Adı Soyadı :Uğurcan ERGÜN-Şeyma RÜZGAR

Numarası : 090202003-080202039

Bölümü : Bilgisayar Mühendisliği (İ.Ö)

Ders Adı :Yazılım Lab. I

Proje Numarası : 3

Proje Konusu : PL/SQL kodlarının Javascript'e dönüştürülmesi

Proje Danışmanı : Arş.Gör. Gürel Yıldız, Arş Gör. Süleyman Eker

Proje Teslim Tarihi : 5 Ocak 2012

Proje Ortamı : Python 2.7.1, Vim 7.3

Geliştirme Platformu : GNU/Linux (Pardus 2011.2)

Proje Beklentileri :

Bir dosyadan okunacak PL/SQL kodlarının Javascript'e dilin söz dizim kurallarına uygun bir şekilde çevrilmesi.

Problemin Analizi ve Geliştirilen Çözümler:

PL/SQL ve Javascript söz dizimleri farklı programlama dili ailelerinden (PL/SQL -> Ada,Pascal Javascript C/C++) geldiği için oldukça farklıdır. Bu yüzden bu dillerden birini diğerine çevirebilmek için üzerlerinde bolca değişiklik yapılması gerekir. Yaptığım incelemede ifadelerden bir çoğunun diğer dillerdeki eş değer ifadeleriyle doğrudan yer değiştirebildiğini ancak bu durumu uymayan bazı durumların bulunduğunu ve bunlar için bazı özel değişiklerin yapılması gerektiğini farkettim.

Genel durumda değişikliklerin kolayca yapılabilmesi için Python dilinin Dictionary (Sözlük) veri yapısını ve standart string fonksyonlarından olan “string.replace(eski,yeni)”i kullandım. Sözlük veri yapısını indisleri geliştrici tarafından belirlenen diziler olarak özetleyebiliriz. Kullanımı şu şekildedir. “sozluk = { anahtar:deger}” Ancak bazı değişimlerin bazılarından daha sonra yapılması gerektiğinden (Örneğin “:=”:”=” değişimi “=”:”==” değişimden sonra yapılmazsa bütün atamalar karşılaştırma gibi gözükecektir.) ve sözlüklerde sıralama rastgele olduğundan ikinci bir sözlük tanımlamam gerekti. Programda kullandığım sözlük yapıları aşağıdadır.

*equ = { "number;":"var","varchar":"var","date;":"var","boolean;":"var"," =":" ==","<>":"! =","||":"+","--":"//","and":"&&","not ":"!","then":"{","else":"} else {","endif;":"}","loop":"{","end loop;":"}","do":"","for":"for","declare":"","begin":"","end;":"","dbms\_output.put\_line":"console.log"}*

*equlater = { ":=":"=" ,"elsif":"} else if"}*

Dosyadan metin okunduktan sonra şöyle bir döngü kullanılarak bütün genel ifadeler değiştirebilir.

*for key in equ.keys():*

*for line in text:*

*text[text.index(line)] = line.replace(key,equ[key])*

Ancak bu yöntem aşağıda gösterilen bazı özel değişikleri yapamamaktadır.

1. Değişken tanımlarında i number; -> var i; gibi dönüşümler. (Bu döngünün üreteceği i var; çıktısı hatalı olacaktır.)
2. Varchar değişkenleri dizge boyutuyla tanımlandıkları ve replace metodu sadece eşit stringleri değiştirebildiği için döngü bu dönüşümü yapamamaktadır.
3. loop ifadesi yalnız başına kullanıldığında ve for veya while ile birlikte kullandığında farklı değişiklikler gerekmektedir. Bunun ayrımını yapamadan döngü sadece bir dönüşüm yapmaktadır.
4. Son olarak döngü for ifadeleri arasındaki farkı çevirememektedir.

Bu durumlar için geliştirdiğim özel çözümler şunlardır. Programın ana döngüsü sözlük anahtarını dolaştığı için bu durumlar için kodda normalde bulunamayacak fason anahtarlar tanımladım.

1- Bu sorunun çözümü için satır stringini boşluklara göre ayırdıktan sonra oluşan dizinin son elemanından “;\n” karakterlerinin sildikten sonra diziyi ters çevirip son elemanına bu karakterleri ekledim ve stringi geri birleştirdim.

2- Bu sorunun çözümü için varchar tanımı varchar2(boyut) gibi bir düzen içerdiği için düzenli ifadeleri kullandım.

Bu iki sorunun çözümünü içeren kod bloğu

*if (equ[key] == "var"):*

*for line in text:*

*if (key == "varchar"):*

*reg = re.compile("varchar2\(.\*\);")*

*tline = reg.sub("var",line)*

*elif (key in ["boolean;","date;","number;"]):*

*tline = line.replace(key,equ[key])*

*if ("var" in tline and tline not in check):*

*temp = tline.split(" ")*

*temp[-1] = temp[-1].rstrip(";\n")*

*temp.reverse()*

*temp[-1] = temp[-1] + ";\n"*

*test = " ".join(temp)*

*check.append(test)*

*text[text.index(line)] = " ".join(temp)*

3- Bu sorunun çözümü için kod içinde “loop” ile başlayan, “end loop” içeren ve “exit” içeren satırların yerlerini birer diziye kaydetmem gerekti. Bulunan “her loop”a en yakın “end loop” ifadesini bulduktan sonra gerekli değişikleri yaptım.

Bu çözümü içeren kod bloğu

*elif (key == "do"):*

*t1 = 0*

*t0 = 0*

*for line in text:*

*i = text.index(line)*

*if (line.startswith("loop")):*

*ch0.append(i)*

*if ("end loop" in line):*

*ch.append(i)*

*if ("exit" in line):*

*chh.append(i)*

*for t1 in ch0:*

*temp = [i for i in ch if i>t1]*

*j=1000*

*for i in temp:*

*if i-t1 < j:*

*j=i*

*text[t1] = "do {\n"*

*text[j] = "} while(true);\n"*

*for t0 in chh:*

*temp = text[t0].replace("exit when","")*

*temp = temp.rstrip(";\n")*

*text[t0] = "if (%s) break;\n" %(temp)*

4- Bu sorunun çözümü için koddaki bütün “for” ifadelerinin satırlarını bir diziye kaydettim. Ardından bazı string işlemleri ile istenen formata ulaştım.

Örneğin “for i in 0..10 loop” için bu stringi boşluklara göre ayırıp oluşan diziden “for”,”in” ve”loop” ifadelerini çıkarınca ilk eleman döngü değişkenini ve ikinci elemanı “..” ile ayırınca oluşan dizi de sınırları gösterecektir.

Bu çözümü içeren kod bloğu:

*elif (key == "for"):*

*lcount = []*

*for line in text:*

*i = text.index(line)*

*if (line.startswith("for")):*

*lcount.append(i)*

*for i in lcount:*

*temp = text[i].split(" ")*

*temp[-1] = temp[-1].rstrip("\n")*

*temp.remove("for")*

*temp.remove("in")*

*temp.remove("loop")*

*temp[1] = temp[1].split("..")*

*test = "%s=%s;%s<%s;%s++" %(temp[0],temp[1][0],temp[0],temp[1][1],temp[0])*

*text[i] = "for(%s) {\n" %test*

Programın kaynak kodları:

*#!/usr/bin/python*

*# -\*- coding: utf-8 -\*-*

*import re*

*import sys*

*def main():*

*equ = { "number;":"var","varchar":"var","date;":"var","boolean;":"var"," =":" ==","<>":"!=","||":"+","--":"//","and":"&&","not ":"!",*

*"then":"{","else":"} else {","end if;":"}","loop":"{","end loop;":"}","do":"","for":"for","declare":"","begin":"","end;":"",*

*"dbms\_output.put\_line":"console.log"}*

*equlater = { ":=":"=" ,"elsif":"} else if"}*

*arg = sys.argv[1:]*

*if ("-o" in arg):*

*ina = arg[0]*

*outa = arg[2]*

*else:*

*ina = arg[0]*

*outa = "output.txt"*

*try:*

*inf = open(ina,"r")*

*except IOError:*

*print ina,"No such file exits"*

*sys.exit()*

*outf = open(outa,"w")*

*text = inf.readlines()*

*for key in equ.keys():*

*check = []*

*chh = []*

*ch0 = []*

*ch = []*

*if (equ[key] == "var"):*

*for line in text:*

*if (key == "varchar"):*

*reg = re.compile("varchar2\(.\*\);")*

*tline = reg.sub("var",line)*

*elif (key in ["boolean;","date;","number;"]):*

*tline = line.replace(key,equ[key])*

*if ("var" in tline and tline not in check):*

*temp = tline.split(" ")*

*temp[-1] = temp[-1].rstrip(";\n")*

*temp.reverse()*

*temp[-1] = temp[-1] + ";\n"*

*test = " ".join(temp)*

*check.append(test)*

*text[text.index(line)] = " ".join(temp)*

*elif (key == "do"):*

*t1 = 0*

*t0 = 0*

*for line in text:*

*i = text.index(line)*

*if (line.startswith("loop")):*

*ch0.append(i)*

*if ("end loop" in line):*

*ch.append(i)*

*if ("exit" in line):*

*chh.append(i)*

*for t1 in ch0:*

*temp = [i for i in ch if i>t1]*

*j=1000*

*for i in temp:*

*if i-t1 < j:*

*j=i*

*text[t1] = "do {\n"*

*text[j] = "} while(true);\n"*

*for t0 in chh:*

*temp = text[t0].replace("exit when","")*

*temp = temp.rstrip(";\n")*

*text[t0] = "if (%s) break;\n" %(temp)*

*elif (key == "for"):*

*lcount = []*

*for line in text:*

*i = text.index(line)*

*if (line.startswith("for")):*

*lcount.append(i)*

*for i in lcount:*

*temp = text[i].split(" ")*

*temp[-1] = temp[-1].rstrip("\n")*

*temp.remove("for")*

*temp.remove("in")*

*temp.remove("loop")*

*temp[1] = temp[1].split("..")*

*test = "%s=%s;%s<%s;%s++" %(temp[0],temp[1][0],temp[0],temp[1][1],temp[0])*

*text[i] = "for(%s) {\n" %test*

*else:*

*for line in text:*

*text[text.index(line)] = line.replace(key,equ[key])*

*for key in equlater.keys():*

*for line in text:*

*text[text.index(line)] = line.replace(key,equlater[key])*

*outf.writelines(text)*

*print ina,"file has successfully translated to",outa*

*inf.close()*

*outf.close()*

*if \_\_name\_\_ == '\_\_main\_\_':*

*main()*

Programın Kullanılması:

Bu program bir CLI uygulamasıdır ve giriş için komut satır argümanlarını kullanır. Python programları ikili dosya kullanmadan kaynak kod üzerinden çalışır.

Çalıştırmak için

*python program\_090202003\_080202039.py giris.txt*

komutu kullanılır ve çıktıyı aynı dizine output.txt olarak verir. Çıktı dosyasının adını belirlemek için programı

*python program\_090202003\_080202039.py giris.txt -o cikis.txt*

şeklinde çalıştırmak gerekir.