Deploying the VAMDC Query Store Web Service

TABLE OF CONTENT

1 Introduction

This document describes how to set up a running VAMDC Query Store from scratch.

In what follow we are going to describe our procedure in case where the host is a Linux server. The described procedure may be easily adapted to other systems.

2 Prerequisites

The VAMDC Query Store is a set of web services built over the Java Servlet technology. This web services uses an internal database for persisting the processed data and information. Before starting the deployment two elements should be installed on the server(s):

- A servlet container server.
  We chose the open source Apache Tomcat server (http://tomcat.apache.org). Install the latest version of the Tomcat server. Let *tomcat_home* be the path of the directory where tomcat is installed.
- A SQL database engine.
  We chose the MySQL server [1] (https://www.mysql.com/fr/). Install the database engine and create a user account with permission for creating a new database, creating and deleting tables.

Remark: the servlet container and the database may be hosted on two different servers.

3 Deploying and configuring the Query Store services.

Download (or clone) the QueryStore project from the GitHub repository https://github.com/VAMDC/QueryStore.
Import the freshly downloaded project into you favourite Java IDE (we will adopt Eclipse hereafter, https://eclipse.org). Give this project the name "QueryStore".

[1] An alternative and immediately working solution consists in using MariaDB (https://mariadb.org). This database engine is a fork of MYSQL and is an open community based project.

3.1 The database configuration

Run the script *QueryStore.sql* (https://github.com/VAMDC/QueryStore/tree/master/DBScripts) for creating the database on the SQL server.

Edit the table *TechConfig*. This table contains some technical parameters for operating the Query Store services. Add a line as follows

1. Put into the ServletContainerAdress column "Service_URL/QueryStore", where Service_URL is the url users may enter for reaching the tomcat server. If the Tomcat is not fronted by any web server, the service URL is typically http://serverName:8080. If the server is fronted by a web server, refer to your own reverse proxy configuration.
2. Put into the AbsoluteDataPath column the absolute path of the directory where data-files will be downloaded (cf. paragraph 3.1.4 of the Implementing Note[2])
3. Put into the AbsoluteConfigPath column the absolute path of the directory where the file defining the XSL transformation[3] (cf. paragraph 3.1.5 of the Implementing Note) on data will be stored.
4. Put into the secret column the password for using the QueryStore service (cf. section 3 of the Implementing Note).

Edit the table *AuthorizedFields*. For each field you would like to introduce restriction on (cf. section 2 of the Implementing Note) add a line as follows

1. Put into the fieldNames column the name of the field you would like to restrict.
2. Put into the authorizedValues column the restriction.

For example, if you entered the two following lines

- "accededResource"; "database1"
- "accededResource" ; "database2"

Then the query store will accept only the notifications where the parameter "accededResource" is equal to "database1" or "database2", and reject the other configurations.
The set of fields that may be restrained are recalled in the Implementing Note (first item of the section 2).

[2] https://github.com/VAMDC/QueryStore/blob/master/documentation/ImplementingNote.pdf

[3] https://github.com/VAMDC/QueryStore/blob/master/config/XsamsToBibtex.xsl

Edit the table AuthorizedParameters by adding the names of parameters you would like to restrict: if this table is empty, all the parameters are accepted. If this table has entries, only the parameters whose name is in the table are accepted.

3.2 Configuring the web-application

Open the Query Store project with the Eclipse IDE.

1. Edit the DBConnectionBuilder class of the package org.rda.QueryStore.dao
    a. If the servlet container and the database engine are installed on the same server, the value of the String configuration (at line 9) must not be modified. If the servlet container and the database are on different servers, change the address "127.0.0.1" with the address of the server hosting the database engine (make sure that the route is open for reaching the database on port 3306).
    b. Change the value of the Strings dbUser /dbPassword by putting the credential of the database user previously created (cf. paragraph 3.1).

2. Clean and recompile the project.

3. Left click on the project and choose the export option.

4. A popup window will appear. Choose "Web" menu and select "WAR file".

5. Click next. Choose the "QueryStore" as web project to export and chose the destination where the QueryStore.war file will be created.

6. Copy the freshly created war file on the server side. Put the war file into the directory *tomcat_home*/*webapps*

7. Copy the file defining the XSL transformation[4] (cf. paragraph 3.1.5 of the Implementing Note) into the folder you previously defined in the column AbsoluteConfigPath of the table *TechConfig* (cf. paragraph 3.1)

8. Run the tomcat server by executing the command *tomcat_home*/*startup.sh*.

9. Go into the directory *tomcat_home*/webapp/QueryStore/. Create a symbolic link between this folder and the folder entered into the AbsoluteDataPath column of the table *TechConfig* (cf. paragraph 3.1).

[4] https://github.com/VAMDC/QueryStore/blob/master/config/XsamsToBibtex.xsl

10. The Query Store is ready.

3.3 Configuring the purge Daemon

For installing the purger Daemon (cf. Section 5 of the Implementing Note):

1. Make sure you have done the stemps 1 and 2 of the paragraph 3.2.

2. Left click on the project and choose the export option.

3. A popup window will appear. Choose "Java" menu and select "Runnable JAR File".

4. Click next; In the Launch configuration choose-box, select the option corresponding to the Purger Class. In the "Export destination" enter the path were the jar file will be created. Select also the "Package required libraries into generated JAR file" option. Click on the "finish" button.

5. Copy the freshly generate jar file on the server hosting the tomcat server. Let *jar_file_path* be the absolute path of the folder where the jar file is copied server-side (the choice of the directory destination is free).

6. Add the following line in the crontab for triggering the execution of the purge process (in the example, each day at midnight):

   *0 0 * * * java -jar /jar_file_path/purger.jar >> /jar_file_path/purger.log*

4 Overview of the service endpoints:

With the configuration described in this document:

- The NotificationListener service (cf. section 2 of the Implementing Note) is reachable at the endpoint Service_URL/QueryStore/NotificationListener. We recall that Service_URL has been defined in paragraph 3.1.

- The AssociationService (cf. paragraph 4.1 of the Implementing Note) is reachable at the endpoint Service_URL/QueryStore/AssociationService.

- The PortalAssociationService (cf. paragraph 4.2 of the Implementing Note) is reachable at the endpoint Service_URL/QueryStore/PortalAssociationService.

- The InfoQuery service (cf. paragraph 4.3 of the Implementing Note) is reachable at the endpoint Service_URL/QueryStore/InfoQuery.

- The FindQueriesByTime service (cf. paragraph 4.4 of the Implementing Note) is reachable at the endpoint Service_URL/QueryStore/FindQueries.

- The getUUIDByToken service (cf. paragraph 4.5 of the Implementing Note) is reachable at the endpoint Service_URL/QueryStore/GetUUIDByToken.