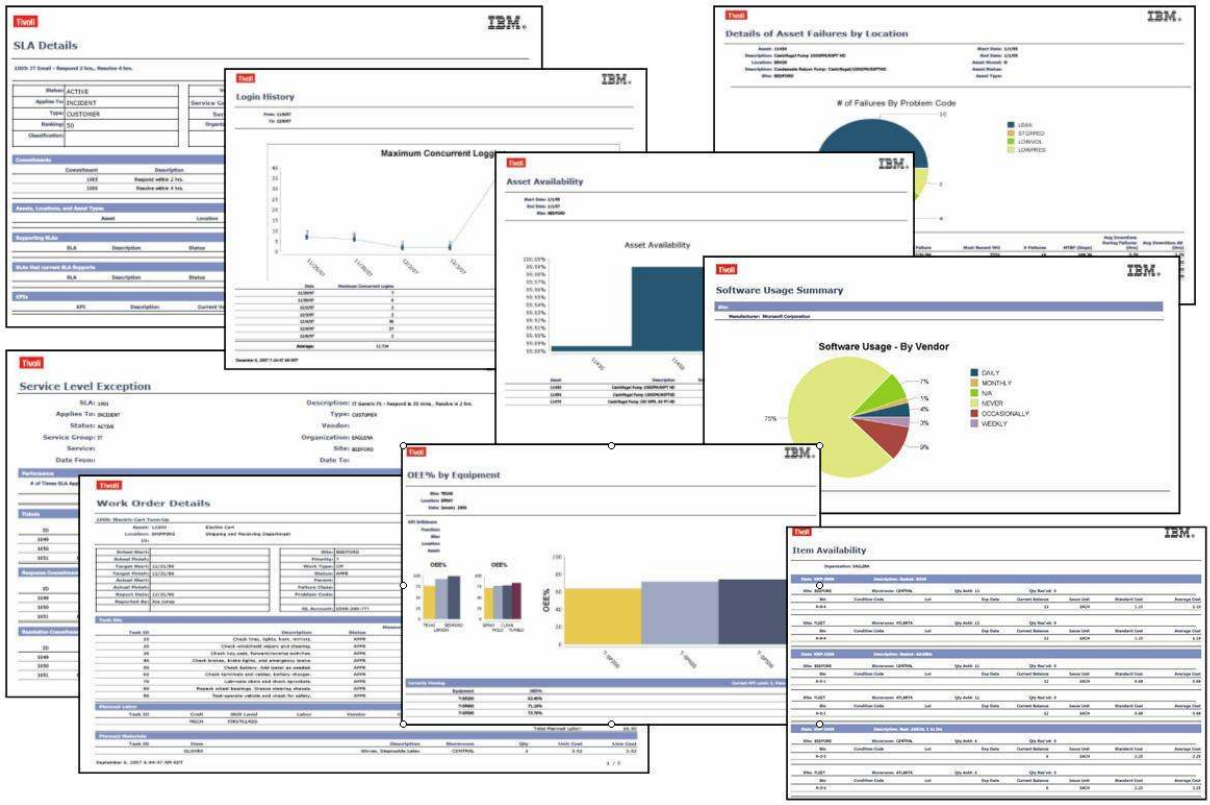
IBM Maximo Asset Management

Report Development Guide



V7 Reporting

To respond to today's dynamic Business Environment critical business information needs to be immediately available. This business information can come in a variety of formats, and is often required as a report – either a formatted business report, known as an Enterprise report, or an Ad Hoc report which is created on the fly by users.

IBM Maximo ® includes an Open Reporting Architecture, which enables you a number of different reporting options to choose from. The report options have been significantly enhanced in Maximo 7 and include a wide range of reporting tools.

The embedded reporting tool in the Maximo 7 Releases is BIRT, Business Intelligence and Reporting. As the embedded reporting tool, it enables the deepest levels of integrations throughout the various Maximo applications.

This guide details the processes in developing BIRT Reports using the BIRT Report Designer. This includes how report designs are utilized in V7, including their file structure of design, library and property files. Additionally, information on customizing, importing, exporting and localization are discussed.

Also, references to other support documentation detailing common report development customizations, including report logging, implementing barcodes and changing report logos are referenced.

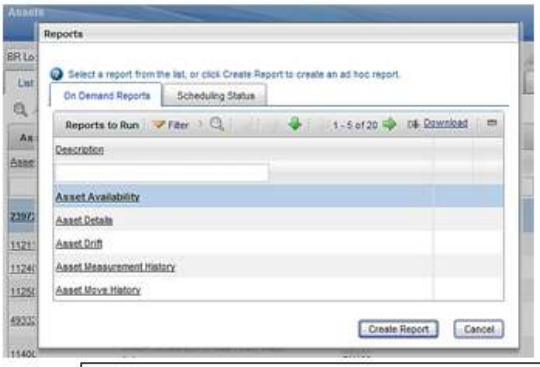
*Note: This document applies only to the embedded report tool in the Maximo® Base Services 7.1x Releases. A separate development guide is available for the Version 7.5x releases, which can be found here:
https://www.ibm.com/developerworks/wikis/display/maximo/Report+Reference+Materials

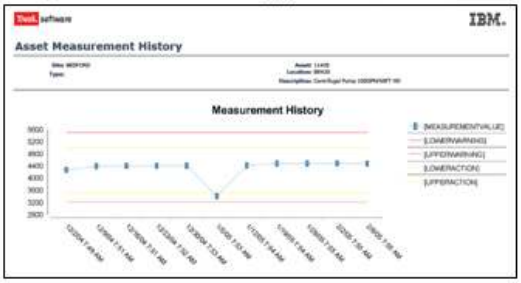
BIRT Report Types

There are two types of V7 BIRT Reports: Enterprise and Ad Hoc Reports. Enterprise reports are created by a Developer in the BIRT Designer tool, whereas Ad hoc reports are created by users on the fly within the applications.

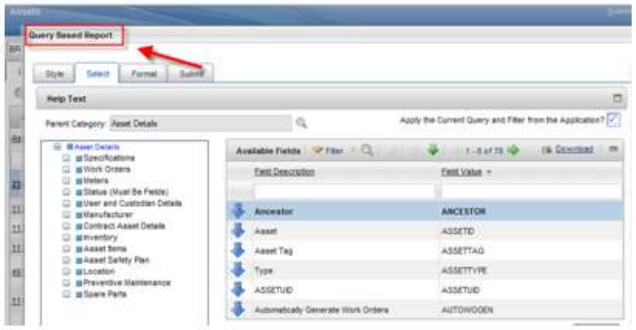
While this guide covers an overview of BIRT Reporting, it will focus on Enterprise reports. Information on BIRT Ad Hoc (QBR) reports is contained in a separate document titled 'V7 QBR Ad Hoc Reporting'. Links to all reference materials are noted on the last pages of this document.

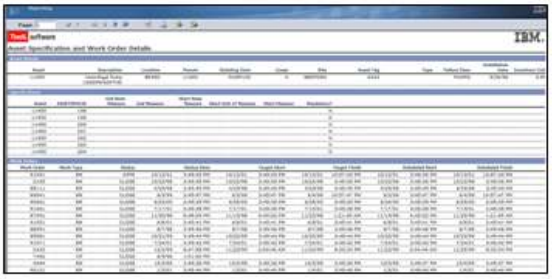
Enterprise Reports





QBR - Ad Hoc - Reports





BIRT Designer

The BIRT Designer is an Eclipse Based Tool that Java Developers use to create and customize V7 Enterprise reports. In V7.1.1 thru 7.1.1.4, BIRT Designer 2.1.2 is used, which is based on Eclipse 3.2.2. Beginning in V7.1.1.5, BIRT Designer 2.3.2 is used, which is based on Eclipse 3.4.2.

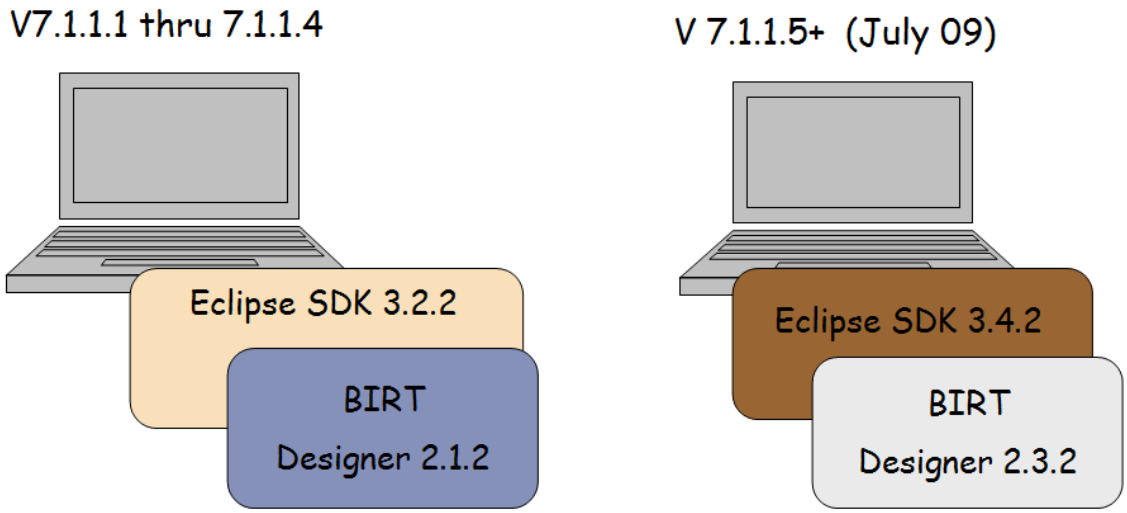
Multiple advantages result from BIRT being built upon the Eclipse Framework. These advantages include

- An overwhelming majority (estimated at 65%) of Java Developers Worldwide currently use Eclipse, which subsequently
  - Reduces the learning curve of the BIRT Designer
  - Enables Use of Java Resources which are readily available
- Use of XML Report Design Files
  - Standardizes with V7 and Industry Platforms
  - Enables quick identification of file differences
- Standardization of V7 Processes like install and localization

To enable the report integration, custom library, style sheet, templates and data sources have been created. These files insure a consistent, look and feel for all reports, plus most importantly, insure that reports will execute correctly from the various applications. These files must be used on all custom reports to insure the report integration executes properly.

The BIRT Designer is installed on the client machine of the Java Developer(s) who will be creating or customizing reports. It is not required to be on each user's machine – only those users who will be physically creating or customizing reports. Since the BIRT Designer executes off the Eclipse Framework, Eclipse must first be installed on the Java Developers Workstation, and then the BIRT Designer is installed within Eclipse.

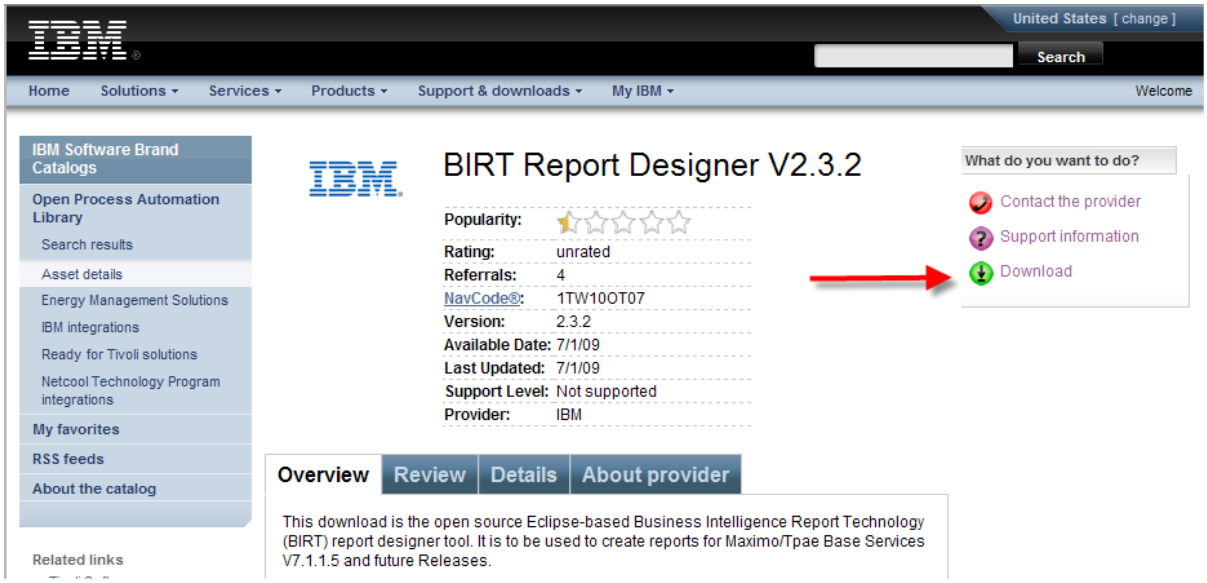
Depending on the Maximo Base Services Release you are using, will determine which version of BIRT you will work with. For Maximo Base Services 7.1.1.5 and future Releases, BIRT 2.3.2 is used. For releases prior to Maximo Base Services 7.1.1.5, BIRT Report Designer 2.1.2 is used.



Downloading the BIRT Report Designer

The first step is utilizing the BIRT Report Designer is to download and install it. After verifying the version of BIRT which correlates to your Maximo Base Services version, download the applicable copy.

The download for BIRT 2.3.2 with Eclipse 3.4.2 (V7.1.1.5 and later releases) can be found here: http://www.ibm.com/software/tivoli/opal?NavCode=1TW10OT07



The download for BIRT 2.1.2 with Eclipse 3.2.2 (V 7.1.1. thru 7.1.1.4) can be found here: https://www14.software.ibm.com/webapp/iwm/web/reg/download.do?source=tivopal&S_PKG=1TW10OT03&lang=en_US&cp=UTF-8#

Configuring the BIRT Report Designer to work within Maximo

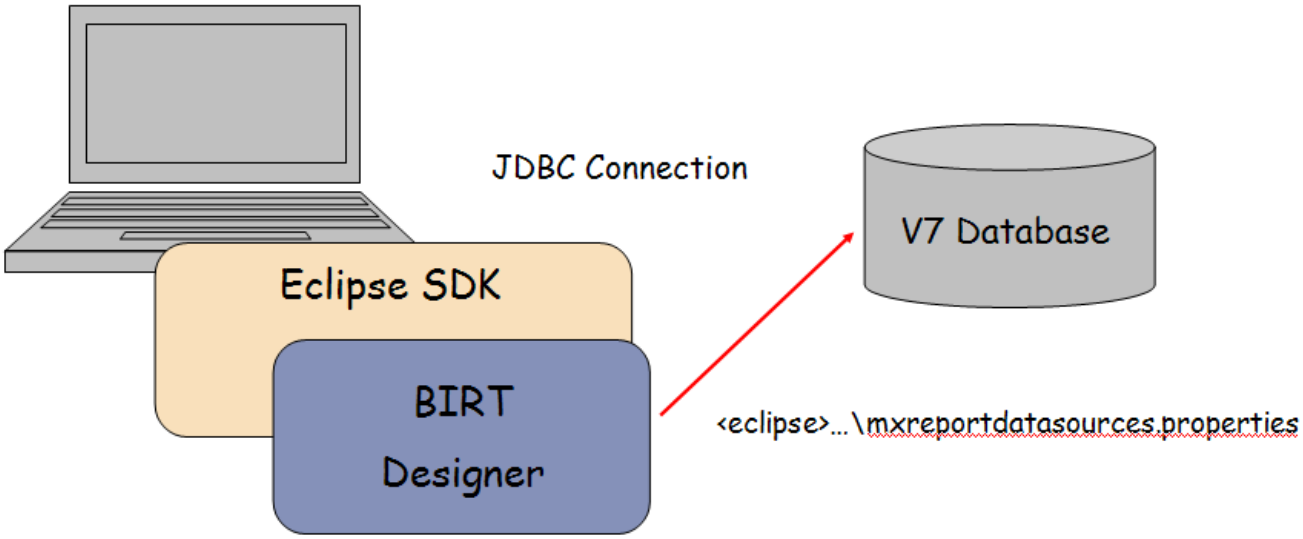
After you have downloaded the BIRT Designer, the next step is to configure it so it works correctly within your Maximo based environment. The steps to do this are detailed in documents below. Please make sure you select the correct configuration document for the version of BIRT you are using.

'Configuring BIRT Designer 2.1.2 with V7'.
The reference number for this document is 1315837.

'Configuring BIRT Designer 2.3.2 with V7'.
The reference number for this document is 1390372.

The steps involved in configuring the BIRT Designer include configuring the mxreportdatasources.properties file. When a developer creates a report in the BIRT Designer, a connection must be available to the database. This enables him to develop and test his reports to make sure they are executing properly. This database is often a test or development instance.

From the BIRT Designer, the connection to the database is made via a JDBC Connection. The specific database property values are set by the developer in the mxreportdatasources.properties file.



Report Developer Database Access

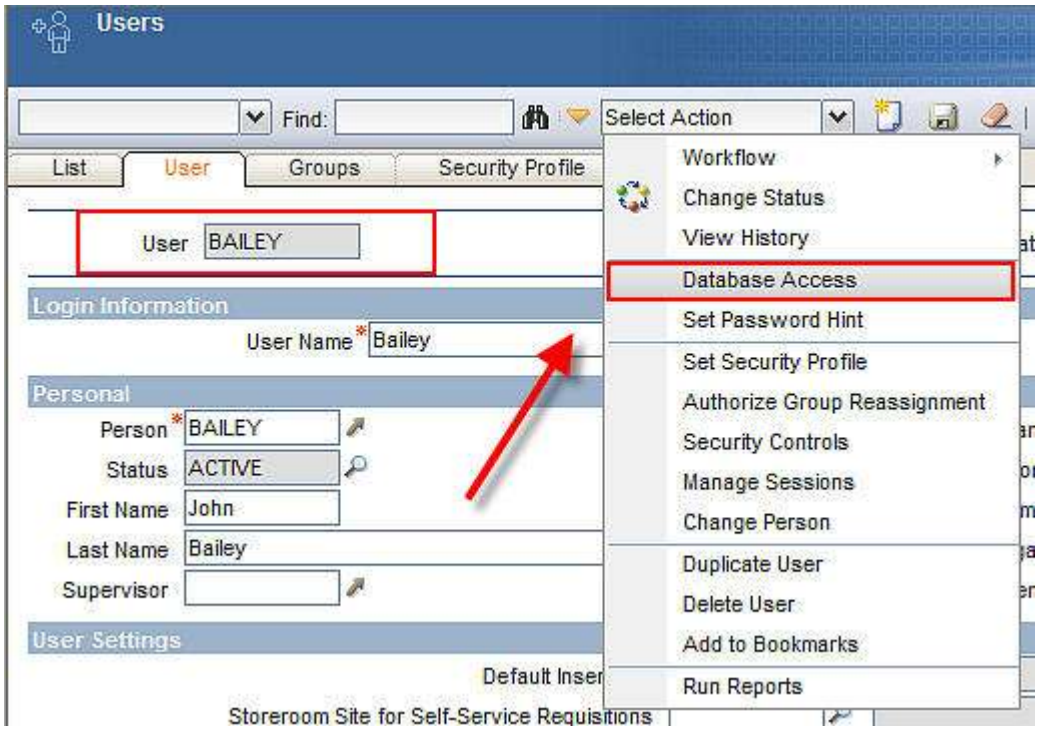
Report developers require database access to create and customize reports. This enables them to test their report design to insure the applicable content is being retrieved.

Clients may not want to grant each report developer full database access by using the system maximo database user privileges ad the developer creates and test report designs. Instead, they want the developer to have restricted database access. This restriction usually requires that the report developer be granted 'read only' access to a limited number of database objects. To do this, a unique database user is required.

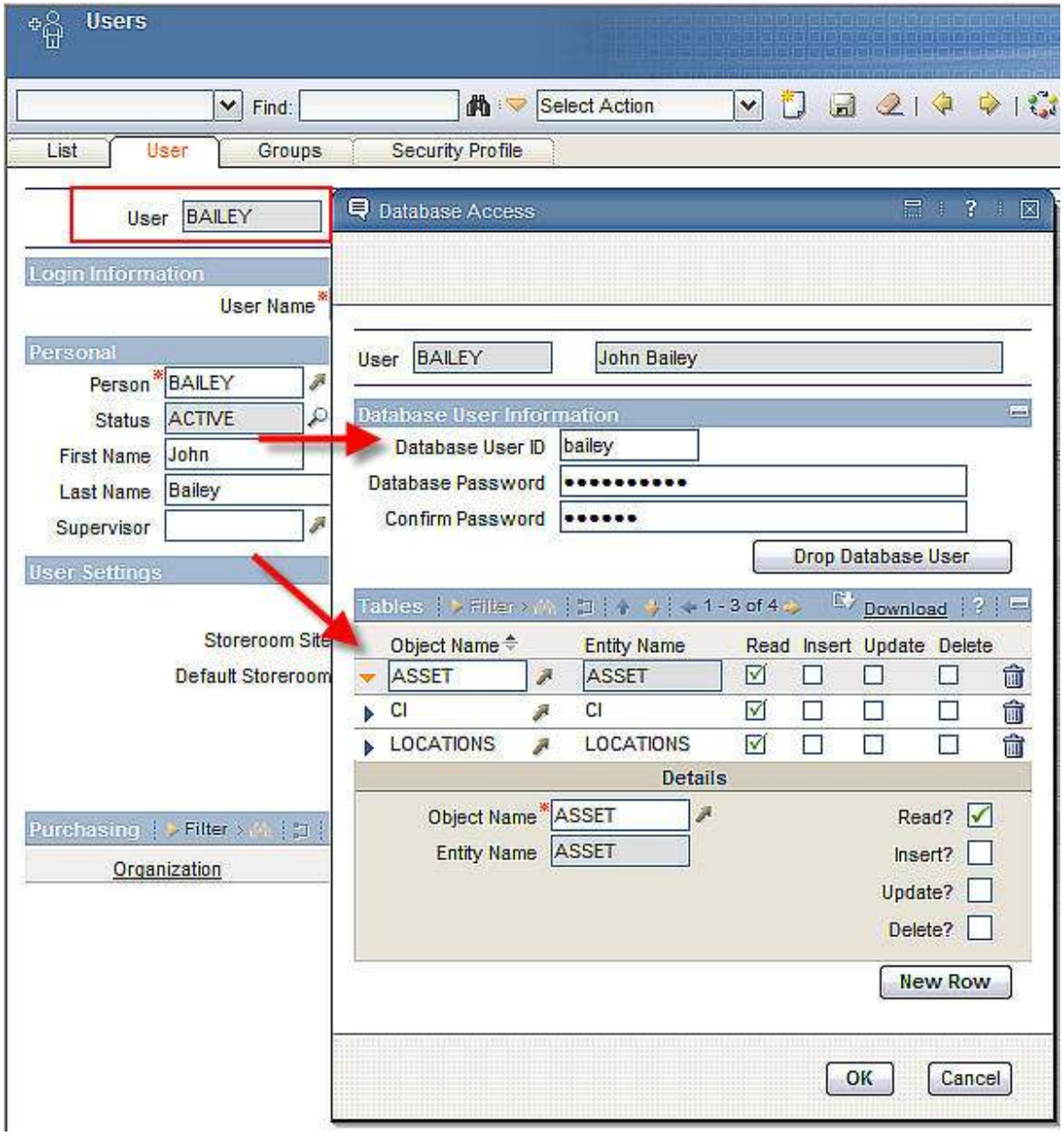
The steps below detail a few different ways on how this restricted access can be granted thru the use of a unique database user. After this unique database user is created, the steps for incorporating the new database user for the report developer are described.

Method 1 – Creating Database User and Access within User Application

If you are using Oracle or SQL Server, you can directly create a new database user through the User Application in V7. To do this, first create a new user for your report developer. Then, from the Action Menu select 'Database Access'.



Enter a unique Database User ID, along with the database password. Then, using the table section in the bottom portion of the dialog, specify the database objects that the report developer should have access to. Grant database 'read only' access to these specific database objects the report developer will be creating reports against. In this example, the developer, Bailey, is given read only database access to the Asset, CI and Locations objects.

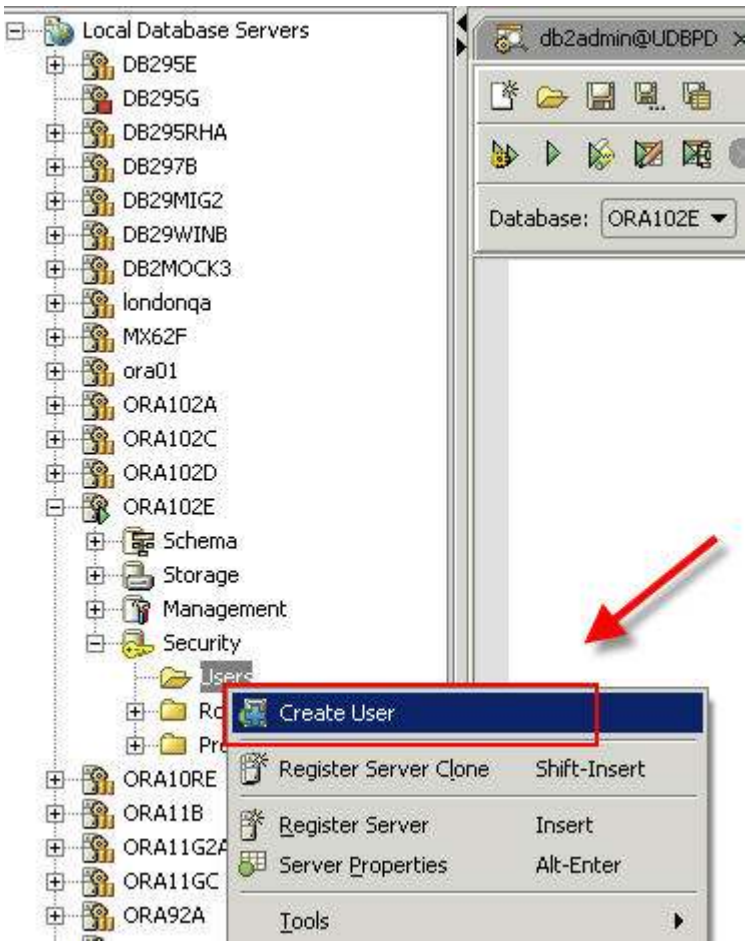


Note: If you are using DB2, the new database user must also be an Operating System (OS) User. Therefore, the DB2 user must first be added as an OS user before performing the action above.

Method 2 – Creating Database User and Access within Database Configuration Tools

You can also create a new database user and specify access through a Database Configuration Tool.

To do this, access the database querying tool, and locate the database instance you are working with. From the tool, add a new database user. (*Note: The method in which you access this functionality will vary by database tool and type.)



Once the database user is created, then grant 'Read only' database privileges via scripts to the specific database tables he will have access to. Example scripts are shown below, where the report developer, Bailey, is granted 'read only' access to the ASSET, CI and LOCATIONS objects.

grant select on MAXIMO.ASSET to bailey

grant select on MAXIMO.CI to bailey

grant select on MAXIMO.LOCATIONS to bailey

Configuring the BIRT Report Designer to use the new report developer database user

After the new database user has been created, this database user will then be used by the report developer in his unique instance. To enable this the mxreportdatasources.properties file will be updated.

This mxreportdatasources.properties file specifies the database url and driver, along with the database user and password that should be used. Using our example above, values for this property file could be:

maximoDataSource.url=jdbc:db2://V7116:50000/HARRIER
maximoDataSource.driver=com.ibm.db2.jcc.DB2Driver
maximoDataSource.username=bailey
maximoDataSource.password=bailey1abc

Report Design Files

Before creating reports, a review of the report design process will be detailed. There are three files created for report designs.

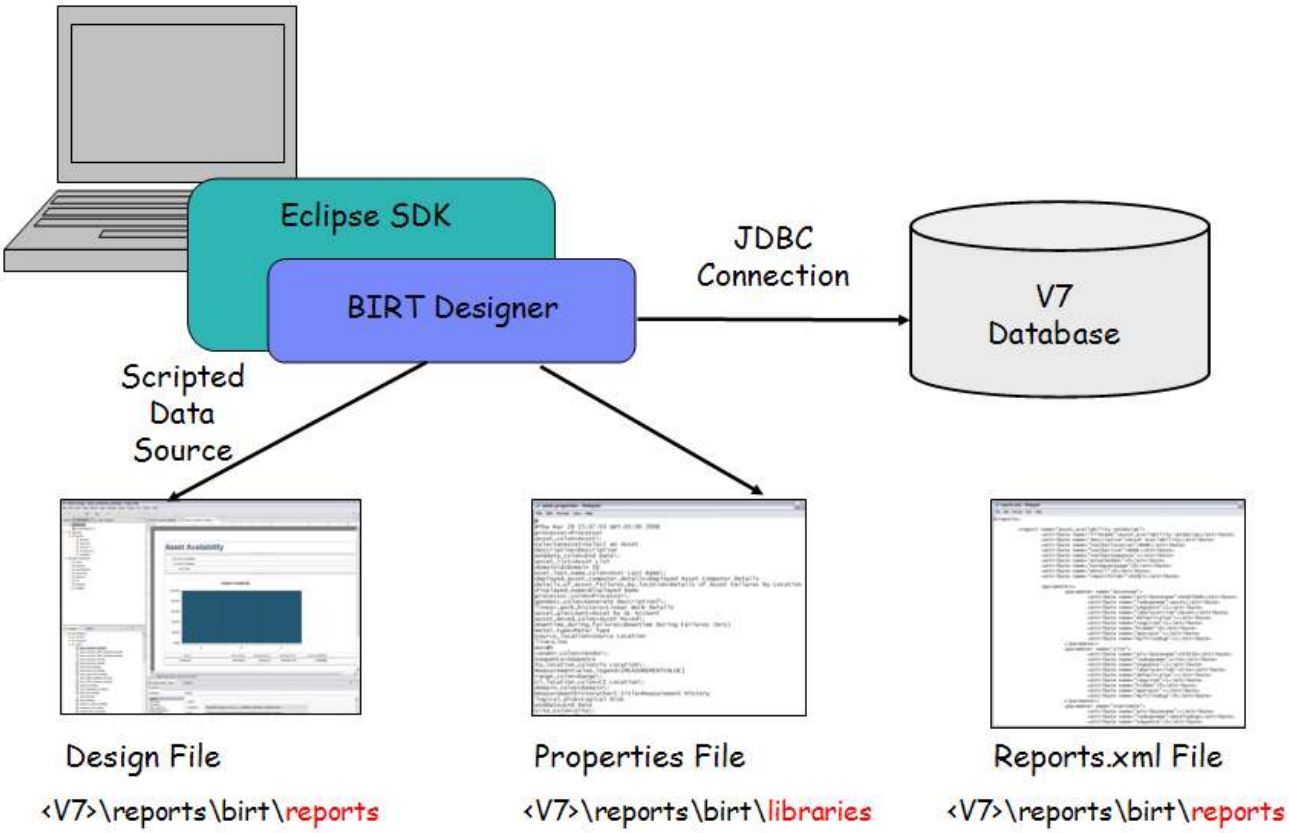
1. Design File. This contains the details on the report – its sql, grouping, sorting, hyperlinking, etc. An example of this is the asset_availability.rptdesign file.

   The Design File uses a custom scripted data source. This is done to fully utilize the specific functionality for Runtime Data Translation and Time Zone Conversions.

   The scripted data source calls the JDBC Connection to execute the report against the V7 Database.

2. Properties File. This contains the text values and keys of each column label and report title. There is one properties file for each application that has reports. This enables the same label values (ex. Description) to be used only once. This property file is one of the major components used in localization. An example of this is the asset.properties file.

3. Reports.xml File. This file defines the report information (its design file name, its parameters, its application etc.) and is used to import the report files into the database. There is one reports.xml file for each application.



The chart below shows how the report files interact with each other. At the top level is the design file, which always has the file extension of .rptdesign.

Each of the reports has a dependency on the Maximo® System Library File. A BIRT Design file can only have a dependency on either another design file (.rptdesign) or another library file (.rptlibrary)

The Maximo System Library file has its own import file, called libraries.xml. If a change is made to the Maximo System Library, the libraries.xml file is used to import that library change into the database.

The Maximo System Library file contains references to the resources, or image files. These typically have a .gif or .jpg extension. When a resource file is imported into the database, the files are converted to .zip format. (These files are stored as BLOB data types in the database.)

The properties files are also resource files. Properties files are referenced in the reports.xml which is used to import the reports into the database.

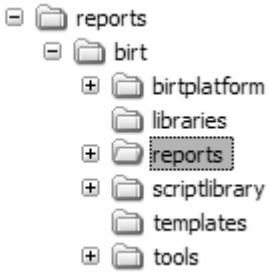
File Name	Dependency	Resource	Description	Location**
Asset.rptdesign			Asset List Design File	reports
	maximoSystemLibrary.rptlibrary		Maximo System Library	libraries
		.gif/.jpg files	Resources or Image Files	libraries
		asset.properties	Asset Property file	libraries
Reports.xml			Information on report and its parameters. Used for importing	reports

**Location in the chart has been condensed. Its full path is <v7>\reports\birt\....
More details on these files are contained in the report file structure below.

Report Design File Structure

The report infrastructure is contained within the application file structure. It contains not only the files required for the BIRT Engine, but also the design files, libraries, templates and various tools used during the report import process.

This report structure in for BIRT is: <V7>



At a top level, the six subfolders contain the following information:

Birtplatform

Contain files required for the BIRT engine. These files should not be modified.

Libraries

Library, Resource and Property files required to support the report design files.

A. Library. Libraries store re-usable components, functionality and images. Reports that use libraries are automatically updated with the latest library information when they are executed.

One system library, called MaximoSystemLibrary.rptlibrary, is used. It contains two core items:

1. Master Pages. This defines items like the margins for printing, and the controls used for page formatting (ex page n of m). This is contained in the library because it is used on all reports, and rarely changes.

2. Themes. This contains the style sheet, which defines the font type, font size and other text characteristics to be used in the reports. The theme in the library is referred to as the style in the report design. The maximoTheme contains the specific colors and formatting for the reports.

The libraries.xml file is used for importing the MaximoSystemLibrary file.

B. Resource. Resource files are .gif or .jpg images used in report designs. Two resource files are used. These are IBM_logo_black.gif and tivoli.gif, which are the two logos displayed at the top right and left hand corner of each V7 Out of the Box report. Resource files are imported into the database as zipped files.

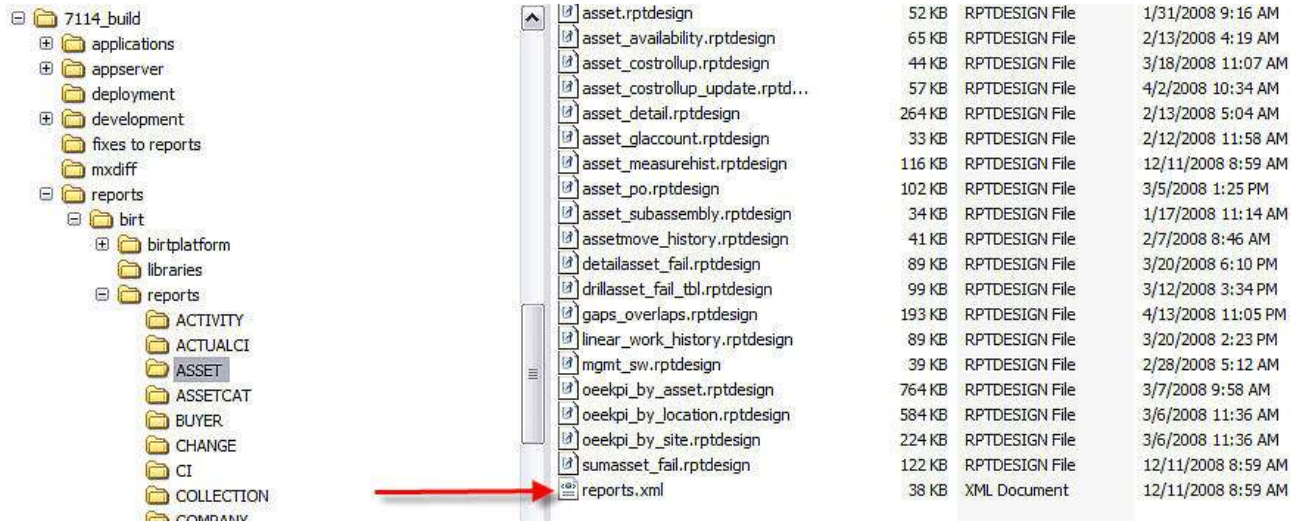
Clients may want to customize the reports to use their own corporate logos. Information on how to do this is in 'Changing Logos in BIRT Reports' referenced on the last page of this document.

C. Properties File. Each of the application's properties file is contained within this subdirectory. Property files contain the text values of the report titles, and column/Subheader labels.

Property files are created at the application level, and not at the report level, because reports within an application frequently share the same text label values. (Example: Asset Reports often use the same labels of Asset, Location, Site, multiple times.)

Reports

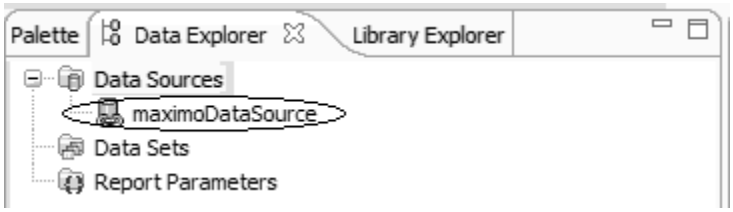
Contains Report Design Files stored within their corresponding application subfolder. Also contains the reports.xml file with information on each report used for importing.



Scriptlibrary

The Script library contains script library classes and the mxreportdatasources.properties file used by BIRT Designer tool to connect to databases.

For the integration of BIRT, when a report developer creates a report, a Custom Scripted Data Source is used. This Scripted Data Source is called 'maximoDataSource'.



A scripted data source is used to fully utilize the specific functionality for Runtime Data Translation and Time Zone Conversions. An example of this functionality is the localized values of Description. If a client is running both English and Spanish environments, and the English values of descriptions been localized into Spanish, the scripted data source is required to insure the localized Spanish descriptions display in reports. The classes for the scripting are contained within this subfolder.

Notes on Script Library:

1. Whenever you update your system to a new patch release or version of Maximo Base Services, the script library may have been updated in the new release. To insure that you use the most recent script libraries in your environment, copy the latest script library from

<V7Directory>\reports\birt\scriptlibrary\classes

To

<birt>\eclicpse\plugins\<birt report viewer directory>\birt|WEB-INF\classes

For example, when you upgrade from Maximo Base Services 7.1.1.5 to Maximo Base Services 7.1.1.6, copy the 7.1.1.6 classes directory to your existing BIRT instance.

2. For details on the script library, including the methods available, reference the V7 Java Docs available on IBM's Integrated Service Management Library website.

http://www-01.ibm.com/software/brandcatalog/ismlibrary/

Report Templates

Six Template files are used as starting point in creating report design files.

File Name	Template Name	Description
maximoListReport	Tivoli Maximo List Report Template	For simple listing report - traditional row, column format
maximoGroupReport	Tivoli Maximo Grouped Report Template	Same as listing report - but contains sections for grouping results - ex. group by site or status
maximoSubreport	Tivoli Maximo Subreport Template	Used for complex reports, including detail reports
maximoChartListReport.	Tivoli Maximo Chart List Report Template	Simple listing report, which includes a graphic for either bar, line or pie chart before the report's results.
maximoChartGroupReport	Tivoli Maximo Chart Grouped Report Template	Grouped report with graphic for either bar, line or pie chart before the report's results.
maximoChartSubreport	Tivoli Maximo Chart Subreport Template	Complex report with graphic for either bar, line or pie chart before the report's results.

*NOTE: When creating any report to be used within Maximo, you must start with one of the Tivoli Maximo templates as they contain the required data source and library file needed for the integration.

Tools

Files used to importing and exporting report design files from database. More information on these tools is contained in the Import and Exporting sections.

Additional Notes on Report Source:

1. There are no separate library or design files for the three database types that are supported. Within the report source, the sql is being written in ANSI Standards, so it will be applicable to any of the 3 database types.
    - There may be a few out of the box reports where the database specific sql is required. In these cases, the sql will be written with conditional statements (ex. If database type = IBM DB2®, do this. If not, do this + that…etc)

Developing a report

This section details how to create a report design within the Report Designer for V7.1.

Note: Before beginning this process, please review the section titled 'Extending Ad Hoc Reports in the BIRT Designer'. This section will detail how you can streamline the report development process by minimizing the steps below thru the use of exported ad hoc reports. It is highly recommended that you utilize this process to save both development time and resources.

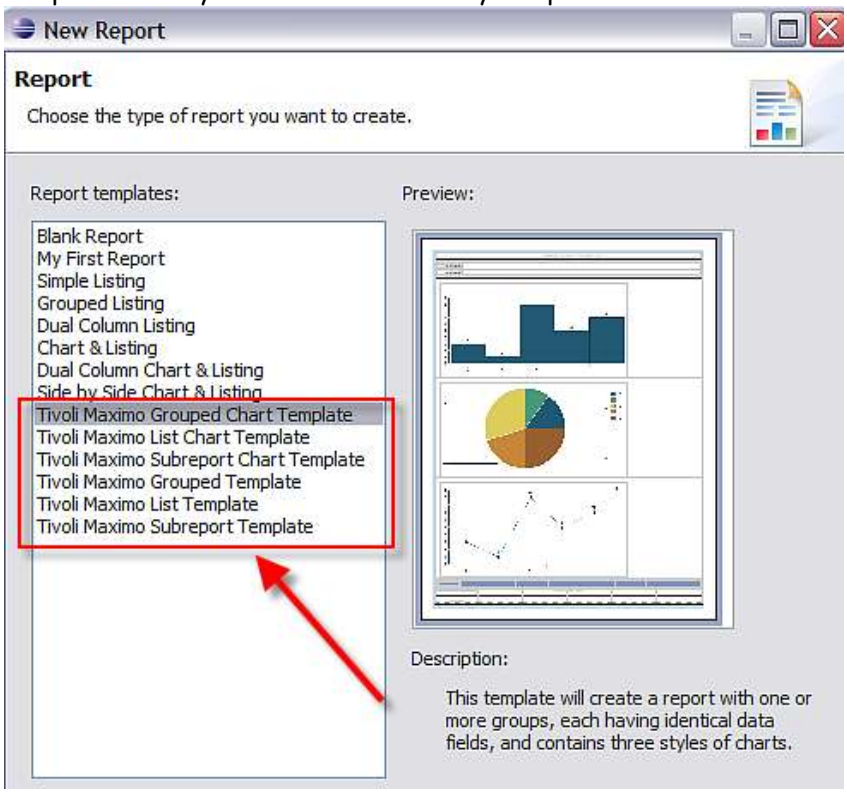
If you want to create a report design file without using the ad hoc exporting process, follow the process below.

There are four steps required to for report development:

1. Specifying the query
2. Creating the output columns
3. Updating the Fetch to map the query columns to the output columns
4. Formatting the report

To begin, access the Report Designer. Then, select File – New – Report or choose New Report from the dropdown list. A number of sample reports and templates are displayed.

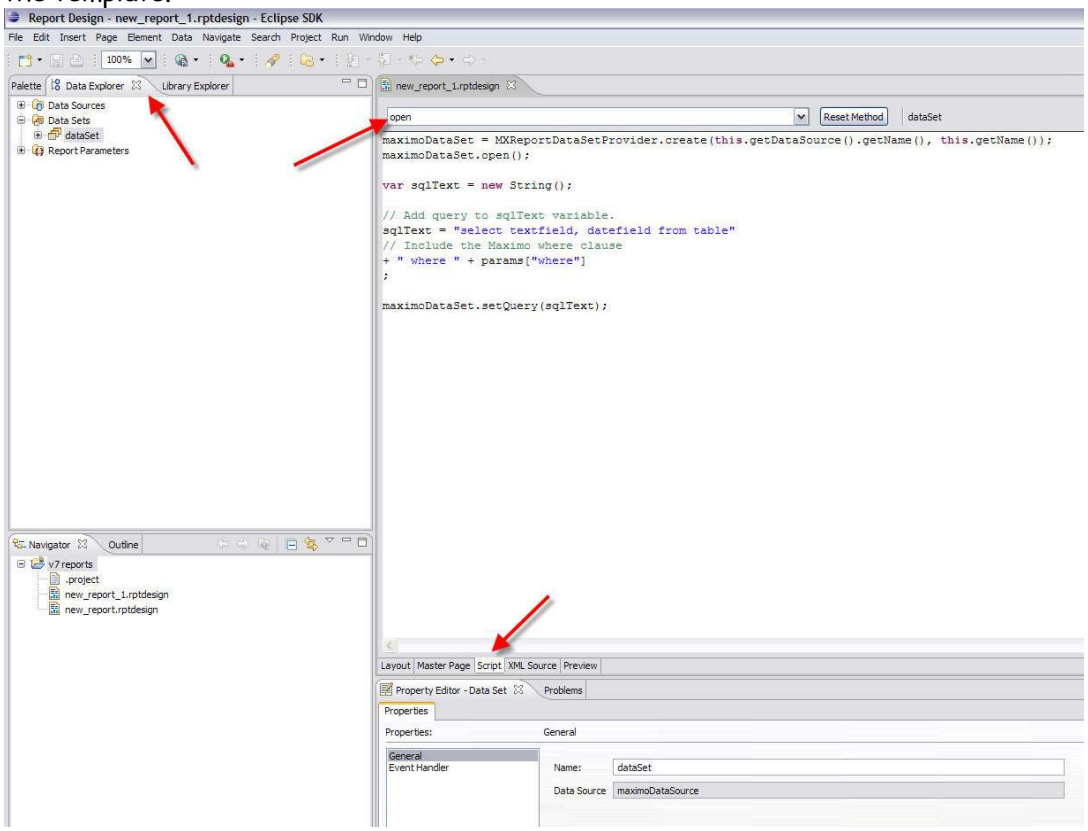
Select the desired Tivoli Maximo template from the list. You must select a Tivoli Maximo template as they contain the necessary scripted data source and library for the integration.



Specifying the Query

The first step in creating a BIRT report is to input the sql statement. When doing this, it is highly recommended that you first develop and test all required queries in your separate database query tool. BIRT does not validate SQL and a query tool will provide clearer error messages.

To input the sql, select the data set in the Data Explorer and choose the Script tab. Select the Open method from the dropdown list. Copy your query from the query tool and paste it into the method body under the existing sample query. Format your query to match the sample provided in the template.



Notes on the sql:

1. It is recommended that ANSI SQL join syntax (left outer, right outer) should be used. ANSI functions such as CASE and COALESCE should be used instead of proprietary functions such as DECODE and ISNULL.

2. Owner qualification (MAXIMO.workorder) should NOT be used

3. Reference all database objects in lower-case.

4. Each report must contain the base table name of the application it will be accessed from in its sql statement.

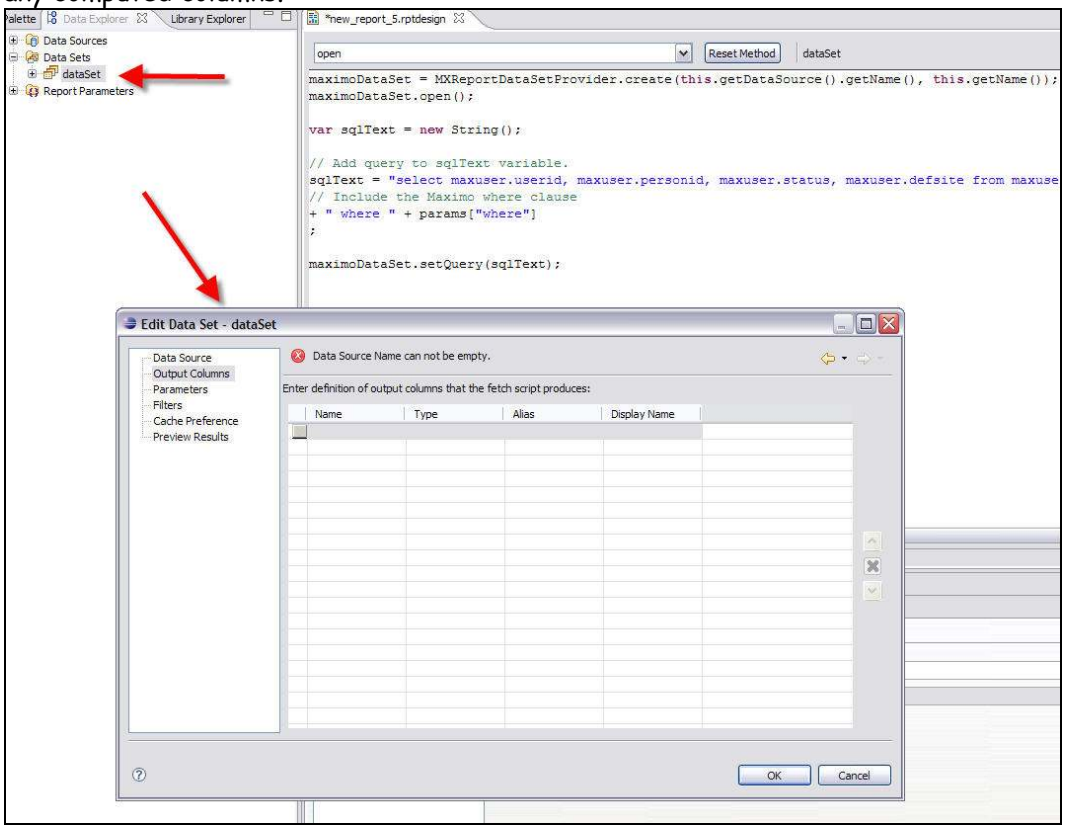
This can be explained using the example of a report that is being created for the Location Application. When the sql for the report is being prepared, the base table name of the application must be included in the sql. You can find the base table name for an application by executing a query similar to what is shown below:

```
select maintbname from maxapps where app = 'LOCATION'
```

Then, once you obtain the base table name, confirm that it is included in your sql. Even if you do not include any fields from the base table, it still must be included in the report's sql.

Creating the Output Columns

Next, the report output columns will be defined. Double-click the data set to open the properties dialog. In the Output Columns editor, enter a column for each field in your query, as well as for any computed columns.



Set the data type for each output column based on the maxtype of the field. The chart below shows the Maximo Database Type, the corresponding BIRT Data Type, and the method used within the BIRT Designer to retrieve its value.

V7.1-BIRT Data Mapping

V7.1 Database Type	BIRT Data Type	Data Set Method used to Retrieve
ALN, CLOB, GL, LONGALN, LOWER, UPPER	String	`getString(String attributeName)`
YORN*	String	`getBooleanString(String attributeName)`
DATE, DATETIME	DateTime	`getTimestamp(String attributeName)`
DATE	Date	`getDate (String attributeName)`
AMOUNT, DECIMAL, DURATION*	Decimal	`getDouble(String attributeName)`
FLOAT	Float	`getFloat(String attributeName)`
DURATION*	String	`getDuration(String attributeName)`
INTEGER, SMALLINT	Integer	`getInteger(String attributeName)`

To determine the Maximo data types (maxtypes) of the fields used in your queries, you can query the maxattribute object directly in the database. An example of this is:

select attributename, maxtype from maxattribute where objectname = 'WORKORDER' order by attributename

Or, you can use Database Configuration application in V7 to look up the maxtypes, using the Type field on the Attributes tab.

Notes:

1. You do not have to give the output columns the same names as the database fields, although it is usually easier to do so.

2. The following Database Types are not supported in reports: BLOB, CLOB, CRYPTO and CRYPTOX.

3. YORN fields are stored in the database as numbers (0 and 1) but are presented in V7 as localized text. The getBooleanString(String attributeName) method will perform both tasks: retrieve the numeric value and translate it to the appropriate text. You also can obtain the translated value from the integer using getBooleanString(int intValue).

4. DURATION is stored in the database as a number (fractional hours) but in V7 it is presented as a string in the format HH:MM. The getDuration method will return the formatted string.

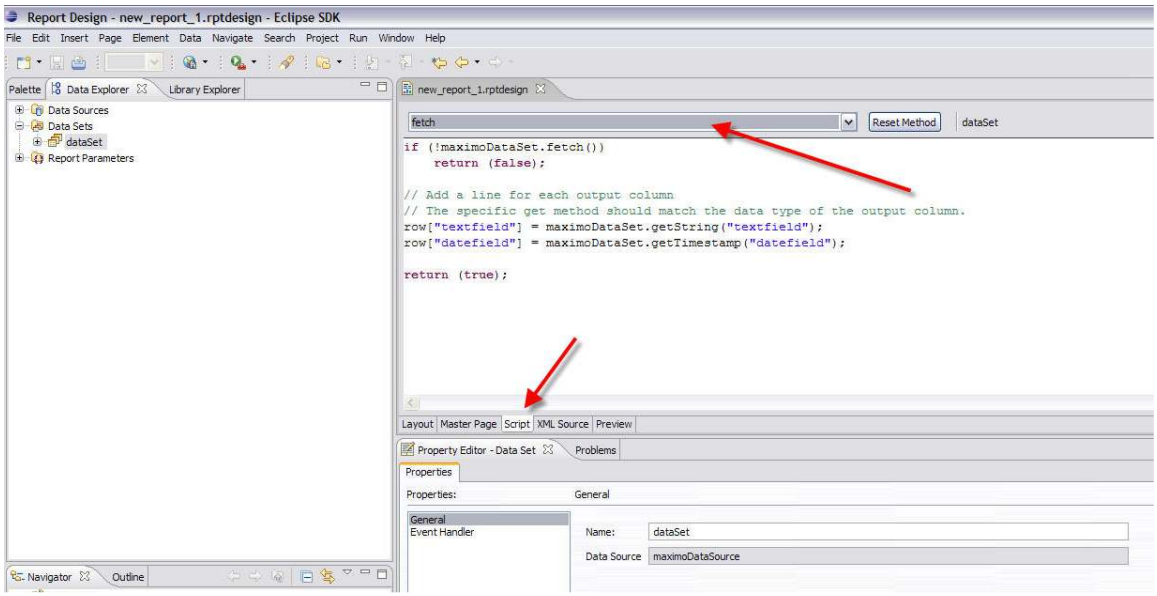
If you require the numeric value instead, you can use getDecimal. An additional utility method, MXReportUtil.getDuration(String attributeName), is provided to perform the conversion from double to string.

5. If you leave the open method visible as you do this, you can use it for reference on the columns.

Updating the Fetch

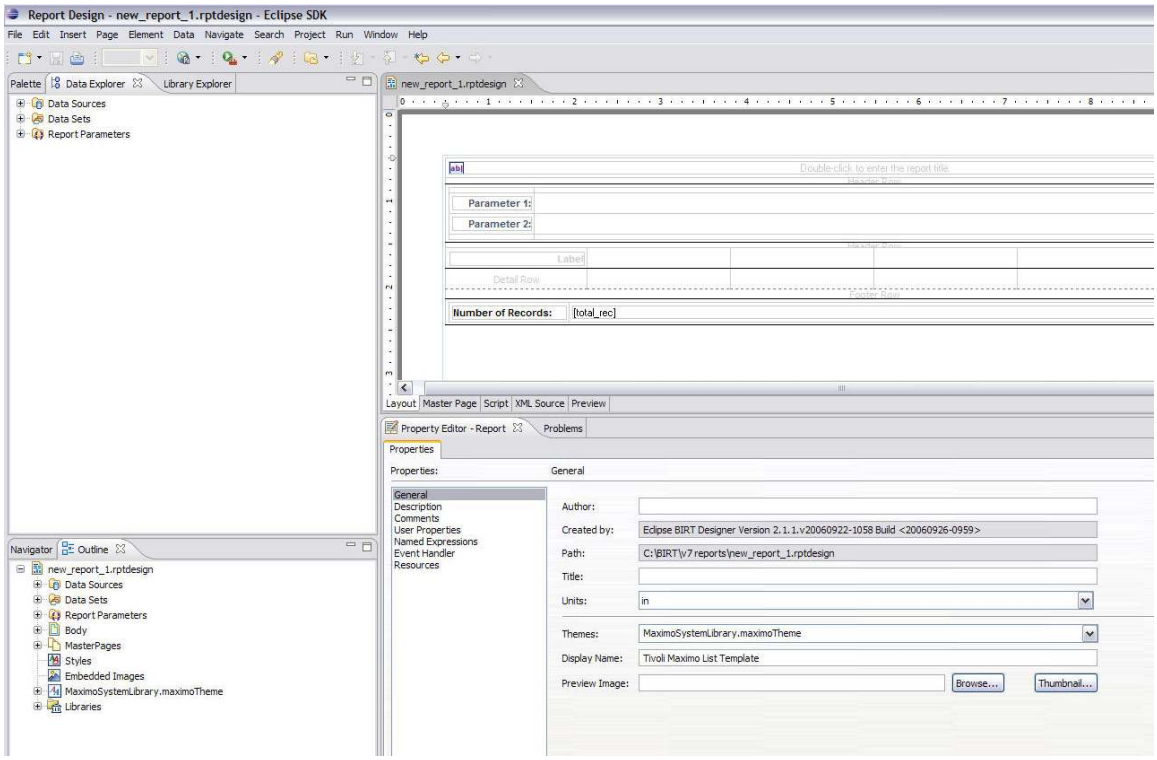
In the third step, you need to map the query columns to the output columns. This is done by updating the fetch method.

On the Script tab, choose the Fetch method from the dropdown. Add a line for each column that retrieves the value of the field from the data set and updates the output column with that value. Use the appropriate method based on the data type of the field, following the Maximo-BIRT Data mapping chart shown in the section above.



Formatting the Report

Begin formatting your report by dragging the fields from the data set to the report. Set a fixed width for each column, otherwise the report will not format correctly in PDF.



Set any parameter display fields. Do not drag parameters from the Data Explorer into the report. Instead, drag a Data element and set the Value Expression to the parameter.

If there are groups, set the keys.

Formatting Notes

1. All Table elements should have widths set to 100%. Some templates included fixed table widths (in inches) and this is incorrect. You can also remove the height if it is set.

2. The style "titlesub" can be used for text that appears directly under the title. Examples of out of the box reports that use this style are detail reports, like Work Order Details, for the detail report key and description. f this (woprint.rptdesign).

3. All subreports exist in a single cell, stacked on top of each other.

4. To receive a page break after the last subreport, a group was added. The group key is set to the unique key for the report - for example in Person Details (person_details.rptdesign) it is set to Personid. The page break after property on the group is set to "Always excluding last".

Now there will be a page break after each person record (including the related subreports) but not after the last person, which would cause a blank page at the end. The report footer rows have been deleted, again because this would cause a trailing blank page.

5. If you try to view your report within the designer as 'View as PDF', it will not work unless you install the iText jar. You will receive this error:

org.eclipse.birt.report.service.api.ReportServiceException: Report engine fails to create extension to handle this request.

Report Development Considerations

Dates

The static MXReportSqlFormat methods are provided to support date formatting. All return strings of JDBC-formatted date functions that can be used in report SQL statements for all supported databases. For example:

"where actualdate <=" + MXReportSqlFormat.getCurrentDateFunction()
evaluates to:
where actualdate <= { ts '2007-04-01 00:00:00' }

getCurrentDateFunction() – current date
getCurrentTimestampFunction() – current date & time
getDateFunction(Date d) - date based on date input
getTimeFunction(Date d) – time based on date input
getTimestampFunction(Date d) - date & time based on date input
getStartDayTimestampFunction(Date d) – date based on date input, with time component set at start of day (for start date parameters)
getEndDayTimestampFunction(Date d) – date based on date input, with time component set at end of day (for end date parameters)

Date Formats

The reporting tool offers custom date formatting. However, due to localization issues, you are strongly encouraged to use only Date/Time controls using Short, Medium or Long Date/Time formatting will be used.

All 'Out of the Box' Reports will use the date formatting below:

For Dates: Short Date
3/29/07
For Date/Time: General Date
March 29, 2007 4:03:00 PM EDT

When both the date and time need to be displayed in a condensed format – for example, target start, actual start, target finish etc – two controls will be used. These are:

Short Date + Medium Time.

So, within a WO Report where Actual Start within a column needs to display, the field would show as 4/26/07 4:12:34 PM and would be created by using 2 controls: Short Date + Medium Time.

Linking Result Sets

When you run additional queries in the Fetch method, you usually must link them to the current data row. You can do this either by directly including the value or by using data set parameters.

From the Fetch query example above:

```
sqlText = "select description from classstructure where classstructureid=?";
classStrucDataSet.setQuery(sqlText);

// Use value from main query as foreign key in secondary query
classStrucDataSet.setQueryParameterValue(1, maximoDataSet.getString("classstructureid"));
```

In this example, the parameter is set to the value of a field in a data set. The field is a string so the data set getString method is used. The getTimestamp method may also be used but the fetch methods that return primitive data types cannot; instead use the following:

```
getDoubleObject(String attributeName)
getFloatObject(String attributeName)
getIntegerObject(String attributeName)
```

The other common situation where you must link result sets is when linking subreports. Subreport queries are similar to Open method queries in that they are both executed each time a record in the main query is fetched. The main difference is that subreport queries should have their own data sets. The contents of the subreport can be contained in an independent child table, which is bound to the secondary data set and nested in a cell in the parent table.

To link a subreport query to a main query, include the linking fields (foreign keys) in the main query. In the subreport query, reference the linking fields using the "rows" variable:

```
sqlText = "select laborcode, craft from labtrans where refwo = '"
+ rows[0][ "wonum"] + "' and siteid = '" + rows[0]["siteid"] + "'";
```

Hyperlinking

When you specify a report to link to, BIRT validates that the report exists, and reads its parameter information. Therefore, before you can set the hyperlink properties for a field, you must at least create a placeholder .rptdesign for the target report, in the correct application folder and with the correct file name. When initially specifying this, the target report design does not need to be complete. Once the target report is in place, use the following steps to create the link:

1. Select the Data element in the source report and choose Hyperlink in the Properties window. Select the ellipse to open the Hyperlink Options dialog. Set the Hyperlink Type to "Drill-through".

2. Under "Select a target report", enter the relative path to the linked report. If the report is in the same folder, just enter the report name. If the report is in a different folder, use the relative path. For example, to link from PO Details (in the PO folder) to Vendor Details (in the COMPANY folder), enter:
..\COMPANY\vendor_contacts.rptdesign

3. In the Report Parameters area, add the following parameters:
    a. Select the where parameter. In the Values field, enter a where clause that specifies the relationship between the current row and the linked report.
    b. Select the appname parameter. In the Values field, enter params["appname"] if the linked report is registered to the same application as the calling report. If it is registered to another application, enter the correct application name, for example "PO".

For example, to link from PO List to PO Details, enter poprint.rptdesign for the target report, and then create the following parameters. Include the quotes as shown:

where	"poline.ponum='" + row["ponum"] + "' and poline.siteid='" + row["siteid"] + "'"
appname	params["appname"]

4. Under "Show target report in", select "Same Frame".

Notes on Hyperlinks:

1. If you are hyperlinking to a report, and a data restriction is in place, make sure to qualify the table (object) name. If it is not qualified, the hyperlinked report may display blank data.

For example, if the report is registered in the SR application

The query should not be: …. pmcomtype is null and status not in ('DRAFT')
Instead, the query should be qualified as: … sr.pmcomtype is null and sr.status not in ('DRAFT')

2. If you design a report to have hyperlinks targeted to the same report, the report output may not change after drilling though more than once from the initial link.

This occurs as a __requestId internal parameter is used to distinguish each report executed by a user from the browser. This __requestId parameter value is unique within the user's current session for the report that is executed.

When hyperlinks are involved, the Report URL for the hyperlink is generated by the report server and does not contain this internal __requestId parameter. Therefore, this parameter will have a value of null for all hyperlinks. Typically, if a hyperlink is for a different report, the null value and the combination of the hyperlinked report name act as a unique key to distinguish the report execution. But, if the hyperlink is for the same report, then any two such links to the same report will be treated as equal, as the key becomes the same.

In the V7 report integration, this unique key is used to get rid of the temporary files created when a report is run again. (For example, if the same report is run again, then the previous report information is discarded using the previous key stored in the HTTP session.) This minimizes the generation of temporary files for repeated execution of the same report. When this logic is combined with the hyperlinks to the same report, the temporary files are never deleted, as multiple executions are treated the same, because the keys are identical. Because of this, the report output does not appear to change.

To resolve the problem, the hyperlink creation has to be forced to generate the __requestId parameter. This can be accomplished by adding a parameter to the hyperlink.

```
<structure>
   <property name="paramName">__requestid</property>
   <expression name="expression">java.lang.System.currentTimeMillis() + (hyperlinkCounter++) </expression>
</structure>
```

Note that the expression has to have a unique key that is unique to the current user and the current report. Since a report can have multiple hyperlinks, be sure to generate links that are unique within the report for that user's execution. Additionally, the hyperlinkCounter has to be declared in the initialization of the report script code.

3. If a user hyperlinks from one report to another, no additional code is required for localization. The language code is passed through internal report context and is not passed as part of the hyperlink.

- Whether the report is a regular report or a hyperlink report, the report has to go through a single servlet that knows about the already logged in user and the user's locale/languagecode/timezone information. This information is automatically passed to BIRT engine or to the scripting code through a framework provided report context.

*Note: For more details on localization or on date restriction requirements and how to enable for reporting, reference the V7 Report Feature Guide.

4. Many out-of-the box reports contain hyperlinks. You may want to review their specific source code for more examples of how hyperlinks are set. To locate which reports contain hyperlinks, access the V7 Report Booklet referenced at the end of this guide, and at the url below. Search the 'V7 Reports' tab for hyperlink to find delivered reports with this functionality.

http://www-01.ibm.com/support/docview.wss?rs=3214&context=SSLKT6&q1=BIRT&q2=BOOKLET&uid=swg21305005&loc=en_US&cs=utf-8&lang=en

Populating the Data Set

If you need more than one data set (usually only required when creating subreports), you may wish to make a copy of the existing data set before starting.

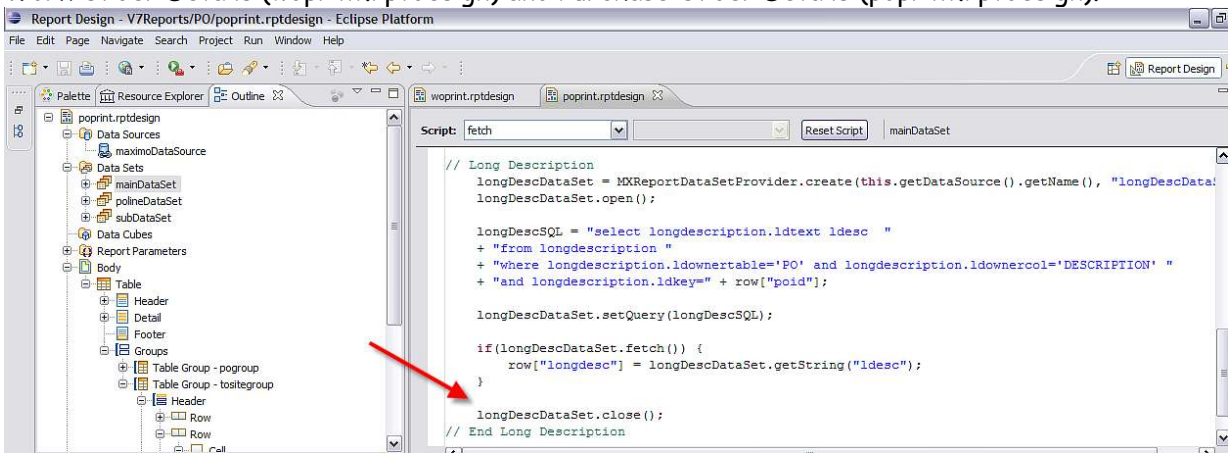
Closing the Data Set

Any data set that is not fully fetched in a loop must be explicitly closed so cursors do not remain open after the report is executed. If the data set is not closed, and the same report is continually executed from the V7 instance over many days, the report may cause a failure of all reports to not show data.

An example of this is shown below using the Long Description data set as an example. Because only one row is fetched from this set, this type of fetch is not closed automatically, and therefore, must change to

```
        if(longDescDataSet.fetch()) {
                row["longdesc"] = longDescDataSet.getString("ldtext");
        }
longDescDataSet.close();
```

Out of the box reports that contain examples of closing the long description data set include Work Order Details (woprint.rptdesign) and Purchase Order Details (poprint.rptdesign).



Note: This is an example of running queries in the fetch() method in the 'Executing Additional Queries' section below that demonstrate that the data set should be closed.

Executing Additional Queries

Additional queries may be run in both the Open and Fetch methods. Each method can have one or more additional queries returning one or more fields.

Queries in the Fetch Method

It is sometimes difficult to provide all data fields for a report with a single SQL statement. You can populate most of the output columns with the main query, and run additional queries to retrieve the remaining fields, for example:

```
if (!maximoDataSet.fetch())
        return (false);

// Set output columns from main query
row["assetnum"] = maximoDataSet.getString("assetnum");

// Execute secondary query
classStrucDataSet = MXReportDataSetProvider.create(this.getDataSource().getName(),"class");
classStrucDataSet.open();

sqlText = "select description from classstructure where classstructureid=? ";
classStrucDataSet.setQuery(sqlText);

// Use value from main query as foreign key in secondary query
classStrucDataSet.setQueryParameterValue(1, maximoDataSet.getString("classstructureid"));

if (classStrucDataSet.fetch())
{
        // Set output columns from secondary query
        row["description"] = classStrucDataSet.getString("description");
}
// Always close the data set
classStrucDataSet.close();

return(true);
```

Dynamically Filtering Data

There are several situations in which you will need to apply a dynamic filter to a report SQL statement. You may filter report results using Report Parameters, which receive values passed from Maximo. You may also use dynamic filters to link multiple queries.

Testing for Null

The COALESCE function is supported on all database types and may be used directly in the query. If you must use a proprietary null conversion function, the following data set method is provided:

maximoDataSet.getNullValueFunction(String param, String nullVal) – Returns NVL, ISNULL, or COALESCE depending on the database type. For example:
"select " + maximoDataSet.getNullValueFunction("parent", "wonum")
evaluates to:
select nvl(parent, wonum) - for Oracle
select coalesce(parent, wonum) - for DB2
select isnull(parent, wonum) - for SQL Server:

If nullVal is a string literal, place it in single quotes:
"select" + maximoDataSet.getNullValueFunction("parent", "'NONE'")

However, be careful with using string literals this way, since they will not be localized.

Scalar Functions

The method MXReportSqlFormat.getScalarFunction(functionName, variable parameters) returns a JDBC scalar function based on the function name and a variable list of parameters. This can be used to access database functions in a database independent manner as suggested in the JDBC specification for commonly used functions.

Parameters

Report parameters are used to filter the report data to meet the user's individual business needs or request. Maximo based reports can execute against a variety of parameter types depending on how they are configured. The three options are:

1. User Inputted Parameters Only

2. Current/Selected/All Record Set

3. Both User Inputted Parameters and the Current/Selected Record Set

Reference: For more details on the functionality of each of these parameters, reference the v7 Report Design Guide noted at the end of this document.

This section will focus on User Inputted Parameters, and their corresponding two types: Bound and Unbound.

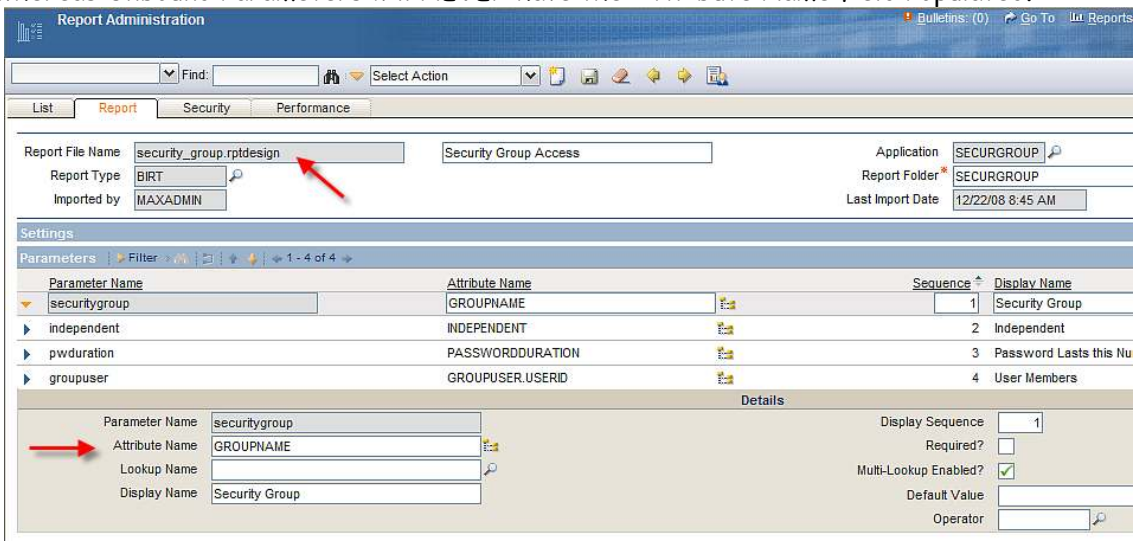
Bound Parameters

Bound parameters either

- exist in the main table of the application the report is registered to or
- exist via a maxrelationship that has been set up for the application.

Bound parameters will be included in the where parameter and do not need to be explicitly included in the report SQL.

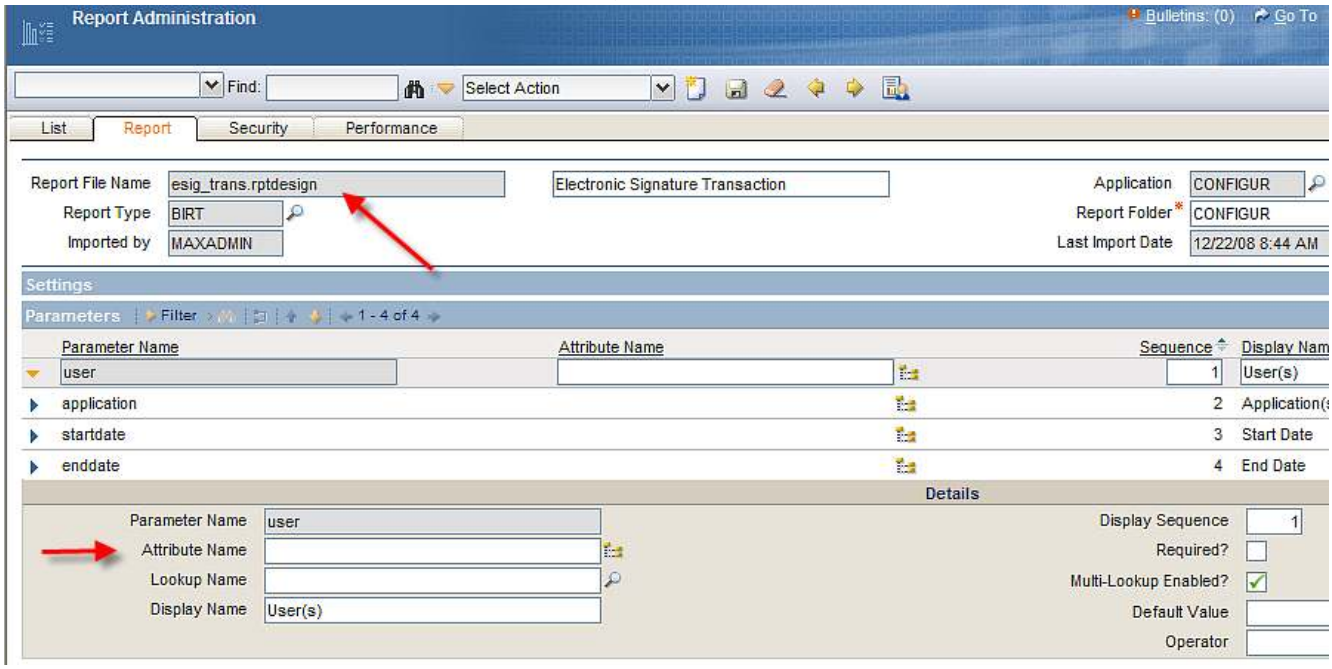
An example of a bound parameter is the Security Group parameter in the Security Group report. Its corresponding entry is shown below from the Report Administration Application. Notice its attribute value. Bound parameters will ALWAYS have the Attribute Name Field Populated – whereas Unbound Parameters will NEVER have the Attribute Name field Populated.



Unbound parameters

- do not exist in the main table of the application and
- are not available through any relationship (defined in maxrelationship) for the main table. Unbound parameters are not included in the where clause.

An example of an unbound parameter is the User parameter in the Electronic Signature Transaction report. This parameter is unbound because it does not exist in the main table of the application (CONFIGUR) and does not exist in one of the maxrelationship to this application. This is shown below. Notice that its Attribute Name field is blank.



The Chart below recaps each of the fields available for parameters in Report Administration, and whether or not they should be populated for bound versus unbound parameters.

	Bound	Unbound
Advantage	Can have lookups, and do not need to be defined in report's design.	Flexibility.
Parameter Name	Do not need to be defined in Report's design file	Must be defined in Report's design file
Attribute Name	ALWAYS Populated	NEVER Populated
Lookup Name	Can either be populated or not	Can only be used for Unbound Dates (*DateLookup Only)
Operator (>, >=, <, <=)	Optional	NEVER Populated
Multi-Lookup Enabled?	Yes or No	Yes or No
Display Sequence	Numeric Value	Numeric Value
Override Label	Any Text	Any Text
Default Value	Can either be populated or not. *NOTE: Default Values are not enabled for localization	Can either be populated or not *NOTE: Default Values are not enabled for localization
Required?	Yes or No	Yes or No
Examples	security_group.rptdesign	eSig_trans.rptdesign

Specifying Bound parameters in the report design

Bound parameters will be added by Maximo to the where parameter, which should be included in the SQL as follows:

```
sqlText="select wonum, description from workorder where " + params["where"];
```

Specifying Unbound parameters in the report design

Unbound parameters must be manually included in the report SQL. The method varies depending on whether the parameter is defined as multi-select or single select.

Multi select unbound parameters.

Multi-select parameters enable users to enter multiple values for a parameter. For example, a user could enter asset1, asset2, asset3 for values in a multi-select asset parameter.

Multi-select parameters will be passed as a comma-delimited string, and must be converted to the correct syntax using the createParamWhereClause method described previously.

For example:

```
"select wonum, description from workorder where " + params["where"] + " and "
+ createParamWhereClause("workorder.status", params["status"]);
```

Single select unbound parameters

Single-select parameters may be directly included, like the where parameter. String parameters must be enclosed in single quotes, and date parameters must be converted to a JDBC function as described in the "SQL Formatting Issues" section. Numbers do not require any special formatting.

For example:

```
sqlText = "select asset, description from asset where " + params["where"]
+ " and asset.siteid = '" + params["siteid"] + "'" //Quoted string
+ " and asset.priority = " + params["priority"]              //Integer
+ " and asset.installdate >= "
+ MXReportSqlFormat.getStartDayTimestampFunction(params["startDate"]);
```

Single-select parameters may also be used in conjunction with data set parameters. In this case, create parameter markers by inserting a question mark in the SQL where each parameter value should go. Then use the data set setQueryParameterValue(int index, Object value) method to resolve them. If there are multiple parameters in one query, they are indexed in the order they appear in the SQL, starting with 1. The query above could be rewritten as:

```
sqlText = "select asset, description from asset where " + params["where"]
+ " and asset.siteid = ? and asset.priority = ? "
+ " and asset.installdate >= "
+ MXReportSqlFormat.getStartDayTimestampFunction(params["startDate"]);

maximoDataSet.setQuery(sqlText);
maximoDataSet.setQueryParameterValue(1, params["siteid"]);
maximoDataSet.setQueryParameterValue(2, params["priority"]);
```

Please review this statement. Is this still true?

Notice that the where and date parameters still use the original format. The where parameter must always be directly included. Unformatted date parameters may be handled as query parameters but currently the JDBC format methods such as getStartDayTimestampFunction cannot be used in conjunction with query parameters

Parsing Unbound Parameters

Unbound parameters will be passed to the report in a comma-delimited string and may contain operators, so the values must be parsed before including in the report SQL. The following method is used for this:

MXReportSqlFormat.createParamWhereClause(String columnName, String paramValue) - Creates a SQL Where clause based on a comma separated list of values contained in paramValue. The parameter value can be specified with a prefix operator where the operator can be any one of <= , < , >= , > , != , = . If no operator is specified, then it assumes that the search is based on operator SQL LIKE. For example:

```
createParamWhereClause("siteid", "=BEDFORD,=MCLEAN")
evaluates to:
((siteid = 'BEDFORD') or (siteid = 'MCLEAN'))

createParamWhereClause("siteid", "!=BEDFORD,!=MCLEAN,TEXAS")
evaluates to:
((siteid != 'BEDFORD') and (siteid != 'MCLEAN')) or ((siteid like '%TEXAS%'))
```

If you have unbound parameters that need to be manually included in the SQL (are not included in the where clause), *do not* directly include them as follows:

```
sqlText = "select asset, description from asset where asset.siteid = '" + params["siteid"] + "'"
```

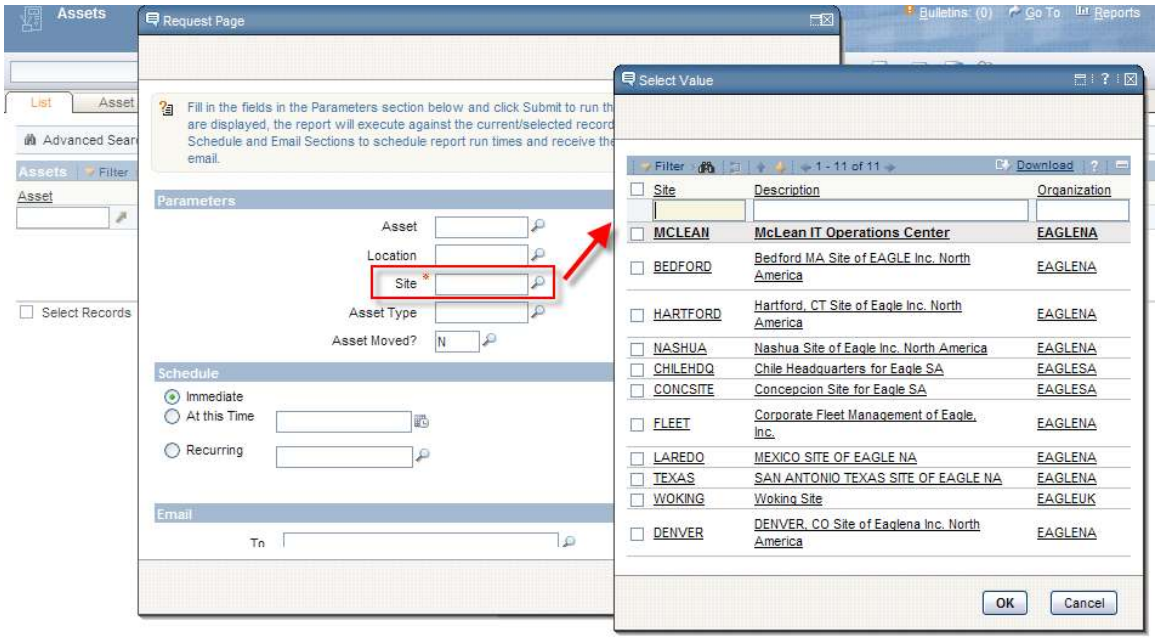
Instead, pass them through the MXReportSqlFormat.createParamWhereClause method:

```
sqlText = "select asset, description from asset where asset.siteid = " +
MXReportSqlFormat.createParamWhereClause("asset.siteid", params["siteid"]);
```

It is advised to use this method on all parameters – not just multi select ones. Use MXReportSqlFormat.createParamWhereClause("<table>.<column>", "="+<value>) when the value is known to be exact. The "=" before the value ensures output as an exact search clause while without it the clause may be generated using like '%<value>%'.

Creating Custom Report Parameter Lookups

You may need to create parameters with lookups for your custom reports. Lookups are accessed from parameters on a report's request page. In the screenshot below, the report's request page with five different parameters is shown below. The lookup for the Site parameter is highlighted.



This section presents a variety of options for you to consider when you need to create custom lookups, including

A. Using valuelists for parameter lookups with fields that have domains
B. Using existing lookups
C. Modifying existing lookups

A. Using valuelists for parameter lookups with fields that have domains

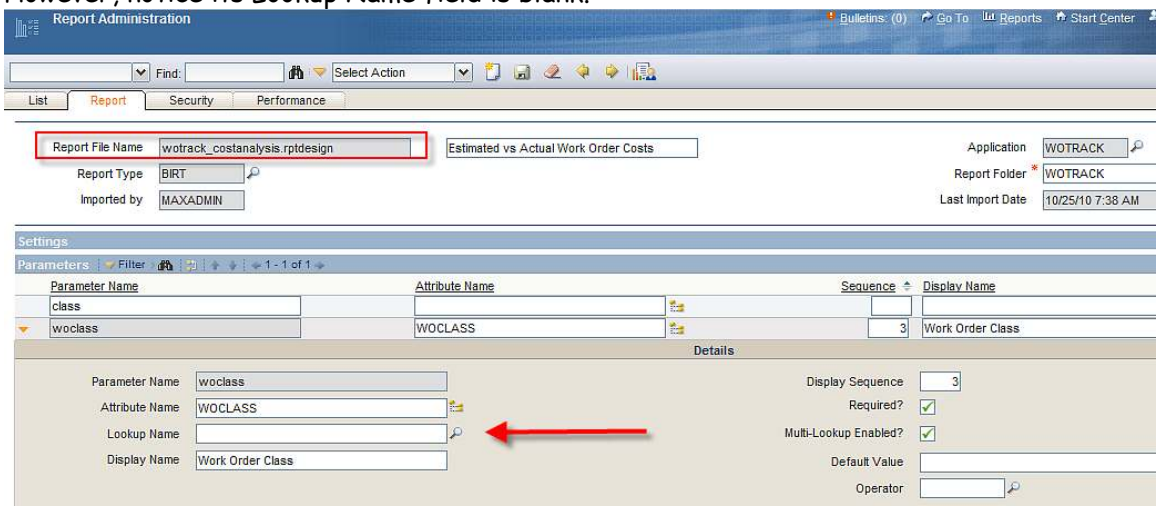
In this method, parameter lookups will be enabled by using valuelists for fields that have domains. Domains have a special status because field validation classes are not required if the field has a domain and the 'valuelist' lookup is used. Lookups for fields with domains can nearly always be used for report parameters.

To illustrate this, a lookup will be created for the Work Order Class parameter on the Version 7 Estimated versus Actual Work Order Cost Report.

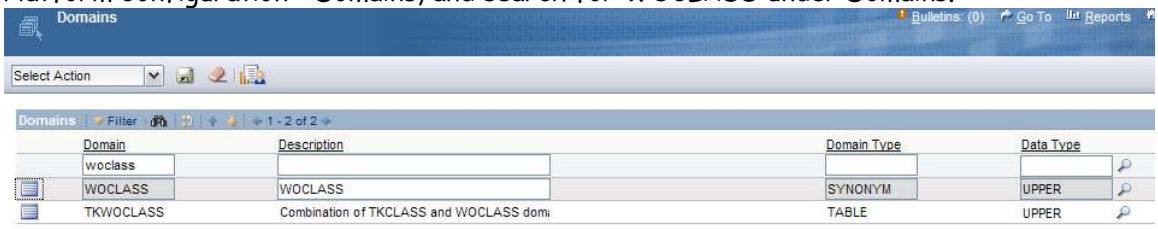
1. Sign into Maximo as a user with access to the following Maximo Applications: Report Administration, Domains, Database Configuration and Application Designer.

2. Access the Report Administration application.

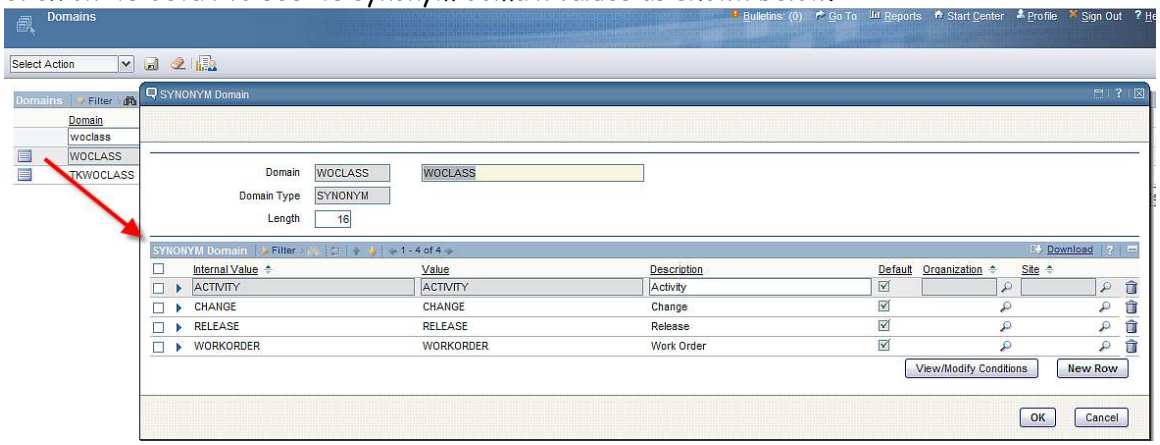
3. Search for the Estimated vs Actual Work Order Cost Report, and open up the Work Order Class parameter. The attribute name for its parameter is populated – so it is a bound parameter. However, notice its Lookup Name field is blank.



4. Next, verify that a domain exists for Work Order Class. Go to System Configuration – Platform Configuration - Domains, and search for WOCLASS under Domains.

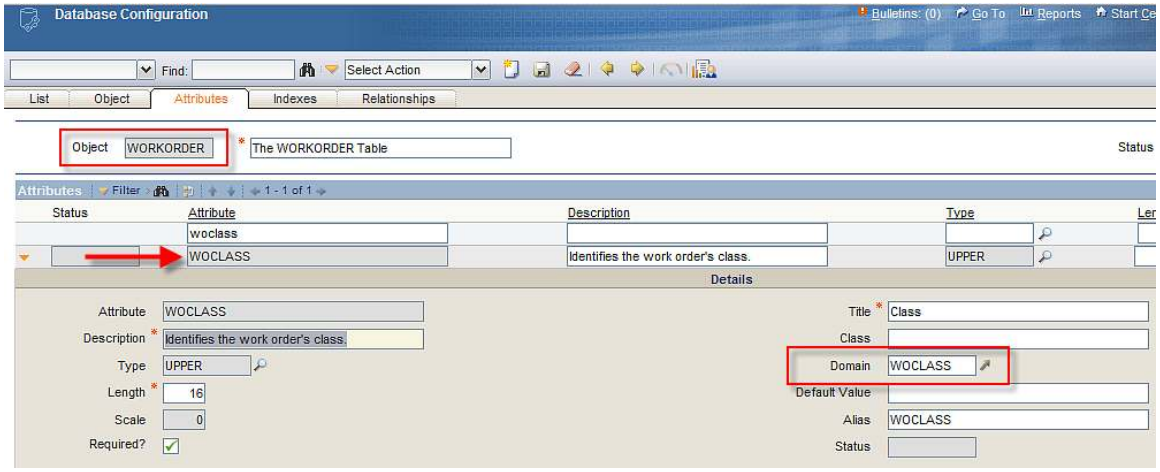


Click on its detail to see its synonym domain values as shown below.



5. Now, verify that the WOCLASS domain is associated with the WORKORDER.WOCLASS attribute. To do this, access System Configuration – Platform Configuration – Database Configuration.

- Search for the Workorder Object.
- Then, search for its attribute WOCLASS. Notice it has a Domain value of WOCLASS.



Notes on Domains:

A. If either the domain, or the attribute's relationship to the domain did not exist, they would have to be created. Details on how to do this are described in the 'Application Developer Guide'

B. For more information on domains, access the 'System Administration Guide'.

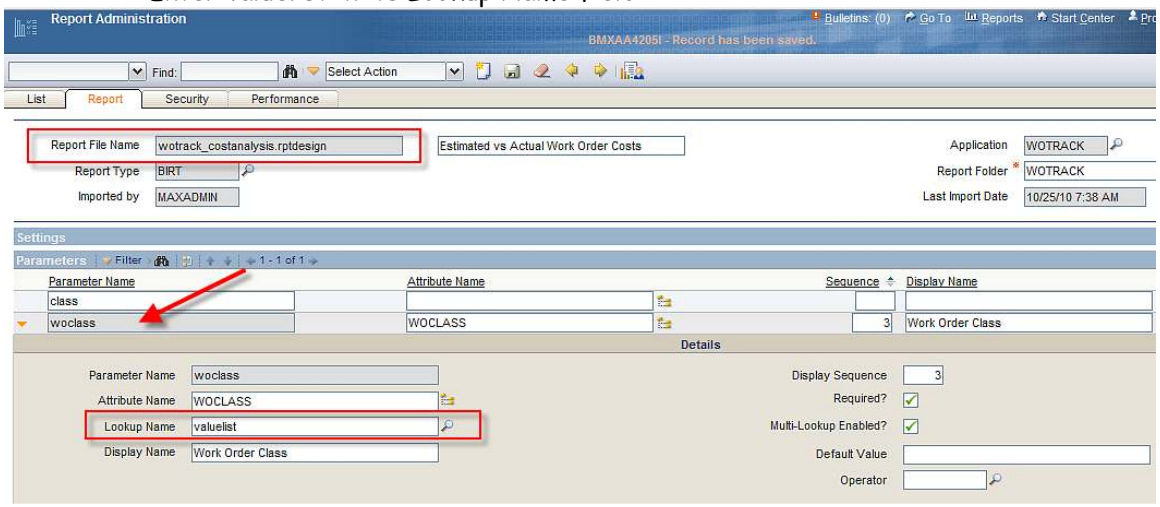
Both of these guides are available at the Information Center below:

http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=/com.ibm.itmaxam.doc/welcome.htm

6. In this step, the attribute's domain will be added to the report, so lookup values can be enabled from the parameter.

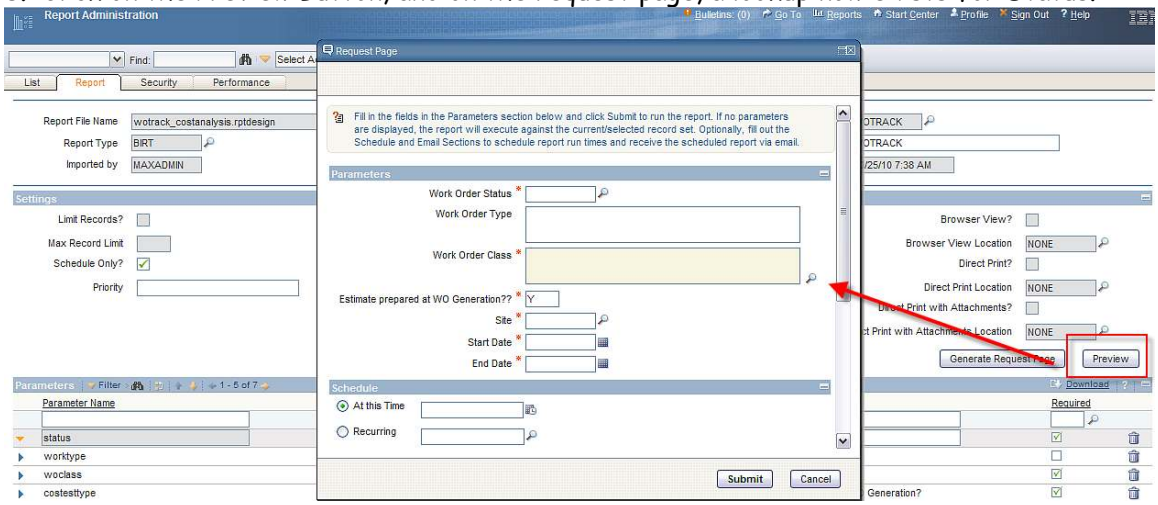
To do this, go back to the Report Administration application.

- Locate the report, and open up the work order class parameter.
- Enter valuelist in its Lookup Name field

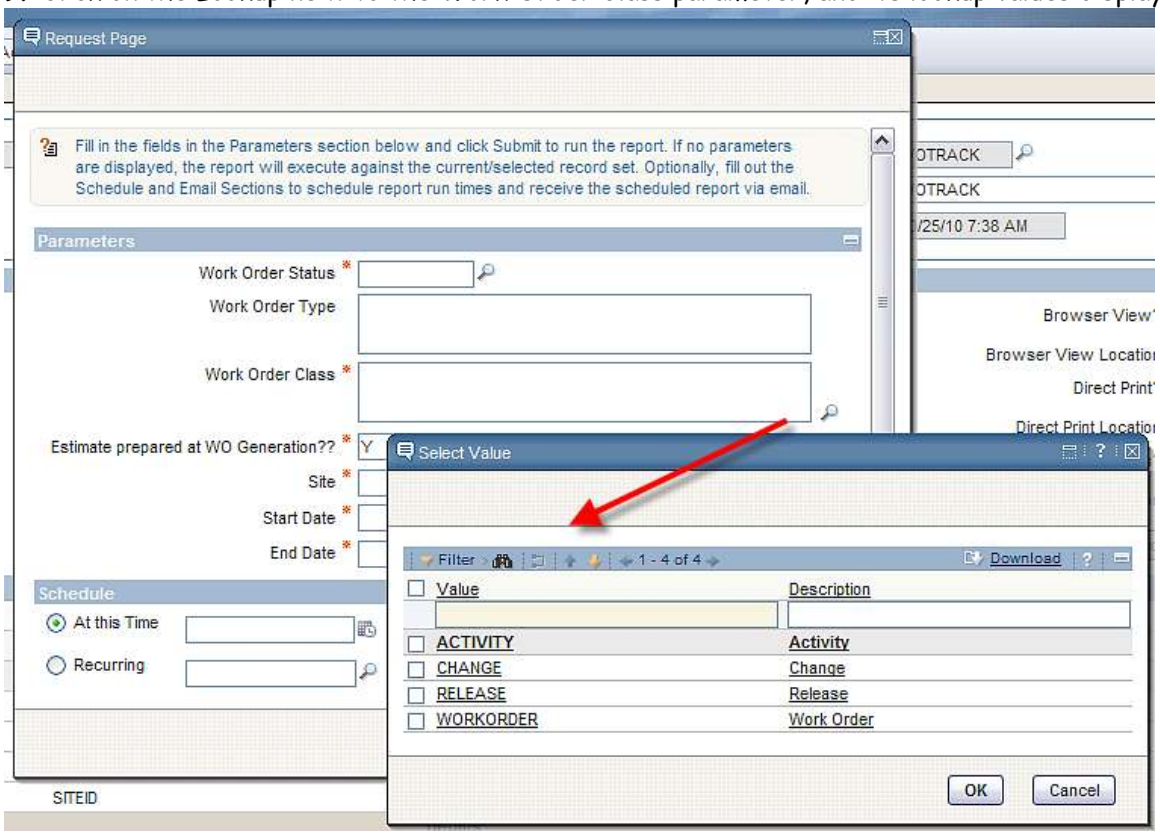


7. Save the change, and recreate the report xml by clicking on the button 'Generate Request Page'.

8. Click on the Preview Button, and on the request page, a lookup now exists for Status.



9. Click on the Lookup next to the Work Order Class parameter, and its lookup values display.



B. Using existing lookups

It may be possible to use existing Maximo lookups with custom bound report parameters. Use the lookup on the Lookup Name field in Report Admin to browse the available lookups.

This is a trial-and-error process since Maximo lookup behavior is controlled by field classes, which are classes that are assigned to the attribute definition in Database Configuration. Many of the default lookups will not work correctly when applied to report parameters, either because there is no field class for the bound attribute, or because there is logic in the field class that inappropriately limits the results of the lookup. In these cases the lookup may return no results, a subset of the expected results, or may contain Invalid Bindings.

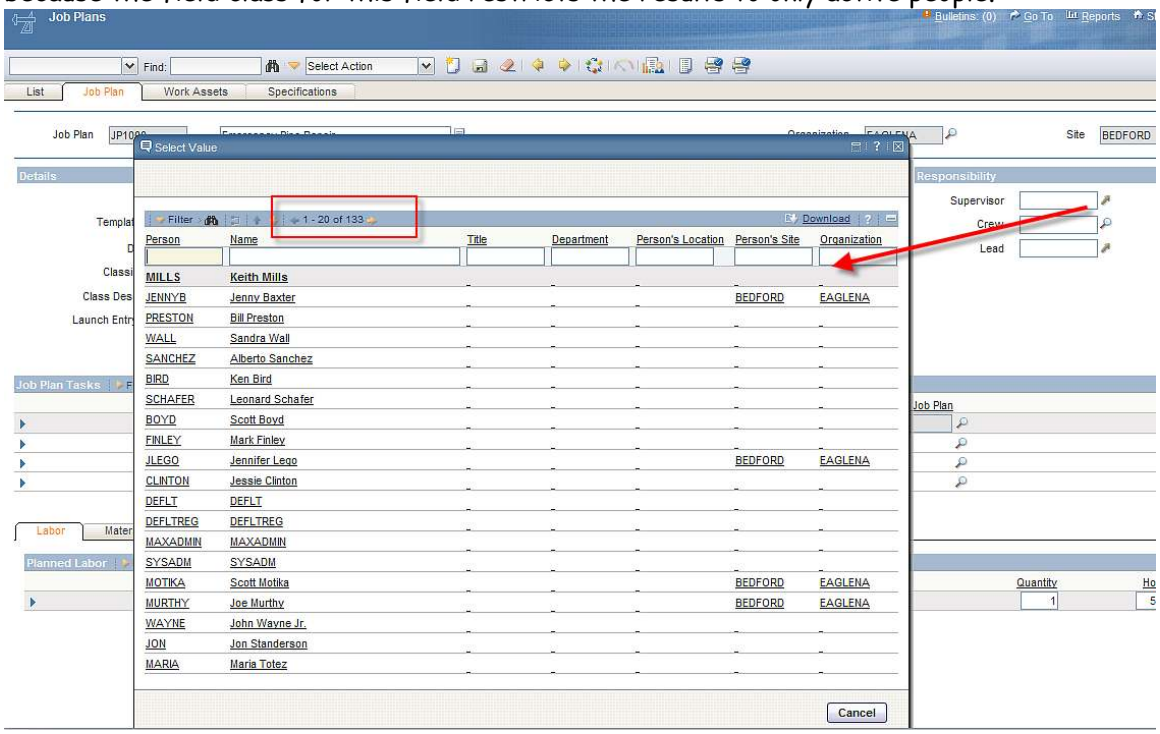
With this method, you simply try out the lookup(s) you identify as possible candidates and evaluate whether they return the desired results. You can use SQL logging to examine the query used to populate the lookup to ensure there are no inappropriate filters applied.

C. Modifying existing lookups

If the lookup attribute does not have the required field class, or the field class is not configured to provide the expected values, it is possible to produce the desired results by creating a copy of the lookup and specifying a value for the mboname attribute. This method also has the advantage that you can modify the fields included in the lookup.

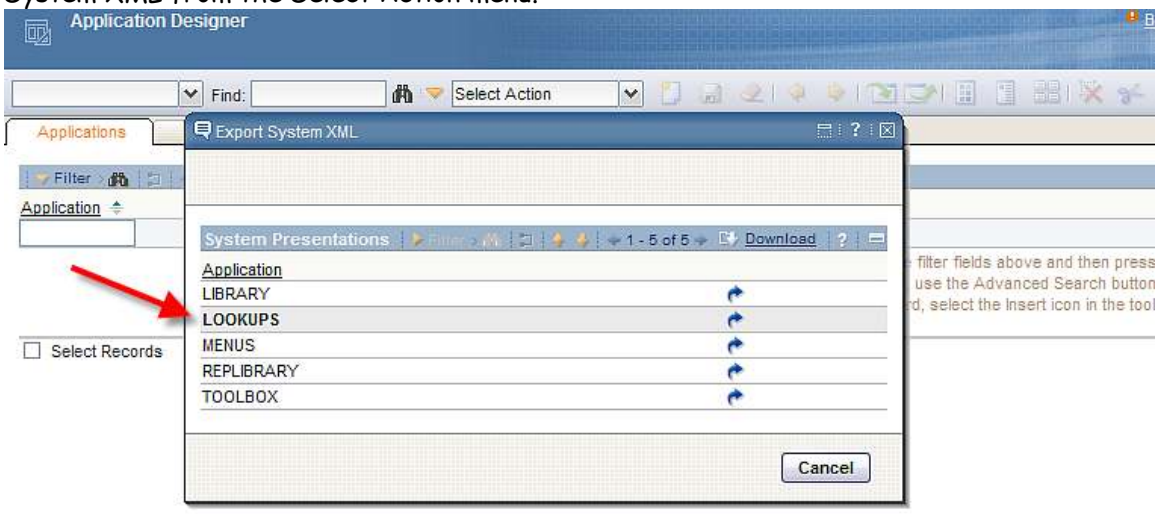
As an example, the person lookup will be modified to use with a parameter bound to the supervisor field in the Job Plan application.

This existing lookup is shown below within the Job Plan application. Notice that the lookup includes 133 people instead of the full 134 in the person table for the maxdemo database. This is because the field class for this field restricts the results to only active people.



To modify the existing lookup for reporting, follow the steps below.

1. Go to System Configuration – Platform Configuration - Application Designer and select Export System XML from the Select Action menu.



2. Open the file in a text editor. Locate the person lookup by searching for id=person. The first line of this is shown below.

```
<table id="person" inputmode="readonly" selectmode="single">
```

3. Copy the person lookup and scroll to the bottom of the file. Insert some lines before the </systemlib> .

4 Paste the copied person lookup, and then modify the first line to include the mboname attribute, for example:

<table id="person" inputmode="readonly" selectmode="single" >

Should be updated to

<table id="person_rpt" inputmode="readonly" selectmode="single" mboname="person" >

5. Replace all remaining occurrences of '**id="person**' with '**id="person_rpt**', for example:

<tablebody id="person_lookup_tablebody" filterexpanded="true" filterable="true" displayrowsperpage="20" >

would become

<tablebody id="person_rpt_lookup_tablebody" filterexpanded="true" filterable="true" displayrowsperpage="20" >

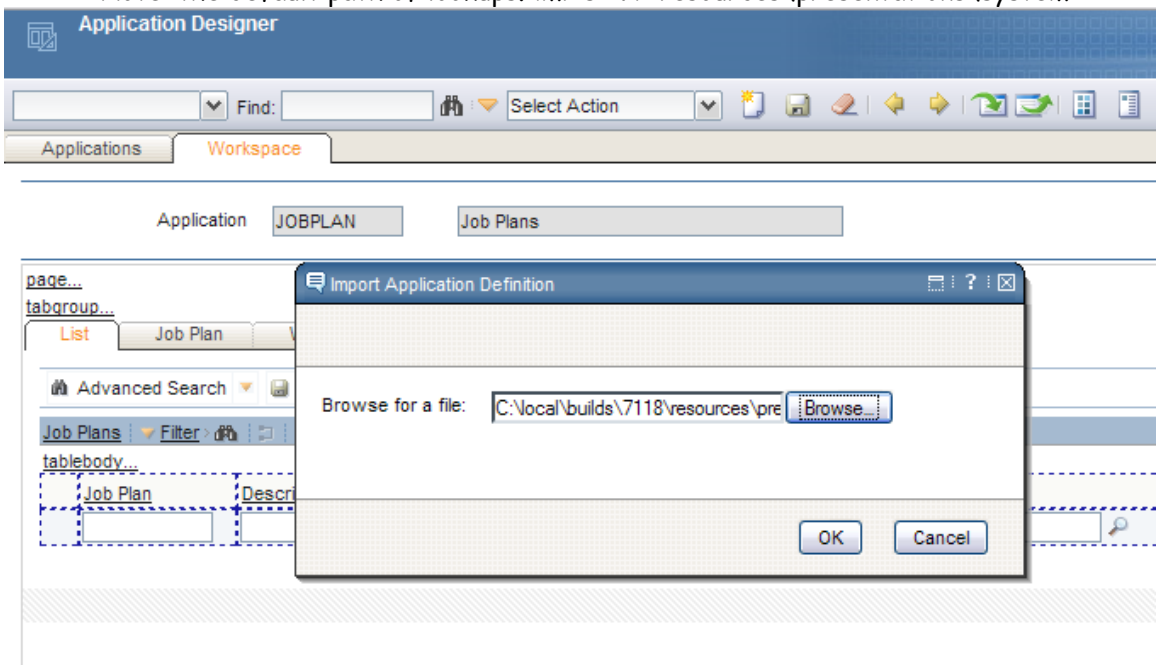
The new entire entry is shown below.

```
        </tablebody>
    </table>
<table id="person_rpt" inputmode="readonly" selectmode="single" mboname="person" >
        <tablebody displayrowsperpage="20" filterable="true" filterexpanded="true" id="person_rpt_lookup_tablebody">
                <tablecol dataattribute="personid" id="person_rpt_lookup_tablebody_col_2" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
                <tablecol dataattribute="displayname" id="person_rpt_lookup_tablebody_col_5" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
                <tablecol dataattribute="title" id="person_rpt_lookup_tablebody_col_6" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
                <tablecol dataattribute="department" id="person_rpt_lookup_tablebody_col_7" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
                <tablecol dataattribute="location" id="person_rpt_lookup_tablebody_col_8" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
                <tablecol dataattribute="locationsite" id="person_rpt_lookup_tablebody_col_9" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
                <tablecol dataattribute="locationorg" id="person_rpt_lookup_tablebody_col_10" mxevent="selectrecord" mxevent_desc="Go To %1" sortable="true" type="link"/>
        </tablebody>
    </table>
```

5. The changes made to the xml file now have to be imported into Version 7. To do this, go back to the Application Designer.

6. Click on the 'Import Application Definition' Icon in the toolbar. Browse to the location of the lookups.xml file that you modified.

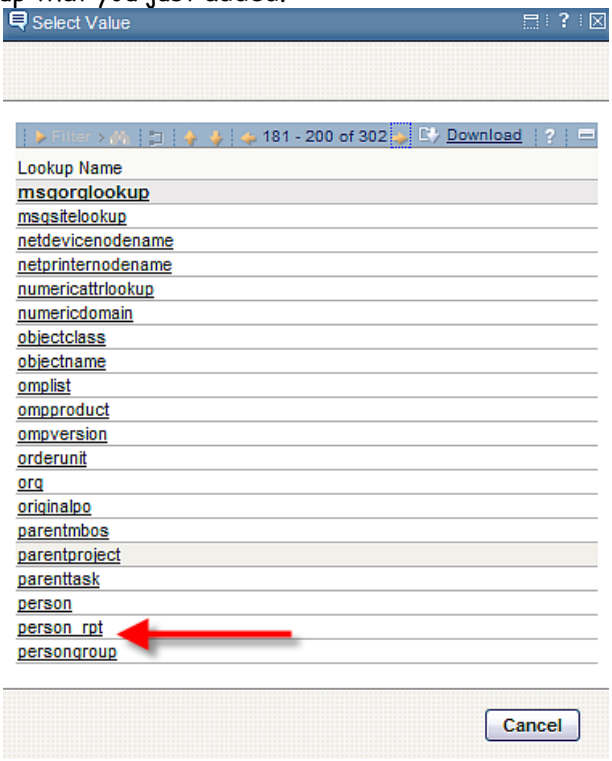
Note: the default path of lookups.xml is <V7>resources\presentations\system



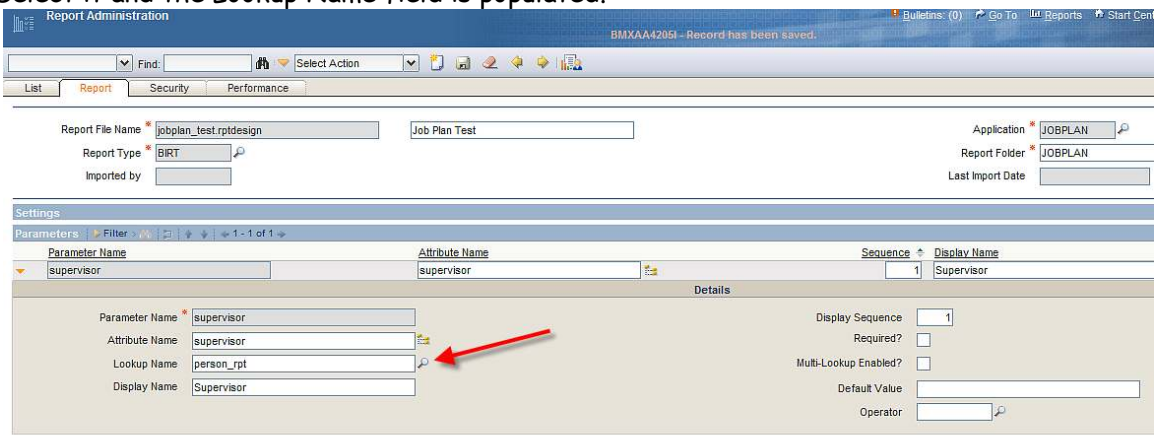
7. Click OK. When it is completed, a message will display in the toolbar.

8. Next, access the Report Administration application. Add a new sample report called Job Plan Test, with a supervisor parameter.

In the Parameter's Lookup Name field, click on the lookup. Scroll to find the new person_rpt parameter lookup that you just added.

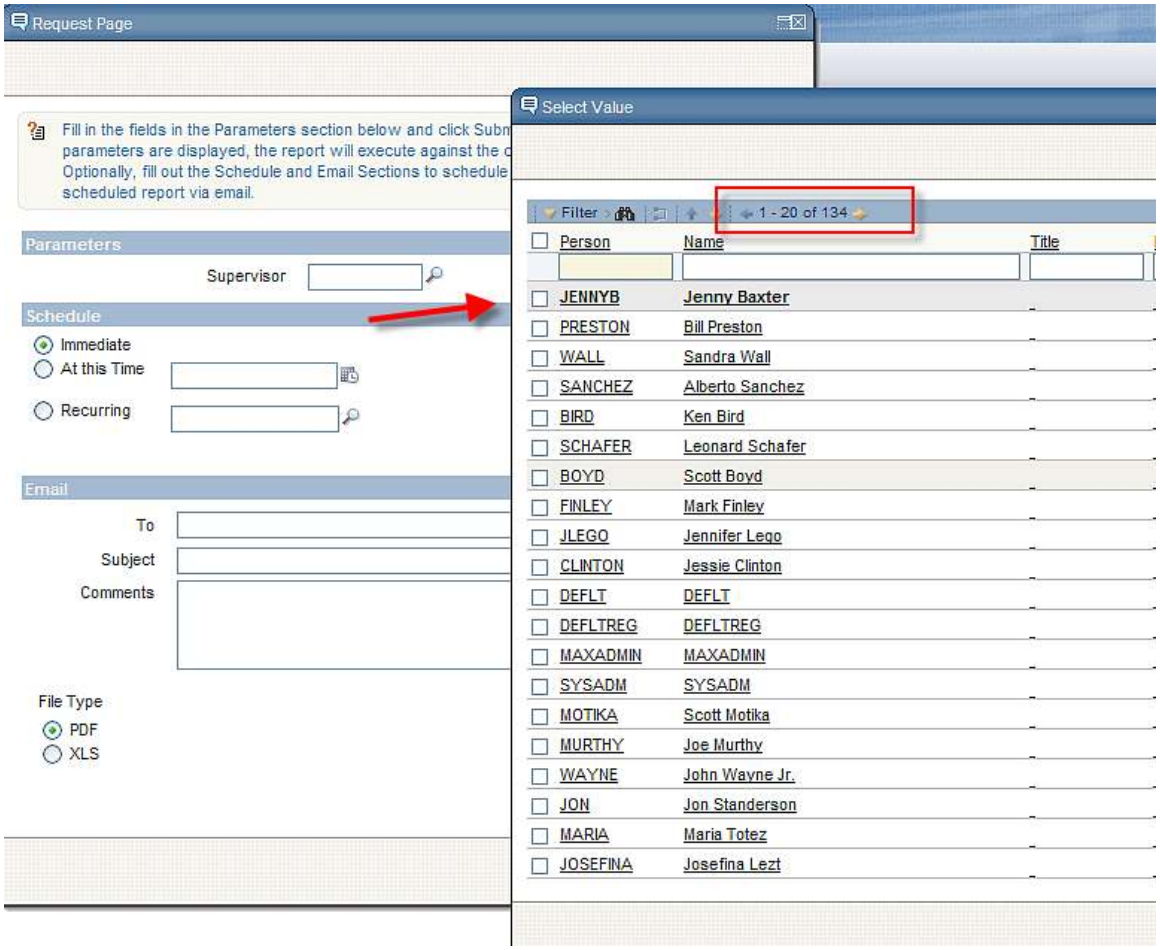


8. Select it and the Lookup Name field is populated.



9. Save the record. Generate the report xml for the test report you have just registered.

After the XML has been created, click on Preview. Next to the Supervisor parameter, a lookup now exists. Click on the Lookup, and the values for Supervisor appear. Notice that the person records are no longer filtered, and all records appear.


Request Page
Fill in the fields in the Parameters section below and click Subn
parameters are displayed, the report will execute against the c
Optionally, fill out the Schedule and Email Sections to schedule
scheduled report via email.
Parameters
Supervisor
Schedule
Immediate
At this Time
Recurring
Email
To
Subject
Comments
File Type
PDF
XLS
Select Value
Filter
1 - 20 of 134
Person
Name
Title
JENNYB Jenny Baxter
PRESTON Bill Preston
WALL Sandra Wall
SANCHEZ Alberto Sanchez
BIRD Ken Bird
SCHAFER Leonard Schafer
BOYD Scott Boyd
FINLEY Mark Finley
JLEGO Jennifer Lego
CLINTON Jessie Clinton
DEFLT DEFLT
DEFLTREG DEFLTREG
MAXADMIN MAXADMIN
SYSADM SYSADM
MOTIKA Scott Motika
MURTHY Joe Murthy
WAYNE John Wayne Jr.
JON Jon Standerson
MARIA Maria Totez
JOSEFINA Josefina Lezt

Parameter Notes

Number of Parameter Values

1. The maximum number of User Inputted Report parameters that are enabled for reports is 23. These include 15 Non-Date Time Parameters, and 8 Date-Time parameters. If more than the 15 Non-Date and 8 Date-Time Parameters are entered, invalid bindings will display on the report's request page.

Utilizing Parameter Values on a Report's Request Page

2. Bound and Unbound parameters will behave the same way when a user enters values on the Request Page. This means that if there is a parameter for Status, the following will occur:

User entered parameter value	Report Results
=APPR	Records where status = APPR
APPR	Records where status = WAPPR, APPR
%APPR	Records where status = WAPPR, APPR

Boolean Parameter Values

3. Reports that use boolean values as parameters must follow the guidelines below:

A. The parameter in the report design must be defined as a string type
This is required for localization purposes.

B. If the parameter value is required to be passed to a SQL statement, then the parameter value must be converted to integer value (1 or 0) as the database has 1 or 0
An API call has been added to the dataset code (getBooleanInteger(string)) that can be used for this purpose. Here is an example:

```
var isActiveFlag = params["isactive"];
mySQL = "select isactive from collection where isactive=?";
myDataSet.setQuery(mySQL);
myDataSet.setQueryParameterValue(1, myDataSet.getBooleanInteger(isActiveFlag));

or

mySQL = "select isactive from collection where isactive=?";
myDataSet.setQuery(mySQL);
myDataSet.setQueryParameterValue(1,
myDataSet.getBooleanInteger(params["isactive"]));
```

Optional Parameters

4. Optional parameters are best handled by direct inclusion. In the following example, site and start date are optional. If values are specified, they are appended to the where parameter (to preserve the existing where parameter content). Priority is still mandatory:

```
var where = params["where"];
if (params["siteid"].value)
where = where + " and " + asset.siteid = '" + params["siteid"] + "'";
if (params["startdate"].value)
where = where + " and matusetrans.actualdate >= " +
MXReportSqlFormat.getStartDayTimestampFunction(params["startdate"]);
sqlText = "select asset, description from asset where " + params["where"]
+ " and asset.priority = " + params["priority"];
```

YORN Lookup

5. In the 7.1.1.4 release, a lookup has been added for Yes or No values. This lookup can be used in reports to eliminate the question of 'Do I enter Yes or Y or 1?' in a parameters value. For the Out of the Box Reports, the Security Group Access report (security_group.rptdesign) has been updated to include the new YORN lookup. This can be used as an example of how you can apply this lookup.

A condensed version of the reports.xml for this report is below to show how it's the YORN parameter is set. To find the complete version, access the file under <7114>\reports\birt\reports\USER

```
<report name="security_group.rptdesign">
        <parameters>
        <parameter name="independent">
                <attribute name="attributename">INDEPENDENT</attribute>
                <attribute name="lookupname">yornlookuplist</attribute>
                <attribute name="sequence">2</attribute>
                <attribute name="labeloverride">Independent</attribute>
                <attribute name="defaultvalue">false</attribute>
```

Viewing Parameters

6. If you drag parameters directly on to the report, you will receive the following errors in the Web Viewer, although report content will not be affected:

A report document error occurred when loading: Subquery
A report document error occurred when loading: Result Class

This happens because the bindings are created only at the cell level, not at the table level. To ensure the correct binding, insert Data elements and using the Expression Builder, set the values to the parameters (choose "Report Parameters" from the Category window).

Requirements for using lookups with Parameters

7. To enable a parameter lookup, the parameter must have an equivalent attribute. This makes the parameter bound as noted in the beginning portion of the parameter section.

Additionally, as noted above, unbound parameter values which have no attributes, cannot have a lookup. Domain lookups can only be used when bound to a field that has the domain assigned to it. The only two exceptions to this are the datelookup and yornlookuplist which can be associated to unbound parameter values.

Extending Ad Hoc Reports in BIRT Designer

To reduce report development time, you can utilize the Ad Hoc reporting functionality as an excellent starting point for your custom report development. The section below details how this can be enabled.

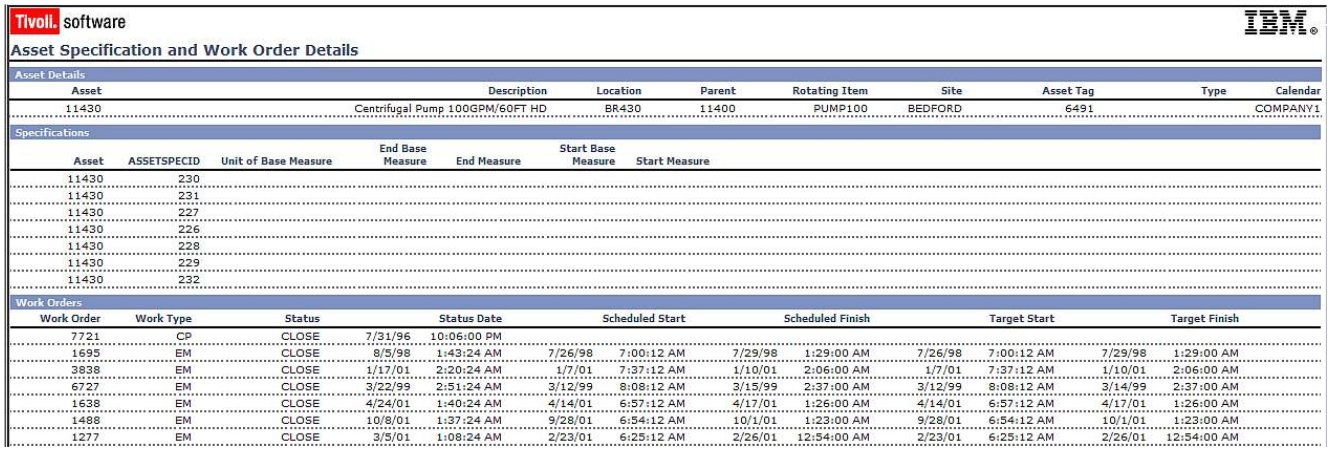
Use Ad Hoc Reporting as a base for Custom Report Development

When Ad Hoc Report is created and saved, its resulting design file (.rptdesign) that was created on the fly is saved to the database. This enables the report to be accessed in the future for immediate run access or scheduling and emailing.

Additionally, once the report is saved in the database, it can be extended within the Report Designer tool. By simply exporting the report and opening it in the design tool, you can quickly build upon the report design by adding calculations, graphs or additional features.

This can become a huge time saving feature for your custom report development because ad hoc reports can be created with complex sql from multiple tables, filters and application queries. Instead of having the developer create all this information – you can let the Maximo framework perform this work, and then build upon the ad hoc report in the designer.

To show how this can be accomplished, an example below is given. First, the Ad Hoc report is created and saved. In this case, the report is created in the Asset Application, and called 'Asset Specifications and Work Order Details.'



Tivoli. software IBM.

Asset Specification and Work Order Details

Asset Details

Asset	Description	Location	Parent	Rotating Item	Site	Asset Tag	Type	Calendar
11430	Centrifugal Pump 100GPM/60FT HD	BR430	11400	PUMP100	BEDFORD	6491		COMPANY1

Specifications

Asset	ASSETSPECID	Unit of Base Measure	End Base Measure	End Measure	Start Base Measure	Start Measure
11430	230					
11430	231					
11430	227					
11430	226					
11430	228					
11430	229					
11430	232					

Work Orders

Work Order	Work Type	Status	Status Date	Scheduled Start	Scheduled Finish	Target Start	Target Finish
7721	CP	CLOSE	7/31/96 10:06:00 PM				
1695	EM	CLOSE	8/5/98 1:43:24 AM	7/26/98 7:00:12 AM	7/29/98 1:29:00 AM	7/26/98 7:00:12 AM	7/29/98 1:29:00 AM
3838	EM	CLOSE	1/17/01 2:20:24 AM	1/7/01 7:37:12 AM	1/10/01 2:06:00 AM	1/7/01 7:37:12 AM	1/10/01 2:06:00 AM
6727	EM	CLOSE	3/22/99 2:51:24 AM	3/12/99 8:08:12 AM	3/15/99 2:37:00 AM	3/12/99 8:08:12 AM	3/14/99 2:37:00 AM
1638	EM	CLOSE	4/24/01 1:40:24 AM	4/14/01 6:57:12 AM	4/17/01 1:26:00 AM	4/14/01 6:57:12 AM	4/17/01 1:26:00 AM
1488	EM	CLOSE	10/8/01 1:37:24 AM	9/28/01 6:54:12 AM	10/1/01 1:23:00 AM	9/28/01 6:54:12 AM	10/1/01 1:23:00 AM
1277	EM	CLOSE	3/5/01 1:08:24 AM	2/23/01 6:25:12 AM	2/26/01 12:54:00 AM	2/23/01 6:25:12 AM	2/26/01 12:54:00 AM

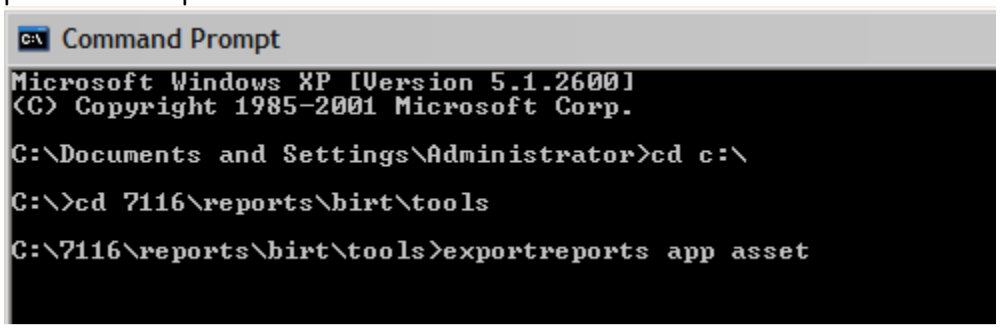
Then, the report is exported for its repository in the database to a local file system. This is done via a command utility. The command utility uses a property file to enable this. Therefore, you must configure the reporttools.properties file. It is located at: <V7>\reports\birt\tools

```
# HostName or IP address of the machine that has MAXIMO application running in an
App Server
maximo.report.birt.hostname=localhost

# HTTP port of the application server (the port used to access maximo from browser)
maximo.report.birt.port=9080

# Indicates whether the SSL communication is enabled or not
```

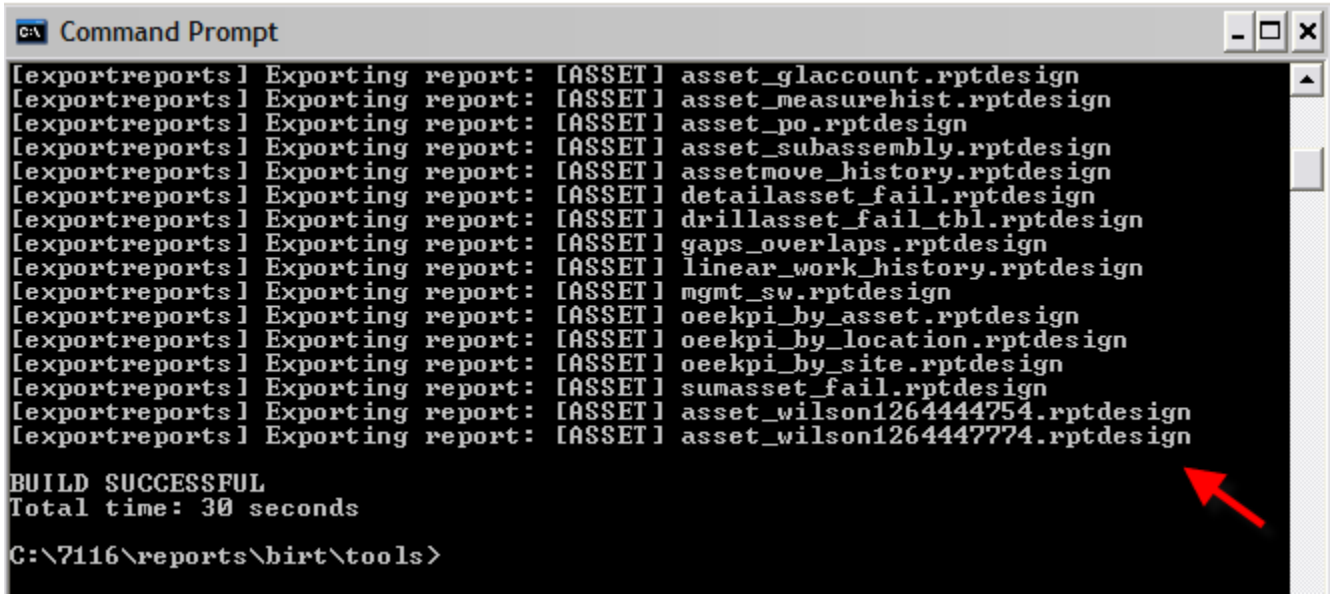
```
maximo.report.birt.ssl=false

# User that has access to perform the operation
maximo.report.birt.username=wilson

# Password of the user that has access to perform the operation
maximo.report.birt.password=wilson

# Output folder used for the export operation
maximo.report.birt.outputfolder=c:/7116/reports/birt/export/asset
```

The output folder highlighted in red details the location of where the exported Ad Hoc Report will be placed.

To export an Ad Hoc Report, open a command window. Navigate down to the exportcommands path…. <V7>reports\birt\tools

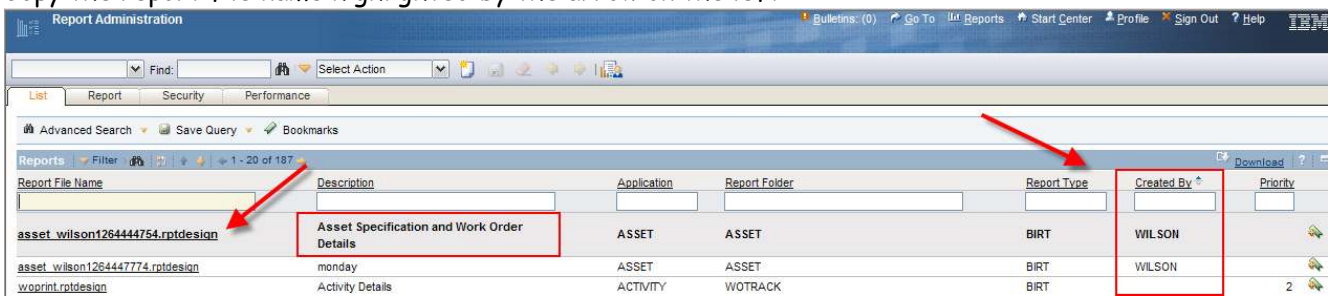


Then, export all of the reports from the asset application by:
exportreports app asset

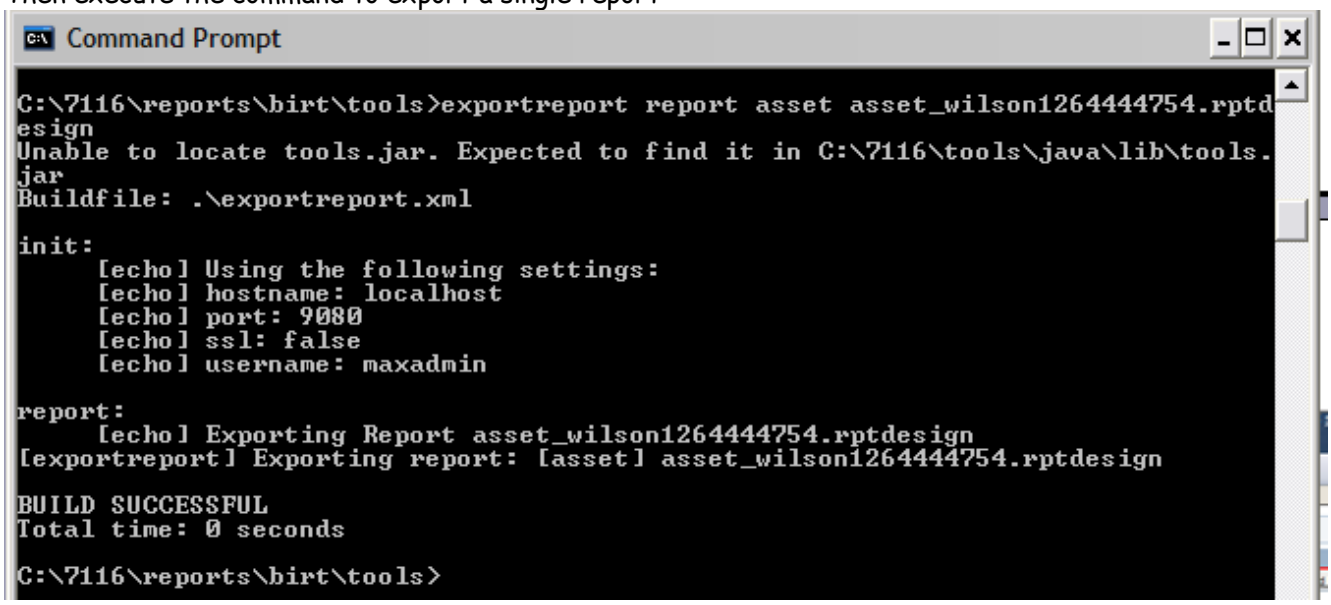


*Note: You can also export only a single report by using the command:
exportreport report <application> <reportfilename>

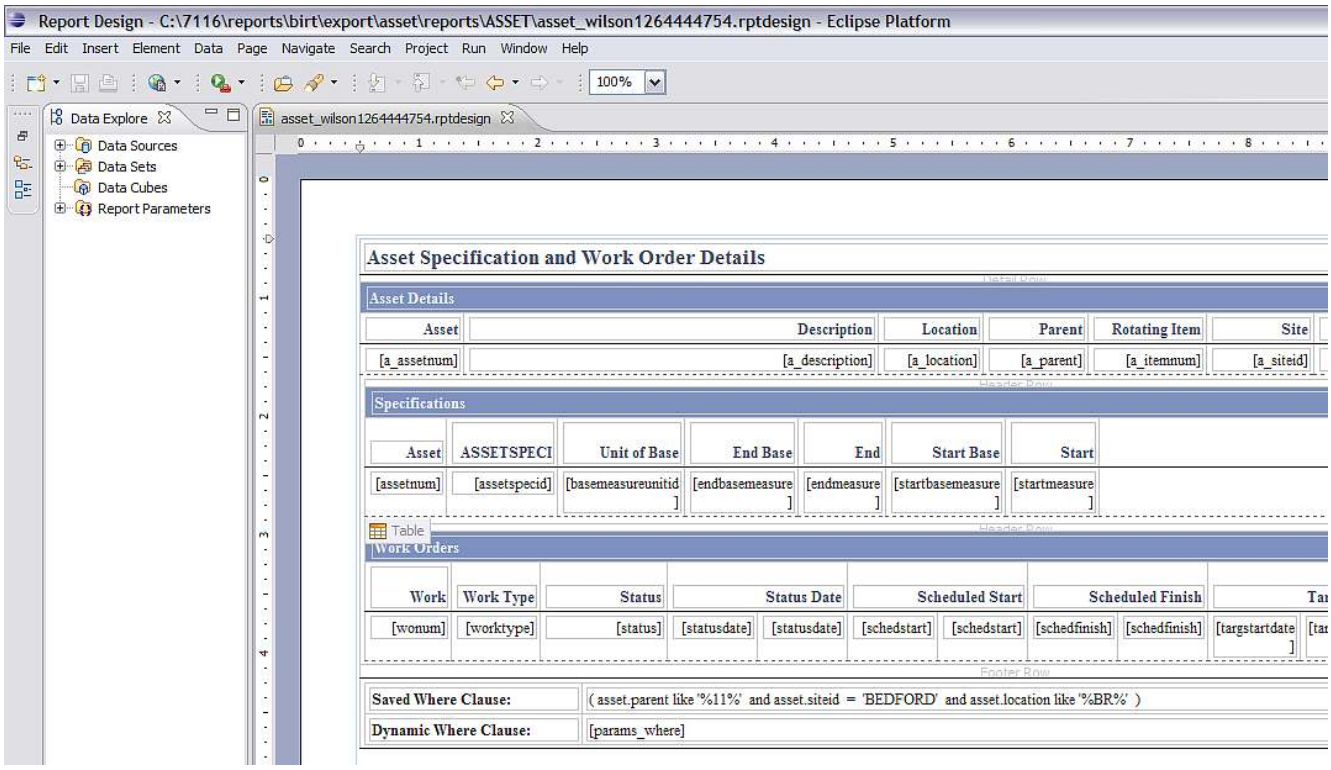
However, all command files work with file names – not the report descriptions that users assign during Ad Hoc Report Creation. So to use this command, you first must know the exact name of the QBR report. To find this name, access Report Admin, and filter on Created By. Ad Hoc reports are identified where the Created By field is not null.
Copy the report file name highlighted by the arrow on the left



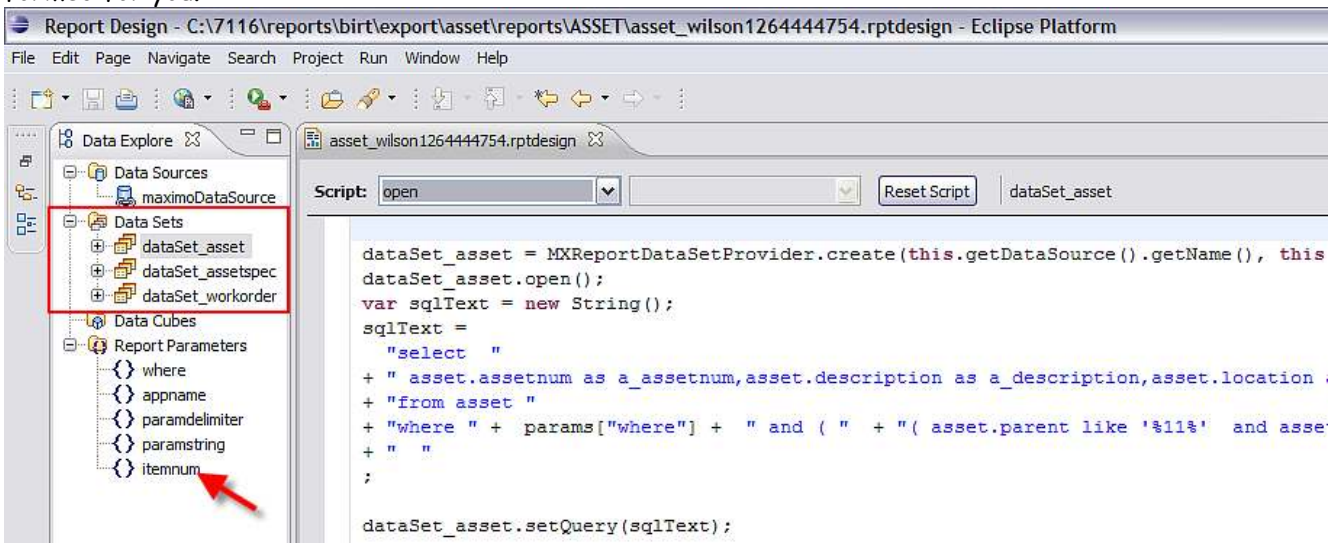
Then execute the command to export a single report



Once you have exported the ad hoc report, open up the Report Designer tool From the menu, click File – Open File and navigate to the directory where you exported your reports. Select the Ad Hoc Report's .rptdesign file, and it displays in the designer.



You can immediately see that you have an excellent beginning to extend this report further for any other customizations you may need. Multiple data sets (subreports) can be already populated, parameters included and complex sql statements including application queries can already be formed for you.



This can become an excellent starting point for your report developer.

Note: If the developer chooses to modify the design file, it is recommended that the report file name be modified to identify it from the original file. If the developer plans on utilizing this report as an Enterprise Report, he would need to create and/or append the reports.xml and properties file for the new enterprise report. Additionally, because it is now an enterprise report, it would need to be imported through the reports import command, or the UI utility in the Report Administration application.

Localization

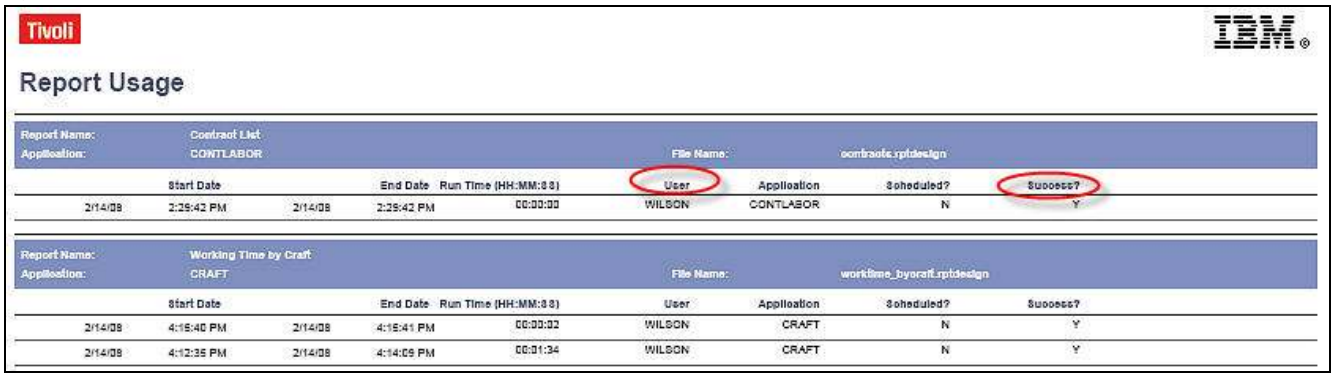
This section details how the report labels and its data can be enabled for localized values.

Reference: For more details on localization, including how to enable it within the V7 environment, reference the V7 Report Feature Guide noted at the end of this document.

How localizing report labels works

Within the report design file, you can enable the report labels for localization. This is enabled by the key value associated to all report labels and titles within the design file. The key value is simply a unique text value of the control. These key values are stored in the REPORTLABEL Database Table.

This can be shown in the example below of the Report Usage Report, reportusage.rptdesign. Two of its labels are circled – User and Success.



These two label values are stored in the REPORTLABEL table shown below.

LABELKEY	LABELVALUE	LANGCODE
user	User	EN
transientreports	Transient Reports	EN
success	Success?	EN
startdate	Start Date	EN
scheduled	Scheduled?	EN
runtime	Run Time (HH:MM:SS)	EN
reportusage.reportname	Report Usage	EN
reportname	Report Name:	EN
recordsselected	Records Selected:	EN
filename.header	File Name:	EN
filename	File Name	EN
enddate	End Date	EN
application.header	Application:	EN
application	Application	EN

For clients using a language other than English, or for clients enabling multiple languages, corresponding database tables identified by L_REPORTLABEL will be used. These tables will hold the translated value(s) of the label values.

Multiple values for a given label value can be stored in the L_REPORTLABEL table, one for each language. The combination of the Owner ID (from REPORTLABEL) and Language Code will make the label value unique.

L_REPORTLABEL.REPORTLABELID	L_REPORTLABEL.LANGCODE	L_REPORTLABEL.LABELVALUE
12345	ES	mi primer informe
12345	FR	mon premier rapport
12346	ES	fecha
12346	FR	date

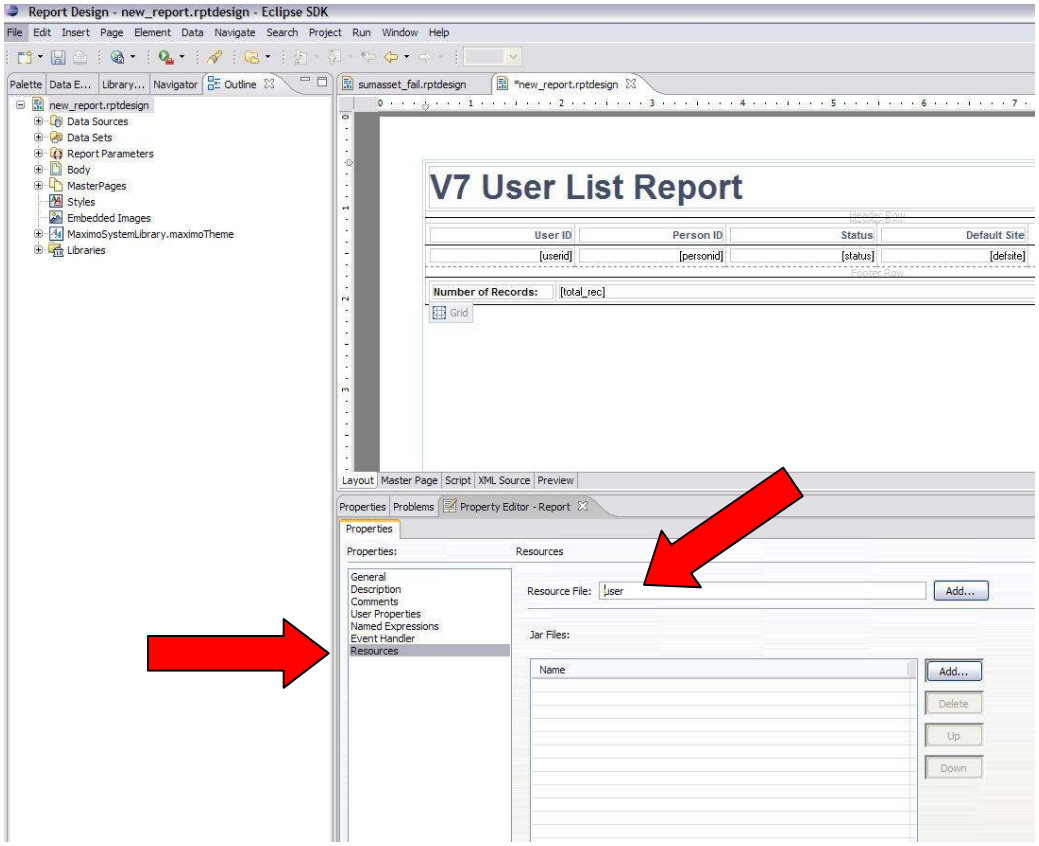
Labelkey and labelvalues for each control in each report are defined by the developer while he creates the report in the report designer tool. This is done through the properties file using the process described below:

How to enable localization of report labels within report design file

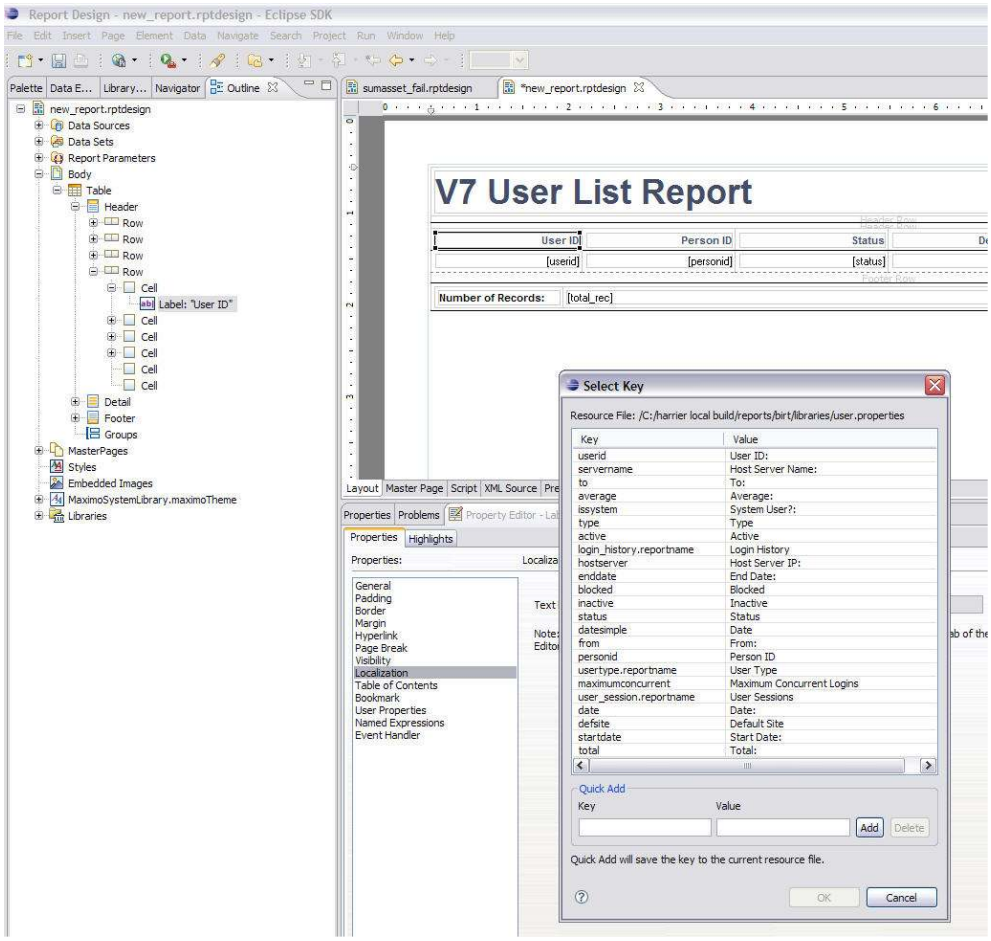
To enable a report design file to use localized report label values, follow the steps below.

1. First, associate the report with the label properties file. In the Outline tab, highlight the report name. In Property Editor - Properties, select Resources. In the Resource File Field,click on the 'Add' Button.

If a properties file already exists for your application, select it. If not, enter the name of the new properties files which corresponds to the application. In the example below of a User List report, the file USER already exists in <V7>\reports\birt\libraries.



2. Finally, enable the labels for localization, by highlighting one with your mouse. Select the Localization property below, and click the browse icon …



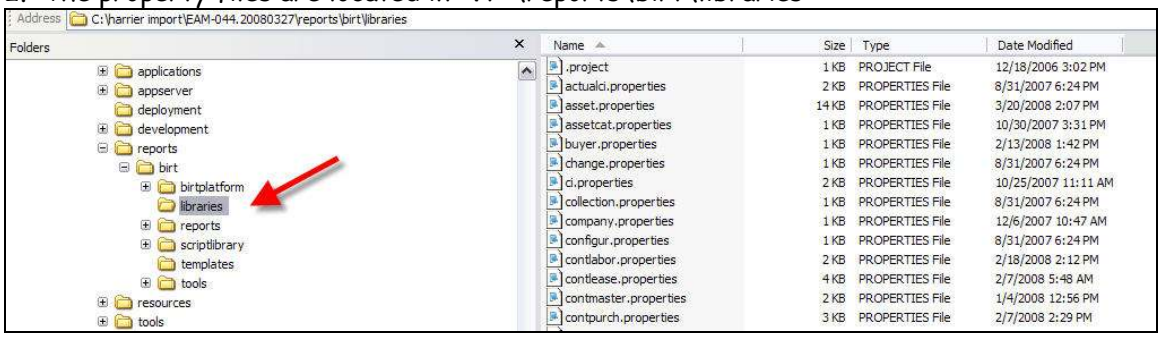
Either enter a new key value pair or select an existing one. To add a new label, enter the Key and Value in the Quick Add section of the dialog. Then click Add. Locate the label in the list and highlight it. Click OK to complete the association. Repeat this procedure for all labels in your report.

*In this example, all 4 labels existed.

Notes on localizing report label values:

1. Property files are application specific, resource files. This means that all reports stored in the SLA application for example will use the same resource file. This enables common labels to be used by multiple reports.

2. The property files are located in <V7>\reports\birt\libraries



3. If one report design file is stored in multiple applications, it only needs to use one property file. You do not have to create individual property files for each application the report is accessed from.

   An example of this are the WO List and WO Details reports. These are available from the Work Order, Change, Release, Activity and Quick Reporting applications. However, the property file values for these reports are only created once in the wotrack.properties file.

Localizing report data

The data within a report can also be enabled for localization. This is enabled thru the runtime data translation.

Runtime Data Translation based on the user's language can be enabled by calling a method for every data set column to be translated.

The report query must include the Unique Id (UID) column(s) from the table(s) containing the translated field(s). It is not necessary to create output columns for these fields.

Call the following method at the end of the data set Open method. The arguments are case-insensitive:

registerDataTranslation(queryColumn, queryUIDColumn, mboName, mboAttributeName)
queryColumn - the field in the query to be translated
queryUIDColumn - the unique ID field in the query from the table containing the translated field
mboName - the maxattribute objectname for the translated attribute
mboAttributeName - the maxattribute attributename for the translated attribute

```
maximoDataSet = MXReportDataSetProvider.create(this.getDataSource().getName(),
this.getName());
maximoDataSet.open();

var sqlText = new String();
sqlText = "select itemnum, description, itemid from item"
maximoDataSet.setQuery(sqlText);

maximoDataSet.registerDataTranslation("description", "itemid", "ITEM", "DESCRIPTION");
```

Report Data Localization Notes:

1. An example of how the runtime data translation can be done is contained in the out of the box report, Job Plan List, jobplan.rptdesign. This is enabled on its description field.

2. Use the getBooleanString method to fetch and translate YORN fields.

3. Do not apply currency symbols. Instead, use the currency code field from Maximo.

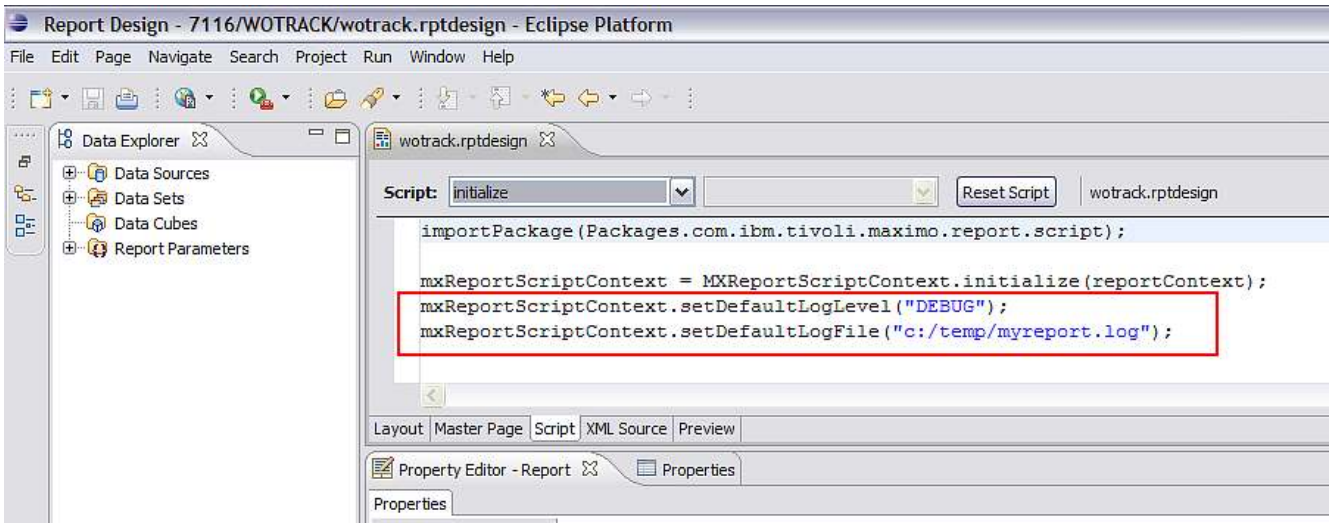
Debugging within the BIRT Report Design tool

You can log information about the report that you are developing within the BIRT Report Design Tool. This logging is used only when you preview a report within the Report Design Tool.

Reference: For more details on report logging, including how to enable it within the V7 application, reference the V7 Report Logging Guide noted at the end of this document.

To do this, access the Report initialize method, and add the following two lines of code:

```
mxReportScriptContext.setDefaultLogLevel("DEBUG");
mxReportScriptContext.setDefaultLogFile("c:/temp/myreport.log");
```



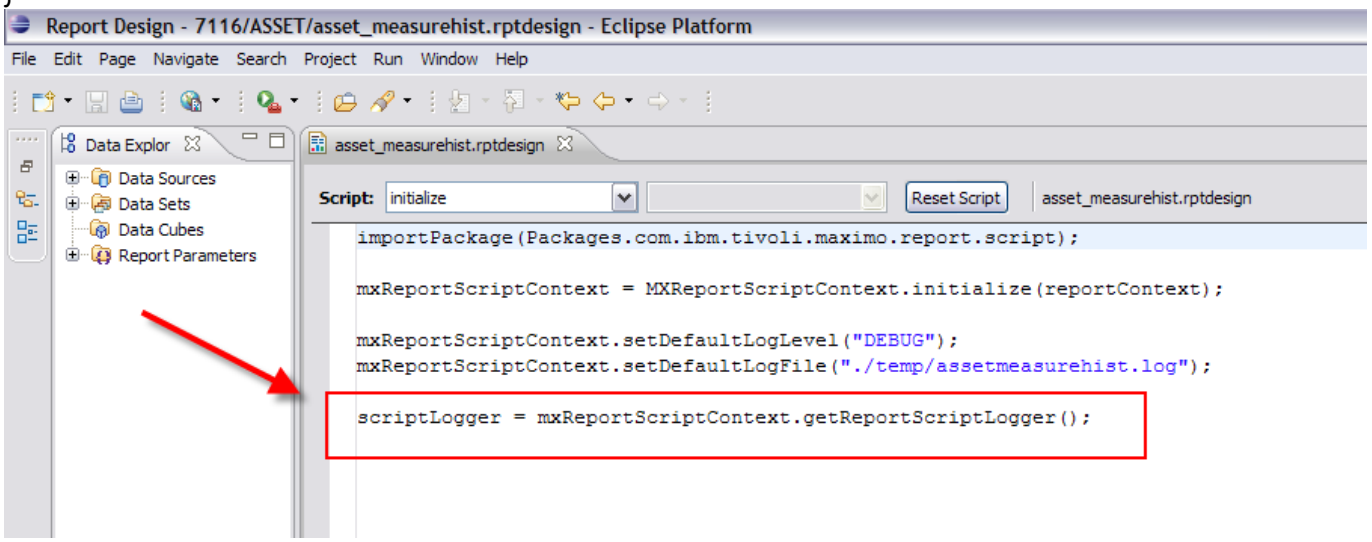
Five different log levels are supported, which are DEBUG, INFO, WARN, ERROR, FATAL. These levels are described in more detail below. Since this logging is primarily used for debugging report design issues, it is recommended that you use the DEBUG level.

Replace the file path location shown here as "c:/temp/myreport.log" with the file path for your individual environment.

Note: This logging is not used when executing a report from the Maximo applications. Once your report runs, you do not need to remove this logging.

Additionally, to log custom information, you can use the mxReportScriptContext variable to get a script logger, which can then be used throughout your report. You can add this to the report initialize method also as shown here.

```
scriptLogger = mxReportScriptContext.getReportScriptLogger();
if (scriptLogger.isDebugEnabled())
{
        scriptLogger.debug("***My Debug Message ****");
}
```



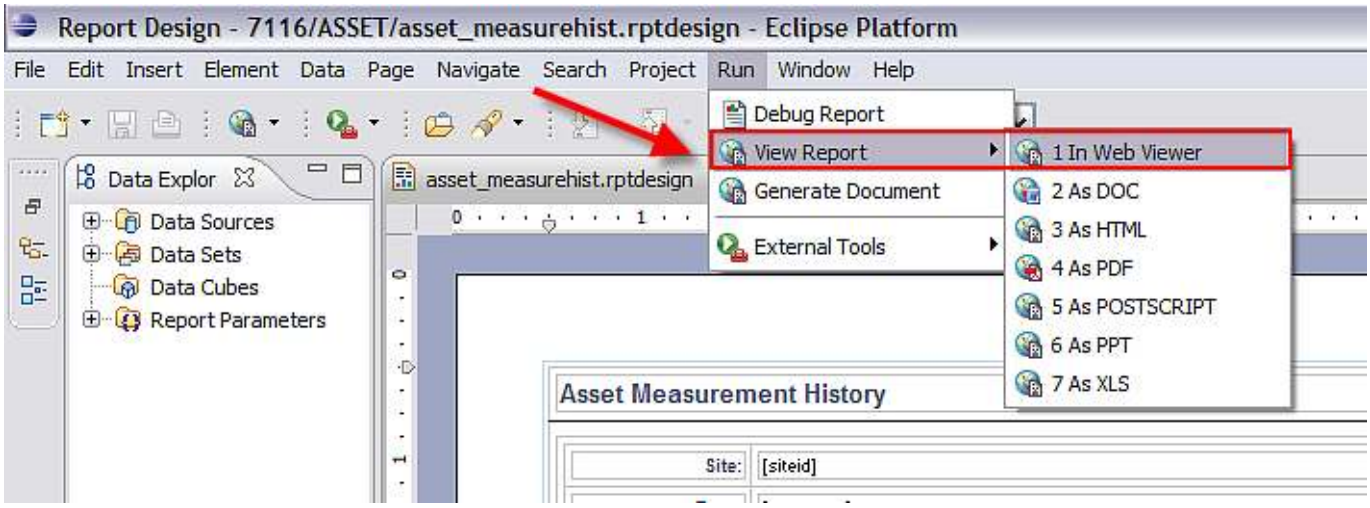
Unlike the default logging, these logging messages are written to the Maximo log files when the report is run from within Maximo. In this case, the default log level specified in the report is ignored. Instead, the maximo.report.birt log level from Maximo is used.

You can use any of the following methods below from ReportLogger to log information.

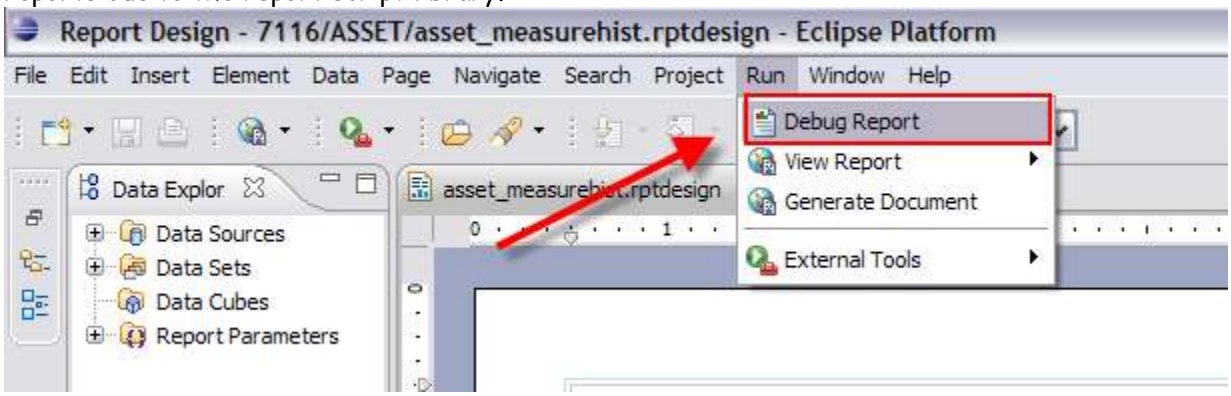
```
boolean isDebugEnabled();
boolean isErrorEnabled();
boolean isFatalEnabled();
boolean isInfoEnabled();
boolean isWarnEnabled();

void debug(Object message);
void info(Object message);
void warn(Object message);
void error(Object message);
void fatal(Object message);
```

Report Designer best practices for debugging

1. Preview reports by using the Web Viewer - View Report Section within the BIRT report designer. This displays the closest representation to report execution from within the various Maximo applications.



2. Within the BIRT 2.3.2 Report Designer, a 'Debug Report' Options is available. Do not use this functionality because it does not properly display information with the Maximo implementation of reports due to the report script library.



Note: More information on debugging within the V7 applications is noted in the Report Logging document referenced at the end of this document.

Miscellaneous Features

Database Update Functionality

The functionality below details how to add database update functionality to reports. Examples of reports that use this functionality include:

Asset Cost Rollup (asset_costrollup.rptdesign, located under ASSET subfolder)
Inventory ABC (inventory_abc.rptdesign, located under INVENTOR subfolder)

With this functionality, the reports should be able to execute Database SQL UPDATE/INSERT/DELETE statements against a specific data source:

Here is an example of how this can be used. (Note: all examples are illustrated with SQL UPDATE stament, but SQL INSERT and DELETE can be used similarly)

1. executing the update within a DataSet (any of the open/describe/fetch/close/beforeOpen/beforeClose/onFetch/afterOpen/afterClose events)

```
myTxn = MXReportTxnProvider.create(this.getDataSource().getName());
myStmt = myTxn.createStatement();
myStmt.setQuery("update ... set .... = ....");
myTxn.save();
```

2. executing the update outside of a DataSet

```
myTxn = MXReportTxnProvider.create("MAXIMODATASOURCE");
myStmt = myTxn.createStatement();
myStmt.setQuery("update ... set .... = ....");
myTxn.save();
```

3. executing multiple updates

```
myTxn = MXReportTxnProvider.create(this.getDataSource().getName());
myStmt1 = myTxn.createStatement();
myStmt1.setQuery("update ... set .... = ....");
myStmt2 = myTxn.createStatement();
myStmt2.setQuery("update ... set .... = ....");
myTxn.save();
```

4. executing with parameters.

```
myTxn = MXReportTxnProvider.create(this.getDataSource().getName());
myStmt = myTxn.createStatement();
```

```
myStmt.setQuery("update ... set .... = ?, ... = ?");
myStmt.setQueryParameterValue(1, new Integer(0)); // using Integer object as an example. Also note the parameter index starts form 1
myStmt.setQueryParameterValue(2, "MyValue");                    // using String object as an example
myTxn.save();
```

Registering a Report to Multiple Applications

Some reports can be accessed from multiple applications. Do not make copies of the design files; instead store the report in the primary application report folder, and register it to the other applications by including it in the reports.xml for each applicaiton.

1. Create the normal import entry in the home application's reports.xml.

2. Copy the entry to the reports.xml file for any other applicaiton that uses the report.

3. Change the <filename> entry to reflect the relative path to the actual report location, for example:

   <attribute name="filename">../PO/po_act.rptdesign</attribute>

4. Change the other report administration values as appropriate.

5. If there are bound parameters, you may need to modify them since bindings that work in one application may not work in another.

Registering a Report with Quick Toolbar Access

Reports that do not contain User inputted parameters can be enabled for Quick Toolbar Access. The types of quick toolbar access are described and shown in the chart below.

BV: (Browser View) Enables the user to click on the BV icon in the application's toolbar, and the report opens immediately in the Report Browser. 

DP: (Direct Print) Enables the user to click on the DP icon in the application's toolbar, and the report prints on the user's default printer. The report does not display in the report browser session. 

DPA: (Direct Print with Attach Documents) Enables the user to click on the DPA icon in the application's toolbar, and the report and any of its printable attachments, print on the user's default printer. The report does not display in the report browser session. 

	Description	Database Field	Toolbar Location	Sequence
BV	Browser View	REPORT.QL	REPORT.QLLOC	REPORT.TOOLBARSEQUENCE
DP	Direct Print	REPORT.DP	REPORT.DPLOC	REPORT.TOOLBARSEQUENCE
DPA	Direct Print with Attached Documents	REPORT.PAD	REPORT.PADLOC	REPORT.TOOLBARSEQUENCE

An example of a report that has these settings defined can be found in the WOTRACK folder

```
<report name="wotrack.rptdesign">
        <attribute name="filename">wotrack.rptdesign</attribute>
        <attribute name="description">Work Order List</attribute>
        <attribute name="qlloc">ALL</attribute>
        <attribute name="ql">1</attribute>
        <attribute name="toolbarsequence">1</attribute>
        <attribute name="attacheddoc">0</attribute>
        <attribute name="norequestpage">0</attribute>
        <attribute name="detail">0</attribute>
        <attribute name="reportfolder">WOTRACK</attribute>

        <resources>
```

Importing and exporting

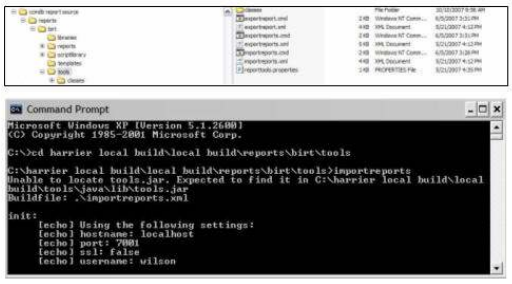
Importing of BIRT Libraries and Design Files

The repository for the report design files is the V7 database. Importing is the process which brings the report design files, and any of their dependant report files (like libraries or resource files) into the database.

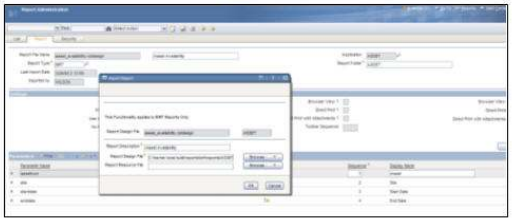
If a new report is created, or an existing report updated, the new or updated file must be brought into the database. This is done through the report import process. For new reports, importing will create a new report record. For existing report in which the report design file exists, importing will over-write the existing file.

Importing of the design files and any dependant report files (libraries, resource files) can be done in two ways - (1) Bulk - Thru Command Utilities or (2) Individually – Through an action in the Report Administration application. The table below describes the differences between the two.

Bulk Import Utilities



Individual UI Action



Type	Bulk Import	Individual Import
Description	To import multiple reports or libraries. Import options include all reports/libraries, libraries only, reports for a specified application.	To individually import a single report or library.
Access	Command Utility available from <V7>\reports\birt\tools	Action Available from Report Administration application.
Where Used	…when a number of new reports are created for a Cloned application. The administrator has access to the application server, and wants to import all the new reports for the Cloned application at one time.	…when an administrator needs to import a single report file or library. This may be a new report, or an update to an existing report. The admin may not have access to the application's console, so he can not utilize the command functions.

Details on Bulk Importing thru Command Utility

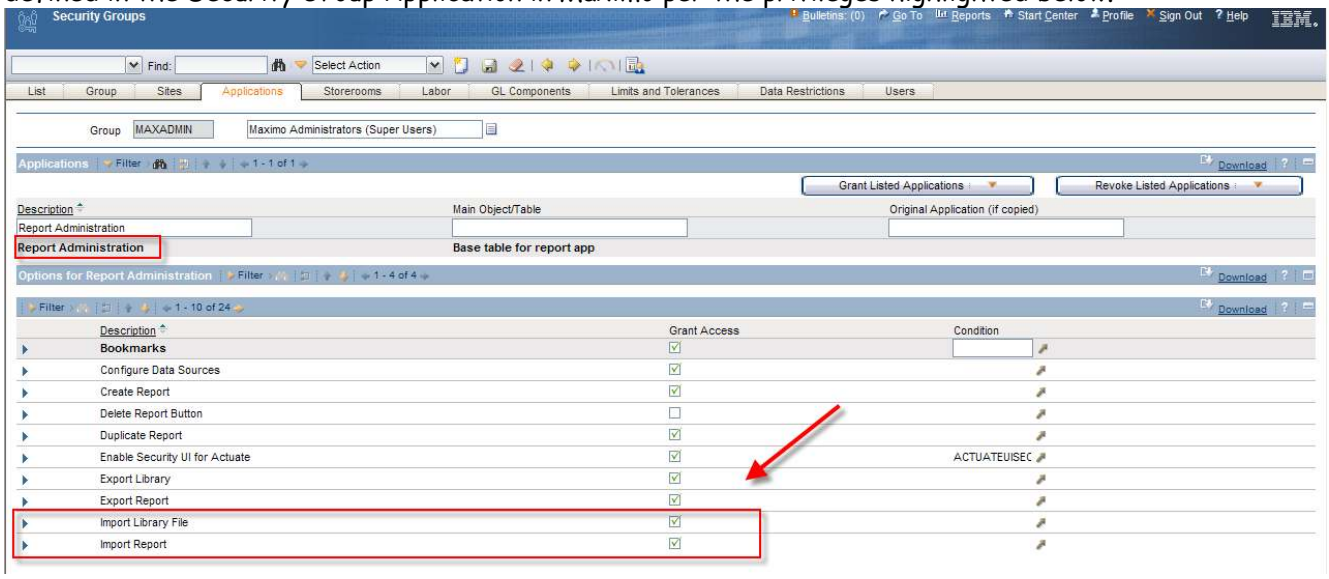
If you have a large number of reports to import, you may want to use the import command utility. This process uses the reports.xml file to import the report design files in the database. A reports.xml file is available for each application that has reports, and is located in the directory <V7>reports\birt\reports.

The reports.xml file references each report design for the individual application. If a report design is not referenced in the reports.xml file, it will not be imported during the command utility process. Details on updating the reports.xml file for any new report files you may create can be found at the end of this document.

To use the import utility, follow the steps below.

1. Browse to the tool location ...<V7> \reports\birt\tools. Locate the reporttools.properties file. This file indicates what server to connect to so the import operations can be performed. The username and password are for the user that has privileges to import reports.

The username and password are for the user that has privileges to import reports. This user is defined in the Security Group Application in Maximo per the privileges highlighted below.



2. Update and save the reporttools.properties** file for your applicable settings. The default settings are shown below:

```
 # HostName or IP address of the machine that has V7 running on an Application Server
maximo.report.birt.hostname=localhost

 # User that has access to perform the operation
maximo.report.birt.username=wilson

 # Password of the user that has access to perform the operation
maximo.report.birt.password=wilson

 # Output folder used for the export operation
maximo.report.birt.outputfolder=./../../birt

 # HTTP port of the application server (the port used to access V7 from browser)
#  Standard values are 7001 for Oracle WebLogic or 9080 for IBM WebSphere®
maximo.report.birt.port=7001

 # Indicates whether the SSL communication is enabled or not
maximo.report.birt.ssl=false
```

3. On the V7 server, open a command prompt window and change to the folder <V7> \reports\birt\tools. Run the importreports.cmd command.

This utility will import all reports, libraries and resource files in the single import action. If you do not want to import all reports and their dependant files (resource and library files), additional variations of the import command are available, including

Updates to reporttools.properties file – 7115+ Release

In the Maximo Base Services 7.1.1.5 release, the reporttools.properties file was updated to include the new property values highlighted below. These values are used in the 'Update Reports Utility' described towards the end of this document.

# Output folder used for the update operation
maximo.report.birt.update.outputfolder=./../../birt

The setting enables you to change the location of the backup files that are exported prior to the update.

# Indicates whether the report updates should be saved to the database
maximo.report.birt.savechanges=false

Changing this value to true defaults the savechanges behavior to true. This may be useful to you so the argument value is displayed in the console.

# Output folder and file name of the log file for updates
maximo.report.birt.updatelog=./update.log

This setting enables you to specify the location and/or name of the log file.

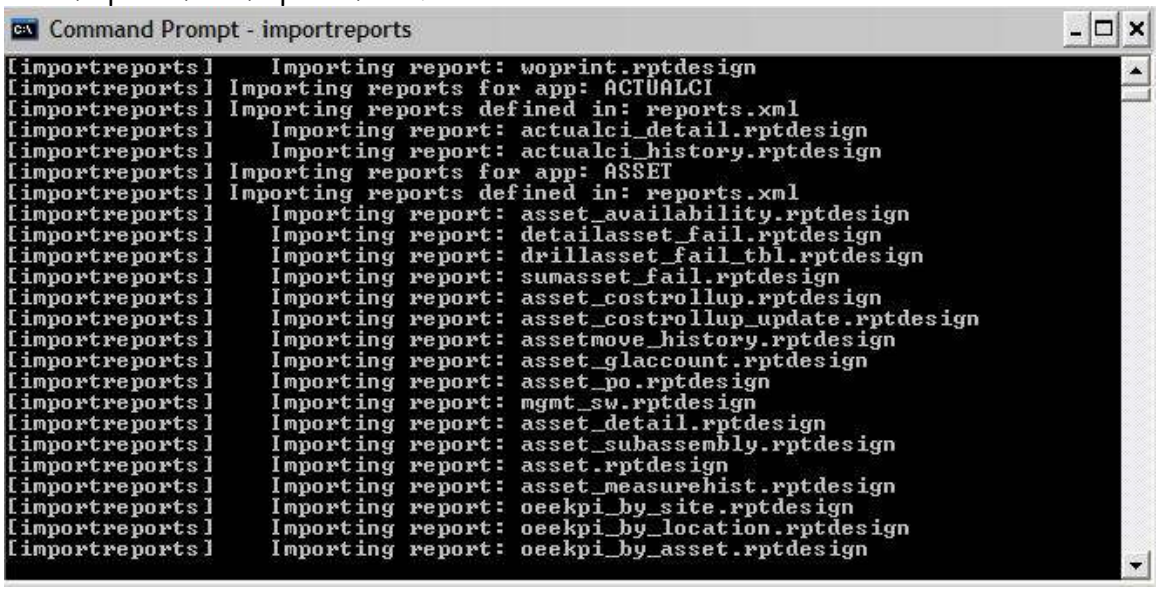
Import Commands

A. importreports help
Displays details on the various import commands

B. importreports libraries
Imports all the libraries

C. importreports reports
Imports all the reports

D. importreports app [appname]
Imports all reports for a specified application. This command is useful to import reports for a single application. For example: importreports app CONFIGUR will import all the reports in the CONFIGURATION application. These are the reports found in the directory below:
<V7> \reports\birt\reports\CONFIGUR

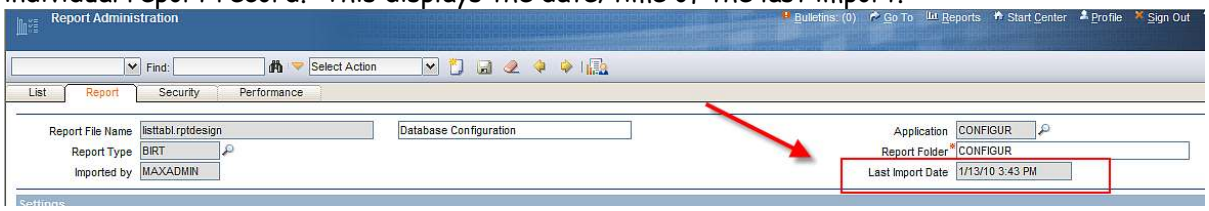


After completing, the updated or new files will be located in the REPORTDESIGN table. The REPORTDESIGN Table holds the design files, resource files and library files.

Library files are identified as REPORTDESIGN.ISLIBRARY = '1'

4. Next, sign into your Application as the administrator. Go to Report Administration, and generate the XML for the reports. This will completes the process for importing reports.

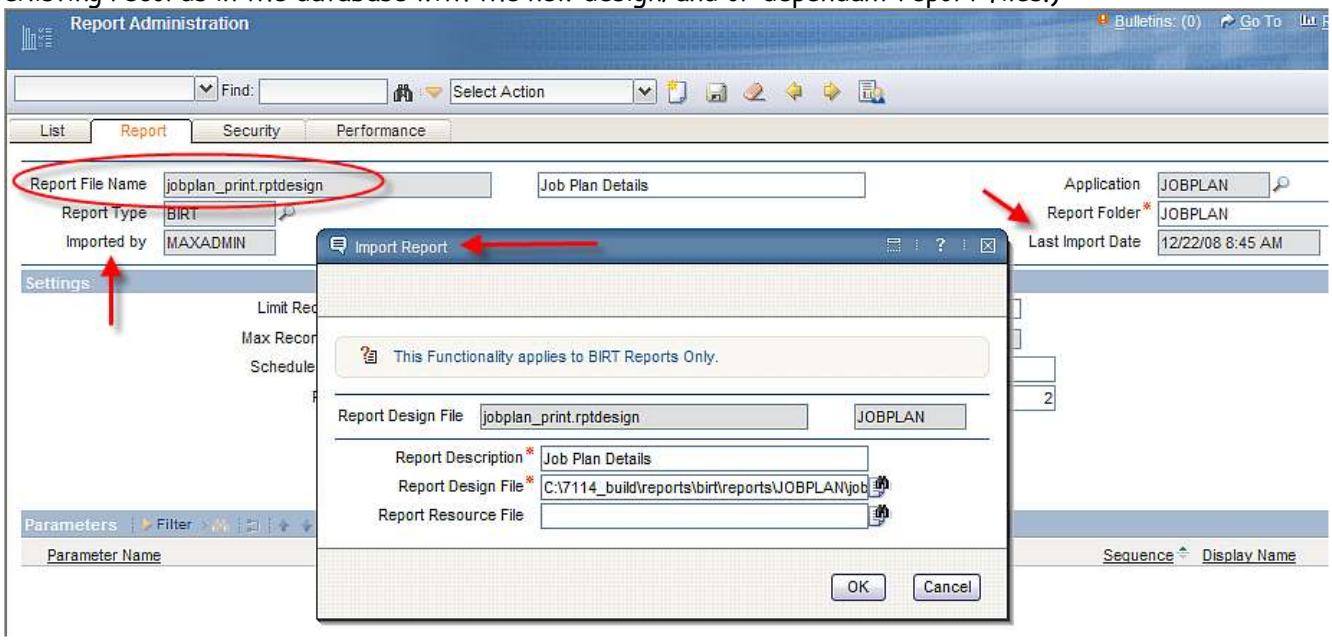
You can also see that the import occurred by looking at the 'Last Import Date' field for an individual report record. This displays the date/time of the last import.



Details on Individual Report Importing thru Report Admin

If you have modified only a single report, or are registering a small number of reports, you may want to use the Report Administration application to import these report designs. To do this, first sign in as an administrator and access the Report Administration application. Then, locate the individual design file, and from the action menu, select 'Import Report'.

Browse to the location of the design file, and the report resource file if required. Follow the prompts in the windows, and once complete, the updated design file will be imported in the database. Once the import is complete, the circled fields of 'Imported By' and 'Last Import Date' will be updated with the latest values. (Again – note that the import process will overwrite the existing records in the database with the new design/and or dependant report files.)



Notes:

1. The screenshot above is from the 7114 release where a new icon is utilized for the Browse functionality. If you are using a release earlier than this, your screen may look slightly different.

2. Do not copy/paste the location of any of the files to be imported. (Report Design File, Report Resource File or Library File.) You must use the browse button to properly import the correct file.

3. Resource files are imported as .zip files. These are typically not updated often, so if no changes have been made to the .gif/.jpg images for example – leave the Report Resource field blank in the dialog above.

Follow the same process above to import a library file via the 'Import Library' Action.

Notes:

1. This functionality applies to BIRT Reports only.

2. To use the individual report import in the Report Administration application, the report record must already exist. Therefore, this action is only available from the Report Tab.
    a. If you are importing a new design file, you must first create its record in Report Administration, and then import the file.

3. There is no validation on the report design's file during the import process. This means that if there is an error in the report's design file, it will still import. It is up to the developer/admin to insure that the design file is correct.
    a. The UI import process will only validate that the design file that is being imported that matches the current record.

4. Libraries must be imported before report designs. If the report design references a library that has not been imported, the report design will not import.
    a. However, if the report references an existing library (MaximoSystemLibrary.rptlibrary) that is already in the database, you do not need to import another copy of that library.

Export Reports Utility

Clients may also need to export report design files. There are many uses cases for exporting, including the need to export an Ad Hoc or Query Based (QBR) report. For example, a user asks that an Ad Hoc report he created be extended to include a bar chart to the report. Therefore, to save the Developer time from recreating this report from the beginning, he simply will export the Ad Hoc report. He can then open it up in the BIRT Designer, add the bar chart, and then import the design file back into the database as an enterprise report.

Note: Exporting of reports is only enabled as a utility.

To enable exporting, go to the server, open a command prompt window and change to the folder <V7> \reports\birt\tools. Then, any of the commands below are available:

1. exportreports
   Exports all libraries and reports.

2. exportreports report
   Exports all reports.

3. exportreports library
   Exports all libraries.

4. exportreports app [appname]
   Exports all reports for the specified application.

5. exportreport report [appname] [reportfilename]
   Exports single, specified report for the specified application.

The report will be exported to the location defined by
(1) the reporttools.properties file and
(2) its report folder that it is registered to in the Report Administration application.

The example below will use wotrack.rptdesign to show the exporting functionality.

The reporttools.properties file has been set to use the output location shown below in red:
maximo.report.birt.outputfolder= c:/version7/reports/birt/reports

Additionally, the wotrack.rptdesign is located in the following applications and report folders:

Report Name	App Name	Report Folder	Description
woprint.rptdesign	QUICKREP	WOTRACK	Quick Reporting
woprint.rptdesign	CHANGE	WOTRACK	Change
woprint.rptdesign	RELEASE	WOTRACK	Release
woprint.rptdesign	ACTIVITY	WOTRACK	Activity

woprint.rptdesign	WOTRACK	WOTRACK	Work Order

Export Commands

When the various exports commands are executed, the following will occur:

1. exportreports
Exports all reports to c:/version7/reports/birt/reports and their various subfolders

AND Exports all libraries to c:/version7/reports/birt/libraries

2. exportreports report
Exports all reports to c:/version7/reports/birt/reports and their various subfolders

3. exportreports library
Export all libraries to c:/version7/reports/birt/libraries

4. exportreports app WOTRACK
Exports all reports registered to WOTRACK to c:/version7/reports/birt/reports/WOTRACK

5. exportreport report WOTRACK woprint.rptdesign
Will export woprint.rptdesign to c:/version7/reports/birt/reports/WOTRACK

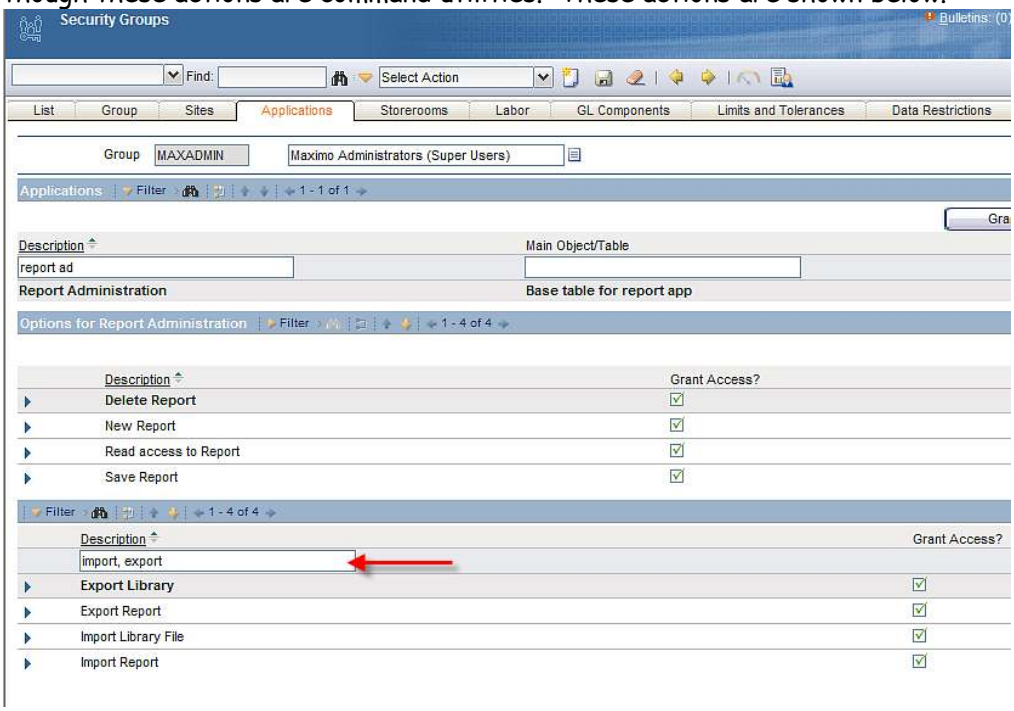
Additional Export Details

A. If a report structure is not available in the location where the export is to occur, a file structure will be created.

B. If a reports.xml is not available in the location where the export is to occur, the reports.xml will be created.
- This may occur if you create a new custom report design file, and register and import the report thru the Report Administration application.
- If you do this and you make subsequent changes to the parameters or settings of the report in the Report Administration application, make sure to export the report design file so any changes you are made are captured in the new reports.xml file.

C. If a reports.xml file does exist - the export will not overwrite the existing file.
- In this case, a new one will be created using a –filename. Ex: In WOTRACK folder, if reports.xml exists and a new export occurs, a new reports-wotrack.xml file will be created. This new reports-wotrack.xml will take precedence over the reports.xml file during any future importing actions.

D. In order to perform the import and export functionality, the security group needs to be granted access to these actions via the Security Group application. This is required even though these actions are command utilities. These actions are shown below.



E. Both the import and export command utility tools use HTTP, not RMI, to support application server security.

Preparing the reports.xml import file

One of the report developer's responsibilities is to create the reports.xml file. This file is required so the report design can be properly imported into the V7 database repository.

As noted in the report design file structure section above, each application folder under reports\birt\reports should have an import file named reports.xml.

The reports.xml includes important information on the report, including its report design file, application, unique settings for the report (including direct print, sequencing or priority values) and parameter information.

If the reports.xml contains parameter values with appropriate attributes, then the import tool inserts or updates the report parameter table (reportlookup) with the information. The attribute names of the parameters are the columns of the reportlookup table. If the parameter name defined in the reports.xml for a given report does not exist in the report, it is ignored. Some of the attribute values are defaulted to what is in the report if they are not specified in the import file.

When specifying a greater than or less than symbol for a parameter operator, you must escape the symbols as follows:

	Symbol	Description
&lt;	<	Less than
&gt;	>	Greater than
&amp;	&	Ampersand
&apos;	'	Apostrophe
&quot;	"	Quotation mark

Example of reports.xml Import file

This section gives an example of creating the reports.xml file used for report importing and exporting. Creation or modification of the reports.xml file can be done in any text editor like word pad or notepad.

The following steps are suggested for updating or creating a reports.xml file:

1. Locate an existing Out of the box report that has parameter values similar to your new report.

    For an example of a report using a parameter, reference the Security Group Report (security_group.rptdesign) located in <V7> \reports\birt\reports\SECURITY

    For an example of reports using the Current/Selected/All Record set and having various toolbar settings enabled, reference the Job Plan List and Detail reports located in <V7> \reports\birt\reports\JOBPLAN

2. If you are unsure what Out of the box have which parameter values, access the V7 Report Booklet referenced at the end of this guide. This contains details on each report's parameters, along with various toolbar settings that are enabled for each report.

3. Once you locate a similar report, copy and paste the values from the out of the box report, and update them for your custom report.

The information below is intended to detail each of the fields that can be used in defining a report in the reports.xml file. Remember each setting is not required, and those settings not required are noted in black text.

RED = Required Fields that must be used in defining any report
BLACK = Optional Fields.
BLUE = Text defining field value

The example uses a sample report.

```
<reports>
        <report name="jobplan_test.rptdesign">
        #Complete File Name of Report Design, including .rptdesign extension
                <attribute name="filename">jobplan_test.rptdesign</attribute>
                #File Name of Report Design, including .rptdesign extension

                <attribute name="description">Job Plan Test</attribute>
                #Description of Report Design which appears in 'Run Reports' Window

                <attribute name="ql">0</attribute>
                #Is Browser View enabled for Report? Can only be enabled if report does not have
                parameters. 0=No/1=Yes. Default is 0

                <attribute name="dp">0</attribute>
                #Is Direct Print enabled for Report? Can only be enabled if report does not have
                parameters. 0=No/1=Yes. Default is 0

                <attribute name=" pad">0</attribute>
                #Is Direct Print with Attachments enabled for Report? Can only be enabled if report does
                not have parameters. 0=No/1=Yes. Default is 0

                <attribute name="toolbarsequence">1</attribute>
                #Order of the report in relation to other reports enabled for toolbar access within the
                application. Value must be unique within a given application.

                <attribute name="qlloc">NONE</attribute>
                #Determines what tabs will display BV icon. Options are:
                #ALL: Displays Report Icon on all toolbars in the app
                #LIST: Only Displays Report Icon on List Tab of app
                #MAIN: Displays Report Icon on all toolbars in app, except List tab
                #NONE: Default Value. Does not display Report Icon in app.

                <attribute name="dploc">NONE</attribute>
                #Determines what tabs will display DP icon. Options are:
                #ALL: Displays Report Icon on all toolbars in the app
                #LIST: Only Displays Report Icon on List Tab of app
                #MAIN: Displays Report Icon on all toolbars in app, except List tab
                #NONE: Default Value. Does not display Report Icon in app.

                <attribute name="padloc">NONE</attribute>
```

#Determines what tabs will display DPA icon. Options are #ALL: Displays Report Icon on all toolbars in the app
#LIST: Only Displays Report Icon on List Tab of app
#MAIN: Displays Report Icon on all toolbars in app, except List tab
#NONE: Default Value. Does not display Report Icon in app.

<attribute name="norequestpage">0</attribute>
#Does the report not require a request page? 0=No/1=Yes. Default is 0 - Report does require a request page. Used only for reports which update database or are only available via hyperlinks.

<attribute name="detail">0</attribute>
#Are limit records enabled for this report? Can only be enabled if report does not have parameters. 0=No/1=Yes. Default is 0.

<attribute name="recordlimit">50</attribute>
#If limit records are enabled (detail = 1), this field must be defined. It is the maximum # of records the report can execute against. Value must be > 0.

<attribute name="priority">2</attribute>
#Priority value of report used for Report Queuing. Priority is based on ascending order - the lower the #, the higher the priority.

<attribute name="usewherewithparam">0</attribute>
#Will the report execute against both current/selected records and user inputted parameters? 0=No/1=Yes. Default is 0. Can only be enabled if report has parameters.

<attribute name="reportfolder">JOBPLAN</attribute>
#Location of report source file subfolder in <Version7>\reports\birt\reports

<parameters>
<parameter name="jpnum">
#Name of parameter. If the parameter is unbound, this text must exactly match the unbound parameter defined in the BIRT Designer (.rptdesign file)

<attribute name="attributename">JPNUM</attribute>
Either the attribute name from the main table of the app, or the attribute from one of the app's Maxrelationships. If this field is populated, the parameter is bound. If the field is not populated, the parameter is unbound.

<attribute name="lookupname"></attribute>
#Name of lookup. Depending on availability, a bound parameter may or may not have a lookup. Unbound parameters can only have lookups for date fields.

<attribute name="sequence">1</attribute>
#Order the parameter is displayed on the request page.

<attribute name="labeloverride">Job Plan</attribute>
#Parameter label text that displays on Request Page.

```
                <attribute name="defaultvalue"></attribute>
                #Default value displayed in parameter field on request pages.  Default values are
                not localized.

                <attribute name="required">0</attribute>
                #Is the parameter required?  0=No/1=Yes. Default is 0.

                <attribute name="operator"></attribute>
                #Optional operators that can be applied to bound parameters.  Values available are
                >, >=, <, <=.  These can not be applied to unbound parameters.

                <attribute name="multilookup">0</attribute>
                #Can multiple values be input for a parameter? 0=No/1=Yes. Default is 0.
                </parameter>
        </parameters>



        <resources>
                <resource>
                        <reference>jobplan.properties</reference>
                        # The property file used by this report
    <filename>${libraryfolder}/jobplan.properties</filename>
                        #The location of the property file.  ${libraryfolder} refers to
                        <Version7>\reports\birt\libraries
                </resource>
        </resources>
    </report>

</reports>
```

Miscellaneous Utilities

Update Reports Utility

Beginning with the Base Services 7.1.1.7 Release (June 2010) two additional report utilities were made available.

These update utilities can be used to automate the process of applying updates to report designs, rather than manually editing each report. These are known as update utilities, and supplement the existing utilities of importing and exporting report designs.

The update utilities are available for both Enterprise Reports, and Ad Hoc or QBR Reports.

For enterprise reports, the four update utilities available are:

1. updatereports
Updates all reports.

2. updatereports savechanges
Updates reports and saves the modified reports to the database.

3. updatereports app [appname]
Updates all reports for the specified app.

4. updatereports app [appname] savechanges
Updates all reports for the specified app and saves the modified reports to the database

For Ad Hoc reports, the update utility available is:
1. updateqbrs
Updates all QBR report designs

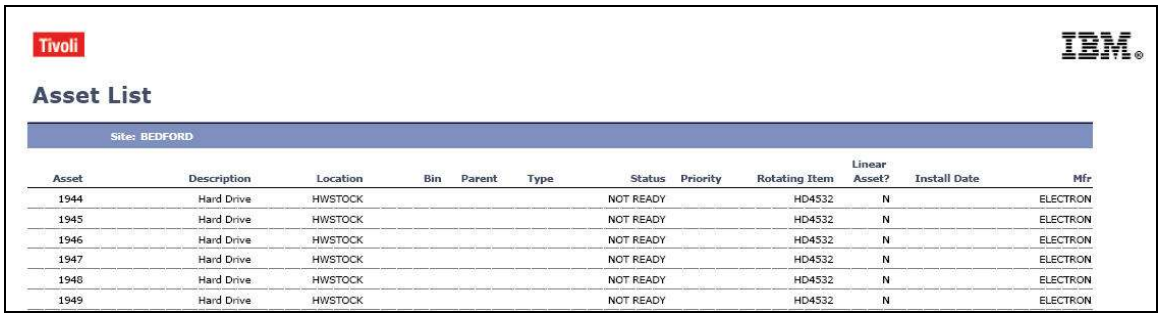
Specific details on how you can use these utilities can be found in the Update Reports Utility document located on IBM's support website. Information on locating this can be found in the Reference Materials section at the end of this document.

Customizing Reports Reference Materials

A number of different reporting reference materials are available to you. This section will highlight a few of those reference materials which specifically focus on customizing reports, and details how you might best be able to use them.

Changing Report Logos

V7 BIRT Reports contain two logos. A Tivoli logo is located on the upper left hand side, and an IBM Logo on the upper right hand side. You may want to update the V7 Reports to use their own corporate logos.



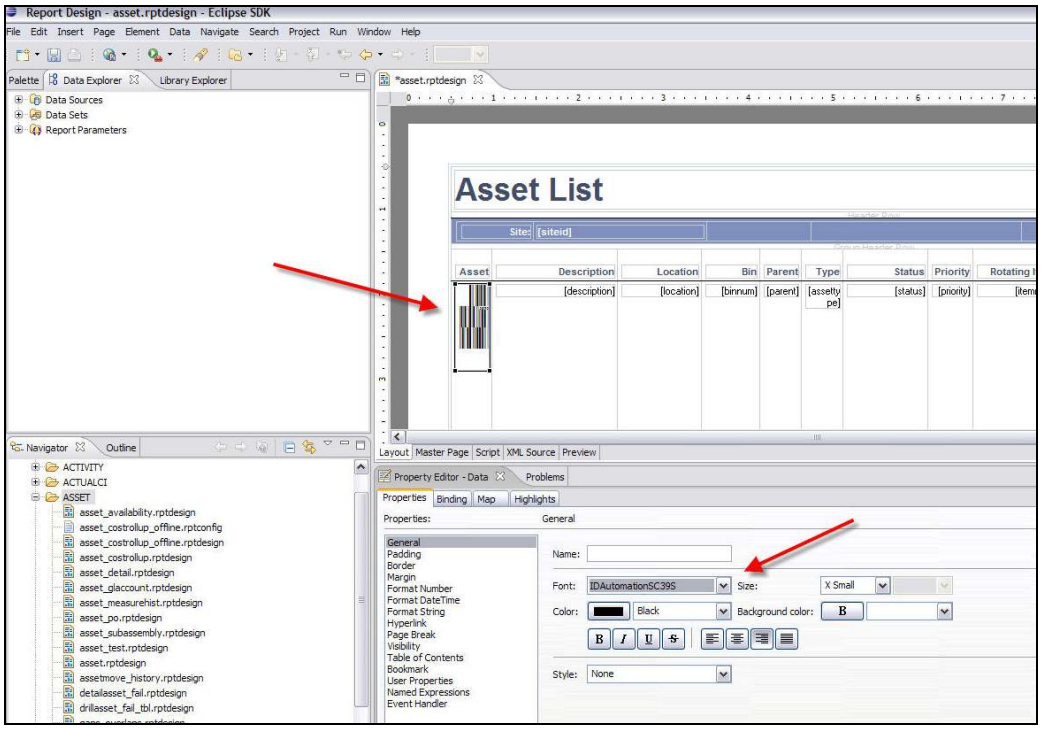
Asset	Description	Location	Bin	Parent	Type	Status	Priority	Rotating Item	Linear Asset?	Install Date	Mfr
1944	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1945	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1946	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1947	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1948	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON
1949	Hard Drive	HWSTOCK				NOT READY		HD4532	N		ELECTRON

Implementing Bar Code Fonts

You may need to implement bar code fonts in your V7 reports. This document details the steps required to do this, including:

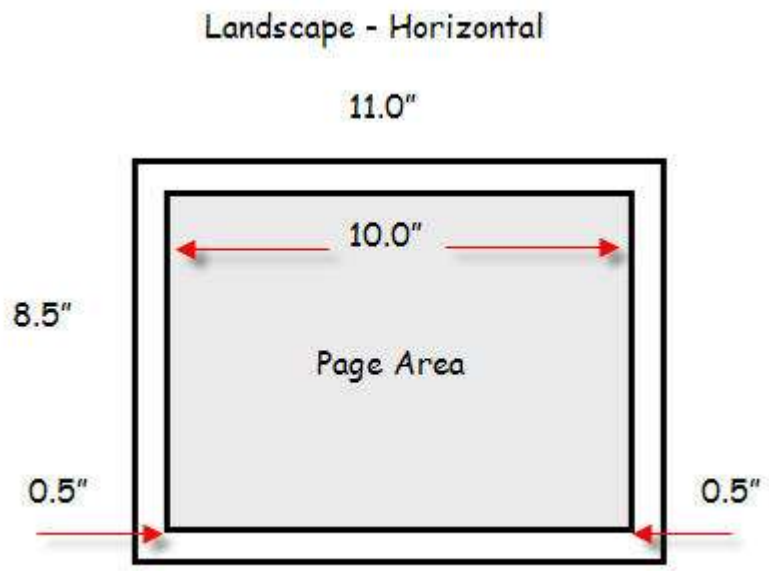
1. Enabling Bar Code fonts on the Client Machine where the BIRT Designer is installed

2. Enabling Bar Code fonts on the V7 Server where the BIRT Report will be executed.



Understanding Report Paper Sizes

This document reviews the components impacting report page sizes and orientation used in the V7 BIRT Reports. It also details how you can customize them to meet your individual business needs.



Modifying OOB Reports

The out-of-the box reports that are supplied to you may not meet your individual business needs. You may need to add or remove fields to these reports to reflect your unique environment.

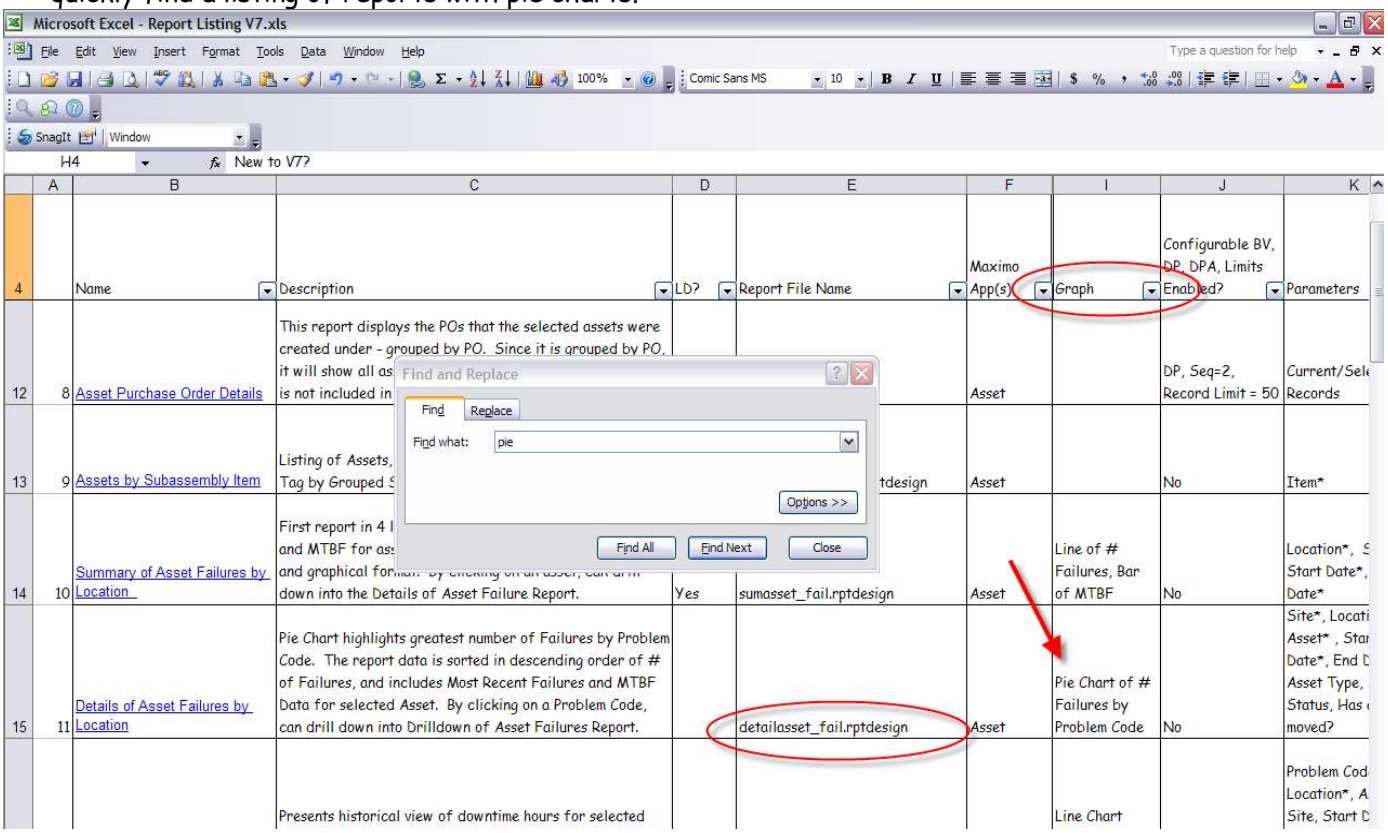
This document details how you can modify the out of the box. It uses the Work Order Details Report as an example. Three examples of modifications are detailed, including

A. Deleting Fields from the Planned Labor Section
B. Deleting Fields from the Actual Labor Section
C. Adding Fields to the Actual Material Section

Utilizing the V7 Report Booklet

The V7 Report booklet details all the reports delivered in the base Maximo Services release. Within the booklet, is an .xls file which lists each report, and other important information on the application it is registered to, if it has any parameters, along with information on its graphs, sorting, grouping and what templates it is used.

If you are required to create a custom report, you may want to review the V7 Report Booklet to quickly find out-of-the-box (OOB) reports which have similar functionality. For example, if you wanted to review the code of a report that contains a pie chart, search within the booklet to quickly find a listing of reports with pie charts.



Additionally, some of the OOB contain Long Descriptions (LD) which give additional information as to the context. These are indentified with a 'Yes' in the LD column of the booklet.

	A	B	C	D	E	F	G	H
4		Name	Description	LD?	Report File Name	Maximo App(s)	Report Folder (Design File, XML File)	New to V7?
9	5	Asset List	Unique report, which displays base asset information, but also contains 'hidden' asset fields. These 'hidden' fields are only available to end user when downloading report to excel. This enables additional fields in report for analysis while maintaining read-ability of report in browser and pdf.	Yes	asset.rptdesign	Asset	ASSET	Yes
10	6	Asset Measurement History	Using Yellow and Red Control Limit Values, displays Meter Readings by Date in Line Chart Format. A page break separates the details, which are displayed in ascending order of Measurement Date. This enables review of the most recent measurements first. Can be executed against single or multiple assets.	Yes	asset_measurehist.rptdesign	Asset	ASSET	No

And then to view this information, access the 'Additional V7 Report Desc' worksheet, where the long description details of the report can be viewed.

V7 Reports | Additional V7 Report Desc | Revisions

	A	B	C
1	#	Report Name	Additional Help Text Description
11			
12	6	*Asset Measurement History Report*	The 7.1.1.4 Report, uses the new YORN parameter lookup, yornlookuplist. The information on this can be found in its Report Admin Entry or in the the reports.xml (<v7114>\reports\birt\reports\ASSET)
13			

Finally, you can see the latest revision information on the out-of-the-box reports by reviewing the 'Revisions' workbook page.

	A	B	C
1		Revision	Description
2	Jun-09	2	Maintenance Cost Rollup. Added recommendations on applying the 'Schedule Only' Functionality to this report. More details in the 'Additional V7 Report Desc' Workpage.
3		2	Job Plan Hierarchy Report. This report is the more complex than the WO and Location Hierarchy reports as there is no database table that stores the relationships between job plans. Multiple queries have to be performed within the report to get this data.
4			Therefore, the following performance improvements have been made (1) adding index on JOBTASK Table, JOBPLANID Column and (2) Set a maximum limit of hierarchy levels displayed to be 6 to optimize the report for immediate viewing. Additionally, added a hyperlink on 6th level to drill down to further levels if required.
5			Note: If you want additional hierarchy levels to display, you may want to consider removing the hiearchy limit - and implement 'Schedule Only' functionlaity for this individual report. The 'Schedule Only' Functionality will prevent users from running the report immediately.
6		2	Linear Work History. Updated PDF by removing graph.
7		2	Database Configuration Report. Updated to include page break after each database table (object) in Base Services 7.1.1.5.
8		2	Details of Asset Failures. Included Additional Report Description on parameters. This report contains a date/time lookup because its date parameters are bound (Bound parametes have attribute values populated - they are not null). If a date parameter is not bound, its lookup can only be date enabled.
9		2	Summary of Asset Failures. Added more details in its descriptions stating this is an example of passing parameters silently from one report to the next.

Additional References

The following lists additional references available at the time this guide was prepared. The best way to locate these documents is to perform a search on IBM's Support Site on the report title highlighted below, or on its support reference number directly next to the title.

IBM's support site can be found at this url:
http://www-947.ibm.com/support/entry/portal/Overview/Software/Tivoli/Maximo_Asset_Management

Additionally, the wiki site below is available with additional details, including common client customization requests.

https://www.ibm.com/developerworks/wikis/display/maximo/Home

Title	Reference Number	Description
V7 Report Feature Guide	1305020	Details how the embedded report engine is utilized, including a review of the file structure, installation and database structure. Includes information on Security, Scheduling, Administration, Queuing, and Property Files.
V7 Report Booklet	1305005	Contains listings, file names, descriptions, details on parameters, formatting (grouping, sorting) and a pdf copy of each of the OOB (Out of the Box) Delivered Reports.
V7 QBR Ad Hoc Reporting	1417417 (7.1.1.6+) 136800 (Pre 7.1.1.6)	Details how users can Create and Execute Ad Hoc Reports, and the Administrative setup work involved in enabling Ad Hoc Reports, including security features and creating Report Object Structures.
Designing V7 Reports	1305009	Clients often require custom reports to communicate their individual business needs. This reviews the data analysis options available, including KPIs, Application List Downloads, QBR and Reports. Details report templates available, parameter options and a variety of items to consider when designing reports
V7 Report Planning Guide	1421371	Reviews each of the data analysis options available in V7, including a detailed review of each report option.
V7 Report Performance	1305031	Details various administration and configuration recommendations to optimize report performance. This includes BIRT Report Only Server (BROS) Configurations, along with Clustering.
Enabling secondary Database Configuration for BIRT reports	1304936	Describes additional ways of configuring your database for reporting, including enabling (1) All reports to execute against a reporting database or (2) A portion of the reports to execute against the reporting database.

Report Developer's Guides		
Report Development Configuration and Download	1390372 (7.1.1.5+) 1315837 (Pre 7.1.1.5)	Contains the designer download link, along with additional details on installing and configuring it.
Report Development Guide	1447958	Intended for the Report Developer, contains detailed information and examples on developing reports within Design Tool, database access, parameters, common development techniques like hyperlinks and date formats and various debug features and utilties.
Customizing out of the box BIRT reports	1438532	Details how to customize out of the box BIRT reports by deleting and adding new fields. The Work Order Details report is used as an example to step you through this process
Report Logging	1423974	Explains the report logging features available to report developers and administrators. Includes features available to report developers within the BIRT report designer, and then reviews the features available to report developers and administrators from within the V7 applications.
V7 Report Update Utility	1433106	Details report update utilities which can be used to automate the process of applying updates to report designs, rather than manually editing each report.
V7 Report Toolbar Access Direct Print and Related Information	1370440	Describes functionality that can be configured with BIRT Reports to enable their quick display, including Browser View, Direct Print and Direct Print with Attachments. Details report requirements, property files and troubleshooting techniques.
Enabling barcodes in BIRT Reports	1304925	Details how to implement bar code fonts for use within BIRT reports.
Changing logos in BIRT reports	1304923	Discusses how you can change the logos displayed within the V7 reports to your unique company logos.
Adding username to a V7 BIRT report	1403958	Provides instructions on adding the username who executed report to the report's header section
V7 BIRT Page Information	1317577	This document reviews the components impacting report page sizes and orientation used in the V7 BIRT Reports. It also details how you can customize them to meet your individual business needs.
Misc Guides		
Upgrading to BIRT 232 in V7	1390274	Discusses the enhancements enabled in BIRT 2.3.2, along with report considerations when upgrading your custom reports from BIRT 2.1.2 to 2.3.2.
System Administration Guide		Reference this guide for additional information on Localization, using the TDT and XLIFF files, along with Advanced Server Configuration details in the System Configuration Chapter.

Report Integration Guides		
Maximo Cognos Integration Installation Guide	1421365	Provides steps for enabling the report integration, including enabling the creation of the meta data layer.
Maximo Cognos Integration Guide	1421312	Details on how the report integration works between the two systems, including publishing report object structures as Cognos packages, security group synchronization and report file creation.
Business Objects/Crystal Report Integration Guide	1303812	Contains information on the Business Objects Enterprise XI ®Release 2, Service Pack 2 Release reporting integration. Details its (1) Licensing Requirements (2) Integration and (3) Development of reports.
ERI (External Report Integration) Guide	1304916	The ERI enables clients to integrate essentially any reporting tool with V7. This integration is similar to the Business Objects/Crystal Integration, except it is report system and report version independent. This document includes the ERI (1) Requirements (2) Installation (3) Administration and (4) FAQs.

Revision History:

Revision 7 – January 2012
(1) Updated report.xml example starting on page 85 (2) Added get(date) data set mapping on page 24

Revision 6 – April 2011
Corrected display of chart on page 74

Revision 5 – February 2011
Added information on creating custom report lookups

Revision 4 – January 26, 2011
Updates to include information/examples on closing data set under report development considerations section

Revision 3 – January 2011
Updates include (1) Added section on 'requirements for using lookups with parameters' on page 43 (2) More details on importing thru the Report Administration application on page 67 (3) Added hyperlink note section on page 31-32 (4) Updated reference materials section

Revision 2 – November 2010
Updates include (1) Additional details in hyperlink section (2) Note on javadocs in Integrated Service Management Library on page 17

Revision 1 – October 2010
Updates include (1) Additional information in sql section on page 22

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims

related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both