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12月25日金曜日

21cのブロックチェーン 表を使ってみる

Oracle APEXに直接は関係しないのですが、21cの新機能であるブロックチェーン表を使ってみました。その作業を記録します。

最初にブロックチェーン表を作りました。以下のDDLです。

```
create blockchain table BC_HOUSEHOLD_EXPENSES
(
   expense_id number generated by default on null as identity,
   place_date date not null,
   flex_column json
)
no drop until 31 days idle no delete locked HASHING USING "SHA2_512" version "v1"
partition by range(place_date)
interval(numtoyminterval(1, 'MONTH'))
(
   partition start_part values less than (to_date('1-12-2020','DD-MM-YYYY'))
);
```

ブロックチェーン表は一旦作成すると表定義を変更することができないため、データについては、これも21cの新機能であるネィティブのJSON型の列に保存することにします。(21c以前はVARCHAR2、CLOB、BLOB型にJSONを保存していました。)

今回はJSON型そのものを使いたいわけではなく、表定義の変更をできるようにしたいだけです。なので、作成したブロックチェーン表が普通の表に見えるようにビューを定義します。

```
create view bc_household_expenses_v
as
select expense_id, place_date, jt.*
from bc_household_expenses,
   json_table(flex_column, '$'
   COLUMNS(
      item varchar2(80) path '$.item',
      category varchar2(200) path '$.category',
      income number path '$.income',
      expense number path '$.expense',
      note varchar2(255) path '$.note'
   )
) as jt;
```

これで、EXPENSE_ID, PLACE_DATE, ITEM, CATEGORY, INCOME, EXPENSE, NOTEが列として見えるビューが定義できました。次に、このビューを対象としてINSERT文が実行できるよう、トリガーを定義します。

```
create or replace trigger bc_household_expenses_v_ins
instead of insert on bc_household_expenses_v
for each row
declare
  l_json json;
  l_jo json_object_t;
begin
  l_jo := json_object_t();
  if :new.item is not null then
    l_jo.put('item', :new.item);
  end if;
```

```
  if :new.category is not null then
    l_jo.put('category', :new.category);
  end if;
  if :new.income is not null then
    l_jo.put('income', :new.income);
  end if;
  if :new.expense is not null then
    l_jo.put('expense', :new.expense);
  end if;
  if :new.note is not null then
    l_jo.put('note', :new.note);
  end if;
  l_json := l_jo.to_json;
  insert into bc_household_expenses(expense_id, place_date, flex_column) values(:new.expense_id, :new.place_date, l_json);
end;
```

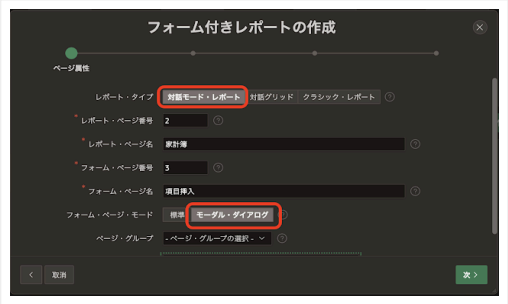
そもそもブロックチェーン表にたいして行える操作はINSERTだけなので、instead of insertのトリガーだけで十分です。

このビューを対象てにして、Oracle APEXのアプリケーションを作ってみます。まず、空っぽのアプリケーションを作成し、それにページを追加します。

追加するページは、**フォーム**です。

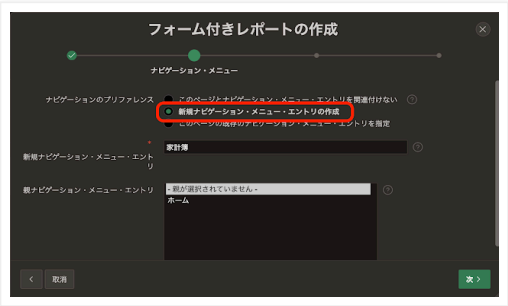
フォーム付きレポートを選びます。

レポート・タイプは**対話モード・レポート**、**フォーム・ページ・モード**は**モーダル・ダイアログ**を選びます。ちょっとした確認のためのアプリケーションなので、他は適当に設定し、次へ進みます。

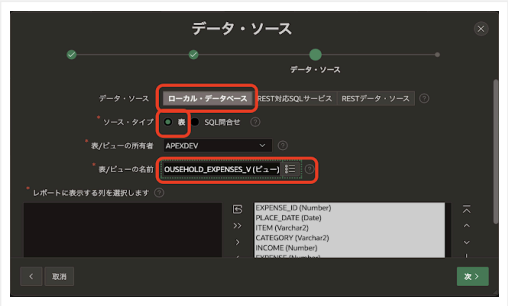


ナビゲーションのプリファレンスは、**新規ナビゲーション・メニュー・エントリの作成**を選びます。

次に進みます。

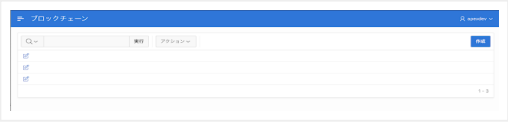


データ・ソースは**ローカル・データベース**、**ソース・タイプ**は（実際にはビューを選択しますが）**表**を選びます。**表/ビューの名前**として先ほど作成したビュー**BC_HOUSEHOLD_EXPENSES_V**を指定します。次に進みます。

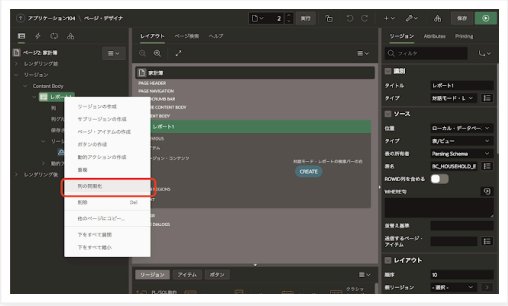


主キー列に**EXPENSE_ID**を選択し、作成します。

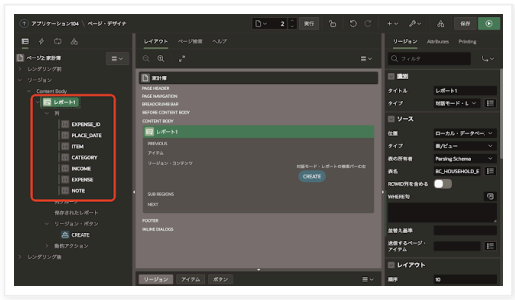
これでページが作成されたので実行してみます。



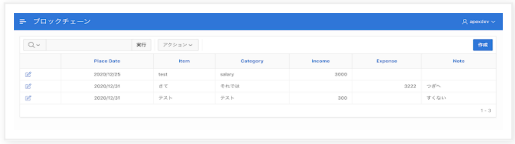
んん？列が正しく認識されていないようです。ページ・デザイナを開いて、作成されているレポートにたいして、**列の同期化**を実行します。



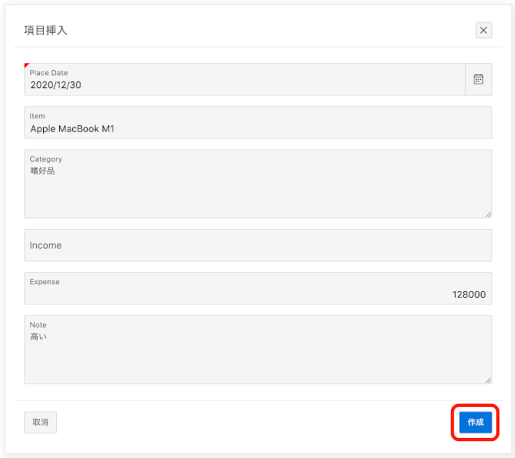
列の同期化を実行すると、列が認識されたことをページ・デザイナから確認できます。



再度ページを実行します。今度はレポートに列がそれぞれ表示されていることが確認できます。



作成したinstead of triggerによって行の挿入ができることを確認します。**作成**をクリックします。適当に値を設定し、**作成**をクリックします。

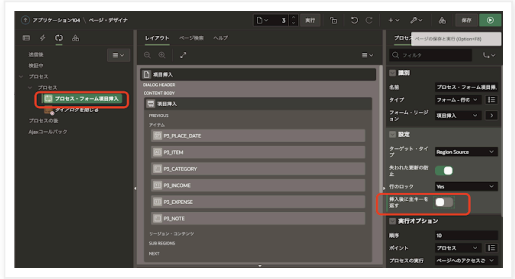


次のようなエラーがでるはずです。

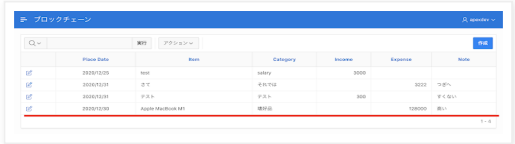
1つのエラーが発生しました

ORA-22816: RETURNING句ではサポートされていない機能です。

ページ・デザイナを再度開いて、フォームのデータを挿入するプロセスを選択し、プロパティの**挿入後に主キーを返す**を**OFF**にします。これはトリガーによる挿入がreturning into句をサポートしていないので、その対応です。



再度、作成をクリックするとデータが挿入されたことが確認できます。



データの編集や削除のボタンもページ作成ウィザードによって作成されていますが、これらは機能しません。

とりあえず、ブロックチェーン表を手軽に使ってみるための定義を紹介してみました。

完

Yuji N. 時刻: 17:14

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