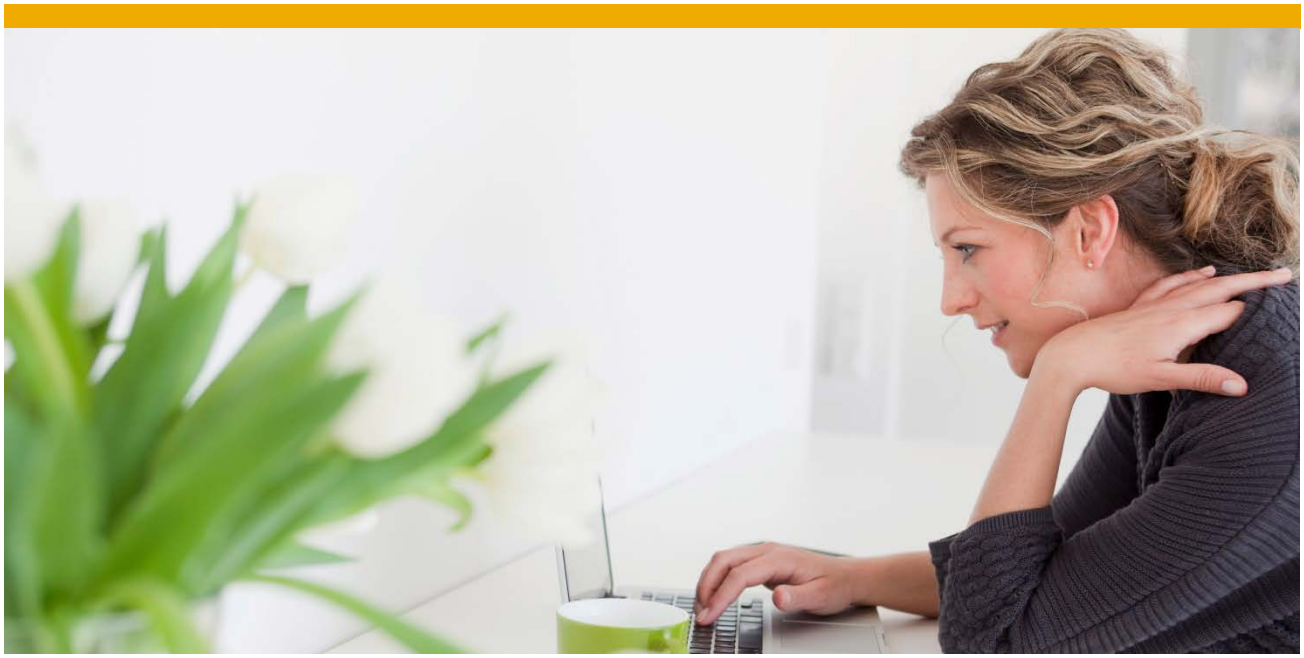


Setting up an Apache Server in Conjunction with the SAP SQL OData Server

PRINCIPAL AUTHORS

Adam Hurst adam.hurst@sap.com
Philippe Bertrand philippe.bertrand@sap.com

REVISION HISTORY

Version 1.0 - June 2013
Version 1.1 – September 2015 – Revised for SQL Anywhere 17

TABLE OF CONTENTS

OVERVIEW

The Apache HTTP Server is the most popular HTTP Server software in use. This document provides an introduction to using the SAP SQL OData Server (herein referred to as the OData Server) to create a web application with rich OData content and functionality where the Apache HTTP Server is preferred as the Internet-facing web server.

OData (Open Data Protocol) enables data services over RESTful HTTP. It allows you to perform operations through URIs (Universal Resource Identifiers) to access and modify information.

The OData Server is a server launched with the SQL Anywhere database server that consists of the following components:

1. **An HTTP server** The HTTP server handles OData requests from clients and acts as a Java Servlet Container for OData Producers.
2. **The OData Producer** The OData Producer is a Java Servlet that provides a particular set of OData services for a single database. It processes OData requests, interfaces with a database, and provides OData responses.

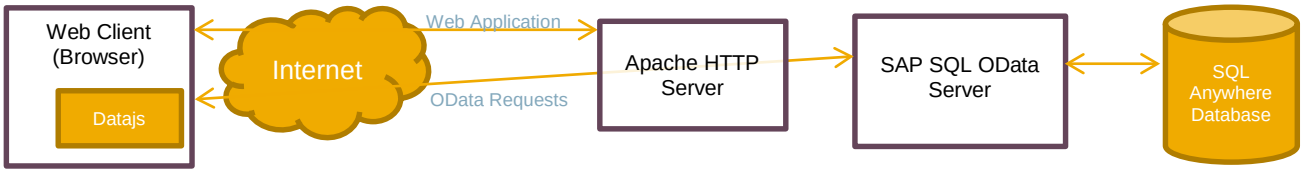
The OData Server can contain multiple OData Producers, for multiple databases running on a single SQL Anywhere database server.

To successfully complete the procedures in this document, you must acquire and install the following software:

- **Apache HTTPD 2.4.12** The Apache HTTP Server is a popular open source HTTP Web Server. This web server is used to host the web application and act as the Internet-facing intermediate for OData traffic between the web clients and the OData Server.
- **SQL Anywhere 17** SQL Anywhere contains the necessary components to allow the OData Server to interface with SQL Anywhere database servers. It also provides a sample SQL Anywhere database that is used as the data source for the web application. The rest of this document assumes that the *$SQLANY17/Bin64* directory is in your PATH and that the *SQLANYSAMP17* environment variable is set to the root of the SQL Anywhere samples (both SQLANY17 and SQLANYSAMP17 are set by the SQL Anywhere installer).
- **Datajs 1.1.2** Datajs is an OData client library for JavaScript. This library is used to facilitate requesting and receiving OData content in the web application.

GOAL

This document provides a procedure for configuring and deploying a web application with rich OData content and functionality where the Apache HTTP Server is used as the Internet-facing web server. The web application files are hosted by the Apache HTTP Server, while OData requests made by the web application are proxied through to the OData Server and satisfied by a SQL Anywhere database server. The following diagram describes the back-end setup of the hosted web application.



HOSTING THE ODATA SERVER

Launching the SQL Anywhere Database and OData Servers

The following steps illustrate how to configure and launch the OData Server to interface with the demo database that is provided in the SQL Anywhere installation and process OData requests:

1. Create a directory to contain the server configuration file and log files. The rest of this document assumes that *ODATA_HOME* is the full path of this directory.
2. Copy the corresponding files that are listed in the appendix to *ODATA_HOME*:
   - ***init.sql*** This SQL file configures the SQL Anywhere demo database with the OData Producer used in this sample.
   - ***odata.osdl*** This OData Service Definition Language file provides the OData Server with additional information regarding how to expose the schema of the SQL Anywhere sample database.
3. Create an OData Producer in the sample database by running the following command from the *ODATA_HOME* direactory:

   dbisql -onerror continue -c "dsn=SQL Anywhere 17 Demo;uid=DBA;pwd=<put-password-here>" init.sql
4. Start the SQL Anywhere database server with an OData Server by running the following command from the *ODATA_HOME* directory:

   dbsrv17 -xs odata (ServerPort=8080;LogFile=odata.log;LogVerbosity=2) "$SQLANYSAMP17/demo.db"
   or
   dbsrv17 -xs odata (ServerPort=8080;LogFile=odata.log;LogVerbosity=2) "%SQLANYSAMP17%\demo.db"
5. Verify that the servlet loaded successfully by visiting the following URL in a web browser:

   http://localhost:8080/ApacheDemo/$metadata.

   An XML document describing the OData service schema appears.

Setting up the Apache HTTP Server

The following steps illustrate how to configure the Apache HTTP Server to proxy OData requests to the OData Server.

After completing these steps, you have a functioning HTTP Server that accepts web requests, and proxies any requests to the */ApacheDemo* path through to the OData Server.

1. Install the Apache HTTP Server according to the instructions provided in the distribution. The rest of this document assumes that *APACHE_HOME* is the full path to the root directory of the Apache HTTP Server installation.
2. Enable the mod_proxy module in the Apache HTTP Server to configure it as a proxy server.
   a. Open the *APACHE_HOME/conf/httpd.conf* file.
   b. Uncomment the following directives, or add them if they do not exist:
      - *LoadModule proxy_module modules/mod_proxy.so*
      - *LoadModule proxy_http_module modules/mod_proxy_http.so*
   c. Add the ProxyPass directive below the LoadModule directives to configure the Apache HTTP Server to redirect incoming requests to the SAP SQL OData Server:

      ProxyPass /odata http://localhost:8080/ApacheDemo
   d. Add the ProxyPreserveHost directive below the ProxyPass directive to configure Apache to preserve the Host header value from the incoming request. This step is required so that any URLs encoded in OData responses correctly point to the Apache server.

```
ProxyPreserveHost       On
```

3. Start or restart the Apache HTTP server (if it is not installed as a service) according to the instructions provided in the distribution.
4. Verify that requests to the */odata* path are being correctly proxied to the OData Server by visiting the following URL in a web browser:

   http://localhost:80/odata/$metadata

   An XML document describing the OData service schema appears.

Setting up the Web Application

The following steps illustrate how to set up the web application that uses the OData service:

1. Copy the web application files to a directory where they can be accessed and served by the Apache HTTP Server.
   a. Create a directory to contain the web application files. The rest of this document assumes that *APP_HOME* is the full path of this directory.
   b. Copy the corresponding files that are listed in the appendix to APP_HOME:
      - *salesOrders.html*
      - *salesOrders.js*
   c. Copy *datajs-1.1.2.js* to *APP_HOME/datajs-1.1.2.js*.
2. Configure the Apache HTTP Server to serve the web application.
   a. Open the *APACHE_HOME/conf/httpd.conf* file.
   b. Add the following Alias and Directory directives to configure the Apache HTTP Server to serve the web application files from the */app* URL path.

```
Alias    /app    "APP_HOME"
<Directory "APP_HOME">
  AllowOverride None
  Options None
  Require all granted
</Directory>
```

   c. Restart the Apache HTTP Server.
   d. Visit *http://localhost:80/app/salesOrders.html* in a web browser.

      The application becomes fully available.

APPENDIX

init.sql

```
-- SQL file for modifying demo.db to launch an OData Producer for
-- Apache White paper

-- Allow the OData Producer to use the UPDATER user to connect
-- WITHOUT A PASSWORD!  Normally only do this with ReadOnly=true producers
-- and users that have very little access permissions.
GRANT VERIFY ODATA TO UPDATER;

-- Create an OData Producer
CREATE OR REPLACE ODATA PRODUCER ApacheDemo
    AUTHENTICATION USER UPDATER
    MODEL FROM FILE 'odata.osdl'
    SERVICE ROOT '/ApacheDemo'
    USING 'ReadOnly=true'
    ;

-- To undo the above, uncomment and run the following SQL
-- DROP ODATA PRODUCER IF EXISTS ApacheDemo;
-- REVOKE VERIFY ODATA FROM UPDATER;
```

odata.osdl

```
service namespace "HttpsSample" {
    "GROUPO"."Employees";
    "GROUPO"."Departments";
    "GROUPO"."Customers";
    "GROUPO"."SalesOrders";
    "GROUPO"."SalesOrderItems";
    "GROUPO"."Products";
}
```

salesOrders.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/strict.dtd">

<html>
<head>
    <title>Apache Sales Orders Sample</title>
    <script type='text/javascript' src='salesorders.js'></script>
    <script type='text/javascript' src='datajs-1.1.2.js'></script>
    <style type="text/css">
         .padding
         {
             height: 30px;
             clear: both;
         }
         .container
         {
             width: 100%;
         }
         .selectList
         {
             float: left;
             width: 250px;
         }
         .selectList select
         {
             width: 200px;
         }
         .display
         {
             width: 400px;
             float: left;
             border: 1px solid #000;
             background-color: #ddd;
         }
         .display dl
         {
             width: 100%;
             padding: 10px;
             margin: 0px;
             float: left;
             font-size: 85%;
         }
         .display dl dt
         {
             width: 100px;
             clear: left;
             float: left;
             font-weight: bold;
         }
         img
         {
             border: 1px solid #000;
             width: 100px;
             height: 100px;
             position: relative;
             top: 10px;
             left: -110px;
             float: left;
         }
    </style>

</head>

<body onLoad="loadCustomers()">

    <div class="container">
         <div class="selectList" id="customerSelectDiv">
             <select size=10 id="customerSelect" onChange="changeSelectedCustomer( this.selectedIndex );">
             </select>
         </div>

         <div class="display" id="customerDisplay">
             <dl id="customerDetails">
             </dl>
         </div>
    </div>

    <div class="padding"></div>
```

```
    <div class="container">
         <div class="selectList" id="orderSelectDiv">
             <select size=5 id="orderSelect" onChange="changeSelectedSalesOrder( this.selectedIndex );">
             </select>
         </div>

         <div class="display" id="orderDisplay">
             <dl id="orderDetails">
             </dl>
         </div>
    </div>

    <div class="padding"></div>

    <div class="container">
         <div class="selectList" id="orderItemSelectDiv">
             <select size=5 id="orderItemSelect" onChange="changeSelectedSalesOrderItem( this.selectedIndex );">
             </select>
         </div>

         <div class="display" id="orderItemDisplay">
             <dl id="orderItemDetails">
             </dl>
         </div>
         <img id="itemPhoto">
    </div>

</body>
</html>
```

salesOrders.js

```
var customerCache  = [];
var orderCache     = [];
var orderItemCache = [];

var customerDisplayProps    = [ "ID", "GivenName", "Surname",
                                "Street", "State", "PostalCode",
                                "Phone", "Company" ];

var ordersDisplayProps      = [ "ID", "Region", "OrderDate" ];

var orderItemsDisplayProps  = [ "ID", "Region", "OrderDate" ];

function loadCustomers() {
    var uri = "/odata/Customers";
    OData.read( { requestUri : uri }, populateCustomerList, error );
}

function loadSalesOrders( selectedCustomerIndex ) {
    OData.jsonHandler.recognizeDates = true;
    if( selectedCustomerIndex >= 0 ) {
         var customerID = customerCache[selectedCustomerIndex]["ID"];
         var uri = "/odata/Customers(" + customerID + ")?$expand=SalesOrders";
         OData.read( { requestUri : uri }, populateSalesOrderList, error );
    }
}

function loadSalesOrderItems( selectedOrderIndex ) {
    if( selectedOrderIndex >= 0 ) {
         var orderID = orderCache[selectedOrderIndex]["ID"];
         var uri = "/odata/SalesOrders(" + orderID + ")?$expand=SalesOrderItems/FK_ProductID_ID";
         OData.read( { requestUri : uri }, populateSalesOrderItemsList, error );
    }
}



function populateCustomerList( response, request ) {
    var customerSelect = document.getElementById( 'customerSelect' );

    // Remove any existing options
    customerSelect.options.length = 0;

    // Update the customer cache with the new customer list
    customerCache = response.results;

    // Add the new customer list to the select input
    for( var i=0; i<customerCache.length; i++ ) {
         var customerName = customerCache[i]['GivenName'] + ' ' + customerCache[i]['Surname'];
         customerSelect.options[i] = new Option( customerName, i, i==0, i==0 );
    }

    changeSelectedCustomer( customerSelect.selectedIndex );
}

function populateSalesOrderList( response, request ) {
    var orderSelect = document.getElementById( 'orderSelect' );

    // Remove the existing options
    orderSelect.options.length = 0;

    // Update the order cache with the new order list
    orderCache = response.SalesOrders.results;

    // Add the new order list to the select input
    for( var i=0; i<orderCache.length; i++ ) {
         var orderID = orderCache[i]['ID'];
         orderSelect.options[i] = new Option( 'OrderID: ' + orderID, i, i==0, i==0 );
    }

    changeSelectedSalesOrder( orderSelect.selectedIndex );
}

function populateSalesOrderItemsList( response, request ) {
    var itemSelect = document.getElementById( 'orderItemSelect' );

    // Remove the existing options
    itemSelect.options.length = 0;
```

```
    // Update the order cache with the new order list
    orderItemCache = response.SalesOrderItems.results;

    // Add the new order list to the select input
    for( var i=0; i<orderItemCache.length; i++ ) {
         var lineNum = orderItemCache[i]['LineID'];
         itemSelect.options[i] = new Option( 'Line ID : ' + lineNum, i, i==0, i==0 );
    }

    changeSelectedSalesOrderItem( itemSelect.selectedIndex );
}



function changeSelectedCustomer( selectedCustomerIndex ) {
    if( selectedCustomerIndex >= 0 ) {
         var customer = customerCache[ selectedCustomerIndex ];
         var detailsView = document.getElementById( "customerDetails" );
         detailsView.innerHTML = ""
                  + "<dt>ID:</dt>"
                  + "<dd>" + customer.ID + "</dd>"
                  + "<dt>Name:</td>"
                  + "<dd>" + customer.GivenName + " " + customer.Surname + "</dd>"
                  + "<dt>Address:</td>"
                  + "<dd>" + customer.Street + ", " + customer.City + ", "
                           + customer.State + "</dd>"
                  + "<dt>Phone:</td>"
                  + "<dd>" + customer.Phone + "</dd>"
                  + "<dt>Company:</td>"
                  + "<dd>" + customer.CompanyName + "</dd>"
                  ;

         loadSalesOrders( selectedCustomerIndex );
    }
}

function changeSelectedSalesOrder( selectedOrderIndex ) {
    if( selectedOrderIndex >= 0 ) {
         var order = orderCache[ selectedOrderIndex ];
         var detailsView = document.getElementById( "orderDetails" );
         detailsView.innerHTML = ""
                  + "<dt>Order ID:</dt>"
                  + "<dd>" + order.ID + "</dd>"
                  + "<dt>Region:</td>"
                  + "<dd>" + order.Region + "</dd>"
                  + "<dt>Order Date:</td>"
                  + "<dd>" + order.OrderDate + "</dd>"
                  ;

         loadSalesOrderItems( selectedOrderIndex );
    }
}

function changeSelectedSalesOrderItem( selectedOrderItemIndex ) {
    if( selectedOrderItemIndex >= 0 ) {
         var item = orderItemCache[ selectedOrderItemIndex ];
         var detailsView = document.getElementById( "orderItemDetails" );
         detailsView.innerHTML = ""
                  + "<dt>Line ID:</td>"
                  + "<dd>" + item.LineID + "</dd>"
                  + "<dt>Quantity:</td>"
                  + "<dd>" + item.Quantity + "</dd>"
                  + "<dt>Name:</td>"
                  + "<dd>" + item.FK_ProductID_ID.Name + "</dd>"
                  + "<dt>Description:</td>"
                  + "<dd>" + item.FK_ProductID_ID.Description + "</dd>"
                  + "<dt>Size:</td>"
                  + "<dd>" + item.FK_ProductID_ID.Size + "</dd>"
                  + "<dt>Color:</td>"
                  + "<dd>" + item.FK_ProductID_ID.Color + "</dd>"
                  + "<dt>Unit Price:</td>"
                  + "<dd>$" + item.FK_ProductID_ID.UnitPrice + "</dd>"
                  ;

         var itemPhoto = document.getElementById( "itemPhoto" );
         itemPhoto.src = " " + item.FK_ProductID_ID.Photo;
    }
}
```

```
function error( err ) {
    alert( 'An error occurred: ('
                    + err.response.statusCode + ') '
                    + err.response.body );
}
```

www.sap.com