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12月13日月曜日

ORDSとOracle RDF Graph Serverを同居させる

Oracle RDF Graph Serverを少し長い期間運用するため、すでにOracle REST Data Servicesを稼働させているコンピュート・インスタンスに同居させることにしました。ARMアーキテクチャのVM.Standard.A1.Flexのインスタンスがターゲットです。これもAlways Free枠で使用しています。

同居させる方法として、以下の２つの方法があるかと思います。

1. TomcatにORDSとOracle RDF Graph Serverの両方をデプロイする。
2. 別のサーバーにしてロード・バランサーで振り分ける。

今回は２番目の方法を採用しています。すでに実装済みのORDSは変更せずに、Oracle RDF Graph Serverをインストールします。SSLはロード・バランサにオフロードします。Oracle RDF Graphs Serverの環境構築についてはこちらの記事と同じ手順ですが、自動起動の設定も追加で行います。

ロード・バランサでリクエストをそれぞれのサーバーに振り分けるために、**ルーティング・ポリシー**を設定します。

Oracle RDF Graph Serverのセットアップ

Oracle RDF Graph Serverは、ユーザー**oracle**で動作させます。Jettyも含め、インストール先は**/home/oracle**以下とします。前提となる作業は以下になります。

1. ユーザー**oracle**の作成
2. Jetty 9.x(今回は**jetty-distribution-9.4.44.v20210927.zip**)の/home/oracleへの配置
3. **orardf-21.4.0.war**の/home/oracleへの配置
4. ディレクトリ**/home/oracle/rdf-doc**の作成と、**orardf_swagger.json**、**orardf_doc_url.txt**の配置

以下、実施する作業を列記します。それぞれの手順については（自動起動関連を除き)、以前の環境構築の記事で説明しています。

1. JDK1.8のインストール

ユーザー**opc**で実施します。Oracle REST Data ServicesはJDK17で動かしていますが、Oracle RDF Graph Serverはそれでは動かなかったので、JDK1.8をインストールします。

sudo dnf -y install jdk1.8

複数のJDKがインストールされている場合、デフォルトの/usr/bin/javaのバージョンがJDK 1.8に変わっていないか確認します。

/usr/bin/java -version

インストール前のバージョンと同じであれば、問題ありません。

2. Jetty 9.xの展開

ユーザー**oracle**で実施します。以下を実行します。Jettyの細かいバージョンは、Jettyのダウンロードする時期で変わってくるかと思います。

/home/oracleで実行します。

cd /home/oracle
unzip jetty-distribution-9.4.44.v20210927.zip

作成されたディレクトリ**jetty-distribution-9.4.44.v20210927**にjettyでアクセスできるよう、ディレクトリの名称を**jetty**に変更します。

mv jetty-distribution-9.4.44.v20210927 jetty

3. orardf-21.4.0.warをjetty/webapps以下にコピー

Oracle RDF Graph ServerのWebアプリケーション・アーカイブをJettyにデプロイします。

cp orardf-21.4.0.war jetty/webapps/orardf.war

4. Basic認証を使ったorardf.xmlの配置

webapps以下にorardf.xmlを作成します。Basic認証を使います。

5. jetty/etc以下にrealm.propertiesを作成

ユーザー名、パスワードおよびロールの割り当てを記載したファイル**realm.properties**を、**jetty/etc**以下に作成します。ユーザー名は**admin**で決めうちにしていますが、**パスワード**はそれぞれ設定します。

echo "admin=" `java -cp jetty/lib/jetty-util-9.4.44.v20210927.jar org.eclipse.jetty.util.security.Password admin パスワード |& grep CRYPT`",rdf-admin-user" > $HOME/jetty/etc/realm.properties

6. ディレクトリ/home/oracle/workspaceの作成

Oracle RDF Graph Serverの構成情報を保存するディレクトリを作成します。

mkdir /home/oracle/workspace

複数のサーバーでロード・バランスをさせる場合、最低限、このディレクトリはサーバー間で共有する必要があります。今回はそこまでの用途ではないのでロード・バランサは構成しますが、バックエンドのサーバーとしては 1 台だけにOracle RDF Graph Serverを実装します。

7. ポート番号の変更

ポート8080番はOracle REST Data Servicesがすでに使用しているため、Oracle RDF Graph Serverには**8081**番を割り当てます。**jetty/start.ini**の以下の部分を変更します。

```
## Connector port to listen on
jetty.http.port=8081
```

8. 起動オプションの設定

ファイル**/home/oracle/.orardfrc**に、以下のようにJettyの起動オプションを記述します。この後でOracle RDF Graph Serverの起動スクリプトの名称を**orardf**とするため、設定ファイルが**.orardfrc**となっています。

```
export JAVA=/usr/java/jdk1.8.0_311-aarch64/bin/java
export JAVA_OPTIONS="-Doracle.rdf.workspace.dir=/home/oracle/workspace -Dfile.encoding=UTF-8"
export JETTY_HOME=/home/oracle/jetty
export JETTY_PID=/home/oracle/jetty/jetty/orardf.pid
```

9. firewalldの接続許可

ユーザー**opc**にて実施します。firewalldにて、ポート8081番への接続を許可します。

sudo firewall-cmd --add-port=8081/tcp
sudo firewall-cmd --list-all
sudo firewall-cmd --runtime-to-permanent

```
[opc@cmordsa1 ~]$ sudo firewall-cmd --add-port=8081/tcp
success
[opc@cmordsa1 ~]$ sudo firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp0s3
  sources:
  services: ssh
  ports: 443/tcp 80/tcp 8081/tcp
  protocols:
  forward: no
  masquerade: no
  forward-ports:
        port=443:proto=tcp:toport=8443:toaddr=
        port=80:proto=tcp:toport=8080:toaddr=
  source-ports:
  icmp-blocks:
  rich rules:
[opc@cmordsa1 ~]$ sudo firewall-cmd --runtime-to-permanent
success
[opc@cmordsa1 ~]$
```

10. gitのインストール

ユーザー**opc**にて実施します。Swagger UIをインストールするため、gitをインストールします。

sudo dnf -y install git

11. Swagger UIのインストール

ユーザー**oracle**にて実施します。GitHubにあるSwagger UIをインストールします。

cd /home/oracle

git clone https://github.com/swagger-api/swagger-ui.git

12. Swagger UIの有効化

ユーザー**oracle**にて実施します。コンピュート・インスタンス上の静的ファイルに外部からアクセスできるように、**jetty/webapps**以下にswagger-ui.xmlおよびrdf-doc.xmlを作成します。/home/**opc**と記載されている部分は/home/**oracle**に書き直します。

13. 自動起動の設定

ユーザー**opc**にて実施します。**/etc/systemd/system/orardf.service**を作成し、以下の内容を記載します。

```
[Unit]
Description=Oracle RDF Graph Server
After=syslog.target network.target remote-fs.target nss-lookup.target

[Service]
Type=forking
ExecStart=/etc/init.d/orardf start
ExecStop=/etc/init.d/orardf stop
ExecReload=/etc/init.d/orardf restart
User=oracle

[Install]
WantedBy=multi-user.target
```

/home/oracle/jetty/bin/jetty.shを**/etc/init.d/orardf**にコピーします。

sudo cp /home/oracle/jetty/bin/jetty.sh /etc/init.d/orardf
sudo chmod 755 /etc/init.d/orardf

自動起動を有効にします。

sudo systemctl enable orardf

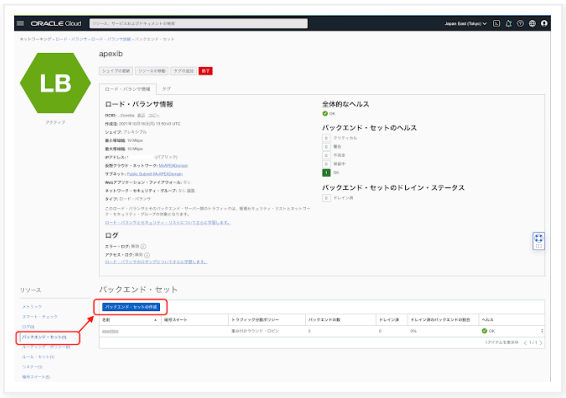
以上で、Oracle RDF Graph Serverのインストールと構成は完了です。Oracle RDF Graph Serverを起動します。

sudo systemctl strart orardf

ロード・バランサの設定

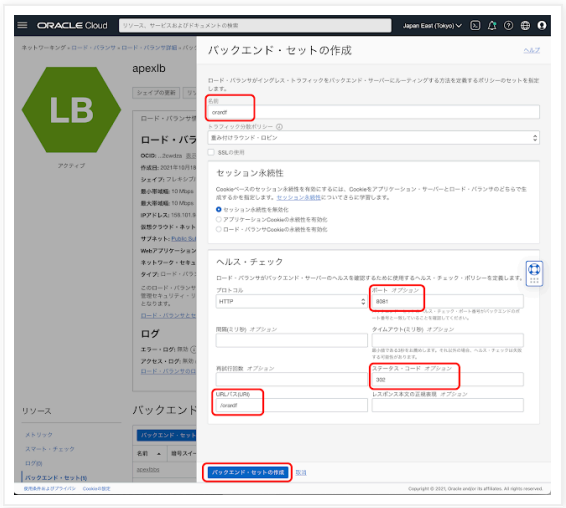
構成済みのロード・バランサに、新たにバックエンド・セットを作成します。

ロード・バランサの**リソース**より**バックセンド・セット**を選択し、**バックエンド・セットの作成**をクリックします。



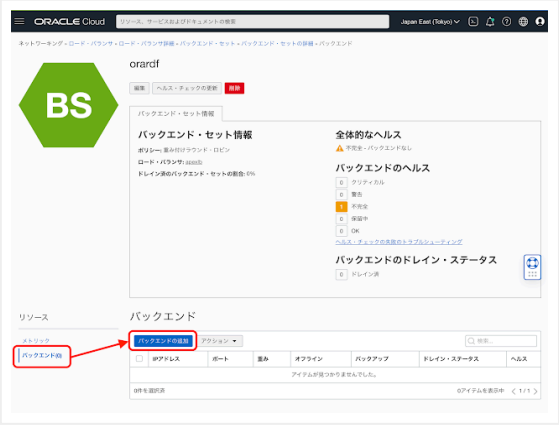
ドロワーが開きます。**名前**を**orardf**、**ポート**は**8081**、**ステータス・コード**は**302**、**URLパス(URI)**には**/orardf**を指定します。それ以外はデフォルトの設定（**トラフィック分散ポリシー**は**重み付けラウンド・ロビン**、**SSLの使用**は**OFF**）とします。

バックエンド・セットの作成をクリックします。



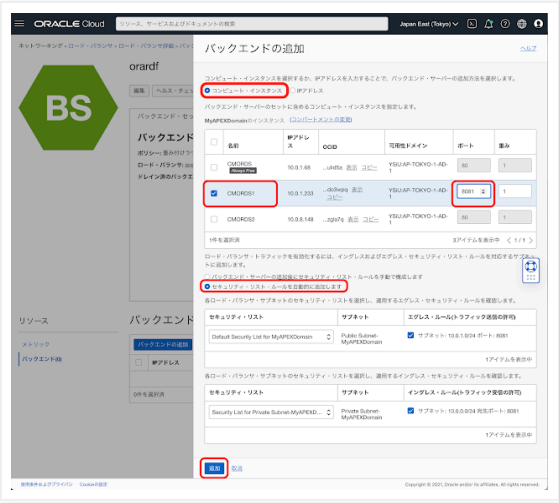
バックエンド・セットが作成されたら、バックエンドの作成を行います。作成されたバックエンド・セットを開きます。

バックエンド・セットの**リソース**より**バックエンド**を選択し、**バックエンドの追加**をクリックします。

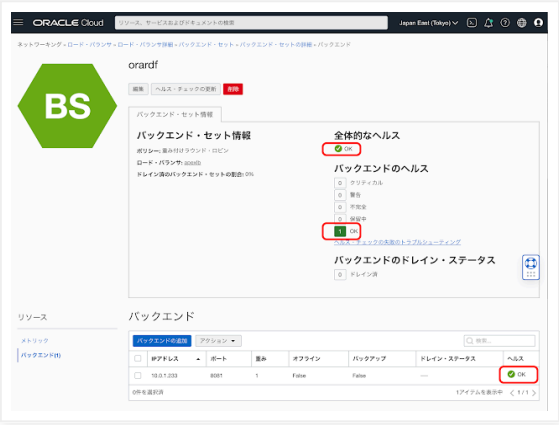


ドロワーが開きます。**バックエンドはコンピュート・インスタンスから選択**するオプションを選び、Oracle RDF Graph Serverを実装した**インスタンスにチェック**を入れます。**ポート**は**8081**に変更します。**セキュリティ・リスト・ルールを自動的に追加します**を選択します。追加が必要なイングレス・ルールやエグレス・ルールはネットワークの構成で変わってきます。

追加をクリックします。

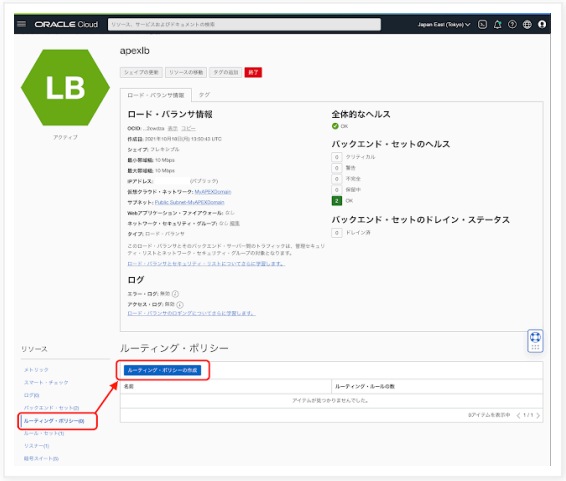


Oracle RDF Graph Serverが起動していれば、各種**ヘルス**チェックが**OK**に変わります。



Oracle REST Data ServicesとOracle RDF Graph Serverのバックエンド・セットが作成されました。受け付けたリクエストをぞれぞれにバックエンド・セットに振り分けるルーティング・ポリシーを作成します。

ロード・バランサの**リソース**の**ルーティング・ポリシー**を選択し、**ルーティング・ポリシーの作成**をクリックします。

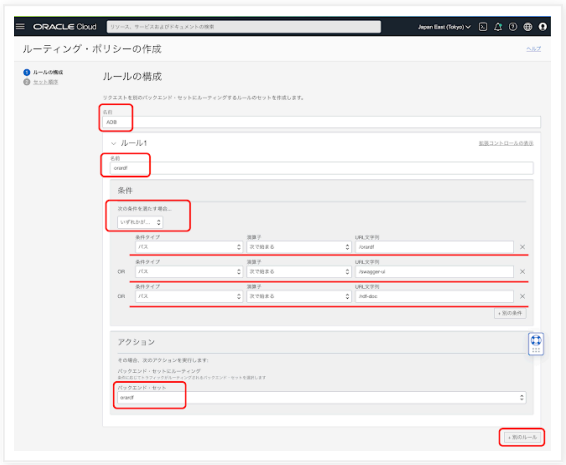


新たに作成するルーティング・ポリシーの**名前**を**ADB**とします。

ルール 1 として、Oracle RDF Graph Serverへのルーティング・ルールを設定します。

名前を**orardf**とします。**条件**の**次の条件を満たす場合...**は**いずれかが一致した場合**を選択します。すべての条件で、**条件タイプ**として**パス**、**演算子**は**次で始まる**を選択し、**URL文字列**として**/orardf**、**/swagger-ui**、**/rdf-doc**を指定します。**アクション**の**バックエンド・セット**として、Oracle RDF Graph Serverである**orardf**を選択します。

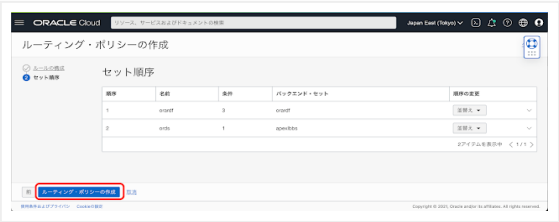
Oracle REST Data Servicesへのルーティング・ルールを作成するため、**別のルール**をクリックします。



Oracle REST Data Servicesのルールの**名前**は**ords**とします。**次の条件を満たす場合...**は**すべてが一致した場合**（条件は一つだけなので、どちらでも結果は同じ）を選び、**条件**として**条件タイプ**は**パス**、**演算子**は**次で始まる**、**URL文字列**として**/ords**を指定します。**アクション**の**バックエンド・セット**として**Oracle REST Data Servicesのバックエンド・セット**（この例では**apexlbbs**）を選択します。

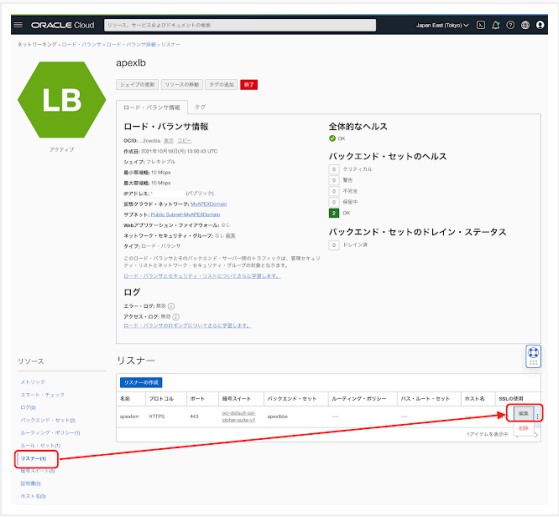
次に進みます。

セット順序の変更は不要なので、そのまま**ルーティング・ポリシーの作成**をクリックします。



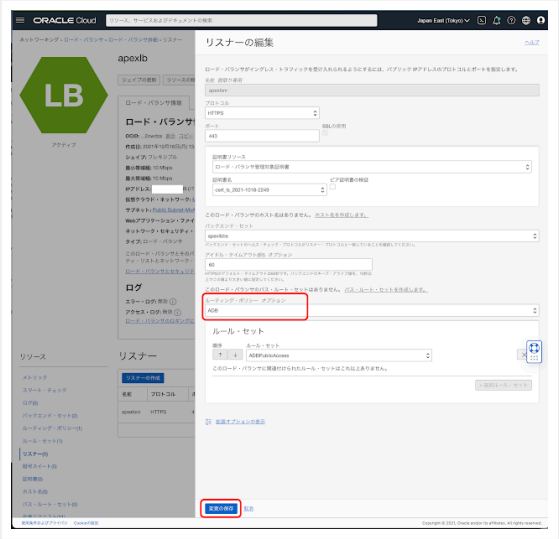
ルーティング・ポリシーが作成されます。

作成したルーティング・ポリシーをリスナーに適用します。ロード・バランサの**リソース**の**リスナー**を開き、作成済みのリスナーの**編集**を行います。



リスナーの編集を行うドロワーが開きます。その中の**ルーティング・ポリシー**として、先ほど作成した**ルーティング・ポリシー**（今までの例では**ADB**）を選択します。

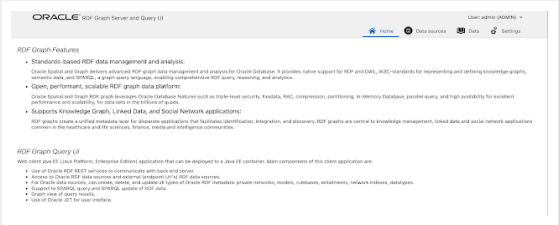
変更の保存をクリックします。



リスナーへのルーティング・ポリシーの適用が完了すると、すべての設定は完了です。

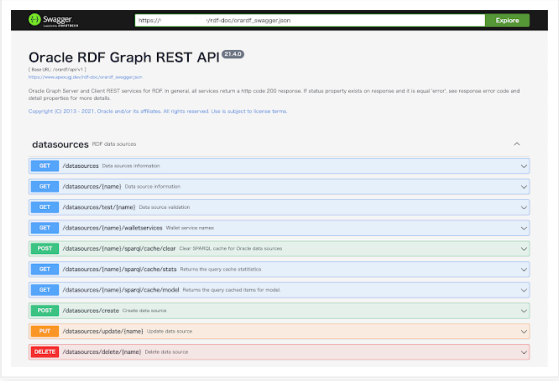
Oracle RDF Graph Serverにアクセスし、動作を確認します。

https://ホスト名/orardf/



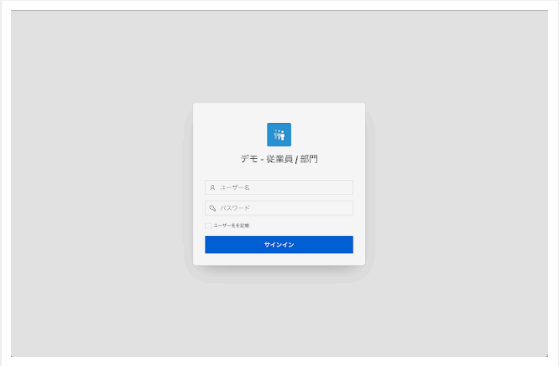
Swagger UIにアクセスします。

https://ホスト名/swagger-ui/dist/
https://ホスト名/rdf-doc/orardf_swagger.json



Oracle REST Data Services（Oracle APEX）にアクセスします。（こちらの記事に記載した、外部ネットワークからの管理ツール、開発ツールのアクセスを禁止した環境であるため、サンプル・サプリケーションのサインイン画面が表示されています。）

https://ホスト名/ords/



以上で確認作業も完了です。

SSLのオフロードを行なっていますが、こちらの記事と同様のCSRF対応を行うことにより、Oracle RDF Graph ServerのREST APIの呼び出しは成功しています。

以上になります。

Oracle APEXの環境構築の参考になれば幸いです。

完

Yuji N. 時刻: 15:45

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