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8月27日金曜日

カスタム認証スキームの保護について

以前の記事でカスタム認証を動作させるために、表のSELECT権限をPUBLICに与えました。そのときの記事の本題ではないとはいえ、あまり良くない方法ではあります。そこで、カスタムの認証スキームの保護について、ユーザー情報を直接参照することをを禁止した上で、認証スキームのファンクションを実行できる実装をいくつか行ってみます。

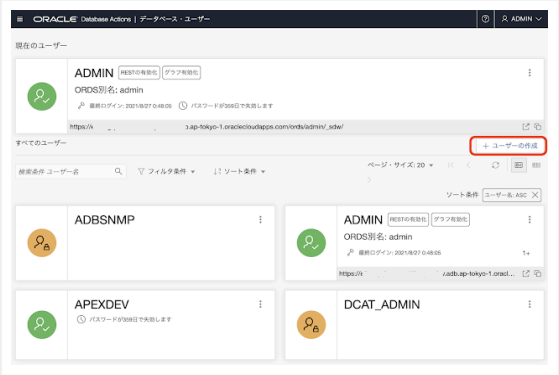
確認に使用する環境はAlways FreeのAutonomous Transaction Processingになります。特別な機能は使用しないので、その他の環境でも同様に実装可能でしょう。APEXのワークスペースとして**APEXDEV**が作成済みで、それを使用して確認を行います。

認証スキームのためのスキーマの作成

最初に認証スキームに使うデータを保持するスキーマを作成します。認証情報を保持するスキーマを、APEXのワークスペースのスキーマや、ユーザー・データを保持するスキーマと分離することにより、認証に使用する情報をそれらのスキーマから直接参照できないようにします。

ユーザー**ADMIN**にて**データベース・アクション**に接続し、**データベース・ユーザー**を開きます。

ユーザーの作成を呼び出します。



ユーザー名をCUSTAUTHとします。**パスワード**を指定し、**Webアクセス**を**ON**にします。**表領域の割り当て制限 DATA**として**25M**を割り当てます。今回の実装例では認証データは微々たるものなので25Mバイトを割り与えていますが、もっと多く(最大は**UNLIMITED**)割り与えてもよいでしょう。**ユーザーの作成**を実行します。

ユーザーCUSTAUTHが作成されました。認証スキームが使用する表やファンクションなど、認証に関するすべてのデータベース・オブジェクトは、このCUSTAUTHのスキーマに作成します。

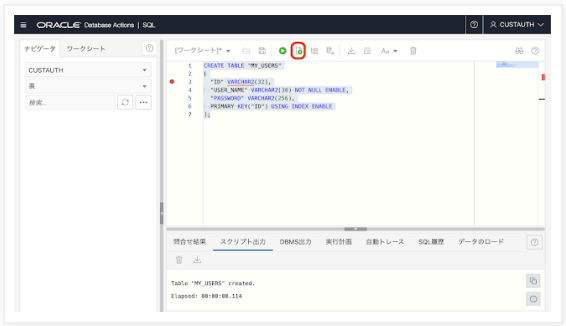
認証スキームに使用するオブジェクトの作成

こちらの資料(Oracle APEX勉強会 - 認証と認可の実装を学ぶ)で紹介した認証スキームを実装します。

ユーザー**CUSTAUTH**で**データベース・アクション**に接続し、**開発**の**SQL**を開き表**MY_USERS**を作成します。以下の**CREATE TABLE文を実行**します。

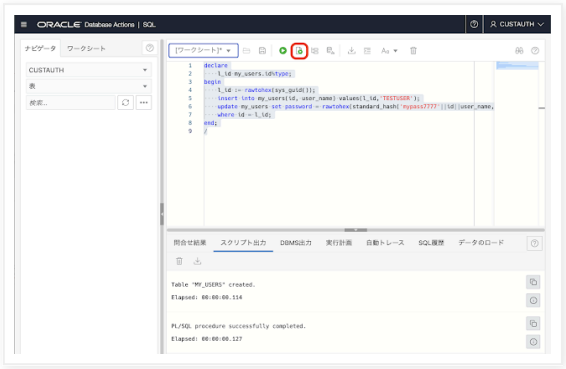
```
CREATE TABLE "MY_USERS"
(
    "ID" VARCHAR2(32),
```

```
    "USER_NAME" VARCHAR2(30) NOT NULL ENABLE,
   "PASSWORD" VARCHAR2(256),
 PRIMARY KEY("ID") USING INDEX ENABLE
);
```



ユーザーの登録を行います。ユーザー名は**TESTUSER**、パスワードは**mypass7777**です。以下のPL/SQLコードを実行します。

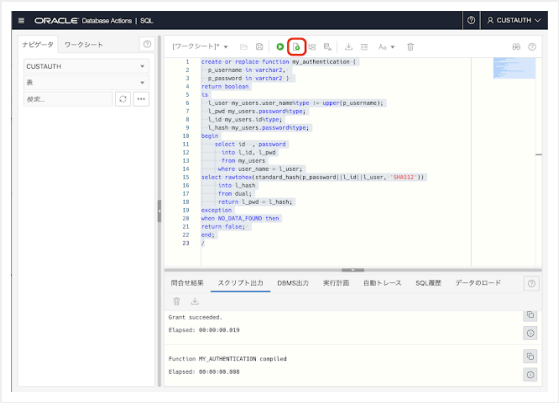
```
declare
   l_id my_users.id%type;
begin
   l_id := rawtohex(sys_guid());
   insert into my_users(id, user_name) values(l_id,'TESTUSER');
   update my_users set password = rawtohex(standard_hash('mypass7777'||id||user_name, 'SHA512'))
   where id = l_id;
end;
/
```



認証スキームとなるファンクション**my_authentication**を作成します。

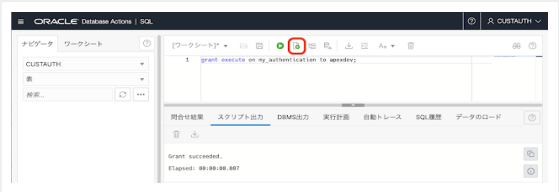
```
create or replace function my_authentication (
 p_username in varchar2,
 p_password in varchar2 )
return boolean
is
 l_user my_users.user_name%type := upper(p_username);
 l_pwd my_users.password%type;
 l_id my_users.id%type;
 l_hash my_users.password%type;
begin
   select id, password
     into l_id, l_pwd
     from my_users
    where user_name = l_user;
select rawtohex(standard_hash(p_password||l_id||l_user, 'SHA512'))
    into l_hash
    from dual;
    return l_pwd = l_hash;
```

```
exception
when NO_DATA_FOUND then
    return false;
end;
/
```



作成したファンクションをユーザーAPEXDEVから呼び出せるように、実行権限を与えます。

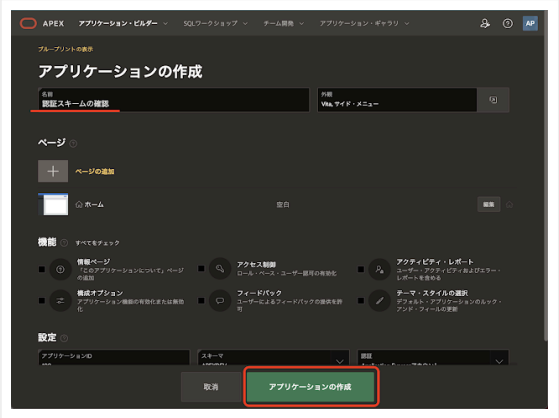
```
grant execute on my_authentication to apexdev;
```



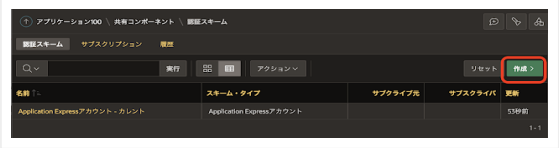
以上で認証スキームとして使用するファンクションや、それが利用する表が作成されました。サインインに使用するユーザーTESTUSERも登録済みです。

テスト用アプリケーションの作成

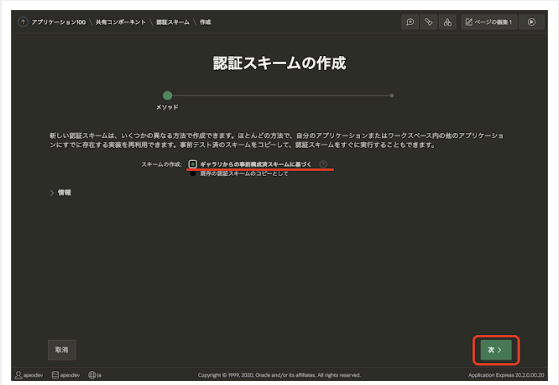
テストに使用するアプリケーションを作成します。**アプリケーション作成ウィザード**を実行し、アプリケーションの**名前**を**認証スキームの確認**として、**アプリケーションの作成**を実行します。何の機能も含まないアプリケーションが作成されます。



アプリケーションが作成されたら、**共有コンポーネント**の**認証スキーム**を開き、**作成**を実行します。



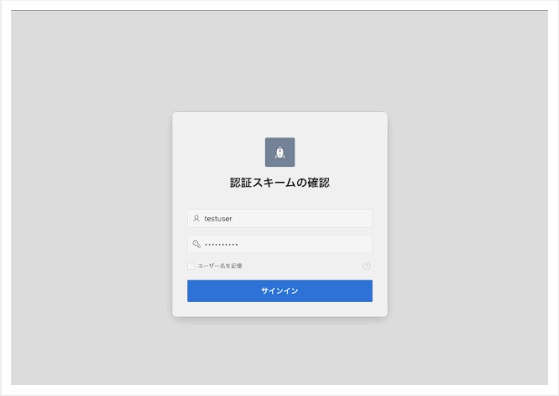
スキームの作成として、**ギャラリからの事前構成済スキームに基づく**を選択します。**次**に進みます。



名前は**CUSTAUTH**とし、**タイプ**として**カスタム**を選択します。**認証ファンクション名**として**custauth.my_authentication**を指定します。**認証スキームの作成**を実行します。

作成された認証スキームはすぐに**カレント**の認証スキームになります。**ページを実行**して、サインインの確認を行います。

ユーザー名**testuser**、パスワード**mypass7777**でサインインに成功します。それ以外ではサインインはできません。



以上でテスト用のアプリケーションも作成できました。

保護の確認

PL/SQLによるプロシージャ、ファンクションおよびパッケージは特別な指定がない限り、デフォルトでは**定義者権限**にて実行されます。定義者権限について、マニュアルでは以下のように説明されています。

> 定義者権限プロシージャのユーザーに必要なのは、そのプロシージャを実行する権限のみで、そのプロシージャでアクセスする基礎となるオブジェクトに対する権限は不要です。これは、定義者権限プロシージャは、その実行者に関係なく、プロシージャを所有するユーザーのセキュリティ・ドメインの下で動作するためです。

今回の例に当てはめて説明します。

作成したファンクション**my_authentication**はスキーマCUSTAUTHに作成されているため、**CUSTAUTHの権限で実行**されます。ファンクションmy_authenticationの実行権限があれば、ファンクション内でアクセスしている表などのオブジェクトのアクセス権限は不要です。それらは**CUSTAUTHが持つの権限でアクセス**されるためです。

ユーザーAPEXDEVは表CUSTAUTH.MY_USERSへのアクセス権限は持ちませんが、ファンクションmy_authenticationの実行権限は与えられています。そのため、受け取ったユーザー名、パスワードが正しいかどうかファンクションmy_authenticationを呼び出して検証することはできます。しかし、登録済みのユーザー名の一覧やハッシュ化されたパスワードを直接参照することはできません。

オラクルでは定義者権限がデフォルトの設定になります。

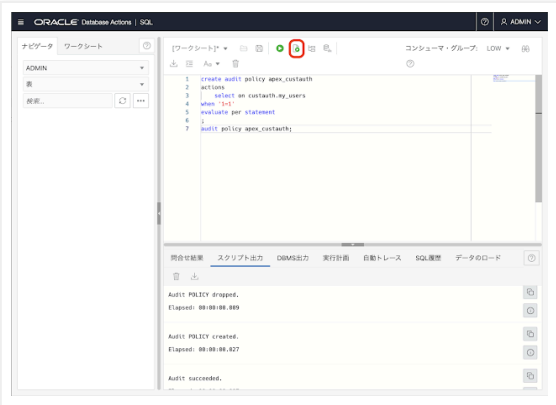
実際に**SQLワークショップ**の**SQLコマンド**より以下のSQLを実行すると、ORA-00942: 表またはビューが存在しません。が発生します。

```
select * from custauth.my_users;
```

テスト用アプリケーションではファンクションmy_authenticationが呼び出され、ユーザーTESTUSERにて認証が成功できていることは確認済みです。実際にどのようなアクセスが発生しているのか、表CUSTAUTH.MY_USERSの監査証跡を取得して確認してみます。

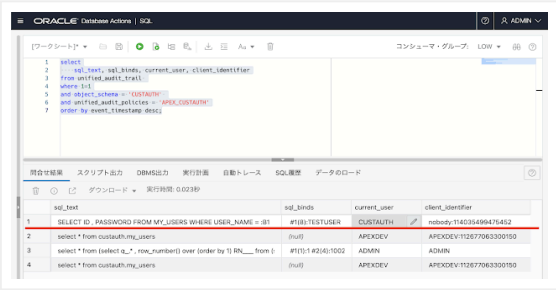
ユーザーADMINにてデータベース・アクションに接続し、以下のSQLを実行します。

```
create audit policy apex_custauth
actions
   select on custauth.my_users
when '1=1'
evaluate per statement
;
audit policy apex_custauth;
```



テスト用のアプリケーションを実行し（サインイン済みの場合は一旦サインアウトし）、サインインを行います。サインインを行なったのち、ビューUNIFIED_AUDIT_TRAILを確認します。

```
select
   sql_text, sql_binds, current_user, client_identifier
from unified_audit_trail
where 1=1
and object_schema = 'CUSTAUTH'
and unified_audit_policies = 'APEX_CUSTAUTH'
order by event_timestamp desc;
```



current_userが**CUSTAUTH**となっていることより、認証スキームのファンクションmy_authenticationが、その認証スキームを実行しているユーザーAPEXDEVではなく、ファンクションmy_authenticationが定義されているユーザーCUSTAUTHにて実行されていることが確認できます。

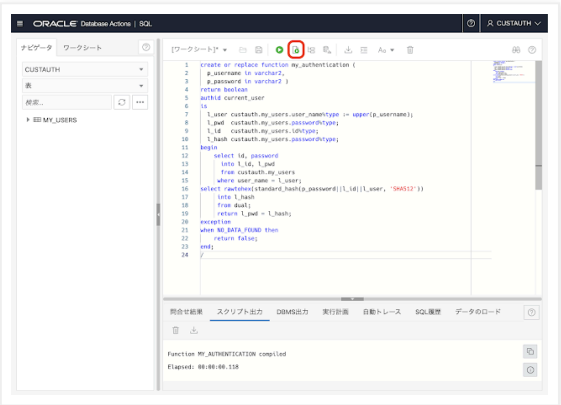
大抵の場合、PL/SQLのプロシージャ、ファンクションおよびパッケージは定義者権限による実行が適切です。

実行者権限でのファンクションの実行

PL/SQLのファンクションは呼び出したユーザーの権限で実行させることも可能です。その場合はコードに**authid current_user**を含めます。ファンクションmy_authenticationを**実行者権限**で動作するように改変します。

ユーザーCUSTAUTHでデータベース・アクションに接続し、SQLより以下を実行します。authid current_userの記述を追加しています。

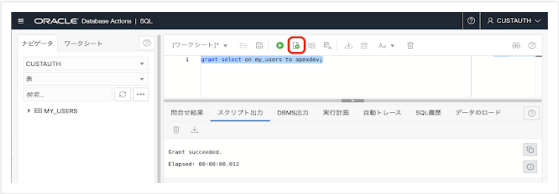
```
create or replace function my_authentication (
  p_username in varchar2,
  p_password in varchar2 )
return boolean
authid current_user
is
  l_user custauth.my_users.user_name%type := upper(p_username);
  l_pwd  custauth.my_users.password%type;
  l_id   custauth.my_users.id%type;
  l_hash custauth.my_users.password%type;
begin
    select id, password
      into l_id, l_pwd
      from custauth.my_users
     where user_name = l_user;
select rawtohex(standard_hash(p_password||l_id||l_user, 'SHA512'))
     into l_hash
     from dual;
     return l_pwd = l_hash;
exception
when NO_DATA_FOUND then
   return false;
end;
/
```



authid current_userの追加以外に、**表MY_USERSにスキーマCUSTAUTHの指定も含めて**います。実行者権限での実行で単に表MY_USERSが指定されている場合は、実行者のスキーマに含まれている表を参照します。(この場合、表APEXDEV.MY_USERSを参照しようとします)

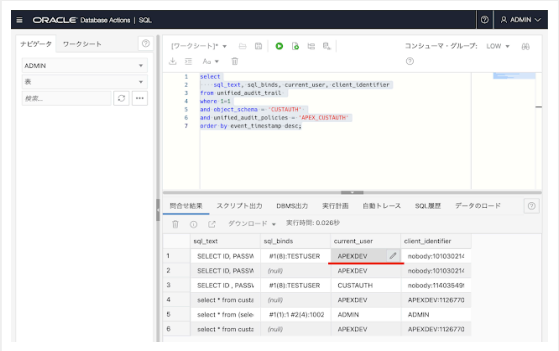
ファンクションmy_authenticationはユーザーAPEXDEVとして実行されるため、表MY_USERSへのSELECT権限が必要です。以下のGRANT文を実行します。

```
grant select on my_users to apexdev;
```



テスト用アプリケーションを実行し、ユーザーTESTUSERでサインインします。正常にサインインされるはずです。

ユーザーADMINにてデータベース・アクションに接続し、ビューUNIFIED_AUDIT_TRAILより、認証を行うために実行されたSQLを確認します。**current_user**が**APEXDEV**であることが確認できます。

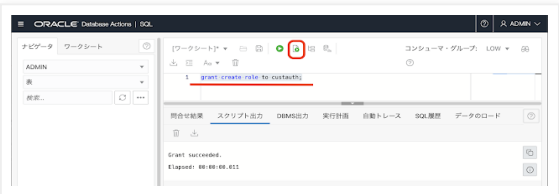


コード・ベース・アクセス制御の設定

表MY_USERSへのSELECT権限をユーザーAPEXDEVに与えると、登録されているすべてのユーザーの情報を、APEXDEVから参照することができるようになります。これは望ましくないため、ファンクションmy_authenticationを通したときだけ、表MY_USERSの参照を可能にします。

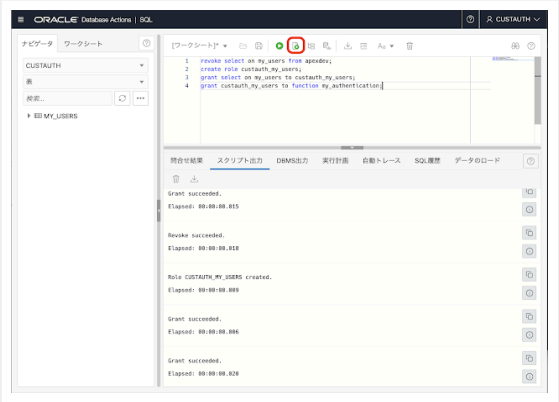
ユーザーCUSTAUTHにロールを作成する権限を与えます。

```
grant create role to custauth ;
```

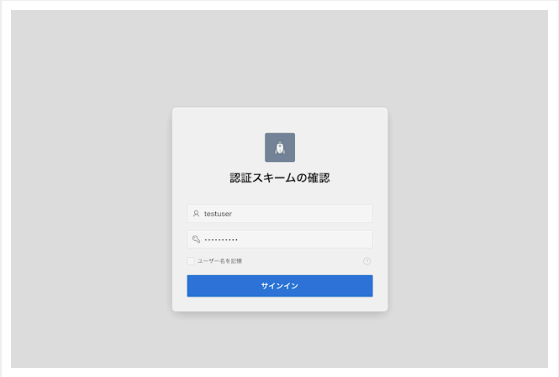


ユーザーCUSTAUTHでデータベース・アクションに接続します。SQLから以下を実行します。ユーザーAPEXDEVより表MY_USERSのSELECT権限を削除します。表MY_USERSのSELECT権限を含んだロール**CUSTAUTH_MY_USERS**を作成し、そのロールをファンクションMY_AUTHENTICATIONへ割り与えています。

```
revoke select on my_users from apexdev;
create role custauth_my_users;
grant select on my_users to custauth_my_users;
grant custauth_my_users to function my_authentication;
```



以上の設定により、テスト用アプリケーションのサインインは問題なく行えるようになります。ユーザー**testuser**、パスワード**mypass7777**にてサインインします。



アプリケーションの画面が開きます。

この状態でSQLコマンドより表CUSTAUTH.MY_USERSを検索すると、ORA-00942: 表またはビューが存在しません。が発生します。これは期待している動作です。

以上で、認証に使用する情報を保護した上で、認証スキームを実装する方法の紹介は終了です。

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: 13:58

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