USING TEIID DESIGNER TO CONSUME A REST-BASED WEB SERVICE AS A RELATIONAL MODEL

Blaine Mincey
10/17/2011

The purpose of this paper is to provide an introduction on how to consume a REST-based Web Service as a relational data source using the Teiid Designer which is part of the JBoss Developer Studio (JBDS) Eclipse-based IDE. The Teiid Designer is a visual tool that enables rapid, model-driven definition, integration, management, and testing of data services. This tool is part of the Enterprise Data Services Platform (EDSP) product. EDSP is a powerful set of tools and runtime components that makes it easy for your applications and business processes to integrate and use data from many data sources.

CONTENTS

OVERVIEW

This technical brief will guide you through the steps required to build a Teiid project that will consume a REST-based web service as a relational data model. This relational model can then be used in conjunction with other virtualized relational models to further extend your data abstraction layer. Specifically, the following steps will be performed:

1. Create and deploy a RESTful service that can be invoked by the Enterprise Data Services Platform
2. Create a project in JBoss Developer Studio using the Teiid Designer
3. Model the physical structure of the RESTful service
4. Connect to the RESTful service through JBoss Developer Studio using a Connection Profile
5. Create the virtual abstraction layer for the data returned by the RESTful service
6. Transform the data into the virtual abstraction layer
7. Test the newly created virtual relational model

ENVIRONMENT

The following environment was utilized in order to produce the required artifacts for this example. It should be noted that this example should work with slightly different version numbers of the listed tools. However, different tool configurations were not tested.

Operating System: RHEL 6.1 (Santiago)

JVM: Sun/Oracle 1.6.0_25 64-bit

NOTE: It should be noted that this specific environment is not required. There are JBDS binaries for not only RHEL, but Windows and Mac-OSX platforms as well. Additionally, the JBoss Enterprise Data Services Platform is supported with a compatible JVM so it is possible to run it on a variety of operating systems including RHEL, Windows, and Solaris.

There is an example REST-based Web Service available within a public repository on GitHub (http://www.github.com) that was utilized as the data source for this paper. Instructions will be provided later in the paper on how to acquire the source artifacts from GitHub as well as how to build and deploy the application. The following tools were used for the REST-based Web Service.

Git: 1.7.1

Maven: 3.0.3

EXAMPLE REST-BASED WEB SERVICE

In order to provide an easy way to get started with this example, a sample REST-based Web Service is available for download. No knowledge of REST or web services is required to build and deploy the application. A general understanding of both Git and Maven is required. Additional information on Git is available at http://git-scm.com/. This link provides information on how to setup Git for your target environment. Additionally, Maven information can be found at http://maven.apache.org/. This link provides a tutorial as well as the ability to download the Maven binaries. If this is your first time using Maven, you will need to point your Maven settings to the JBoss Maven repositories in order to access JBoss artifacts. The following article describes how to do this: http://community.jboss.org/wiki/MavenGettingStarted-Users.

The public GitHub repository for the example REST-based Web Service is at
https://github.com/blainemincey/CustomerRESTWebSvc.

In order to download the repository to your local workstation, execute the following command from a command or terminal window:

```
git clone git://github.com/blainemincey/CustomerRESTWebSvc.git
```

If git has been configured correctly on your system, this command will copy the public repository to your local workstation. After the command is completed, there should be a new directory, CustomerRESTWebSvc. Now, ‘cd’ (change directory) into this newly created directory. There should be a pom.xml file, a README file, and a src directory.

At this point, if Maven has been correctly installed on your system, you can simply execute the following Maven command in order to build the WAR file that will be deployed to the Data Services Platform.

```
mvn clean compile war:war
```

This command will compile the source files as well as build a deployable WAR file. This command will create a ‘target’ directory. This newly created directory will contain the CustomerRESTWebSvc.war file. This file can simply be copied to the running configuration’s ‘deploy’ directory of the Enterprise Data Services Platform. In order to manually test the web service to ensure it is working correctly, open a browser to the following location:

http://localhost:8080/CustomerRESTWebSvc/MyRESTApplication/customerList

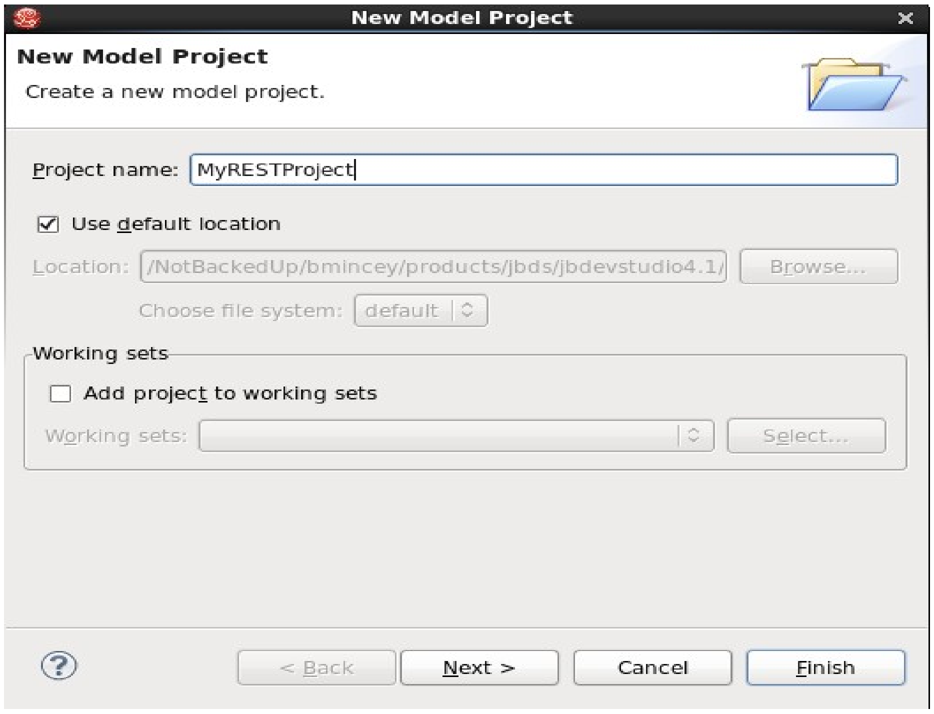
SAMPLE XML

The following XML snippet serves as an example of the literal XML result from the REST-based Web Service that was deployed in the previous step. This is simply a single record from the web service. It should be noted that well over 100 records will be returned upon executing the REST Web Service.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<customers>
      <customer>
            <customernumber>103</customernumber>
            <customername>Atelier graphique</customername>
            <contactlastname>Schmitt</contactlastname>
            <contactfirstname>Blaine</contactfirstname>
            <phone>40.32.2555</phone>
            <addressline1>54, rue Royale</addressline1>
            <addressline2 />
            <city>Nantes</city>
            <state />
            <postalcode>44000</postalcode>
            <country>France</country>
            <salesrepemployeenumber>1370</salesrepemployeenumber>
            <creditlimit>21000.0</creditlimit>
      </customer>
</customers>
```

CREATE NEW MODEL PROJECT

With JBDS opened in the Teiid Designer perspective, right-click within the Model Explorer section and select "New" and then "Teiid Model Project".

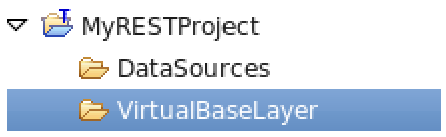


New Model Project
Create a new model project.

Project name: MyRESTProject
☑ Use default location
Location: /NotBackedUp/bmincey/products/jbds/jbdevstudio4.1/
Choose file system: default
Working sets
☐ Add project to working sets
Working sets:

Enter a "Project name". For the purposes of this paper, the project name "MyRESTProject" has been chosen.

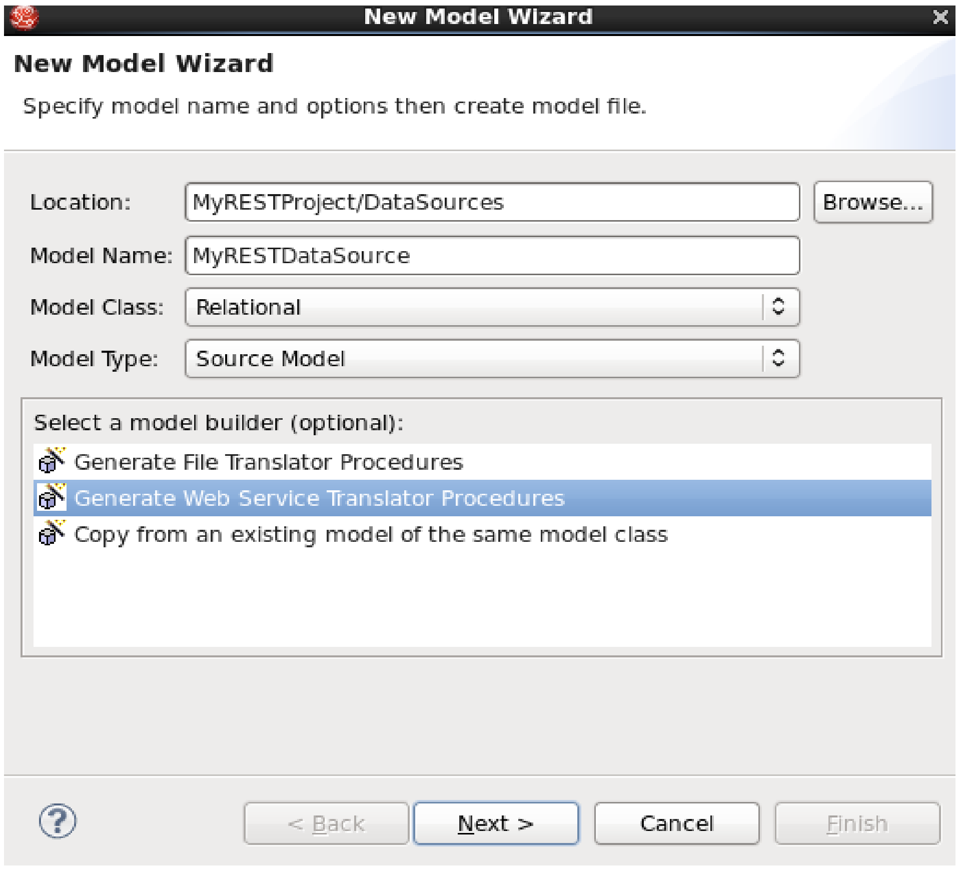
Next, following Teiid Designer best practices, create two folders within the project. This can be done by right-clicking on "MyRESTProject" in the Model Explorer view and selecting "New" and then "Folder". Create the following two folders: DataSources and VirtualBaseLayer. Your project should now resemble the image below.



PHYSICAL DATA SOURCE

The solution to many of today's data challenges is data virtualization. JBoss Enterprise Data Services Platform provides you with the ability to create a data abstraction, or data virtualization, layer on top of your physical data sources. It is then possible to create a variety of virtual data layers on top of your physical data sources and expose them as a single interface. For the purpose of this paper, you will be able to model a REST based web service and make it appear as a relational data source. From this point, it is possible to join the data source with other virtual models providing you with the means to turn the data you have into the information you need.

At this point, the physical data source can be modeled. Right-click on the folder "DataSources" and select "New" and then "Teiid Metadata Model".

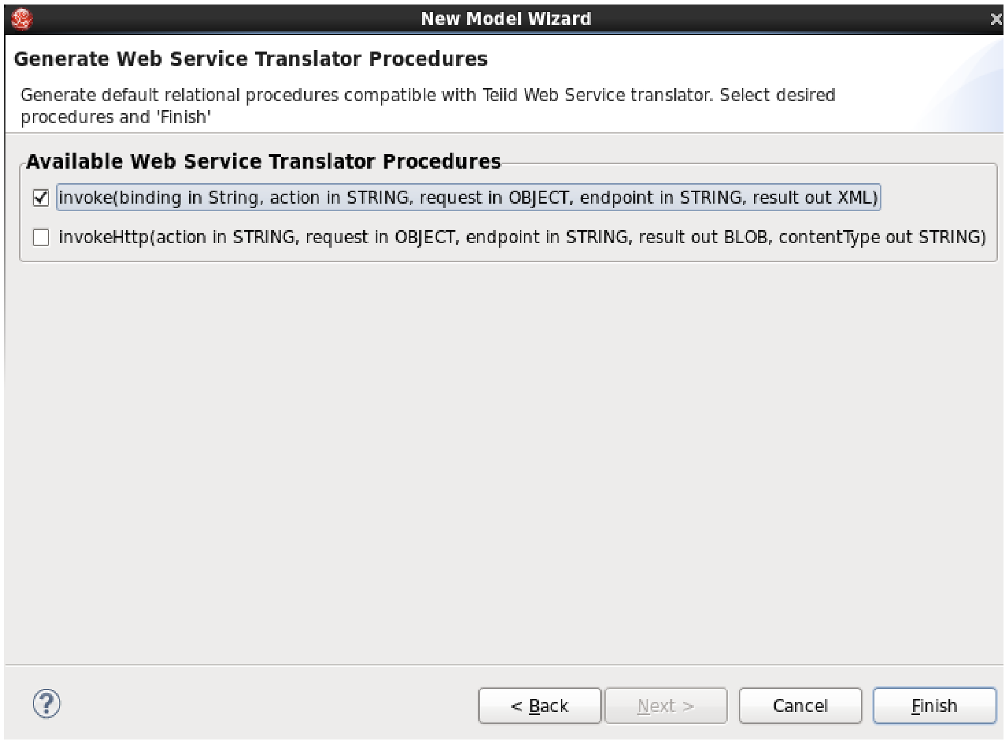


Location: MyRESTProject/DataSources
Model Name: MyRESTDataSource
Model Class: Relational
Model Type: Source Model

Select a model builder (optional):
- Generate File Translator Procedures
- Generate Web Service Translator Procedures
- Copy from an existing model of the same model class

For this example, the location of this metadata model should be within the datasources folder. As indicated above, the Model Name is "MyRESTDataSource", Model Class is "Relational", and Model Type is "Source".

It is also important to be sure and select “Generate Web Service Translator Procedures”. At this point, click the “Next” button in the bottom right-hand corner. This will bring up the window below.



New Model Wizard

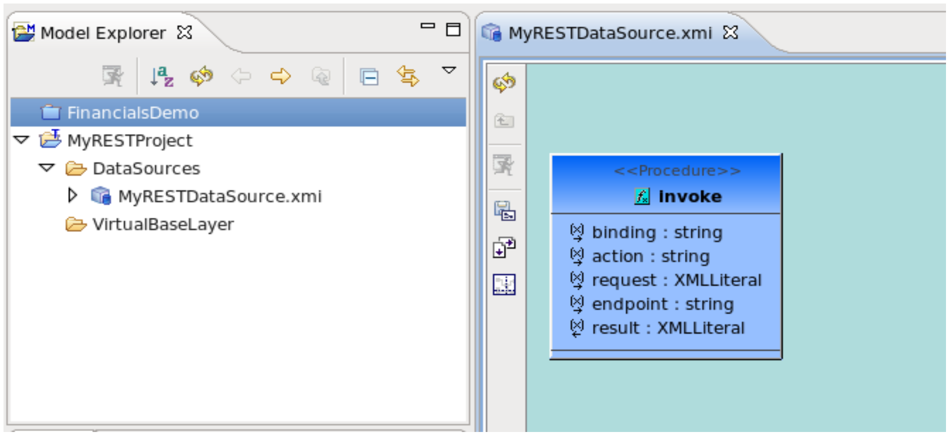
Generate Web Service Translator Procedures

Generate default relational procedures compatible with Teiid Web Service translator. Select desired procedures and 'Finish'

Available Web Service Translator Procedures

- [✓] invoke(binding in String, action in STRING, request in OBJECT, endpoint in STRING, result out XML)
- [ ] invokeHttp(action in STRING, request in OBJECT, endpoint in STRING, result out BLOB, contentType out STRING)

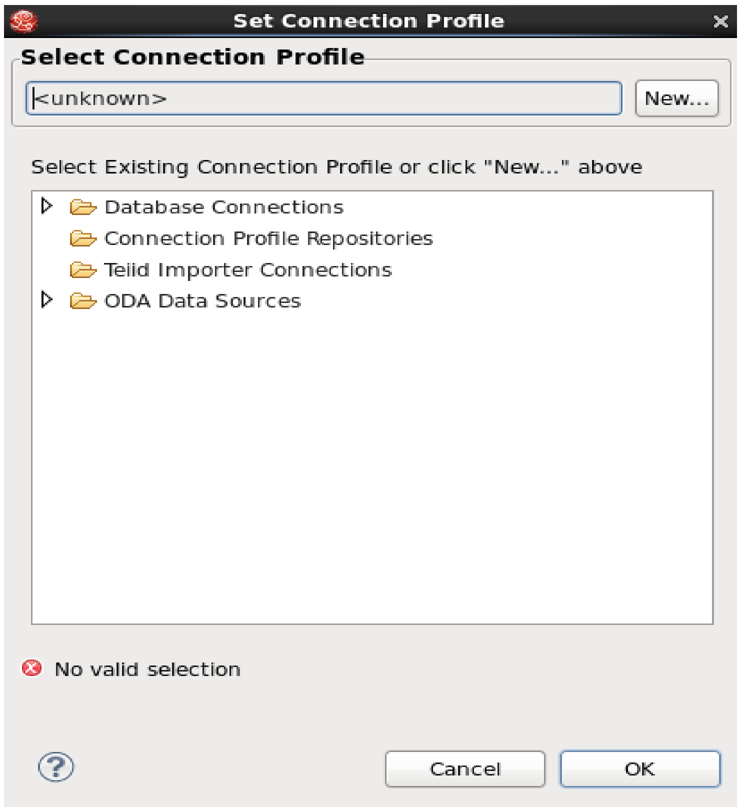
The window indicates that there are two separate Available Web Service Translator Procedures: invoke and invokeHttp. For the purposes of this example, select the first method which is “invoke”. After clicking the “Finish” button, the Teiid Designer perspective should resemble the image below.



CONNECTION PROFILE

A connection profile is used to identify the actual data source required for the physical data source representation within the Teiid Designer. In fact, it might be easier to associate this with being the *-ds.xml (data source connection file) that is required by the JBoss Enterprise Application Platform for JNDI based data sources. In other words, the connection profile defines the actual data source connection from within the context of JBoss Developer Studio.

At this point, it is important to point the metadata model to a connection profile. Right-click on "MyRESTDataSource.xmi". Select "Modeling" and then "Select Connection Profile". The following pop-up window should open.


Set Connection Profile

Select Connection Profile

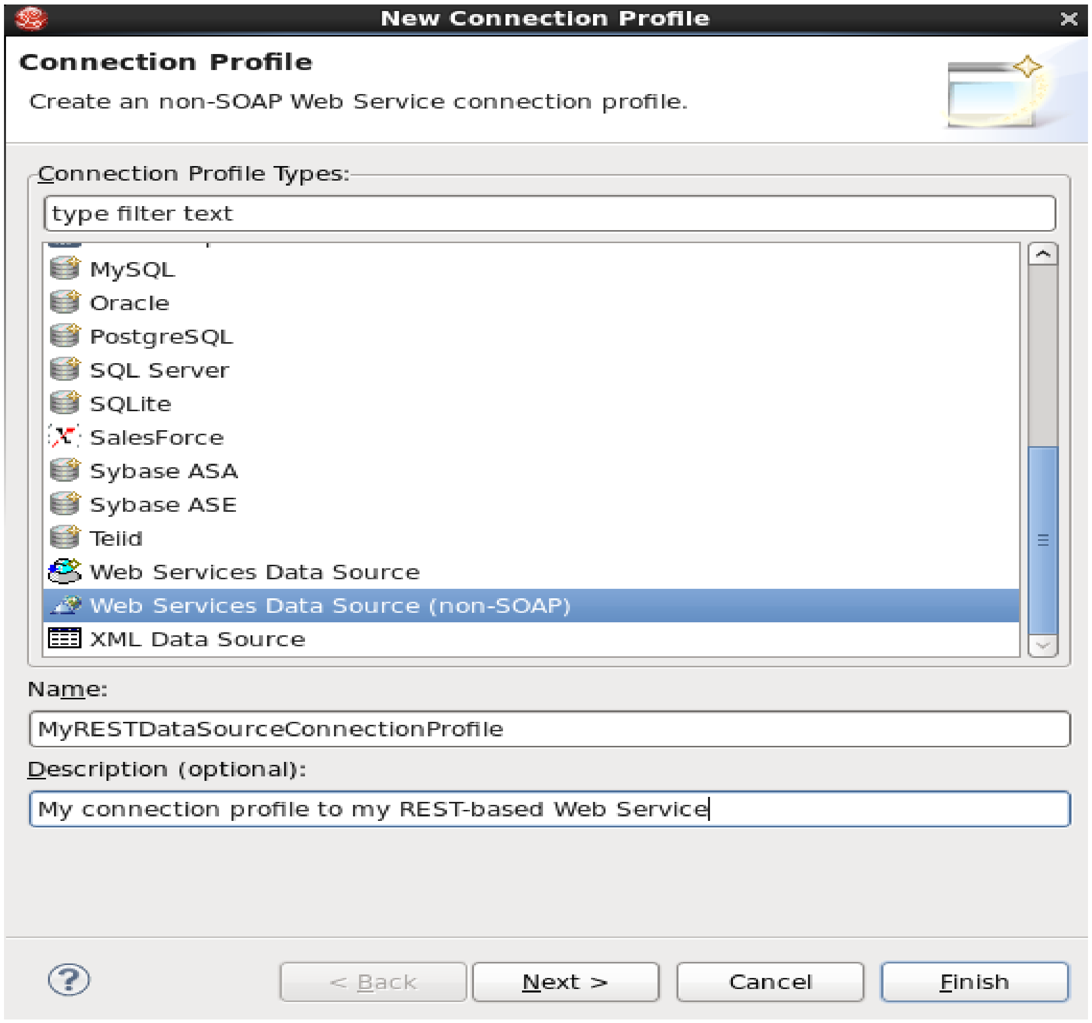
<unknown>

Select Existing Connection Profile or click "New..." above

- Database Connections
- Connection Profile Repositories
- Teiid Importer Connections
- ODA Data Sources

No valid selection

Click the “New” button along the top row to create a new connection profile. The “New Connection Profile” window should open as indicated in the image below.



New Connection Profile

Connection Profile

Create an non-SOAP Web Service connection profile.

Connection Profile Types:

type filter text

- MySQL
- Oracle
- PostgreSQL
- SQL Server
- SQLite
- SalesForce
- Sybase ASA
- Sybase ASE
- Teiid
- Web Services Data Source
- Web Services Data Source (non-SOAP)
- XML Data Source

Name:

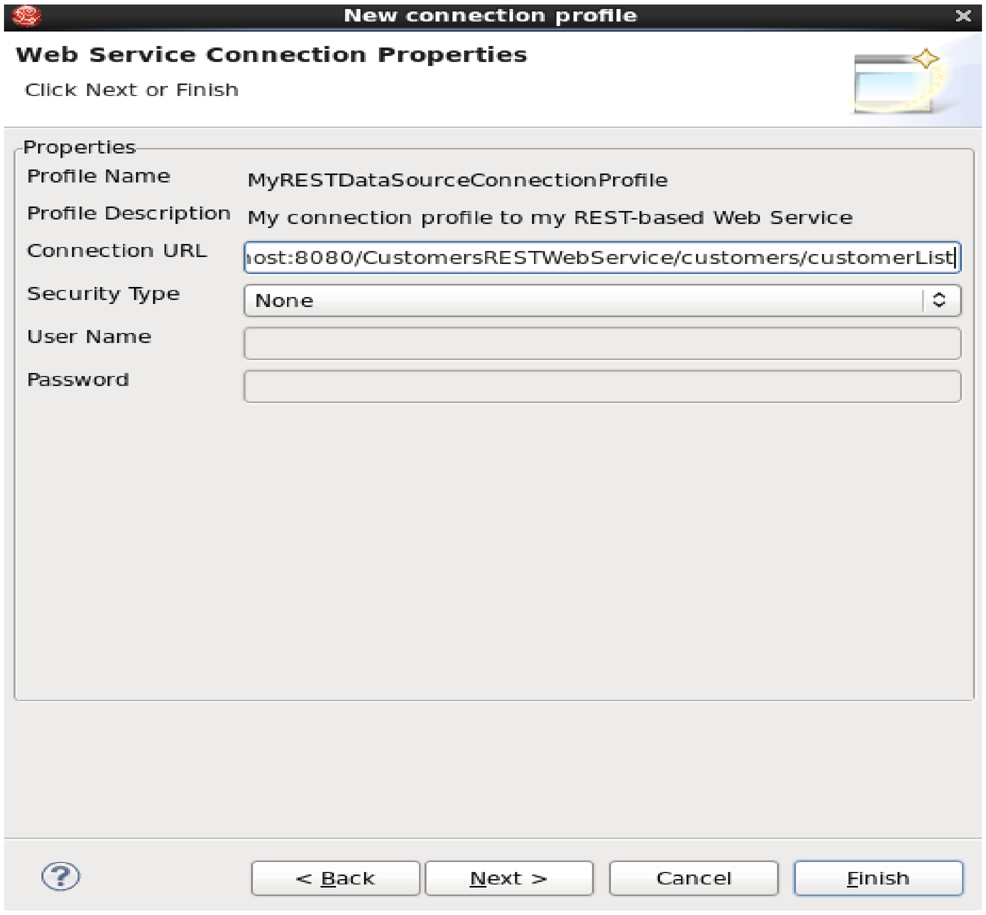
MyRESTDataSourceConnectionProfile

Description (optional):

My connection profile to my REST-based Web Service

Scroll down the list of available Connection Profile types. Select "Web Services Data Source (non-SOAP)". As indicated in the image, the name of the connection profile is "MyRESTDataSourceConnectionProfile" and the Description is "My connection profile to my REST-based Web Service".

Now, click the "Next" button along the bottom row. The following window will open.



New connection profile

Web Service Connection Properties

Click Next or Finish

Properties

Profile Name: MyRESTDataSourceConnectionProfile

Profile Description: My connection profile to my REST-based Web Service

Connection URL: ıost:8080/CustomersRESTWebService/customers/customerList

Security Type: None

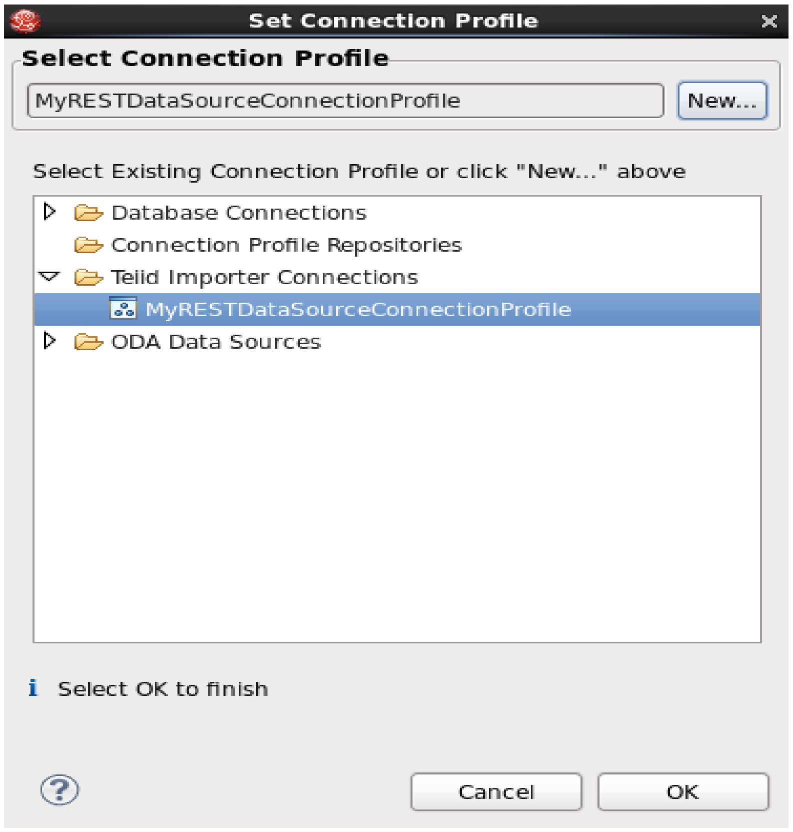
User Name:

Password:

Within this window, be sure to enter the "Connection URL" of the REST-based web service that is being modeled. Additionally, if the web service requires authentication, the values can be entered here as well. For purposes of this paper, a non-secured web service was utilized. For the purposes of this example, use the URL for the example REST-based Web Service that was deployed to the platform earlier. If you are using this REST-based Web Service, the Connection URL to use will be:

http://localhost:8080/CustomerRESTWebSvc/MyRESTApplication/customerList

If you are using a different REST-based Web Service, be sure to enter the correct URL for that one. Once this has been entered, click the "Finish" button.


Set Connection Profile

Select Connection Profile

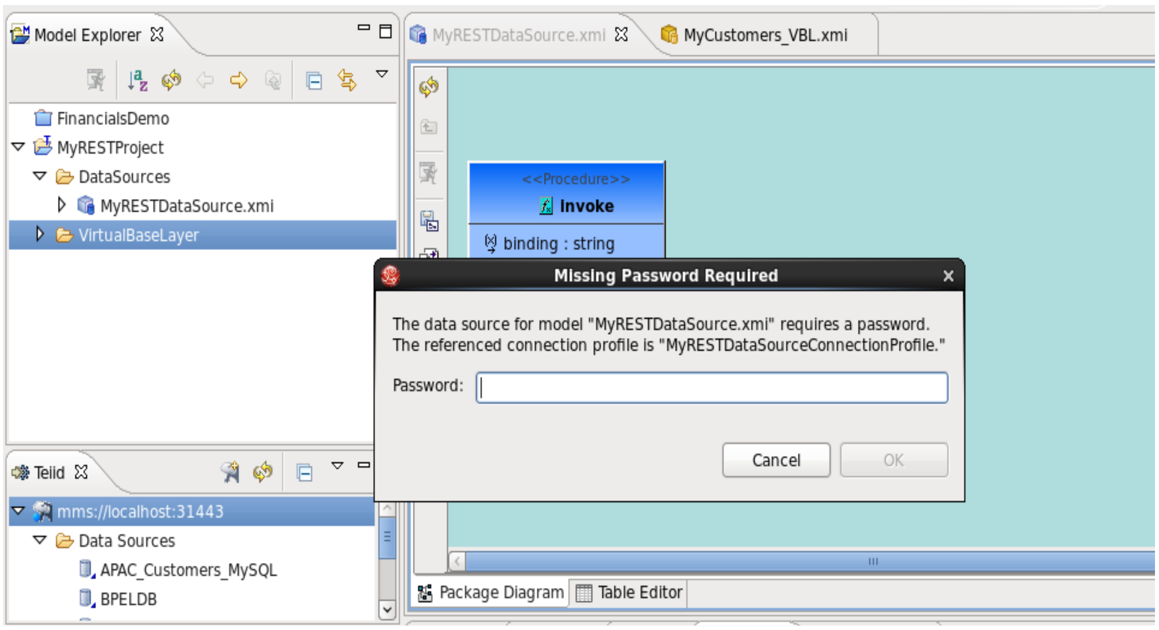
MyRESTDataSourceConnectionProfile

Select Existing Connection Profile or click "New..." above

- Database Connections
- Connection Profile Repositories
- Teiid Importer Connections
  - MyRESTDataSourceConnectionProfile
- ODA Data Sources

Select OK to finish

Now, be sure to select the Connection Profile that was just created: “MyRESTDataSourceConnectionProfile”.



Missing Password Required

The data source for model "MyRESTDataSource.xmi" requires a password.
The referenced connection profile is "MyRESTDataSourceConnectionProfile."

Password:

At this point, a pop-up window should appear requesting the password for the referenced connection profile. It is important to note that this password is required for this example to work. You did not overlook the password creation in an earlier section within this document because it was not required. The reason for a password is so that the Teiid Designer will be able to properly initiate the creation of a preview data source that is used strictly by the Teiid Designer. Again, this password simply needs to be any series of characters or a single character. It will simply allow the OK button to become enabled.

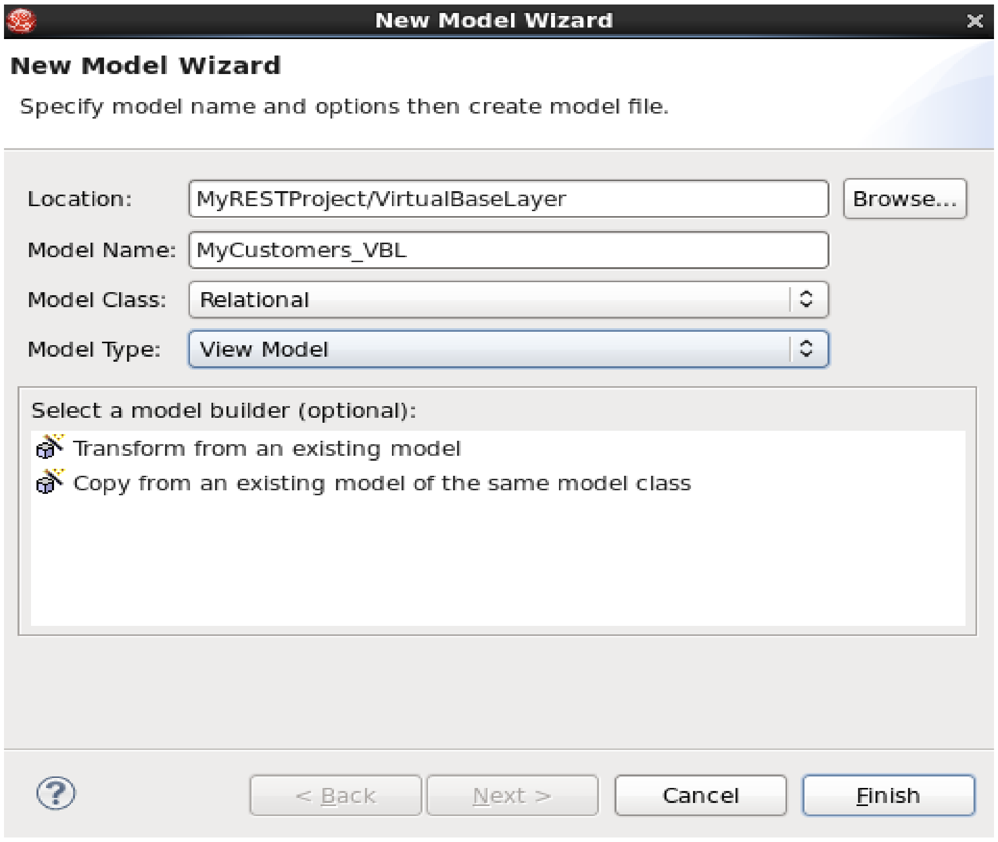
Be sure to enter **any** password. Again, this is required so that a PREVIEW data source file will be generated and deployed to the running EDSP instance that is configured with JBDS. For this example, the password used was “user” which is the default Teiid JDBC connection password.

VIRTUAL BASE LAYER

At this point, you should begin to be able to see the power of data virtualization. You have worked through creating essentially the pointers to the physical data source, the REST web service. Now, it is time to begin building the virtual layer on top of the physical data source. Ultimately, this provides you with the capabillity to model your virtual layers to be exactly what you need them to be. It would now be possible to "join" this virtual relational model with another virtual relational model and expose it as a single virtual table through an external interface. To your applications, these disparate data sources will now be able to appear as a single relational table within a virtual database!

Now that we have our physical data source modeled, we can create the virtual base layer that will act as our relational model. It should be noted that additional virtual layers can be built upon the virtual base layer.

Right-click on the "VirtualBaseLayer" folder. Select "New" and "Teiid Metadata Model".



New Model Wizard

Specify model name and options then create model file.

Location: MyRESTProject/VirtualBaseLayer

Model Name: MyCustomers_VBL

Model Class: Relational

Model Type: View Model

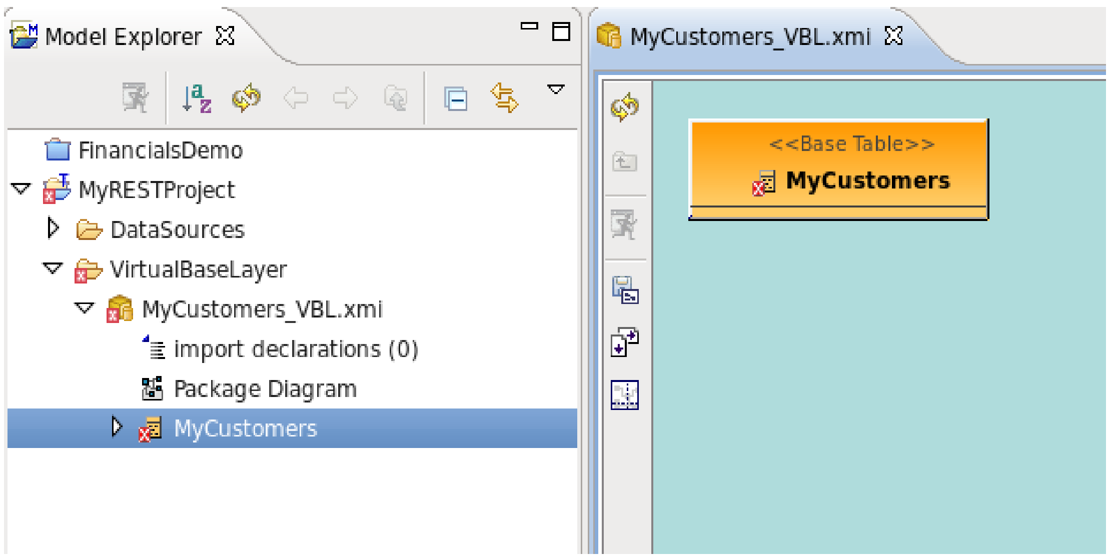
Select a model builder (optional):

Transform from an existing model

Copy from an existing model of the same model class

As indicated above, the New Model Wizard window will open. Be sure to check that the location of this model is within the VirtualBaseLayer folder. In this instance, the Model Name is "MyCustomers_VBL", the Model Class is "Relational", and the Model Type is"View Model". **Do not select a model builder from the list.** Now, click the "Finish" button.

Now right-click within the model that was just created. Select "New Child" and then "Base Table". Double-click within the Base Table and rename it to "MyCustomers". The Teiid Designer perspective should look like the image below.



Notice that there is a red "X" which indicates that there are errors within this model. This is due to not having any columns within this model. Now, double-click on the MyCustomers base table to open up the Transformation Editor.

At this point, it is extremely important to note the format of the literal XML data that is returned from your REST-based web service. The following transformation is specific to the XML snippet that is referenced at the beginning of this document.

TRANSFORMATION

The following transformation can simply be copied and pasted within the Transformation Editor in the Teiid Designer perspective.

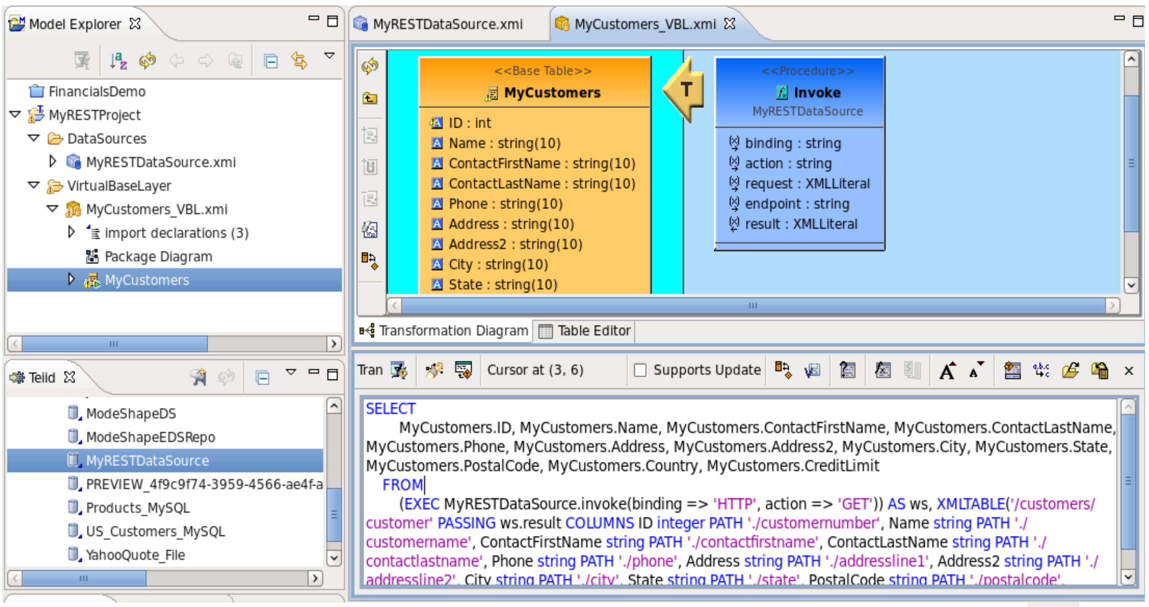
```
SELECT
        MyCustomers.ID,
        MyCustomers.Name,
        MyCustomers.ContactFirstName,
        MyCustomers.ContactLastName,
        MyCustomers.Phone,
        MyCustomers.Address,
        MyCustomers.Address2,
        MyCustomers.City,
        MyCustomers.State,
        MyCustomers.PostalCode,
        MyCustomers.Country,
        MyCustomers.CreditLimit

FROM
        (EXEC MyRESTDataSource.invoke(binding => 'HTTP', action => 'GET')) AS ws,
        XMLTABLE('/customers/customer' PASSING ws.result COLUMNS
        ID integer PATH './customernumber',
        Name string PATH './customername',
        ContactFirstName string PATH './contactfirstname',
        ContactLastName string PATH './contactlastname',
        Phone string PATH './phone',
        Address string PATH './addressline1',
        Address2 string PATH './addressline2',
        City string PATH './city',
        State string PATH './state',
        PostalCode string PATH './postalcode',
        Country string PATH './country',
        CreditLimit bigdecimal PATH './creditlimit')

AS MyCustomers
```

There are a few important items to take note of. This is simply a standard SQL SELECT statement integrated with some Teiid constructs and XPath. The SELECT portion is very easy to follow as these are the columns that will make up our base table, MyCustomers. The FROM clause executes the physical data source that was created in earlier steps. Then, the XMLTABLE function was utilized in order to query the result from the web service and parse the elements. When parsing the various elements, be sure to utilize correct data types as well as the XPath elements that correspond to your literal XML result.

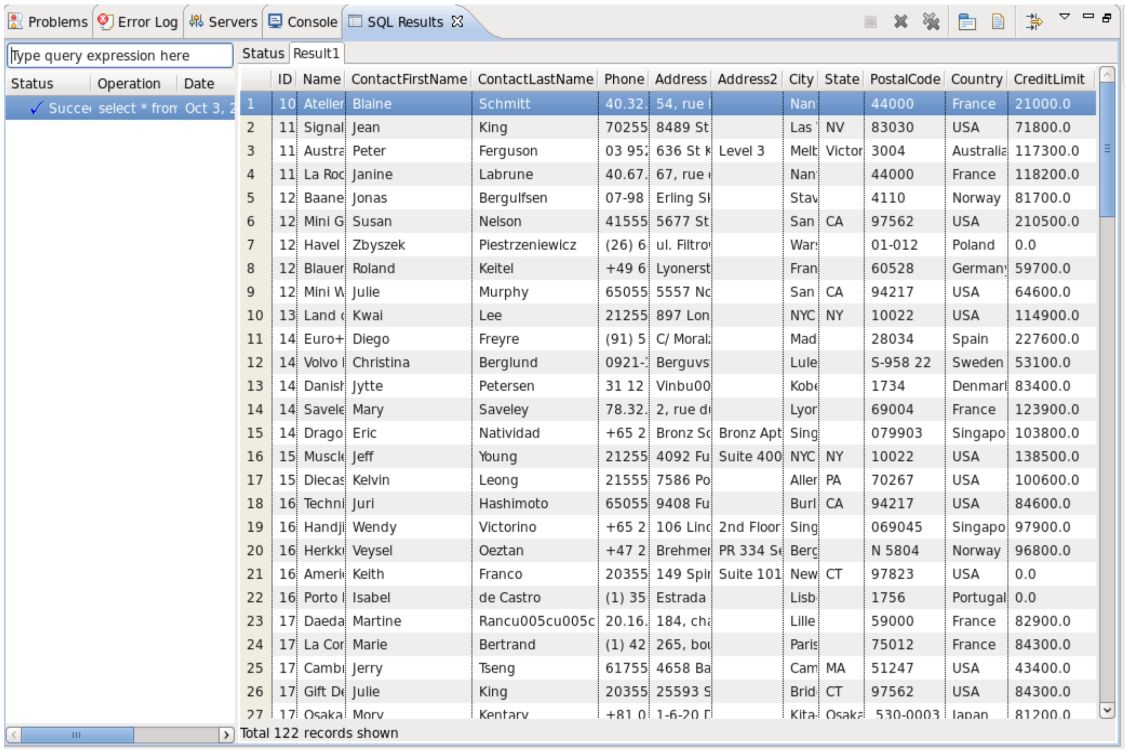
Using the example transformation above, the resulting transformation should look similar to the image below.



```
SELECT
        MyCustomers.ID, MyCustomers.Name, MyCustomers.ContactFirstName, MyCustomers.ContactLastName,
MyCustomers.Phone, MyCustomers.Address, MyCustomers.Address2, MyCustomers.City, MyCustomers.State,
MyCustomers.PostalCode, MyCustomers.Country, MyCustomers.CreditLimit
    FROM
        (EXEC MyRESTDataSource.invoke(binding => 'HTTP', action => 'GET')) AS ws, XMLTABLE('/customers/
customer' PASSING ws.result COLUMNS ID integer PATH './customernumber', Name string PATH './
customername', ContactFirstName string PATH './contactfirstname', ContactLastName string PATH './
contactlastname', Phone string PATH './phone', Address string PATH './addressline1', Address2 string PATH './
addressline2', City string PATH './city', State string PATH './state', PostalCode string PATH './postalcode',
```

At this point, the MyCustomers base table can be clicked in order to enable the “Preview” button,

For illustrative purposes, the image below is the resulting preview of the data.



Status	Operation	Date
✓ Succe	select * from	Oct 3, 2

	ID	Name	ContactFirstName	ContactLastName	Phone	Address	Address2	City	State	PostalCode	Country	CreditLimit
1	10	Atelier	Blaine	Schmitt	40.32.	54, rue		Nan		44000	France	21000.0
2	11	Signal	Jean	King	70255	8489 St		Las	NV	83030	USA	71800.0
3	11	Austra	Peter	Ferguson	03 95	636 St K	Level 3	Melb	Victor	3004	Australia	117300.0
4	11	La Roc	Janine	Labrune	40.67.	67, rue		Nan		44000	France	118200.0
5	12	Baane	Jonas	Bergulfsen	07-98	Erling Sk		Stav		4110	Norway	81700.0
6	12	Mini G	Susan	Nelson	41555	5677 St		San	CA	97562	USA	210500.0
7	12	Havel	Zbyszek	Piestrzeniewicz	(26) 6	ul. Filtro		Wars		01-012	Poland	0.0
8	12	Blauer	Roland	Keitel	+49 6	Lyonerst		Fran		60528	German	59700.0
9	12	Mini W	Julie	Murphy	65055	5557 Nc		San	CA	94217	USA	64600.0
10	13	Land c	Kwai	Lee	21255	897 Lon		NYC	NY	10022	USA	114900.0
11	14	Euro+	Diego	Freyre	(91) 5	C/ Moral		Mad		28034	Spain	227600.0
12	14	Volvo I	Christina	Berglund	0921-1	Berguvs		Lule		S-958 22	Sweden	53100.0
13	14	Danish	Jytte	Petersen	31 12	Vinbu00		Kobe		1734	Denmark	83400.0
14	14	Savele	Mary	Saveley	78.32.	2, rue d		Lyon		69004	France	123900.0
15	14	Drago	Eric	Natividad	+65 2	Bronz Sc	Bronz Apt	Sing		079903	Singapo	103800.0
16	15	Muscle	Jeff	Young	21255	4092 Fu	Suite 400	NYC	NY	10022	USA	138500.0
17	15	Diecas	Kelvin	Leong	21555	7586 Po		Aller	PA	70267	USA	100600.0
18	16	Techni	Juri	Hashimoto	65055	9408 Fu		Burl	CA	94217	USA	84600.0
19	16	Handji	Wendy	Victorino	+65 2	106 Linc	2nd Floor	Sing		069045	Singapo	97900.0
20	16	Herkku	Veysel	Oeztan	+47 2	Brehmer	PR 334 Se	Berg		N 5804	Norway	96800.0
21	16	Ameri	Keith	Franco	20355	149 Spir	Suite 101	New	CT	97823	USA	0.0
22	16	Porto I	Isabel	de Castro	(1) 35	Estrada		Lisb		1756	Portugal	0.0
23	17	Daeda	Martine	Rancu005cu005c	20.16.	184, cha		Lille		59000	France	82900.0
24	17	La Cor	Marie	Bertrand	(1) 42	265, bou		Paris		75012	France	84300.0
25	17	Cambi	Jerry	Tseng	61755	4658 Ba		Cam	MA	51247	USA	43400.0
26	17	Gift De	Julie	King	20355	25593 S		Brid	CT	97562	USA	84300.0
27	17	Osaka	Mory	Kentary	+81 0	1-6-20 D		Kita	Osaka	530-0003	Japan	81200.0

Total 122 records shown

RESOURCES

JBoss Enterprise Data Services Documentation

http://docs.redhat.com/docs/en-US/JBoss_Enterprise_Data_Services/index.html

QUESTIONS/COMMENTS/ISSUES

If you have questions or comments about this whitepaper, please enter them in the Red Hat customer portal for this specific whitepaper: https://access.redhat.com/knowledge/techbriefs . If you have a technical issue following this whitepaper please open a support case: https://access.redhat.com/support/cases/new.