

日日是Oracle APEX

Oracle APEXを使った作業をしていて、気の付いたところを忘れないようにメモをとります。

2020年7月21日 火曜日

Oracle APEXでのデータ・ロード(5) - XMLパーサー

CSV形式およびExcel形式のファイルをデータベースにロードするために、APEX_DATA_PARSERパッケージを使用しました。現在は多くの自治体が、これらの形式で新型コロナウイルス感染症の陽性患者属性のデータを提供しています。とはいえ、すべての自治体ではありません。HTMLの表としてデータが公開されているケースも多いです。

これからHTMLの表をOracle APEXで扱えるよう、データベースの表に取り込むコードを紹介します。HTML表のパーズにはOracle Databaseが提供しているXMLパーサーの機能を利用します。APEX_DATA_PARSERパッケージのPARSEファンクションはXMLファイルも扱うことができます。しかし、HTMLページ自体はXMLではなく、また、一連の文章の中に表が含まれていたり、1ページにいくつものHTML表があったりします。そのため、取り込むHTML表を含んだページから、読み込みの対象部分を切り出したり、不要なタグなどを取り除くなど、前処理が必要です。結果として、APEX_DATA_PARSERパッケージのPARSEファンクションでは扱いにくくなっています。

HTMLページのロード

CSVおよびExcelと同様に、バイナリ・データとしてデータベースに保存します。これらはキャッシュの役割になります。

秋田県を例にとって説明します。秋田県のデータは以下のURLで公開されています(2020年7月21日現在)。

<https://www.pref.akita.lg.jp/pages/archive/47957>

このページをそのままの形でデータベースに保存するためのコードは、CSVやExcel形式の場合とまったく同じです。

```
declare
  l_url covid19_municipalities.content_url%type;
  l_file_name covid19_municipalities.file_name%type;
begin
  l_file_name := 'akita.html';
  l_url := 'https://www.pref.akita.lg.jp/pages/archive/47957';
  update covid19_municipalities
  set content_blob = apex_web_service.make_rest_request_b(l_url, 'GET'),
      content_url = l_url,
      file_name = l_file_name,
      last_update_date = systimestamp
  where name = '秋田県';
end;
```

ファイル名として与えることのできるデータが無い場合、ここではファイル名をakita.htmlとしています。

HTML表を読み込んで表データを返す、APEX_DATA_PARSER.PARSEファンクションと等価な表関数を作成します。使用する関数はXMLTABLEです。

マニュアルの説明は[こちら](#)になります。

まず最初に、BLOBで保存したHTMLページをCLOBで取り出す必要があります。その処理を行うために、以下のファンクションを作成しました。

```
create or replace function get_content_as_clob
(
  p_prefecture_name varchar2,
  p_charset          varchar2
)
return clob
is
  l_blob blob;
  l_clob clob;
  ls integer := 1;
  le integer := 1;
  l_lang_ctx integer := DBMS_LOB.DEFAULT_LANG_CTX;
  l_warning integer;
begin
  select content_blob into l_blob from covid19_municipalities where name = p_prefecture_name;
  if dbms_lob.getlength(l_blob) = 0 then
    return null;
  end if;
  dbms_lob.createtemporary(l_clob, true, dbms_lob.call);
  dbms_lob.converttocab(
    l_clob,
    l_blob,
    dbms_lob.lobmaxsize,
    ls,
    le,
    NLS_CHARSET_ID(p_charset),
    l_lang_ctx,
    l_warning);
  return l_clob;
end get_content_as_clob;
```



```
select td from xmltable('tbody/tr' passing xmltype(l_html)
                        columns td xmltype path './td')
```

以下のような、TD要素の(TR要素毎の)集まりとなります。

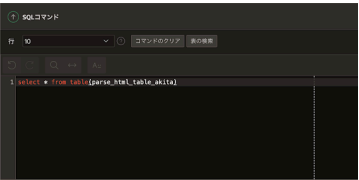
```
<td style="text-align: center;">16例目</td><td>4月14日</td><td>50歳代</td><td>男性</td><td>湯沢保健所管内</td><td>会社員</td><td>調査終了</td><td> </td>
<td style="text-align: center;">15例目</td><td>4月11日</td><td>10歳代</td><td>男性</td><td>秋田市</td><td>中学生</td><td>調査終了</td><td>秋田市6例目、県内
<td style="text-align: center;">14例目</td><td>4月11日</td><td>10歳代</td><td>女性</td><td>秋田市</td><td>高校生</td><td>調査終了</td><td>秋田市5例目、県内
<td style="text-align: center;">13例目</td><td>4月10日</td><td>50歳代</td><td>男性</td><td>大仙保健所管内</td><td>自営業</td><td>調査終了</td><td> </td>
```

続くSELECT文で、一行分の集まりより、このセルのデータを取り出します。

```
open c_record for
select cell from xmltable('./td' passing r.td
                        columns cell varchar2(200) path '.');
```

SQLコマンドにて動作確認を行います。

```
select * from table(parse_html_table_akita);
```



The screenshot shows a SQL command window with the following content:

```
SQLコマンド
1: select * from table(parse_html_table_akita);
```

Below the command window, a table is displayed with the following data:

COL001	COL002	COL003	COL004	COL005	COL006	COL007	COL008	COL009
16例目	4月14日	50歳代	男性	湯沢保健所管内	会社員	調査終了		
15例目	4月11日	10歳代	男性	秋田市	中学生	調査終了	秋田市内6例目、県内は例目の調査対象外	
14例目	4月11日	10歳代	女性	秋田市	高校生	調査終了	秋田市内5例目、県内は例目の調査対象外	
13例目	4月10日	50歳代	男性	大仙保健所管内	自営業	調査終了		

SELECT文で表形式のデータが取得できたので、後はCSVやExcel形式と同様にMERGE文(もしくはDELETE/INSERT文)によって表COVID19_PATIENTS表に読み込みます。秋田県の例では以下になります。

```
merge into covid19_patients p
using
(
select
to_number(regexp_replace(col001, '^s*(\d+)例目','\1')) "No",
50008 municipality_code,
'秋田県' prefecture_name,
null city_name,
to_date(col002, 'MM"月"DD"日"') published_date,
null onset_date,
col005 patient_location,
case
when regexp_like(col003, '^d+歳代') then
replace(col003, '歳')
else
col003
end patient_age,
col004 patient_sex,
col006 patient_occupation,
null patient_status,
null patient_symptom,
null patient_travel_history,
null patient_left_hospital,
col008 remark
from table(parse_html_table_akita)
minus
select
"No", municipality_code, prefecture_name, city_name,
published_date, onset_date,
patient_location, patient_sex, patient_occupation,
patient_status, patient_symptom,
patient_travel_history, patient_left_hospital, remark
from covid19_patients
) n
on (p."No" = n."No" and p.prefecture_name = n.prefecture_name)
when matched then
update set
p.city_name          = n.city_name,
p.published_date     = n.published_date,
p.onset_date         = n.onset_date,
p.patient_location   = n.patient_location,
p.patient_age        = n.patient_age,
p.patient_sex        = n.patient_sex,
p.patient_occupation = n.patient_occupation,
p.patient_status     = n.patient_status,
p.patient_symptom    = n.patient_symptom,
p.patient_travel_history = n.patient_travel_history,
p.patient_left_hospital = n.patient_left_hospital,
p.remark             = n.remark
when not matched then
insert(
```

```
"No", municipality_code, prefecture_name, city_name,
published_date, onset_date,
patient_location, patient_age, patient_sex, patient_occupation,
patient_status, patient_symptom,
patient_travel_history, patient_left_hospital, remark
)
values
(
  n."No", n.municipality_code, n.prefecture_name, n.city_name,
  n.published_date, n.onset_date,
  n.patient_location, n.patient_age, n.patient_sex, n.patient_occupation,
  n.patient_status, n.patient_symptom,
  n.patient_travel_history, n.patient_left_hospital, n.remark
);
```

長いSQLになっていますが、以下のSELECT文だけがHTML表、および、秋田県に対応した部分で、それ以外はCSVやExcel形式の読み込みと同じです。他の自治体のHTML表についても、`parse_html_table_akita`表関数と同等の表関数を作成することで、データの取り込みが可能になります。

```
select
  to_number(regexp_replace(col001, '^\s*(\d+)例目','\1')) "No",
  50008 municipality_code,
  '秋田県' prefecture_name,
  null city_name,
  to_date(col002, 'MM"月"DD"日"') published_date,
  null onset_date,
  col005 patient_location,
  case
    when regexp_like(col003, '^\d+歳代') then
      replace(col003, '歳')
    else
      col003
  end patient_age,
  col004 patient_sex,
  col006 patient_occupation,
  null patient_status,
  null patient_symptom,
  null patient_travel_history,
  null patient_left_hospital,
  col008 remark
from table(parse_html_table_akita)
```

データの取り込みについての解説は以上で終了です。次は取り込んだデータをCSV形式、または、JSON形式で出力する方法について紹介します。

参考情報

HTML表としてデータを公開している自治体

秋田県、茨城県、京都府、鳥取県、広島県、徳島県、香川県、佐賀県、宮崎県、鹿児島県。ただし、広島県は居住地、年代、性別などの属性をHTML表に含まないケースが多いです。

CSV、Excel、HTML表以外の形式の自治体

PDFで公開している自治体は、群馬県、千葉県、愛知県、沖縄県。滋賀県、和歌山県、島根県については、陽性患者属性の情報は症例ごとに提供。

続く

Yuji N. 時刻: 16:42

共有

◀

ホーム

▶

[ウェブ バージョンを表示](#)

自己紹介

Yuji N.

日本オラクル株式会社に勤務していて、Oracle APEXのGroundbreaker Advocateを拝命しました。こちらの記事につきましては、免責事項の参照をお願いいたします。

[詳細プロフィールを表示](#)