



Working with Pentaho Interactive Reporting and Metadata

HITACHI

Inspire the Next

Change log (if you want to use it):

Date	Version	Author	Changes

Contents

- Overview..... 1
- Before You Begin..... 1
- Other Prerequisites **Error! Bookmark not defined.**
- Use Case: Reporting from a Large, Slow Dataset Without Crashing the Server..... 1
- Pentaho Interactive Reporting and Data..... 2
- Best Practices for Pentaho Interactive Reports 2
- Disable Auto Refresh – Users 2
- Limit the Number of Query Rows (Pentaho 6.x Only) – System Administrator..... 2
- Building a Metadata Model for Interactive Reporting..... 3
- Tips and Recommendations for Building a Metadata Model 3
- Best Practices for Metadata Models..... 4
- Special Considerations for Big Data (Hadoop) Data Sources..... 4
- Applying Tips and Recommendations..... 5
- Related Information..... 6
- Finalization Checklist..... 6

This page intentionally left blank.

Overview

This document covers best practices related to Pentaho Interactive Reporting (PIR) and building the metadata layer that PIR uses. This document also covers standard implementations and special conditions that apply for Big Data use cases.

Our intended audience consists of developers, administrators, and those who wish to create interactive reports whether working with large data sets or slow databases.

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

Software	Version(s)
Pentaho	6.x, 7.x, 8.0

The [Components Reference](#) in Pentaho Documentation has a complete list of supported software and hardware.

Before You Begin

This document assumes that you have knowledge of Pentaho, Big Data (Hadoop) Data sources, and Metadata modeling.

Use Case: Reporting from a Large, Slow Dataset Without Crashing the Server

Wade wants to build reports ad hoc, but he has vast amounts of data and millions of records in a database that operates slowly. He needs to find a good way to build his reports so that it does not affect the other data users in his company or crash the server.

Wade has decided to use Pentaho Interactive Reporting, which will allow him to build his reports without adversely affecting other data users. He will be able to manage his large, slow dataset while still getting the reports he needs.

Pentaho Interactive Reporting and Data

Pentaho Interactive Reporting is a drag-and-drop, browser-based design environment for interactive reports that allows you to quickly add elements to your report and format them to your preference. For example, tables referred to as `CUST_TBLE` or `ORDR_TBLE` in the data model can be presented in your report as **Customers** or **Orders**. For more information about working with PIR, see [Use Pentaho Interactive Reporting](#) in Pentaho Documentation. Common use cases are to extract listings and details stored in your data center.

Best Practices for Pentaho Interactive Reports

There are a couple of recommendations for working with large data sets or slower databases when creating interactive reports:

Disable Auto Refresh – Users

- **Ease of Use Recommendation for All Versions:** Disabling Auto Refresh allows the user to work in a data-less mode to create and edit the report. This allows for multiple changes before manually running the report to see the results.
- **Recommendation for Pentaho 6.x only:** Disable Auto Refresh while building the report layout if you are working with large datasets or if there is slow connectivity between PUC and the server or relational data source. You can manually refresh the report at any point in the process.
- **Description:** You might find that it is quicker to build the report if the data is not constantly refreshing or returning many rows. You can return to Auto Refresh mode once the report layout is complete. Data retrieval occurs once, and your report displays the requested data.
- **Rationale:** This reduces the number of requests and processing of data, while dragging all the fields to the canvas.

Limit the Number of Query Rows (Pentaho 6.x Only) – System Administrator

- **Recommendation:** Limit the number of rows that are displayed in user reports during the design process. You should also limit the number of seconds a query runs before timeout.
- **Description:** You can prevent too many resources from hitting your database server simultaneously by setting a system-wide maximum row limit for PIR. Users can still define their own design-time row limits in PIR; they will not be able to exceed the number of rows that you have specified. You can find the steps for setting system-wide maximum row limits for [Interactive Reports](#) in the Pentaho Documentation.
- **Rationale:** Imposing row limits and timeouts on queries will help avoid out-of-memory errors, or processes that consume too many resources on the database server.

Building a Metadata Model for Interactive Reporting

PIR uses the metadata layer of data. A Pentaho metadata model maps the physical structure of your database into a logical business model. These mappings are stored in a centralized metadata repository and allows administrators to do the following:

- Create business-language definitions for complex or cryptic database tables.
- Decrease the cost and impact associated with low-level database changes.
- Set security parameters limiting user's report access to data.
- Drive formatting on text, date, and numeric data, which improves report maintenance.
- Localize the information to the user's regional settings.

The following topics are covered in this section:

- [Tips and Recommendations for Building a Metadata Model](#)
- [Applying Tips and Recommendations](#)

For more information about metadata models, different layers and functions, and security, see [Work with Relational Data Models](#) in Pentaho's Documentation.

Tips and Recommendations for Building a Metadata Model

Building a data model with optimal performance can be difficult, and each use case requires that you use different approaches. This section will recommend guidelines for building the physical and logical layers for metadata usage.

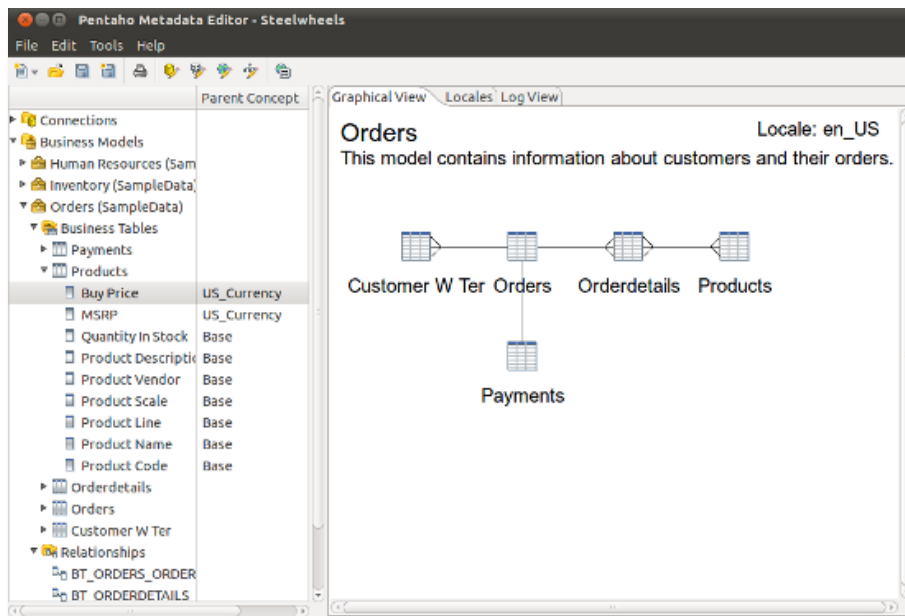


Figure 1: Pentaho Metadata Editor Interface

Best Practices for Metadata Models

The following is a list of recommendations for working with metadata models:

- Models should normally have **one grain table** (fact) and other tables that are attributes or dimensions. This is called a star schema or a snowflake schema.
- Try to **split multiple metadata models** or business layers to solve different data marts or fact tables. This means that each business view will use one fact table. For example, use separate models for billing, orders, inventory transactions, etc.
- Make sure there are **no circular relations** with more than one path to solve the same request.
- Use data source users with the [least \(but sufficient\) authority](#) to handle the task. For example, with data sources containing many objects, the database user should only be able to access the objects referenced in the model.
- Expose only **key** tables within the metadata model to keep the size of the model smaller.
- Make sure **measures and metrics** have the **proper aggregation functions** that apply to the measure and metric nature. For example, if a metric is a balance amount, we need to uncheck the possibility to SUM those values.
- **Create your formulas or calculations** ahead of time for the underlying table(s) to reduce engine overhead when possible.
- Make sure that **columns referenced in relationships are appropriately indexed** in your database, if needed.
- Avoid using complex metadata models (for example, models that contain numerous joins) for **static reports** such as those created with **PRD**. Use SQL instead.
- **Do not create only one metadata model** to solve all of your use cases. Use different domains with multiple models if you have several different use cases.

Special Considerations for Big Data (Hadoop) Data Sources

This section offers information on [Streamlined Data Refinery](#) and how to [Work with Streamlined Data Refinery](#) when dealing with Hadoop, Hive, and metadata.

- For Hive limitations, refer to [Hadoop Hive-Specific SQL Limitations](#).
- It is important to set proper timeouts and row limits, and to avoid the use of Auto Refresh functionality, because users can request large datasets.
- Use PIR and metadata in combination with the Streamlined Data Refinery for a better user experience.
- Consider reducing the number of Joins and use big FAT tables to speed performance.

Applying Tips and Recommendations

We will use an example to highlight potential issues and solutions for building a metadata model for interactive reporting. This will clarify the above tips and recommendations. Keep in mind that every case may be different, so certain tips or recommendations may not apply in every scenario.


For this example, we will use this tables that are commonly found in retail:

- customer
- order_header
- order_detail
- product
- vendor
- stock_movement
- stock_balance

In a relational third normal format (3NF), all the tables above are linked, but creating one model with all combinations could end up showing the wrong data.

We need to define different models based on what we want to report, like the following example converted to a simple star schema:

Table 1: Model Definitions

Model	Description
Sales	The main fact table is <code>order_detail</code> . The attributes are customer data, <code>order_header</code> information, and <code>product</code> and <code>vendor</code> . Metrics and measures are linked to the items sold count and total per product. The recommended calculations are product price line quantity in new field, and the amount of tax applied to the line, etc.
Inventory Current Stock	The main fact table is <code>stock_balance</code> , and details are in <code>product</code> . Metrics and measures are stock quantity and delta change.  <i>This model is normally a snapshot model with current stock for a specific date; therefore, stock quantity should not be aggregated.</i>
Inventory Movements	The main fact table is <code>stock_movement</code> . The attributes and dimensions are <code>product</code> , <code>order_detail</code> , and <code>order_header</code> .

Related Information

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

- Pentaho
 - [Components Reference](#)
 - [Create Relationships between Business Tables](#)
 - [Metadata Editor](#)
 - [Pentaho Interactive Reports](#)
 - [Set System Max Row Limit for Interactive Reports](#)
 - [Streamlined Data Refinery](#)
 - [Work with the Streamlined Data Refinery](#)
- [Principle of Least Privilege](#)

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
(Users) Did you disable Auto Refresh while building the layout for your report?	YES _____ NO _____	
(System Admin) Did you limit the number of query rows and the number of seconds a query runs before timeout?	YES _____ NO _____	
Did you create one metadata model to solve each use case, instead of one model for all?	YES _____ NO _____	