

---

# **Db2 Web Query (REST based) Application Extension**

## **Usage Instructions**



*Updated October 07, 2020*

---

<b>1</b>	<b>Overview .....</b>	<b>4</b>
<b>2</b>	<b>Initial Setup.....</b>	<b>5</b>
<b>3</b>	<b>Using the Extension.....</b>	<b>6</b>
<b>3.1</b>	<b>Modes of use.....</b>	<b>6</b>
3.1.1	Browse Mode .....	6
3.1.2	Direct Mode.....	10
<b>3.2</b>	<b>Report Parameter Considerations .....</b>	<b>15</b>
3.2.1	Specifying Report Parameters .....	15
3.2.2	Omitting Report Parameters .....	15
<b>3.3</b>	<b>Authentication Methods.....</b>	<b>17</b>
3.3.1	Basic Authentication.....	17
3.3.2	URL Parameter Authentication .....	18
3.3.3	Static Authentication .....	18
3.3.4	Application Authentication.....	19
<b>3.4</b>	<b>Dynamic Run Time Environment support .....</b>	<b>23</b>
<b>3.5</b>	<b>Restrictions .....</b>	<b>23</b>
<b>4</b>	<b>Properties file and options.....</b>	<b>25</b>
<b>4.1</b>	<b>Server .....</b>	<b>25</b>
<b>4.2</b>	<b>Port.....</b>	<b>25</b>
<b>4.3</b>	<b>Basic Authentication .....</b>	<b>25</b>
<b>4.4</b>	<b>User/Password.....</b>	<b>26</b>
<b>4.5</b>	<b>Browse Mode .....</b>	<b>26</b>
<b>4.6</b>	<b>Cache Timeout.....</b>	<b>26</b>
<b>4.7</b>	<b>Token Timeout .....</b>	<b>26</b>
<b>4.8</b>	<b>Template Heading.....</b>	<b>26</b>
<b>4.9</b>	<b>Template Footing .....</b>	<b>26</b>
<b>4.10</b>	<b>Parameter Prompt Limit .....</b>	<b>26</b>



---

<b>4.11</b>	<b><i>Result Schema .....</i></b>	<b>27</b>
<b>4.12</b>	<b><i>Enable Statistics.....</i></b>	<b>27</b>
<b>4.13</b>	<b><i>Debug Mode .....</i></b>	<b>27</b>
<b>4.14</b>	<b><i>Parameter Prompt.....</i></b>	<b>27</b>
<b>4.15</b>	<b><i>About Link .....</i></b>	<b>27</b>
<b>4.16</b>	<b><i>WQRAX_USER fex.....</i></b>	<b>27</b>
<b>4.17</b>	<b><i>Timeout redirect URL.....</i></b>	<b>28</b>

---

# 1 Overview

The Db2 Web Query REST-based Application Extension, or **WQRAX** for short, is a Db2 Web Query **Standard** Edition feature that allows a Db2 Web Query reporting object to be run and the output presented by invoking a URL. This allows web browsers or other web applications to run and display those reports directly from an application and eliminates the need for users to log in to the Db2 Web Query BI Portal, locate a report, and run it. The reporting object type can be any of the following:

- Report
- Graph
- Dashboard
- Compound document
- HTML page

The basic premise of WQRAX is simple: each reporting object is represented by a specific URL. To run the report from a web application, simply invoke the representative URL.

This document contains the information needed for configuring and integrating the REST based Db2 Web Query Application Extension.

---

## 2 Initial Setup

For Db2 Web Query **Standard** Edition installations, WQRAX is automatically installed with the 5733WQX product and a default configuration is provided. Consequently, it is available for use immediately. Several configuration options do exist to control specific behaviors of WQRAX - these are stored in a configurable properties file and an interface is provided to alter them. For more information on the properties file and the various options, see Properties file and options on page 25.

It is highly recommended to enable secure sockets layer (SSL) for Web Query. When Web Query is configured for SSL, WQRAX uses the Hypertext Transfer Protocol Secure (**HTTPS**). This document assumes the HTTPS scheme for URLs. If Web Query is not enabled for SSL, use HTTP instead.

---

## 3 Using the Extension

WQRAX is an abstraction layer that is based upon the Db2 Web Query REST based web services. Its intention is to simplify the use and integration of the web services, but it does require a basic understanding of the modes of use and the various ways to authenticate the user. These topics are discussed in this section.

### 3.1 Modes of use

WQRAX has two modes of use: browse and direct. Each is explained in the following sections.

#### 3.1.1 Browse Mode

All reporting object content is physically stored in the Db2 Web Query repository. However, from a logical, end-user interface perspective, these objects are organized and stored in either top level folders or sub-folders – in a tree-like structure. End users can find these objects by “traversing the tree” and navigating to the folder that contains the objects.

The browse mode of WQRAX provides an interface for users to traverse the tree and browse the reporting object content without knowing the specific URL for a top level folder, subfolder, or report. This mode displays the contents of the folder to users, allowing them to browse the contents of the folder. They can then simply click on a report to run it or they can click on a subfolder to jump to that level and show the sub folder contents (which could be reports and/or more sub folders). As with the BI Portal interface, only content that the user is authorized to see is shown.

You must have browse mode enabled in the properties file. See the section **Browse Mode** for more details.

To begin working in browse mode, simply open the following URL in your web browser:

**https://<yourserver>:12331/wqrax**

where **<yourserver>** is your IBM i's IP address or DNS name.

If Web Query

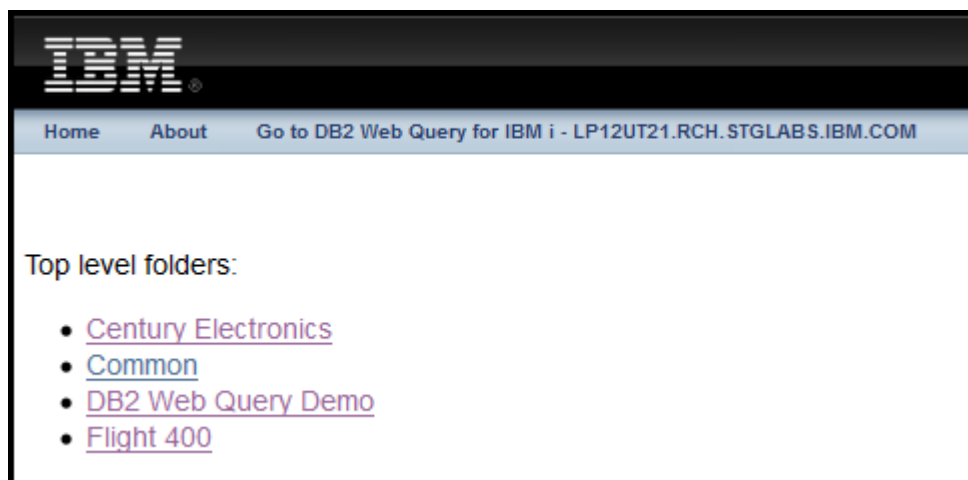
**Note:** Notice that this link is similar to the one used to log in to the Db2 Web Query BI Portal which is the following:

<https://<yourserver>:12331/webquery>

The same port number (12331) is used for both, only the last portion is different.

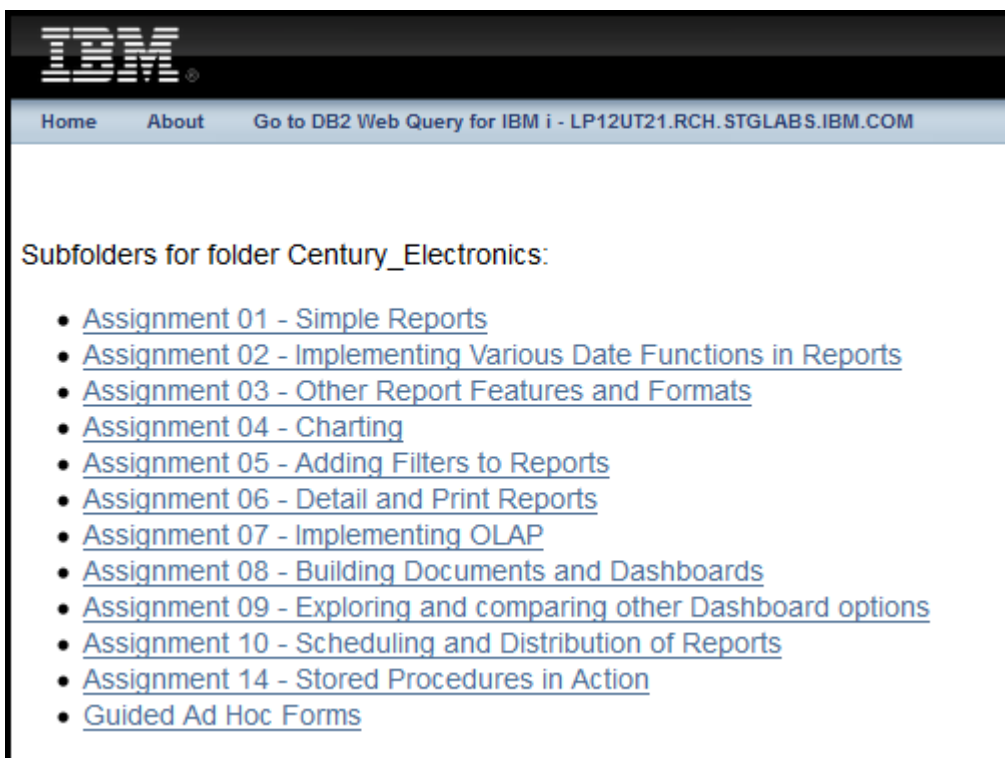
With the installation default configuration, you are prompted to log in with your IBM i user profile and password. You must be either a registered licensed Web Query user or a member of a registered run time group. For more information on authentication, please see section **Authentication Methods**.

The security implementation is consistent with the BI Portal interface. Once you are logged in, all the top level folders you are authorized to are shown. An example is shown in Figure 1 below.



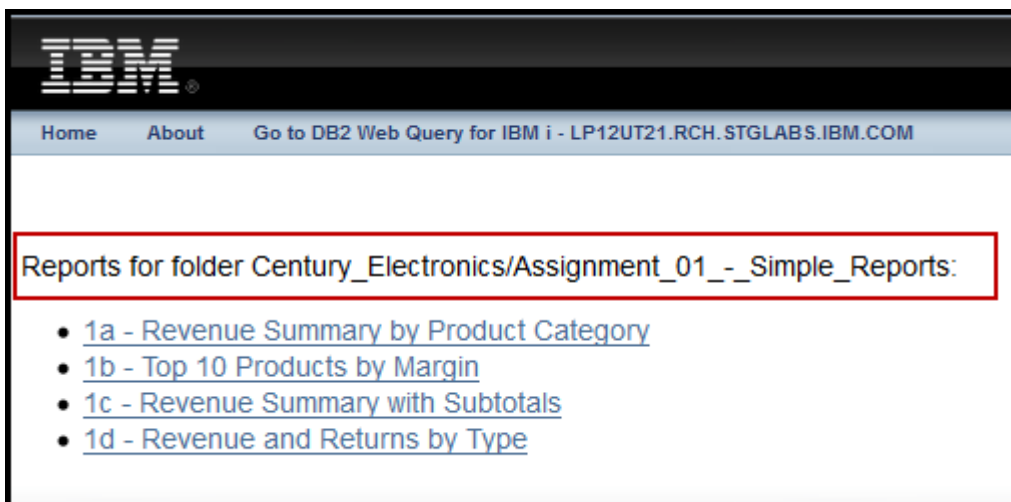
**Figure 1 - WQRAX top level folder**

Simply click on a top level folder to see a list of its contents. This could be subfolders and/or reporting objects. Below you see the result of selecting the Century Electronics top level folder. At this level we see all the subfolders under Century Electronics.



**Figure 2 - WQRAX Subfolder level**

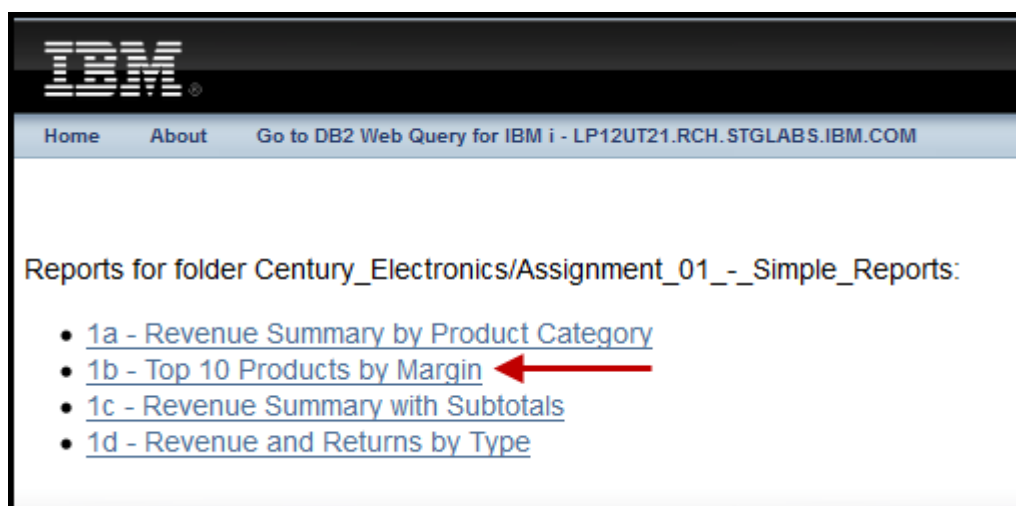
By clicking on a sub folder you can jump to that level to see its contents. In Figure 3 below, we have selected the "Assignment 01 – Simple Reports" subfolder. Notice that when browsing folders, the current level is displayed at the top.



**Figure 3 - WQRAX reporting objects in a subfolder**

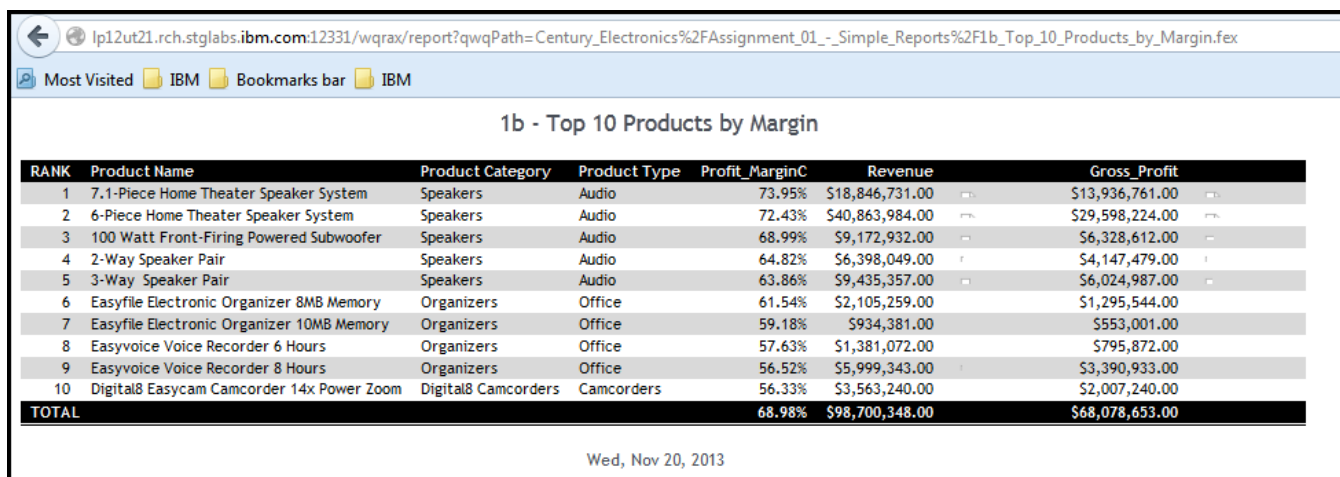
If the folder contains reports, charts, dashboards, or documents, they are displayed. To run a report, simply click it as shown in Figure 4.





**Figure 4 - Select a report to run**

The report runs and, if the report output is HTML or Active Reports, the results are displayed in the browser.



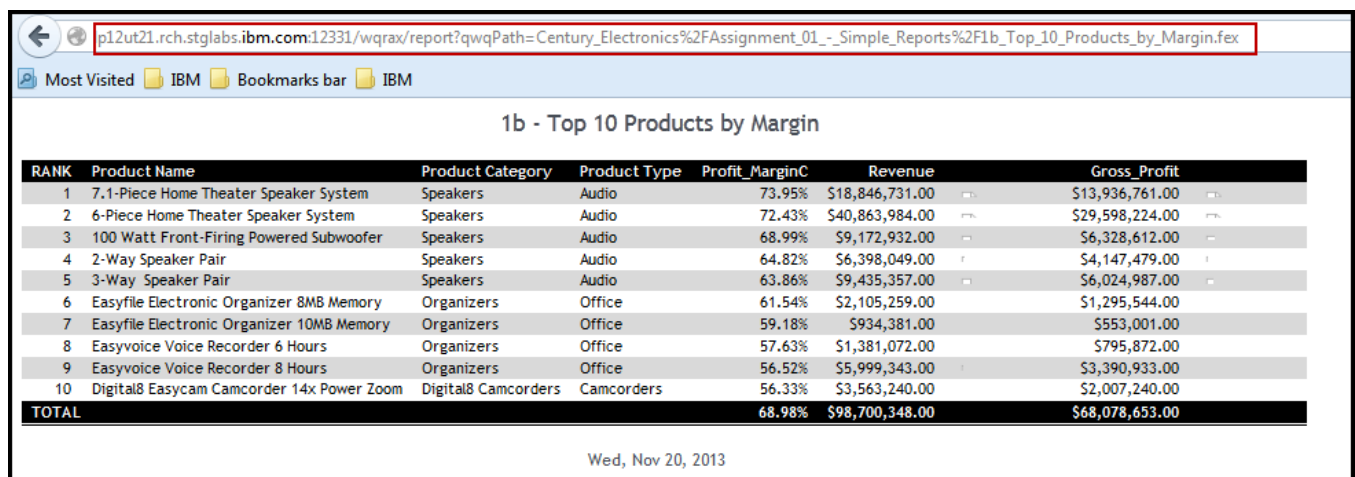
RANK	Product Name	Product Category	Product Type	Profit_MarginC	Revenue	Gross_Profit
1	7.1-Piece Home Theater Speaker System	Speakers	Audio	73.95%	\$18,846,731.00	\$13,936,761.00
2	6-Piece Home Theater Speaker System	Speakers	Audio	72.43%	\$40,863,984.00	\$29,598,224.00
3	100 Watt Front-Firing Powered Subwoofer	Speakers	Audio	68.99%	\$9,172,932.00	\$6,328,612.00
4	2-Way Speaker Pair	Speakers	Audio	64.82%	\$6,398,049.00	\$4,147,479.00
5	3-Way Speaker Pair	Speakers	Audio	63.86%	\$9,435,357.00	\$6,024,987.00
6	Easyfile Electronic Organizer 8MB Memory	Organizers	Office	61.54%	\$2,105,259.00	\$1,295,544.00
7	Easyfile Electronic Organizer 10MB Memory	Organizers	Office	59.18%	\$934,381.00	\$553,001.00
8	Easyvoice Voice Recorder 6 Hours	Organizers	Office	57.63%	\$1,381,072.00	\$795,872.00
9	Easyvoice Voice Recorder 8 Hours	Organizers	Office	56.52%	\$5,999,343.00	\$3,390,933.00
10	Digital8 Easycam Camcorder 14x Power Zoom	Digital8 Camcorders	Camcorders	56.33%	\$3,563,240.00	\$2,007,240.00
<b>TOTAL</b>				<b>68.98%</b>	<b>\$98,700,348.00</b>	<b>\$68,078,653.00</b>

Wed, Nov 20, 2013

**Figure 5 - Report is run and results displayed**

In general, once a report is selected, the user is prompted to enter any necessary parameter values, then the report is run and the output is shown in the output format defined in the report. This could be HTML back to the browser, a PDF file, or an Excel file.

At any time during the tree navigation process, notice that the URL bar contains the top level folder, subfolder, and report as shown in Figure 6 below. This is a key concept when using the other browse method: *Direct Mode*.



RANK	Product Name	Product Category	Product Type	Profit_MarginC	Revenue	Gross_Profit
1	7.1-Piece Home Theater Speaker System	Speakers	Audio	73.95%	\$18,846,731.00	\$13,936,761.00
2	6-Piece Home Theater Speaker System	Speakers	Audio	72.43%	\$40,863,984.00	\$29,598,224.00
3	100 Watt Front-Firing Powered Subwoofer	Speakers	Audio	68.99%	\$9,172,932.00	\$6,328,612.00
4	2-Way Speaker Pair	Speakers	Audio	64.82%	\$6,398,049.00	\$4,147,479.00
5	3-Way Speaker Pair	Speakers	Audio	63.86%	\$9,435,357.00	\$6,024,987.00
6	Easyfile Electronic Organizer 8MB Memory	Organizers	Office	61.54%	\$2,105,259.00	\$1,295,544.00
7	Easyfile Electronic Organizer 10MB Memory	Organizers	Office	59.18%	\$934,381.00	\$553,001.00
8	Easyvoice Voice Recorder 6 Hours	Organizers	Office	57.63%	\$1,381,072.00	\$795,872.00
9	Easyvoice Voice Recorder 8 Hours	Organizers	Office	56.52%	\$5,999,343.00	\$3,390,933.00
10	Digital8 Easycam Camcorder 14x Power Zoom	Digital8 Camcorders	Camcorders	56.33%	\$3,563,240.00	\$2,007,240.00
<b>TOTAL</b>				<b>68.98%</b>	<b>\$98,700,348.00</b>	<b>\$68,078,653.00</b>

Wed, Nov 20, 2013

**Figure 6 - Report URL is shown**

### 3.1.2 Direct Mode

If you know the specific URL of the report you wish to run, you can invoke that URL directly. This eliminates the need for users to traverse the tree in browse mode and allows them to run the report directly from an application. The structure of direct mode is consistent with that of browse mode – the only difference is that you specify the “full path” of the report as the value of the qwqPath construct in the URL. For example, to run the report with the full path value of “My\_Top\_Level\_Folder/My\_Report.fex” on your server, specify the following URL:

[https://your\\_server:12331/wqrx/report?qwqPath=My\\_Top\\_Level\\_Folder/My\\_Report.fex](https://your_server:12331/wqrx/report?qwqPath=My_Top_Level_Folder/My_Report.fex)

Once you have constructed the URL, you can use that string directly in various implementations. Examples of this include:

- Link to it from a web page or PC application.
- Set it as the source of an HTML iframe (in-line frame) on an existing web page.
- Email the link to your colleagues.

Keep in mind that since it is a URL, every end user will need to be able to have access to the IBM i and be a valid Db2 Web Query user or member of a registered runtime group. In a corporate environment, this could mean establishing a VPN connection or gaining clearance to a firewall that may be protecting your server.

The URL is static and persistent – this means that, as long as the report still exists in the same folder structure and the user has authority to it, it can always be used to run the

---

report. The URL can be bookmarked, copied/pasted into another web application, or sent as a link in an email.

## **Determining the URL of a Report**

In order to implement direct mode in an application or a browser, you of course need to know the specific reporting object URL ahead of time. There are two ways of obtaining the necessary information to construct this URL:

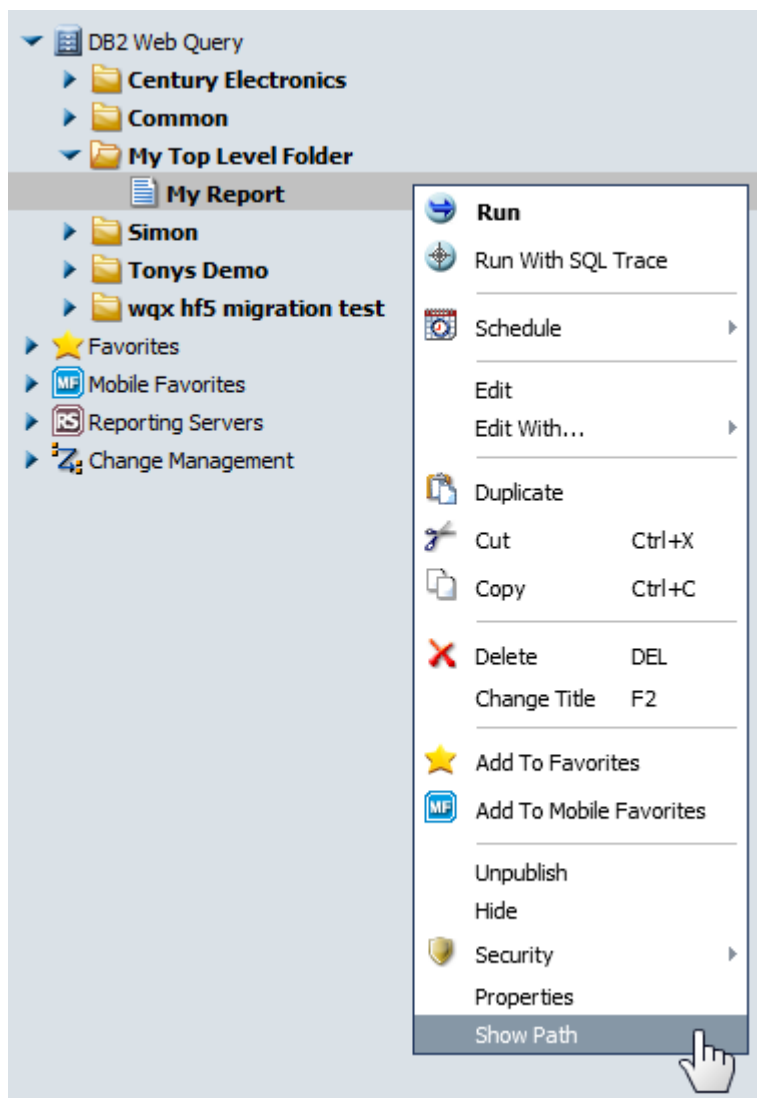
- Use browse mode to find report and record/save the URL
- Use information from BI Portal to determine the report's fullpath and plug that into the URL

### **Browse Mode to Obtain URL**

As mentioned previously, the URL of a report or folder can be obtained while using browse mode. As you select the folder or report in browse mode, the specific URL for that folder or report is available in the browser's address bar. An example was shown previously in Figure 6.

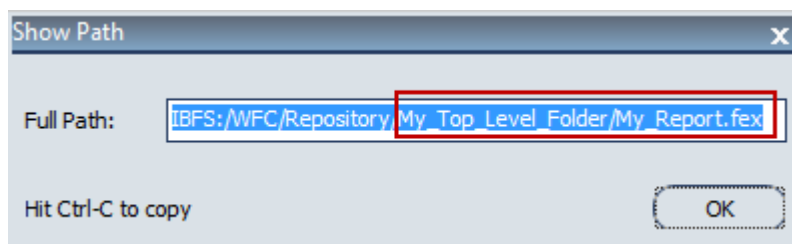
### **Show Path Option in BI Portal**

Another technique to determine the URL is to use the show path option from the BI Portal to obtain the report's full path. Log into the BI Portal (<https://<yoursystem>:12331/webquery>), find the report, and select the **Show Path** option from the right click menu as shown.



**Figure 7 - Show Path option**

The path is presented in a dialog window as shown.



**Figure 8 - Full Path is displayed**

Only the portion after the "IBFS:/WFC/Repository/" string is required. Select this portion and press Ctrl-C to copy it into your clipboard. Then paste it into the URL following the ?qwqPath= portion of the WQRAX URL. When you are finished, the URL structure should look something like this:

`https://<yourserver>:12331/wqrax/report?qwqPath=My_Top_Level_Folder/My_Report.fex`

## Short URL Implementation

A slightly different twist to the Direct Mode is a feature called the WQRAX Short URL. This feature essentially provides a hashing algorithm to convert the report's fullpath to a shorter (12 character) string that can be used in the URL in direct mode instead of the full URL. This can optionally include parameters passed in the URL.

This implementation is useful when you need a shorter URL (as is the case when launching the browser from the STRPCCMD command) or if you have a need to obfuscate the longer URL to something more cryptic.

For example, the following long URL:

[https://your\\_server:12331/wqrax/report?qwqPath=My\\_Top\\_Level\\_Folder%2FMy\\_Report.fex&COUNTRY=Canada&PRODUCTTYPE=Office](https://your_server:12331/wqrax/report?qwqPath=My_Top_Level_Folder%2FMy_Report.fex&COUNTRY=Canada&PRODUCTTYPE=Office)

Can be substituted with this shortened version:

[https://your\\_server:12331/wqrax/reports?id=r3skSZblbgvj](https://your_server:12331/wqrax/reports?id=r3skSZblbgvj)

## Determining Short URL

To determine the short URL of a WQRAX long URL, call the procedure GENERATEID in the service program QWEBQRY/QWQGNHURL.

The prototype for the GENERATEID procedure is as follows:

```
int GENERATEID
    (const unsigned int port,
     const char* uri,
     char* outputID)
```

Example RPGLE prototype:

d GENERATEID	pr	10i 0
d portNumber		10u 0 value
d uriPtr	*	value options(*string:*trim)
d shortID		20a

---

Example RPGLE code snippet to pass in long URL and receive short URL from GENERATEID procedure:

```
portNumber = 12336;  
uriString = '/wqrax/report?&qwqPath= My_Top_Level_Folder/My_Report.fex ';  
%str(uriPtr:urisize) = uriString;  
ReturnCode = GENERATEID(portnumber : uriPtr : shortID);  
shortID=%trim(shortID);  
shortURL = 'https://my_server:12331/wqrax/reports?id='+ shortID;
```

Note that when the user passes the port number to the GENERATEID procedure, that port number should be 12336. But when the user gets the output short url ID and wants to use that in the url, the port number in the url should be 12331.

## 3.2 Report Parameter Considerations

Many reports have filters to allow end users to select a subset of the data at runtime. When those reports run, end users are prompted for parameters so that they can specify the value for those filters. When such reports are invoked using WQRAX, the parameters can either be specified in the URL or they can be omitted entirely from the URL. If omitted, WQRAX will automatically build the necessary prompt controls and prompt the user to specify/select the missing values. This is the case for both browse and direct mode.

### 3.2.1 Specifying Report Parameters

To specify parameter values in the WQRAX URL:

1. Determine the name of each parameter
2. Append an ampersand (&) character prefix at the beginning of each parameter name
3. At the end of the URL, specify the parameter name (with the ampersand prefix), followed by the equals character (=), followed by the parameter value. In the following example we specify the values "United States" for the COUNTRY parameter and "Office" for the PRODUCTYPE parameter:

#### Example:

`https://your_server:12331/wqrax/report?qwqPath=My_Top_Level_Folder%2FMy_Report.fex&COUNTRY=United States&PRODUCTYPE=Office`

**Note:** Global Variables (those with && preceding them) are not officially supported within WQRAX.

### 3.2.2 Omitting Report Parameters

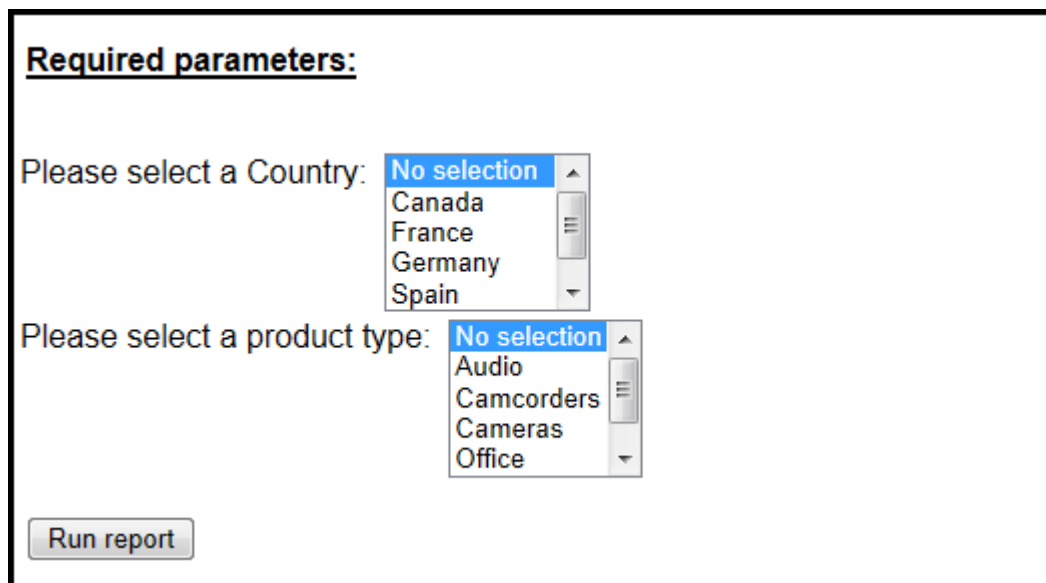
If a report has parameters that are not specified in the URL, WQRAX detects this and automatically handles the building and presentation of the necessary prompt controls. The user will be prompted using the same method as the report run through the Db2 Web

Query BI Portal: via drop down, selection list, radio button, text box, etc. An example is shown below where the user is prompted after selecting a report with filters for Country and Product Type.

**Example:**

[https://your\\_server:12331/wqrax/report?qwqPath=My\\_Top\\_Level\\_Folder%2FMy\\_Report.fex](https://your_server:12331/wqrax/report?qwqPath=My_Top_Level_Folder%2FMy_Report.fex)

When the URL is submitted, WQRAX detects and builds the necessary prompt controls for any parameters not specified in the URL, as shown by the example in Figure 9:



The screenshot displays a web interface titled "Required parameters:". It contains two prompts: "Please select a Country:" and "Please select a product type:". Each prompt is followed by a dropdown menu. The "Country" dropdown shows options: "No selection", "Canada", "France", "Germany", and "Spain". The "Product type" dropdown shows options: "No selection", "Audio", "Camcorders", "Cameras", and "Office". At the bottom left of the form is a button labeled "Run report".

**Figure 9 - Prompt controls built and displayed**

**Note:** Global Variables (those with && preceding them) are not officially supported within WQRAX. Global variables may not be detected by WQRAX and could cause the report to fail to run.



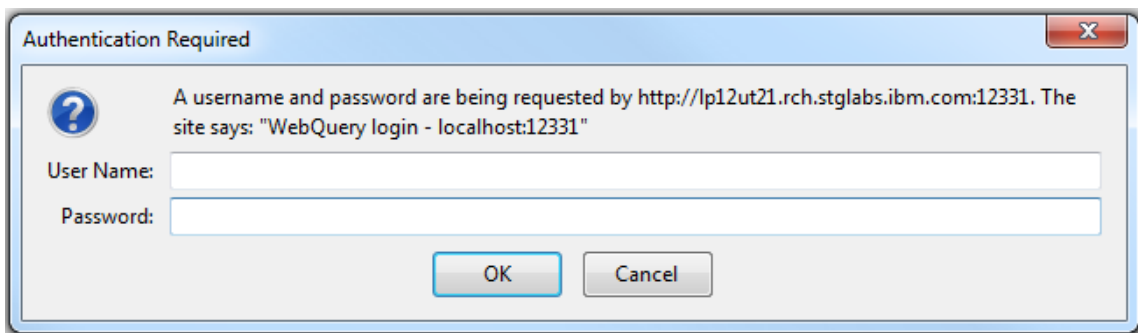
## 3.3 Authentication Methods

As is the case when running reports in the Db2 Web Query BI Portal, users of WQRAX must identify themselves to Db2 Web Query through an authentication process. WQRAX provides four distinct methods of authentication:

- Basic authentication
- URL parameter authentication
- Static authentication
- Application specified authentication

### 3.3.1 Basic Authentication

Basic authentication simply prompts users for their IBM i credentials the first time they attempt to use WQRAX in the current browser session. An example prompt is shown below.



The user would provide their IBM i user profile and password. The user profile must be either a registered licensed Web Query user or a member of a registered run time group. Once authenticated, the necessary information is encrypted and passed along in the HTTP header of subsequent WQRAX requests in the same browser session. This avoids users being prompted for their credentials for each WQRAX request – it is only done once per session. If you close your browser and re-open it, you will be prompted to enter your credentials again.

Basic Authentication is the default (shipped) configuration. It is enabled by specifying the following in the WQRAX properties file:

***wqraxBasicAuthEnabled=true***

**Note:** When basic authentication is enabled, it is always enforced and users will always be prompted for their credentials. If you wish to use another method of authentication, you must first disable basic authentication.

### 3.3.2 URL Parameter Authentication

The user profile and password values can be specified as parameters in the URL when a report is requested through WQRAX. The parameter names are **&wqraxUser** and **&wqraxPassword** (the ampersand prefix denotes they are parameters and is required). For example, to run a report and log in as user profile FRED with password ABC123, the following URL is specified:

[https://your\\_server:12331/wqrax/report?gwqPath=My\\_Top\\_Level\\_Folder%2FM\\_y\\_Report.fex&wqraxUser=FRED&wqraxPassword=ABC123](https://your_server:12331/wqrax/report?gwqPath=My_Top_Level_Folder%2FM_y_Report.fex&wqraxUser=FRED&wqraxPassword=ABC123)

Note: Basic authentication must be disabled for this method of authentication to be accepted. Specify the following in the properties file:

***wqraxBasicAuthEnabled=false***

### 3.3.3 Static Authentication

If you are in a corporate environment where Db2 Web Query is accessed from within a secure intranet configuration and you have no need to distinguish between individual users of WQRAX, you can effectively bypass the log in process. This is accomplished by having each user log in implicitly using a shared IBM i user profile. The shared user profile must also be either a registered licensed Web Query user or a member of a registered run time group. The user profile and password that is used to log all users in is stored in the properties file. Here is an example properties file configuration:

*user=FRED*

*password=ABC123*

In this example, all users of WQRAX are logged in as user FRED – there is no other form of authentication performed. This may be the desired behavior but use with caution. All

users will look the same to the Web Query application. If you are using reports that are dependent upon knowing the true identity of the user, for example reports based upon SQL views with row level filter security, this method is not recommended.

Note: Basic authentication must be disabled for this method of authentication to be accepted. Specify the following in the properties file:

***wqraxBasicAuthEnabled=false***

### 3.3.4 Application Authentication

#### Main Flow

A fourth method of authenticating users is by specifying the credentials from within a web based application before the first WQRAX request is submitted. In this situation, there is a web application that is controlling or directing access into Web Query and the intent is to integrate Web Query reports as part of that overall application.

Note: Basic authentication must be disabled for this method of authentication to be accepted. Specify the following in the properties file:

***wqraxBasicAuthEnabled=false***

In this mode, the application becomes responsible for performing the authentication into Web Query on behalf of the user. The application passes the authentication information through a POST method (for security). This process requires consideration of two aspects:

1. the application is driving the authentication and
2. the application must work with the browser in order to get the session authentication set in the browser on behalf of the user.

When the application needs to authenticate to Web Query (usually the first time the user navigates to the applicable part of the application), the application sends a POST request that includes the wqraxUser and wqraxPassword parameters. The application POSTs the URL request using the getToken method:

---

<https://<yourserver>:12331/wqrax/getToken?&wqraxUser=xxxx&wqraxPassword=yyyy>

Server <yourserver> is the IP or DNS name of the Web Query server, xxxx is the IBM i user profile and yyyy is its password. Again, user xxxx must be either a registered licensed Web Query user or a member of a registered run time group.

The POST will return a token, which is a simple string. We will denote this token as tttt for reference below. Note again that the **getToken** request is only be honored on a POST request for increased security. Attempting to invoke it directly from the browser's URL (a GET request) will fail to return a token.

Once the application receives the token, it must place it on as the &token= parameter of the subsequent URL, placed on the html page, to render the signOn request or the actual WQRAX report to run. When that URL runs the actual authentication is performed and the resulting session information is returned to the browser. This ensures the session authentication is registered in the browser session.

Here is an example html code snippet that shows the token being used in the report request.

```
<form name="RunForm"
  action=
  "https://<yourserver>:12331/wqrax/report?qwqPath=myreports/country_revenue.htm&token=tttttttt" method="post">
  <table cellpadding="0" cellspacing="0">
    <tr>
      <td >
        <input type="submit" value="Run the Report"/>
      </td>
    </tr>
  </table>
</form>
```

A portion of JavaScript (or a reference to JavaScript) could also be added to the html page as well to drive the form to run. A simple JavaScript example to use for the form above would be:

```
<script language="JavaScript" type="text/javascript">
```

---

```
document.RunForm.submit();  
</script>
```

AJAX could be used to run the signOn request asynchronously as well. This may be preferable to make the page experience more seamless, and faster, for the end user.

In either case, when the javascript/form is run it will perform the actual signon authentication under the browser's control. In this way the browser automatically handles the resulting authentication session id, freeing the application from having to deal with the authentication session id for the duration of the session.

Note that the returned token tttt discussed above can be configured for how long it remains valid. This can be controlled by the **tokenTimeout** value in the properties file.

Once the authentication is complete, the user's session has now been authenticated to WQRAX. At this point, the application can show additional desired WQRAX URL on the web page for the user to click on, for example <https://<yourserver>:12331/wqrax> for browse mode or [https://<yourserver>:12331/wqrax/report?qwqPath=top\\_level\\_folder/report.fex](https://<yourserver>:12331/wqrax/report?qwqPath=top_level_folder/report.fex) for a direct mode. In addition, if a report contains drilldowns or launches new pages, WQRAX will automatically pass along the authentication information for those reports as well.

### Optional WQRAX\_USER Parameter

In some Enterprises, the desire is to use a shared user profile to authenticate to Web Query for multiple end users, but still provide some way to uniquely identify each end user within Web Query reports. WQRAX provides this capability through the reserved WQRAX\_USER parameter. WQRAX\_USER is a predefined parameter within the WQRAX environment that can be optionally set and used. It is referenced as a global variable in Web Query synonym and its value is passed 'silently' throughout the WQRAX environment. It provides a way for reports to filter on a 'user', even though a shared user profile is being used.

To use it, the application must set WQRAX\_USER at the same time as when the authentication token is being requested. WQRAX will then associate the given

WQRAX\_USER value as the parameter's value for the duration of the session. To set WQRAX\_USER, the POST URL using getToken shown previously would be augmented as:

https://<yourserver>:12331/wqrax/getToken?&wqraxUser=xxxx&wqraxPassword=yyyy&  
**WQRAX\_USER=zzzz**

where zzzz is the value which a report will receive if it references global variable WQRAX\_USER. The returned token will account for the Web Query user, password, and the WQRAX\_USER value. As with the authentication, if a report contains a drill down to another report, the WQRAX\_USER value will be available to that report as well.

**Note:** in order to use WQRAX\_USER, a one line fex must be created in the Common folder and accessible to all WQRAX users. See the [\*\*About Link\*\*](#)

**This** parameter determines whether the 'About' link shows for the main browser session.

**Example value:** *wqraxShouldAboutLink=true*

*WQRAX\_USER* fex parameter in the properties file for more instructions.

**Note2:** if a report references WQRAX\_USER as a parameter, but the application fails to set WQRAX\_USER during the authentication, the end user will be prompted to enter WQRAX\_USER. This is almost always undesirable, but it provides an easy way to debug when an application fails to set the WQRAX\_USER value correctly.

## 3.4 Dynamic Run Time Environment support

WQRAX supports Dynamic Run Time Environments (RTE's) and will honor the currently active RTE for the logged in user. Therefore if a library list and/or an exit program is configured for the user's active RTE, the library list will be changed and the exit program called before the report is run through WQRAX.

In addition, the user you can dynamically change the active RTE at any time in WQRAX by using the `qwqActiveRuntimeEnvironment` parameter.

Specify `qwqActiveRuntimeEnvironment=your_runtime_environment_name` in the URL as a parameter will change the current active runtime environment to `runtime_environment_name`.

The request to run reports later on will continue to honor the active run time environment you just set.

## 3.5 Restrictions

WQRAX has some restrictions that must be considered during report design and implementation.

- **No InfoMini support**

The WQRAX extension does not support reports that generate InfoMini output. Any reports launched from WQRAX that include InfoMini output will not render properly nor function correctly.

- **No OLAP support**

The WQRAX extension does not support reports that use the OLAP feature. Any reports launched from WQRAX that include OLAP output will not render properly nor function correctly.

- **No global variable support**

The WQRAX extension does not support use of global variables (references preceded with **&&**), with the exception of the predefined `WQRAX_USER` global

---

variable. Any reports launched from WQRAX that include global variables (other than WQRAX\_USER) may cause the report to prompt for the global variable inappropriately and fail to run correctly.



---

## 4 Properties file and options

The properties file is /qibm/userdata/qwebqry/extensions/wqrax.properties. By default, the properties file is set to provide functionality without any configuration needed. However, there are properties you can change with a few simple parameters. In this section, we will describe each setting in the properties file. Each section does have a brief comment included in line prefaced by a hash (#). Web Query provides a web interface at <https://server:12331/wqrax/config> to make changes to this file. You must have \*SECADM special authority in addition to being a Web Query administrator to access this page.

### 4.1 *Server*

This parameter defines the Db2 Web Query server to connect to. By default, it will be localhost, meaning the local system. However, it is possible to have more than one server with Web Query on it and you may want to point the Application Extension running on one server to Db2 Web Query running on the other. Keep in mind that it will still check the license on the system where the Application Extension is running, even if it points to a remote Db2 Web Query server.

**Example value:** *server=localhost*

### 4.2 *Port*

This parameter is the port that Db2 Web Query is running on. This should always be 12331.

**Example value:** *port=12331*

### 4.3 *Basic Authentication*

This parameter determines the WQRAX login credentials approach for each user. By default the setting is **true**. This means the end user will be prompted with an authentication window. Used in conjunction with the **user** and **password** parameters or properties, this can also be set to **false** to use alternate methods for authentication described in the section [Authentication Methods](#).

**Example value:** *wqraxBasicAuthEnabled=true*

---

#### 4.4 User/Password

These parameters define the shared user profile commonly used by a set of users when using the static authentication method provided when **wgraxBasicAuthEnabled=false**. See section **Static Authentication** for more details.

*Example values:*

**user=MYUSRPRF**

**password=myPassword**

#### 4.5 Browse Mode

This parameter determines if browse mode is enabled. By default is it set to true. See section **Browse Mode** for details on browsing. For security reasons, you may choose to disable browse mode by setting this to false and allow only direct access mode. Specify true to enable and false to disable this feature.

*Example value: browse=true*

#### 4.6 Cache Timeout

This parameter determines the number of seconds before any cached report graphs are discarded.

*Example value: cacheTimeout=14400*

#### 4.7 Token Timeout

This parameter determines the number of seconds before the token returned from the **getToken** method becomes invalid.

*Example value: tokenTimeout=7200*

#### 4.8 Template Heading

This parameter controls what is used as the header for each HTML page.

*Example value: wgraxTemplateHead=ibm\_header.html*

#### 4.9 Template Footing

This parameter controls what is used as the footer for each HTML page.

*Example value: wgraxTemplateFoot=ibm\_footer.html*

#### 4.10 Parameter Prompt Limit

This parameter controls the maximum number of parameters that WQRAX will prompt for on a given report.

---

**Example value:** *wqraxSelectListSize=5*

#### 4.11 Result Schema

This parameter controls the target schema (library) for storing saved reports and statistics.

**Example value:** *wqraxSchema=QWQRAX*

#### 4.12 Enable Statistics

This parameter controls whether statistics about WQRAX run reports are kept.

**Example value:** *wqraxEnableStats=false*

#### 4.13 Debug Mode

This parameter determines if debug messages are sent to the servlet log. You may need to add "com.ibm.ejs.ras.level=INFO" into the servlet's tracing specification if it is not already there. Use this option under guidance from a support representative.

**Example value:** *debug=false*

#### 4.14 Parameter Prompt

This parameter determines if the user will be prompted for any parameters not provided on the URL invocation. By default the user will be prompted.

**Example value:** *wqraxPrompt=true*

#### 4.15 About Link

This parameter determines whether the 'About' link shows for the main browser session.

**Example value:** *wqraxShouldAboutLink=true*

#### 4.16 WQRAX\_USER fex

This parameter determines the fex (report) that should be run to set the WQRAX\_USER global variable. By default it is set `rax_user.fex`. The target fex (`rax_user.fex`) must be in the Web Query **Common** folder, must be publicly accessible for WQRAX users, and must be a one line fex with the following statement:

**-SET &&WQRAX\_USER = &INUSER;**

The fex itself is NOT shipped with the product. It MUST be created by you if you intend to use WQRAX\_USER.

**Example value:** *wqraxUserFex=rax\_user.fex*

---

#### ***4.17 Timeout redirect URL***

WQRAX invokes Web Query's REST web service APIs to run reports. Before running the report WQRAX has to connect to REST to sign on. The connections to REST can time out, according to the session time out configuration of Web Query.

Users now can configure a WQRAX timeout redirect URL in wqrax.properties. When the connection to REST times out, WQRAX will redirect the user to the configured URL.

Note that only users who have set up the wqraxUserFex parameter will be redirected to the configured URL, for other users WQRAX will automatically reconnect to REST if the connection times out.

***Example value: wqraxRedirectURL=https://www.google.com***