Philips **Health**Suite Hackathon: OData Cookbook

March 6th-8th, 2015

PHILIPS

OData Cookbook

Contents

The Ingredients

OData allows systems to share objects with other systems via REST services. By following certain conventions and message formats, a consuming system can discover details about the objects available in the other system and then perform queries to retrieve lists of objects (feeds) or individual objects. The entities exposed by the **Health**Suite OData product are Patient and Observation.

While the OData specification supports the notion of remote systems being able to create, modify, or delete objects, the initial version of the the **Health**Suite OData product only supports read operations.

OData 2.0

Developers consuming an OData service will probably be using a framework or platform that shields them from some of the underlying details of how OData works. Nevertheless, it's good to have a basic understanding.

An OData service will have a base URL that will be at the root of all requests to that service. If you access the base URL directly (e.g., https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/) then a list of the available objects is returned. Here's an example of the list returned by the **Health**Suite OData service:

```
<?xml version="1.0"?>
<service xmlns="http://www.w3.org/2007/app"
                    xmlns:atom="http://www.w3.org/2005/Atom"
                    xml:base="https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/">
  <workspace>
    <atom:title>Default</atom:title>
    <collection href="Patients">
      <atom:title>Patients</atom:title>
    </collection>
    <collection href="Observations">
      <atom:title>Observations</atom:title>
    </collection>
  </workspace>
</service>
```

Per the OData specification, appending "$metadata" to the base URL () will return details about each of the objects in that service including their fields, the types for those fields, and relationships between objects. Here's the $metadata output for the **Health**Suite OData service:

```
<?xml version="1.0"?>
<edmx:Edmx xmlns:edmx="http://schemas.microsoft.com/ado/2007/06/edmx" Version="1.0">
  <edmx:DataServices xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
m:DataServiceVersion="1.0">
    <Schema xmlns="http://schemas.microsoft.com/ado/2008/09/edm" Namespace="com.philips.odata2.ODataDhp">
      <EntityType Name="Patient">
        <Key>
          <PropertyRef Name="id"/>
        </Key>
        <Property Name="id" Type="Edm.String" Nullable="true"/>
        <Property Name="healthsuite-identifier" Type="Edm.String" Nullable="true"/>
      <Property Name="gender" Type="Edm.String" Nullable="true"/>
      <Property Name="birthdate" Type="Edm.DateTime" Nullable="true"/>
      <Property Name="family" Type="Edm.String" Nullable="true"/>
      <Property Name="given" Type="Edm.String" Nullable="true"/>
      <Property Name="email" Type="Edm.String" Nullable="true"/>
      <Property Name="phone" Type="Edm.String" Nullable="true"/>
      <Property Name="address" Type="Edm.String" Nullable="true"/>
      <Property Name="city" Type="Edm.String" Nullable="true"/>
      <Property Name="state" Type="Edm.String" Nullable="true"/>
      <Property Name="zip" Type="Edm.String" Nullable="true"/>
      <Property Name="country" Type="Edm.String" Nullable="true"/>
      <Property Name="maritalStatus" Type="Edm.String" Nullable="true"/>
      <Property Name="managingOrganization" Type="com.philips.odata2.ODataDhp.Reference"/>
      <Property Name="active" Type="Edm.Boolean" Nullable="true"/>
    </EntityType>
    <EntityType Name="Observation">
      <Key>
        <PropertyRef Name="id"/>
      </Key>
      <Property Name="id" Type="Edm.String" Nullable="true"/>
      <Property Name="name" Type="Edm.String" Nullable="true"/>
      <Property Name="quantity" Type="Edm.String" Nullable="true"/>
      <Property Name="units" Type="Edm.String" Nullable="true"/>
      <Property Name="appliesDateTime" Type="Edm.DateTime" Nullable="true"/>
      <Property Name="appliesPeriodStart" Type="Edm.DateTime" Nullable="true"/>
      <Property Name="appliesPeriodEnd" Type="Edm.DateTime" Nullable="true"/>
      <Property Name="patientId" Type="Edm.String" Nullable="true"/>
      <Property Name="relatedObservationId1" Type="Edm.String" Nullable="true"/>
      <Property Name="relatedObservationId2" Type="Edm.String" Nullable="true"/>
      <Property Name="relatedObservationId3" Type="Edm.String" Nullable="true"/>
      <Property Name="status" Type="Edm.String" Nullable="true"/>
      <Property Name="reliability" Type="Edm.String" Nullable="true"/>
    </EntityType>
    <ComplexType Name="Reference">
      <Property Name="display" Type="Edm.String"/>
      <Property Name="reference" Type="Edm.String"/>
    </ComplexType>
    <EntityContainer Name="ODataDhpEntityContainer" m:IsDefaultEntityContainer="true">
      <EntitySet Name="Patients" EntityType="com.philips.odata2.ODataDhp.Patient"/>
      <EntitySet Name="Observations" EntityType="com.philips.odata2.ODataDhp.Observation"/>
    </EntityContainer>
  </Schema>
 </edmx:DataServices>
</edmx:Edmx>
```

You can read more about the OData 2.0 specification here:
http://www.odata.org/documentation/odata-version-2-0/

OData Query String Options

One of the key features when using OData is that it allows the client to pass in query parameters in the request URL that can perform actions similar to SQL queries. For example, a client can request to service to only return specific fields, to filter the results by one or more criteria, and to order the results by a particular column.

Currently the **Health**Suite OData product only supports a specific subset of these query string options which are listed below:

Parameter	**Purpose**	**Supported by HDSP OData?**
$filter	Allows the caller of the OData service to restrict the output of the feed list to be items that meet certain criteria. A $filter may contain one or more constraints and the "or" or "and" keywords can be used to indicate whether the results need to meet all or any of the criteria. In theory we should be able to filter by any field but as of now only certain fields are supported.	*Partially.* Some of the $filter operators are supported but only for specific fields in the objects. Refer to tables in the Patient and Observation section.
$inlinecount	When present in the request, adds a <count /> to the feed results which indicates *total number* of items exist that meet the $filter criteria (or all the items if there are no filters). This isn't the same as the number of items in the current feed result as the request may be using $top and $skip for pagination. The <count /> allows the consuming application to determine whether there are more results left on the server.	**Yes**
$orderby	When used this should result in the feed list being ordered by specific fields (e.g., by last name ascending).	*Partially.* Sorting is supported but only by specific fields. Refer to tables in the Patient and Observation section.
$select	When specified, only the listed fields will come back in the field results.	**Yes**
$skip	Used for pagination. The value of $skip should be an integer and it tells the service to skip over the first XX items in the feed (after any $filter criteria has been applied). For example, if you wanted to show five items per page and now you're on page 3 then $skip=10 would be used so the list is starting on the 11th item.	**Yes**

$top	Used for pagination, the $top tells the service to only return the first XX items in the list (after any filter criteria is applied and $skip is taken into account). The value is an integer and we expect the resulting feed list to be equal or less than the number. For example, if $top=5 is specified then no more than five items should come back.	**Yes**

You can find more information about the ODate 2.0 system query options (including example URLs) here:

http://www.odata.org/documentation/odata-version-2-0/uri-conventions#QueryStringOptions

OAuth 2.0

Access to the **Health**Suite OData service is controlled using OAuth 2.0. A full explanation about OAuth and how it works is beyond the scope of this document. There are some helpful resources online that explain what happens for each step in the OAuth flow but here is a high-level overview for the impatient.

1. Your application sends a request to https://gateway.api.pcftest.com:9004/v1/oauth/code to get an authorization code.
2. Your user (probably a developer or IT administrator) is prompted to give their consent for your application to access to access the organization's Patient and Observation data.
3. If the user consent is received then the **Health**Suite OData service responds with an authorization code.
4. Your application makes a subsequent call to https://gateway.api.pcftest.com:9004/v1/oauth/token asking to exchange the authorization code for an access token.
5. The **Health**Suite OData servers posts the token to your applications callback URL. For the sake of simplicity during the development phase, the access token has no expiration.
6. In subsequent requests, the access token is included in the requests to the **Health**Suite OData servers so they know the request is coming from your application and that it's allowed to see the data.

The expectation is that the **Health**Suite OData service is consumed by healthcare organizations which will have many users (physicians, nurses, assistants, et al). The access token is issued to the application rather than individual users of the application so these OAuth steps would only be completed once during the initial set-up of the consuming application by a developer or IT administrator. It's the responsibility of the consuming application (i.e., your application) to control which users have access to the patient data.

Here are some links regarding OAuth 2.0 for server to server scenarios:

https://help.salesforce.com/apex/HTViewHelpDoc?id=remoteaccess_oauth_web_server_flow.htm
https://developers.google.com/accounts/docs/OAuth2WebServer
https://aaronparecki.com/articles/2012/07/29/1/oauth2-simplified#web-server-apps

Patient

A Patient represents an individual in the healthcare database. Typically a patient will be associated with a particular healthcare organization (e.g., hospital, physician group, etc) but for the sample data all patients belong to the same Organization. Patients are accessed with the relative URI of "/Patients".

The the **Health**Suite OData product returns Patients in a feed and depending on the query sent by the consuming application, the service will return either a list of many, one, or zero patients in XML format.

```
<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
          xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
          xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
          xml:base="https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/">
 <id>https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/Patients</id>
 <title type="text">Patients</title>
 <updated>2015-02-25T09:31:11.507Z</updated>
 <author>
  <name/>
 </author>
 <link href="Patients" rel="self" title="Patients"/>
 <entry>
  <id>https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/Patients('a105')</id>
  <title type="text">Patients</title>
  <updated>2015-02-25T09:31:11.507Z</updated>
  <category term="com.philips.odata2.ODataDhp.Patient"
                    scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
  <link href="Patients('a105')" rel="edit" title="Patient"/>
  <content type="application/xml">
   <m:properties>
    <d:id>a105</d:id>
    <d:healthsuite-identifier>karen.young</d:healthsuite-identifier>
    <d:gender>Female</d:gender>
    <d:birthdate>1980-01-18T00:00:00</d:birthdate>
    <d:family>Young</d:family>
    <d:given>Karen</d:given>
    <d:email>karen.young@mail.com</d:email>
    <d:phone>(773) 122 2135</d:phone>
    <d:address>3rd Avenue</d:address>
    <d:city>Chicago</d:city>
    <d:state>IL</d:state>
    <d:zip>60601</d:zip>
    <d:country>US</d:country>
    <d:maritalStatus>unmarried</d:maritalStatus>
    <d:managingOrganization m:type="com.philips.odata2.ODataDhp.Reference">
     <d:display m:null="true"/>
     <d:reference>Organization/phr</d:reference>
    </d:managingOrganization>
```

```
      <d:active>true</d:active>
    </m:properties>
  </content>
</entry>
<entry>
  <id>https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/Patients('a103')</id>
  <title type="text">Patients</title>
  <updated>2015-02-25T09:31:11.507Z</updated>
  <category term="com.philips.odata2.ODataDhp.Patient"
                    scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
  <link href="Patients('a103')" rel="edit" title="Patient"/>
  <content type="application/xml">
    <m:properties>
      <d:id>a103</d:id>
      <d:healthsuite-identifier>charlie.miller</d:healthsuite-identifier>
      <d:gender>Male</d:gender>
      <d:birthdate>1970-02-05T00:00:00</d:birthdate>
      <d:family>Miller</d:family>
      <d:given>Charlie</d:given>
      <d:email>charlie.miller@mail.com</d:email>
      <d:phone>(510) 555 6113</d:phone>
      <d:address>123 Emerville St</d:address>
      <d:city>Hayward</d:city>
      <d:state>CA</d:state>
      <d:zip>94540</d:zip>
      <d:country>US</d:country>
      <d:maritalStatus>Married</d:maritalStatus>
      <d:managingOrganization m:type="com.philips.odata2.ODataDhp.Reference">
        <d:display m:null="true"/>
        <d:reference>Organization/phr</d:reference>
      </d:managingOrganization>
      <d:active>true</d:active>
    </m:properties>
  </content>
</entry>
</feed>
```

The Important Bits

Entry

Each <entry /> element in the feed represents a single object in the remote system.

properties

The properties element contains an array of individual fields for this object (Patient or Observation).

To learn more about the OData 2.0 specification you can access the following site:

http://www.odata.org/documentation/odata-version-2-0/

OData system query options for Patient

The following table details which fields in the Patient support the OData $filter and/or $orderby operations.

Patient field	**$filter operators**	**$orderby**
id	eq, or	
healthsuite-identifier	eq, and, or	asc, desc
gender		asc, desc
birthdate	gt, ge, lt, le	asc, desc
family	eq, and, or	asc, desc
given	eq, and, or	asc, desc
email		
phone		asc, desc
address		
city		
state		
zip		
country		
maritalStatus		
managingOrganization		
active		asc, desc

Observation

Observations are at the heart of the system. Any activity that produces a measurable result can be expressed as an observation. Observations are accessed with the relative URI of "/Observations".

As with Patients, the Observations are returned by the OData service as a feed that can include one, many, or zero entries.

```
<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
          xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
          xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
          xml:base="https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/">
  <id>https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/Observations</id>
  <title type="text">Observations</title>
  <updated>2015-02-25T09:51:29.056Z</updated>
  <author>
    <name/>
  </author>
  <link href="Observations" rel="self" title="Observations"/>
  <entry>
    <id>https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/Observations('65538')</id>
    <title type="text">Observations</title>
    <updated>2015-02-25T09:51:29.056Z</updated>
    <category term="com.philips.odata2.ODataDhp.Observation"
                    scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
    <link href="Observations('65538')" rel="edit" title="Observation"/>
    <content type="application/xml">
    <m:properties>
      <d:id>65538</d:id>
      <d:name>MDC_PAIN_LEVEL</d:name>
      <d:quantity>1.00</d:quantity>
      <d:units>pain level</d:units>
      <d:appliesDateTime>2014-02-28T22:32:24</d:appliesDateTime>
      <d:appliesPeriodStart m:null="true"/>
      <d:appliesPeriodEnd m:null="true"/>
      <d:patientId>a101</d:patientId>
      <d:relatedObservationId1 m:null="true"/>
      <d:relatedObservationId2 m:null="true"/>
      <d:relatedObservationId3 m:null="true"/>
      <d:status>registered</d:status>
      <d:reliability>ok</d:reliability>
    </m:properties>
  </content>
</entry>
<entry>
  <id>https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/Observations('11349')</id>
  <title type="text">Observations</title>
  <updated>2015-02-25T09:55:16.266Z</updated>
  <category term="com.philips.odata2.ODataDhp.Observation"
                    scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
  <link href="Observations('11349')" rel="edit" title="Observation"/>
  <content type="application/xml">
    <m:properties>
      <d:id>11349</d:id>
      <d:name>MDC_PRESS_BLD</d:name>
      <d:quantity m:null="true"/>
      <d:units m:null="true"/>
      <d:appliesDateTime>2014-01-03T20:19:21</d:appliesDateTime>
      <d:appliesPeriodStart m:null="true"/>
      <d:appliesPeriodEnd m:null="true"/>
      <d:patientId>a103</d:patientId>
      <d:relatedObservationId1>11340</d:relatedObservationId1>
      <d:relatedObservationId2>11343</d:relatedObservationId2>
      <d:relatedObservationId3>11346</d:relatedObservationId3>
      <d:status>registered</d:status>
      <d:reliability>ok</d:reliability>
    </m:properties>
  </content>
</entry>
<entry>
  <id>https://gateway.api.pcftest.com:9004/v1/fhir_odata/FHIROData.svc/Observations('16083')</id>
  <title type="text">Observations</title>
  <updated>2015-02-25T09:57:23.963Z</updated>
  <category term="com.philips.odata2.ODataDhp.Observation"
                    scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
  <link href="Observations('16083')" rel="edit" title="Observation"/>
  <content type="application/xml">
    <m:properties>
      <d:id>16083</d:id>
      <d:name>MDC_HF_DISTANCE</d:name>
      <d:quantity>9873.00</d:quantity>
      <d:units>MDC_DIM_STEP</d:units>
      <d:appliesDateTime m:null="true"/>
      <d:appliesPeriodStart>2014-05-09T15:00:00</d:appliesPeriodStart>
      <d:appliesPeriodEnd>2014-05-10T15:00:00</d:appliesPeriodEnd>
      <d:patientId>a105</d:patientId>
      <d:relatedObservationId1 m:null="true"/>
      <d:relatedObservationId2 m:null="true"/>
      <d:relatedObservationId3 m:null="true"/>
      <d:status>registered</d:status>
      <d:reliability>ok</d:reliability>
    </m:properties>
  </content>
  </entry>
</feed>
```

The Important Bits

patientId

The patientId field is what links an Observation to the Patient (as the name implies, by the Patient's ID). If you're building an application you may want to display only the Observations for a particular Patient. You can do a $filter operation on the patientId field to only get back those Observations you need.

relatedObservationId#

There are three fields in the Observation records (relatedObservationId1, relatedObservationId2, and relatedObservationId3) which can be used to reference other related (child) Observations. This is specifically used by the blood pressure (MDC_PRESS_BLD) Observations to refer to three other Observations that hold the components of the blood pressure. For Observation types that don't have child Observations these fields will be null.

Observation Coding

The type of observation is indicated by its coding in the "name" element; the value of the name corresponds to the "Display" code. The system indicates which code system is being used to identify the observation. The code systems themselves can be complex, with generalized codes divided into more specific sub-codes, and many hospitals have dedicated terminology systems to sort all of this out. Fortunately, we have included the codes for common observations for two major code systems which will be used in our test data.

More information on terminology code systems can be found here:
http://www.hl7.org/implement/standards/fhir/terminologies-systems.html

The following tables can be used to populate queries for some common observation types.

IEEE-11073 Codes (Code system https://rtmms.nist.gov)

ISO/IEEE 11073 is an identifier coding system designed to bridge the gap between health and wellness systems. It includes standard medical measurements such as weight, glucose, and blood pressure, as well as fitness activities, step counters, and sleep monitors.

You can search for more IEEE-11073 codes using the online database at
https://rtmms.nist.gov/rtmms/index.htm#!rosetta

Common IEEE-11073 Codes (https://rtmms.nist.gov)

Name	**Code**	**Display**
Weight	188736	MDC_MASS_BODY_ACTUAL
Systolic Blood Pressure	150017	MDC_PRESS_BLD_SYS
Diastolic Blood Pressure	150018	MDC_PRESS_BLD_DIA
Mean Blood Pressure	150019	MDC_PRESS_BLD_MEAN
Blood Pressure (all 3 components together)	150016	MDC_PRESS_BLD
Steps	8454247	MDC_HF_DISTANCE
Glucose	160364	MDC_CONC_GLU_UNDETERMINED_WHOLEBLOOD
Body Temperature	150364	MDC_TEMP_BODY
Heart Rate	8454258	MDC_HF_HR
Respiratory Rate	151562	MDC_RESP_RATE
Mood	67108865	MDC_PHYSIO_MOOD
Pain Level	67108866	MDC_PAIN_LEVEL
Energy Expended	8454263	MDC_ HF_ENERGY
Glycated Hemoglobin	160220	MDC_CONC_HBA1C
Prothrombin Time	160260	MDC_RATIO_INR_COAG
Pulse Rate (taken with blood pressure)	149546	MDC_PULS_RATE_NON_INV
Pulse Rate (taken with oximeter)	149530	MDC_PULS_OXIM_PULS_RATE
Sleep	8455148	MDC_HF_ACT_SLEEP
Sleep Efficiency	67108866	MDC_SLEEP_EFFICIENCY
SPO2 Oxygen	150456	MDC_PULS_OXIM_SAT_O2

OData system query options for Observation

The following table details which fields in the Observation support the OData $filter and/or $orderby operations.

Observation field	**$filter operators**	**$orderby**
id	eq, or	
name	eq, and, or	
quantity		asc, desc
units		
appliesDateTime	gt, ge, lt, le, and	asc, desc
appliesPeriodStart		asc, desc
appliesPeriodEnd		asc, desc
patientId	eq, or	
relatedObservationId1		
relatedObservationId2		
relatedObservationId3		
status		asc, desc
reliability		asc, desc

NOTE: There are some Observation types that have a single timestamp (appliesDateTime) and some that have a date range (appliesPeriodStart -> appliesPeriodEnd) but they won't have both. If you filter for a date range using the appliesDateTime field then you'll get back the Observations where either the appliesDateTime **or** the appliesPeriodStart -> appliesPeriodEnd is within the requested range.

Recipes

While OData can work with many languages and platforms including Java and JavaScript, the focus of this cookbook will be on using OData from the Salesforce1 platform.

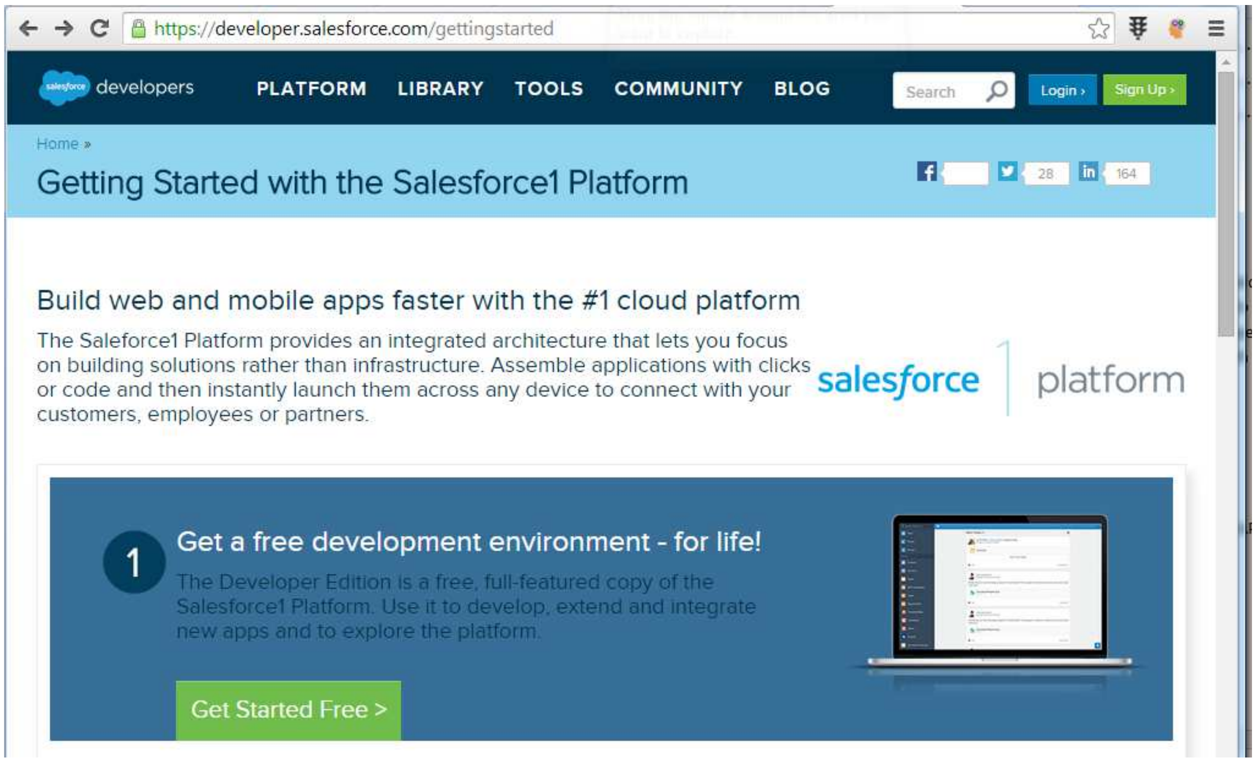
Salesforce1 Platform

Salesforce1 has native support for OData using the External Data Source feature. A Salesforce developer can create a new source pointing to an OData service and then choose which object(s) they want to make available in their Salesforce1 application. Once the external objects are selected, the developer can use these in a manner similar to native Salesforce1 custom objects: they can be associated with tabs, page layouts, or retrieved via SOQL queries.

Request a Salesforce1 developer environment

You can get started on the Salesforce1 platform by signing up for a free Developer Edition. The Developer Edition (DE) has all the features you'd need to work with the Philips **Health**Suite OData API.

Begin by going to the "getting started" section of the Salesforce developer web site.
https://developer.salesforce.com/gettingstarted



Getting Started with the Salesforce1 Platform

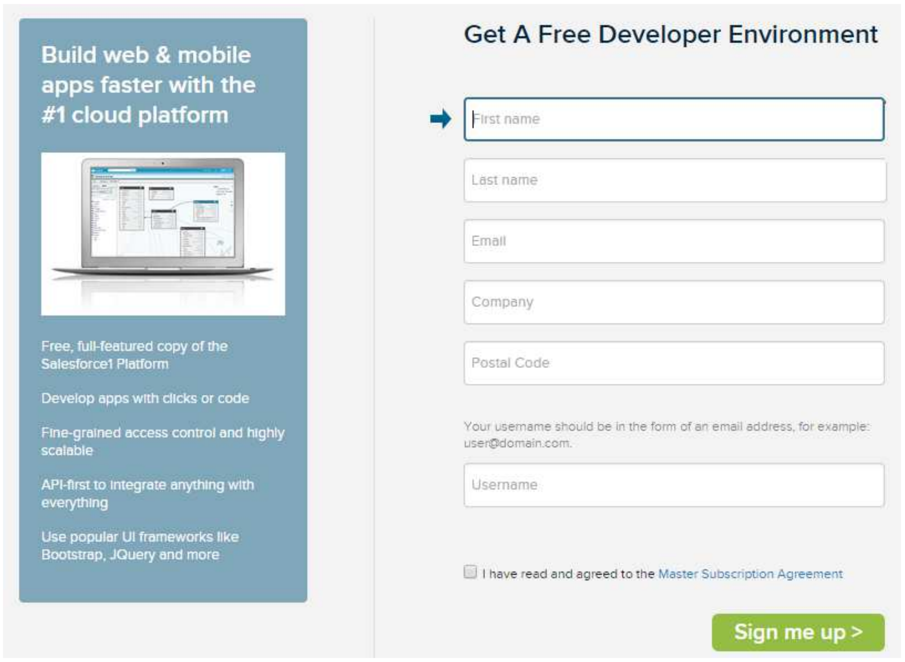
Build web and mobile apps faster with the #1 cloud platform

The Saleforce1 Platform provides an integrated architecture that lets you focus on building solutions rather than infrastructure. Assemble applications with clicks or code and then instantly launch them across any device to connect with your customers, employees or partners.

1 Get a free development environment - for life!

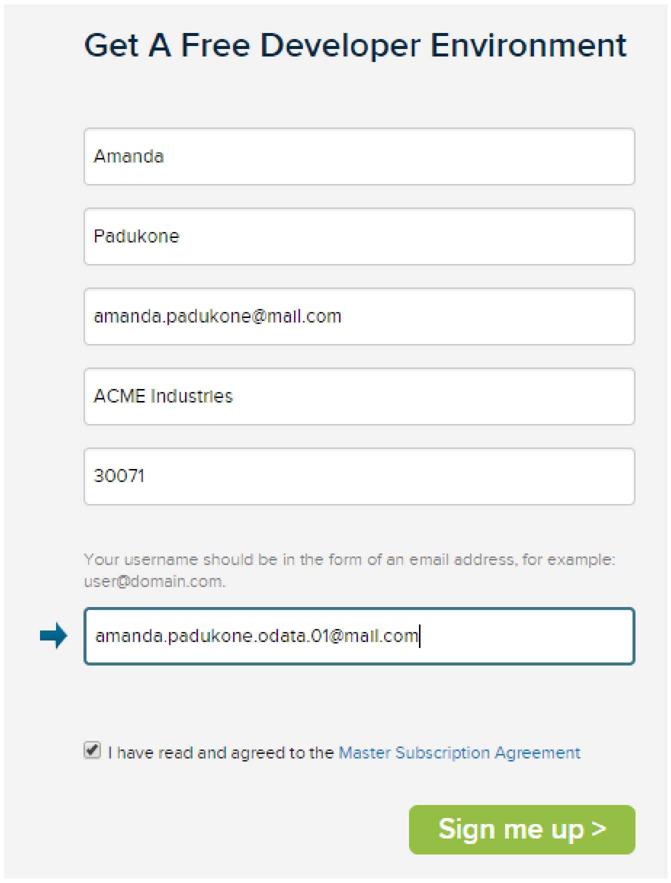
The Developer Edition is a free, full-featured copy of the Salesforce1 Platform. Use it to develop, extend and integrate new apps and to explore the platform.

Clicking on the "Get Started Free >" button will take you to the sign-up form for a Developer Edition account.
https://developer.salesforce.com/signup


Build web & mobile apps faster with the #1 cloud platform
Free, full-featured copy of the Salesforce1 Platform
Develop apps with clicks or code
Fine-grained access control and highly scalable
API-first to integrate anything with everything
Use popular UI frameworks like Bootstrap, JQuery and more
Get A Free Developer Environment
First name
Last name
Email
Company
Postal Code
Your username should be in the form of an email address, for example: user@domain.com.
Username
I have read and agreed to the Master Subscription Agreement
Sign me up >

A few items to keep in mind when completing the form. The “Email” field must be a valid e-mail you have access to because as part of the registration process you’ll receive an e-mail with a link to your new org.

Even though the “Username” field must match the *format* of an e-mail address, it does not have to be an actual e-mail address. Because Salesforce1 is a public cloud platform the username must be unique across all users in all Salesforce organizations. Your e-mail address would be unique but in case you want to create additional Developer Edition orgs in the future you should add something to it. For example, if your e-mail is “Amanda.Padukone@mail.com” then you could make your username “Amanda.Padukone.OData.01@mail.com”. This way if you want to create another Developer Edition then you could use “Amanda.Padukone.OData.02@mail.com” and so forth.



Get A Free Developer Environment

Amanda
Padukone
amanda.padukone@mail.com
ACME Industries
30071

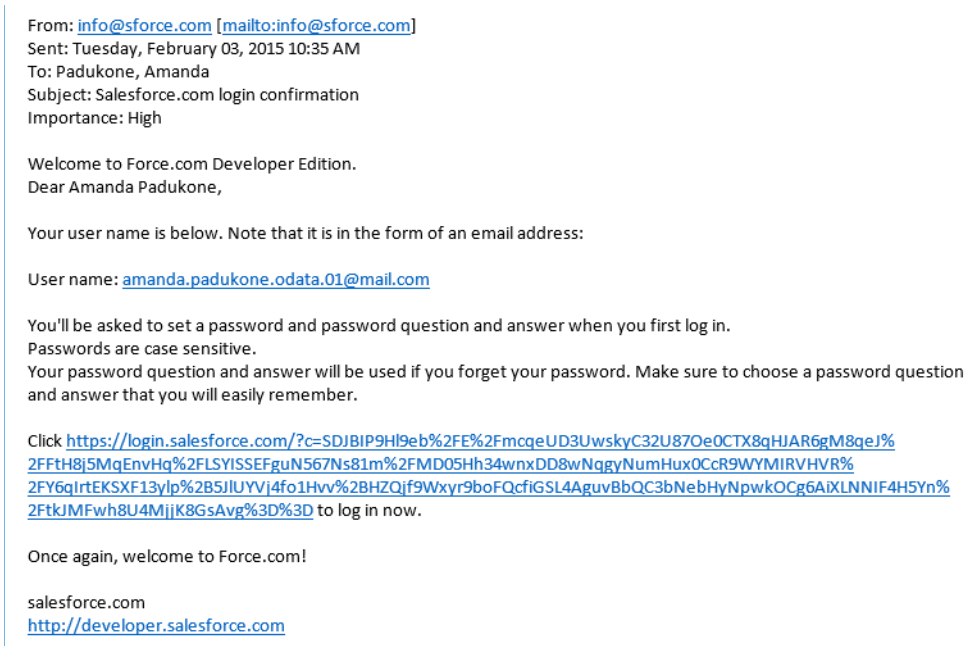
Your username should be in the form of an email address, for example: user@domain.com.

amanda.padukone.odata.01@mail.com

[✓] I have read and agreed to the Master Subscription Agreement

You'll notice there is no place to specify a password. That step comes later in the process.

After you submit the form, you'll receive an e-mail message containing a link. Clicking the link will take you to a form where you can set your password and after that your Developer Edition org will be ready for use.


From: info@sforce.com [mailto:info@sforce.com]
Sent: Tuesday, February 03, 2015 10:35 AM
To: Padukone, Amanda
Subject: Salesforce.com login confirmation
Importance: High

Welcome to Force.com Developer Edition.
Dear Amanda Padukone,

Your user name is below. Note that it is in the form of an email address:

User name: amanda.padukone.odata.01@mail.com

You'll be asked to set a password and password question and answer when you first log in.
Passwords are case sensitive.
Your password question and answer will be used if you forget your password. Make sure to choose a password question and answer that you will easily remember.

Click https://login.salesforce.com/?c=SDJBIP9Hl9eb%2FE%2FmcqeUD3UwskyC32U87Oe0CTX8qHJAR6gM8qeJ%2FFtH8j5MqEnvHq%2FLSYISSEFguN567Ns81m%2FMD05Hh34wnxDD8wNqgyNumHux0CcR9WYMIRVHVR%2FY6qIrtEKSXF13ylp%2B5JlUYVj4fo1Hvv%2BHZQjf9Wxyr9boFQcfiGSL4AguvBbQC3bNebHyNpwkOCg6AiXLNNIF4H5Yn%2FtkJMFwh8U4MjjK8GsAvg%3D%3D to log in now.

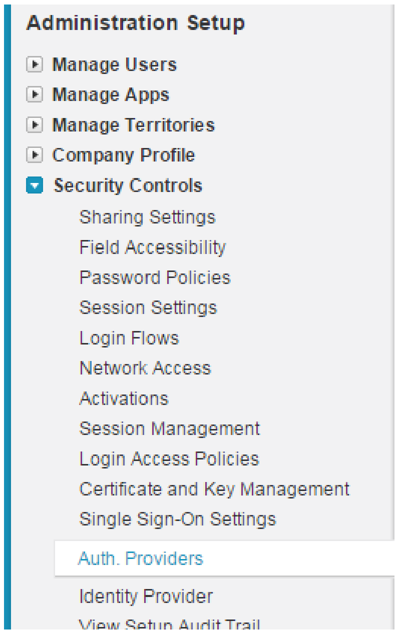
Once again, welcome to Force.com!

salesforce.com
http://developer.salesforce.com

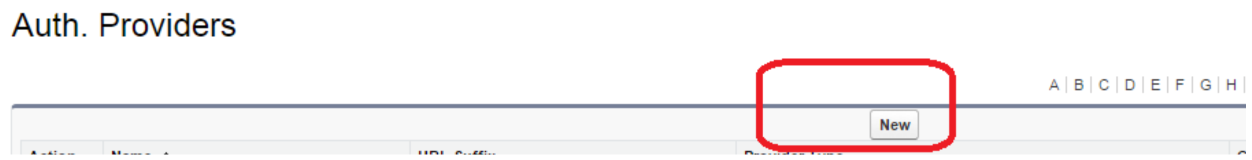
Creating an Authentication Provider

The Philips OData service utilizes OAuth 2 to control access. Before creating an External Data Source you need to create an Authentication Provider in Salesforce.

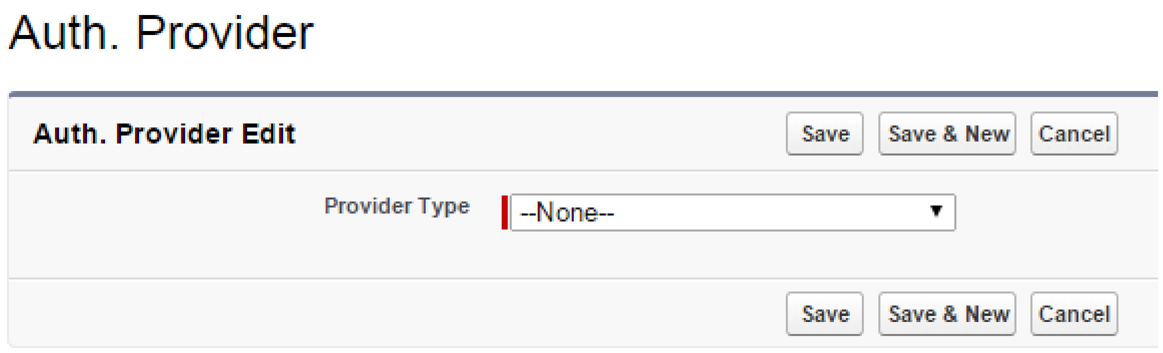
Start by logging into your Salesforce development org and go to the setup console. Under "Administration Setup" navigate to "Security Controls" -> "Auth. Providers".



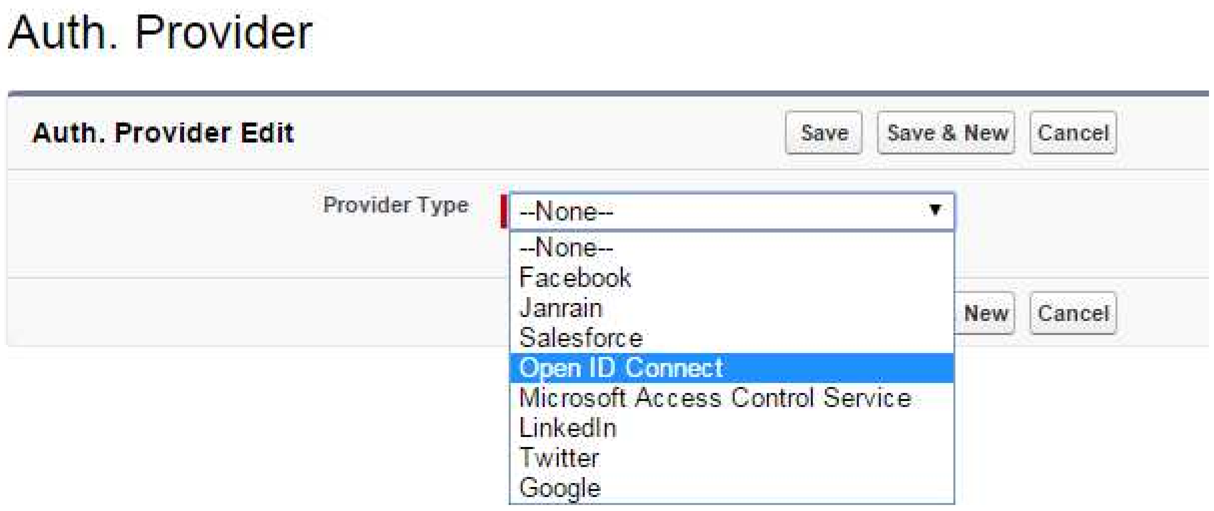
From the “Auth. Providers” list view, click on the “New” button.



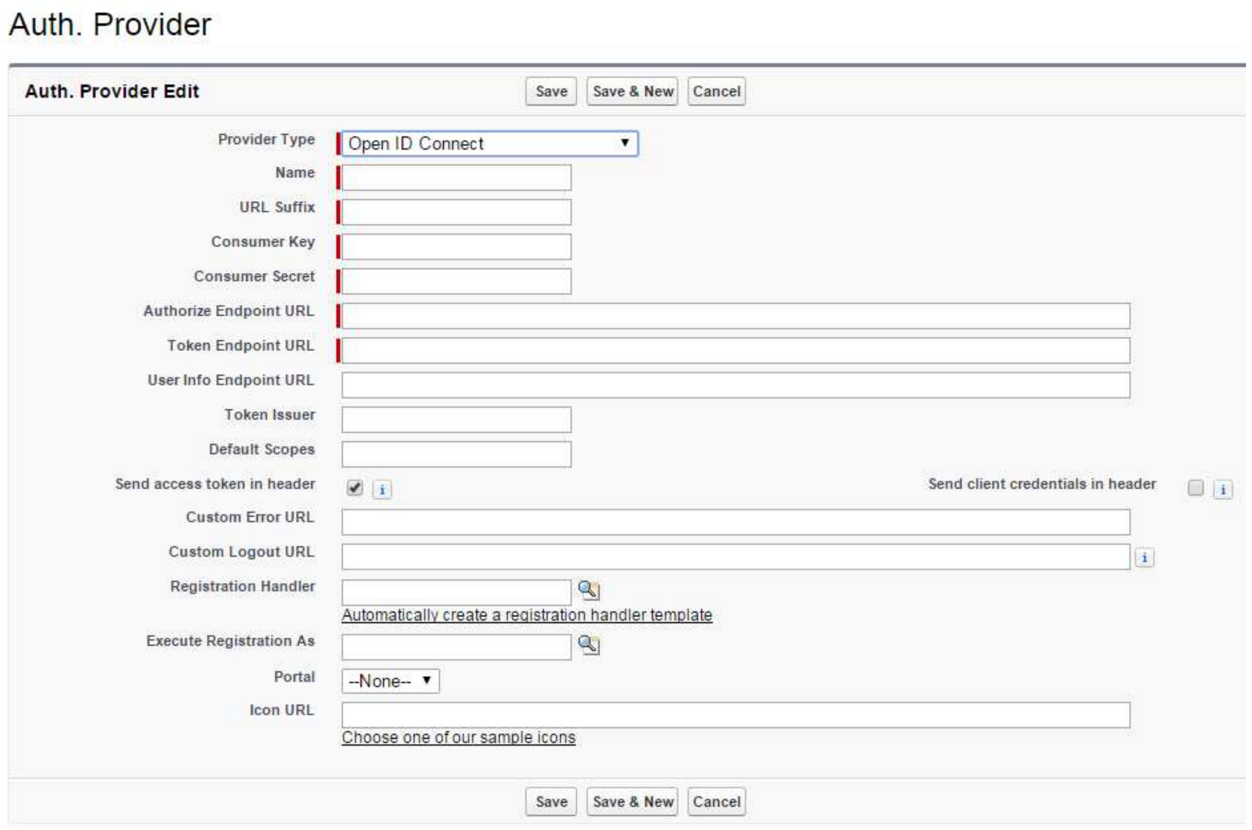
The form for creating a new authentication provider will display.



From the Provider Type drop-down, choose “Open ID Connect”.



The form will refresh to display the inputs relevant for the provider type you selected.


Auth. Provider

Auth. Provider Edit

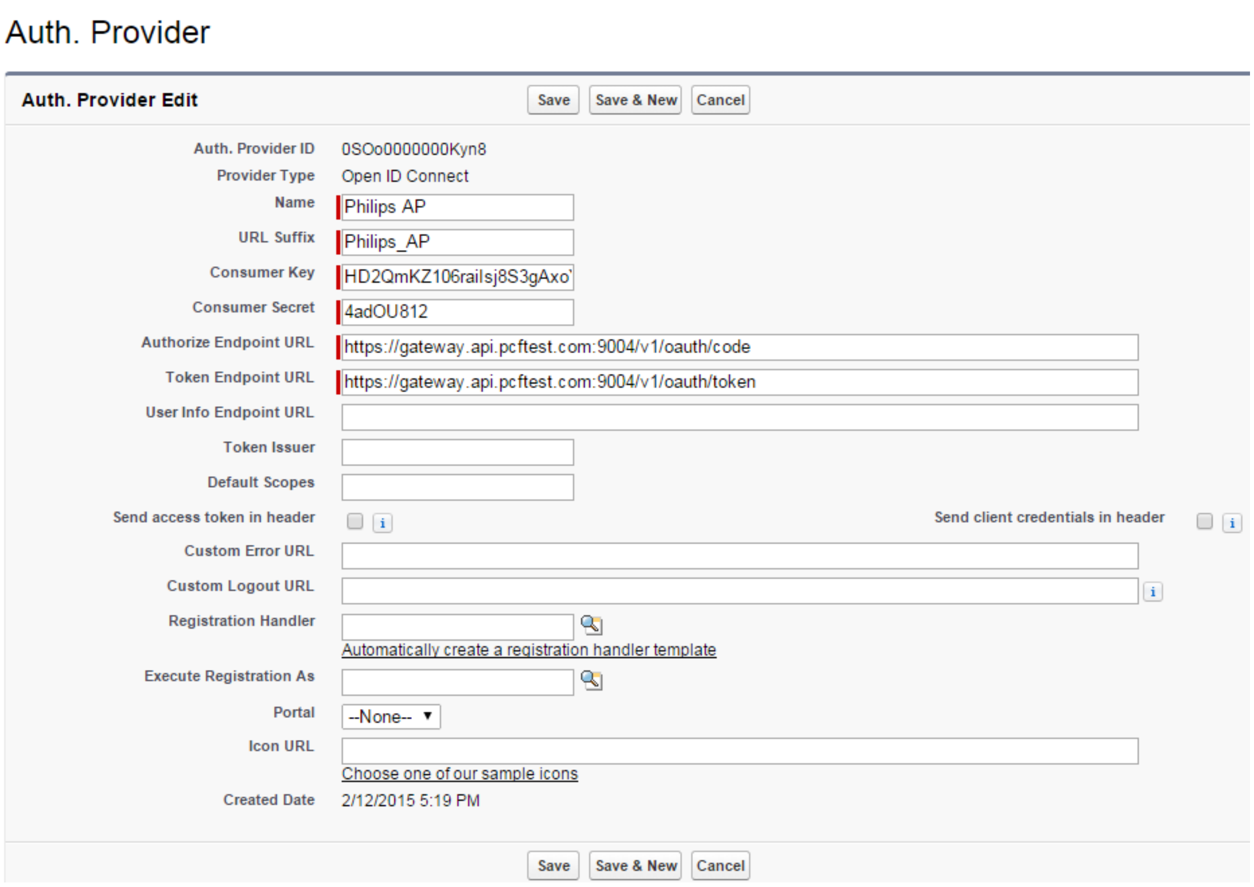
Provider Type	Open ID Connect
Name	
URL Suffix	
Consumer Key	
Consumer Secret	
Authorize Endpoint URL	
Token Endpoint URL	
User Info Endpoint URL	
Token Issuer	
Default Scopes	
Send access token in header	✓
Send client credentials in header	☐
Custom Error URL	
Custom Logout URL	
Registration Handler	Automatically create a registration handler template
Execute Registration As	
Portal	--None--
Icon URL	Choose one of our sample icons

Give the new authentication provider a name of your choosing, the URL Suffix field will auto-populate based on the name.
To complete this form you will need to know the key and secret from you or your team's OData app in the Philips developer portal. Those will go into the "Consumer Key" and "Consumer Secret" fields respectively.

Use the following values for the required OAuth URLs:

Authorize Endpoint URL:	https://gateway.api.pcftest.com:9004/v1/oauth/code
Token Endpoint URL:	https://gateway.api.pcftest.com:9004/v1/oauth/token

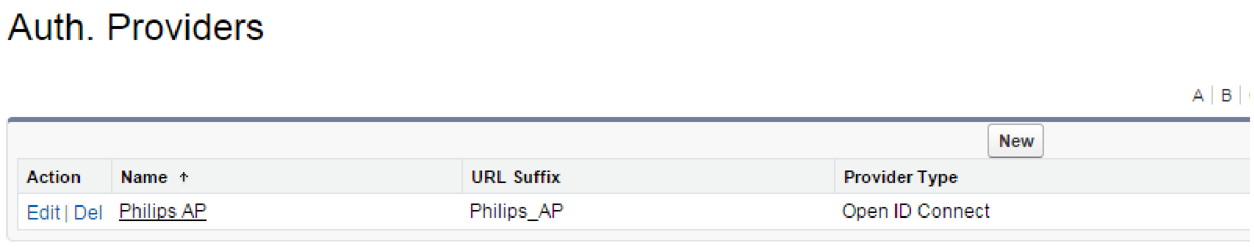
Be sure that the "Send access token in header" and "Send client credentials in header" boxes are both **unchecked**.


Auth. Provider

Auth. Provider Edit

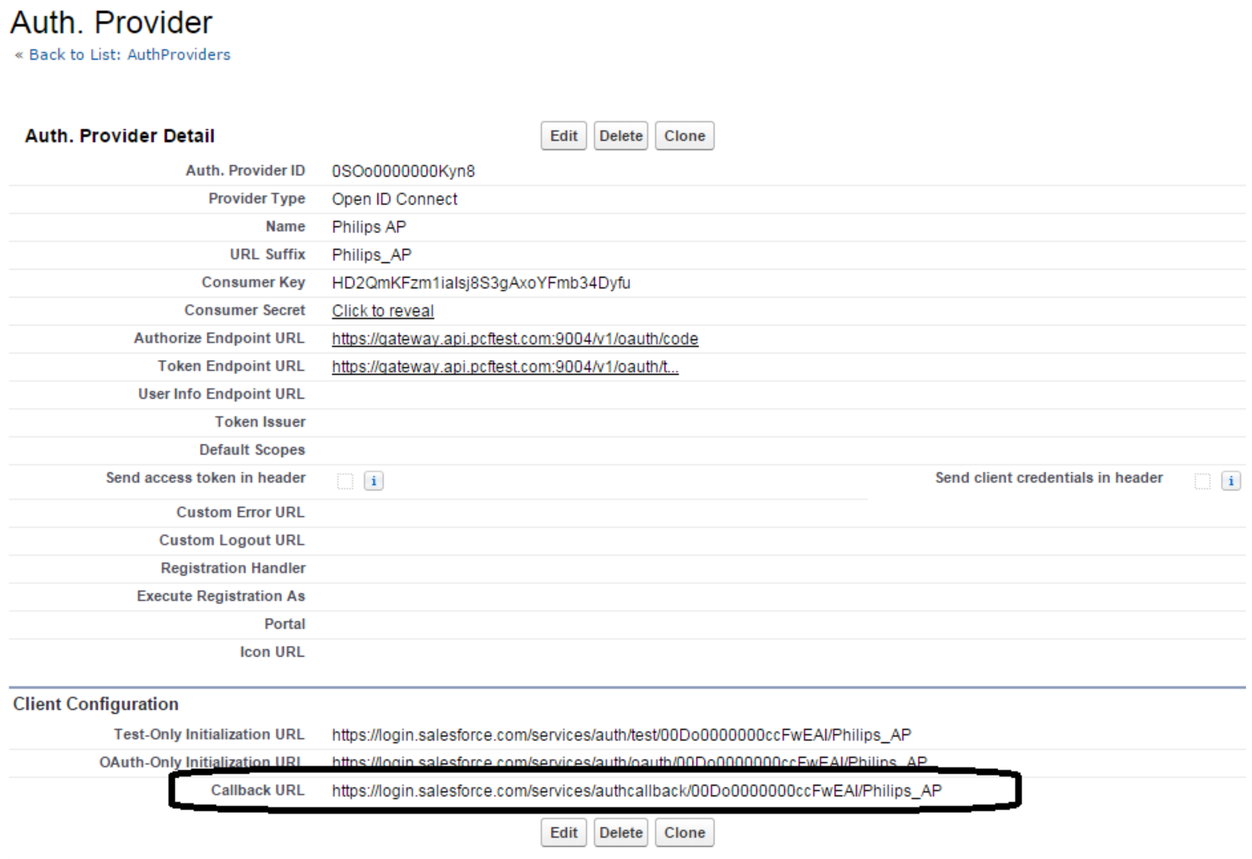
Auth. Provider ID	0SOo0000000Kyn8
Provider Type	Open ID Connect
Name	Philips AP
URL Suffix	Philips_AP
Consumer Key	HD2QmKZ106railsj8S3gAxo'
Consumer Secret	4adOU812
Authorize Endpoint URL	https://gateway.api.pcftest.com:9004/v1/oauth/code
Token Endpoint URL	https://gateway.api.pcftest.com:9004/v1/oauth/token
User Info Endpoint URL	
Token Issuer	
Default Scopes	
Send access token in header	☐
Send client credentials in header	☐
Custom Error URL	
Custom Logout URL	
Registration Handler	Automatically create a registration handler template
Execute Registration As	
Portal	--None--
Icon URL	Choose one of our sample icons
Created Date	2/12/2015 5:19 PM

When you're done, click the "Save" button and you should see your new auth provider in the list view.


Auth. Providers

Action	Name ↑	URL Suffix	Provider Type
Edit | Del	Philips AP	Philips_AP	Open ID Connect

Clicking on the name of your auth provider in the list will take you to the detail view. It's important that you copy the value of the "Callback URL" and paste that into your OData app in the Philips developer portal.



Auth. Provider

« Back to List: AuthProviders

Auth. Provider Detail

Auth. Provider ID	0SOo0000000Kyn8
Provider Type	Open ID Connect
Name	Philips AP
URL Suffix	Philips_AP
Consumer Key	HD2QmKFzm1ialsj8S3gAxoYFmb34Dyfu
Consumer Secret	Click to reveal
Authorize Endpoint URL	https://gateway.api.pcftest.com:9004/v1/oauth/code
Token Endpoint URL	https://gateway.api.pcftest.com:9004/v1/oauth/t...
User Info Endpoint URL	
Token Issuer	
Default Scopes	
Send access token in header	☐
Send client credentials in header	☐
Custom Error URL	
Custom Logout URL	
Registration Handler	
Execute Registration As	
Portal	
Icon URL	

Client Configuration

Test-Only Initialization URL	https://login.salesforce.com/services/auth/test/00Do0000000ccFwEAI/Philips_AP
OAuth-Only Initialization URL	https://login.salesforce.com/services/auth/oauth/00Do0000000ccFwEAI/Philips_AP
Callback URL	https://login.salesforce.com/services/authcallback/00Do0000000ccFwEAI/Philips_AP

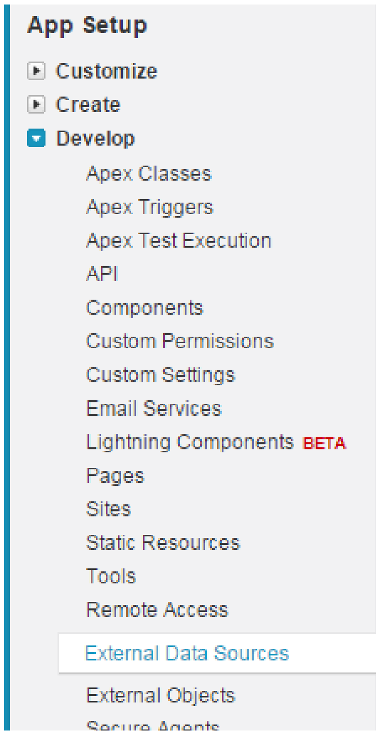
Optionally you can test your new auth provider before moving on to the next step. To do this, copy the value of the "Test-Only Initialization URL" from the auth provider detail page and paste that into a browser address field. If everything is configured properly then you'll get back an XML document listing some user attributes.

```
<user>
  <org_id>00Do0000000ccFw</org_id>
  <portal_id>000000000000000</portal_id>
</user>
```

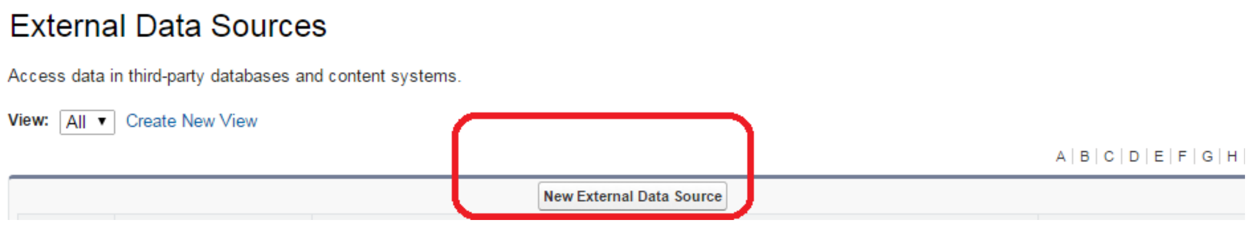
Creating an External Data Source

Once you have an Authentication Provider, the next step is to create an External Data Source.

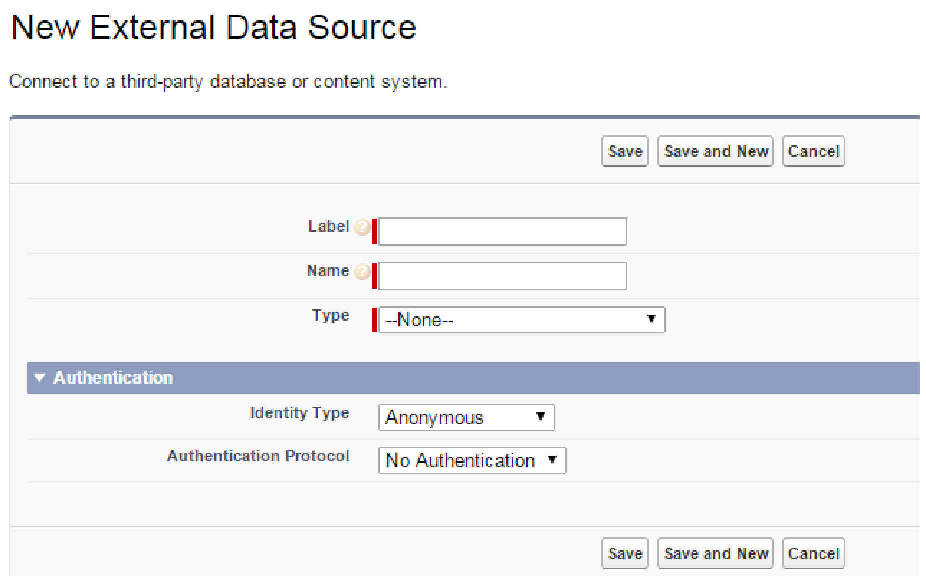
From the Salesforce setup console, go to “App Setup” and then navigate to “Develop” -> “External Data Sources”.



From the External Data Sources list view, click the “New External Data Source” button.


External Data Sources
Access data in third-party databases and content systems.
View: All
Create New View
A | B | C | D | E | F | G | H |
New External Data Source

The New External Data Source form should display.


New External Data Source

Connect to a third-party database or content system.

Label	
Name	
Type	--None--

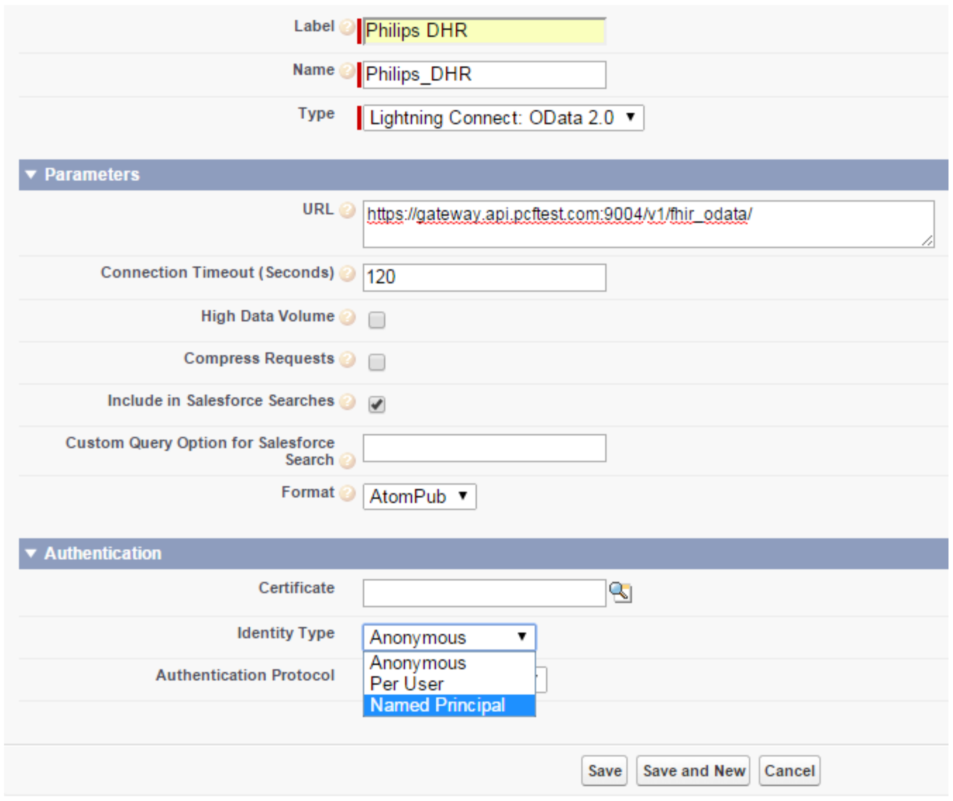
Authentication

Identity Type	Anonymous
Authentication Protocol	No Authentication

Give the new data source a name and label of your choosing. For the Type field, be sure to choose "Lightning Connect: OData 2.0".

In the Parameters section, specify "https://gateway.api.pcftest.com:9004/v1/fhir_odata/" as the URL. Accept the defaults for the other fields.

In the Authentication section, choose “Named Principal” for the Identity Type.



Label	Philips DHR
Name	Philips_DHR
Type	Lightning Connect: OData 2.0

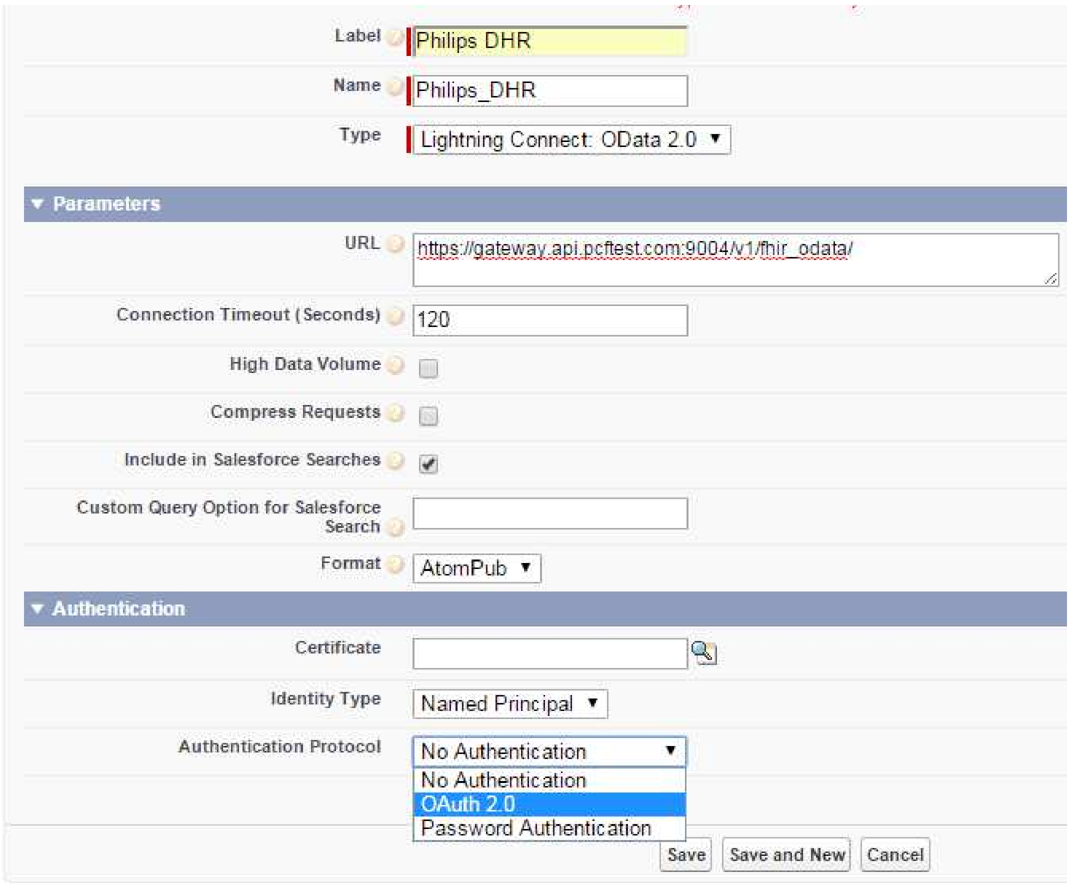
Parameters

URL	https://gateway.api.pcftest.com:9004/v1/fhir_odata/
Connection Timeout (Seconds)	120
High Data Volume	☐
Compress Requests	☐
Include in Salesforce Searches	✓
Custom Query Option for Salesforce Search	
Format	AtomPub

Authentication

Certificate	
Identity Type	Anonymous
Authentication Protocol	

Continuing in the Authorization section, choose "OAuth 2.0" for the Authentication Protocol.



Label	Philips DHR
Name	Philips_DHR
Type	Lightning Connect: OData 2.0

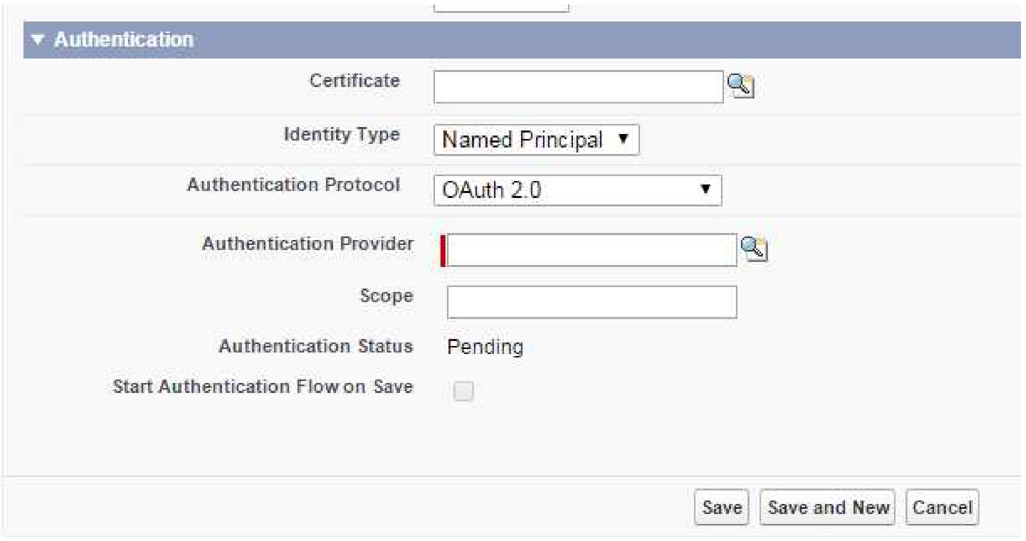
Parameters

URL	https://gateway.api.pcftest.com:9004/v1/fhir_odata/
Connection Timeout (Seconds)	120
High Data Volume	☐
Compress Requests	☐
Include in Salesforce Searches	✓
Custom Query Option for Salesforce Search	
Format	AtomPub

Authentication

Certificate	
Identity Type	Named Principal
Authentication Protocol	No Authentication

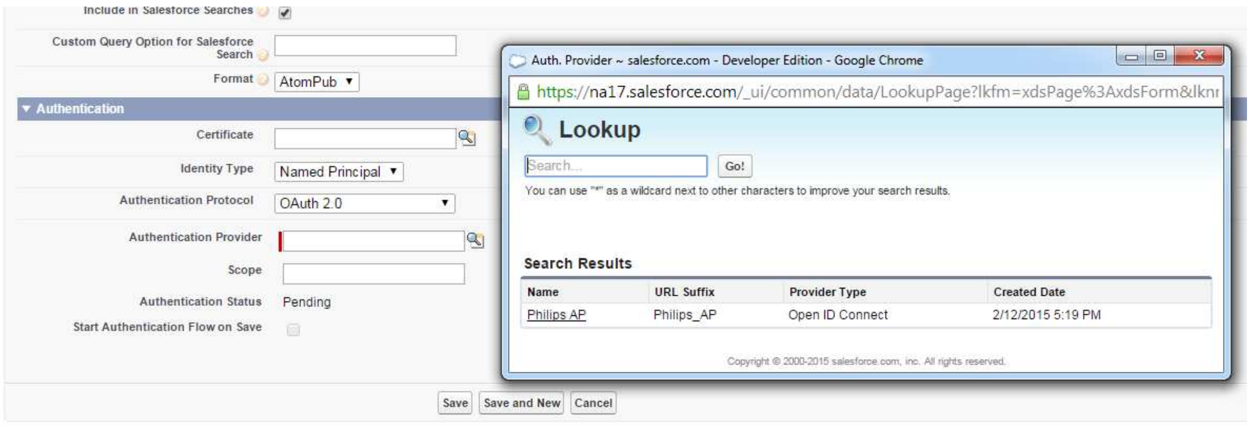
After choosing the Authentication Protocol, the form should update to reveal a new input named Authentication Provider. Click on the magnifying glass icon next to Authentication Provider.



Authentication

Certificate	
Identity Type	Named Principal
Authentication Protocol	OAuth 2.0
Authentication Provider	
Scope	
Authentication Status	Pending
Start Authentication Flow on Save	☐

A window will open with a list of the available authentication providers. Choose the auth provider you created in the previous recipe.


Include in Salesforce Searches: ✓
Custom Query Option for Salesforce Search:
Format: AtomPub
Authentication
Certificate:
Identity Type: Named Principal
Authentication Protocol: OAuth 2.0
Authentication Provider:
Scope:
Authentication Status: Pending
Start Authentication Flow on Save:

Auth. Provider ~ salesforce.com - Developer Edition - Google Chrome
https://na17.salesforce.com/_ui/common/data/LookupPage?lkfm=xdsPage%3AxdsForm&lknı

Lookup

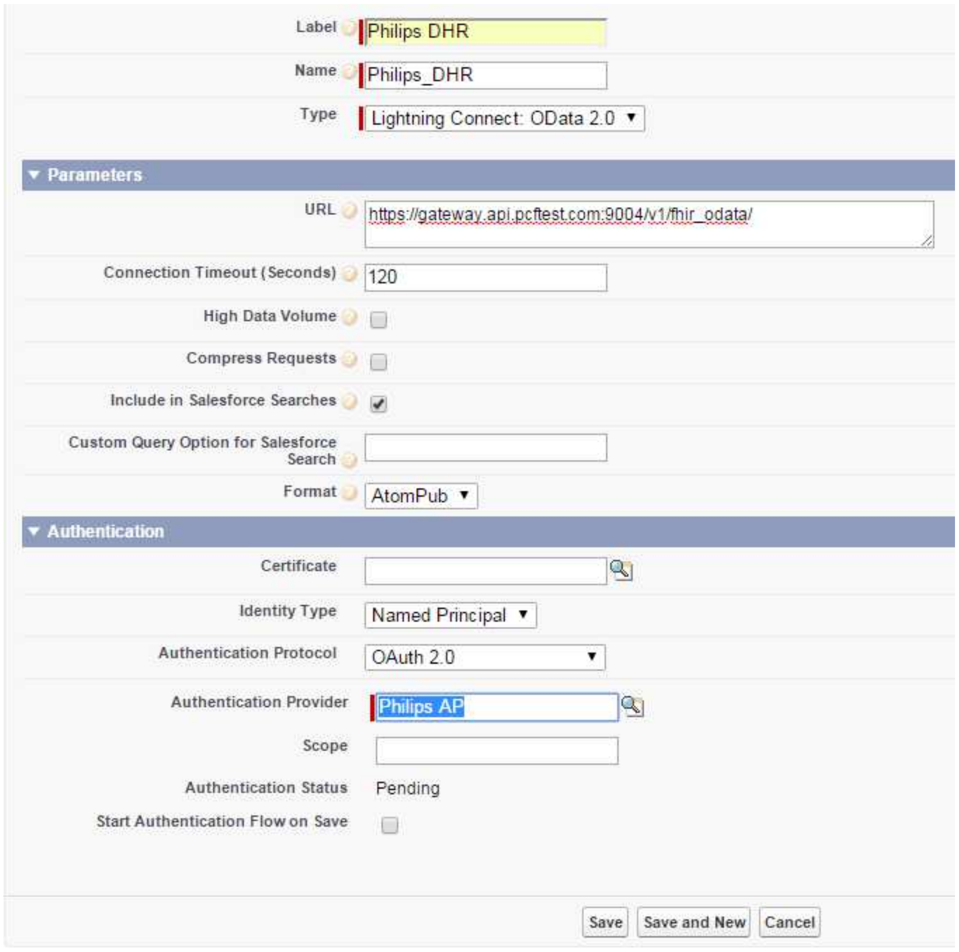
You can use "*" as a wildcard next to other characters to improve your search results.

Search Results

Name	URL Suffix	Provider Type	Created Date
Philips AP	Philips_AP	Open ID Connect	2/12/2015 5:19 PM

Copyright © 2000-2015 salesforce.com, inc. All rights reserved.

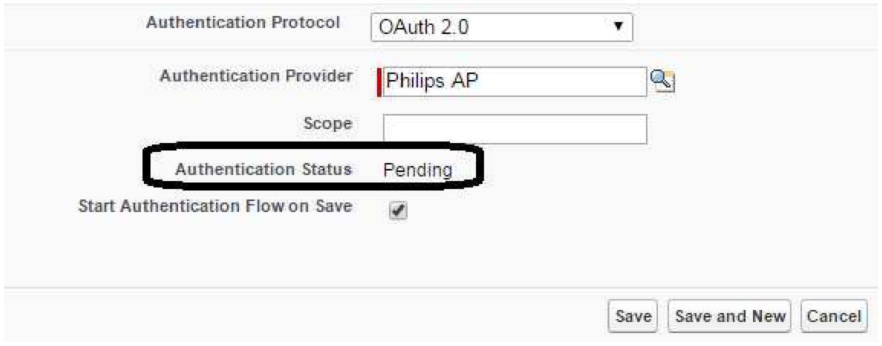
The completed External Data Source form should look something like this:


Label: Philips DHR
Name: Philips_DHR
Type: Lightning Connect: OData 2.0

Parameters
URL: https://gateway.api.pcftest.com:9004/v1/fhir_odata/
Connection Timeout (Seconds): 120
High Data Volume:
Compress Requests:
Include in Salesforce Searches: ✓
Custom Query Option for Salesforce Search:
Format: AtomPub

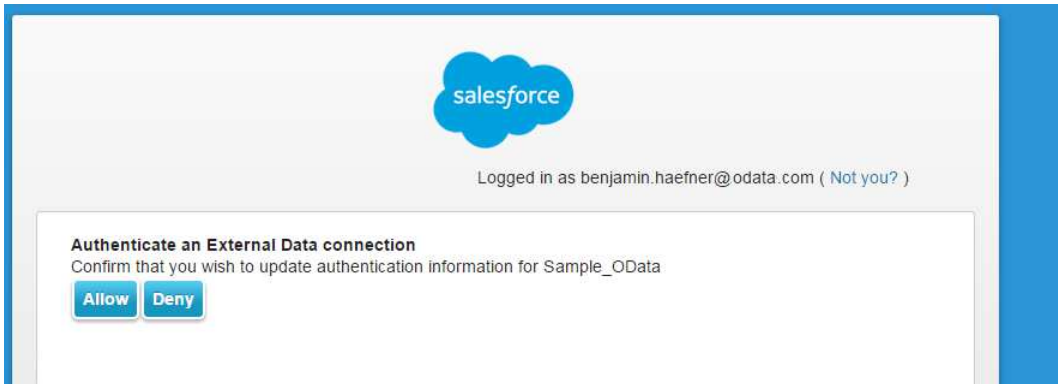
Authentication
Certificate:
Identity Type: Named Principal
Authentication Protocol: OAuth 2.0
Authentication Provider: Philips AP
Scope:
Authentication Status: Pending
Start Authentication Flow on Save:

You can click the "Save" button now but eventually you'll need to authenticate this data source. Initially the value of the Authentication Status is "Pending". To trigger Salesforce to authenticate the data source, check the "Start Authentication Flow on Save" box before clicking the "Save" button.



Authentication Protocol	OAuth 2.0
Authentication Provider	Philips AP
Scope	
Authentication Status	Pending
Start Authentication Flow on Save	✓

If everything was configured properly, Salesforce will initiate the OAuth "dance" with the Philips API server and you'll be presented with a screen similar to the one before asking you whether you want to allow the external data source to access the Philips OData service on your behalf. Click the "Allow" button.

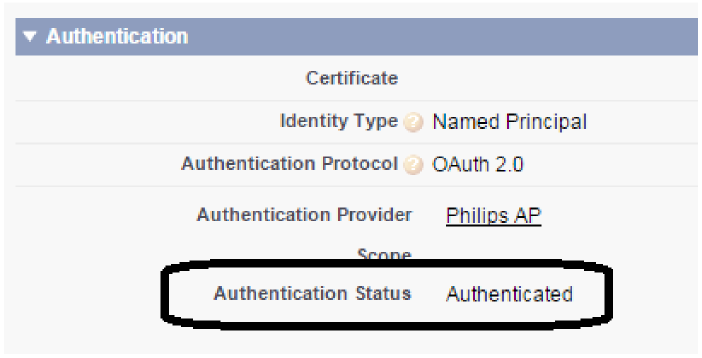


Logged in as benjamin.haefner@odata.com (Not you?)

Authenticate an External Data connection

Confirm that you wish to update authentication information for Sample_OData

If all goes well, you'll be redirected back to the External Data Source detail view and if you scroll down you'll see that the Authentication Status is now "Authenticated". If not then you'll need to troubleshoot your OAuth configuration.



Authentication

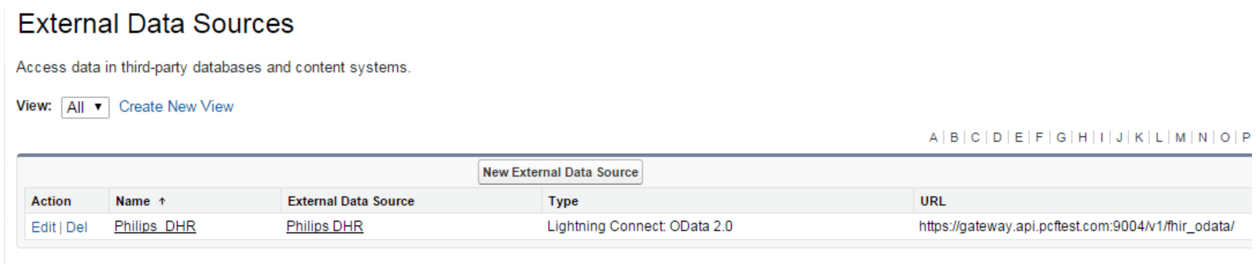
Certificate	
Identity Type	Named Principal
Authentication Protocol	OAuth 2.0
Authentication Provider	Philips AP
Scope	
Authentication Status	Authenticated

You're now done setting up your External Data Source and are ready to generate the external objects.

Syncing External Objects

After creating a new External Data Source for OData, the next step is generate the External Objects in Salesforce. External Objects are similar to Salesforce custom objects except under the covers they're mapped to the fields in the objects that come back from the OData service.

Go to the setup console in your Salesforce org and navigate to the External Data Sources list.



External Data Sources

Access data in third-party databases and content systems.

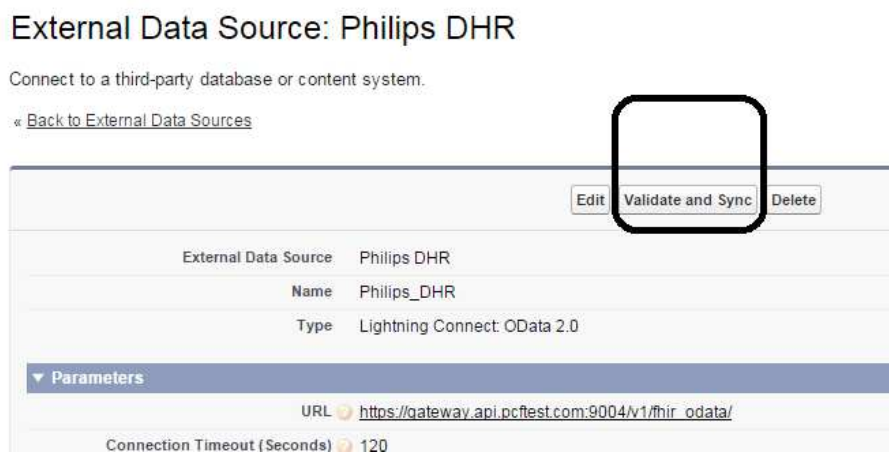
View: All Create New View

A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P

New External Data Source

Action	Name ↑	External Data Source	Type	URL
Edit | Del	Philips_DHR	Philips DHR	Lightning Connect: OData 2.0	https://gateway.api.pcftest.com:9004/v1/fhir_odata/

Click on the name of your External Data Source to be taken to the detail view. From there, click on the "Validate and Sync" button.



External Data Source: Philips DHR

Connect to a third-party database or content system.

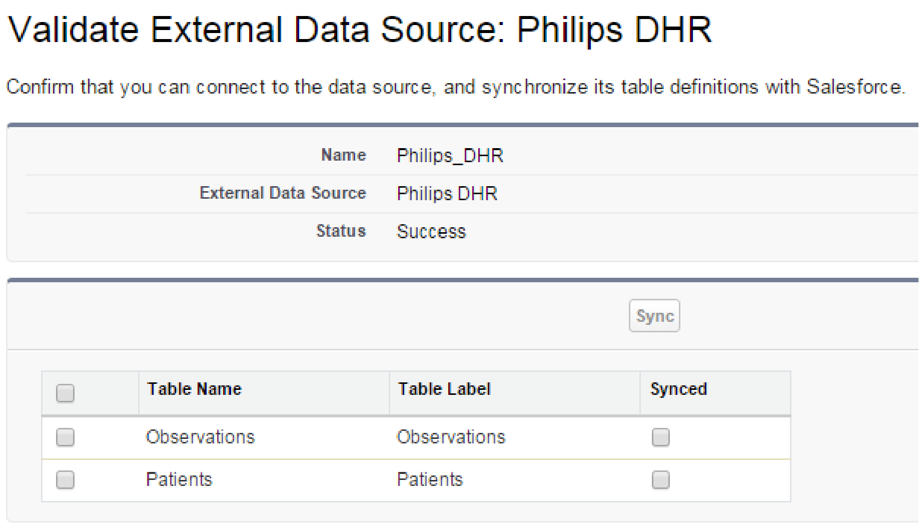
« Back to External Data Sources

External Data Source	Philips DHR
Name	Philips_DHR
Type	Lightning Connect: OData 2.0

Parameters

URL	https://gateway.api.pcftest.com:9004/v1/fhir_odata/
Connection Timeout (Seconds)	120

Behind the scenes, Salesforce will make a $metadata call to the OData service and get back a list of the available objects and will display these on the next screen. For the sample Philips OData service the two objects will be Patients and Observations.



Validate External Data Source: Philips DHR

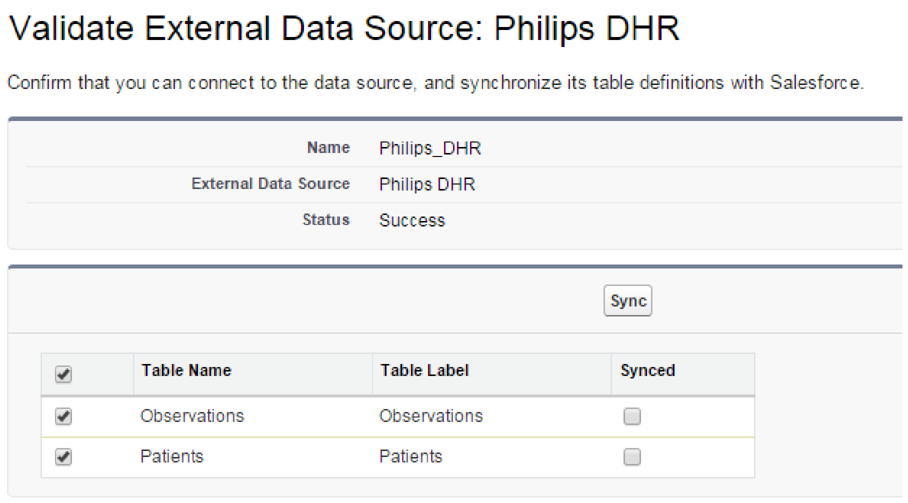
Confirm that you can connect to the data source, and synchronize its table definitions with Salesforce.

Name	Philips_DHR
External Data Source	Philips DHR
Status	Success

Sync

☐	Table Name	Table Label	Synced
☐	Observations	Observations	☐
☐	Patients	Patients	☐

Check the box next to the objects you want to generate in Salesforce and then click the “Sync” button.



Validate External Data Source: Philips DHR

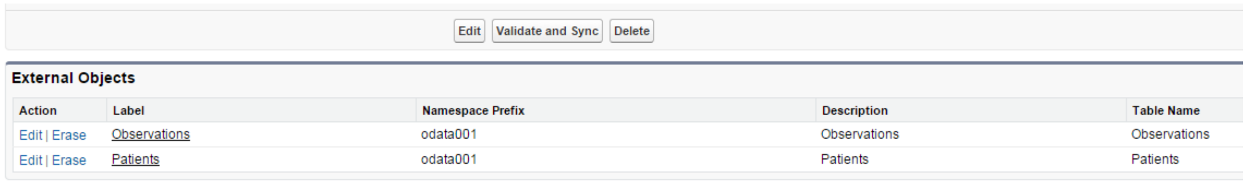
Confirm that you can connect to the data source, and synchronize its table definitions with Salesforce.

Name	Philips_DHR
External Data Source	Philips DHR
Status	Success

Sync

✓	Table Name	Table Label	Synced
✓	Observations	Observations	☐
✓	Patients	Patients	☐

If the objects were created successfully then you’ll be redirected back to the External Data Source detail page and if you scroll to the bottom you’ll see a list of the external objects that were generated.



Edit | Validate and Sync | Delete

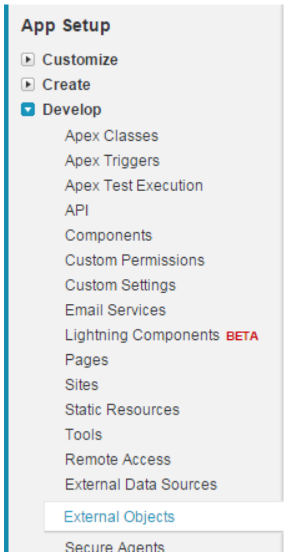
External Objects

Action	Label	Namespace Prefix	Description	Table Name
Edit | Erase	Observations	odata001	Observations	Observations
Edit | Erase	Patients	odata001	Patients	Patients

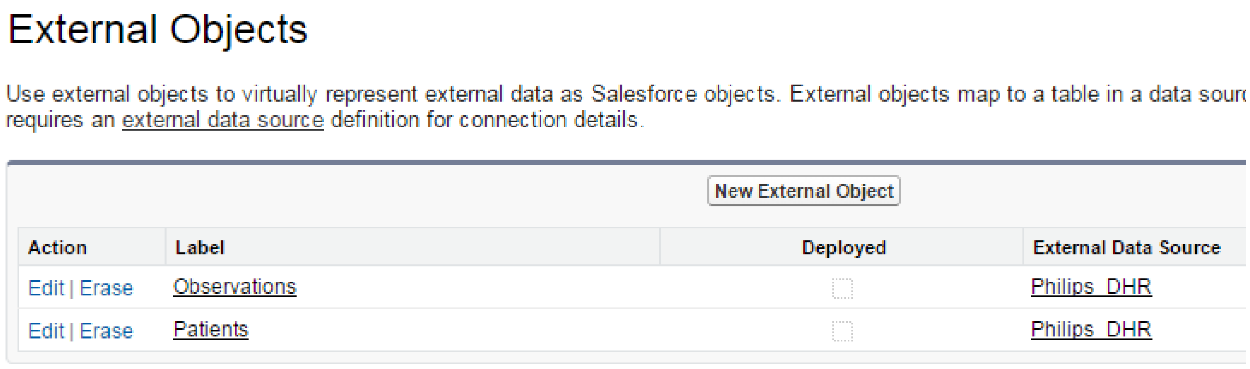
Modifying External Objects

Once you've synced the External Objects from the OData $metadata, you have the opportunity to changes characteristics of these objects such as their labels.

You can access the External Objects directly by navigating to "Develop" -> "External Objects" under.



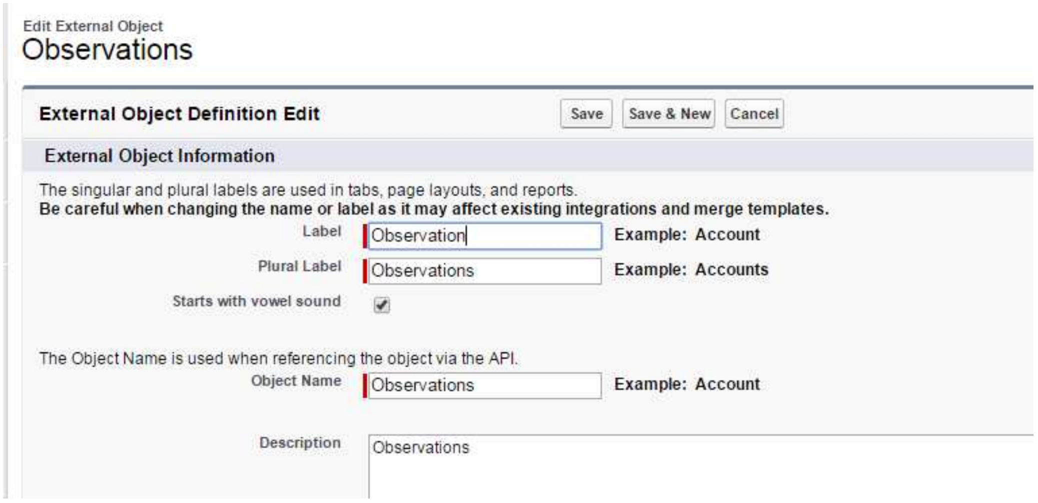
You should see a list of the External Objects that have been created in your org.


External Objects

Use external objects to virtually represent external data as Salesforce objects. External objects map to a table in a data sour
requires an external data source definition for connection details.

Action	Label	Deployed	External Data Source
Edit | Erase	Observations	☐	Philips_DHR
Edit | Erase	Patients	☐	Philips_DHR

Clicking on the “Edit” link will take you to a screen where you can change the single and plural labels for the object and add a description.



Edit External Object

Observations

External Object Definition Edit

Save | Save & New | Cancel

External Object Information

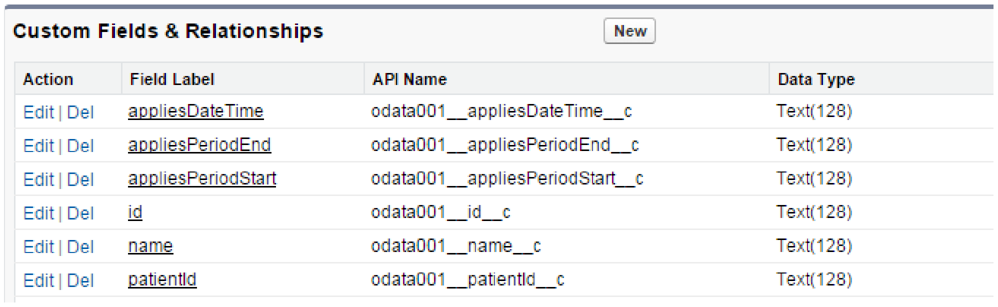
The singular and plural labels are used in tabs, page layouts, and reports.
Be careful when changing the name or label as it may affect existing integrations and merge templates.

Label	Observation	Example: Account
Plural Label	Observations	Example: Accounts
Starts with vowel sound	✓	

The Object Name is used when referencing the object via the API.

Object Name	Observations	Example: Account
Description	Observations	

If you return to the External Object list, clicking on the object name will take you to the detail view screen and on there you can access the custom fields for the object.

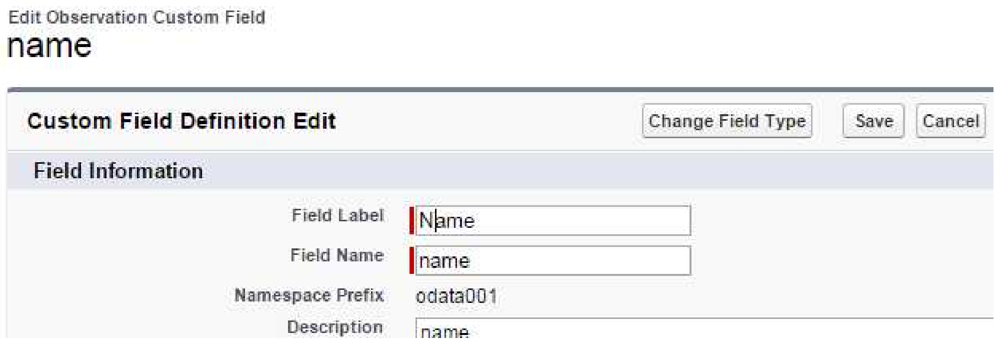


Custom Fields & Relationships

New

Action	Field Label	API Name	Data Type
Edit | Del	appliesDateTime	odata001__appliesDateTime__c	Text(128)
Edit | Del	appliesPeriodEnd	odata001__appliesPeriodEnd__c	Text(128)
Edit | Del	appliesPeriodStart	odata001__appliesPeriodStart__c	Text(128)
Edit | Del	id	odata001__id__c	Text(128)
Edit | Del	name	odata001__name__c	Text(128)
Edit | Del	patientId	odata001__patientId__c	Text(128)

If you click the “Edit” link next to a custom field you can change the label and description for that field.



Edit Observation Custom Field

name

Custom Field Definition Edit

Change Field Type | Save | Cancel

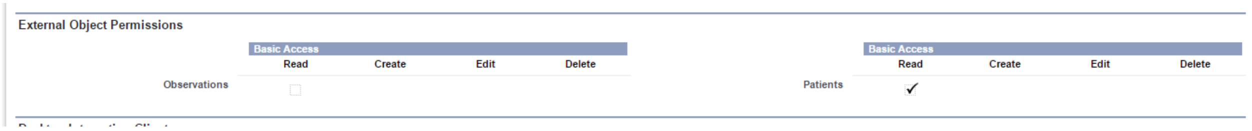
Field Information

Field Label	Name
Field Name	name
Namespace Prefix	odata001
Description	name

Controlling access to External Objects and their fields using custom Profiles

The Salesforce1 platform comes with built in tools for controlling user access to objects or fields within the objects. The two main features for determining access are profiles and permission sets. The particulars of how and when to use profiles and/or permission sets is beyond the scope of this cookbook but I'll explain how they can be used with External Objects.

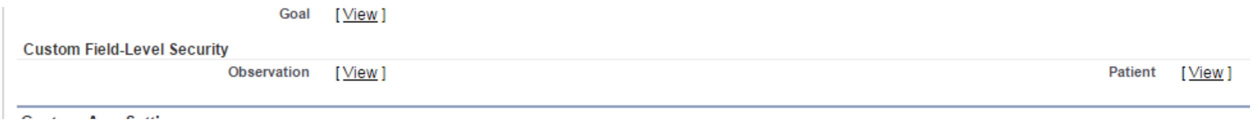
On the Profile detail page, the External Object Permissions are in their own grouping (separate from Standard and Custom SObjects). If you edit the profile you can check or uncheck the "Read" box for the External Objects (even though they display, the Credit, Edit, and Delete permissions don't apply to External Objects).



External Object Permissions

	Basic Access					Basic Access			
	Read	Create	Edit	Delete		Read	Create	Edit	Delete
Observations					Patients	✓			

Scrolling further down on the Profile detail page, you'll see that the External Objects are listed in the "Custom Field-Level Security" section. From here you can click the "View" link to see a list of the fields for that object.

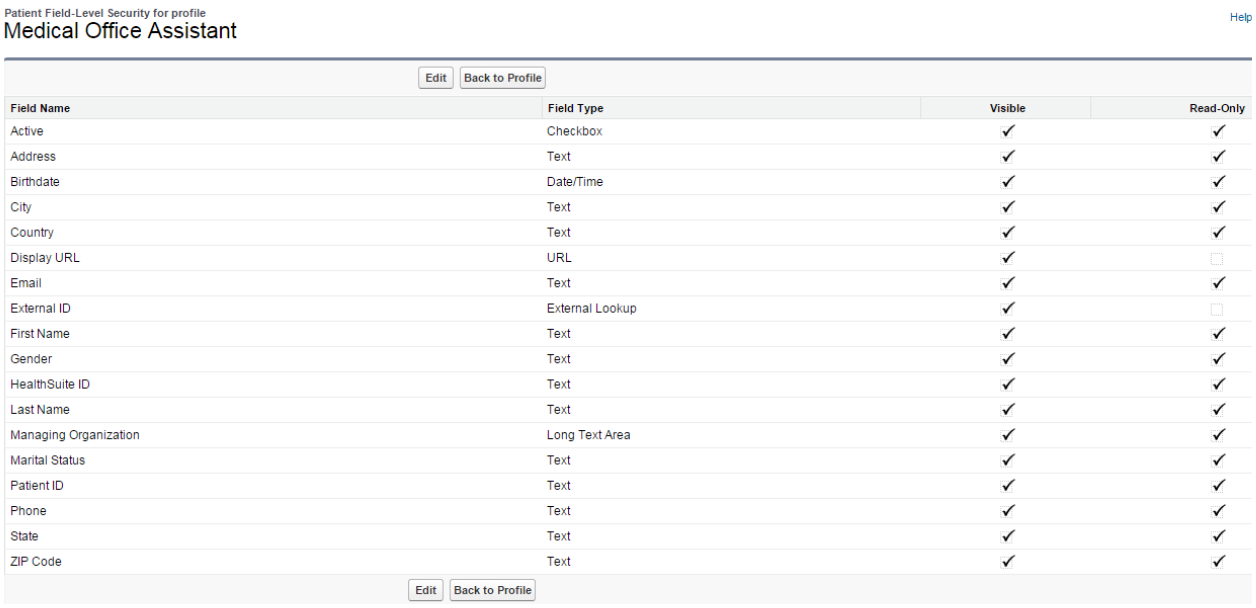


Goal [View]

Custom Field-Level Security

Observation [View] Patient [View]

On the FLS screen for a particular object, you can see which fields of that External Object are visible to users in the current Profile.

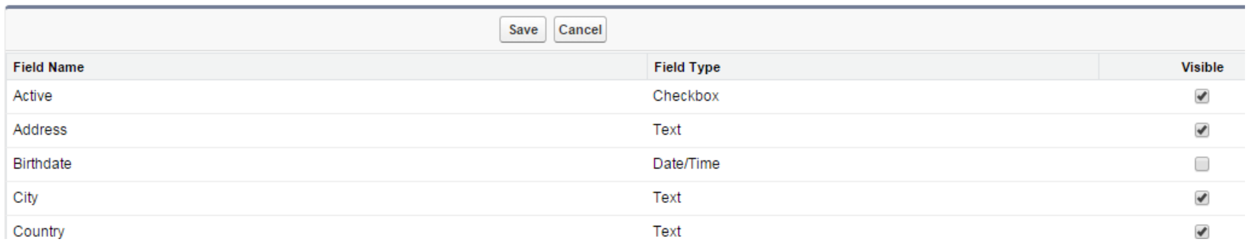


Patient Field-Level Security for profile

Medical Office Assistant

Field Name	Field Type	Visible	Read-Only
Active	Checkbox	✓	✓
Address	Text	✓	✓
Birthdate	Date/Time	✓	✓
City	Text	✓	✓
Country	Text	✓	✓
Display URL	URL	✓	
Email	Text	✓	✓
External ID	External Lookup	✓	
First Name	Text	✓	✓
Gender	Text	✓	✓
HealthSuite ID	Text	✓	✓
Last Name	Text	✓	✓
Managing Organization	Long Text Area	✓	✓
Marital Status	Text	✓	✓
Patient ID	Text	✓	✓
Phone	Text	✓	✓
State	Text	✓	✓
ZIP Code	Text	✓	✓

Clicking the "Edit" button on the FLS list will allow you to check or uncheck the "Visible" boxes for the fields in your object.



Save Cancel

Field Name	Field Type	Visible
Active	Checkbox	✓
Address	Text	✓
Birthdate	Date/Time	☐
City	Text	✓
Country	Text	✓

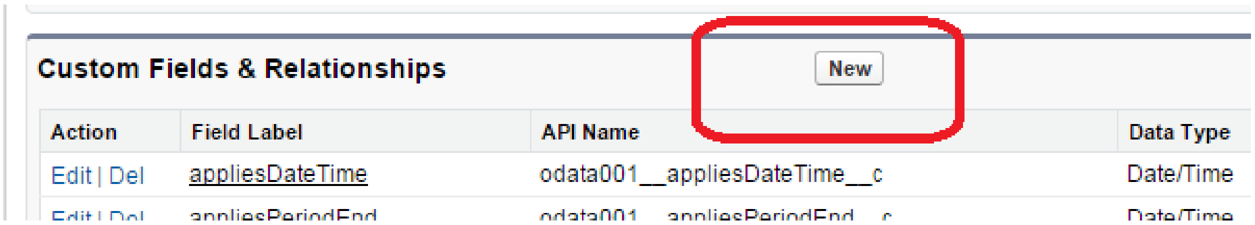
I won't go into detail regarding Permission Sets but suffice to say it works similarly to Profiles.

If you would like more information about how Profiles and/or Permission Sets work in general, please follow one of these links.
https://help.salesforce.com/HTViewHelpDoc?id=admin_userprofiles.htm
https://help.salesforce.com/apex/HTViewHelpDoc?id=permissions_about_users_access.htm&language=en_US

Create relationship between External Objects

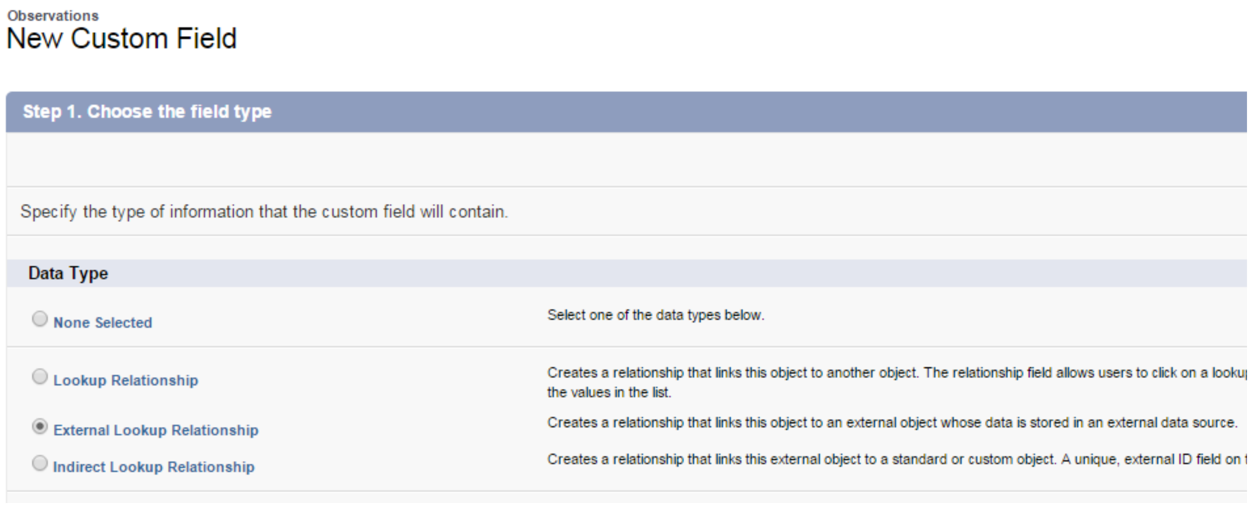
Salesforce allows relationships between two External Objects (or between an External Object and a Standard or Custom object) to be specified. The specific type of relationship that salesforce supports is called "External Lookup" and it means that the child object in the relationship contains a field with the ID of the related parent. One such relationship that exists in the HSDP sample data is that between the Observation and Patient records. The Observation records contain a field named patientId which holds the ID of the Patient record that Observation pertains to.

To create a custom relationship, start by navigating to the detail view for the object in question. In this example I'm going to create a relationship between Patient and Observation so I'll start by going to the Observation object detail page. Scroll down to the "Custom Fields & Relationships" section and click the "New" button



Action	Field Label	API Name	Data Type
Edit | Del	appliesDateTime	odata001__appliesDateTime__c	Date/Time
Edit | Del	appliesPeriodEnd	odata001__appliesPeriodEnd__c	Date/Time

Initially you'll be prompted for the type of field, be sure to choose "External Lookup Relationship."


Observations

New Custom Field

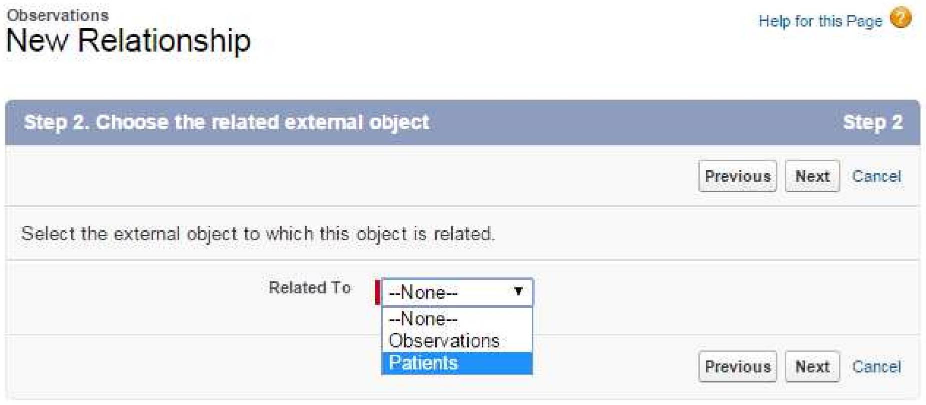
Step 1. Choose the field type

Specify the type of information that the custom field will contain.

Data Type

None Selected	Select one of the data types below.
Lookup Relationship	Creates a relationship that links this object to another object. The relationship field allows users to click on a looku the values in the list.
[✓ External Lookup Relationship]	Creates a relationship that links this object to an external object whose data is stored in an external data source.
Indirect Lookup Relationship	Creates a relationship that links this external object to a standard or custom object. A unique, external ID field on t

In the second step you'll be prompted to choose the object you're relating to (i.e., the "parent" object). In this example I'm choosing the Patient object.


Observations

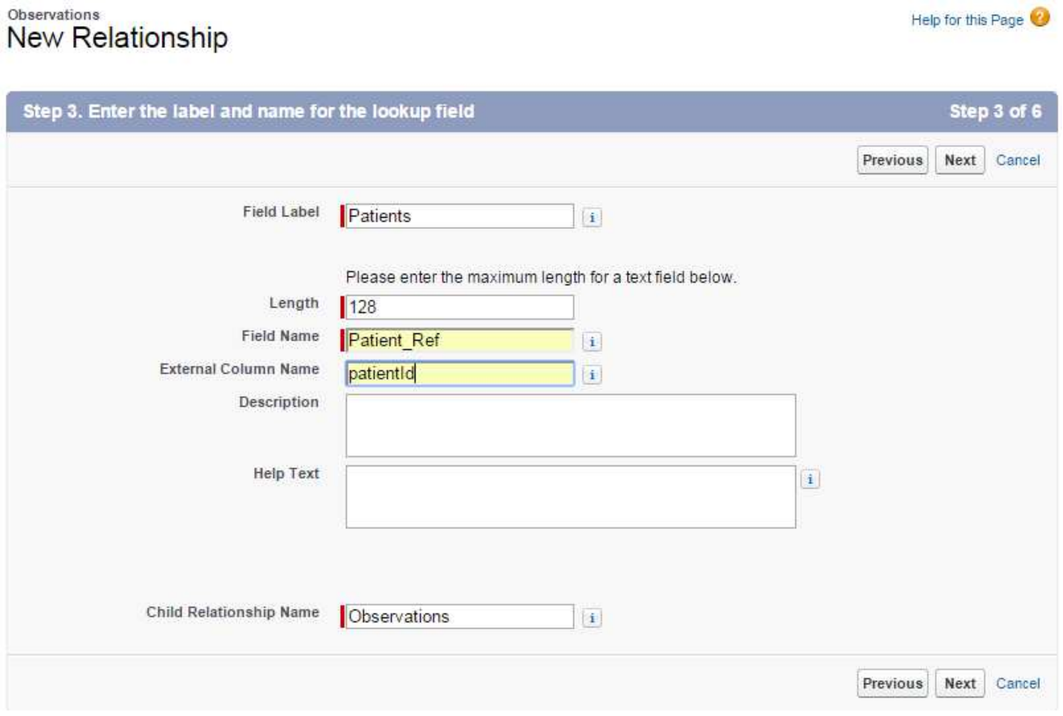
New Relationship

Step 2. Choose the related external object

Select the external object to which this object is related.

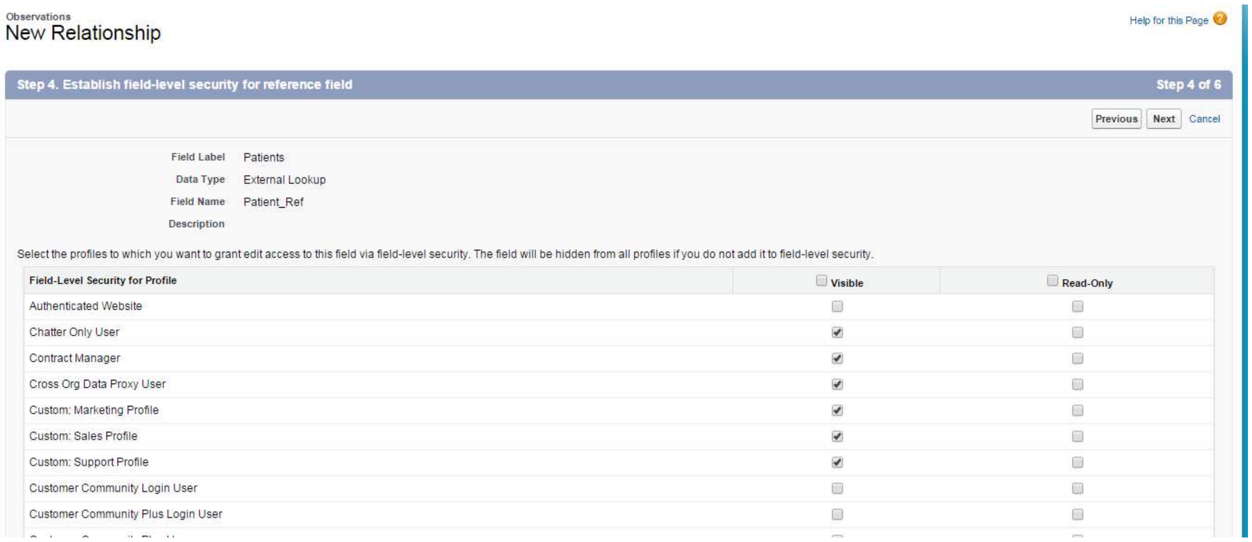
Related To: --None-- / Observations / [✓ Patients]

In step 3 you specify the label for the new lookup field along with the length (use 128 for string ID fields). Later in the process you'll be given the option whether you want this field to display on the page layout.



Field Label	Patients
	Please enter the maximum length for a text field below.
Length	128
Field Name	Patient_Ref
External Column Name	patientId
Description	
Help Text	
Child Relationship Name	Observations

The fourth steps prompts you to check which Profiles will be able to see the new lookup field.

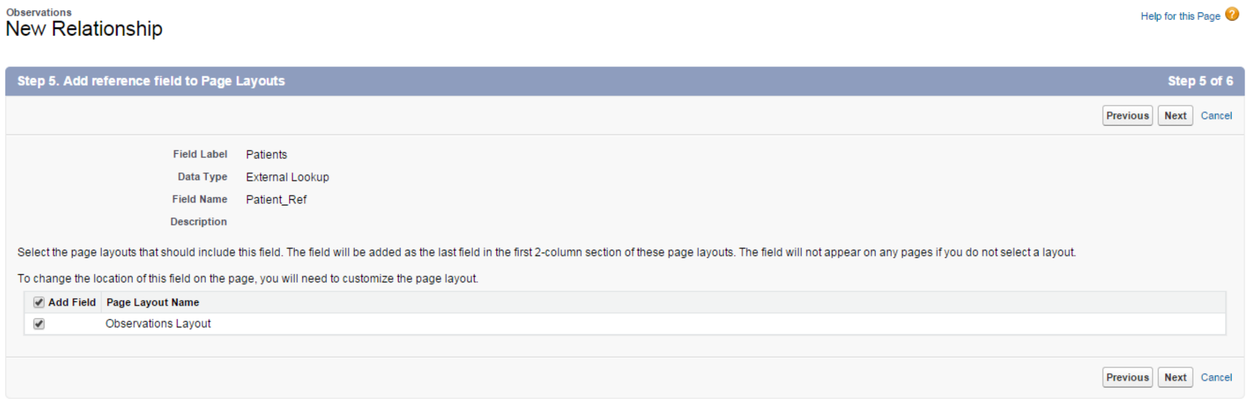


Field Label	Patients
Data Type	External Lookup
Field Name	Patient_Ref
Description	

Select the profiles to which you want to grant edit access to this field via field-level security. The field will be hidden from all profiles if you do not add it to field-level security.

Field-Level Security for Profile	☐ Visible	☐ Read-Only
Authenticated Website	☐	☐
Chatter Only User	☑	☐
Contract Manager	☑	☐
Cross Org Data Proxy User	☑	☐
Custom: Marketing Profile	☑	☐
Custom: Sales Profile	☑	☐
Custom: Support Profile	☑	☐
Customer Community Login User	☐	☐
Customer Community Plus Login User	☐	☐

In step 5 you can specify which page layouts for the *child* object will display the ID of the parent record.



Observations
New Relationship

Step 5. Add reference field to Page Layouts — Step 5 of 6

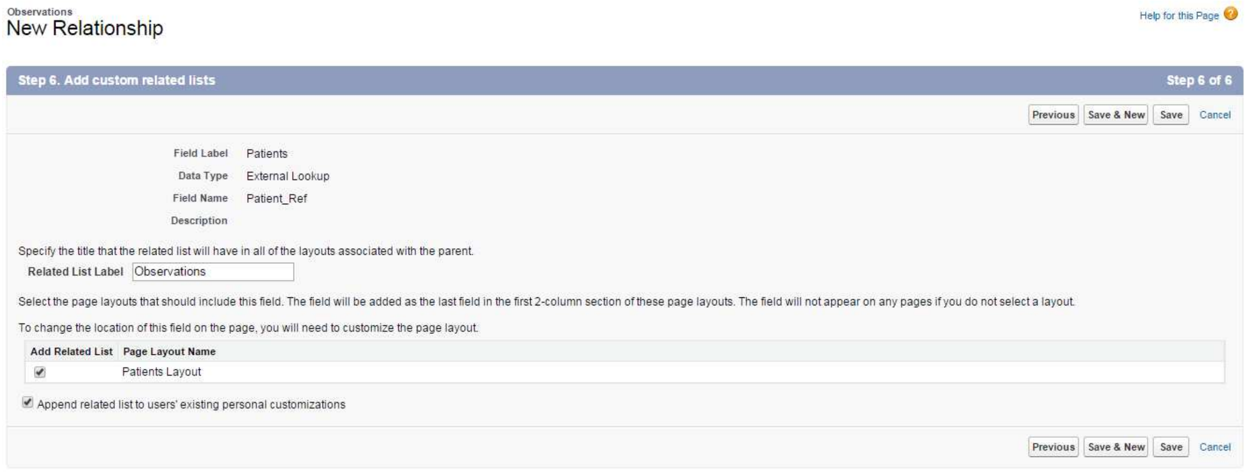
Field Label	Patients
Data Type	External Lookup
Field Name	Patient_Ref
Description	

Select the page layouts that should include this field. The field will be added as the last field in the first 2-column section of these page layouts. The field will not appear on any pages if you do not select a layout.

To change the location of this field on the page, you will need to customize the page layout.

✓ Add Field	Page Layout Name
✓	Observations Layout

Finally in step 6 you can specify whether you want Salesforce to add a related list to the *parent* object.



Observations
New Relationship

Step 6. Add custom related lists — Step 6 of 6

Field Label	Patients
Data Type	External Lookup
Field Name	Patient_Ref
Description	

Specify the title that the related list will have in all of the layouts associated with the parent.

Related List Label: Observations

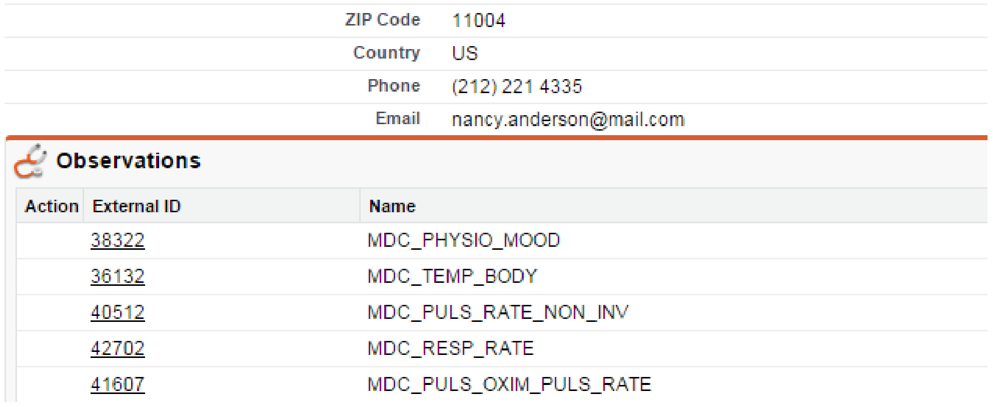
Select the page layouts that should include this field. The field will be added as the last field in the first 2-column section of these page layouts. The field will not appear on any pages if you do not select a layout.

To change the location of this field on the page, you will need to customize the page layout.

Add Related List	Page Layout Name
✓	Patients Layout

✓ Append related list to users' existing personal customizations

Once you've saved the external lookup relationship you can navigate to one of the Patient object instances and you'll now see the Observations related list at the bottom of the page.



ZIP Code	11004
Country	US
Phone	(212) 221 4335
Email	nancy.anderson@mail.com

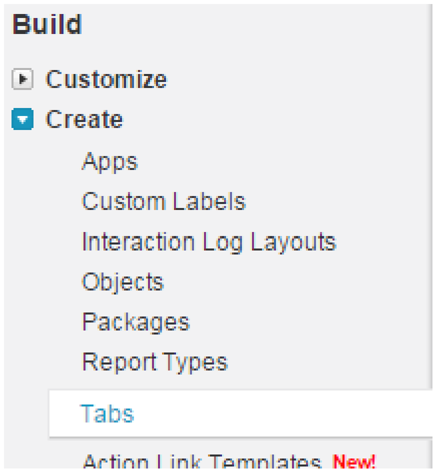
Observations

Action	External ID	Name
	38322	MDC_PHYSIO_MOOD
	36132	MDC_TEMP_BODY
	40512	MDC_PULS_RATE_NON_INV
	42702	MDC_RESP_RATE
	41607	MDC_PULS_OXIM_PULS_RATE

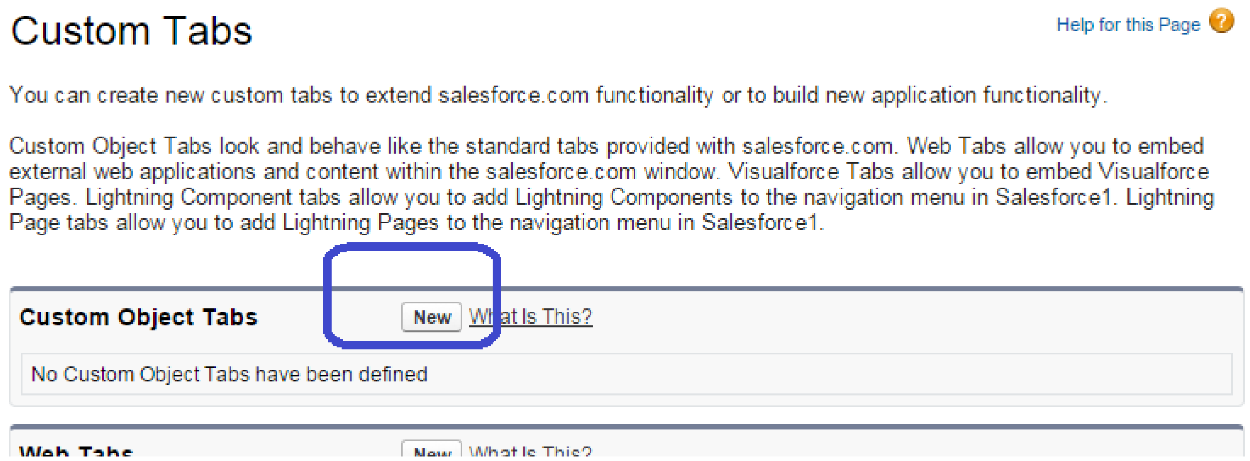
Create a custom tab for an External Object

Salesforce displays tabs across the top of the pages that allow users to navigate to specific objects.

A developer can add new custom tabs to access External Objects. Start by navigating to "Create" -> "Tabs".



On the "Custom Tabs" page click the "New" button. This will launch a three step process to create the new tab.



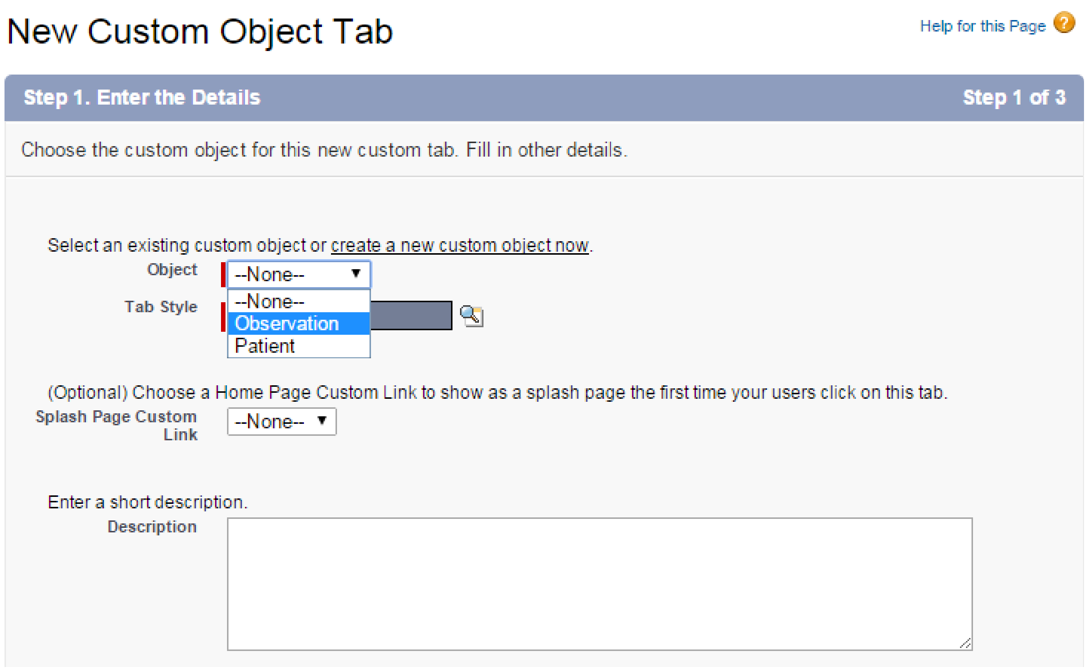
Custom Tabs

Help for this Page

You can create new custom tabs to extend salesforce.com functionality or to build new application functionality.

Custom Object Tabs look and behave like the standard tabs provided with salesforce.com. Web Tabs allow you to embed external web applications and content within the salesforce.com window. Visualforce Tabs allow you to embed Visualforce Pages. Lightning Component tabs allow you to add Lightning Components to the navigation menu in Salesforce1. Lightning Page tabs allow you to add Lightning Pages to the navigation menu in Salesforce1.

The first page in the flow prompts you to specify the object the new tab will represent.


New Custom Object Tab

Help for this Page

Step 1. Enter the Details — Step 1 of 3

Choose the custom object for this new custom tab. Fill in other details.

Select an existing custom object or create a new custom object now.

Object: --None--

Tab Style

--None--
Observation
Patient

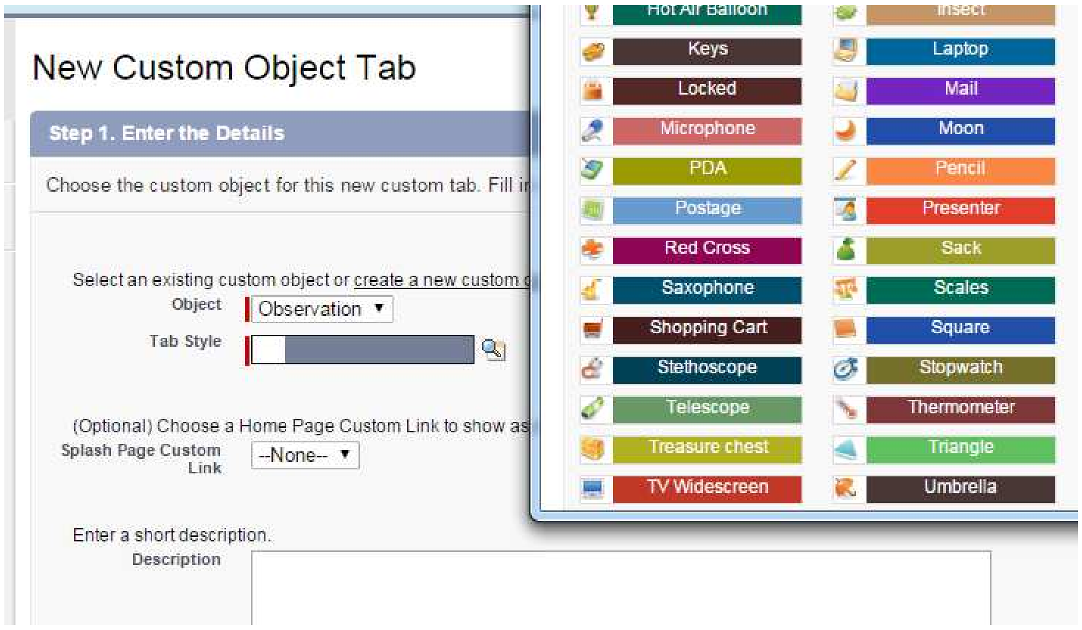
(Optional) Choose a Home Page Custom Link to show as a splash page the first time your users click on this tab.

Splash Page Custom Link: --None--

Enter a short description.

Description

Next you need to choose the tab style. This will determine the color of the tab and the icon that will appear at the top of the list and view pages for that object. Clicking on the magnifying glass icon launches a new window with a list of the out of the box tab styles. You can choose one of these or create a new style.


New Custom Object Tab

Step 1. Enter the Details

Choose the custom object for this new custom tab. Fill in

Select an existing custom object or create a new custom o

Object: Observation

Tab Style

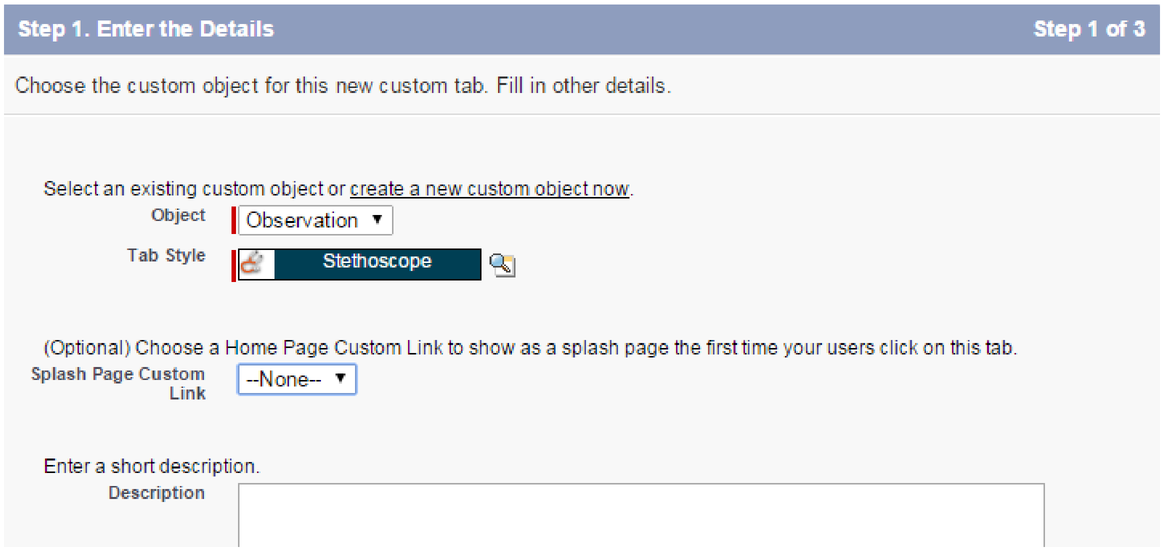
(Optional) Choose a Home Page Custom Link to show as

Splash Page Custom Link: --None--

Enter a short description.

Description

Once you've specified the object and tab style, click the "Next" button.



Step 1. Enter the Details — Step 1 of 3

Choose the custom object for this new custom tab. Fill in other details.

Select an existing custom object or create a new custom object now.

Object	Observation ▼
Tab Style	Stethoscope

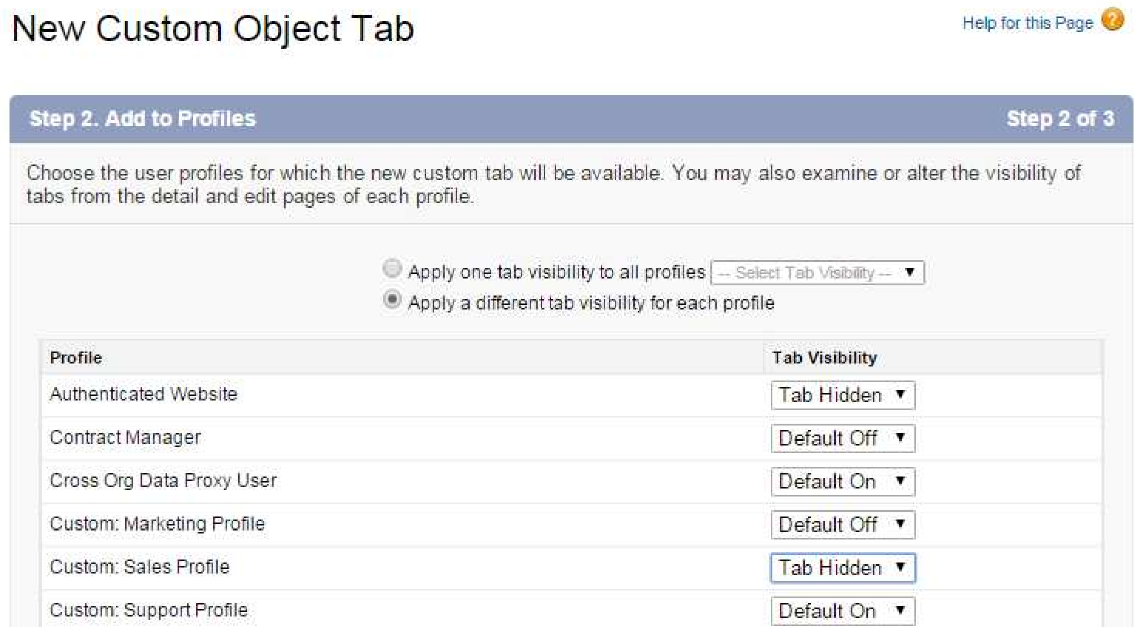
(Optional) Choose a Home Page Custom Link to show as a splash page the first time your users click on this tab.

Splash Page Custom Link	--None-- ▼

Enter a short description.

Description

On the next screen you'll be able to specify whether or not that tab is visible to each of the profiles in your Salesforce1 org.



New Custom Object Tab

Help for this Page

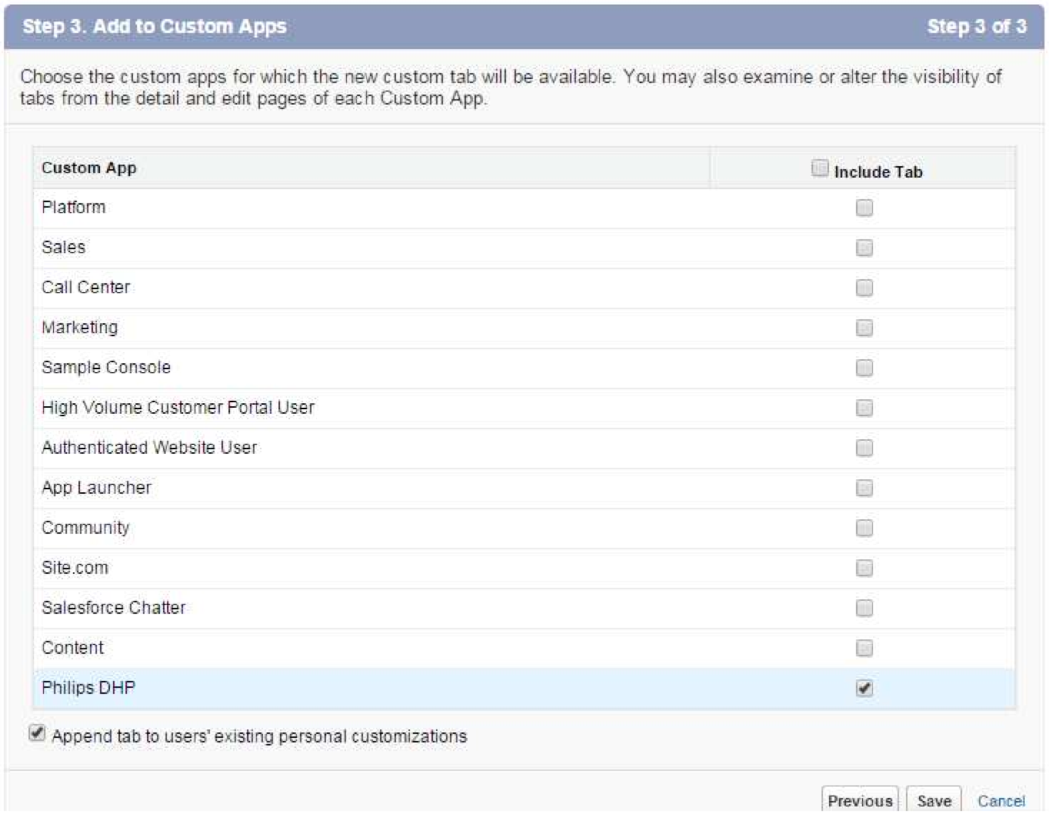
Step 2. Add to Profiles — Step 2 of 3

Choose the user profiles for which the new custom tab will be available. You may also examine or alter the visibility of tabs from the detail and edit pages of each profile.

○ Apply one tab visibility to all profiles -- Select Tab Visibility -- ▼
● Apply a different tab visibility for each profile

Profile	Tab Visibility
Authenticated Website	Tab Hidden ▼
Contract Manager	Default Off ▼
Cross Org Data Proxy User	Default On ▼
Custom: Marketing Profile	Default Off ▼
Custom: Sales Profile	Tab Hidden ▼
Custom: Support Profile	Default On ▼

On the third and final page you can check which apps the new custom tab will be available from.



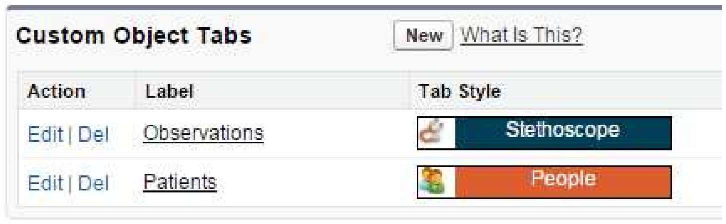
Step 3. Add to Custom Apps — Step 3 of 3

Choose the custom apps for which the new custom tab will be available. You may also examine or alter the visibility of tabs from the detail and edit pages of each Custom App.

Custom App	☐ Include Tab
Platform	☐
Sales	☐
Call Center	☐
Marketing	☐
Sample Console	☐
High Volume Customer Portal User	☐
Authenticated Website User	☐
App Launcher	☐
Community	☐
Site.com	☐
Salesforce Chatter	☐
Content	☐
Philips DHP	✓

✓ Append tab to users' existing personal customizations

Here are the tab styles I choose for the Patient and Observation objects.



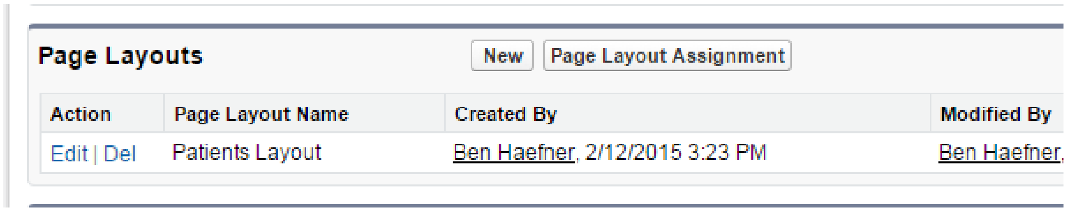
Custom Object Tabs

Action	Label	Tab Style
Edit | Del	Observations	Stethoscope
Edit | Del	Patients	People

Customizing page layout for External Object

The Salesforce1 platform automatically generates list and view pages for all Standard and Custom objects and this also applies to External Objects.

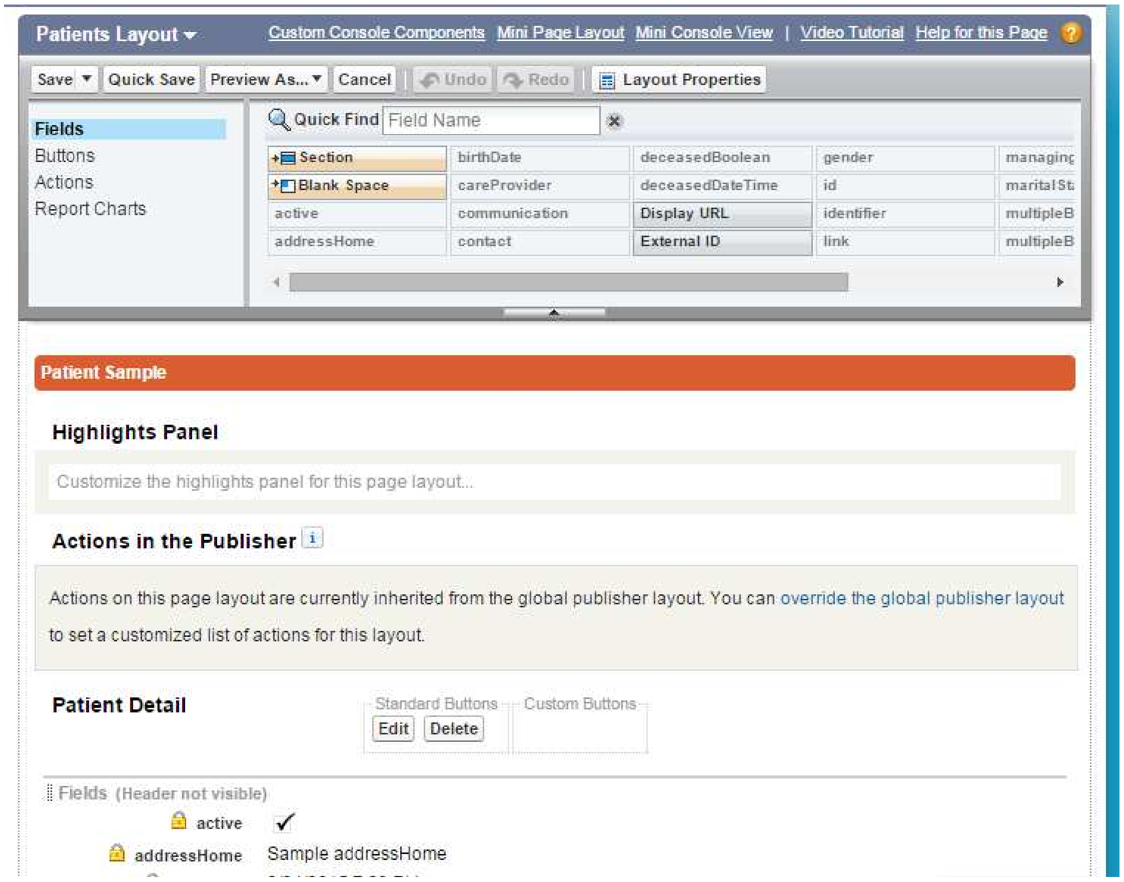
Begin by navigating to the External Object whose page layout you're interested in changing. In this example I'm using the Patient object.



Page Layouts New | Page Layout Assignment

Action	Page Layout Name	Created By	Modified By
Edit | Del	Patients Layout	Ben Haefner, 2/12/2015 3:23 PM	Ben Haefner,

Clicking the "Edit" link brings up the page layout edit screen.

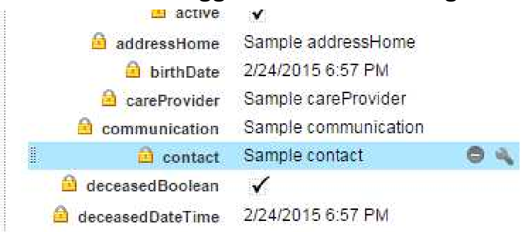


Customize the highlights panel for this page layout...

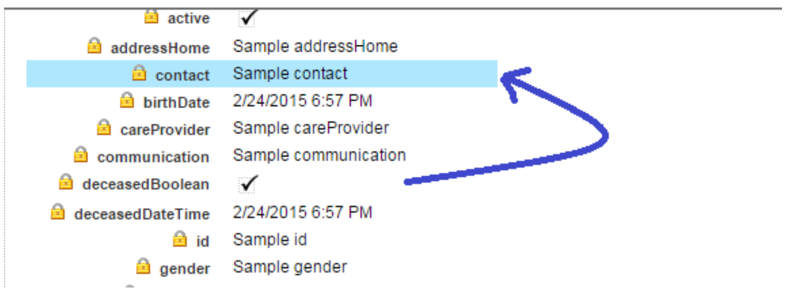
Actions on this page layout are currently inherited from the global publisher layout. You can override the global publisher layout to set a customized list of actions for this layout.

active	✓
addressHome	Sample addressHome

Fields can be dragged around to change their sequence in the page layout.

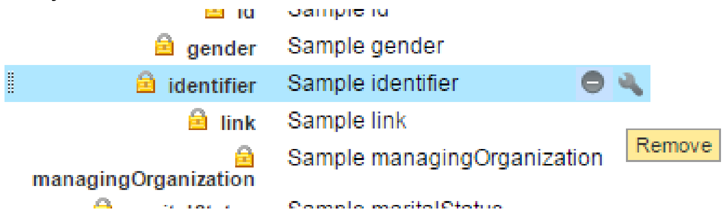


active	✓
addressHome	Sample addressHome
birthDate	2/24/2015 6:57 PM
careProvider	Sample careProvider
communication	Sample communication
contact	Sample contact
deceasedBoolean	✓
deceasedDateTime	2/24/2015 6:57 PM



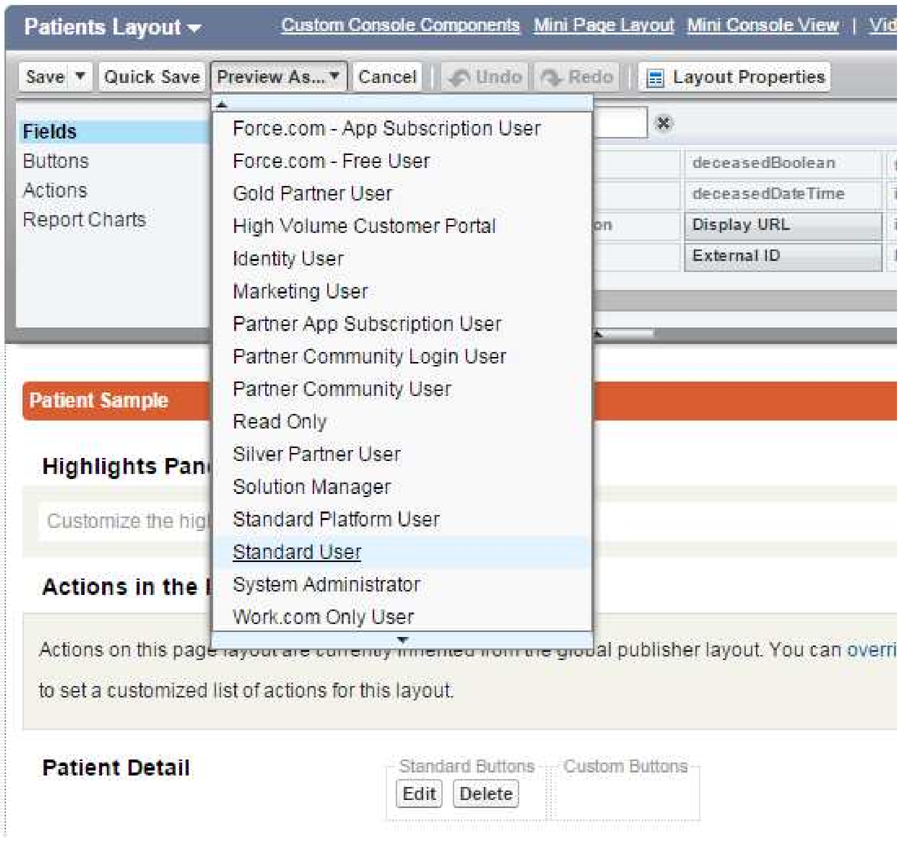
active	✓
addressHome	Sample addressHome
contact	Sample contact
birthDate	2/24/2015 6:57 PM
careProvider	Sample careProvider
communication	Sample communication
deceasedBoolean	✓
deceasedDateTime	2/24/2015 6:57 PM
id	Sample id
gender	Sample gender

If you mouse over the individual fields you'll have the option to remove the field from the layout or change its attributes. **NOTE:** removing a field from a page layout does *not* remove the custom field from the object itself so it will still be available to use in the future or on other page layouts.

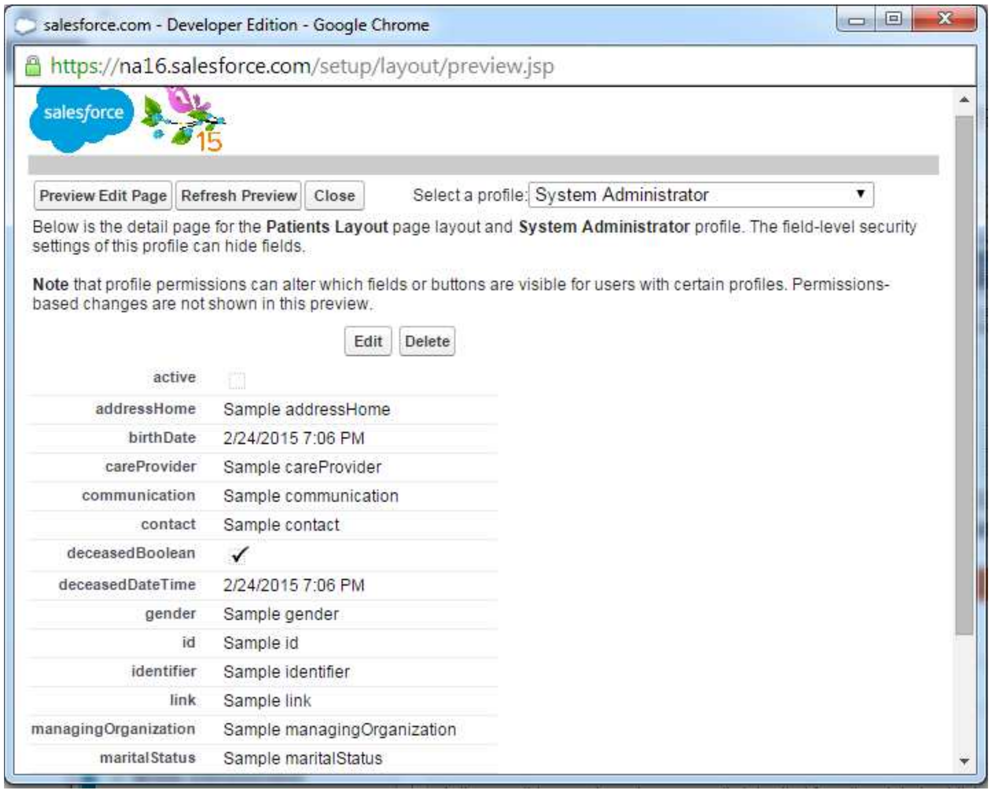


gender	Sample gender
identifier	Sample identifier
link	Sample link
managingOrganization	Sample managingOrganization

From the page layout edit screen you can choose to preview the page layout as a user with a particular profile. The reason you might want to see one profile versus another is because field-level security on the profile dictates which fields will be visible. To do this, choose a profile from the “Preview As…” drop down near the top of the page.



A new window will pop-up which shows an example of what the current page layout would look like for a user of the given profile. From this pop-up you can switch to other profiles.


salesforce.com - Developer Edition - Google Chrome
https://na16.salesforce.com/setup/layout/preview.jsp
salesforce
15
Preview Edit Page
Refresh Preview
Close
Edit
Delete

Select a profile: System Administrator

Below is the detail page for the **Patients Layout** page layout and **System Administrator** profile. The field-level security settings of this profile can hide fields.

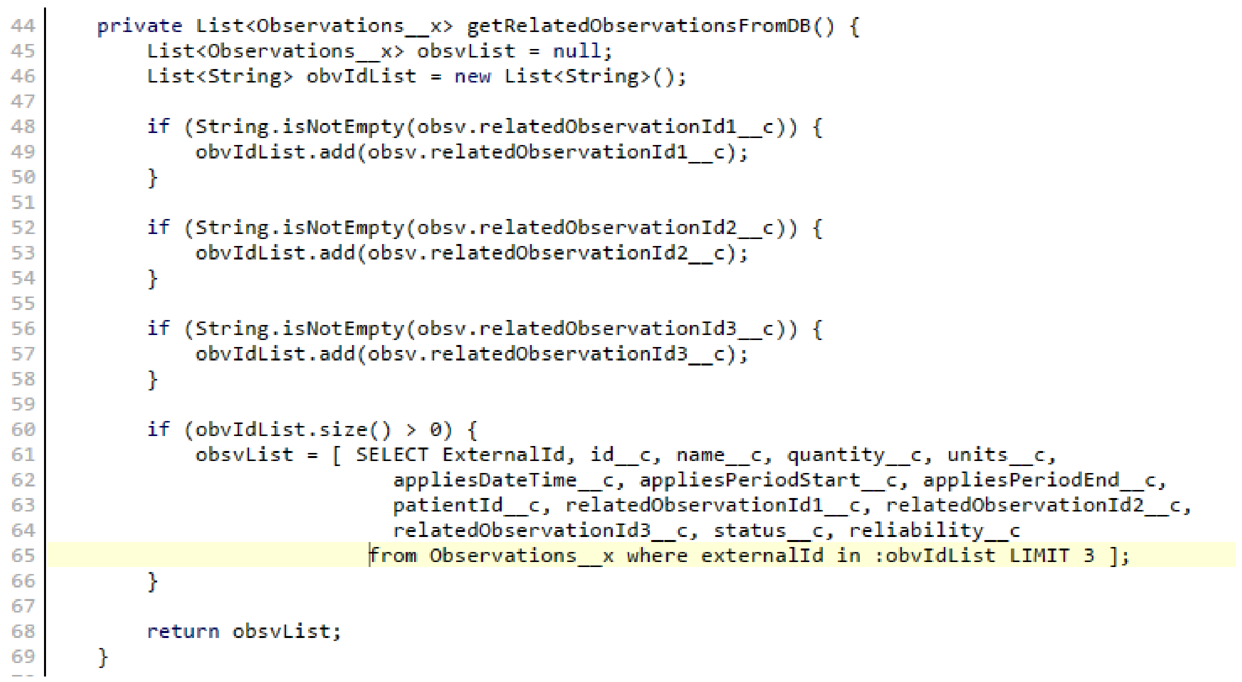
Note that profile permissions can alter which fields or buttons are visible for users with certain profiles. Permissions-based changes are not shown in this preview.

active	
addressHome	Sample addressHome
birthDate	2/24/2015 7:06 PM
careProvider	Sample careProvider
communication	Sample communication
contact	Sample contact
deceasedBoolean	✓
deceasedDateTime	2/24/2015 7:06 PM
gender	Sample gender
id	Sample id
identifier	Sample identifier
link	Sample link
managingOrganization	Sample managingOrganization
maritalStatus	Sample maritalStatus

Accessing External Objects from SOQL queries in Apex code

The External Objects can be accessed from Apex code using SOQL queries in the same way you can access standard and custom SObjects.

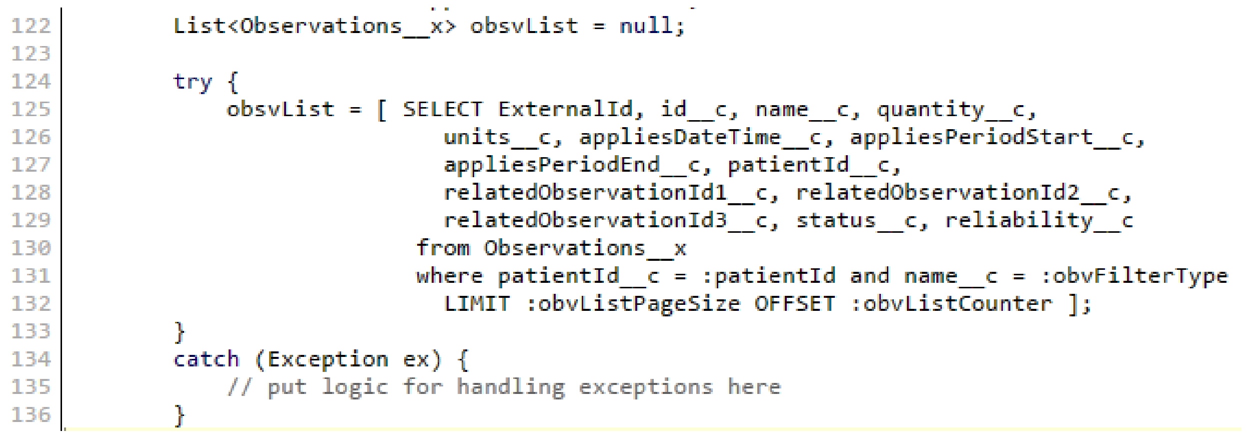
Here's an example snippet of code that is retrieving a list of child Observations for a particular parent. An Observation can have 0 – 3 related or child Observations. This logic checks to see if any of the relatedObservationId#__c fields are populated and adds their value to a list of IDs.



```
    private List<Observations__x> getRelatedObservationsFromDB() {
        List<Observations__x> obsvList = null;
        List<String> obvIdList = new List<String>();

        if (String.isNotEmpty(obsv.relatedObservationId1__c)) {
            obvIdList.add(obsv.relatedObservationId1__c);
        }

        if (String.isNotEmpty(obsv.relatedObservationId2__c)) {
            obvIdList.add(obsv.relatedObservationId2__c);
        }

        if (String.isNotEmpty(obsv.relatedObservationId3__c)) {
            obvIdList.add(obsv.relatedObservationId3__c);
        }

        if (obvIdList.size() > 0) {
            obsvList = [ SELECT ExternalId, id__c, name__c, quantity__c, units__c,
                                appliesDateTime__c, appliesPeriodStart__c, appliesPeriodEnd__c,
                                patientId__c, relatedObservationId1__c, relatedObservationId2__c,
                                relatedObservationId3__c, status__c, reliability__c
                              from Observations__x where externalId in :obvIdList LIMIT 3 ];
        }

        return obsvList;
    }
```

A couple things to note. In standard or custom SObjects there is a standard field named "Id" which you can use to query an object by its unique identifier. The same concept exists for External Objects but the name of the field is "ExternalId". Also, when querying a list of objects you should include the LIMIT constraint at the end of your SOQL (e.g., " LIMIT 3"). To prevent overloading the HSDP OData service we're requiring a LIMIT on all queries and that number must be less than or equal to 50.

Here's a more complex SOQL query that is retrieving the Observations for a particular Patient while filtering the results based on the Observation type. It's also using LIMIT and OFFSET to implement pagination.



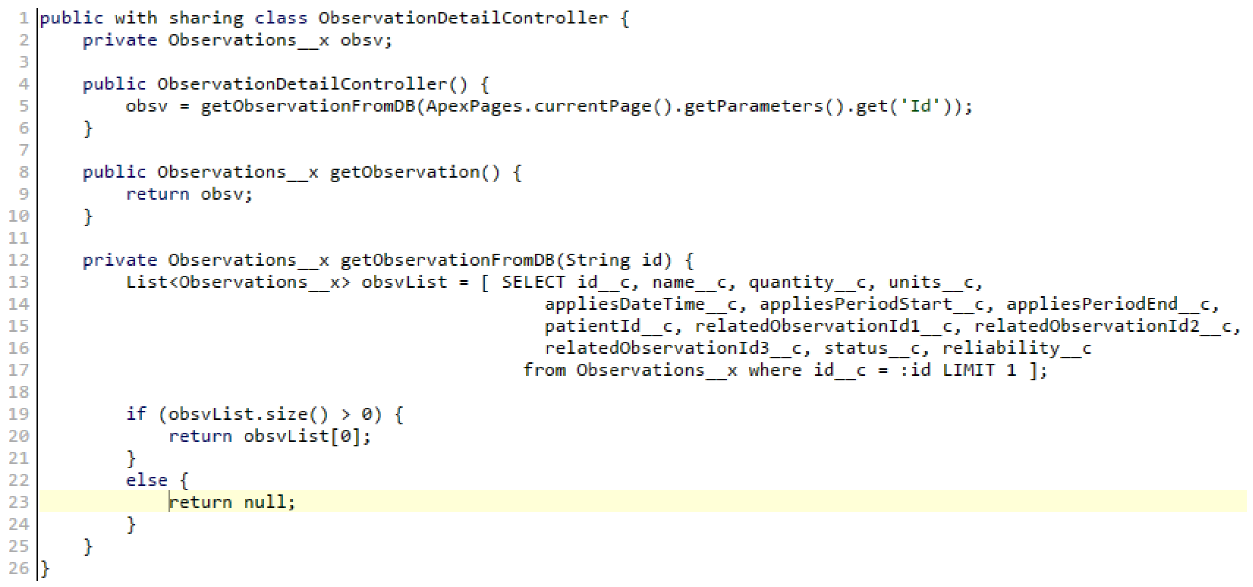
```
        List<Observations__x> obsvList = null;

        try {
            obsvList = [ SELECT ExternalId, id__c, name__c, quantity__c,
                                units__c, appliesDateTime__c, appliesPeriodStart__c,
                                appliesPeriodEnd__c, patientId__c,
                                relatedObservationId1__c, relatedObservationId2__c,
                                relatedObservationId3__c, status__c, reliability__c
                              from Observations__x
                              where patientId__c = :patientId and name__c = :obvFilterType
                                LIMIT :obvListPageSize OFFSET :obvListCounter ];
        }
        catch (Exception ex) {
            // put logic for handling exceptions here
        }
```

There are some specific limitations you should be aware of when creating SOQL queries for External Objects. This link points to the relevant information in the Salesforce1 documentation: https://help.salesforce.com/apex/HTViewHelpDoc?id=platform_connect_considerations_soql.htm&language=en_US

NOTE: Because the External Objects are currently read-only, DML operations are *not* supported.

Alternative to standard controllers for External Objects

According to the Salesforce1 documentation you should be able to extend a standard controller for an External Object in the same way you can for standard and custom objects. However, it has been the experience of the author that this is doesn't work as expected. Thus, when I needed to implement my own controller behavior I created a new Apex controller class and added logic in the class' constructor to retrieve the ID of my external object from a query parameter and retrieve the object by ID.



```
public with sharing class ObservationDetailController {
    private Observations__x obsv;

    public ObservationDetailController() {
        obsv = getObservationFromDB(ApexPages.currentPage().getParameters().get('Id'));
    }

    public Observations__x getObservation() {
        return obsv;
    }

    private Observations__x getObservationFromDB(String id) {
        List<Observations__x> obsvList = [ SELECT id__c, name__c, quantity__c, units__c,
                                               appliesDateTime__c, appliesPeriodStart__c, appliesPeriodEnd__c,
                                               patientId__c, relatedObservationId1__c, relatedObservationId2__c,
                                               relatedObservationId3__c, status__c, reliability__c
                                             from Observations__x where id__c = :id LIMIT 1 ];

        if (obsvList.size() > 0) {
            return obsvList[0];
        }
        else {
            return null;
        }
    }
}
```

For this controller to work you'll need to append "?id={observationId}" to the end of the URL for your custom Visualforce page. For example: /apex/ObservationDetailPage?id=17898

Speaking of the Visualforce page, even though you're using a custom controller the Visualforce page can still use the typical components for displaying the object data such as apex:outputField to show the field labels and values.

```
<apex:page controller="ObservationDetailController" tabStyle="Observations__x">
    <apex:sectionHeader title="Observation" subtitle="{!observation.id__c}" />

    <apex:pageBlock mode="maindetail">
        <apex:pageBlockSection title="Observation Details" columns="1">
            <apex:outputField value="{!observation.id__c}" />
            <apex:outputField value="{!observation.name__c}" />
            <apex:outputField value="{!observation.patientId__c}" />
            <apex:outputField value="{!observation.appliesDateTime__c}" />
            <apex:outputField value="{!observation.quantity__c}" />
            <apex:outputField value="{!observation.reliability__c}" />
            <apex:outputField value="{!observation.status__c}" />
        </apex:pageBlockSection>
    </apex:pageBlock>
</apex:page>
```