

Dokumentation Teiid Daten Virtualisierung

Inhaltsverzeichnis

- Dokumentation Teiid Daten Virtualisierung
 - Inhaltsverzeichnis
 - 1. Einleitung
 - 1.1 Verwendete Technologien
 - 1.2 Grundlagen
 - 1.3 Kompatibilität
 - 2. Installation
 - 2.1 Verbinden per Putty
 - 2.2 Verbinden per FileZilla
 - 2.3 Installieren von Java und Maven
 - 2.3.1 Java
 - 2.3.2 Maven
 - 2.4 Wildfly mit Teiid
 - 2.4.1 Freigabe von Ports
 - 2.4.2 Erklärung der Ordnerstruktur
 - 2.4.2.1 bin
 - 2.4.2.2 modules
 - 2.4.2.3 standalone
 - 2.5 JBoss Developer Studio mit Teiid Designer
 - 3. Benutzung des Applikationsservers
 - 3.1 Verbinden von JBoss Developer Studio mit dem Wildfly-Server
 - 3.2 User anlegen
 - 3.3 Starten des Servers
 - 3.3.1 Starten über das Developer Studio
 - 3.3.2 Starten über die Shell
 - 3.3.2.1 Im Vordergrund
 - 3.3.2.2 Im Hintergrund
 - 3.3.3 Beenden über die Shell
 - 3.3.3.1 Im Vordergrund
 - 3.3.3.2 Im Hintergrund
 - 3.4 Öffnen der Web Management Console
 - 4. Verbinden mit einer Datenquelle
 - 4.1 Oracle
 - 4.2 MongoDB
 - 4.3 Hive
 - 4.4 Excel
 - 4.5 XML
 - 5. Modellieren von Daten und Erstellung von VDBs

1. Einleitung

In dieser Dokumentation wird die Installation der Teiid Daten Virtualisierung beschrieben, sowie das Aufsetzen des Servers und das Verbinden mit diversen Datenquellen. Zudem werden generelle Grundlagen der Verwendung des Wildfly Servers und Verwendung der JBoss Developer Studio Oberfläche erklärt. Der Ordner mit benötigten Dateien ist [hier](#) zu finden.

Die folgende Grafik zeigt einen Überblick über das gesamte System, das wir im Laufe dieser Dokumentation aufbauen und deren Schnittstellen.

!Schritt0](./Bilder/Schritt 0.png)

1.1 Verwendete Technologien

- Rechner 1(Server)
 - OS: Red Hat Enterprise Linux
 - Applikationsserver: Wildfly (Version 11)
 - Datenvirtualisierung: Teiid (Version 11.0.1)
 - Java 1.8
- Rechner 2(User)
 - Red Hat JBoss Developer Studio mit Teiid Designer (Version 11.1.2)
 - Putty
 - FileZilla

1.2 Grundlagen

Dieses Kapitel beschäftigt sich mit der verwendeten Begriffsterminologie und dem groben Aufbau der Infrastruktur. Für die Datenvirtualisierung wird ein zweiteiliges System verwendet. Auf der einen Seite befindet sich ein Server (VM oder ein physischer Rechner) und auf der anderen Seite der User mit seiner lokalen Maschine. In diesem Beispiel läuft eine VM auf einem anderen Rechner und wir werden mit Putty auf den Server zugreifen. Auf dem Server läuft das Betriebssystem Red Hat Enterprise Linux. Hierfür wird ein Account bei Red Hat benötigt. Auf dieser Maschine wird ein Applikationsserver für die Datenvirtualisierung installiert. Hier gibt es zwei Möglichkeiten:

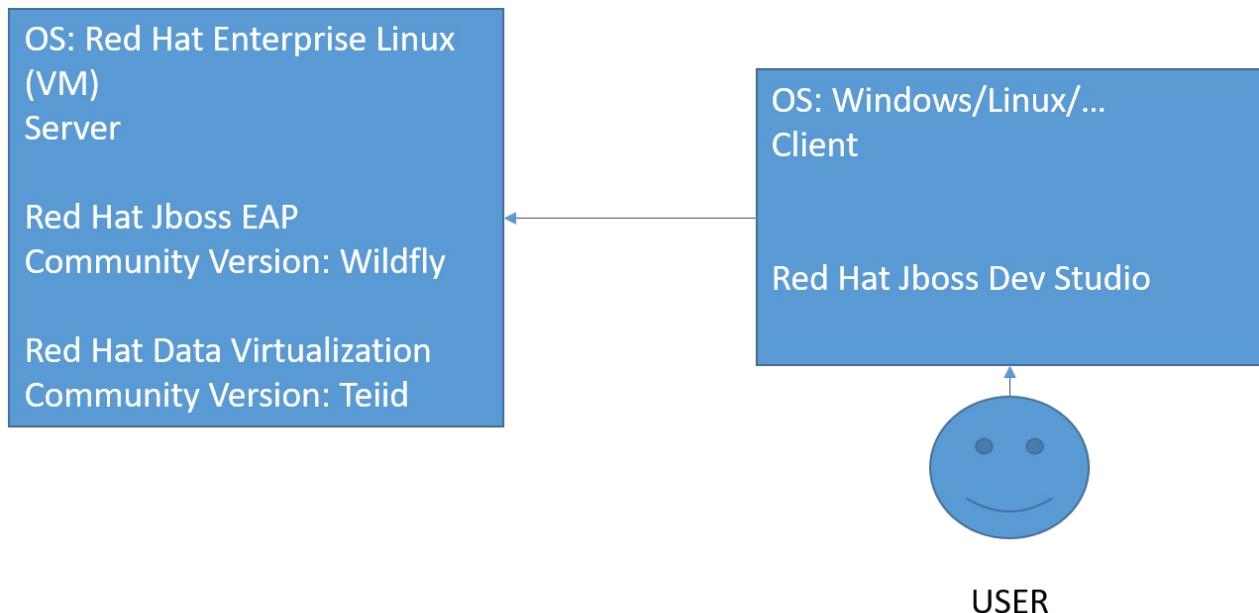
- JBoss Enterprise Application Platform (JBoss EAP)
- Wildfly

Bei EAP handelt es sich um die offizielle Version von Red Hat und bei Wildfly um die Community Version. EAP basiert auf den Komponenten von Wildfly. Zusätzlich muss eine Datenvirtualisierung auf diesem Applikationsserver installiert werden. Auch hierbei gibt es wieder zwei Möglichkeiten:

- JBoss Data Virtualization
- Teiid

JBoss Data Virtualization die angebotene Version von Red Hat und Teiid die Community Version. JBoss Data Virtualization wird von Red Hat in Zukunft nicht weiter entwickelt und der Support wird eingestellt (Stand: 07.08.2018).

Aus diesem Grund wird im Laufe der Dokumentation ein Wildfly Server mit installiertem Teiid verwendet.



Um eine einfachere Konfiguration der Datenvirtualisierung zu erhalten, wird der [Teiid Designer](#) benötigt. Diese Installation ist optional wird jedoch empfohlen. Wir verwenden den Teiid Designer als Plugin für das [Red Hat Developer Studio](#).

1.3 Kompatibilität

Ein bestehendes Problem sind die vielen zu Verfügung stehenden Versionen von Teiid und Wildfly, sowie des Teiid Designers. Die neueste Version von Teiid ist 11.0.1 (Stand 07.08.2018) und ist mit Wildfly 11 kompatibel. Die neueste Version von Wildfly ist Wildfly 13, jedoch gibt es hierfür keine Teiid Version. Zudem ist der Teiid Designer 11.1.2 nur vollständig kompatibel und getestet mit Teiid Version 9, was ebenfalls Wildfly 9 benötigt.

Wildfly	Teiid	Teiid Designer	Java
13	-	-	1.8
11	11.0.1	-	1.8
9	9.3.7	11.1.2	1.7

Also ergeben sich zwei Möglichkeiten. Entweder es wird Wildfly 9 mit Teiid 9.3.7 und dem Teiid Designer 11.1.2 verwendet. Dies ist eine etwas veraltete Variante, jedoch funktioniert der Teiid Designer vollständig. Oder es wird Wildfly 11 mit Teiid 11.0.1 und ebenfalls dem Teiid Designer 11.1.2, jedoch können hier Probleme mit der Oberfläche im Teiid Designer entstehen und es muss für bestimmte Tasks ggf. auf die Command-Line gewechselt werden. Die Java Version ist bedingt durch die verwendeten Komponenten und wird bei der späteren Installation eine Rolle spielen.

Im Folgenden wird Wildfly 11 mit Teiid 11.0.1 und der Teiid Designer 11.1.2 verwendet

2. Installation

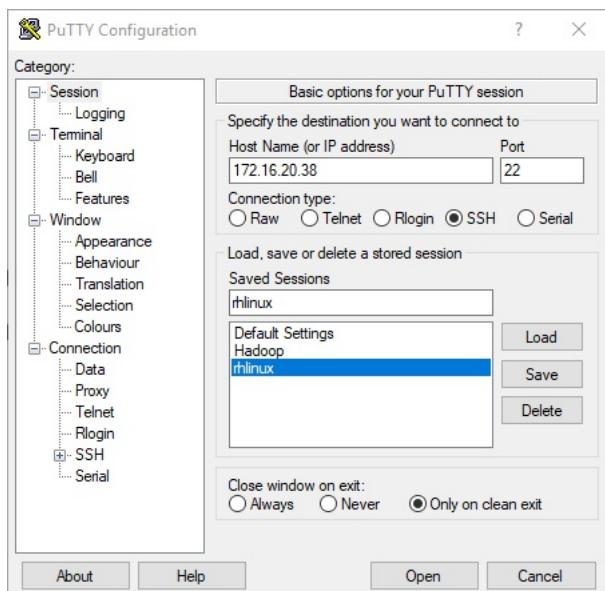
Dieses Kapitel beschäftigt sich mit der Installation der genannten Komponenten sowie deren Konfiguration zur weiteren Benutzung. Es wird davon ausgegangen, dass bereits ein Server mit dem Betriebssystem [Red Hat Enterprise Linux](#) vorliegt. Diese Maschine sollte per SSH erreicht werden.

Im Kontext des Gesamtsystems setzen wir hier den rot markierten Bereich auf und die Verbindung von unserer lokalen Maschine dazu.

!{Schritt1}./Bilder/Schritt 1.png

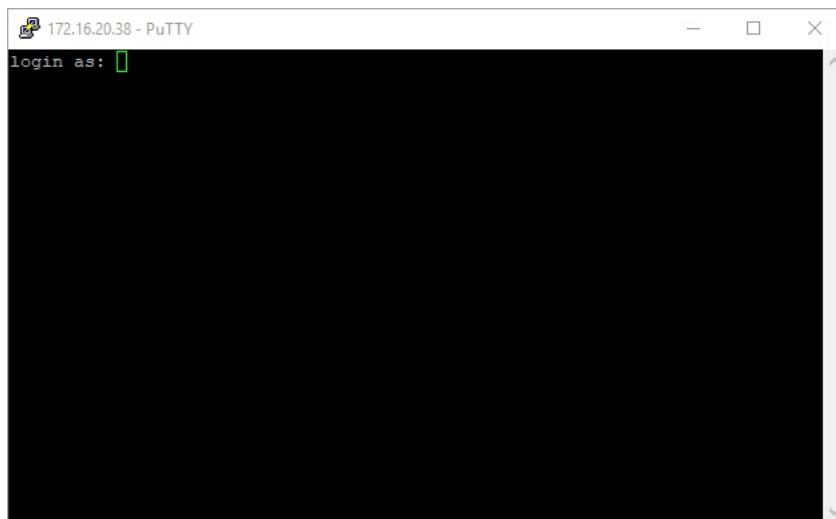
2.1 Verbinden per Putty

Um auf das Filesystem und die Shell des Servers zuzugreifen benötigen wir [Putty](#). Installieren Sie dies auf Ihrer lokalen Maschine und starten Sie Putty.



Stellen Sie den **Connection type** auf **SSH** um und tragen Sie in das Feld **Host Name** die IP-Adresse des Servers ein sowie den **Port** (Standardmäßig Port: 22). Klicken Sie anschließend auf **Load**.

Nun öffnet sich die Shell des Servers.

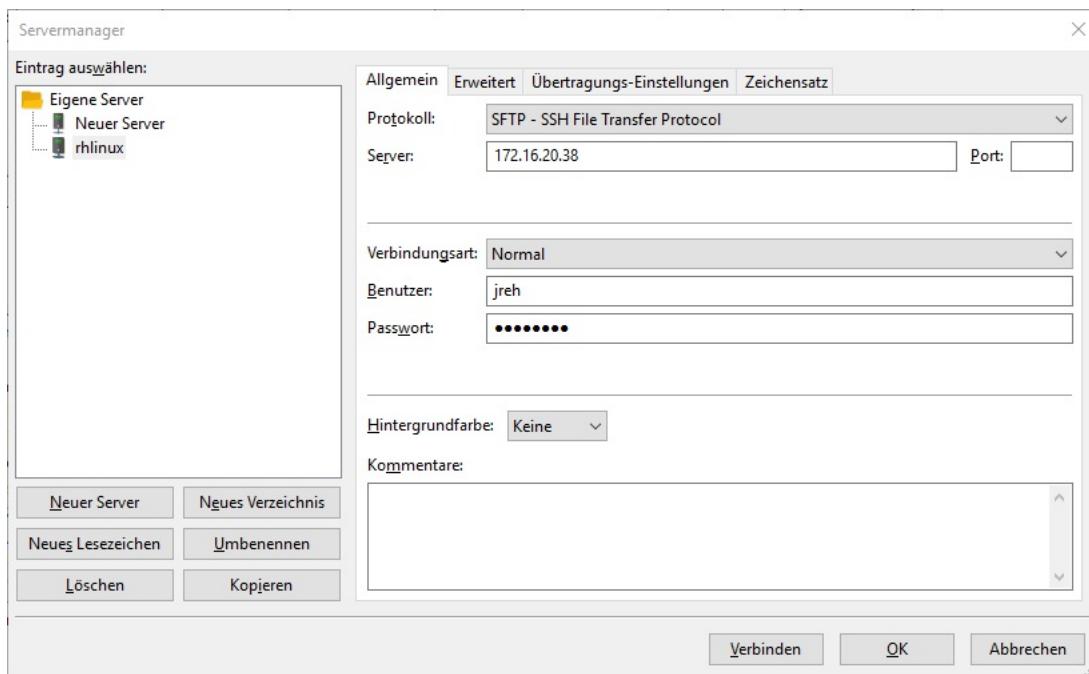


Melden Sie sich mit den Benutzerdaten an. Hier ggfs. den Systemadministrator fragen.

2.2 Verbinden per FileZilla

FileZilla wird für den Anfang benötigt, um den Ordner mit dem Applikationsserver auf den Server zu verschieben. Hierfür installieren Sie [FileZilla](#) und starten Sie die Anwendung.

Öffnen Sie über Datei -> Servermanager die Serververwaltung legen Sie einen neuen Server an.



Stellen Sie das **Protokoll** auf SFTP um und tragen Sie die IP-Adresse in das Feld **Server**. Ebenfalls tragen Sie ihren Benutzer und das Passwort in die vorgesehenen Felder ein und klicken Sie auf verbinden. Nun haben Sie Zugriff auf das Filesystem des Servers.

2.3 Installieren von Java und Maven

Java und Maven werden als Installation auf dem Server benötigt, damit Wildfly und Teiid funktionieren.

2.3.1 Java

Dafür verbinden Sie sich per Putty mit dem Server und melden Sie sich an. Sie sind im Home-Verzeichnis des Users, navigieren Sie in das Root-Verzeichnis des Servers mit:

```
$ cd ..
```

Nun installieren Sie über den bereits vorhandenen Package-Manager yum die benötigte Java Version:

Für Java 1.8

```
$ yum install java-1.8.0-openjdk-devel
```

oder für Java 1.7

```
$ yum install java-1.7.0-openjdk-devel
```

Um die Installation zu bestätigen, führen Sie den Befehl

```
$ /usr/sbin/alternatives --config java
```

```
[jreh@srv-im-rhlinux ~]$ /usr/sbin/alternatives --config java
Es gibt 2 Programme, welche »java« zur Verfügung stellen.

Auswahl      Befehl
-----
1           java-1.7.0-openjdk.x86_64 (/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.181-2.6.14.8.el7_5.x86_64/jre/bin/java)
*+ 2         java-1.8.0-openjdk.x86_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.181-3.b13.el7_5.x86_64/jre/bin/java)

Eingabe um die vorgegebene Auswahl[+] zu behalten, oder geben Sie die Nummer an:2
```

und besätnigen Sie ihre Auswahl. Liegt nur eine Java Installation vor, enthält die List nur einen Eintrag.
Denselben Vorgang wiederholen Sie für den Befehl

```
$ /usr/sbin/alternatives --config javac
```

2.3.2 Maven

Um **Maven** zu installieren, downloaden Sie **Maven** auf ihren lokalen Rechner. Wählen Sie dabei den im Bild gezeigten Download-Link unter **Binary Zip archive**.

Files

Maven is distributed in several formats for your convenience. Simply pick a rea
In order to guard against corrupted downloads/installations, it is highly recomme

Link

Binary tar.gz archive	apache-maven-3.5.4-bin.tar.gz
Binary zip archive	apache-maven-3.5.4-bin.zip
Source tar.gz archive	apache-maven-3.5.4-src.tar.gz
Source zip archive	apache-maven-3.5.4-src.zip

und transferieren Sie die Zip-Datei per FileZilla in das Home-Verzeichnis ihres Users (/home/user/apache-maven-3.5.4-bin). Navigieren Sie über Putty in diesem Ordner, entpacken Sie die Zip-Datei und löschen Sie diese anschließend.

```
$ cd home/user/
$ unzip apache-maven-3.5.4-bin.zip
$ rm apache-maven-3.5.4-bin.zip
```

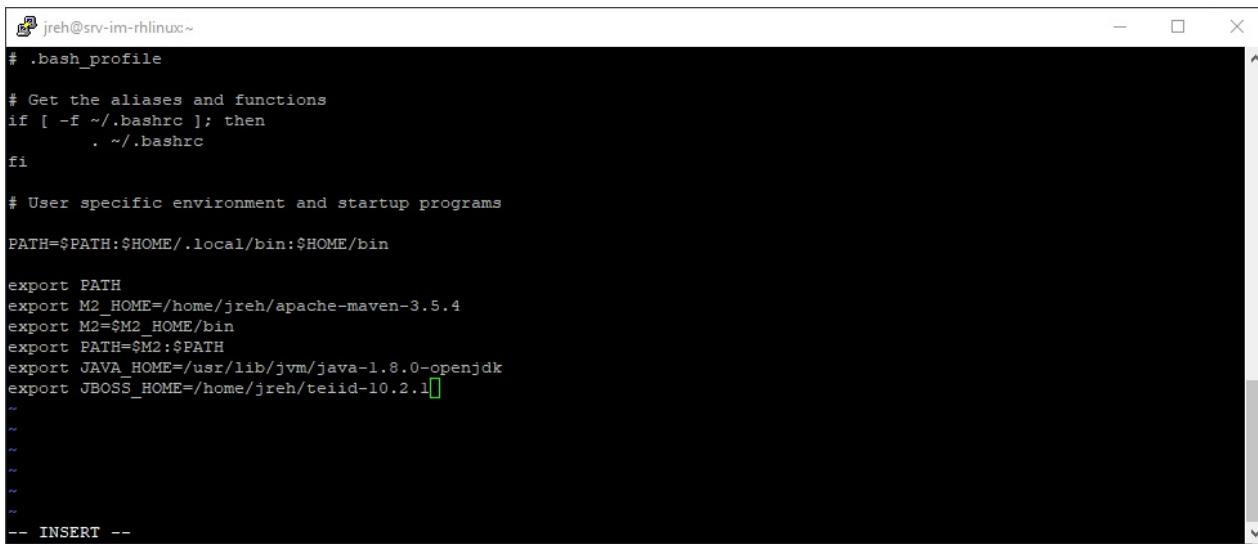
Nun öffnen Sie Ihr Bash_Profile mit dem Texteditor vim

```
$ vi ~/.bash_profile
```

und tragen Sie die Variablen

```
export M2_HOME=/home/user/apache-maven-3.5.4
export M2=$M2_HOME/bin
export PATH=$M2:$PATH
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk
```

ein. Beispielhaft:



```
jreh@srv-im-rhlinux: ~
# .bash_profile
#
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/.local/bin:$HOME/bin

export PATH
export M2_HOME=/home/jreh/apache-maven-3.5.4
export M2=$M2_HOME/bin
export PATH=$M2:$PATH
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk
export JBOSS_HOME=/home/jreh/teiid-10.2.1
~

-- INSERT --
```

Beenden Sie den Editor, indem Sie **ESC** drücken und **:wq!** eingeben und mit **Enter** bestätigen. Damit die Änderung in Kraft tritt, führen Sie folgenden Befehl aus:

```
$ source ~/.bash_profile
```

Um die Installation zu überprüfen, geben Sie

```
$ mvn --version
```

ein.

2.4 Wildfly mit Teiid

[Externer Guide](#)

[Externer Guide](#)

Hier ([interner Link](#)) finden Sie eine Wildfly 11 Version mit bereits installiertem Teiid 11.0.1. Downloaden Sie dafür die Zip-Datei unter dem Link (with Wildfly/Console)

Teiid Wildfly Downloads

Runtime

Latest Release Announcement: 11.0.0 Released on July 2nd, 2018

Teiid 11.0.0 for Wildfly 11.0.0 is now available. The main features are:

- [TEIID-4520](#) Added the Exasol translator.
- [TEIID-5356](#) Improved column metadata with DDL views - and other metadata fixes such as TEIID-5361 and TEIID-5359
- [TEIID-3439](#) Added support for table aliases to contain a period "." character
- Various SQL compatibility issues to support JOQL
- [TEIIDTOOLS-437](#) OpenTracing integration for remote JDBC clients. Further automatic integration will be provided via Thorntail.

Latest

- **Version 11.0.1**
 - [Teiid Runtime](#) (with Wildfly/Console)
 - [Teiid AdminShell](#)
 - [JDBC Driver](#)
 - [Teiid Source](#)
 - [Release Notes/Change Log](#)
 - [Teiid Web Console](#)

Verschieben Sie diese Datei ebenfalls mit FileZilla in den Home-Ordner ihres Users (/home/user/teiid-11.0.1-wildfly-server.zip) und entpacken bzw. löschen Sie anschließend den Zip-Ordner.

```
$ unzip teiid-11.0.1-wildfly-server.zip
$ rm teiid-11.0.1-wildfly-server.zip
```

Fügen Sie in ihr Bash_Profile die Zeile

```
export JBOSS_HOME=/home/user/teiid-11.0.1
```

ein. Benutzen Sie eine andere Version, benennen Sie den Pfad dementsprechend. Geben Sie wieder

```
$ source ~/.bash_profile
```

ein, um die Änderung aktiv werden zu lassen.

Wildfly mit Teiid ist nur erfolgreich auf Ihrem System.

2.4.1 Freigabe von Ports

Wildfly ermöglicht die Administration des Servers über eine Weboberfläche. Damit alle Funktionen von Wildfly verwendet werden können, müssen auf dem Server bestimmte Ports freigegeben werden. Führen Sie dafür die Befehle:

```
$ sudo firewall-cmd --add-port 8787/tcp  
$ sudo firewall-cmd --add-port 9990/tcp  
$ sudo firewall-cmd --add-port 31000/tcp
```

aus. Um die offenen Ports aufzulisten:

```
$ sudo firewall-cmd --list-ports
```

8787

Wird für Debugzwecke verwendet.

9990

Port für Managementservices. Ist dieser Port nicht offen, kann das Developer Studio einen gestarteten Server nicht erkennen und die Weboberfläche ist nicht verfügbar.

31000

Dies ist ein Port, der die Schnittstelle zu Teiid bildet. Ist dieser nicht offen, wird keine laufende Instanz von Teiid erkannt.

2.4.2 Erklärung der Ordnerstruktur

In diesem Abschnitt gebe ich Ihnen einen kleinen Überblick über die vorhandene Ordnerstruktur und benenne die wesentlichen Dateien, die für die weitere Konfiguration des Wildfly-Servers von Bedeutung sein werden.

Die relevanten Ordner sind **bin**, **modules** und **standalone**

```
$ teiid-11.0.1
```

```
├── bin  
│   ├── jboss-cli.sh  
│   ├── add-user.sh  
│   ├── standalone.sh  
│   ├── ...  
├── modules  
│   └── system  
│       ├── layers  
│       └── ...  
└── standalone  
    ├── configuration  
    │   ├── standalone-teiid.xml  
    │   ├── application-users.properties  
    │   ├── application-roles.properties  
    │   ├── mgmt-users.properties  
    │   ├── mgmt-groups.properties  
    │   └── ...  
    ├── ...  
    └── ...
```

Alle hier aufgelisteten Dateien und Ordner sind von Relevanz und werden im folgenden näher erklärt.

2.4.2.1 bin

Im **bin** Ordner befinden sich die ausführbaren Dateien. Die **jboss-cli.sh** ist das Skript, mit welchem das CommandLineInterface des Wildfly-Servers aufgerufen werden kann. Genau Benutzung wird später erklärt. **add-user.sh** startet das Skript zum Hinzufügen eines Users und **standalone.sh** startet den Server.

2.4.2.2 modules

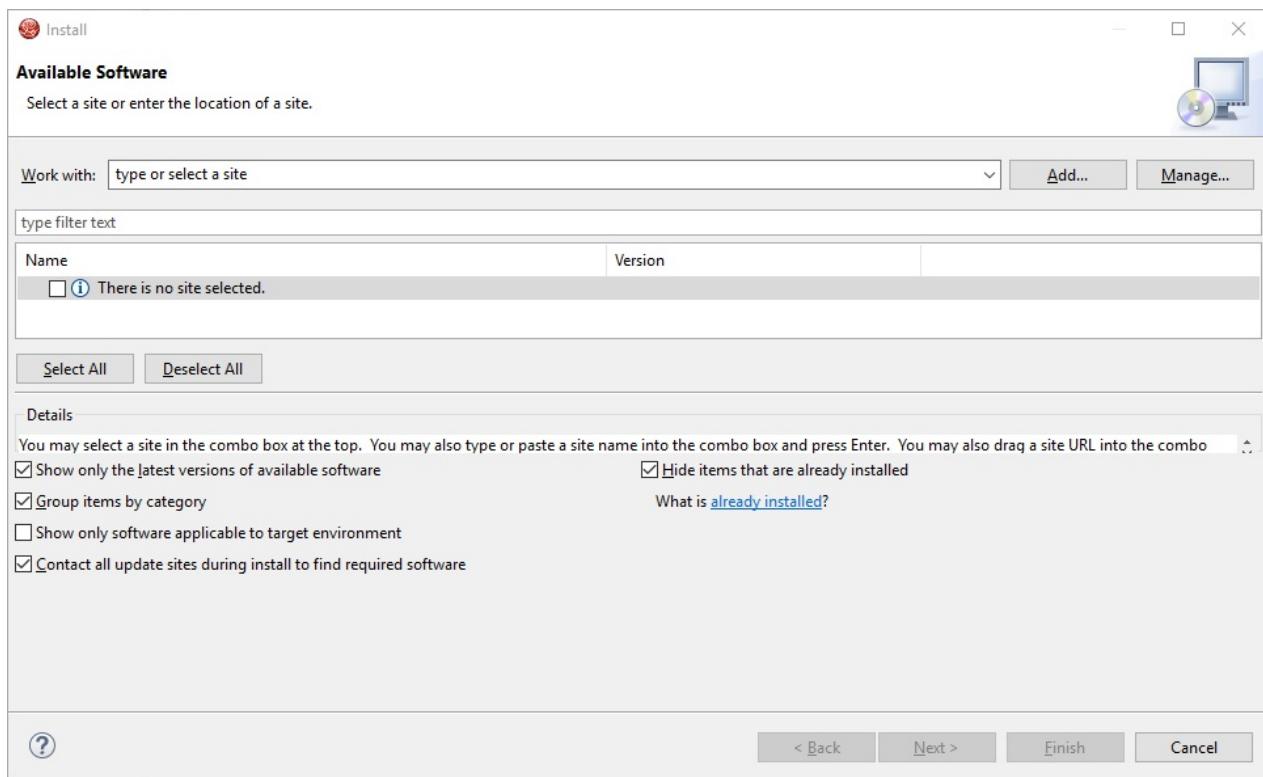
Im **modules**-Ordner befinden sich alle möglichen Module des Servers. Darunter befinden sich ebenfalls die Treiber für Datenbankschnittstellen, wofür wir diesen Ordner im späteren Verlauf verwenden werden.

2.4.2.3 standalone

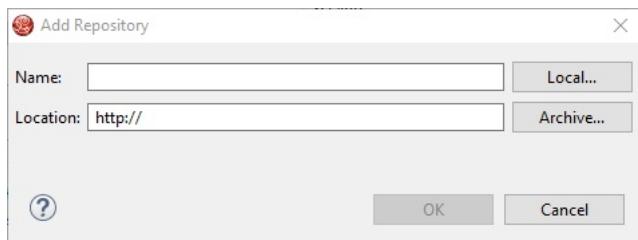
Der **standalone**-Ordner beinhaltet die Configurationsdatei des Servers. Im Unterordner **configuration** befindet sich die Datei **standalone-teiid.xml**, aus beim Start des Servers die Einstellungen genommen werden. Die ***.properties**-Dateien besitzen Angaben zu den Users, sowie deren Rollen.

2.5 JBoss Developer Studio mit Teiid Designer

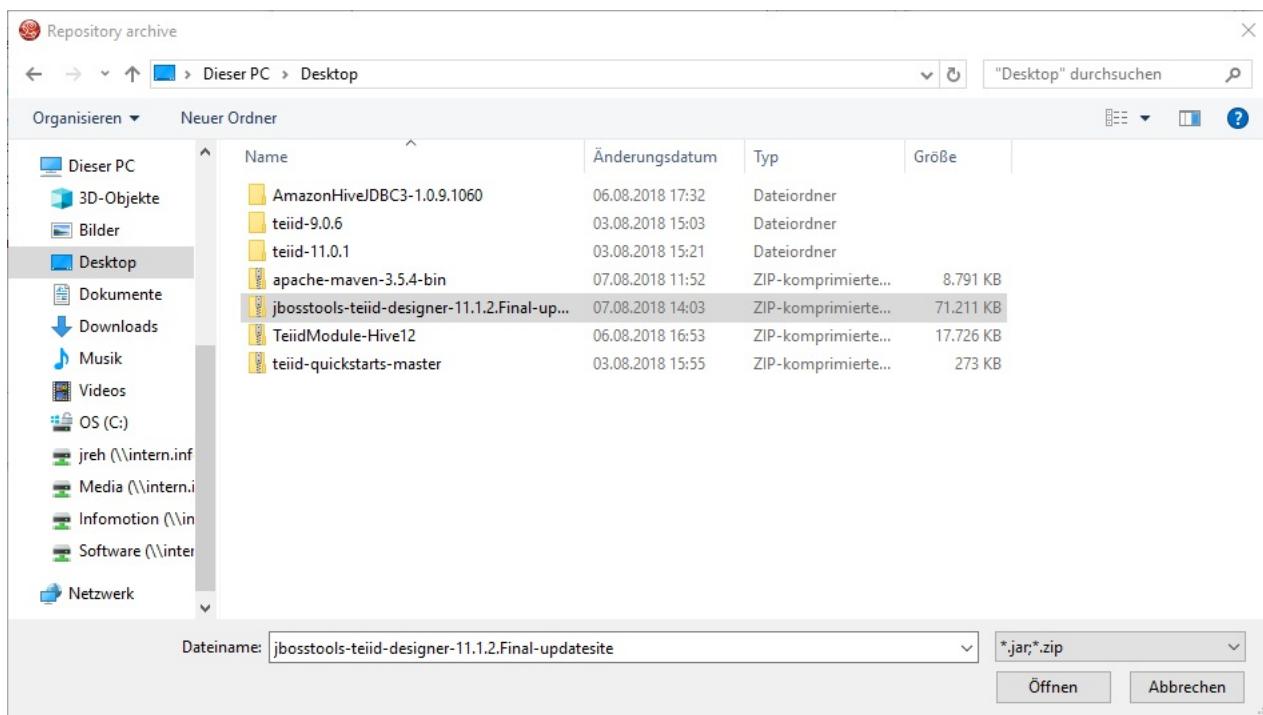
Installieren Sie das [JBoss Developer Studio \(interner Link\)](#) auf ihrer lokalen Maschine und downloaden Sie den [teiid-Designer \(interner Link\)](#). Hierbei handelt es sich um ein Plugin für das JBoss Developer Studio. Öffnen Sie das installierte JBoss Developer Studio und klicken Sie auf den Reiter **Help** und klicken Sie auf **Install new Software...**



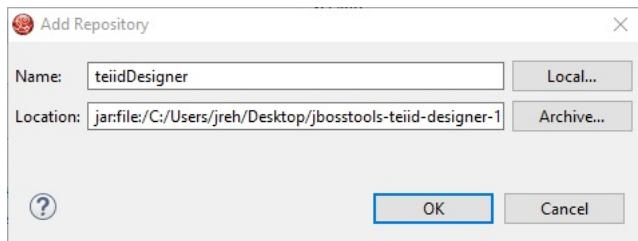
In diesem Fenster klicken Sie rechts auf **Add...**



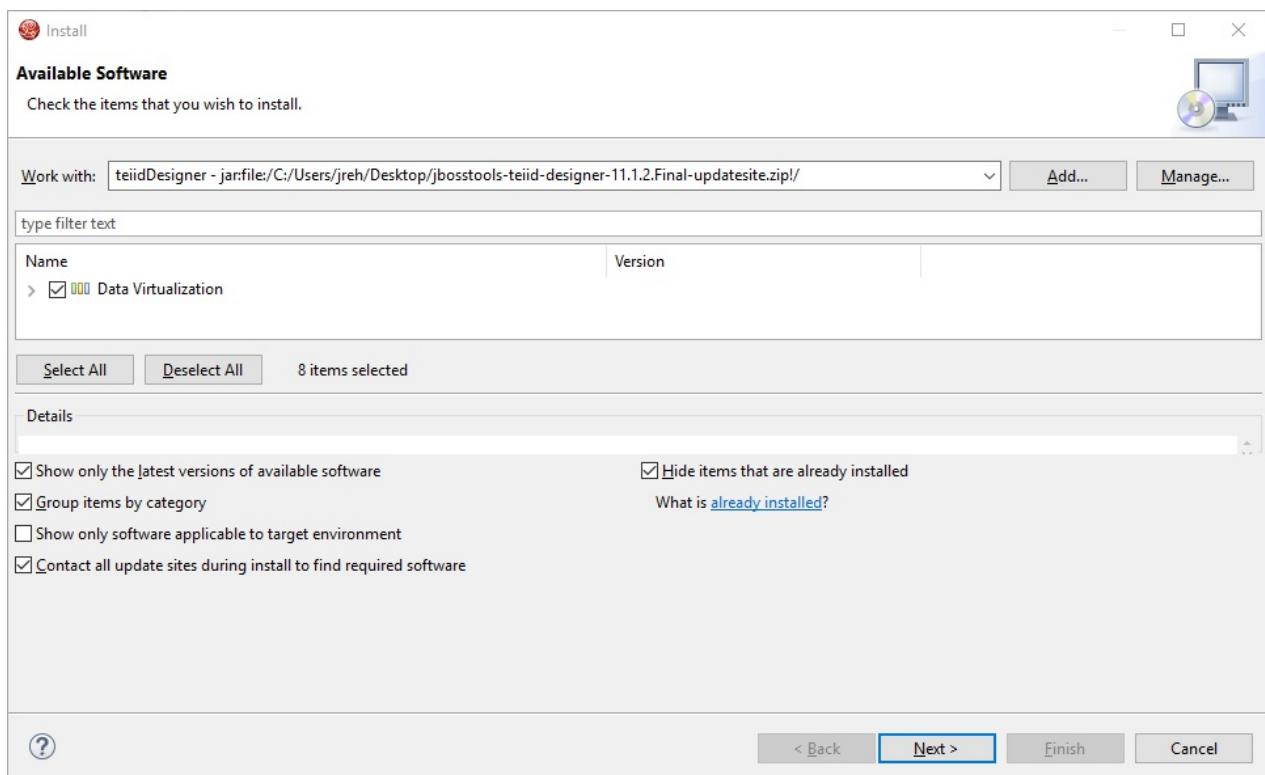
im geöffneten Fenster klicken Sie auf **Archive** und wählen die .zip-Datei des Teiid-Designers, die im vorherigen Schritt heruntergeladen wurde.



Klicken Sie auf **Öffnen** und vergeben Sie einen beliebigen Namen



und bestätigen Sie mit **OK**.



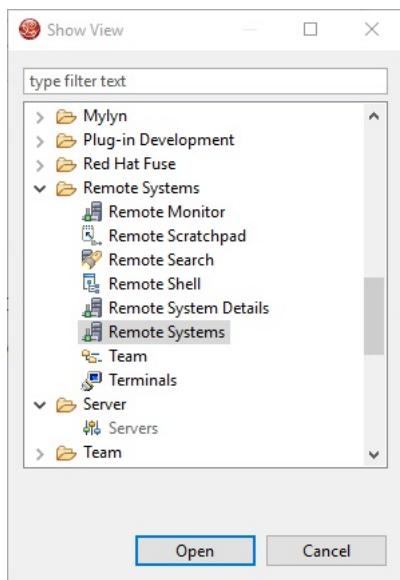
Wählen Sie nun **Data Virtualization** aus und klicken Sie auf **Next**. Die benötigten Komponenten werden nun installiert. Das Developerstudio ist nun erfolgreich installiert.

3. Benutzung des Applikationsservers

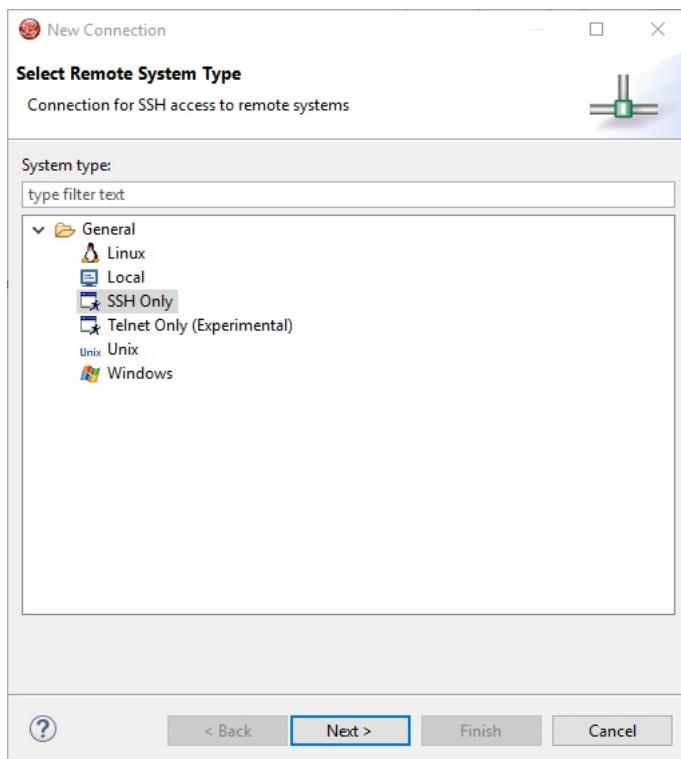
3.1 Verbinden von JBoss Developer Studio mit dem Wildfly-Server

Externer Guide

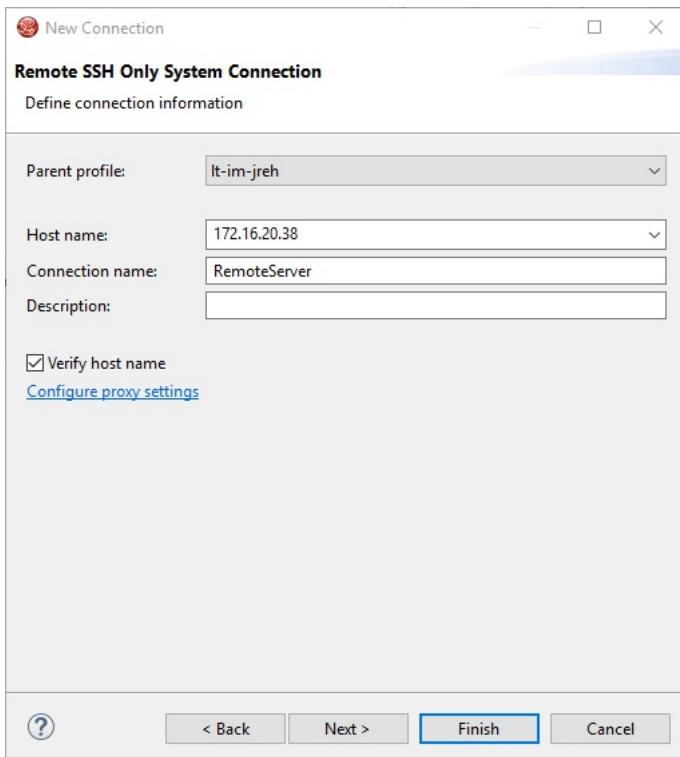
Wir haben nun den Wildfly-Server auf unserem Server installiert und haben das Developer Studio lokal auf unserem Rechner. Nun verbinden wir das Developer Studio mit unserem Server. Öffnen Sie dafür die **Remote System**-Ansicht. Diese ist in der Menüleiste unter Window -> Show View -> Other zu finden.



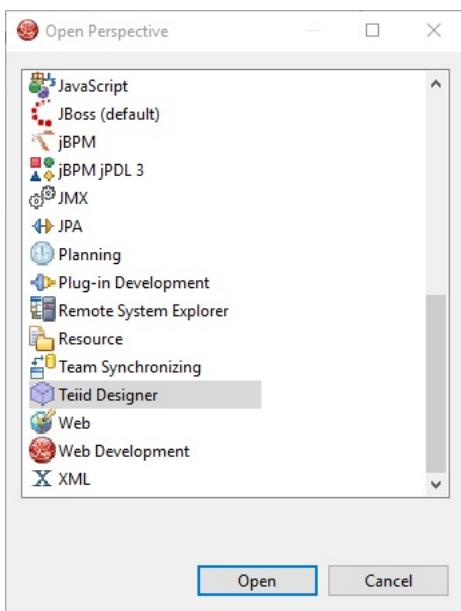
Dort öffnen Sie Remote Systems -> Remote Systems. Legen Sie dort eine neue Verbindung an und wählen Sie **SSH Only**.



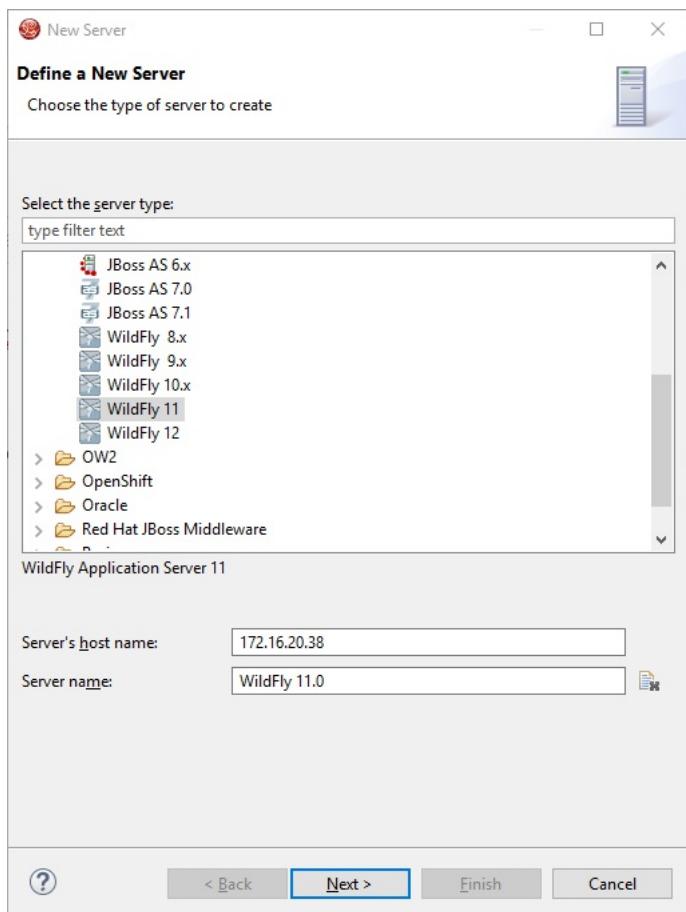
Tragen Sie dort unter **Host name** die IP-Adresse des Servers ein und vergeben Sie einen beliebigen **Connection name**.



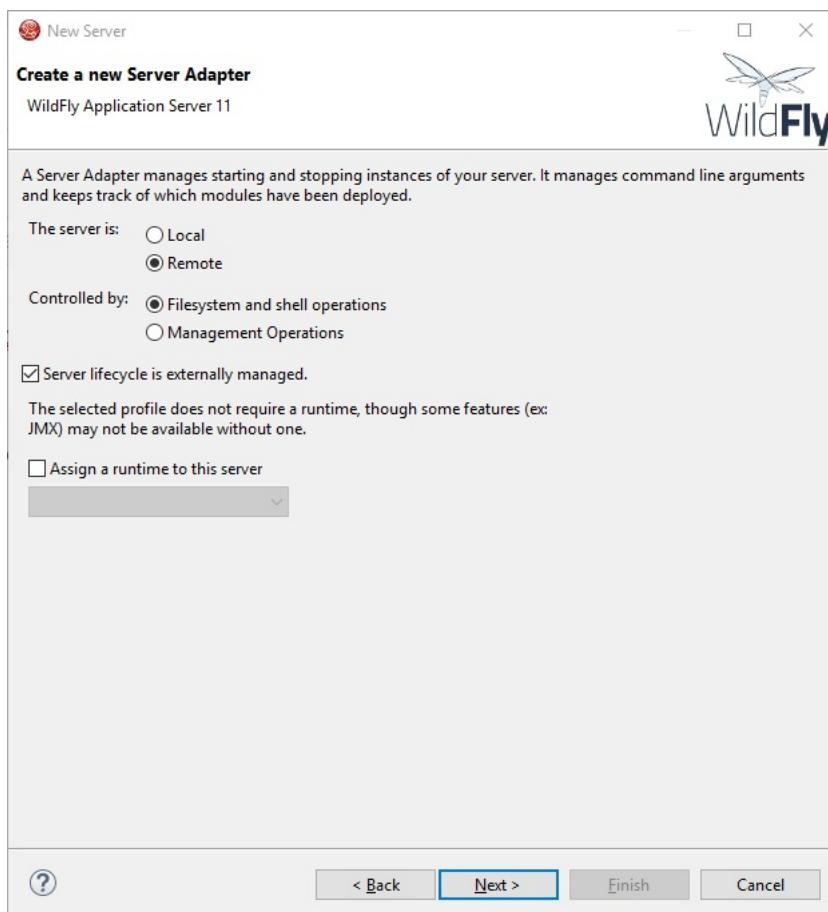
Nun haben Sie Zugriff auf das Filesystem Ihres Servers über das Developer Studio. Ab diesem Zeitpunkt wird FileZilla nicht mehr benötigt, da alle Filetransfers nun über das Developer Studio stattfinden können. Öffnen Sie nun unter Window -> Perspective -> Open Perspective -> Other... den Teiid Designer.



Am rechten Rand gehen Sie nun auf den Reiter **Servers**. Legen Sie per Rechtsklick -> new -> Server einen neuen Server an. Wählen Sie in der Kategorie **JBoss Community** Ihre Serverversion, in unserem Fall Wildfly 11.

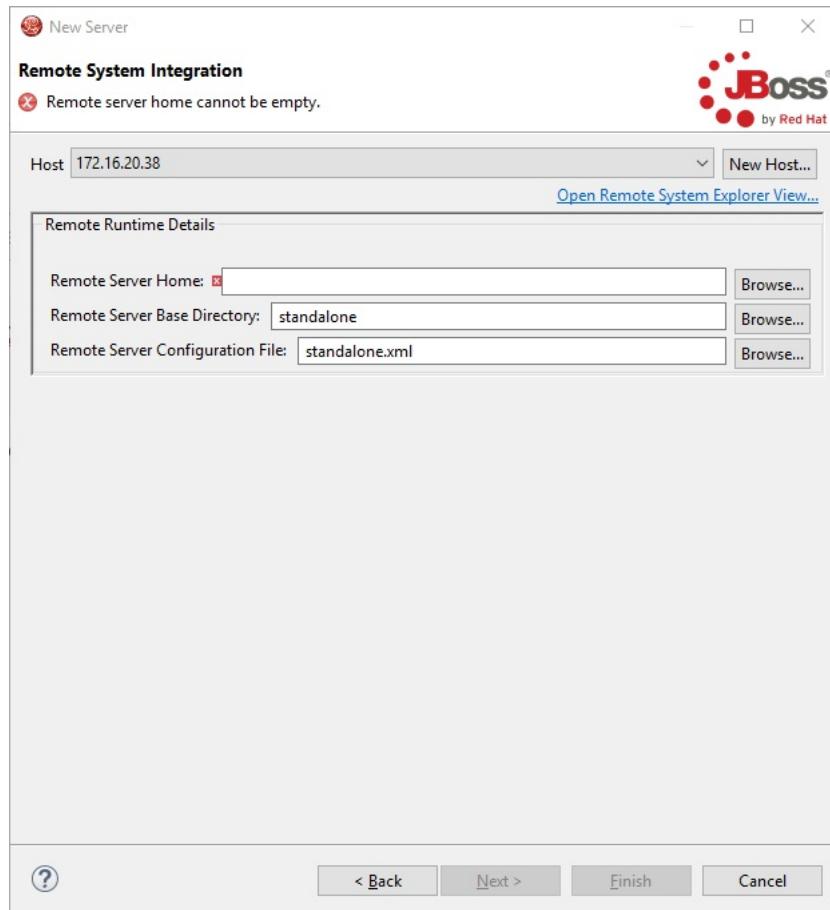


Tragen Sie im Feld **Server's host name** ihren eben erstellen Host ein und vergeben Sie einen beliebigen **Server name**.

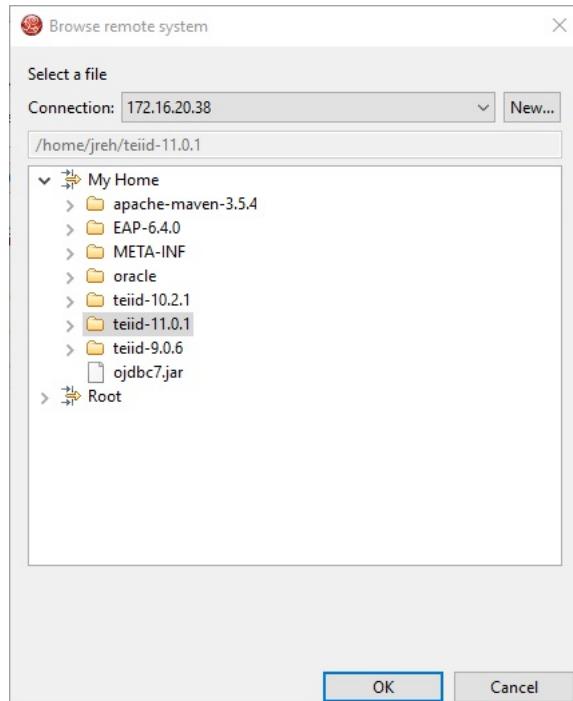


Nun stellen Sie bei **The server is Remote** ein und **Controlled by** auf Filesystem and shell operations. Wählen Sie **Assign a runtime to this server** ab. Ob ein Haken bei **Server lifecycle is externally managed** hängt davon ab, ob Sie den Applikationsserver über das Developer Studio starten wollen oder über die Shell auf dem Server direkt. Es wird empfohlen den Haken nicht zu setzen und die letztere Variante zu wählen. Dazu später mehr.

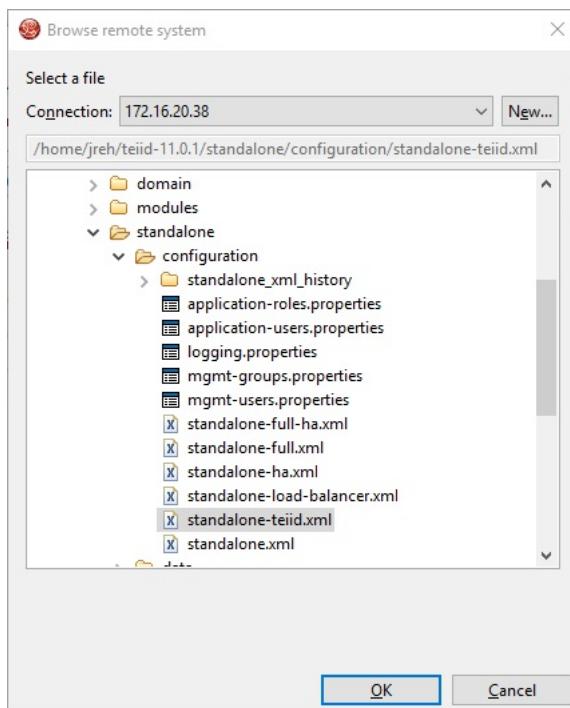
Im nächsten Fenster wählen Sie ihren Host aus und klicken auf **Browse...** neben **Remote Server Home**.



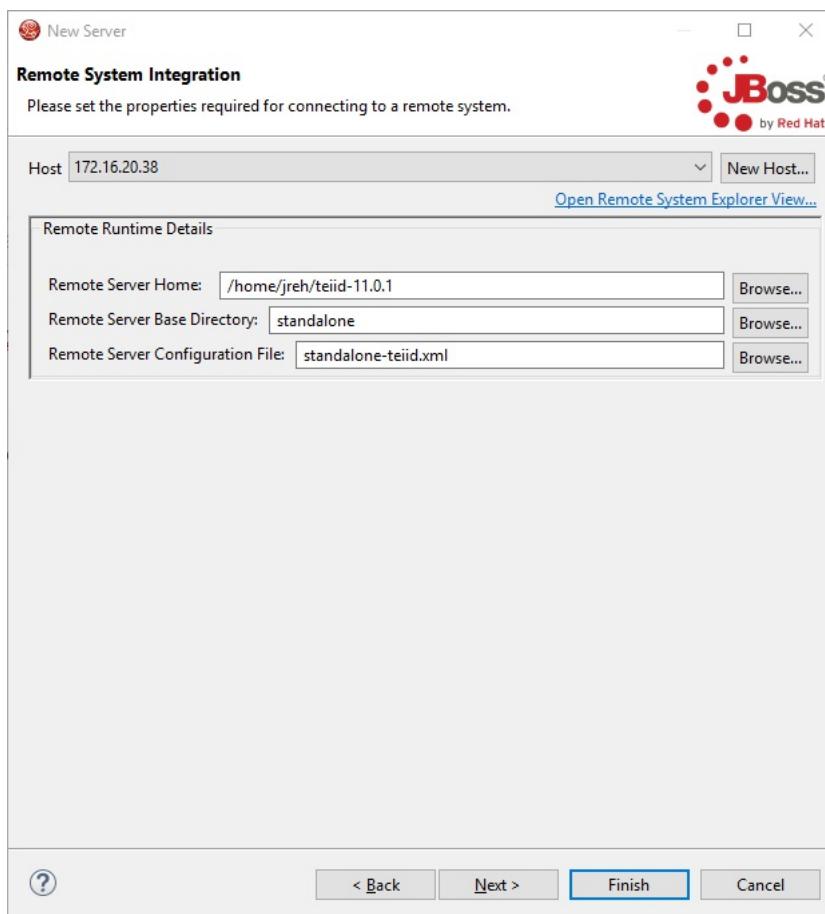
Wählen Sie dort den Home-Server ihres gewählten Applikationsservers aus und bestätigen Sie.



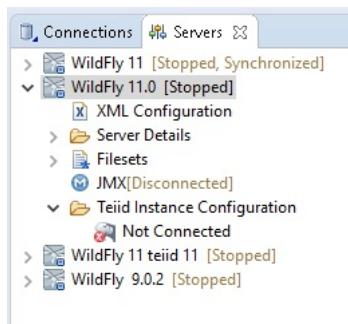
Nun wählen Sie **Browse...** hinter **Remote Server Configuration File**. Navigieren Sie dort in standalone -> configuration und wählen Sie die **standalone-teiid.xml**.



Und beenden Sie mit **Finish**.



Nun können Sie im Reiter **Servers** ihren konfigurierten Server finden.



Doppelklicken Sie ihren Server, um das Fenster für die Servereinstellungen zu öffnen.

Um sich nun mit dem Wildfly-Server verbinden zu können, muss ein Managementuser erstellt werden, dessen Credentials in **User Name** und **Password** eingetragen werden.

3.2 User anlegen

Externer Guide

Es gibt zw ei Arten von Usern:

- Managementuser
- Applikationsuser

Hierfür wird das Skript **add-user.sh** verwendet, das früher bereits angesprochen wurde. Managementuser werden für alles verwendet, was mit der Serverkonfiguration zu tun hat und Applikationsuser für alles was mit Teiid und Datenmodellierung zusammenhängt. Wir werden im folgenden für beide Klassen einen User anlegen. Navigieren Sie dafür über die Shell in Putty zum Ordner der die **add-user.sh** beinhaltet (`$TEIID_HOME/bin`) und starten Sie das Skript:

```
$ sh add-user.sh
```

```
jreh@srv-im-rhelinux:~/teiid-11.0.1/bin

What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): a

Enter the details of the new user to add.
Using realm 'ManagementRealm' as discovered from the existing property files.
Username : admin
User 'admin' already exists and is enabled, would you like to...
 a) Update the existing user password and roles
 b) Disable the existing user
 c) Type a new username
(a): a
Password recommendations are listed below. To modify these restrictions edit the add-user.properties configuration file.
- The password should be different from the username
- The password should not be one of the following restricted values {root, admin, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s), 1 digit(s), 1 non-alphanumeric symbol(s)
Password :
Re-enter Password :
What groups do you want this user to belong to? (Please enter a comma separated list, or leave blank for none)[ ]:
Updated user 'admin' to file '/home/jreh/teiid-10.2.1/standalone/configuration/mgmt-users.properties'
Updated user 'admin' to file '/home/jreh/teiid-10.2.1/domain/configuration/mgmt-users.properties'
Updated user 'admin' with groups to file '/home/jreh/teiid-10.2.1/standalone/configuration/mgmt-groups.properties'
Updated user 'admin' with groups to file '/home/jreh/teiid-10.2.1/domain/configuration/mgmt-groups.properties'
Is this new user going to be used for one AS process to connect to another AS process?
e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.
yes/no? no
[jreh@srv-im-rhelinux bin]$
```

Wählen Sie dort mit **a** einen Management User und geben Sie einen beliebigen Username und Passwort ein. Überspringen Sie mit Enter die Vergabe einer Gruppe und geben Sie anschließend **no** ein. Damit ist ein Managementuser erfolgreich angelegt. Zum Anlegen eines Applikationsusers wählen Sie im ersten Schritt **b** und folgen Sie den weiteren Schritten. Wir werden im weiteren einen Applikationsuser mit dem Namen **teiidUser** verwenden.

Tragen Sie nun die die Logindetails des Managementusers im Developer Studio ein. Damit kann mit einer laufenden Instanz des Applicationsservers verbunden werden. Wechseln Sie am unteren Rand auf **Teiid Instance**.

The screenshot shows the Red Hat Central interface with the following tabs at the top: Red Hat Central, WildFly 11 teiid 11, WildFly 9.0.2, and *WildFly 11.0. The *WildFly 11.0 tab is active.

Teiid Instance

Overview

- JBoss Server: WildFly 11.0
- Host: 172.16.20.38
- Teiid Runtime Version: 10.2.1

Administration Connection

Administration is performed via the jboss management configuration

- Port: 9990
- [Test Administration Connection](#): OK

JDBC Connection

- User name: teiidUser
- Password: (redacted)
- Port: 31000
- Port Override: 31000
- JDBC connection uses SSL:
- [Test JDBC Connection](#): OK

At the bottom, there is a navigation bar with tabs: Overview, Deployment, and Teiid Instance. The Teiid Instance tab is selected.

Tragen Sie dort die Credentials des Applikationsusers ein. Klicken Sie auf **Test Administration Connection** und **Test JDBC Connection**. Erscheint ein **OK** ist alles richtig konfiguriert. Nun muss der Server gestartet werden und wir können uns über das Developer Studio mit der laufenden Instanz verbinden.

3.3 Starten des Servers

Externer Guide

Um den Server zu starten gibt es zwei Möglichkeiten:

- Über das Developerstudio
- Über die Shell

Es wird empfohlen die zweite Variante zu wählen. Hier haben wir die Möglichkeit die zu erreichende IP-Adresse anzugeben und den Prozess des Applikationsservers ggf. in den Hintergrund zu packen, damit die bestehende SSH-Verbindung unterbrochen werden kann und der Server weiterläuft. Für letzteres muss **Server lifecycle is externally managed** abgewählt werden.

3.3.1 Starten über das Developer Studio

Um so den Server zu starten, wählen Sie den Server im Developer Studio aus und klicken Sie auf Start. Der Server ist nun gestartet und Sie sind direkt mit der laufenden Instanz verbunden. Schließen Sie nun das Developer Studio wird auch der Server heruntergefahren. Aus diesem Grund wird die Variante mit der Shell empfohlen.

3.3.2 Starten über die Shell

Hier wird beschrieben, wie der Server über die Shell gestartet werden kann. Dabei wird der Server einmal normal gestartet und einmal im Hintergrund.

3.3.2.1 Im Vordergrund

Verbinden Sie sich mit dem Server über Putty und navigieren Sie in den (`$TEIID_HOME/bin`)-Ordner. Dort liegt das Skript `standalone.sh`, welches den Server startet. Beim starten geben wir dem Server jedoch weitere Einstellungen mit.

```
$ sh standalone.sh -c standalone-teiid.xml -b 172.16.20.38 -bmanagement 172.16.20.38
```

Dies startet den Server mit der Konfigurationsdatei `standalone-teiid.xml`. Wird diese Datei nicht angegeben, wird standardmäßig `standalone.xml` verwendet. Hiermit ist aber keine Teiid-Instanz möglich, welche für die Datenvirtualisierung benötigt wird. `-b` und `-bmanagement` erhalten die IP-Adresse des Servers. Werden diese nicht angegeben, wird der Server auf localhost gestartet und wir erhalten von außerhalb keinen Zugriff auf den Applikationsserver. Um sich nun mit der laufenden Instanz zu verbinden, wählen Sie im Developer Studio den Server aus und klicken Sie start. Stellen Sie sich, dass **Server lifecycle is externally managed** abgewählt ist. Schließen Sie nun ihren Putty-Session wird auch der Applikationsserver beendet. Deshalb kann der Server auch im Hintergrund gestartet werden.

3.3.2.2 Im Hintergrund

Um den Prozess von der Shell zu koppeln muss

```
$ nohup sh standalone.sh -c standalone-teiid.xml -b 172.16.20.38 -bmanagement 172.16.20.38 &
```

eingegeben werden. Der bisherige Befehl wird um `nohup` am Anfang erweitert und hinten wird ein `&` angefügt. Nun kann die SSH-Session geschlossen werden und der Server läuft trotzdem weiter.

3.3.3 Beenden über die Shell

Je nachdem, wie der Applikationsserver gestartet wurde, muss er auch wieder beendet werden.

3.3.3.1 Im Vordergrund

Wurde der Applikationsserver im Vordergrund, also nicht im Hintergrund gestartet, kann er mit der Tastenkombination **STRG+C** beendet werden.

3.3.3.2 Im Hintergrund

Wurde der Prozess in der Hintergrund gelegt, werden ein paar mehr Schritte benötigt, um den Server wieder zu beenden. Navigieren Sie dazu wieder in den (`$TEIID_HOME/bin`)-Ordner und starten Sie das Skript `jboss-cli.sh`

```
$ sh jboss-cli.sh
```

geben Sie nun

```
[disconnected /] connect --controller=172.16.20.38
...
mit ihrer entsprechenden IP-Adresse ein. Sie sind nun mit dem laufenden Prozess verbunden. Mit ***shutdown*** fahren Sie den Applikationsserver nun herunter.
```

Shell

```
[standalone@172.16.20.38:9990 /] shutdown
```

...
Verlassen Sie die Ansicht mit **STRG+C**.

3.4 Öffnen der Web Management Console

Ist der Server gestartet, kann sich mit der Web Management Console verbunden werden. Diese läuft im Regelfall unter dem Port 9990. Sie erreichen diese, indem Sie in Ihrem Browser die IP-Adresse Ihres Servers auf dem Port 9990 öffnen. In unserem Beispiel unter: <http://172.16.20.38:9990>. Melden Sie sich dort mit den Credentials des erstellten Managementusers an.

4. Verbinden mit einer Datenquelle

Nun liegt ein gestarteter Server vor, mit dessen Instanz Sie über das Developer Studio verbunden sind. Im anschließenden wird Ihnen gezeigt, wie Sie sich mit diversen Datenquellen verbinden können.

Im Kontext ist das folgender Bereich:

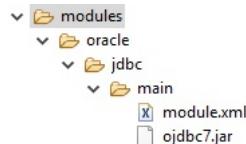
./Bilder/Schritt 2.png

Zu jeder Datenquelle, mit der sich verbunden werden soll, muss der entsprechende Treiber auf dem Applikationsserver vorliegen, hier in der Grafik JDBC Treiber genannt. Mit diesem wird eine Datasource erstellt, die nur Metadaten beinhaltet.

4.1 Oracle

Externer Guide

Zum verbinden mit einer Oracle-Datenbank liegt kein installiertes Modul vor. Aus diesem Grund müssen wir die JDBC treiben per Hand auf dem Server installieren. Hier kommt der bereits angesprochene **modules** Ordner ins Spiel. Dort werden die benötigten Treiber abgelegt. Wildfly setzt dafür eine genau definierte Ordnerstruktur vor. Öffnen Sie dafür im Developer Studio wieder das **Remote Systems** Fenster. Navigieren Sie in den Ordner ihrer Wildflyinstallation und öffnen Sie den **modules**-Ordner. Erstellen Sie dort die Ordner oracle -> jdbc -> main.



In diesen Ordner legen sie den JDBC-Treiber **ojdbc7.jar** ([interner Link](#)).

Zusätzlich wird eine **module.xml**-Datei angelegt. Es ist wichtig, dass die Datei diesen Namen besitzt, da sie sonst nicht automatisch erkannt wird. Der Inhalt der Datei wird hier gezeigt:

```xml

```
<resources>
 <resource-root path="ojdbc7.jar"/>
</resources>

<dependencies>
 <module name="javax.api"/>
 <module name="javax.transaction.api"/>
</dependencies>
```

Sie referenziert die verwendete JDBC-Datei sowie weitere bereits installierte Abhängigkeiten und erhält einen eindeutigen Namen **oracle.jdbc**. Nun ist das Modul angelegt und muss in der **standalone-teiid.xml** (\$TEIID\_HOME/bin/configuration/) referenziert werden. Öffnen Sie dafür die Datei im Text-Editor und suchen sie die Treiber ```.

```
<drivers>
 <driver name="h2" module="com.h2database.h2">
 <xa-datasource-class>org.h2.jdbcx.JdbcDataSource</xa-datasource-class>
 </driver>
 <driver name="teiid-local" module="org.jboss.teiid">
 <driver-class>org.teiid.jdbc.TeiidDriver</driver-class>
 <xa-datasource-class>org.teiid.jdbc.TeiidDataSource</xa-datasource-class>
 </driver>
 <driver name="teiid" module="org.jboss.teiid.client">
 <driver-class>org.teiid.jdbc.TeiidDriver</driver-class>
 <xa-datasource-class>org.teiid.jdbc.TeiidDataSource</xa-datasource-class>
 </driver>
 <driver name="hive12" module="org.apache.hadoop.hive12">
 <driver-class>org.apache.hadoop.hive.jdbc.HiveDriver</driver-class>
 </driver>
 <driver name="OracleJDBCDriver" module="oracle.jdbc"/>
</drivers>
```

Fügen Sie dort die Zeile

```
 <driver name="OracleJDBCDriver" module="oracle.jdbc"/>
```

ein. module referenziert das eben erstellte Modul und geben dem Treiber den Namen **OracleJDBCDriver**. Unter diesem Namen ist nun der Treiber bekannt. Jetzt können wir uns mit einer Oracle-Datenbank verbinden. Dafür suchen wir ebenfalls in **standalone-teiid.xml** die **<datasources>** und fügen eine neue hinzu.

```
<datasources>
 <datasource jndi-name="java:jboss/datasources/ExampleDS" pool-name="ExampleDS" enabled="true" use-java-context="true">
 <connection-url>jdbc:h2:mem:test;DB_CLOSE_DELAY=-1;DB_CLOSE_ON_EXIT=FALSE</connection-url>
 <driver>H2</driver>
 <security>
 <user-name>sak</user-name>
 <password>sak</password>
 </security>
 </datasource>
 <datasource jta="true" jndi-name="java:/OracleSchulungSNDDev" pool-name="OracleSchulungSNDDev" enabled="true" spy="true" use-ccm="true" statistics-enabled="true">
 <connection-url>jdbc:oracle:thin:@srv-im-etl.intern.infomotion.de:1521/pdb01d</connection-url>
 <driver-class>oracle.jdbc.OracleDriver</driver-class>
 <driver>OracleJDBCDriver</driver>
 <security>
 <user-name>dbs_dwh_cw</user-name>
 <password>dbs_dwh_cw</password>
 </security>
 </datasource>
</datasources>
```

```
<datasource jta="true" jndi-name="java:/OracleSchulungSNDDev" pool-name="OracleSchulungSNDDev" enabled="true" spy="true" use-ccm="true" statistics-enabled="true">
 <connection-url>jdbc:oracle:thin:@srv-im-etl.intern.infomotion.de:1521/pdb01d</connection-url>
 <driver-class>oracle.jdbc.OracleDriver</driver-class>
 <driver>OracleJDBCDriver</driver>
 <security>
 <user-name>dbs_dwh_cw</user-name>
 <password>dbs_dwh_cw</password>
 </security>
</datasource>
```

Diese Datasource verwendet den von uns angelegten **driver** **OracleJDBCDriver**. Der **jndi-name** muss mit **java:/** beginnen. Dahinter kommt ein beliebiger Name, unter welchem unsere Datenbank zu erreichen ist. Die **<connection-url>** beinhaltet die Adresse der Oracledatenbank, mit der sich verbunden werden will, sowie unter **<security>** die Anmelddaten der Datenbank. Damit diese Änderung aktiv ist, muss der Applikationsserver neu gestartet werden. Ist der Applikationsserver neu gestartet, öffnen Sie die Weboberfläche der Managementconsole und navigieren Sie zu Subsystems -> Datasources -> Non XA -> **Ihre Datasource** und klicken auf **View**.

WildFly 11.0.0.Final | Messages: 0 | admin ▾

Home Deployments Configuration Runtime Access Control Patching

Configuration	Subsystem (19)	Type	Datasource	Add
Subsystems	JCA	Non-XA	ExampleDS	
Interfaces	Datasources	XA	ExcelTestVDB	
Socket Binding	Resource Adapters		HadoopDS	
Paths	Mail		OracleSchulungSNDev	<a href="#">View</a> ▾
System Properties	Transactions		OracleSchulungVDB	
	EJB 3			
	EE			

Wählen Sie nun den Reiter **Connection** und klicken Sie auf **Test Connection**.

JDBC datasource 'OracleSchulungSNDev' (enabled)

JDBC datasource configurations.

[Disable](#)

Attributes Connection Pool Security Properties Validation Timeouts Statements

[Test Connection](#) Need Help?

[Edit](#)

**Connection URL:** jdbc:oracle:thin:@srv-im-etl.intern.infomotion.de:1521/pdb01d

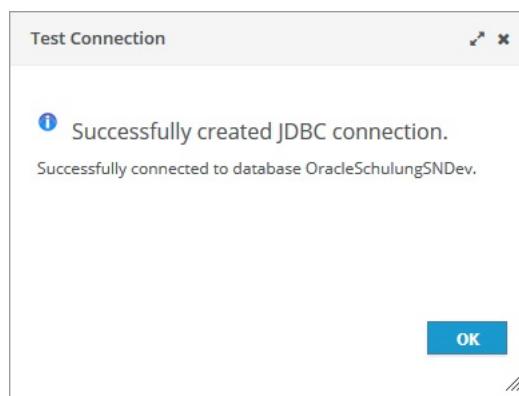
**New Connection Sql:**

**Transaction Isolation:**

**Use JTA?:** true

**Use CCM?:** true

Erhalten Sie folgendes Fenster, ist die Datasource erfolgreich konfiguriert und die Daten der Datenbank stehen zur weiteren Verarbeitung zur Verfügung.



## 4.2 MongoDB

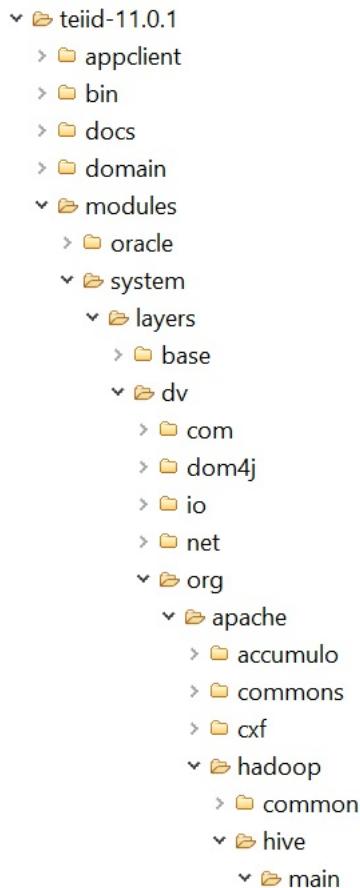
[Externer Guide](#)

## 4.3 Hive

[Externer Guide](#)

Um uns mit einer Hive-Instanz zu verbinden, benötigen wir ebenfalls die entsprechenden JDBC-Treiber. Da unsere Hive-Instanz im AWS läuft, benutzen wir die von Amazon angebotenen Treiber. Diese sind [hier](#) zu finden.

Diese müssen ebenfalls im **modules**-Ordner platziert werden, im Pfad \$TEIID\_HOME/modules/system/layers/dv/org/apache/hadoop/hive/main



Dazu wird wieder eine **modules.xml** angelegt mit folgendem Inhalt:

```

<?xml version="1.0" encoding="UTF-8"?>
<module name="org.apache.hadoop.hive" xmlns="urn:jboss:module:1.0">
<resources>
<resource-root path="commons-codec-1.3.jar"/>
<resource-root path="commons-logging-1.1.1.jar"/>
<resource-root path="hive metastore.jar"/>
<resource-root path="hive service.jar"/>
<resource-root path="HiveJDBC41.jar"/>
<resource-root path="httpClient-4.1.3.jar"/>
<resource-root path="httpcore-4.1.3.jar"/>
<resource-root path="libfb303-0.9.0.jar"/>
<resource-root path="libthrift-0.9.0.jar"/>
<resource-root path="log4j-1.2.14.jar"/>
<resource-root path="ql.jar"/>
<resource-root path="slf4j-api-1.5.11.jar"/>
<resource-root path="slf4j-log4j12-1.5.11.jar"/>
<resource-root path="TCLIServiceClient.jar"/>
<resource-root path="zookeeper-3.4.6.jar"/>
</resources>
<dependencies>
<module name="org.slf4j"/>
<module name="org.apache.commons.logging"/>
<module name="javax.api"/>
<module name="javax.resource.api"/>
</dependencies>
</module>

```

Hier werden alle abgelegten Treiberdateien referenziert. Unser Modul erhält den Namen **org.apache.hadoop.hive**. Nun muss dieser Treiber ebenfalls in der **standalone-teiid.xml** angegeben.

```

<driver name="hive" module="org.apache.hadoop.hive">
<driver-class>com.amazon.hive.jdbc41.HS2Driver</driver-class>
</driver>

```

Der Treiber erhält den Namen **hive\*** und die Treiberklasse ist **com.amazon.hive.jdbc41.HS2Driver**. Mit diesem Treiber legen wir nun die passende Datasource an.

```

<datasource jta="true" jndi-name="java/HadoopDS" pool-name="HadoopDS" enabled="true" use-ccm="true">
<connection-url>jdbc:hive2://ec2-35-159-52-226.eu-central-1.compute.amazonaws.com:10000/default</connection-url>
<driver-class>com.amazon.hive.jdbc41.HS2Driver</driver-class>
<driver>hive</driver>
</datasource>

```

Die Connection-url beinhaltet die DNS unseres Amazon-EMR-Masters. Dort läuft Hive standardmäßig auf **Port 10000**, angeführt von **jdbc:hive2**. Nun kann wie bei der Einrichtung der Oracle-Datasource überprüft werden, ob die Verbindung erfolgreich hergestellt wurde.

## 4.4 Excel

Externer Guide

## 4.5 XML

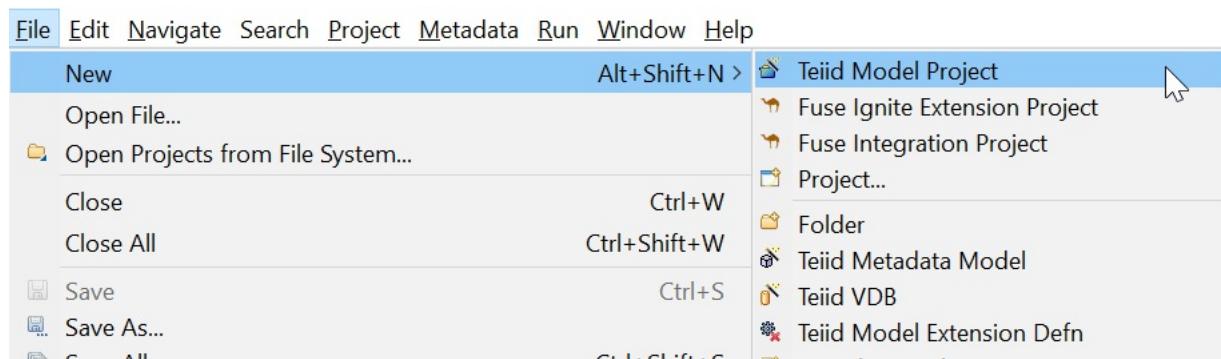
Externer Guide

# 5. Modellieren von Daten

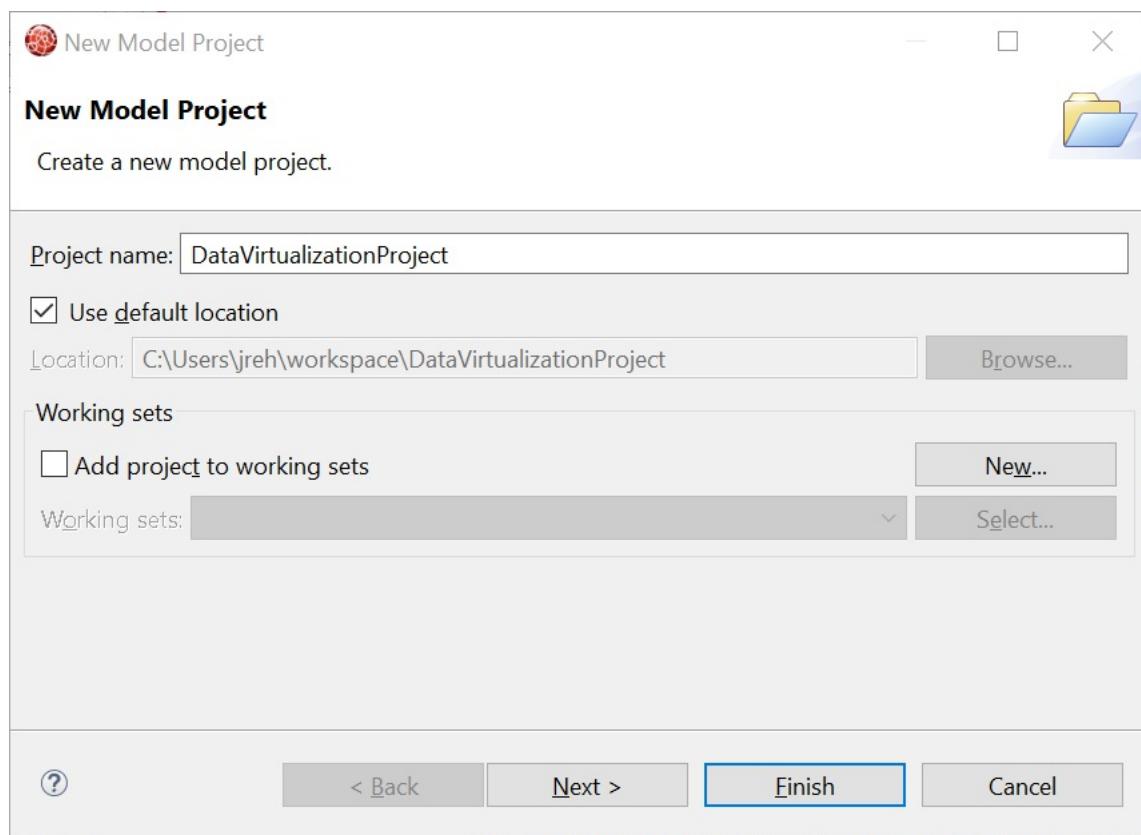
In diesem Kapitel wird das Erstellen von Model und Views beschrieben, mit denen später eine VDB erzeugt wird. Im Kontext sind das die markierten Bereiche.

!Schritt3(./Bilder/Schritt 3.png)

Um nun ein Metadaten-Modell für die verschiedenen Quellen zu erzeugen, muss im Developer Studio ein neues Projekt angelegt werden.



Geben Sie einen Namen für das Projekt ein und bestätigen Sie.



Es wird ein Projekt angelegt mit folgender Ordnerstruktur:



In den Ordner **sources** werden die Modelle abgelegt. Um solch ein Modell zu erzeugen und direkt mit der Datasource zu verbinden, drücken wir im Reiter **Guides** und **Model Teiid Data Source** auf **Create source model from teiid data source DDL**.

The screenshot shows the 'Model Explorer' interface with the 'Guides' tab selected. Below it, the 'Modeling Actions...' section is open, showing the 'Action Sets' dropdown set to 'Model Teiid Data Source'. Underneath, there are three options: 'Define Teiid Model Project >> <undefined>', 'Create Data Source', and 'Create source model from Teiid data source DDL >> <undefined>'.

## 5.1 Oracle

Um ein Modell mit der Oracle-Datasource zu erstellen, wählen Sie die Oracle-Datasource aus und klicken Sie auf **next**.

The screenshot shows the 'Import using a Teiid Connection' dialog. It includes a warning message about using the JDBC Importer if the selected source is JDBC. Below is the 'Importer Description' section, which states that the importer utilizes a Teiid runtime data source connection to return consistent Teiid DDL runtime metadata. The 'Data Sources' table lists various data sources with their JNDI names and driver/module IDs. The 'java:/OracleSchulungSNDDev' entry is selected. The 'Data Source Properties' table is empty. At the bottom, there are buttons for '?', '< Back' (disabled), 'Next >', 'Finish' (disabled), and 'Cancel'.

Name	Value

Wählen Sie im nächsten Fenster als translator **oracle** und geben Sie im Feld **Schema Pattern DBS\_DWH\_CW%** ein. Das nimmt alle Schemas aus der Datenbank, die mit **DBS\_DWH\_CW** beginnen.

Import using a Teiid Connection

### Select the translator for the import

Press the "Next >" button to continue.

**Default Server: WildFly 11 teiid 11**

**Source Definition**

DataSource:	java/OracleSchulungSNDDev
Driver or Module ID:	OracleJDBCDriver
Translator:	oracle

**Import Properties**

Property	Value
Rename Duplicate Columns	false
Rename Duplicate Tables	false
Schema Pattern	DBS_DWH_CW%
Sequence Name Pattern	
Table Name Pattern	
Table Types	TABLE
Use Any Index Cardinality	false
Use Catalog Name	true

Note: See Teiid documentation for details on importer properties

**Optional Source Import Properties**

Name	Value

Klicken Sie auf **next**, benennen Sie das Model und wählen Sie einen Ordner zum Ablegen des Modells.

Import using a Teiid Connection

### Select the target model for the import

Press the "Next >" button to continue.

**Default Server: WildFly 11 teiid 11**

**Target Model Definition** Advanced

Location:	DataVirtualizationProject/sources
Name:	OracleModel

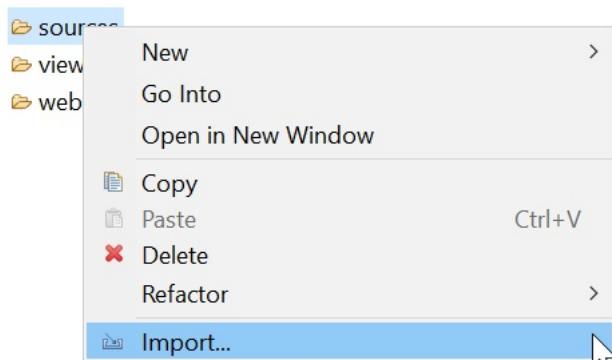
**Model Status**

Model name defined: OracleModel

Update existing model  
 Filter unique constraints that mimic PKs.  
 Create and apply a Connection Profile.

Klicken Sie auf **next** und wählen Sie die Tabellen zum importieren aus. Tritt dabei ein Fehler auf, dass die DDL nicht importiert werden kann, erstellen Sie lokal eine DDL dabei und kopieren Sie den angezeigten DDL-Text in diese Datei. Dort entfernen Sie in den Klammern der **bigdecimal**-Variablen die Zahl hinter dem Komma, sowie das Komma. Entfernen Sie zudem die oberste Zeile, in der der Namespace gesetzt wird.

Um nun das Model zu erzeugen, klicken Sie mit Rechtsklick auf den **sources**-Ordner in Ihrem Projekt und klicken Sie **import**.



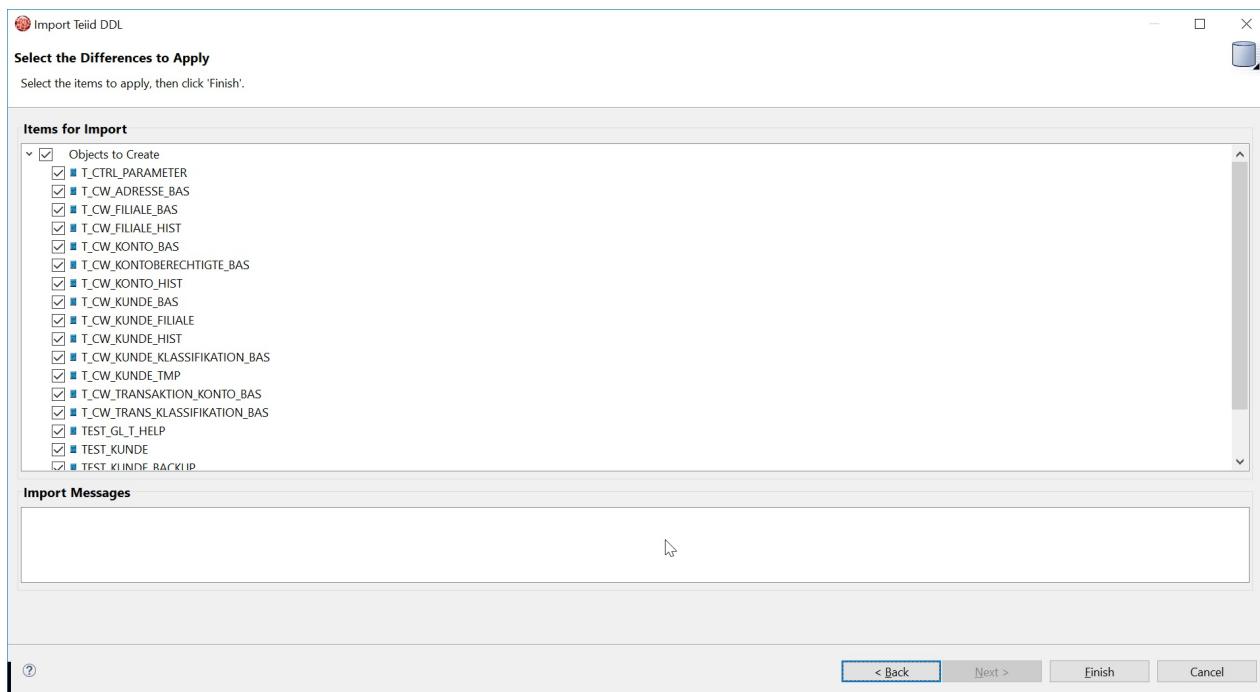
Dort wählen Sie unter Teiid Designer **DDL File(Teiid)**.

The screenshot shows the 'Import' dialog box with the title 'Import'. The main area is titled 'Select' with the sub-instruction: 'Import metadata from Teiid DDL file into a new or existing relational source or view model.' Below this is a section titled 'Select an import wizard:' with a 'type filter text' input field. A tree view lists import wizards: 'Team' (expanded), 'Teiid Designer' (selected), and several sub-options like 'DDL File (General) >> Source or View Model' and 'File Source (Flat) >> Source and View Model'. At the bottom are navigation buttons: '?', '< Back', 'Next >' (highlighted in blue), 'Finish', and 'Cancel'.

Klicken Sie auf **Choose from file system** und navigieren Sie zu der eben erstellten DDL-Daten und klicken Sie auf **next**.

The screenshot shows the 'Provide DDL source' dialog box with the title 'DDL File Definition'. It contains fields for 'File' (set to 'C:\Users\jreh\Desktop\OracleModel.ddl'), 'Model folder' ('/DataVirtualizationProject/sources'), 'Model name' ('OracleModel'), and 'Model type' ('Source Model'). There is also a checkbox for 'Generate valid default SQL (SELECT null AS column\_name, etc...)'. Below this is a 'Notes' section with a note about escaped quotes and a sample string. Navigation buttons at the bottom are: '?', '< Back', 'Next >' (highlighted in blue), 'Finish', and 'Cancel'.

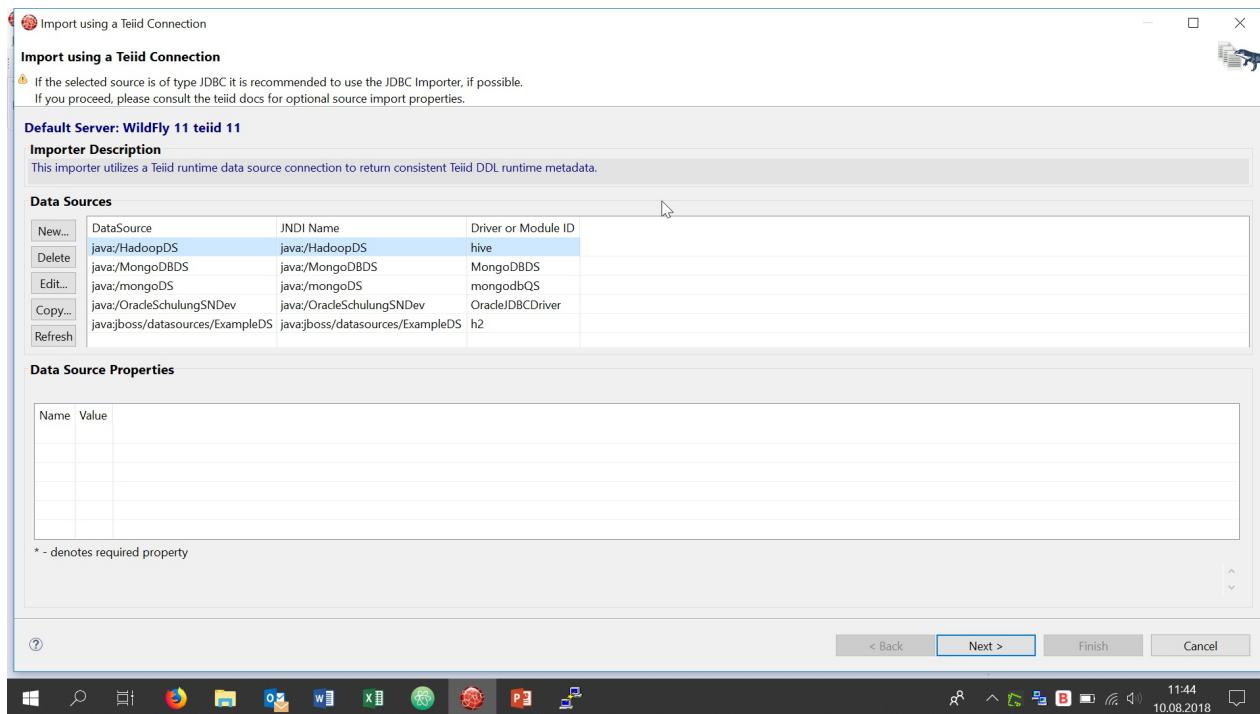
Nun können Sie in diesem Menü die gewünschten Tabellen auswählen.



Das Modell ist nun erfolgreich importiert.

## 5.2 Hive

Um ein Modell für eine Hive-Datasource zu erzeugen, beginnen Sie wie bei den Schritten zum Modell der Oracle-Datasource. Klicken Sie im Guides Fentster ebenfalls auf **Create source model from teiid data source DDL** und wählen Sie dort die Hive-Datasource.



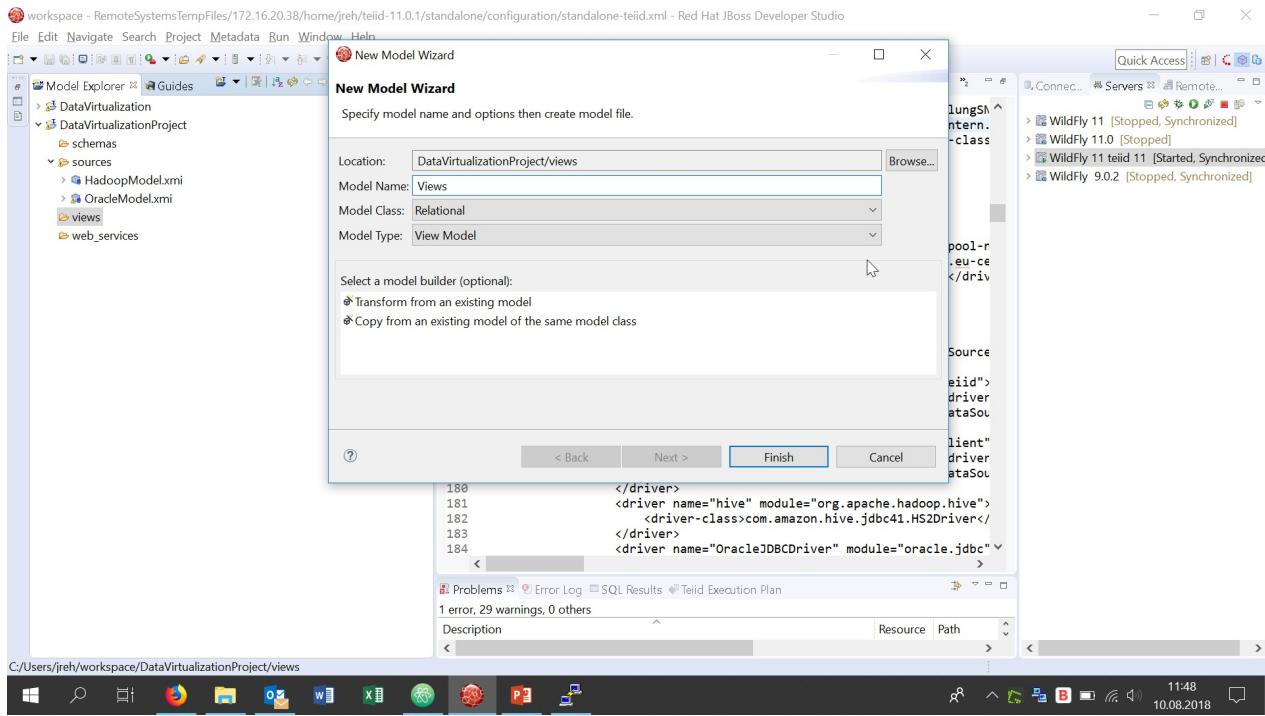
Klicken Sie auf **next** und stellen Sie den translator auf **hive**.



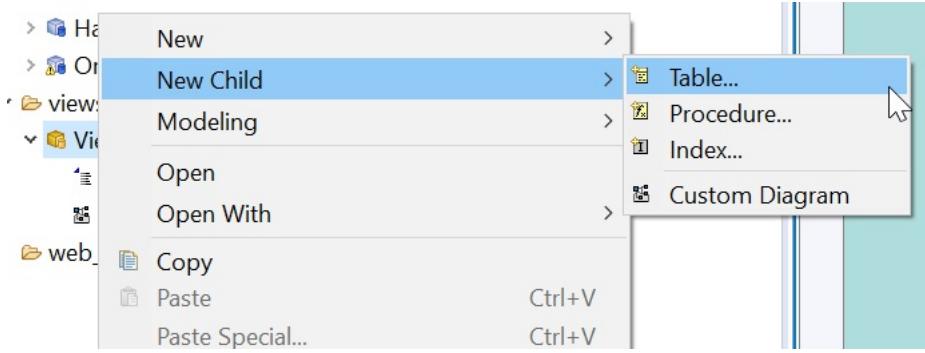
Nach dem Import wählen Sie die gewünschten Tabellen und klicken Sie auf **Finish**. Das Modell ist erfolgreich erstellt.

## 5.3 View

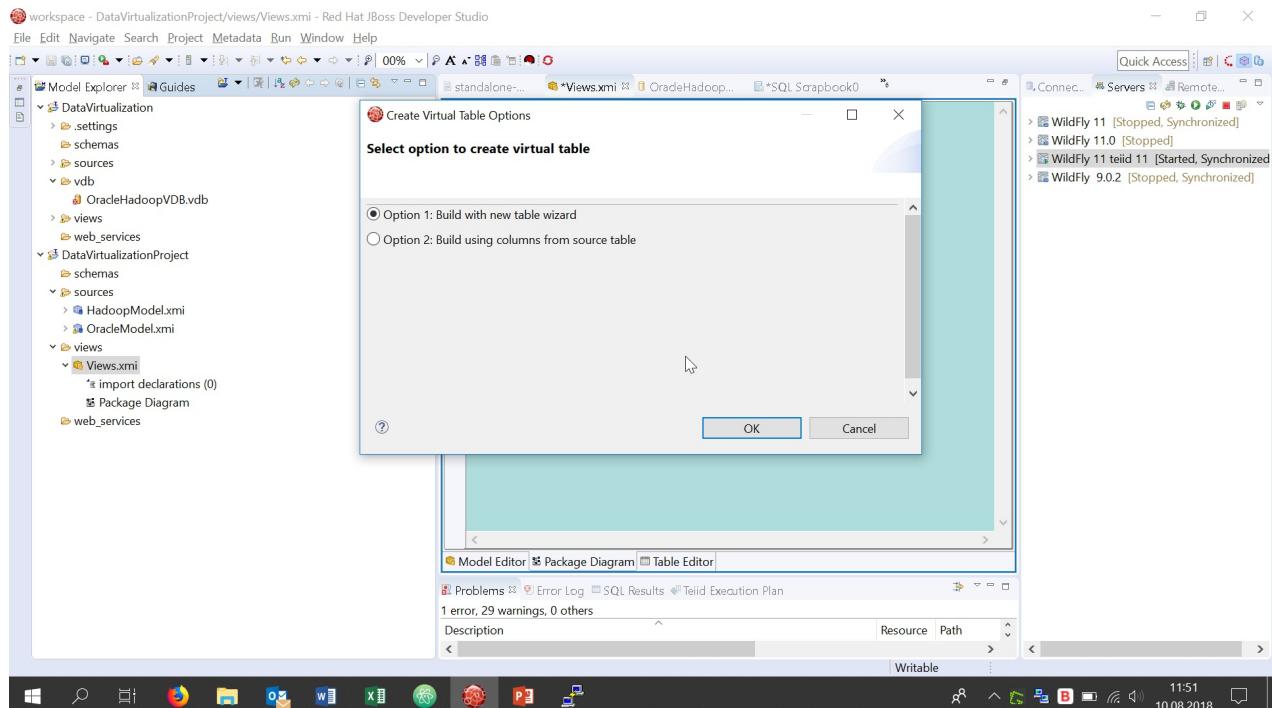
Um eine View zu erzeugen, erstellen Sie ein neues Medadaten Modell. Stellen Sie jedoch den Model Type auf **View Model** und vergeben Sie einen Namen.



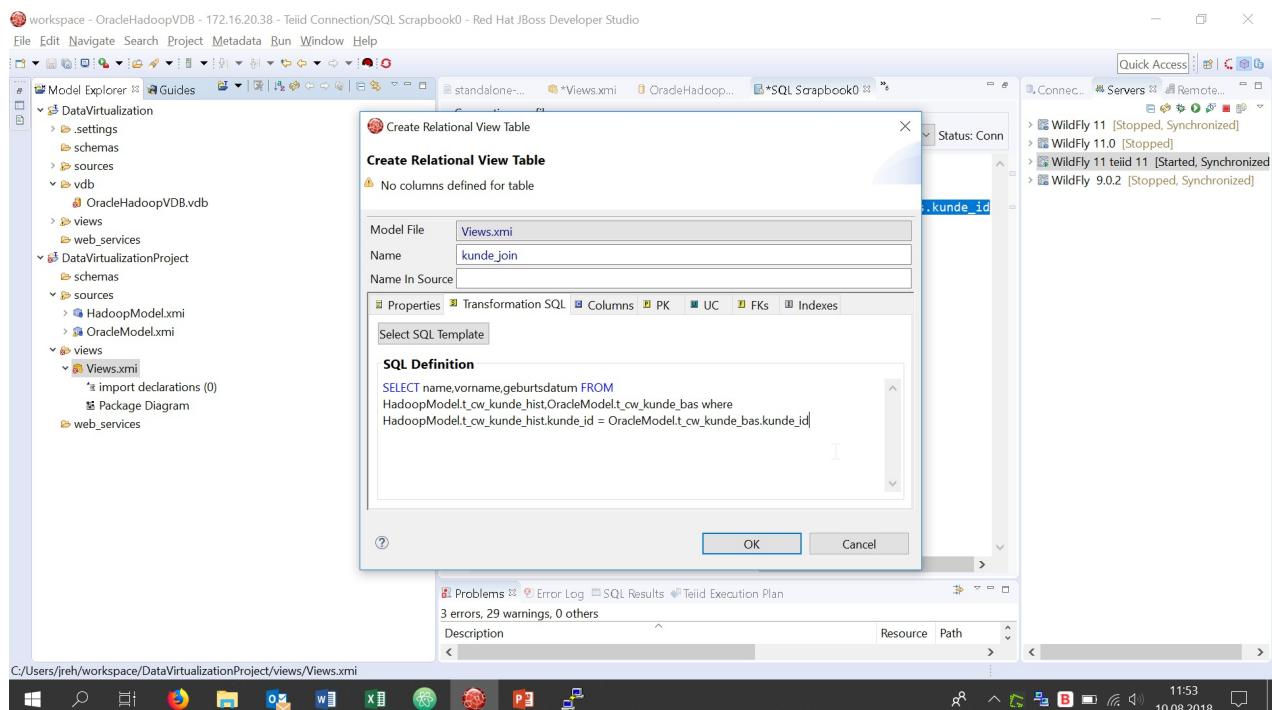
Rechtsklicken Sie auf das erzeugte Modell und fügen Sie ein neue Table an durch New Child -> Table...



Wählen Sie Option 1



und geben Sie im Reiter **Transformation SQL** das SQL-Statement ein.



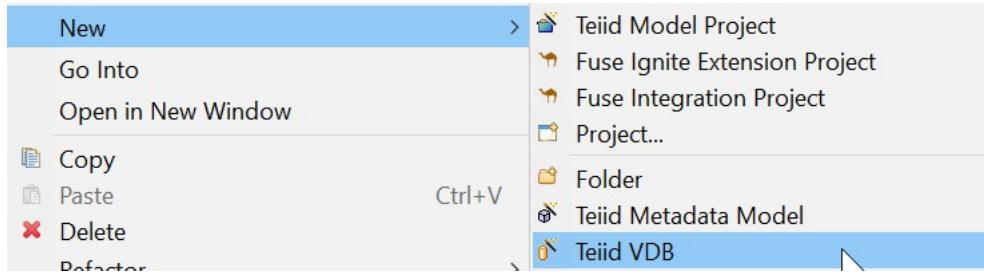
Die View ist erfolgreich angelegt.

## 6. Erstellung der VDB

Die Erstellung der Modelle ist bisher nur lokal gewesen. Nun wird eine VDB mit diesem Modellen erzeugt, die anschließend auf dem Applicationsserver platziert wird. Aktuell befinden wir uns hier:

!Schritt3,5](./Bilder/Schritt 3,5.png)

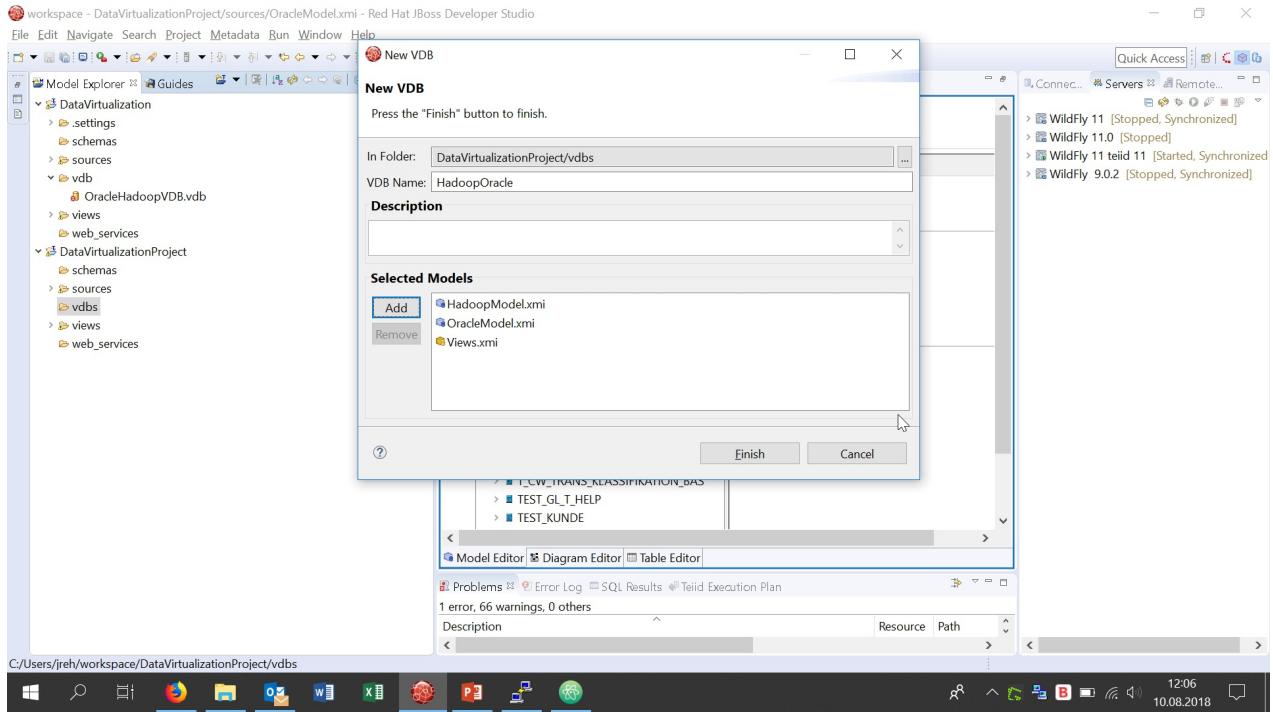
Um eine VDB zu erzeugen, rechtsklicken Sie auf einen Ordner im Projekt und wählen Sie unter new **Teiid VDB**.



Klicken Sie im nächsten Fenster auf **Add** und wählen Sie die erstellten Modelle und die View.

The image shows two overlapping windows. On the left is the 'New VDB' dialog, which has fields for 'In Folder:' (set to 'DataVirtualizationProject/vdbs') and 'VDB Name:' (set to 'HadoopOracle'). It also contains sections for 'Description' (empty) and 'Selected Models' (which currently has an 'Add' button and a 'Remove' button). On the right is the 'Select Models' dialog, titled 'Select models from your workspace to add to this new VDB'. It lists several model files under a tree structure: 'DataVirtualizationProject' (expanded), 'sources' (expanded), 'vddb' (expanded), and 'views' (expanded). Under 'sources', 'HadoopModel.xmi' and 'OracleModel.xmi' are selected, indicated by a blue selection bar. Under 'views', 'Views.xmi' is selected. A question mark icon is visible at the bottom left of the 'Select Models' window.

Beenden Sie das nächste Fenster mit **Finish**.



Die VDB ist erfolgreich angelegt.

## 6.1 Data Roles

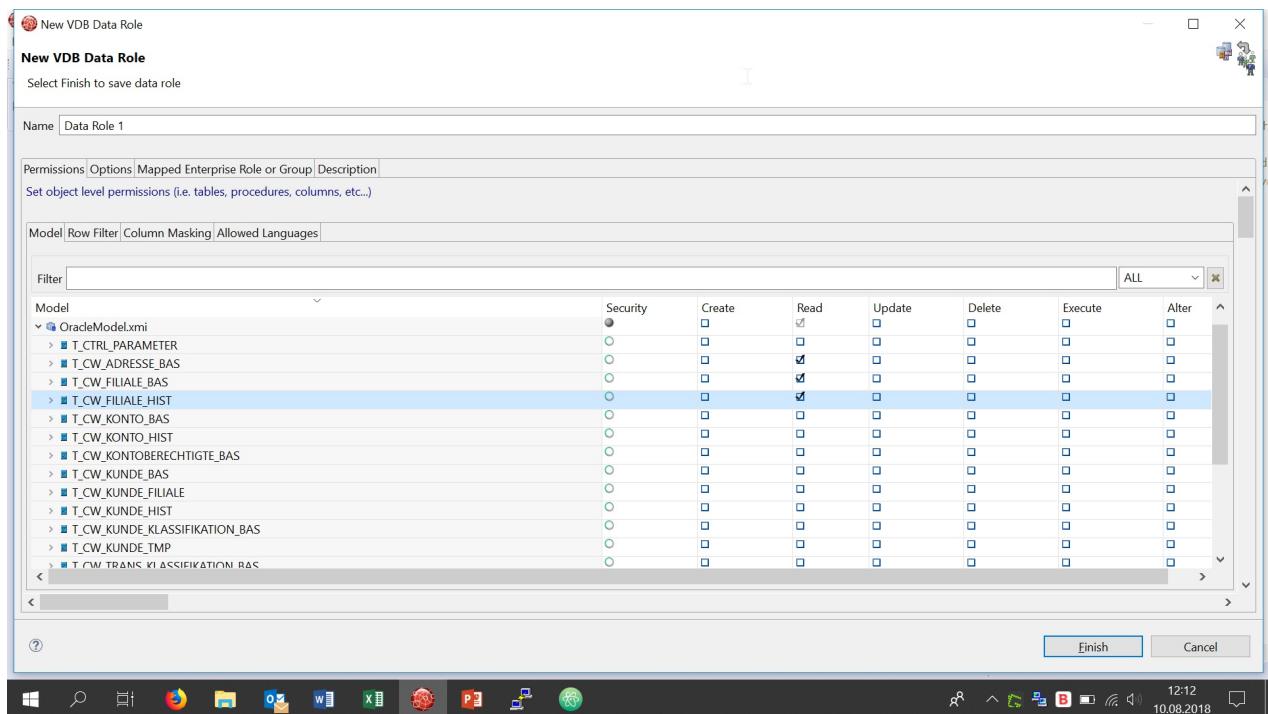
Um Zugriffsbeschränkungen auf die VDB zu legen, werden Data Roles verwendet. Eine Data Role wird einer Applikationsuser Rolle zugeordnet. Jeder Applikationsuser, der dieser Rolle angehört, erhält die Beschränkungen, die auf der Data Role liegen. Diese können erstellt werden, indem Sie einen Doppelklick auf die eben erstellte VDB machen und auf den Reiter **Data Roles** wechseln.

Location: /DataVirtualizationProject/vdbs		Last validated : Fri A... 201
Version	1	Synchronize All Show Import VDBs Deploy Test Save as XML
<a href="#">Models</a> <a href="#">Schemas</a> <a href="#">UDF Jars</a> <a href="#">Other Files</a> <a href="#">Description</a> <a href="#">Data Roles</a> <a href="#">Properties</a> <a href="#">User Properties</a> <a href="#">Translator Overrides</a>		
Name	Description	

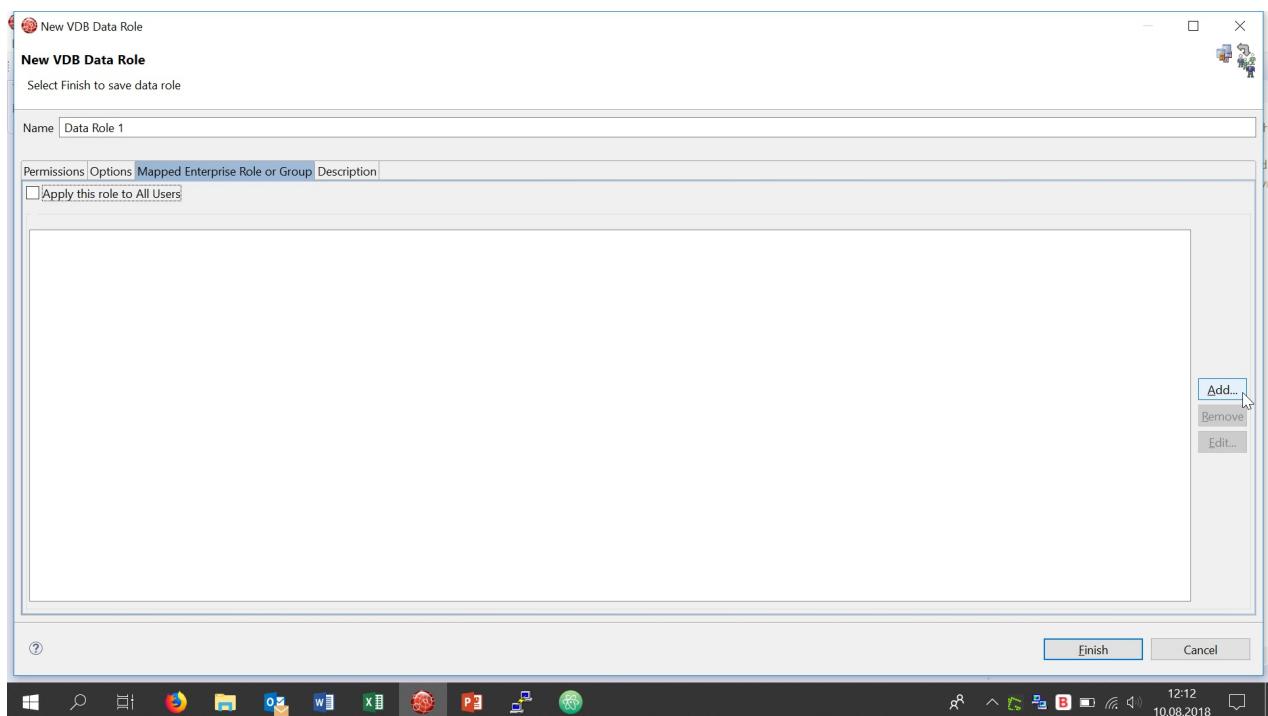
Dort klicken Sie unten links auf das gezeigte Symbol.



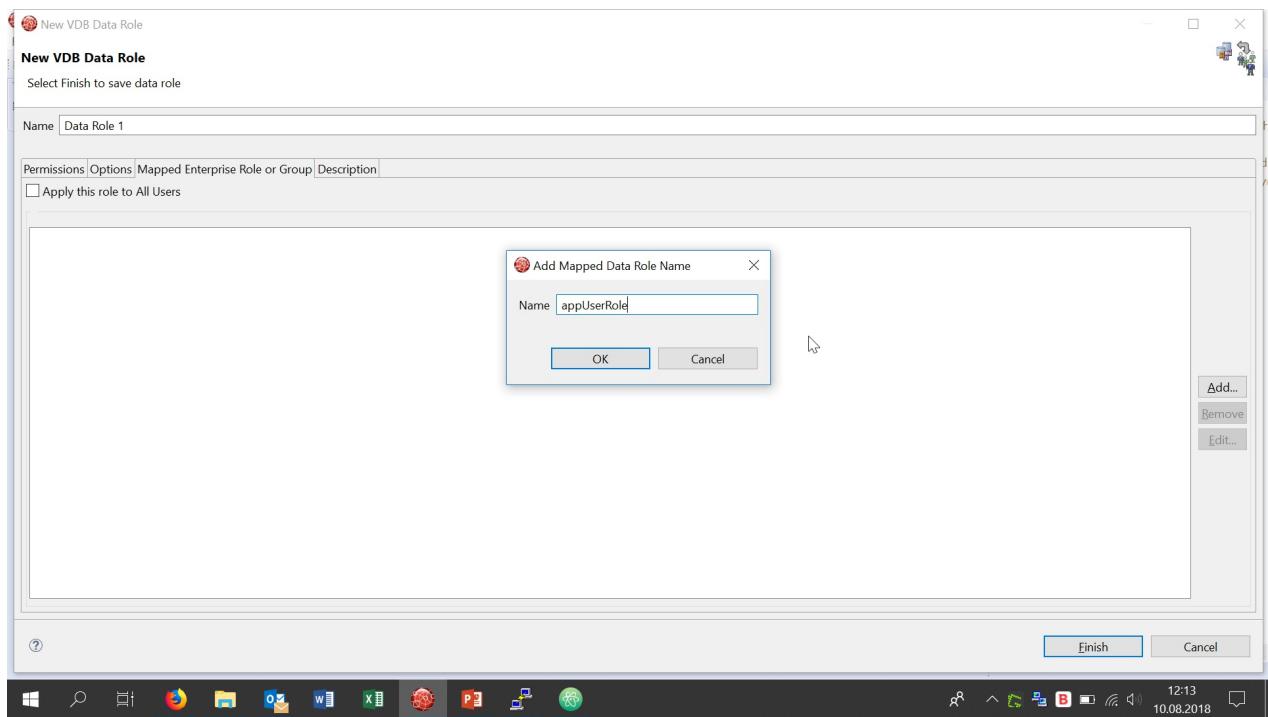
Nun können Sie dort die Zugriffsberechtigungen auf die einzelnen Spalten jeder Tabelle und jedes Models festlegen.



Nun wechseln Sie auf den Reiter **Mapped Enterprise Role or Group**. Dort kann die erstellte Data Role einer Applikationsuser Role oder einer ganzen Gruppe zugewiesen werden.



Geben Sie den Namen der Rolle ein und bestätigen Sie. In diesem Beispiel gilt die erstellte Data Role für alle User, die der Role appUserRole angehören.



Es können beliebig viele Data Roles erzeugt werden und auf die Rollen aufgeteilt werden. Zu beachten ist, dass eine Erlaubnis stärker gewichtet ist, als ein Verbot. Hat ein User keine positive Erlaubnis auf einer bestimmte Tabelle, hat er auch keinen Zugriff drauf. Hat ein User zwei Data Roles zugewiesen, wobei eine die Benutzung einer Spalte verbietet und die andere Data Role die Benutzung der Spalte erlaubt, hat der User im generellen eine Erlaubnis.

#### #6.2 Testen der VDB

Um die VDB zu testen, öffnen Sie die VDB und klicken Sie auf **test**. Kommt es hier zu einem Fehler, dass ein Translator nicht eingestellt ist, oder die DataSource nicht angegeben ist, muss dem genannten Model per Rechtsklick -> Modeling die DataSource zugewiesen werden bzw. der Translator gesetzt werden.

Location: /DataVirtualizationProject/vdbs      Last validated

Version 1    Synchronize All    Show Import VDBs    Deploy    **Test**    Save as XML

Wenn auf **test** geklickt wird, wird die VDB auf dem Server deployed. Nun kann über das geöffnete Scrapbook die Verbindung getestet werden, indem SQL-Abfragen erstellt werden.

Links neben dem Scrapbook finden Sie die Verbindung zu der Datenbank. Dies läuft intern über den Teiid JDBC Treiber. Per Rechtsklick -> Properties -> Driver Properties auf der Verbindung können Sie die Anmeldeoptionen variieren. Hier können bspw. die Data Roles und die Zugriffsberechtigungen getestet werden.

## 7. Verbinden mit VDB

Die VDBs befinden sich nun auf dem Server und haben alle Metadaten um erfolgreiche Abfragen an die grundlegenden Datenquellen zu stellen. Von außen kann sich nun mit den VDBs verbunden werden.

### 7.1 Tableau

Um sich per Tableau mit einer VDB zu verbinden, benötigen wir einen ODBC-Treiber. Hierbei läuft auf dem Wildfly Server auf Port 35432 eine emulierte Schnittstelle zu einer PostgreSQL-Datenbank, welcher wir nutzen werden. Dafür benötigen wir einen PostgreSQL-ODBC Treiber. Diese sind [hier](#) zu finden. Installieren Sie den neusten Treiber passend zu Ihrem Betriebssystem. Unter Windows öffnen Sie das Programm zum Einrichten von ODBC-Datenquellen. Hier unbedingt auf das benutzte Betriebssystem achten, ob 32 oder 64-Bit und das dementsprechende Programm öffnen.

ODBC-Datenquellen-Administrator (64-Bit)

Benutzer-DSN System-DSN Datei-DSN Treiber Ablaufverfolgung Verbindungspooling Info

Benutzerdatenquellen:

Name	Plattform	Treiber
Excel Files	32-Bit	Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *.xlsb)
MS Access Database	32-Bit	Microsoft Access Driver (*.mdb, *.accdb)

Hinzufügen...  
Entfernen  
Konfigurieren...

Der Treiber dieses Benutzer-DSNs ist nur in der 32-Bit-Version verfügbar. Dieser kann nur mit dem 32-Bit-ODBC-Datenquellen-Administrator entfernt oder konfiguriert werden.

OK Abbrechen Übernehmen Hilfe

Wählen Sie unter **System-DSN PostgreSQL Unicode(64x)** aus und klicken Sie Hinzufügen.

Neue Datenquelle erstellen

Wählen Sie einen Treiber aus, für den Sie eine Datenquelle erstellen möchten.

Name	Version	Firm
Amazon Redshift (x64)	1.03.07.1000	Am
ODBC Driver 13 for SQL Server	2015.130.811.168	Mic
Oracle in OraClient12Home1	12.01.00.02	Ora
Oracle in OraClient12Home2	12.01.00.02	Ora
PostgreSQL ANSI(x64)	10.03.00.00	Pos
<b>PostgreSQL Unicode(x64)</b>	10.03.00.00	Pos
SQL Server	10.00.17134.01	Mic

< Zurück Fertig stellen Abbrechen

Geben Sie dort die Verbindungsdaten mit Ihrem Server ein und den Port **35432** sowie die Logindaten.

PostgreSQL Unicode ODBC Driver (psqlODBC) Setup

Data Source	TeiidDatasource	Description	<input type="text"/>
Database	OracleHadoopVDB	SSL Mode	disable
Server	172.16.20.38	Port	35432
User Name	teiidUser	Password	●●●●●●●●●●

Options

Unter Datasource stellen Sie die gezeigten Einstellungen ein.

Advanced Options (adminTeiid) 1/3

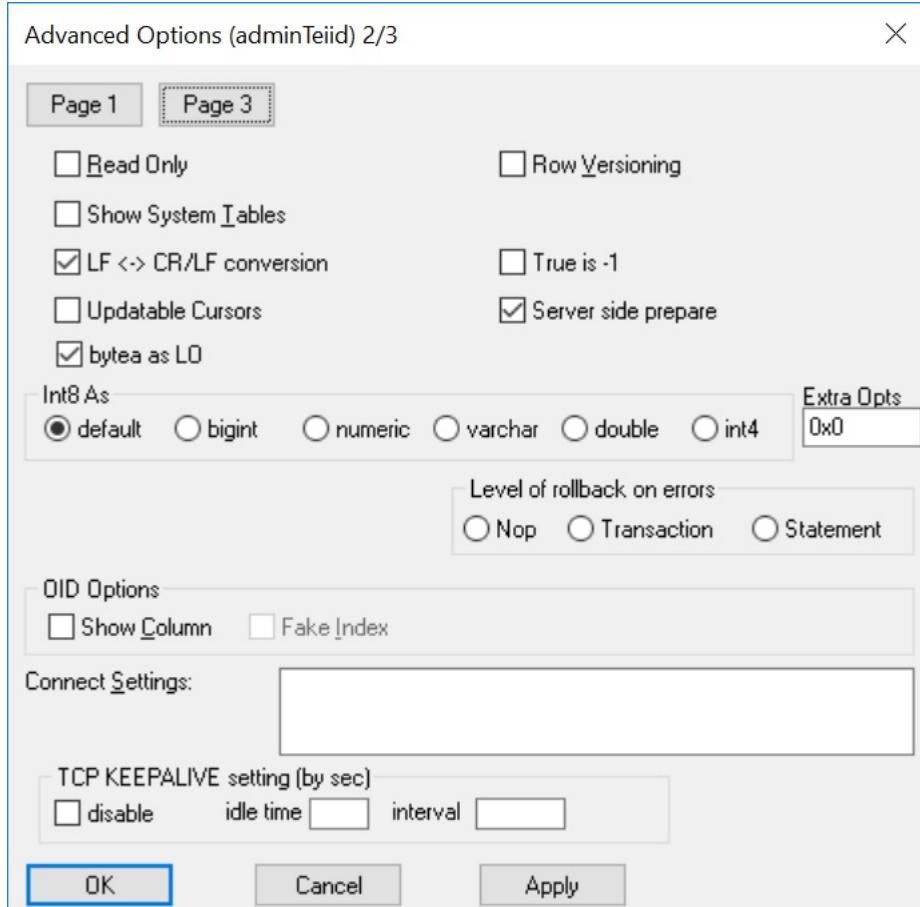
CommLog (C:\pgsqlodbc\_xxxx.log)  
 Parse Statements

Recognize Unique Indexes  
 Use Declare/Fetch  MyLog (C:\mylog\_xxxx.log)

Unknown Sizes  
 Maximum  Don't Know  Longest

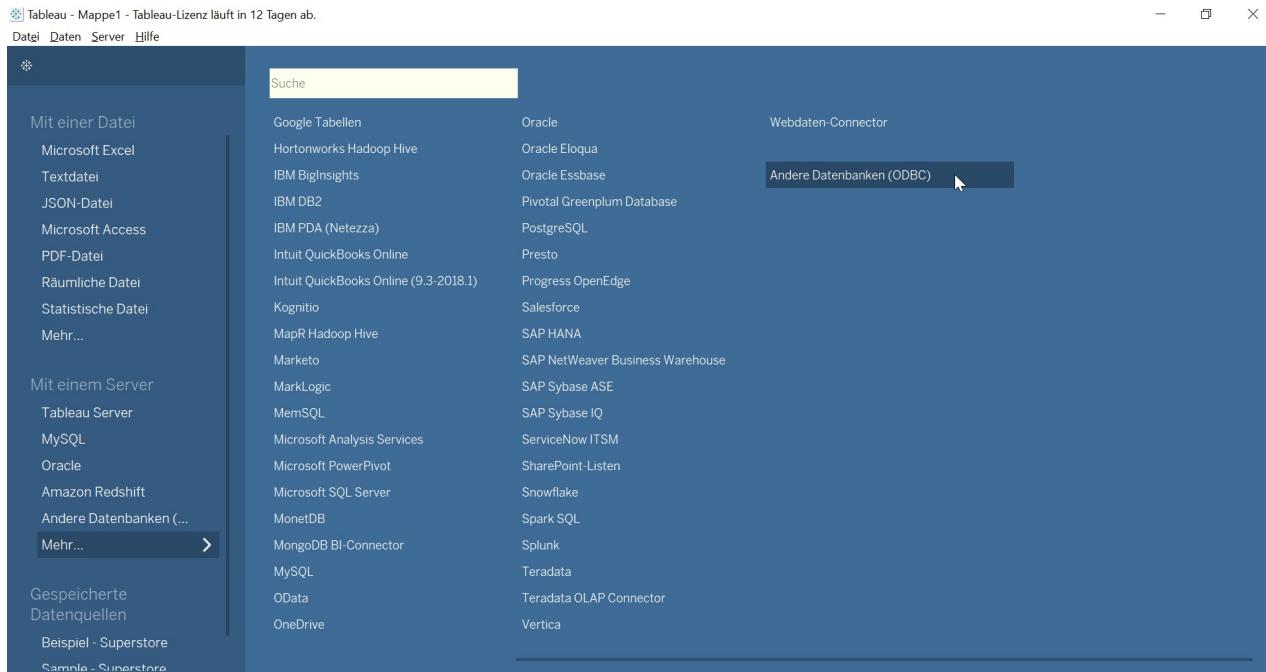
Data Type Options  
 Text as LongVarChar  Unknowns as LongVarChar  Bools as Char

Miscellaneous  
Max Varchar:  Max LongVarChar:   
Cache Size:  SysTable Prefixes:



Damit ist einer Verbindung eingerichtet und testen Sie diese ggf. mit dem Button **test**.

Öffnen Sie nun Tableau und wählen Sie **Andere Datenbanken (ODBC)** aus.



Im geöffneten Fenster wählen Sie die erstellte DSN aus und klicken auf **Verbinden** und anschließend auf **Anmelden**.

## Andere Datenbanken (ODBC)

Verbindung per

Eine generische ODBC muss zusätzlich konfiguriert werden, damit die Veröffentlichung erfolgreich durchgeführt werden kann. Wählen Sie DSN (Data Source Name) aus für eine plattformübergreifende Portabilität. Ein DSN mit demselben Namen muss auf Tableau Server konfiguriert werden.

DSN:

adminTeiid

Treiber:

PostgreSQL Unicode(x64)

**Verbinden**

Verbindungsattribute

Server:

172.16.20.38

Port:

35432

Datenbank:

OracleHadoopVDB

Benutzername:

admin

Kennwort:

\*\*\*\*\*

Zeichenfolgen-Extras:

```
SSLmode=disable;ReadOnly=0;Protocol=7.4;FakeOidIndex=0>ShowOidColumn=0;RowVersioning=0>ShowSystemTables=0;Fetch=100;UnknownSizes=0;MaxVarcharSize=1000;MaxLongVarcharSize=8190;Debug=0;CommLog=0;UseDeclareFetch=1;TextAsLongVarchar=1;UnknownsAsLongVarchar=0;BoolsAsChar=1;Parse=1;ExtraSysTablePrefixes=;LFConversion=1;UpdatableCursors=0;TrueIsMinus1=0:BI=0:BvteaAs
```

**Anmelden**

Sie sind nur per Tableau mit der VDB verbunden.