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。

2021年5月28日金曜日

改行で区切ったJSON(Newline Delimited JSON)のデータをロードする

政府CIOポータルのワクチン接種ダッシュボードのページより、都道府県別接種回数詳細のデータがオープンデータとして、NDJSON - 改行区切りJSON形式でダウンロードできるようになっています。

最新のOracle APEX - 21.1でもNDJSONのロードはサポートしていないため、ちょっとコードを書いてロードしてみました。

5月28日現在ですが、都道府県別接種回数詳細のデータをこちらからダウンロードすることができました。このデータを取り込むので、あらかじめダウンロードしておきます。ダウンロードしたファイル名はprefecture.ndjsonでした。

以下、作業ログになります。

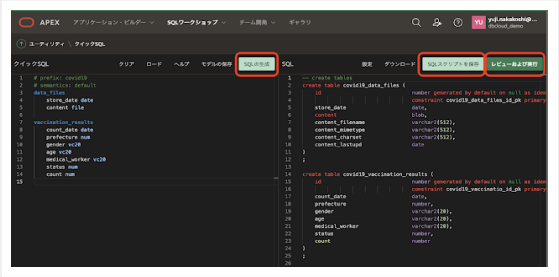
データを取り込む表の定義と、取り込んだデータを扱うアプリケーションの作成を行います。

クイックSQLを開き、以下のモデルより表COVID19_DATA_FILES、COVID19_VACCINATION_RESULTSを作成します。

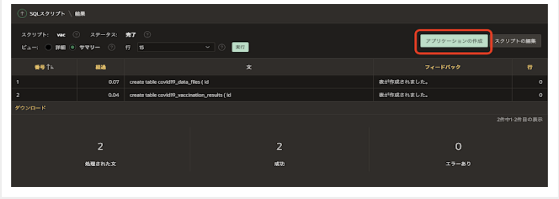
```
# prefix: covid19
# semantics: default
data_files
    store_date date
    content file

vaccination_results
    count_date date
    prefecture num
    gender vc20
    age vc20
    medical_worker vc20
    status num
    count num
```

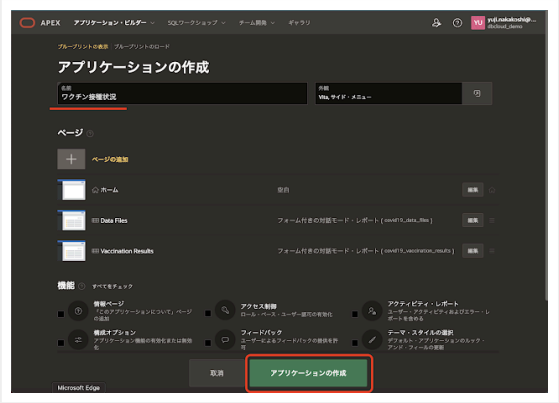
SQLの生成、SQLスクリプトの保存、レビューおよび実行を続けて行い、SQLをレビューしたのち**即時実行**まで実施します。



表が作成されたら、**アプリケーションの作成**を実行します。確認画面が開くので、そこでも**アプリケーションの作成**をクリックします。



アプリケーションの**名前**を**ワクチン接種状況**とし、**アプリケーションの作成**をクリックします。



アプリケーションを実行し、ダウンロードしてあるファイルprefecture.ndjsonをデータベースに保存します。メニューから**Data Files**のページを開き、**作成**をクリックします。

Contentにダウンロードした**prefecture.ndjson**を指定し、**作成**をクリックします。

行が一行追加されていることがわかります。レポートとして調整できていませんが、対応は後回しとします。

NDJSONには対応していませんが、Oracle APEXではJSON形式のデータのロードには対応しています。ですのでNDJSONに含まれる1行のJSONを対象にして、Oracle APEX 21.1の新機能を使ったデータ・ロード定義を作ります。

prefecture.ndjsonに含まれる1行を**test.json**として、ファイルに切り出します。例えば、以下のようにheadコマンドを使うことができます。

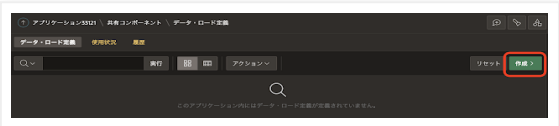
```
% head -1 prefecture.ndjson > test.json
```

任意の一行なので、ファイルの内容をコピペして作成することもできます。

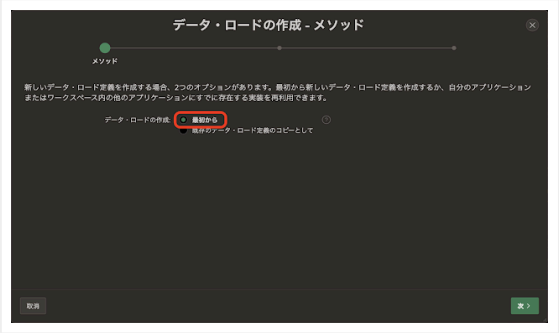
```
{"date":"2021-04-12","prefecture":"01","gender":"F","age":"65-
","medical_worker":false,"status":1,"count":84}
```

サンプルとして使用するファイルが作成されたら、**共有コンポーネント**の**データ・ロード定義**を開きます。

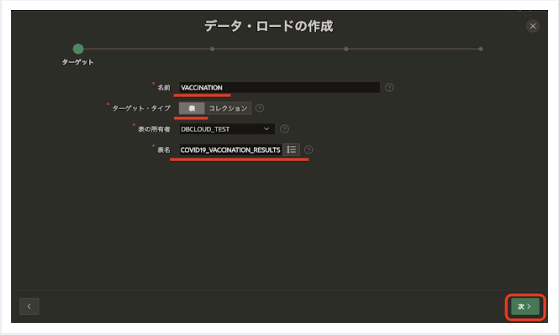
データ・ロード定義の一覧画面より、**作成**をクリックします。



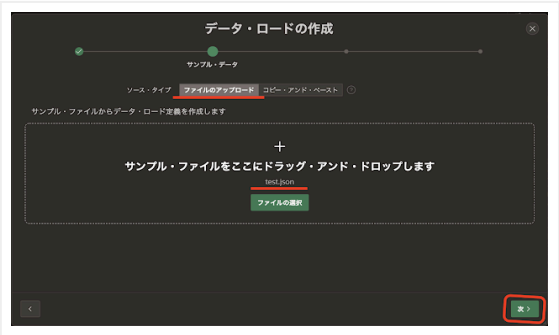
データ・ロードの作成は**最初から**行います。



データ・ロード定義の**名前**として**VACCINATION**を指定します。**ターゲット・タイプ**は**表**、**表名**に**COVID19_VACCINATION_RESULTS**を指定します。**次**へ進みます。



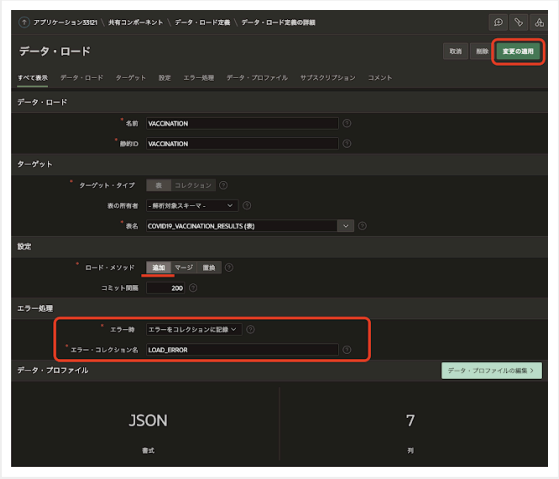
ソース・タイプとして**ファイルのアップロード**を選択し、**ファイルの選択**をクリックして**test.json**を指定します。**次**へ進みます。



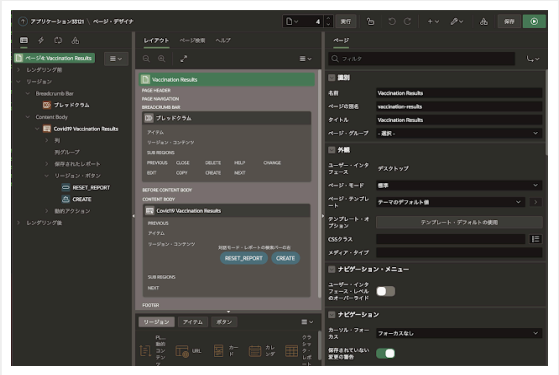
DATE_(Date)の**マップ先**が決まっていないので、**COUNT_DATE (Date)** を選びます。**データ・ロードの作成**をクリックします。ページの作成は行いません。

データ・ロード定義が作成されました。開いて内容を編集します。

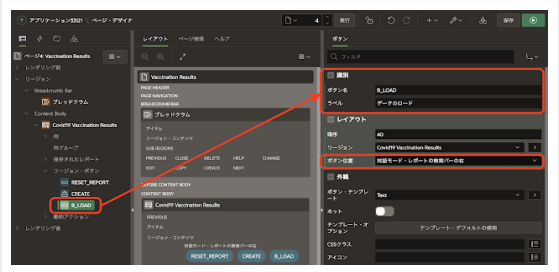
静的IDは**VACCINATION**です。これはデータ・ロードを行うプロシージャの呼び出し時に指定します。次に**ロード・メソッド**が**追加**になっていることを確認します。主キーの設定をしていないため、デフォルトは追加です。エラー処理の**エラー時**として**エラーをコレクションに記録**を選択し、**エラー・コレクション名**に**LOAD_ERROR**を指定します。**変更の適用**をクリックします。



これでデータをロードする準備が整いました。**ページ番号４のVaccication Results**のページを開きます。



データのロードを実行するボタンを追加します。リージョン・ボタンの上でコンテキスト・メニューを表示させ、**ボタンの作成**を実行します。**識別**の**名前**を**B_LOAD**とし、**ラベル**は**データのロード**とします。**ボタンの位置**として**対話モード・レポートの検索バーの右**を指定します。

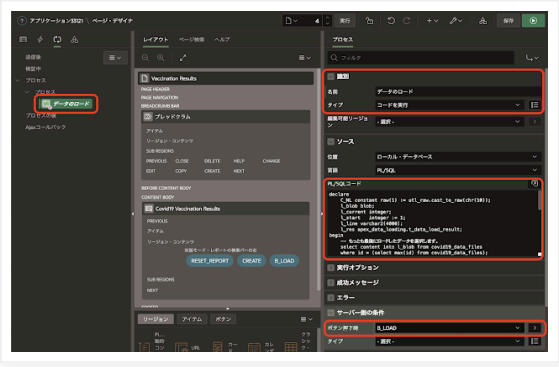


ボタンB_LOADをクリックしたときに実行されるプロセスを作成します。PL/SQLコードは以下を使います。

```
declare
    C_NL constant raw(1) := utl_raw.cast_to_raw(chr(10));
    l_blob blob;
    l_tmp_blob blob;
    l_current integer;
    l_start integer := 1;
    l_line varchar2(32767);
    l_res covid19_vaccination_results%rowtype;
    l_vres apex_data_loading.t_data_load_result;
    l_total integer := 0;
```

```
begin
    -- もっとも最後にロードしたデータを選択します。
    select content into l_blob from covid19_data_files
    where id = (select max(id) from covid19_data_files);
    -- 一旦メモリにロードします。
    dbms_lob.createtemporary(l_tmp_blob, true);
    dbms_lob.copy(
        dest_lob => l_tmp_blob,
        src_lob => l_blob,
        amount => dbms_lob.lobmaxsize
    );
    -- データは入れ替えます。
    delete from covid19_vaccination_results;
    -- JSONのデータを1行ごとに取り出し、APEX_DATA_LOADING.LOAD_DATAを呼び出します。
    while true
    loop
        -- 改行位置を見つける。
        l_current := dbms_lob.instr(l_tmp_blob, C_NL, l_start);
        -- 一行を取り出す。
        l_line := utl_raw.cast_to_varchar2(
            dbms_lob.substr(l_tmp_blob, (l_current - l_start), l_start)
        );
        -- 改行が見つからなければ終了。
        -- ファイルの最終行でも改行がある - いきなりEOFにはならないのが前提。
        exit when (l_current = 0);
        l_vres := apex_data_loading.load_data
        (
            p_static_id => 'VACCINATION',
            p_data_to_load => l_line
        );
        l_total := l_total + l_vres.processed_rows;
        -- 次の行の処理へ移る。
        l_start := l_current + 1;
    end loop;
    apex_debug.info('Total Number of COVID19_VACCINATIION_RESULTS: ' || l_total);
    dbms_lob.freetemporary(l_tmp_blob);
end;
```

プロセスの作成を実行し、**識別**の**名前**を**データのロード**とします。**タイプ**は**コードを実行**、**ソース**の**PL/SQLコード**として、上記のコードを記述します。**サーバー側の条件**の**ボタン押下時**に**B_LOAD**を指定します。



ページを実行し、**データのロード**をクリックします。

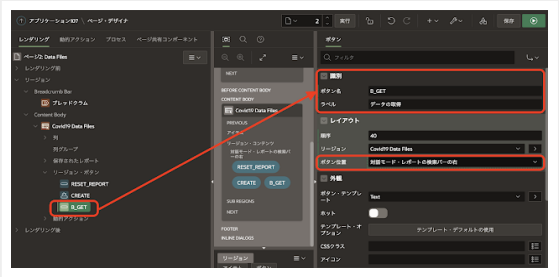
残念ですが、apex.oracle.comでは、リソース・マネージャの制限でエラーが発生するため、最後までデータのロードができません。手元のデータベースにOracle APEX 21.1を導入した環境では成功しています。(Temporary LOBを使用することで、こちらのエラーは回避できました。)

毎回データ・ロード定義を参照するAPEX_DATA_LOADING.LOAD_DATAを呼び出すのは、さすがに時間がかかります。以下のように、直接JSONをパースするコードを記載すると処理速度は向上します。ただし、データのフォーマット変更にはコード変更で対応する必要があります。

```
declare
    C_NL constant raw(1) := utl_raw.cast_to_raw(chr(10));
    l_blob blob;
    l_tmp_blob blob;
    l_current integer;
    l_start integer := 1;
    l_line varchar2(32767);
    l_vres covid19_vaccination_results%rowtype;
    l_json json_object_t;
    l_total integer := 0;
begin
    -- もっとも最後にロードしたデータを選択します。
    select content into l_blob from covid19_data_files
    where id = (select max(id) from covid19_data_files);
    -- 一旦メモリにロードします。
    dbms_lob.createtemporary(l_tmp_blob, true);
    dbms_lob.copy(
        dest_lob => l_tmp_blob,
        src_lob => l_blob,
        amount => dbms_lob.lobmaxsize
    );
    -- データは入れ替えます。
    delete from covid19_vaccination_results;
    -- JSONのデータを1行ごとに取り出し、JSONを処理します。
    while true
    loop
        -- 改行位置を見つける。
        l_current := dbms_lob.instr(l_tmp_blob, C_NL, l_start);
        -- 1行のJSON形式のデータを取り出す。
        l_line := utl_raw.cast_to_varchar2(
            dbms_lob.substr(l_tmp_blob, (l_current - l_start), l_start)
        );
        -- 改行が見つからなければ終了。
        -- ファイルの最終行でも改行がある - いきなりEOFにはならないのが前提。
        exit when (l_current = 0);
        -- ロードする - 直接JSONを使う。
        l_json := json_object_t.parse(l_line);
        l_vres.count_date := to_date(l_json.get_String('date'),'YYYY-MM-DD');
```

```
        l_vres.prefecture := l_json.get_Number('prefecture');
        l_vres.gender     := l_json.get_String('gender');
        l_vres.age        := l_json.get_String('age');
        l_vres.medical_worker := l_json.get_String('medical_worker');
        l_vres.status := l_json.get_Number('status');
        l_vres.count := l_json.get_Number('count');
        insert into covid19_vaccination_results values l_vres;
        l_total := l_total + 1;
        -- 次の行の処理へ移る。
        l_start := l_current + 1;
    end loop;
    apex_debug.info('Total Number of COVID19_VACCINATIION_RESULTS: ' || l_total);
    dbms_lob.freetemporary(l_tmp_blob);
end;
```

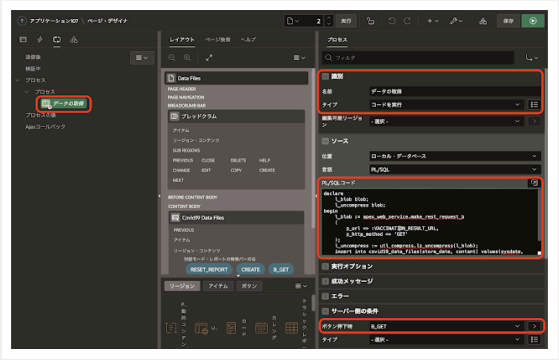
データの取得を一度にできるように、Data Filesのページにボタンを追加します。ページ番号2のData Filesのページを開きます。**リージョン・ボタン**の上でコンテキスト・メニューを開き、**ボタンの作成**を実行します。**識別**の**ボタン名**を**B_GET**とし、**ラベル**を**データの取得**とします。**ボタン位置**は**対話モード・レポートの検索バーの右**を指定します。



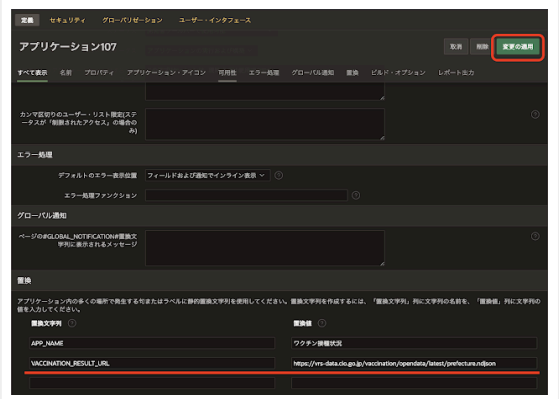
データの取得には以下のコードを使用します。

```
declare
    l_blob blob;
    l_uncompress blob;
begin
    l_blob := apex_web_service.make_rest_request_b
    (
        p_url => :VACCINATION_RESULT_URL,
        p_http_method => 'GET'
    );
    l_uncompress := utl_compress.lz_uncompress(l_blob);
    insert into covid19_data_files(store_date, content) values(sysdate, l_uncompress);
end;
```

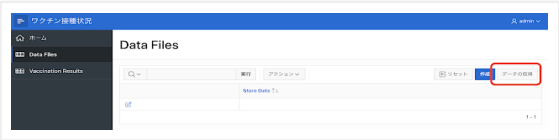
プロセスの作成を行います。**識別**の**名前**を**データの取得**、**タイプ**を**コードの実行**とします。**ソース**の**PL/SQLコード**として上記を記述し、**サーバー側の条件**の**ボタンの押下時**として**B_GET**を指定します。



コード内でデータを取得するURLを**VACCINATION_RESULT_URL**としているので、これを**アプリケーション定義**の**置換文字列**として設定します。



Data Filesのページを実行し、**データの取得**を実行します。

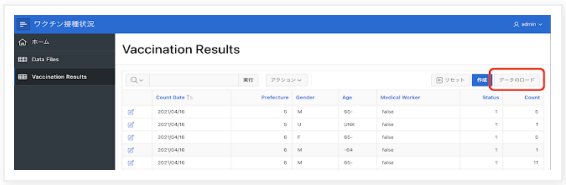


ネットワークのアクセスに関するエラー(ORA-29273: HTTPリクエストに失敗しました)が発生するときは、ACLの追加を行ないます。ACLの追加を行うコードの例は以下になります。

```
begin
  dbms_network_acl_admin.create_acl(acl => 'vrs.xml',
                                    description => 'Vaccine Dashboard',
                                    principal => 'APEX_210100',
                                    is_grant  => true,
                                    privilege => 'connect');
  dbms_network_acl_admin.assign_acl(acl => 'vrs.xml',
                                    host => 'vrs-data.cio.go.jp');
end;
/
commit;
```

データの取得が成功すると表示が一行増え、Store Dateにデータを取得した日付も表示されます。

Vaccination Resultsのページを開き、**データのロード**を実行すると、直近で取得したデータ(Data Filesのページで**データの取得**を行なったデータ)で入れ替わります。



以上で今回目的としていた、NDJSON形式のデータ・ロードを行うアプリケーションは完成です。
データの取得とロードはほぼPL/SQLでコーディングしているため、Oracle APEXの自動化を使って実行させるように変更するのも容易でしょう。

データさえ表に取り込めば、色々なレポーティングを実装することができます。

今回作成したアプリケーションのエクスポートを以下に置きました。
https://github.com/ujnak/apexapps/blob/master/exports/load-ndjson.sql

Oracle APEXのアプリケーション作成の参考になれば幸いです。

完

Yuji N. 時刻: 19:00

共有