HITACHI
Inspire the Next

Report Bursting with Pentaho

HITACHI
Inspire the Next

Change log (if you want to use it):

Date	Version	Author	Changes

Contents

Overview

This document covers some best practices on report bursting using the Pentaho platform, including ways in which you can configure Pentaho components to achieve report bursting.

The intention of this document is to speak about topics generally; however, these are the specific versions covered here:

Software	Version(s)
Pentaho	7.x, 8.x
Tomcat	6.x+

The Components Reference in Pentaho Documentation has a complete list of supported software and hardware.

Before You Begin

Before beginning, use the following information to prepare for the procedures described in the main section of the document.

Terms You Should Know

Here are some terms you should be familiar with:

- **Data modeling:** the process of creating a logical model of how data elements, usually in a database or other data store, relates to one another
- **REST API:** Representational State Transfer Application Programming Interface that allows for easy-to-set-up cross-platform communication using different protocols, most notably HTTP. The Pentaho platform includes a rich REST API library that allows for interoperability of the various components of the platform with little effort.
- **Scheduler:** the scheduling engine used by Pentaho for managing and executing schedules for jobs and reports used on DI and BA applications

Other Prerequisites

This document assumes that you are familiar with and have installed Pentaho. It also assumes that you are familiar with creating reports using Pentaho Report Designer (PRD) and data integration jobs and transformations using Pentaho Data Integration (PDI).

Use Cases

Use cases employed in this document include the following:

Distribute Personalized Content to Email Inboxes

Wade is an administrator. He is employed by an organization that delivers analytics to its business users. He is seeking an easier way for his sales team to receive personalized reports through email on their mobile devices, without the need of the Pentaho User Console (PUC). He will need to design a report solution to meet this goal.

Distribute Personalized Content to Network Share Drives

Janice works at a manufacturing company and wants to make it easier for her warehouse managers to receive inventory stock-level reports. She has been using the Pentaho platform for delivering analytics to its business users. Because the company has many global warehouses, they need a solution that retrieves weekly data for each warehouse and generates personalized reports from managers. The Man also wants to log these reports for many years and be able to easily access them. Creating a report bursting solution that delivers reports to folders in network shared drives would help the manufacturing company reach its goal.

Report Bursting

Report bursting is a process by which you refresh a report against designated data sources, and personalize the generated report content before delivering the resulting document (such as a PDF, or Excel, or text file) to intended recipients or locations within or outside the Pentaho platform. Personalized content can be delivered to recipients' email inboxes or designated directories on secured or unsecured network drives.

The Pentaho platform includes a few report types:

- Pentaho Analyzer reports built using the Pentaho Analyzer plugin in PUC
- Pentaho Reports (`.prpt` files) built using PRD
- Pentaho Interactive Reports (`.prpti`) built using PUC

Report bursting depends on the type of report and platform being used:

- Use the Pentaho Reporting Output step within transformations or jobs.
- Set role/user security in PUC.
- Use PUC to schedule report bursting.
- Build data models using either Pentaho Metadata Editor for relational data or Pentaho Schema Workbench for multidimensional data.

The Old Way

In the past, you could use action sequences known as **xactions** for report bursting on the Pentaho platform. Xactions are sequences of actions or instructions, expressed as entries in an XML file, that the Pentaho platform performs in a specific order. They execute a report based on users and parameters, with the resulting file being delivered to designated destinations.

Since we no longer support the interface for creating and editing sequence actions (Eclipse Studio), we no longer suggest using xactions.

Although xactions still function on the Pentaho platform, they are no longer necessary to trigger events. PDI now provides a better alternative to trigger events on the platform using the REST API.

The New Ways

Now, you have two choices for setting up report bursting with the Pentaho platform: using a PDI transformation stored in the Pentaho Enterprise Repository, or using a combination of PUC and a PDI job:

- Method 1: PDI Transformation
- Method 2: Pentaho Server and Data Models

Method 1: PDI Transformation

Pentaho allows you to create parameterized reports with a PDI transformation, providing an easier way to develop, debug, and maintain report bursting solutions. This example transformation retrieves report data (sales data) from one or more tables in a database and joins the data with recipient email addresses (sales managers). In this case, the sales data includes a field that is common with the recipient email address dataset, which is the `employee ID`.

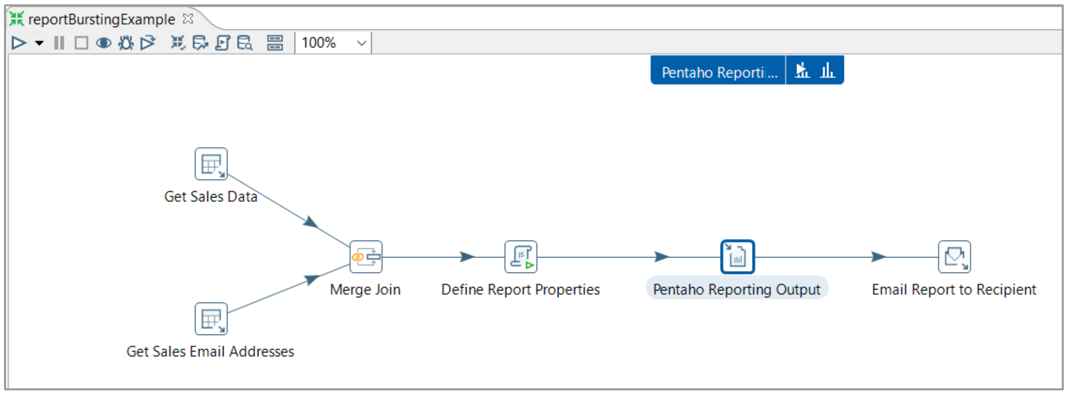
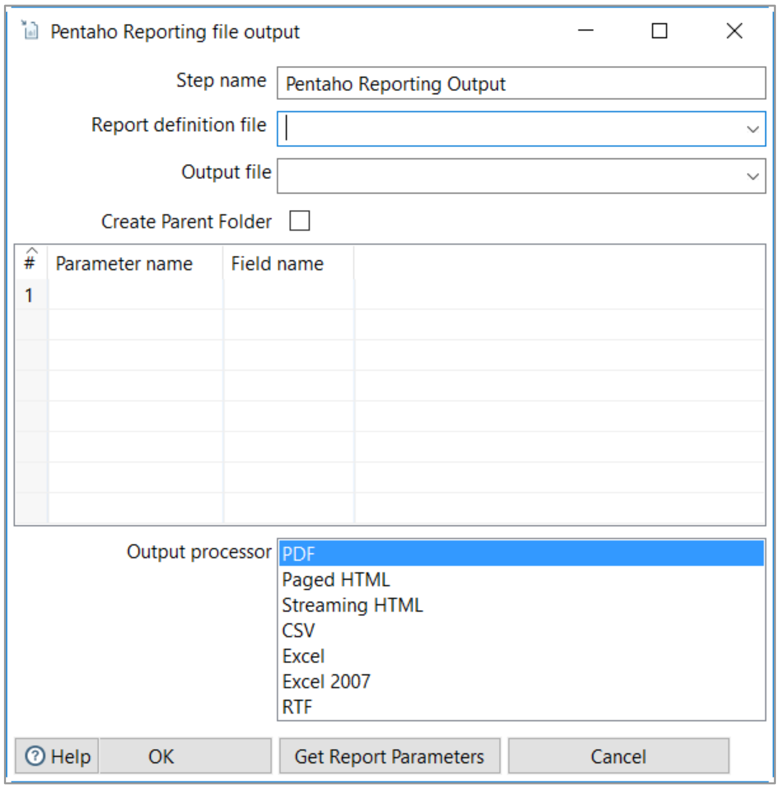


Figure 1: PDI Transformation

After you set report properties, the **Pentaho Reporting Output** step receives the data, allowing for the definition of the report file, output file, parameters used in the report, and output file type.:



Step name: Pentaho Reporting Output

Output processor: PDF

Figure 2: Pentaho Reporting Output Step

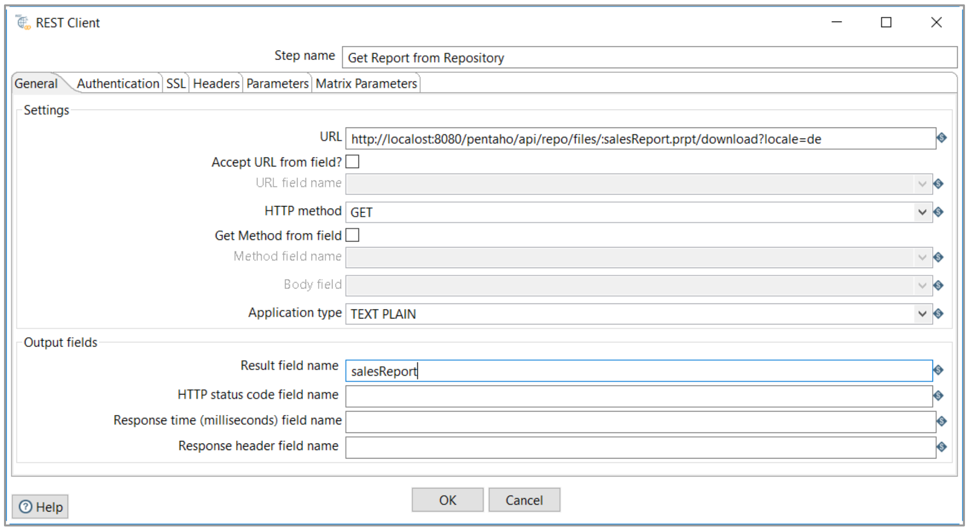
*The **Pentaho Reporting Output** step only supports reports designed with Pentaho Report Designer (`.prpt` files). For report bursting options with other file types, see subsequent sections of this document.*

The **Pentaho Reporting Output** step cannot connect directly to the Pentaho repository to find and use a report. The report file must be on the local filesystem where the transformation will be run.

There are alternatives to solving this problem to avoid keeping and maintaining a local copy of the report on the filesystem, and another in the Pentaho Repository. You could extend the example transformation shown in Figure 1, or call it from a job that first downloads the report file locally in one of these ways:

REST API Call

Use a REST API call (`/repo/files/{pathId}/download`) in the PDI job to download the report file from the repository to the Pentaho repository, with the **REST Client** step in PDI:



Step name: Get Report from Repository

URL: http://localost:8080/pentaho/api/repo/files/:salesReport.prpt/download?locale=de
Accept URL from field? ☐
URL field name:
HTTP method: GET
Get Method from field ☐
Method field name:
Body field:
Application type: TEXT PLAIN

Result field name: salesReport
HTTP status code field name:
Response time (milliseconds) field name:
Response header field name:

Figure 3: Get Report from Repository

Import/Export Script

Export the report file from the repository using the import/export script (`.bat`/`.sh`):

1. Open a command prompt.
2. Go to the location where you have a local copy of the Pentaho Server installed, such as `C:\dev\pentaho\pentaho-server`.
3. Enter a space, then type the arguments for download into the command line interface. A completed download argument would look something like this (edit the download `path` as needed):

```
import-export.bat --export --url=http://localhost:8080/pentaho --
username=dvader --password=password --charset=UTF-8 --
path=/public/Marketing/WeeklySales.prpt --file-
path=C:/Users/dvader/Downloads/WeeklySales.prpt.zip --overwrite=true --
permission=true --retainOwnership=true
```

Method 2: Pentaho Server and Data Models

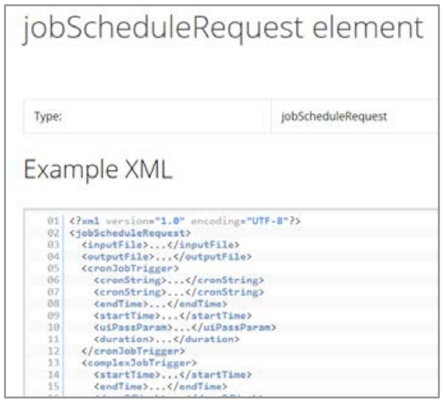
Another method for report bursting uses a combination of tools within the Pentaho platform. This method is more applicable to reports that use data models built using Pentaho modeling tools, such as Schema Workbench for multidimensional data (data marts → OLAP cubes) and Metadata Editor for relational data.

This method may result in the creation of many roles and schedules in the Pentaho Enterprise Repository, depending on the number of intended recipients or destinations of personalized content. Although you can delete schedules using the REST API, they could still add processing overhead per execution.

1. Use PUC and your users or roles to apply row-level security to constraints applied in the data model used to create reports, mapping the roles/users to the security filter being applied to the model.

We recommend you create roles to which rights can be assigned in PUC, and then *assign users to those rights. For example, create roles for each sales region that should receive personalized content.*

2. Use PUC to schedule the report as a user/role member, to trigger the reports to generate personalized content.
3. Use a PDI job that retrieves the users and uses them to generate an XML or JSON `jobScheduleRequest` object that is injected into BA or Pentaho Server, using the `/scheduler/triggerNow` API call, using the **REST Client** step in PDI (see the following three figures):

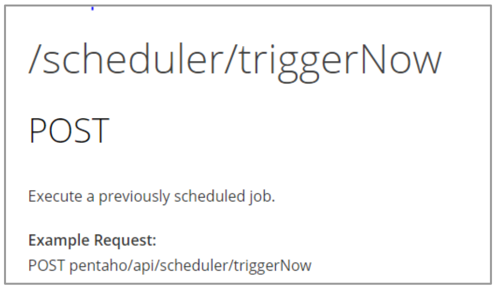

jobScheduleRequest element

Type:	jobScheduleRequest

Example XML

```
<?xml version="1.0" encoding="UTF-8"?>
<jobScheduleRequest>
  <inputFile>...</inputFile>
  <outputFile>...</outputFile>
  <cronJobTrigger>
    <cronString>...</cronString>
    <cronString>...</cronString>
    <endTime>...</endTime>
    <startTime>...</startTime>
    <uiPassParam>...</uiPassParam>
    <duration>...</duration>
  </cronJobTrigger>
  <complexJobTrigger>
    <startTime>...</startTime>
    <endTime>...</endTime>
```

Figure 4: `jobScheduleRequest` Element

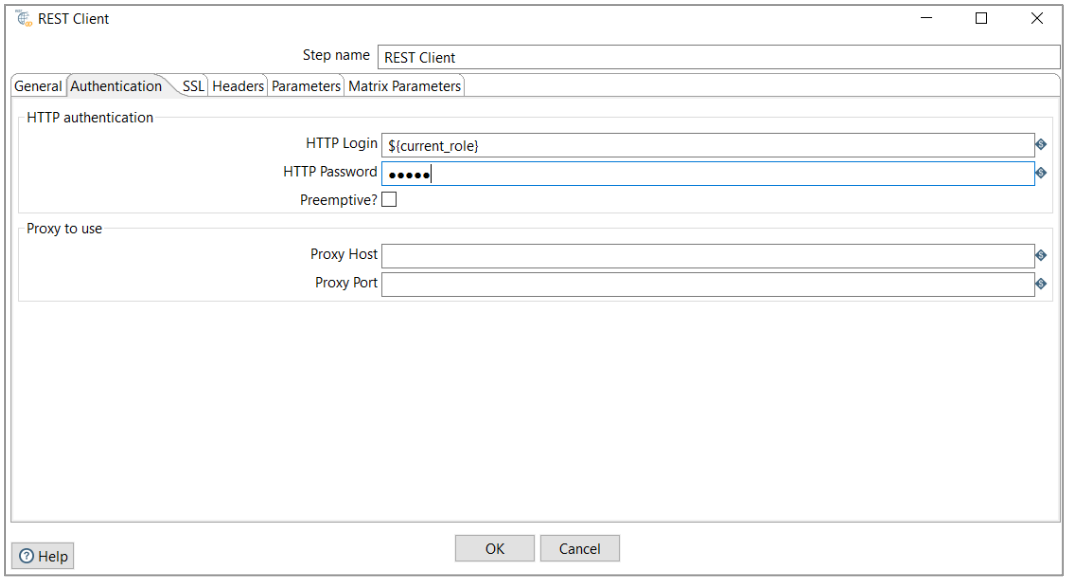

/scheduler/triggerNow

POST

Execute a previously scheduled job.

Example Request:
POST pentaho/api/scheduler/triggerNow

Figure 5: `/scheduler/triggerNow`


REST Client

Step name: REST Client

General | Authentication | SSL | Headers | Parameters | Matrix Parameters

HTTP authentication
HTTP Login: ${current_role}
HTTP Password: •••••
Preemptive? ☐

Proxy to use
Proxy Host:
Proxy Port:

*Figure 6: **REST Client** Step Configuration*

4. Content will be generated and stored in destination specified in the `jobScheduleRequest`.

Because this method of report bursting depends on data models and security in PUC, it is applicable to Analyzer and Interactive Reports.

Report Bursting Design Recommendations

When you have a resource-intensive query, work with your database administrator (DBA) to tune the query:

- The DBA helps optimize aspects of the query based on the Query Plan and database statistics gathering (`UPDATE STATISTICS`).
- The DBA may add a needed index or two to one or more tables to greatly improve the query performance.
- The DBA may look at creating a view (materialized or regular) which, when combined with a query optimization may dramatically improve query performance.
- The DBA may implement (or already have implemented) sharding/partitioning and recommend changes to the query that take advantage of how the data is physically organized.
- The DBA may implement a proper data warehouse which should be optimized for querying.

Although you may *load all data and filter using PDI steps for resource-intensive queries, it should not be your* first *option.*

When using PUC, data models, and scheduling to achieve report bursting, consider the following:

- Design the model to include security filters on fields or columns that are indexed at the database level, to improve performance.
- Periodically delete schedules for roles that are no longer in use.

Follow this report bursting example video for a better understanding.

Related Information

Here are some links to information that you may find helpful while using this best practices document:

- Adding Row Level Security to a Pentaho Metadata Model
- Command Line Arguments Reference
- Components Reference
- Download Files from the Pentaho Enterprise Repository
- Download from the Command Line
- `jobScheduleRequest`
- Manage Users and Roles in Pentaho User Console
- Multidimensional Data Modeling in Pentaho
- Pentaho Metadata Editor
- Pentaho Reporting Output Step
- Pentaho Schema Workbench
- REST API Reference
- Schedule Reports
- Report Bursting Demonstration

Finalization Checklist

This checklist is designed to be added to any implemented project that uses this collection of best practices, to verify that all items have been considered and reviews have been performed.

Name of the Project:____

Date of the Review:____

Name of the Reviewer:____

Item	Response	Comments
Did you use the **Pentaho Reporting Output** step or plugin within transformations or jobs?	YES_______ NO_______	
Did you use a REST API call in the PDI job to download the report file from the repository to the Pentaho repository?	YES_______ NO_______	
Did you export the report file from the repository using the `.bat/.sh` import/export script?	YES_______ NO_______	
Did you create roles to which the rights can be assigned in PUC, and then assign them to users?	YES_______ NO_______	