

日々是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3月15日月曜日

RESTデータ・ソースを使った高度な同期化(1) - REST API定義

オラクルのCarsten CzarSKIさんがブログ記事として、RESTデータ・ソースの同期化の使い方について解説しています。

Synchronize Parent-Child REST Sources

<https://blogs.oracle.com/apex/synchronize-parent-child-rest-sources>

内容は以下になります。

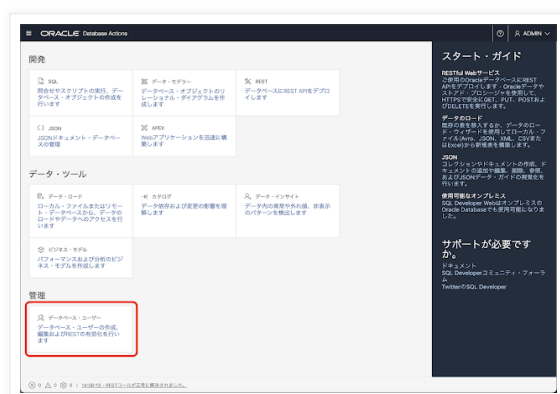
1. サンプル・データセットのEMP/DEPTを使って、REST APIを作る。
2. 表DEPTを最初に同期し、次に表EMPのデータ(従業員)を部門ごとに同期する。これを宣言的に実装する。
3. 上記をPL/SQLコードで実装する。
4. 自動化を使って同期処理を行う。

RESTデータ・ソースを使ったデータの同期については、以前に記事を書いています。この記事では単に同期化を設定しただけですが、こちらは複数のアクションを定義するなど、より実践的な内容になっています。なので、ちょっと自分でも試してみて、その過程を記載します。

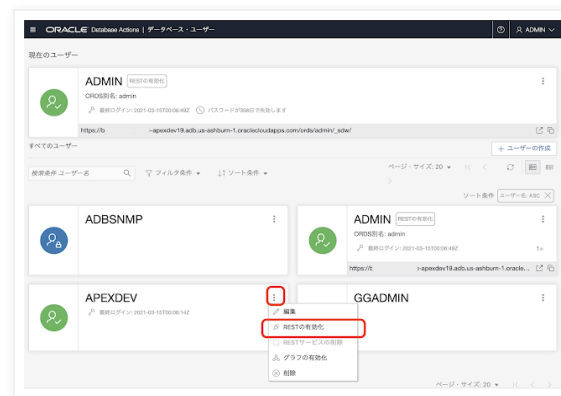
記事は4つに分けて記載します。最初は準備、ということでREST API自体を作成します。元のブログでは、PL/SQLのコードを実行することによりREST APIを定義していますが、本記事ではSQL Developer WebのGUIを使って登録します。

確認作業には、Always FreeのAutonomous Transaction Processingのインスタンスを使用しています。APEXのワークスペースとしてAPEXDEVが作成済みで、かつ、そのワークスペースにサンプル・データセットのEMP/DEPTがインストールされている状態から始めます。最初はAshburnリージョンのOracle Database 21cで試したのですが、SQL Developer Webが不安定でした。ですので、本機能の確認にはOracle Database 19cを使用してください。

SQL Developer WebにADMINでサインインします。管理のデータベース・ユーザーを開きます。



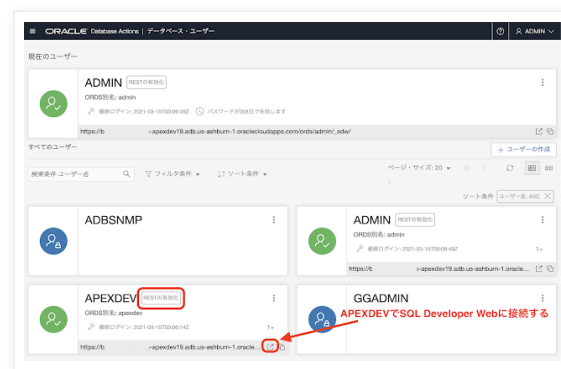
ユーザー**APEXDEV**にて、SQL Developer WebからRESTサービスを定義できるようにするため、**RESTの有効化**を実行します。



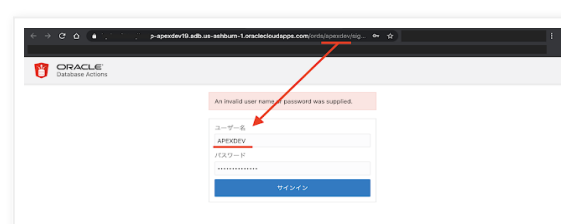
承認が必要をONにし、**REST対応ユーザー**をクリックします。



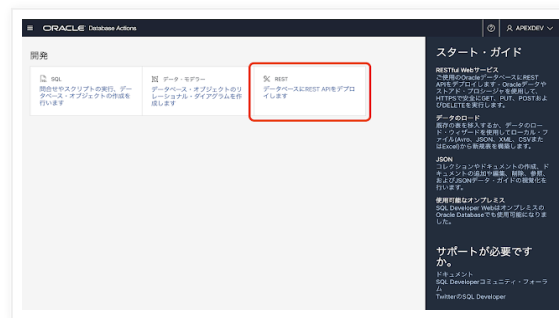
データベース・ユーザー名（ここでは**APEXDEV**）のすぐ隣に**RESTの有効化**と表示されます。**リンクの部分**をクリックして、APEXDEVにてSQL Developer Webに接続します。



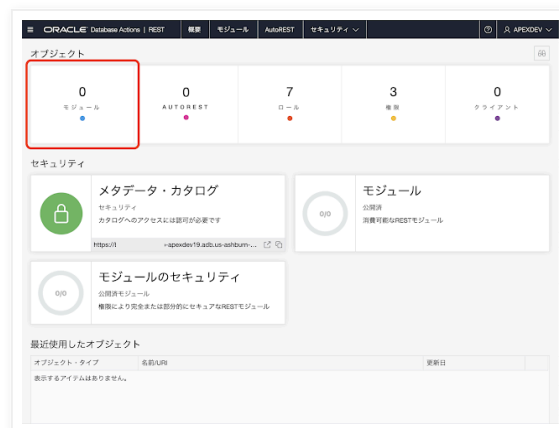
それぞれのユーザーでSQL Developer WebにサインインするURLには、ユーザー名が含まれます。**URLに含まれるユーザー名と同じ名前をユーザー名に指定します。**"An Invalid user name or password was supplied."と表示されることがありますが、すでにADMINでサインインしていた場合であれば無視できます。



開発の**REST**を開いて、RESTサービスを実装します。



モジュールを開きます。



最初にモジュールの作成を実行します。



元記事にてPL/SQLコードで定義しているものと、同じモジュール、テンプレートおよびハンドラを登録していきます。

モジュールは、**モジュール名**をsyncdemo.parent、**ベース・パス**は/syncdemo/を設定し、**作成**を実行します。

モジュールの作成

モジュール定義	使用可能な起点
モジュール名 * <input type="text" value="syncdemo.parent"/>	
ベース・パス * <input type="text" value="/syncdemo/"/>	
プレビューURL https://t...ip-apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apex:	
公開 <input checked="" type="checkbox"/>	ページ区切りサイズ <input type="text" value="25"/>
コメント <div></div>	
<input checked="" type="checkbox"/> 作成後モジュールに移動	
<div> <input type="button" value="作成"/> <input type="button" value="取消"/> </div>	

モジュールが作成されたので、続いて**テンプレートの作成**を実行します。

The screenshot shows the Oracle APEX interface for the 'syncdemo.parent' module. The 'テンプレート' (Template) section is visible, and a red box highlights the '+ テンプレートの作成' (Create Template) button.

表DEPTを操作の対象とする**URIテンプレート**として、**dept/**を作成します。

テンプレートの作成

モジュール名
syncdemo.parent

ベース・パス
/syncdemo/

URIテンプレート *
dept/

プレビューURL
https://b1-apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apex:

優先度 HTTPエンティティ・タグ・タイプ
0 セキュア・ハッシュ

コメント

☒ 作成後ハンドラに移動

作成 取消

テンプレートが作成されたら、実際の処理を記述するハンドラをテンプレートに定義します。**ハンドラの作成**を実行します。

ORACLE Database Actions | REST | 概要 | モジュール | AutoREST | セキュリティ

REST / モジュール / syncdemo.parent / dept/

dept/

プレビューURL
https://b1-apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apexdev/syncdemo/dept/

ハンドラ [+ ハンドラの作成](#)

メソッドはGET、**ソース・タイプ**は収集問合せ、**ソース**としては以下を指定します。指定を行った後、**作成**をクリックします。

```
select * from dept
```

ハンドラの作成

ハンドラ定義	使用可能なMIME
<p>モジュール名</p> <p>syncdemo.parent</p> <p>完全なURL</p> <p>https://bj 1>apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apex:</p> <p>メソッド *</p> <p>GET</p> <p>ページ当たりのアイテム数</p> <p>25</p> <p>ソース・タイプ</p> <p>収集問合せ</p> <p>ソース *</p> <pre>1 select * from dept</pre> <p>コメント</p> <p>作成 取消</p>	

ハンドラが設定されます。これでREST APIが機能するようになりました。REST APIを呼び出して動作を確認します。

ORACLE Database Actions | REST | 検索 | モジュール | AutoREST | セキュリティ

REST / モジュール / syncdemo.parent / dept / GET

GET

最終更新: 29秒前

説明も取るコメントはありません

(ソース・タイプ: jsoncollection) (ページ・サイズ: 25)

https://19 apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apexdev/syncdemo/dept/

ソース

```
1 select * from dept
```

パラメータ

検索 パラメータ

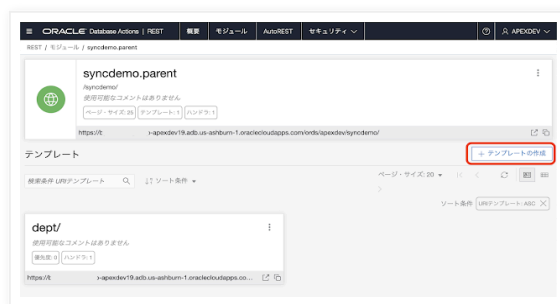
ページ・サイズ: 25

表DEPTの内容がJSONで返されることが確認できます。

```
{
  "items": [
    {
      "deptno": 10,
      "dname": "ACCOUNTING",
      "loc": "NEW YORK"
    },
    {
      "deptno": 20,
      "dname": "RESEARCH",
      "loc": "DALLAS"
    },
    {
      "deptno": 30,
      "dname": "SALES",
      "loc": "CHICAGO"
    },
    {
      "deptno": 40,
      "dname": "OPERATIONS",
      "loc": "PORTLAND"
    }
  ],
  "totalCount": 4,
  "offset": 0,
  "pageSize": 25,
  "orderBy": "deptno",
  "direction": "asc",
  "links": {
    "self": "https://19 apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apexdev/syncdemo/dept/",
    "next": "https://19 apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apexdev/syncdemo/dept/?offset=4",
    "previous": "https://19 apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apexdev/syncdemo/dept/?offset=-4"
  }
}
```

次に部門番号DEPTNOを指定して、表EMPより、それに紐づく従業員のみを返すREST APIを作成します。

モジュールのページを開き、**テンプレートの作成**を実行します。



URIテンプレートとして**emp/:deptno**を設定します。コロンで始まる文字列はパラメータとして扱われます。つまり、このREST APIは、実際には**emp/10**、**emp/20**といったURIから呼び出されます。

テンプレートの作成

モジュール名

syncdemo.parent

ベース・パス

/syncdemo/

URIテンプレート *

emp/:deptno

プレビューURL

https://t p-apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apex:

優先度

0

HTTPエンティティ・タグ・タイプ

セキュア・ハッシュ

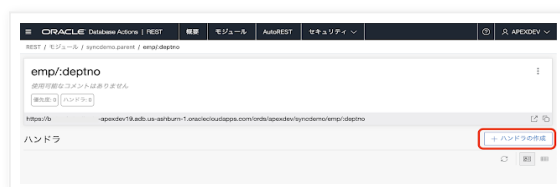
コメント

☒ 作成後ハンドラに移動

作成

取消

ハンドラの作成を行います。



メソッドは**GET**、**ソース・タイプ**は**収集問合せ**、**ソース**として以下を指定します。指定を行った後、**作成**をクリックします。URIに含まれる部門番号を持つ従業員を、表EMPから返します。

```
select * from emp where deptno = :deptno
```

ハンドラの作成

ハンドラ定義	使用可能なMIME
<p>モジュール名</p> <p>syncdemo.parent</p> <p>完全なURL</p> <p>https://b. . p-apexdev19.adb.us-ashburn-1.oraclecloudapps.com/ords/apexdemo</p> <p>メソッド *</p> <p>GET</p> <p>ページ当たりのアイテム数</p> <p>25</p> <p>ソース・タイプ</p> <p>収集問合せ</p> <p>ソース *</p> <pre>1 select * from emp where deptno = :deptno</pre> <p>コメント</p> <p>作成 取消</p>	

ハンドラが作成されたので、動作を確認します。

ORACLE Database Actions | REST | 検索 | モジュール | AutoREST | セキュリティ | 4PEXDEV

emp/deptno

GET

最終更新: 1秒前

使用可能なコメントはありません

ソース・タイプ: jsoncollection ページ・サイズ: 25

ソース

```
1 select * from emp where deptno = :deptno
```

パラメータ

検索 パラメータ

そのまま実行すると、:deptnoが引数になるため、ORA-1722が発生します。

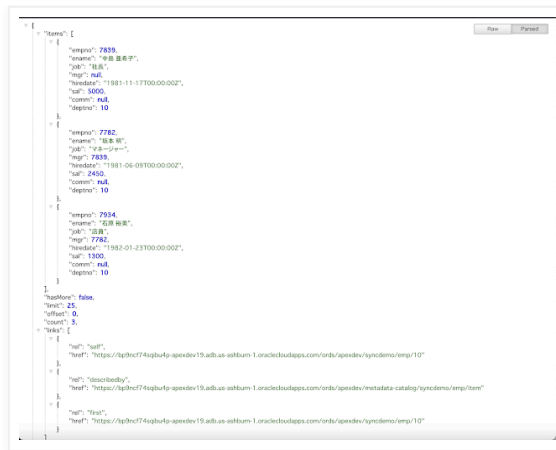
ORACLE REST Data Services

555 User Defined Resource Error

2021-03-10T10:40:00.000Z /ordssyncdemo/emp/deptno/10 / 0000 25001

このリソースに関連付けられているSQL文を評価しようとしているときにエラーが発生したため、リクエストを処理できませんでした。SQL文が正しく構文設定され、エラーなしで実行されることを確認してください。SQLエラー・コード: 1722, エラー・メッセージ: ORA-01722: invalid number

URLの:deptnoの部分を10に変更して、再度アクセスします。部門番号10の従業員の一覧がJSONで返されます。



以上で元記事で行っている準備作業は完了です。

折角なので、もう少しREST APIの定義を行ってみます。

元記事は部門番号によって従業員を選択するREST APIのテンプレートとして**emp/:deptno**、例えば**emp/10**といった形式を選んでいますが、今回はあくまでRESTデータ・ソースを使用した同期化の実装例を紹介するためのものですので、単純化のためにこうしているのだと思います。

一般的には、**emp/数値** というURIテンプレートでの数値は従業員番号とし、表EMPに含まれる特定の従業員を指定します。そして、部門番号で絞り込みをするためには**emp?deptno=部門番号**というURIにします。

作ってみましょう。最初にテンプレートを作成します。**URLテンプレート**は**emp**です。**deptno**は**パラメータ**として指定するので、**URIテンプレートには含めません**。

テンプレートが作成されたら、続いてハンドラを作成します。

メソッドはGET、ページ当たりのアイテム数を10、ソース・タイプは収集問合せとします。ソースは以下です。

```
select * from emp
where
(
  :deptno is null
or
  :deptno = deptno
)
```

パラメータのdeptnoは指定されていないこともあるため、その場合は表EMPの行がすべて返されるようにしています。

ハンドラの作成

ハンドラ定義 使用可能なMIME

モジュール名
syncdemo.parent

完全なURL
https://bq...>apexdev19.adb.us-ashburn-1.oraclecloudapps.com/fords/apex

メソッド * ページ当たりのアイテム数
GET 10

ソース・タイプ
収集問合せ

ソース *
1 select * from emp
2 where
3 (
4 :deptno is null
5 or
6 :deptno = deptno
7)

コメント

作成 取消

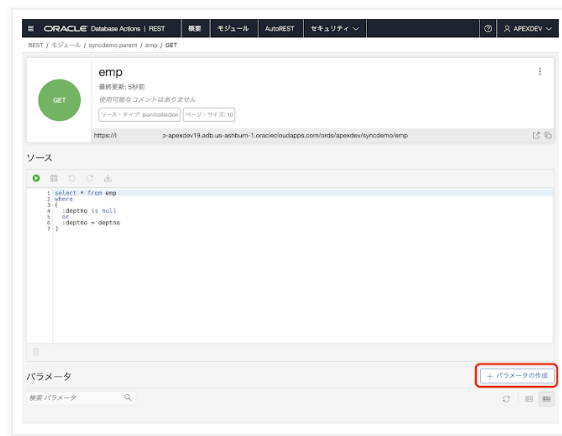
ハンドラのメソッドとして、GET以外にPOST、PUT、DELETEを選択することができます。

GET以外はソース・タイプはPL/SQLのみで、必ずPL/SQLコードで処理を記述します。GETについては、SELECT文の結果として複数行が返される場合は収集問合せ、単一行が返される場合は、コレクション・アイテムをソース・タイプとして指定します。ソース・タイプのメディアは、コンテンツ・タイプとそのコンテンツ・タイプでBLOB/CLOBを出力します。通常はデータベースに保存されている画像やファイルを出力するために使用します。PL/SQLは出力する内容を、すべてPL/SQLコードにて記述します。

コレクション・アイテムの場合、(1行の)対象行がJSONオブジェクトとして返されます。収集問合せでは、返される行それぞれをJSONオブジェクトとしたJSON配列が返されます。

ページ当たりのアイテム数の指定があると、収集問い合わせの結果の行数(オブジェクト数)は、そこで指定された数値を上限とします。そして、ページネーション(ページ送り)を行うためのURLが、追加情報としてREST APIの呼び出し元に返されます。

ハンドラが作成されたら、パラメータの作成をおこないます。



パラメータ名(URLに現れる名前)はdeptno、バインド変数名(SQLに現れる名前)をdeptno、ソース・タイプはURIとします。パラメータ・タイプはINT、アクセス・メソッドは入力です。

パラメータの作成

パラメータ名 *

バインド変数名 *

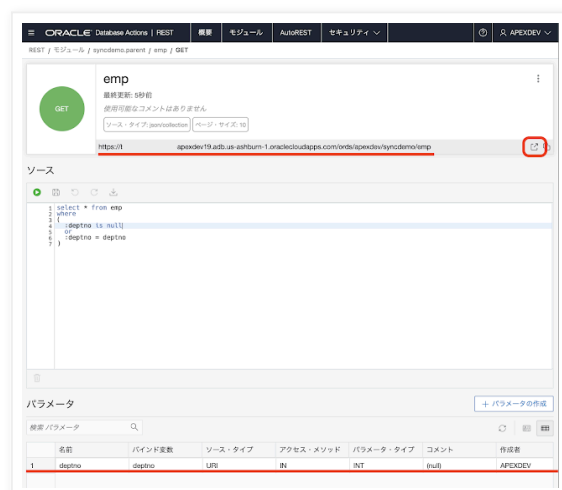
ソース・タイプ

パラメータ・タイプ

アクセス・メソッド

コメント

パラメータが登録されたので、動作を確認します。



出力には10行(JSON配列の要素が10)のみ含まれ、最下部には次の10行を取得するためのURLが"rel":"next"として返されています。"rel":"next"が返されなくなるまでアクセスすることにより、表EMPの全行を取得します。

```
{
  "empno": 7369,
  "ename": "WARD",
  "job": "MANAGER",
  "mgr": 7566,
  "hiredate": "1981-09-28T00:00:00Z",
  "sal": 1250,
  "comm": 1400,
  "deptno": 30
},
{
  "next": "https://apexcloudapps.com/ords/apexdev/syncdemo/emp"
},
{
  "rel": "self",
  "href": "https://apexcloudapps.com/ords/apexdev/syncdemo/emp"
},
{
  "rel": "description",
  "href": "https://apexcloudapps.com/ords/apexdev/syncdemo/emp"
},
{
  "rel": "next",
  "href": "https://apexcloudapps.com/ords/apexdev/syncdemo/emp?offset=10"
}
```

Oracle APEXのRESTデータ・ソースのタイプが、Oracle REST Data Sourceとなっている場合は、ページネーションの対応はAPEXが行ってくれるので利用者が意識することはありません。

emp?deptno=10としてアクセスした結果は以下となり、deptno=10の条件に一致した行のみが返されていることが確認できます。

```
{
  "empno": 7369,
  "ename": "WARD",
  "job": "MANAGER",
  "mgr": 7566,
  "hiredate": "1981-09-28T00:00:00Z",
  "sal": 1250,
  "comm": 1400,
  "deptno": 30
},
{
  "next": "https://apexcloudapps.com/ords/apexdev/syncdemo/emp?deptno=10"
},
{
  "rel": "self",
  "href": "https://apexcloudapps.com/ords/apexdev/syncdemo/emp?deptno=10"
},
{
  "rel": "description",
  "href": "https://apexcloudapps.com/ords/apexdev/syncdemo/emp?deptno=10"
},
{
  "rel": "next",
  "href": "https://apexcloudapps.com/ords/apexdev/syncdemo/emp?deptno=10"
}
```

最後に注意ですが、ページネーションのアクセスは、それぞれ別のトランザクションになります。つまり、最初にempをアクセスし、次にemp?offset=10をアクセスする間にデータが挿入/更新/削除されると、その更新された情報がemp?offset=10で選択されます。

Oracle APEXにてレポートのソースとしてRESTデータ・ソースを用いる場合には問題にならないと思いますが、データの同期に使用する場合は問題になるかもしれません。その場合はデータ・ソースにフラッシュバック・クエリを検討する必要があります。

データ同期に使用するREST APIの定義の説明は以上です。

すこし脇道にそれましたが、Oracle APEXのアプリケーション開発の一助になれば幸いです。

次の記事は、Oracle APEXのアプリケーションで、宣言的にデータ同期を行う設定を行ってみます。

続く

[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.
