

日々は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。

2023年4月7日 金曜日

PGQLを使ったAPEXアプリを23cのSQL Property Graphで置き換えてみる

PGQLをトランスレートして生成したSQLをレポートのソースとした、Oracle APEXのアプリケーションを作成したことがあります。

PGQLの2層モードで生成されるSQLをAPEXのレポートに組み込む

Oracle Database 23cよりSQL/PGQ (Property Graph Queries) がサポートされ、グラフ形式のデータを直接扱うことができるようになりました。

以前の記事で作成したAPEXアプリケーションを、SQL/PGQを使うように見直してみます。

使用するのは[こちらの記事](#)で作成した、Oracle Database 23c FreeにOracle APEX 22.2をインストールした環境です。Oracleが提供している[VirtualBoxのAPEXを日本語化した環境](#)でも同様に作業ができるでしょう。

ワークスペースとしてAPEXDEV (紐づいているスキーマもAPEXDEV) が作成されていることを前提とします。ワークスペースやスキーマ名が異なる場合は、APEXDEVの部分を置き換えてください。

最初にスキーマAPEXDEVにSQL Property Graphを扱う権限を与えます。

4.1.5 Granting System and Object Privileges for SQL Property Graphs

```
GRANT CREATE PROPERTY GRAPH, CREATE ANY PROPERTY GRAPH,  
      ALTER ANY PROPERTY GRAPH, DROP ANY PROPERTY GRAPH,  
      READ ANY PROPERTY GRAPH TO APEXDEV;
```

ユーザーsystemで接続し、スキーマAPEXDEVに権限を与えます。

```
[oracle@apex ~]$ sqlplus system/*****@localhost/freepdb1
```

```
SQL*Plus: Release 23.0.0.0.0 - Developer-Release on Fri Apr 7 15:34:54 2023  
Version 23.2.0.0.0
```

```
Copyright (c) 1982, 2023, Oracle. All rights reserved.
```

```
Last Successful login time: Fri Apr 07 2023 15:34:44 +09:00
```

```
Connected to:  
Oracle Database 23c Free, Release 23.0.0.0.0 - Developer-Release  
Version 23.2.0.0.0
```

```
SQL> GRANT CREATE PROPERTY GRAPH, CREATE ANY PROPERTY GRAPH,  
      ALTER ANY PROPERTY GRAPH, DROP ANY PROPERTY GRAPH,  
      READ ANY PROPERTY GRAPH TO APEXDEV; 2 3
```

```
Grant succeeded.
```

```
SQL> exit
Disconnected from Oracle Database 23c Free, Release 23.0.0.0.0 - Developer-Release
Version 23.2.0.0.0
[oracle@apex ~]$
```

元記事ではSQLclでPGQLを使用するために、色々な作業を行っています。SQL Property Graphを使う場合は、これらの作業は不要です。

以下のSQLを実行し、頂点であるアカウントとエッジであるトランザクションを保存する表を作成します。表SQLCL_BANK_ACCTS、SQLCL_BANK_TXNSは元記事とまったく同じ表です。

```
create table sqlcl_bank_accts (
  acct_id          number not null constraint sqlcl_bank_accts_acct_id_pk primary key
  name             varchar2(64) not null
)
;

create table sqlcl_bank_txns (
  txn_id           number not null constraint sqlcl_bank_txns_txn_id_pk primary key,
  from_acct_id     number not null
                  constraint sqlcl_bank_txns_from_acct_id_fk
                  references sqlcl_bank_accts on delete cascade,

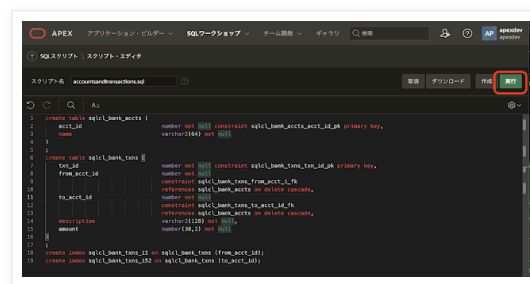
  to_acct_id       number not null
                  constraint sqlcl_bank_txns_to_acct_id_fk
                  references sqlcl_bank_accts on delete cascade,

  description      varchar2(128) not null,
  amount           number(38,2) not null
)
;

create index sqlcl_bank_txns_i1 on sqlcl_bank_txns (from_acct_id);
create index sqlcl_bank_txns_i52 on sqlcl_bank_txns (to_acct_id);
```

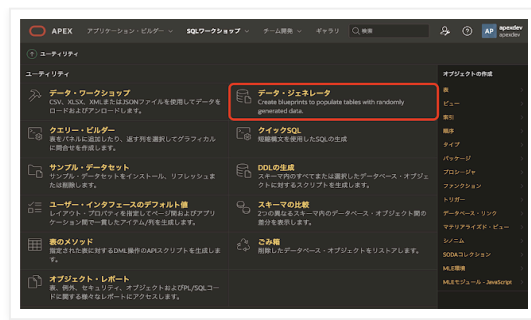
accounts-and-transactions.sql hosted with ❤ by GitHub

[view raw](#)



作成した表にテスト・データを投入するために、ユーティリティのデータ・ジェネレータを使用します。

データ・ジェネレータでの作業も元記事とまったく同じです。同じ手順を記載するのは冗長になるため、テスト・データの準備については元記事を参照してください。



データ・ジェネレータによるテスト・データの投入が完了したところから継続します。

SQLコマンドより、表SQLCL_BANK_ACCTSに投入されたデータを確認します。

```
select count(*) from sqlcl_bank_accts;
```



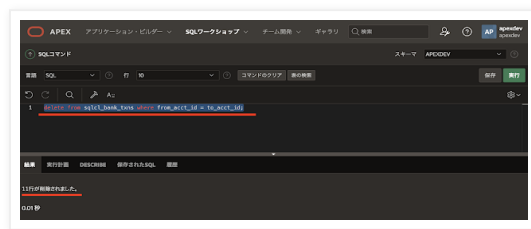
表SQLCL_BANK_TXNSに投入されたデータを確認します。

```
select count(*) from sqlcl_bank_txns;
```



テスト・データより、送金元と送金先が同じ行を削除します。

```
delete from sqlcl_bank_txns where from_acct_id = to_acct_id;
```



以上でテスト・データの準備が完了しました。

この表を元にプロパティ・グラフSQLCL_BANK_GRAPHを作成します。

```
create property graph sqlcl_bank_graph
vertex tables (
  sqlcl_bank_accts as accounts
  key (acct_id)
```

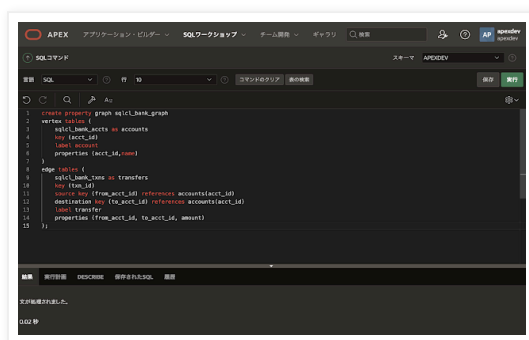
```

label account
properties (acct_id,name)
)
edge tables (
  sqlcl_bank_txns as transfers
  key (txn_id)
  source key (from_acct_id) references accounts(acct_id)
  destination key (to_acct_id) references accounts(acct_id)
  label transfer
  properties (from_acct_id, to_acct_id, amount, description)
);

```

sqlcl_bank_graph.sql hosted with ❤ by GitHub

[view raw](#)



元記事で作成したAPEXアプリケーションをインポートします。
<https://github.com/ujnak/apexapps/blob/master/exports/pgql-reports.zip>

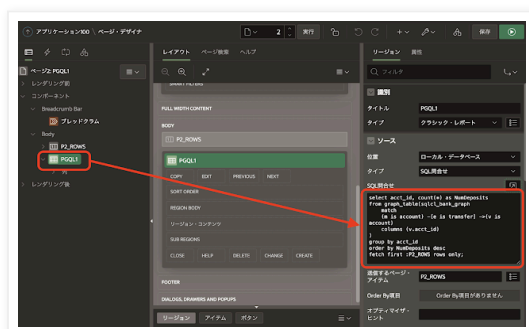
インポートしたアプリケーションを編集します。

ページ番号2のPGQL1のレポートのソースを以下に変更します。

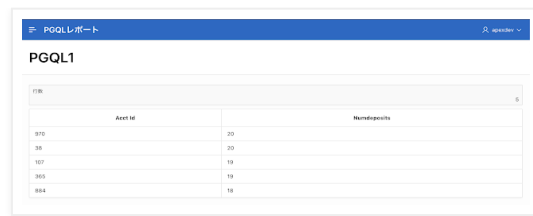
```

select acct_id, count(*) as NumDeposits
from graph_table(sqlcl_bank_graph
  match
    (m is account) -[e is transfer] -> (v is account)
  columns (v.acct_id)
)
group by acct_id
order by NumDeposits desc
fetch first :P2_ROWS rows only;

```



ページを実行すると、以下のように表示されます。



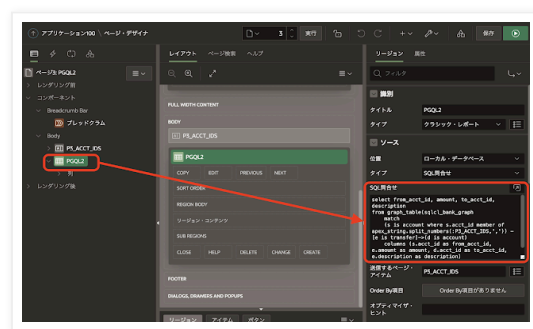
| Acct Id | Amounts |
|---------|---------|
| 570 | 20 |
| 38 | 20 |
| 107 | 19 |
| 303 | 19 |
| 554 | 19 |

PGQLより生成したSQLをSQL/PGQを使うように置き換えます。マニュアルの主な参照箇所は以下になります。

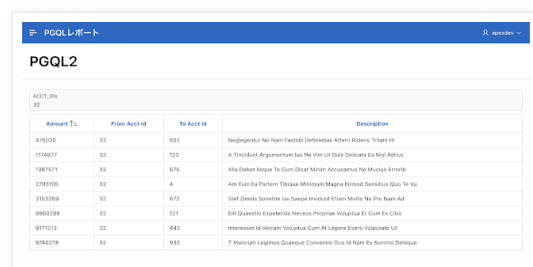
5 SQL GRAPH_TABLE Queries

ページ番号3のPGQL2のレポートのソースは、以下に変更します。

```
select from_acct_id, amount, to_acct_id, description
from graph_table(sqlcl_bank_graph
match
(s is account where s.acct_id member of apex_string.split_numbers(:P3_ACCT_IDS,',')) -[e is transfer]->(d is account)
columns (s.acct_id as from_acct_id, e.amount as amount, d.acct_id as to_acct_id, e.description as description)
);
```



ページを実行すると、以下のように表示されます。



| Acct Id | Amount | From Acct Id | To Acct Id | Description |
|---------|--------|--------------|------------|--|
| 476020 | 32 | 601 | 601 | Neglegentur Non Nam Facillit Orefabae Alteri Idemsi Trisat H6 |
| 170607 | 32 | 625 | 625 | A Teneant Argentum Non Nam Ut Quis Delicatus Ea Huius Adhuc |
| 186707 | 32 | 676 | 676 | Allo Dolor Deser Te Cum Dicit Minus Accusatio Non Natus Errore |
| 273705 | 32 | 4 | 4 | Alio Qui Ea Pectus Tiliqua Minus Magna Elitend Desiderio Qui Te Yu |
| 375309 | 32 | 673 | 673 | Uat Quis Quisquam Non Quis Involunt Etiam Modis Ne Quo Nam Ad |
| 690208 | 32 | 121 | 121 | Eit Quisquam Expendio Necesse Propter Voluptate Et Cum Ea Cibus |
| 877053 | 32 | 442 | 442 | Interesset Id Veniam Voluptate Cum At Legere Everti Voluptate Ut |
| 976276 | 32 | 932 | 932 | T Minus Legimus Quisquam Conventio Duis Id Nam Ea Summo Desiderio |

PGQLからSQLを生成してレポートに組み込むより、はるかに簡単にグラフ構造のデータを活用できるようになっています。

現時点（バージョン22.2）のOracle APEXには、グラフを描画する標準のコンポーネントはありません。地理空間情報に対応しているマップ・リージョンのように、グラフ構造のデータについても専用のリージョンが提供されると、もっとグラフの使い勝手が良くなることが期待されます。

今回の変更を加えたAPEX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/sqlpgq-report.zip>

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

[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](#)

Powered by Blogger.