]
[7521, Ward, 7698, 1250]
[7566, Jones, 7839, 5000]
[7801, John, 7566, 2500]

İşçilerin maaşını 2000 yap.

İşçileri göster.
[7499, Allen, 7698, 2000]
[7521, Ward, 7698, 2000]
[7566, Jones, 7839, 2000]
[7801, John, 7566, 2000]

Çizim 5: NALAN-TS'in örnek çalışması

Empno	ename	mgr	Sal
7499	Allen	7698	2000
7521	Ward	7698	2000
7566	Jones	7839	2000
7801	John	7566	2000

Tablo 11: Çalışmadan sonra Emp tablosunun içeriği

KAYNAKÇA:

[1] ***Çetinoğlu Ö.*** *, "TOY", A Prolog Based Natural Language Processing Infrastructure for Turkish, MSc Thesis, Boğaziçi Üniversitesi,2001.*

[2] ***Covington M. A.*** *, Natural Language Processing for Prolog Programmers, Prentice Hall, New Jersey, 1994.*

[3] ***SWI Prolog,*** *http://www.swi-prolog.org*

[4] ***Java****, http://java.sun.com/*

[5] ***SWI-Prolog to SQL Bridge****, http://www.geocities.com/SiliconValley/Bit/1116/PrologSQL.html*

[6] ***Oracle Database,*** *http://www.oracle.com/products/*