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。

2022年1月11日火曜日

APEXビューの比較をするPL/SQLスクリプト

先日、APEXのリポジトリに対する変更の有無を確認するスクリプトを書いてみました。少々気になったので、それぞれのビューの変更点を見つけるスクリプトの書き方について、調べてみました。

ビューの親子関係の情報（これはビューAPEX_DICTIONARYから得られる）と、行を一意で認識する列の情報（これは簡単に得る方法はない - ただ、ビューの定義を見ると分かる）は必要でした。

```
declare
    CRLF constant varchar2(2) := chr(13) || chr(10);
    l_sql_old varchar2(32767);
    l_sql_new varchar2(32767);
    l_column_list varchar2(32767);
    l_column_list_join varchar2(32767);
    l_sql    clob;
    l_cursor  integer;
    l_result  integer;
    desctab   DBMS_SQL.DESC_TAB;
    colcnt    NUMBER;
    namevar   VARCHAR2(4000);
    namevar_old varchar2(4000);
    namevar_new varchar2(4000);
    numvar    NUMBER;
    numvar_old number;
    numvar_new number;
    datevar   DATE;
    datevar_old date;
    datevar_new date;
    l_op varchar2(1);
    type col_t is table of varchar2(30) index by pls_integer;
    defcols col_t;
    j integer;
begin
    /*
     * スナップショットとして作成されている表を比較対象とする。
     * 手順の確認コードなので、APEX_APPLICATION_PAGE_REGIONSのみを対象とする。
     */
    for t in (
```

```
    select table_name from user_tables
    where table_name = 'B001_APPLICATION_PAGE_REGIONS'
)
loop
    l_column_list      := ''; -- サブクエリの列リスト
    l_column_list_join := ''; -- ジョイン後の列リスト
    /*
     * ジョイン後のSELECT文の列名のリストを配列DEFCOLSに保存する。
     */
    defcols.delete();
    j := 1;
    defcols(j) := 'OP'; -- 先頭の列は、行の操作モードとする。
    /*
     * 比較対象の表に定義されている列から、SELECT文を生成する。
     */
    for c in
    (
        select column_name, data_type from user_tab_cols
        where table_name = t.table_name order by column_id
    )
    loop
        /*
         * 同じ列名を2つずつ定義する。先に現れるのは変更前、次に変更後の列。もともと同じ列なので、
         * 列名も同じである。
         */
        j := j + 1;
        defcols(j) := c.column_name;
        j := j + 1;
        defcols(j) := c.column_name;
        -- 列名を追加するために , をアペンドする。
        if lengthb(l_column_list) > 0 then
            l_column_list := l_column_list || ',';
        end if;
        if lengthb(l_column_list_join) > 0 then
            l_column_list_join := l_column_list_join || ',';
        end if;
        -- データタイプに応じて比較方法を変更する。
        if c.data_type in ('VARCHAR2') then
            /*
             * 出力のデバッグをしやすくするために、ハッシュではなく文字列を直接比較する。
            l_column_list := l_column_list || 'ora_hash(' || c.column_name || ') '
                || c.column_name || chr(13) || chr(10);
            */
            l_column_list := l_column_list || c.column_name || CRLF;
        elsif c.data_type in ('CLOB','NCLOB','BLOB') then
            /*
             * 4000バイトを超えるとora_hashは使えない。重い処理だが、SHA256(引数4)のハッシュを取る。
```

```
            */
            l_column_list := l_column_list || 'case when ' || c.column_name || ' is null th
                || c.column_name || CRLF;
        else
            /*
             * 生データで比較する。
             */
            l_column_list := l_column_list || c.column_name || CRLF;
        end if;
        /*
         * ジョイン後の列リストは、変更前と変更後の列をそれぞれ選択する。
         * 変更前は添字o（スナップショットの表が対象）、現行は添字 n、（APEXビュー）
         */
        l_column_list_join := l_column_list_join || 'o.' || c.column_name || ',n.' || c.col
    end loop;
    /*
     * SELECT文を構築する。SELECT文が32Kバイトを超えるのでCLOBで保存する。
     * また、動的SQLでは実行できなくなるので、DBMS_SQLを使用する。
     */
    l_sql_old := 'select' || CRLF || l_column_list || 'from ' || t.table_name;
    l_sql_new := 'select' || CRLF || l_column_list || 'from ' || 'APEX' || substr(t.table_n
    l_sql := 'select' || CRLF
        || 'case when o.region_id is null then ''I''' || CRLF
        || 'when n.region_id is null then ''D''' || CRLF
        || 'else ''U'' end op,' || CRLF
        || l_column_list_join
        || 'from ('
        || CRLF || l_sql_old || CRLF || 'minus' || CRLF || l_sql_new || CRLF
        || ') o full outer join ('
        || CRLF || l_sql_new || CRLF || 'minus' || CRLF || l_sql_old || CRLF
        -- APEXビューの行を一意で認識する列の情報が必要。
        || ') n on o.application_id = n.application_id and o.page_id = n.page_id and o.regi
        || 'where' || CRLF
        -- 上位のコンポーネントとして追加、削除が行われている場合、そのコンポーネントに含まれる変更はレポートしない。(
        || '(o.application_id, o.page_id) in (select application_id, page_id from apex_appl
        || CRLF || ' or ' || CRLF
        || '(n.application_id, n.page_id) in (select application_id, page_id from apex_appl
        /*
         * 追加、削除されたアプリケーションや、同様に追加、削除されたページに含まれるリージョンはレポートの対象から除く
         * よりレポートが見やすくなる。
         */
    l_cursor := dbms_sql.open_cursor;
    dbms_sql.parse(l_cursor, l_sql, dbms_sql.native);
    l_result := dbms_sql.execute(l_cursor);
    dbms_sql.describe_columns(l_cursor, colcnt, desctab);

    -- Define columns:
```

```
FOR i IN 1 .. colcnt
LOOP
    IF desctab(i).col_type = 2 THEN
        DBMS_SQL.DEFINE_COLUMN(l_cursor, i, numvar);
    ELSIF desctab(i).col_type = 12 THEN
        DBMS_SQL.DEFINE_COLUMN(l_cursor, i, datevar);
    ELSE
        DBMS_SQL.DEFINE_COLUMN(l_cursor, i, namevar, 4000);
    END IF;
END LOOP;

-- Fetch rows with DBMS_SQL package:
WHILE DBMS_SQL.FETCH_ROWS(l_cursor) > 0
LOOP
    -- 行の操作を確認する。 I （挿入） D （削除） または U （更新）
    DBMS_SQL.COLUMN_VALUE(l_cursor, 1, namevar);
    l_op := namevar;
    dbms_output.put_line('Row Op = ' || l_op);
    --
    FOR i IN 2 .. colcnt
    LOOP
        -- 2行づつ処理をする。
        if mod(i,2) = 1 then
            continue;
        end if;
        /*
         * 変更のあった値を印刷する。
         */
        IF (desctab(i).col_type = 2) THEN
            DBMS_SQL.COLUMN_VALUE(l_cursor, i, numvar);
            numvar_old := numvar;
            DBMS_SQL.COLUMN_VALUE(l_cursor, i+1, numvar);
            numvar_new := numvar;
            if numvar_old is null and numvar_new is null then
                continue;
            elsif numvar_old = numvar_new then
                continue;
            else
                dbms_output.put_line(defcols(i) || ': ' || numvar_old || ' = ' || numva
            end if;
        ELSIF (desctab(i).col_type = 12) THEN
            DBMS_SQL.COLUMN_VALUE(l_cursor, i, datevar);
            datevar_old := datevar;
            DBMS_SQL.COLUMN_VALUE(l_cursor, i+1, datevar);
            datevar_new := datevar;
            if datevar_old is null and datevar_new is null then
                continue;
```

```
                    elsif datevar_old = datevar_new then
                        continue;
                    else
                        dbms_output.put_line(defcols(i) || ': ' || datevar_old || ' = ' || date
                    end if;
                ELSE
                    DBMS_SQL.COLUMN_VALUE(l_cursor, i, namevar);
                    namevar_old := namevar;
                    DBMS_SQL.COLUMN_VALUE(l_cursor, i+1, namevar);
                    namevar_new := namevar;
                    if namevar_old is null and namevar_new is null then
                        continue;
                    elsif namevar_old = namevar_new then
                        continue;
                    else
                        dbms_output.put_line(defcols(i) || ': ' || namevar_old || ' = ' || name
                    end if;
                END IF;
            END LOOP;
        END LOOP;
        dbms_sql.close_cursor(l_cursor);
        /*
        dbms_output.put_line(substr(l_sql,1,30000));
        dbms_output.put_line(substr(l_sql,30000));
        */
    end loop;
end;
```

apex_compare_sample.sql hosted with ❤ by GitHub view raw

以上になります。

Oracle APEXのアプリケーション開発の参考になれば幸いです。

完

Yuji N. 時刻: 14:27

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