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。

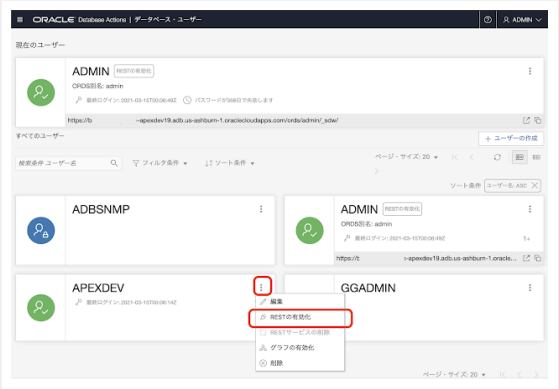
RESTデータ・ソースを使ったデータの同期については、以前に記事を書いています。この記事では単に同期化を設定しただけですが、こちらは複数のアクションを定義するなど、より実践的な内容になっています。なので、ちょっと自分でも試してみて、その過程を記載します。

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 Databse 19cを使用してください。

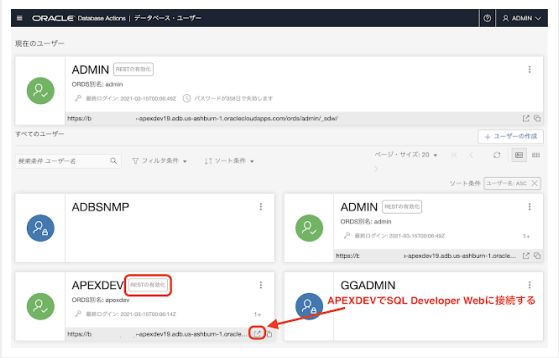
SQL Developer WebにADMINでサインインします。**管理**の**データベース・ユーザー**を開きます。

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。

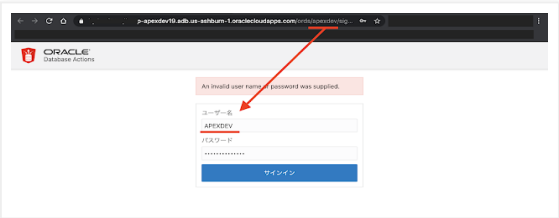


承認が必要を**ON**にし、**REST対応ユーザー**をクリックします。

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。

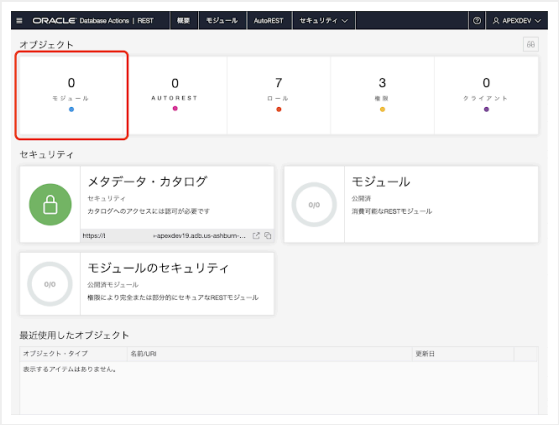


それぞれのユーザーでSQL Developer Webにサインインするurlには、ユーザー名が含まれます。URLに含まれるユーザー名と同じ名前をユーザー名に指定します。"An Invalid user name or password was supplied."と表示されることがありますが、すでにADMINでサインインしていた場合であれば無視できます。



開発の**REST**を開いて、RESTサービスを実装します。

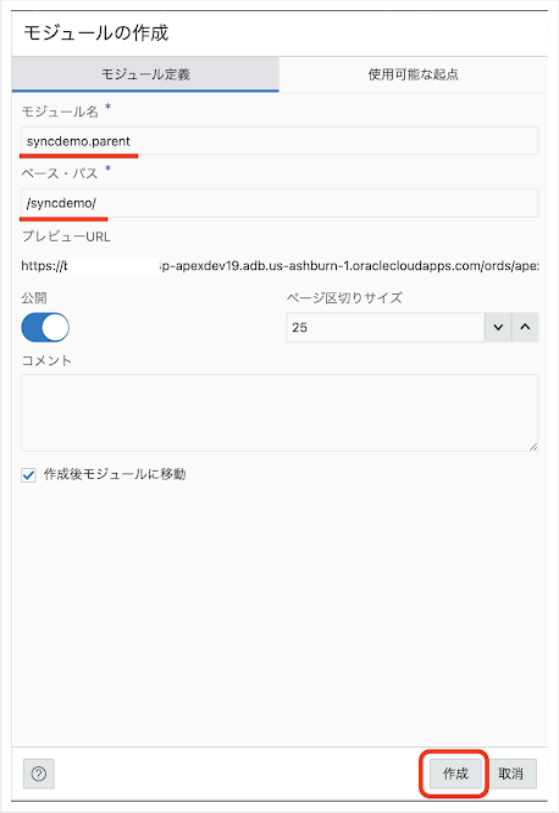
モジュールを開きます。



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。



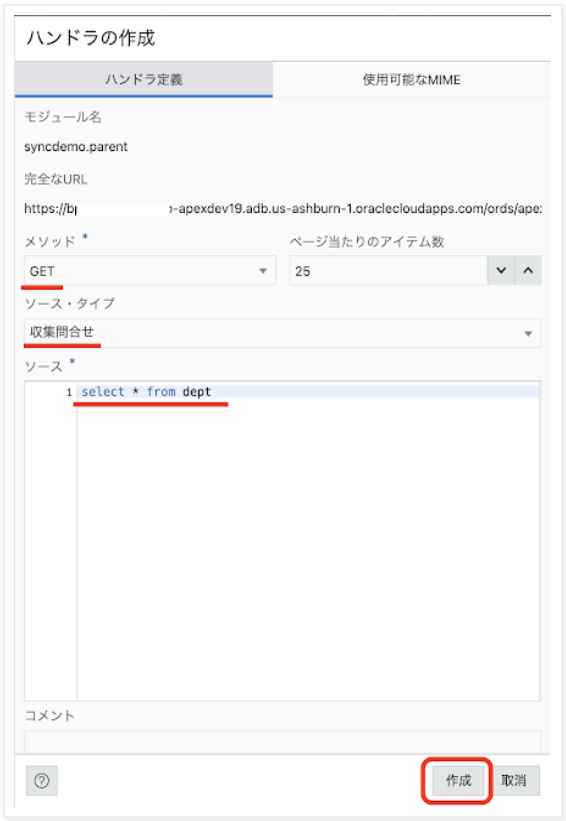
モジュールが作成されたので、続いて**テンプレートの作成**を実行します。

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。

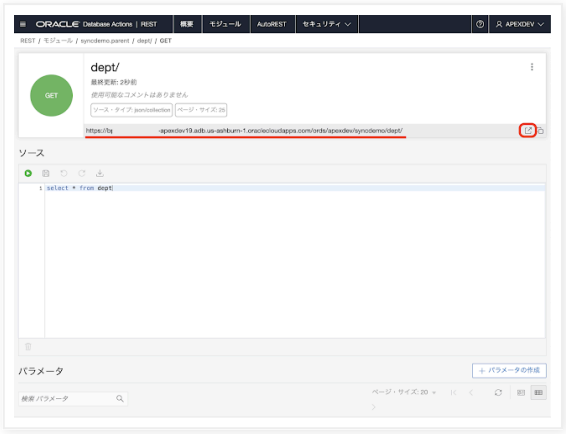
テンプレートが作成されたら、実際の処理を記述するハンドラをテンプレートに定義します。**ハンドラの作成**を実行します。

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。

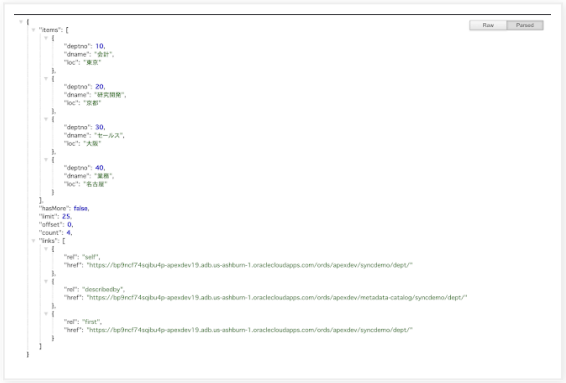
select * from dept



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。

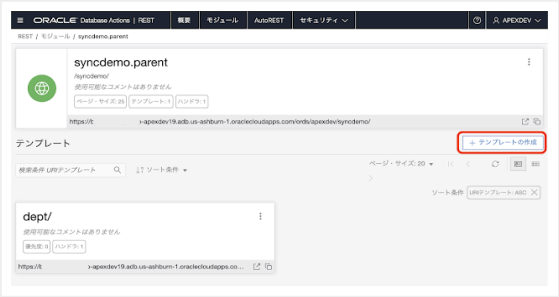


表DEPTの内容がJSONで返されることが確認できます。



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。

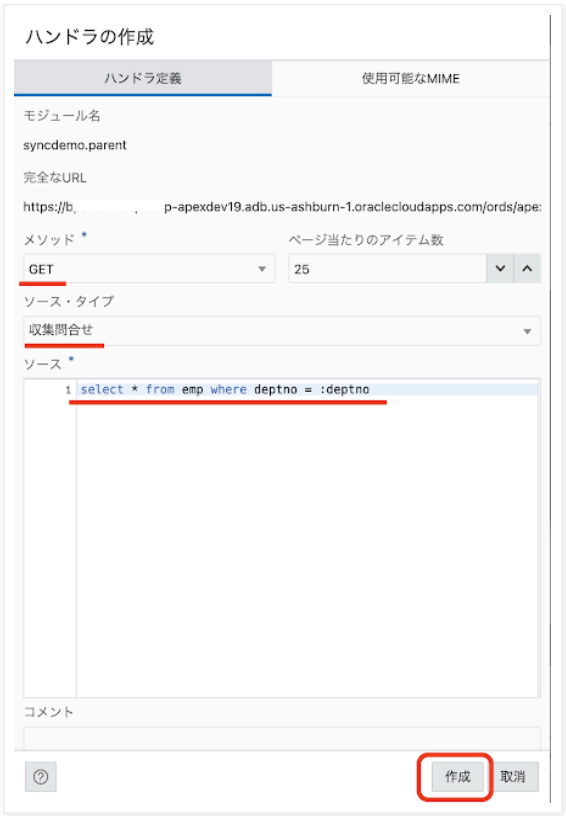


URIテンプレートとして**emp/:deptno**を設定します。コロンで始まる文字列はパラメータとして扱われます。つまり、このREST APIは、実際にはemp/10、emp/20といったURIから呼び出されます。

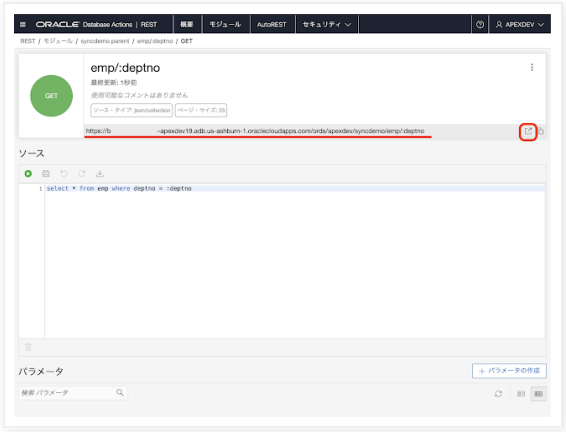
ハンドラの作成を行います。

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
```

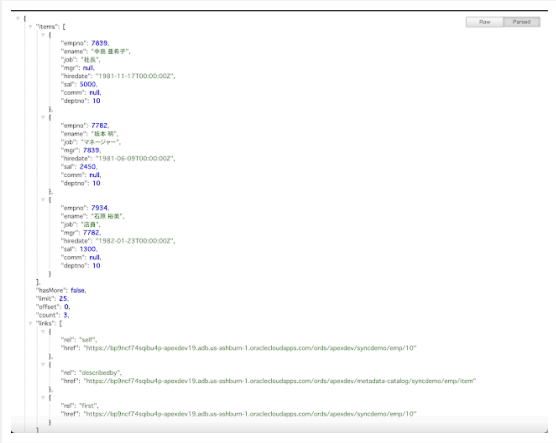


ハンドラが作成されたので、動作を確認します。



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。

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。



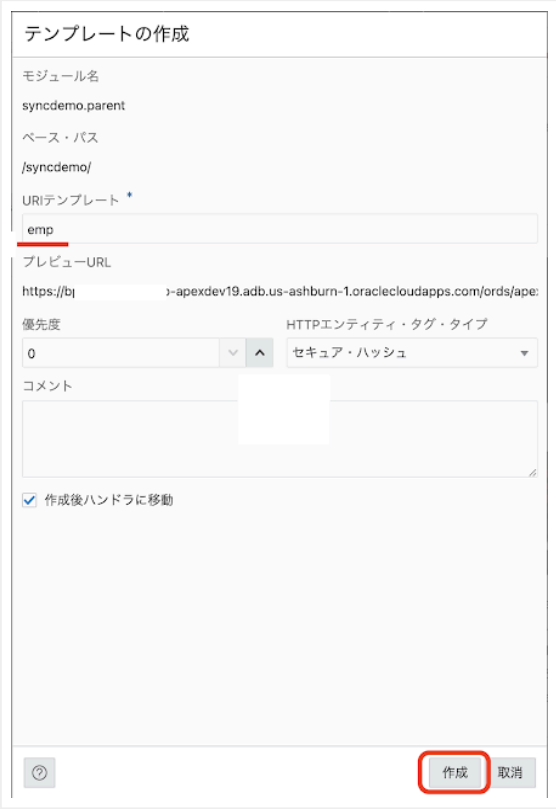
以上で元記事で行っている準備作業は完了です。

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。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。

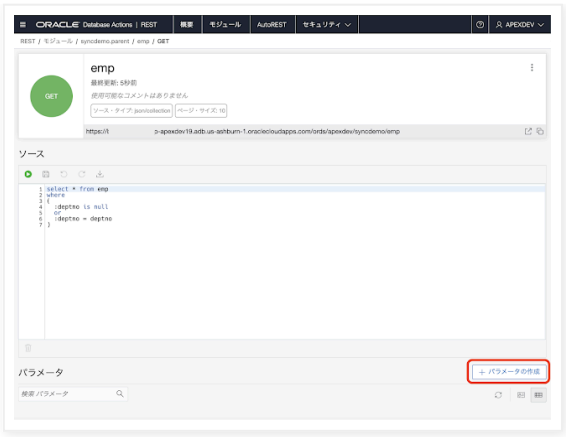
ハンドラのメソッドとして、GET以外にPOST、PUT、DELETEを選択することができます。

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。

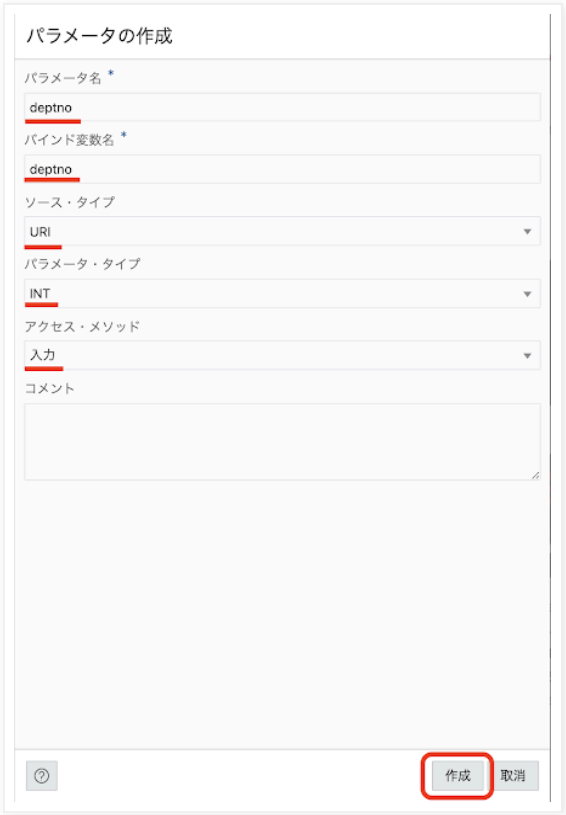
コレクション・アイテムの場合、（１行の）対象行がJSONオブジェクトとして返されます。収集問合せでは、返される行それぞれをJSONオブジェクトとしたJSON配列が返されます。

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。

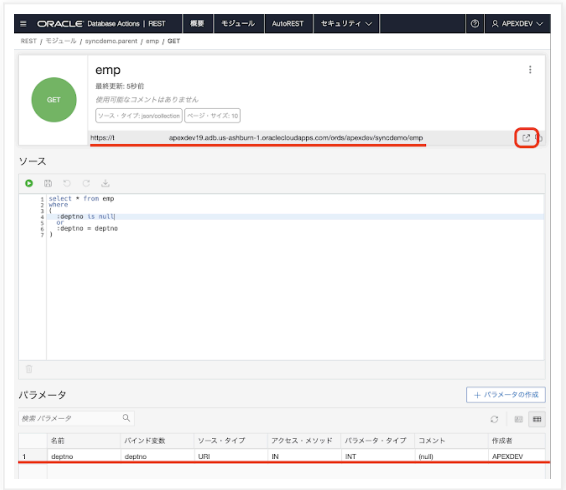
ハンドラが作成されたら、**パラメータの作成**をおこないます。



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。



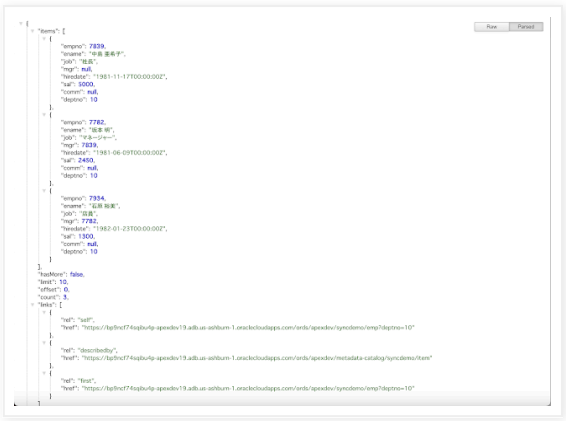
パラメータが登録されたので、動作を確認します。



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。

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。



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そうです。

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

Yuji N. 時刻: 14:05

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