BIRT Reporting Guideline

Introduction

Reporting is an essential usage of the IBM CLM Solution for IoT. There are a couple of reporting capabilities available to fulfill the reporting requirements. This guideline is about the built-in BIRT solution, which allows running a report right in the context of the respective application. The usage of BIRT is possible in Rational Team Concert (RTC) and Rational Quality Manager (RQM). Due to the fact BIRT reporting is officially only supported for RTC this guideline just covers the report development with RTC.

The reporting in BIRT is done using the Jazz Data Source which includes various views on either RTC Live database tables or RTC Data Warehouse database tables. (see also Ref. 1).

Due to the fact the BIRT reporting is embedded into the RTC application it is important to take care during the design and deployment process of new reports. Once a report has been deployed to an RTC server and executed this report could potentially harm the server's stability if the amount of data fetched during the report execution increases dramatically. This could lead into a hanging RTC server process which in most of the cases can only be fixed by restarting the RTC server instance. To take care of the RTC application this guideline will provide some basic knowledge to prevent poor report design and best practices for BIRT report designers.

Retrieve Data

In general, a report developer should consider to fetch only those columns from the database which are required for a certain report because the data is processed in a Data Set which loads the entire data into memory. Therefore, each column and each row fetched from the database consumes memory on the application server.

In order to fetch data from the live or data warehouse database there exists three kinds of data sets:

- Advanced Data Set
- Simple Data Set
- Parameter Data Set.

This section is about the Advanced and the Simple Data Set. The Parameter Data Set is similar to the Simple Data Set and isn't explicitly described in this guideline.

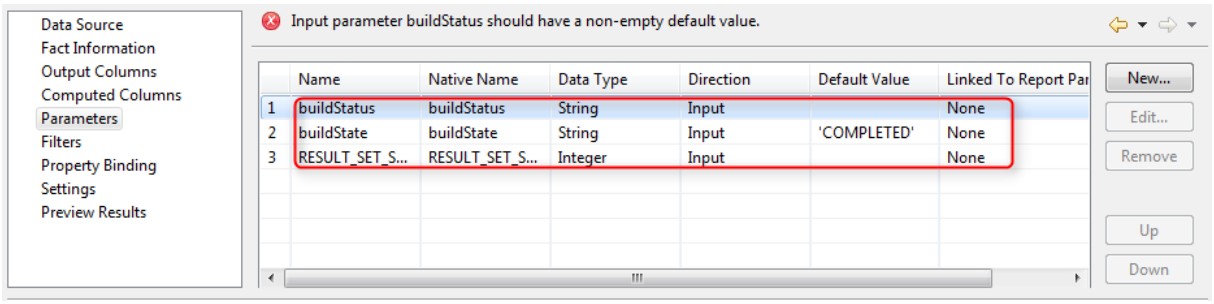
Parameter vs Filters

Once a BIRT data source has been created and a data set to query a database table has been added to the report this data set usually fetches all rows and the defined columns from this table. In a worst case this could result in fetching all available rows and columns from a data warehouse database table when the report is deployed and executed on the server.

Due to the fact the data warehouse is usually updated every night with the status or changes of the RTC live data this could result in a huge amount of data, e.g.:

The data set is about to query the WORKITEMS_STATES table in the WORKITEMS_SNAPSHOT. Let's assume there are actually 1.000 work items in the RTC LIVE database and the RTC server is up and running for 100 days. This will result in 100.000 database rows fetched with the defined data set and this number increases day by day and additionally by every new work item.

To reduce the number of fetched database rows it's essential to use a data set's parameter option. A parameter can be set for each query-able column in a database field and it behaves as a kind of condition in an SQL where clause.



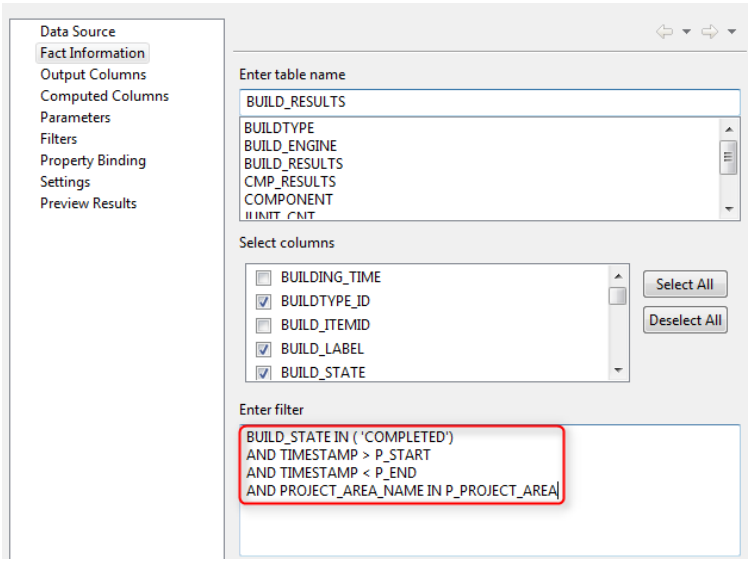
Filtering using a couple of very comfortable conditions is the other option to reduce the number of rows shown as a result of a data set query. Anyway before the filter in a data set is applied it fetches all rows from the database table. This means the entire data will be loaded into the applications memory and filtered in the application's memory.

A BIRT report developer should prevent the usage of filters even at a glance they seem to be more comfortable because of the available conditions. The parameter usage should be the first choice to minimize the data to be fetched.

Advanced Data Set

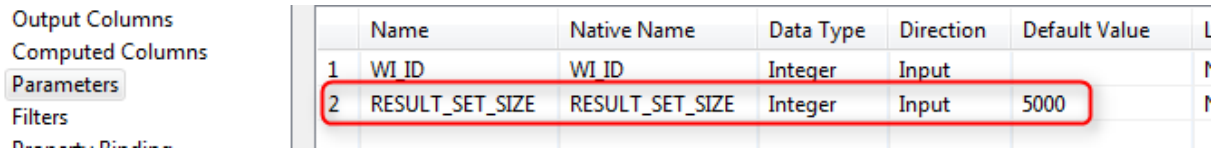
The Advanced Data Set is a more customizable data set to query the RTC database. It allows the developer to create an SQL-Style parameter which includes SQL keywords like "IN, LIKE, AND, OR" and it's also possible to use the available comparison operators " =, <>, >=, <=".

If the Simple Data Set based on equal and the logical AND does not fit the developer's filter criteria a report developer should consider the Advanced Data Set to reduce the number of fetched rows from the database.

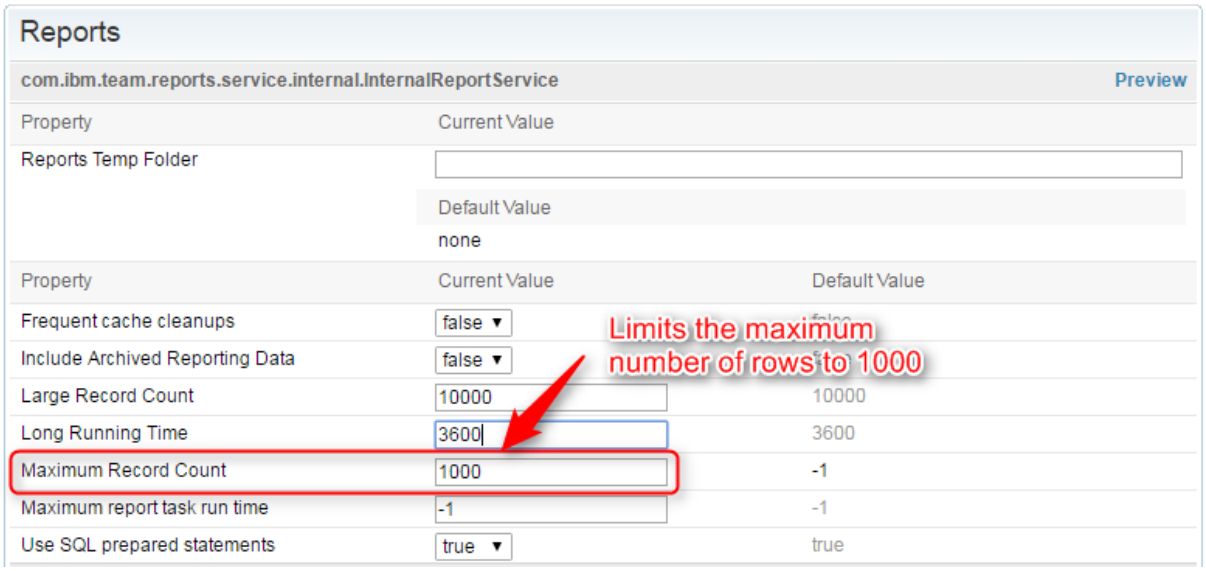


Limit the Result Set Size

To prevent a data set to fetch a huge amount of database rows it's possible to define the maximum number of rows which could be fetched from the database table. If the total number of rows is not known a report developer can limit the number of fetched rows by setting this parameter in a data set's parameter section.



The maximum number of rows fetched by a data set can also be set by a JazzAdmin in the RTC server's advanced properties.



Special Value Variables

Some BIRT parameter types have special values which may be used. This means those parameters accept special variables which will be expanded during runtime. The most common used variable is the '{Current Project Area}' which populates the project area name or item id parameter with the respective value of the project area the report is actually executed. (see also Ref. 2)

Scripting

One of BIRT's strongest capabilities is the possibility to use Java Script in a report. This allows report developers to update and convert data and makes it possible to apply calculations on this data. The Rhino Java Script Engine used by BIRT has also the capability to embed Java Classes like to the Java Script code which gives a report developer a wide range of objects that can be used.

Memory Consumption

A developer should always be aware that a Java Script or Java object requires memory of the RTC server. Therefore, it is mandatory to reduce the number of fetched rows as well as the number of fetched columns by row to the minimum of required data.

To show the memory consumption a bit more in detail refer to the following example:

```
var object =
 {
    'boolean' : true,
    'number'  : 1,
    'string'  : 'a',
    'array'   : [1, 2, 3]
 };
```

This simple Java Script object consumes approximately 78 Bytes.

Let's assume this example represents a row fetched from the RTC server.
If 100 of those rows are fetched 780 Bytes are used, 100.000 rows consume 78.000.000 Byte which is approximately 78 Megabytes. (see also Ref 3)

WI_ID	SUMMARY	CONTRIBUTOR_NAME	CREATOR_NAME	STATE_NAME	TIME_SPENT
31	Retrospective for Sprint 1	Peter Lustig (Dandelion Ltd.)	Peter Lustig (Dande...	com.ibm.team.workitem.retrospectiveWorkflow.state.inprogress	-1

The above row from the RTC LIVE_WORKITEM_CNT table consumes approximately 390 bytes.

Loops

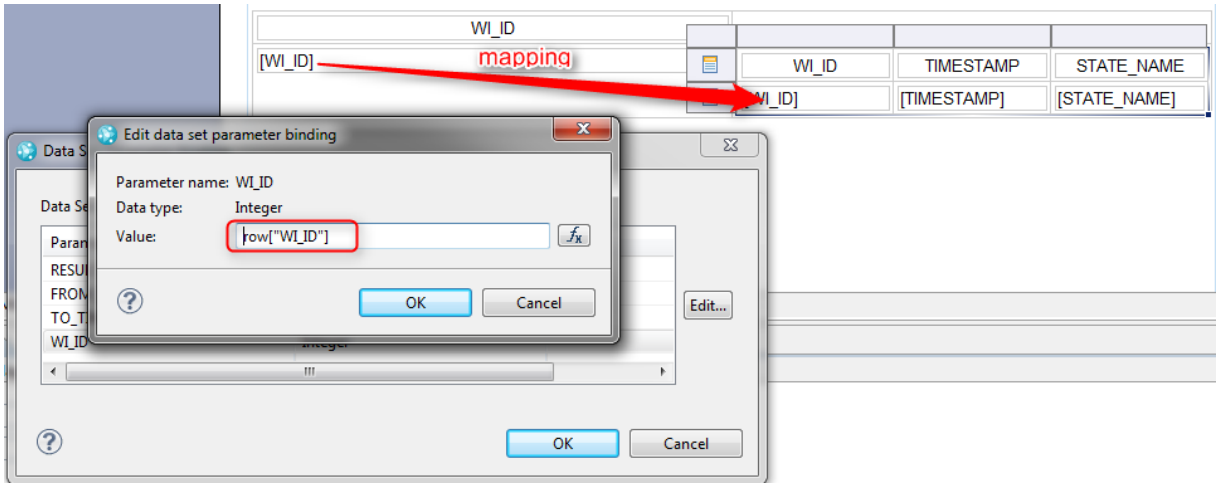
A report developer should take care of loops used in the Java Script code. Whenever possible the developer should implement a kind of exit criteria to prevent endless loops.

Nested Tables

In some cases, it is not possible to use the given parameter capabilities of an Advanced or Simple Data Set, e.g., fetching specific historical information for certain work items in the data warehouse. This can be mastered by using the nested table concept.

Within this concept there is a result table embedded into a kind of criteria table which contains the filter criteria for each row in the result table.

Let's assume you would like to fetch the trend data for a couple of work items. Then you simply have to create an outer table containing the work item ids and map the work item id to an inner table containing the trend data. The work item id will be the filter criteria and shows only information for the mapped work items.



The above shown screenshot shows an example of a nested table. The outer table is bound to the LIVE_WORKITEM_CNT table and simply fetches the work item ids which match a certain criterion. This work item id is mapped to an inner table bound to WORKITEM_STATES which shows the work item states by day.

Dashboard

Dashboards in the IBM C/ALM solution give users a brief overview on a project's health and the activities. Reports can also be embedded to such a Dashboard and enhances the user experiences with trend or status overviews in a small widget.

BIRT Dashboard Widgets

The space on a dashboard is limited. Therefore, a report developer should ensure that the shown widget does not consume the entire tab of a dashboard. A dashboard widget should also be expandable so that a user can enlarge the widget based on its demand.

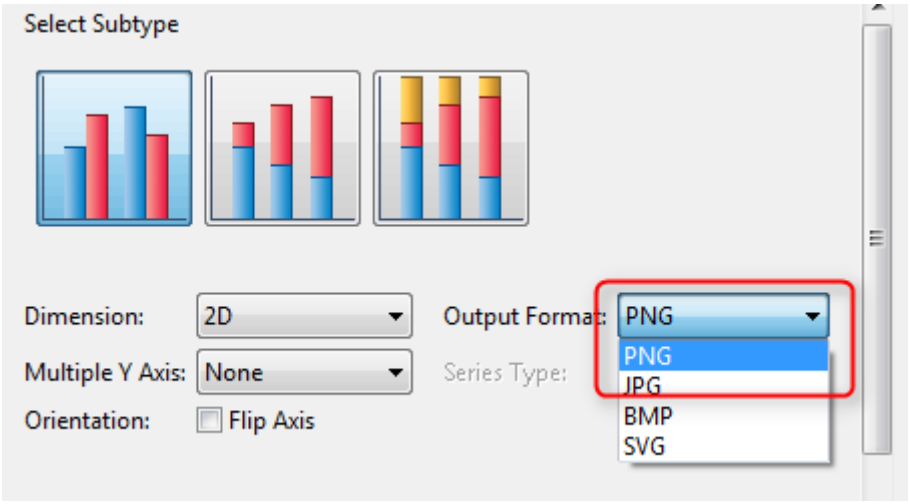
Shown Data

A report developer should always think about the data which is shown in a widget. If a report even though it's developed based on the above listed hints with regards to memory usage reduction, consumes a big amount of server memory the widget shouldn't be implemented. This is with regards to the possibility that there can be more than one of the same widget on a dashboard tab with different configurations. Each instance of a widget consumes its own memory range which means the more widgets of the same type are used the more memory will be consumed.

Additional Best Practices

Charting

Charts can be rendered in four different formats: PNG, JPG, BMP and SVG. The default selected when inserting a chart is SVG. In order to use SVG the web browser which shows the report needs a special plug-in. As a common format PNG proved itself in practice. Therefore it's recommended to use this format for each chart added to a report.



Report Layout

Within BIRT it's possible to create hidden elements which can be used for e.g. calculations on fetched data. It's most common to use such a hidden dynamic text to fetch data from the RTC database which will later be used for a scripted data set. Obsolete elements can increase the memory consumption even though they don't have any usability. Therefore, ensure to remove all not needed elements from the report design.

References

1. BIRT Reporting on Jazz.net (https://jazz.net/wiki/bin/view/Main/ReportsMain)
2. BIRT Report Parameter Handling (https://jazz.net/wiki/bin/view/Main/ReportsParameters)

3. Memory Usage of Java Script Object (http://code.stephenmorley.org/javascript/finding-the-memory-usage-of-objects/)

4. Create Custom Reports with BIRT (http://www.ibm.com/developerworks/rational/library/create-custom-reports-birt-rtc/)

5. Jazz.net Forum (https://jazz.net/forum/tags/birt/)

Author : Matthias Buettgen, IT Specialist,IBM Watson IoT