
ORACLE®

ORACLE®

Oracle BI Publisher 10g
Best Practices -- Session # 8633

Mike Donohue
Director, BI Product Management

Noelle Bartlam
Senior Member of Technical Staff, Development

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.

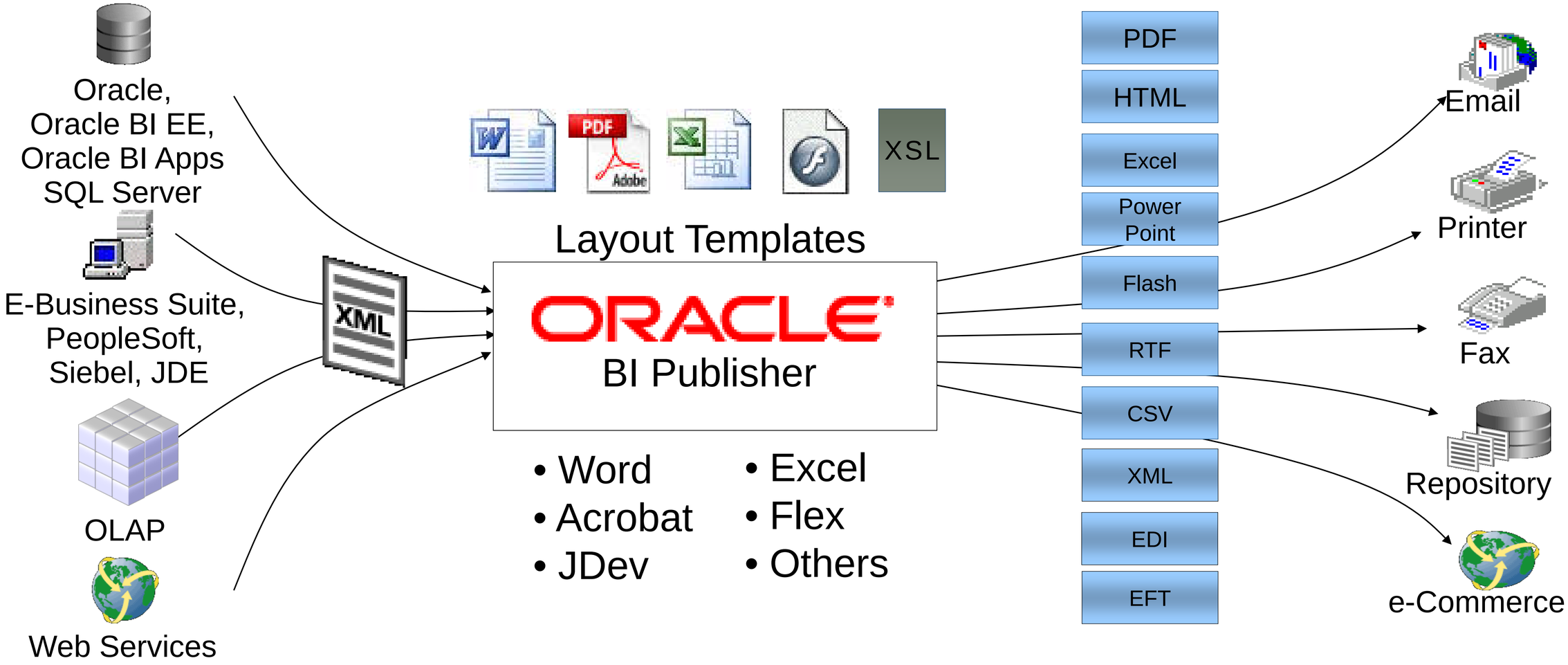
BI Publisher 10g Best Practices
Agenda

- Oracle BI Publisher Enterprise Overview
- System Configuration Guidelines
- BI Publisher Server Configuration Options
- Report Design Best Practices
  - Data Modeling Guidelines
  - Template Design Guidelines
- Q & A

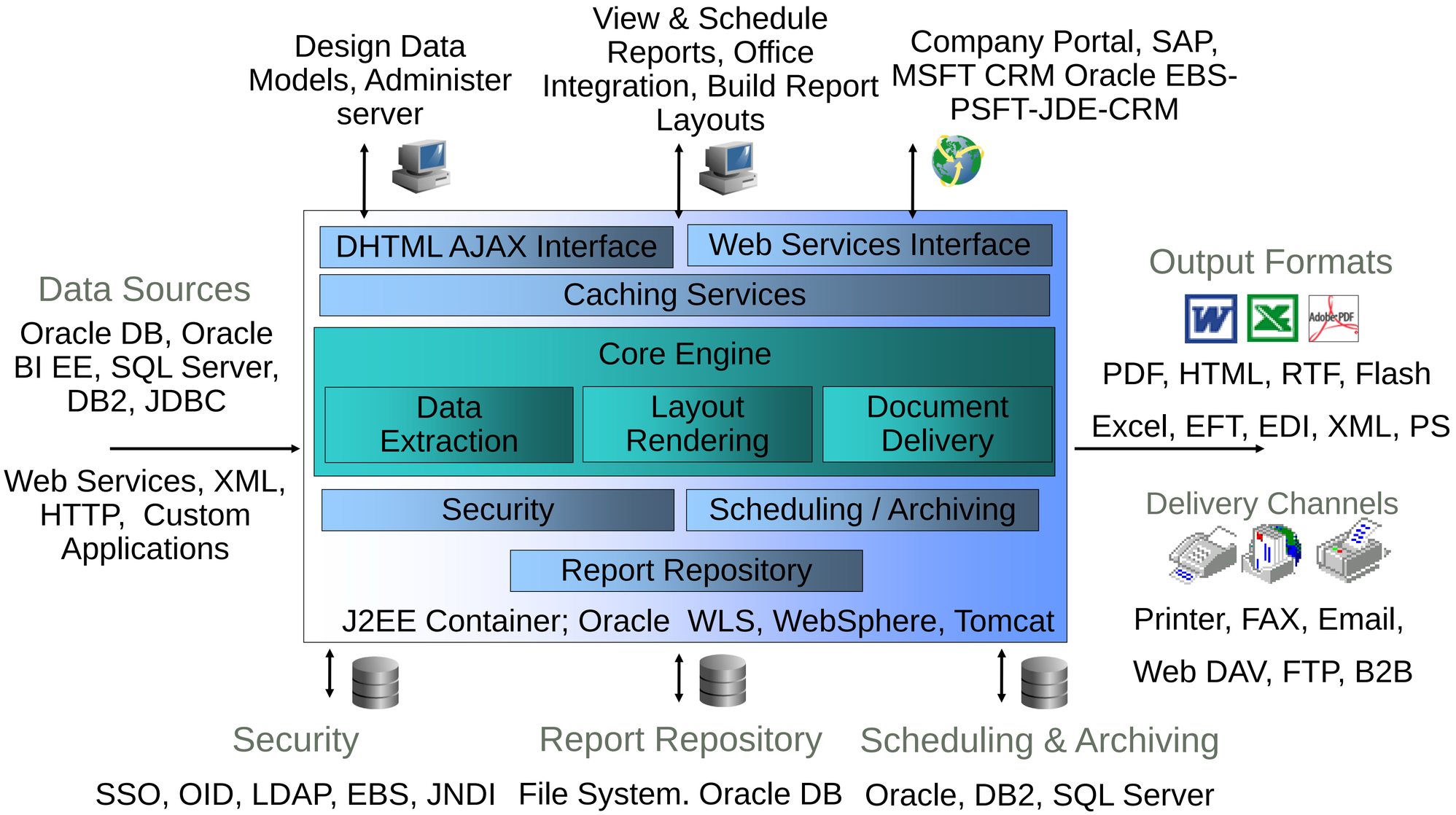
Oracle BI Publisher Enterprise
One Solution

- One Environment
  - Author
  - Generate
  - Deliver
- Benefits
  - Eliminate complexity
  - Simplify report development & maintenance
  - Reduce costs

Oracle BI Publisher Enterprise
From Data to Document to Delivery



Architecture



BI Publisher 10g Best Practices
Agenda

- Oracle BI Publisher Enterprise Overview
- System Configuration Guidelines
- BI Publisher Server Configuration Options
- Report Design Best Practices
  - Data Modeling Guidelines
  - Template Design Guidelines
- Q & A

Hardware & Software Requirements
Java, Memory & Storage

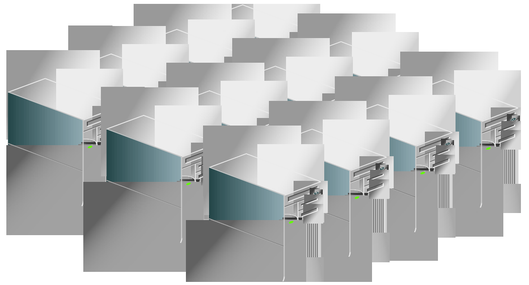
- JVM settings & JDK version
  - 64 bit JVM (on a 64 bit OS)
  - JDK version 1.6 (update 2) or higher
- Memory (RAM)
  - 8 GB on 64 bit OS is recommended for large, high volume use
  - 2 GB on 32 bit OS suitable for small to mid volume use
- Storage
  - 30 GB Hard disk space (repository) could be shared for cluster
  - 20 GB Temp space (document processing) not shared

Please refer to our certification matrix for supported OS platforms and J2EE Web servers.

The link is available in the Appendix.

Sizing Guidelines – How Many Severs?

- Usage type
  - Online/concurrent report viewing
  - Bursting
- Total number & size of reports
  - Online/concurrent scenario (Small, Medium, Large)
  - Bursting (total # of reports, time window)
- Number of CPU/server & CPU rating



Sizing Example

Metalink Note 948841.1

Concurrent Usage	
Sizing Data Input	
# user with small reports (~ 10 Pages)	20
# user with medium report (~ 50 pages)	20
# user with Large report (>100 Pages)	10
# user with extremely large report (> 100,000 rows)*	1
* Recommend offhour or dedicated server	

Bursting Usage		
Sizing Data Input		
Total Number of Reports	10000	
Time to Generate Reports	60	Minutes

Reports Distribution (Optional)	
Small reports (~ 10 pages)	80%
Medium reports (~ 50 pages)	10%
Large reports (> 100 pages)	10%

Production Machine	
Total Number of CPUs/Server	2
Spec2006 Ratings Info	
CPU CINT2006 Rates Base Score	47.10
# of CPUs used for benchmarking	2
Results	
Recommended Number of Servers	1
Server(s) utilization	68%

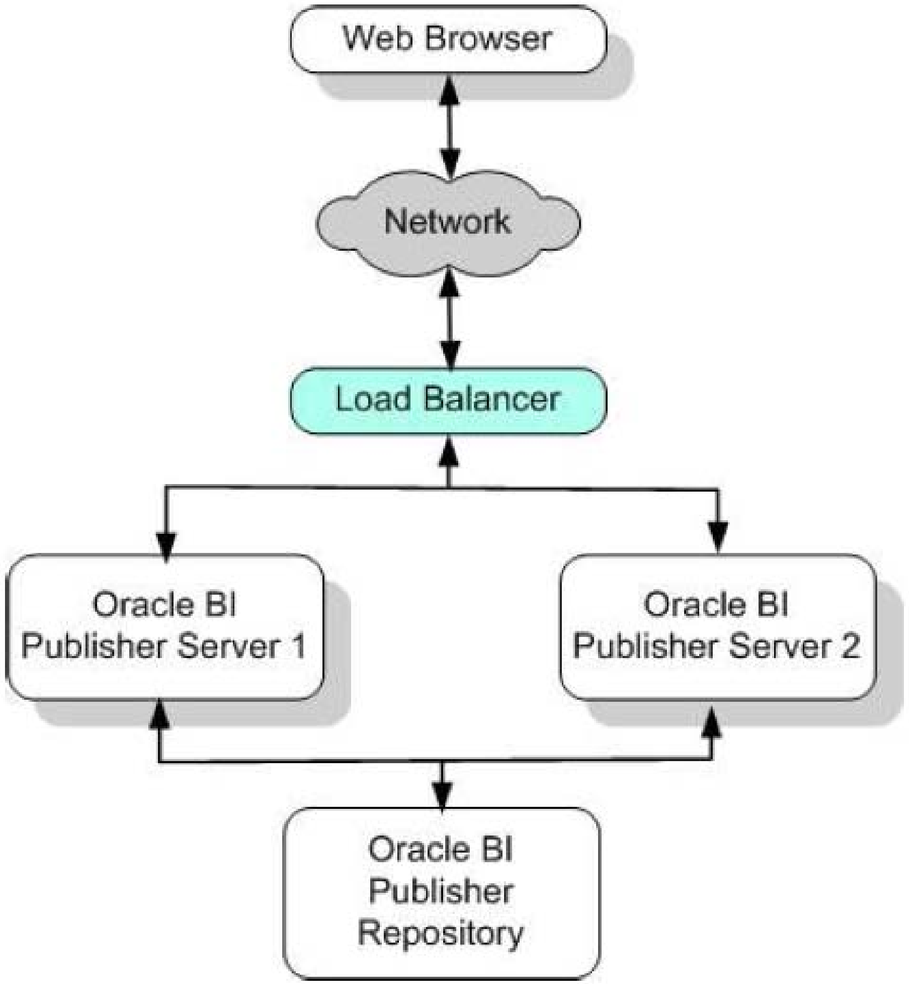
Production Machine	
Total Number of CPUs/Server	2
Spec2006 Ratings Info	
CPU CINT2006 Rates Base Score	24.10
# of CPUs used for benchmarking	1
Results	
Recommended Number of Servers	1
Server(s) utilization	13%

Deployment Options – Cluster Deployment

- High availability
  - Improve availability of the system
- Load balancing
  - Increase throughput

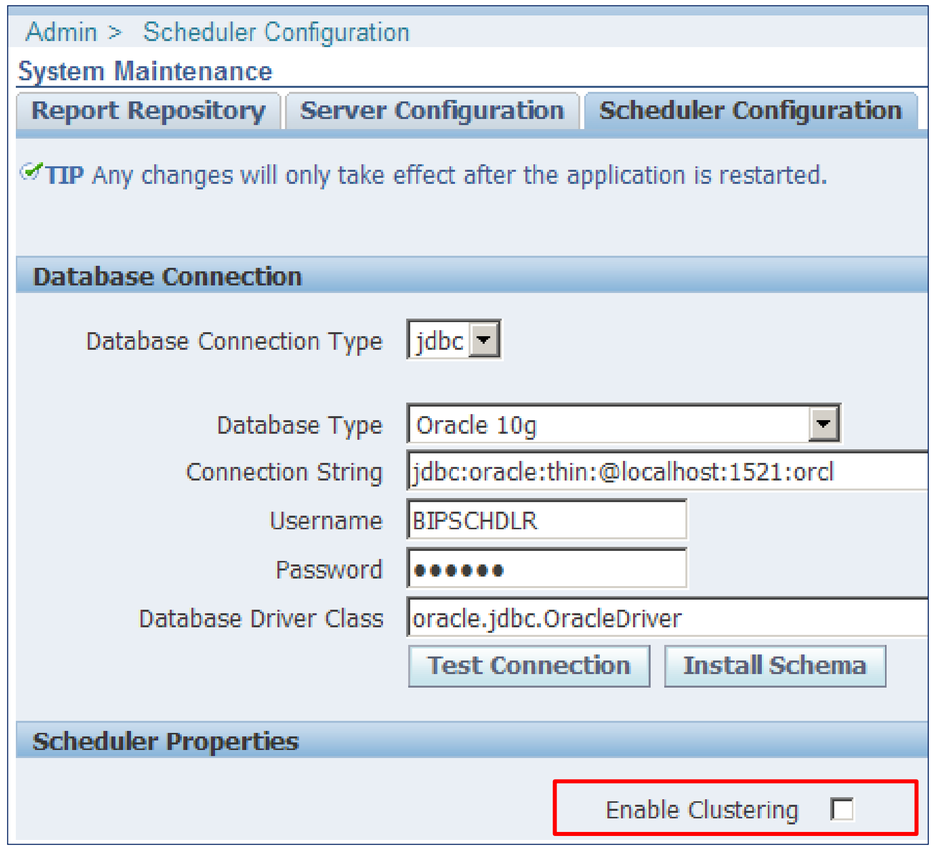
For more details on clustering, please refer to our White Paper on High Availability.

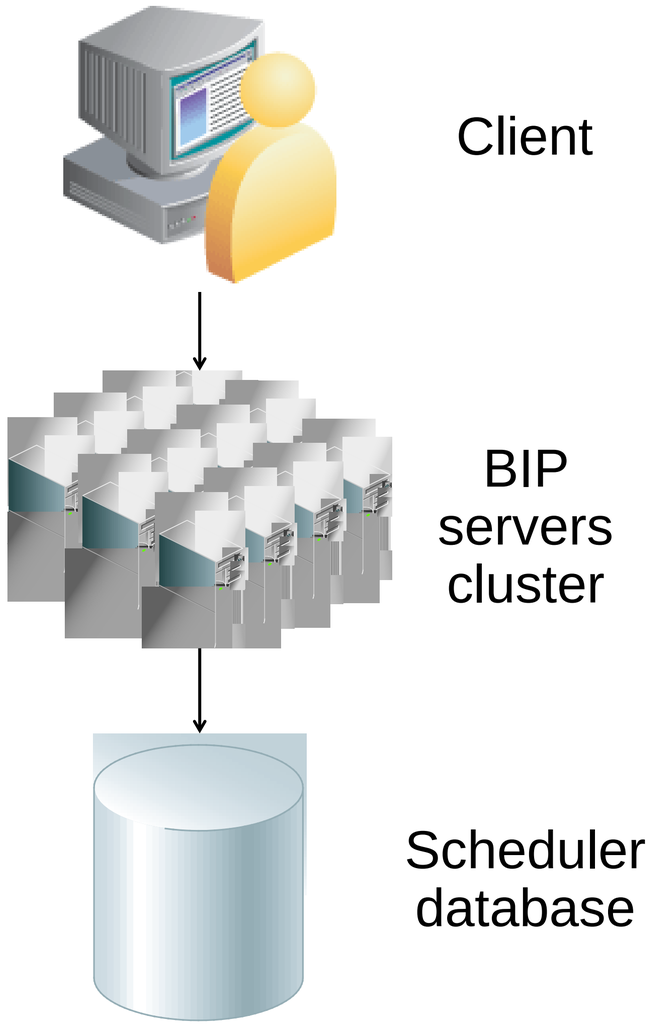
The link is available in the Appendix.



Deployment Options – Scheduler Configuration

- Enable clustering



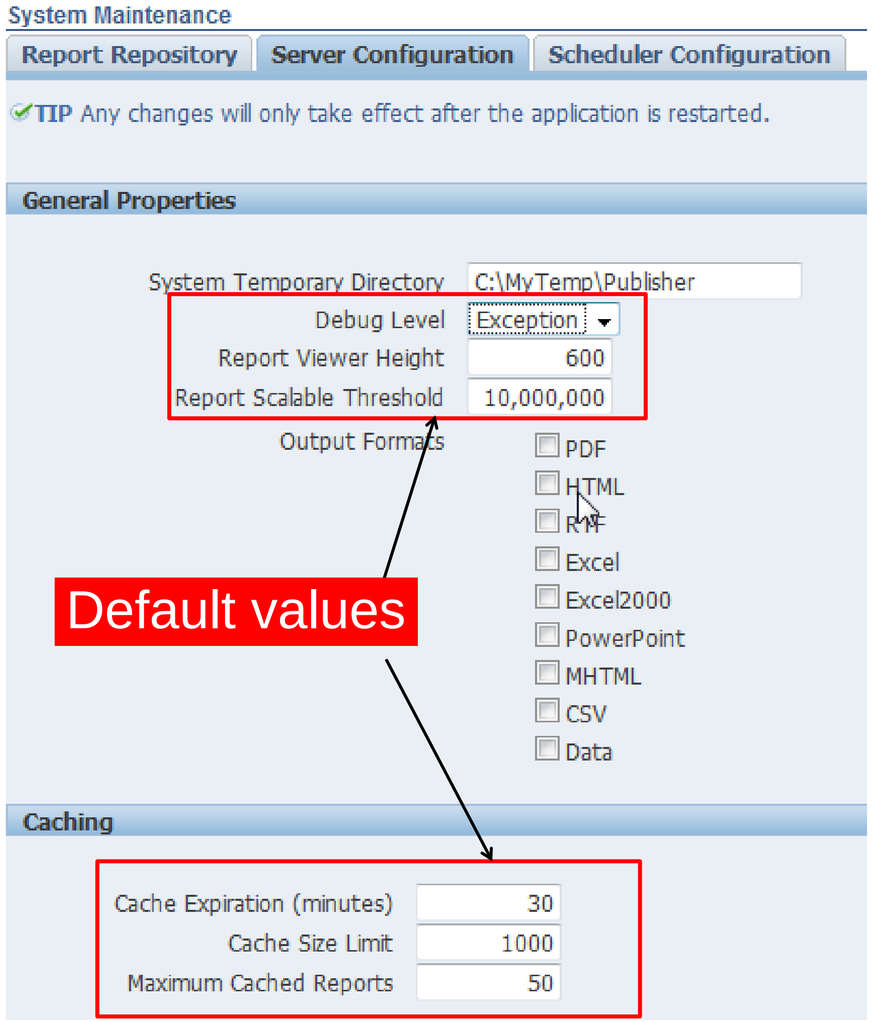


BI Publisher 10g Best Practices
Agenda

- Oracle BI Publisher Enterprise Overview
- System Configuration Guidelines
- BI Publisher Server Configuration Options
- Report Design Best Practices
  - Data Modeling Guidelines
  - Template Design Guidelines
- Q & A

Server Configuration Options

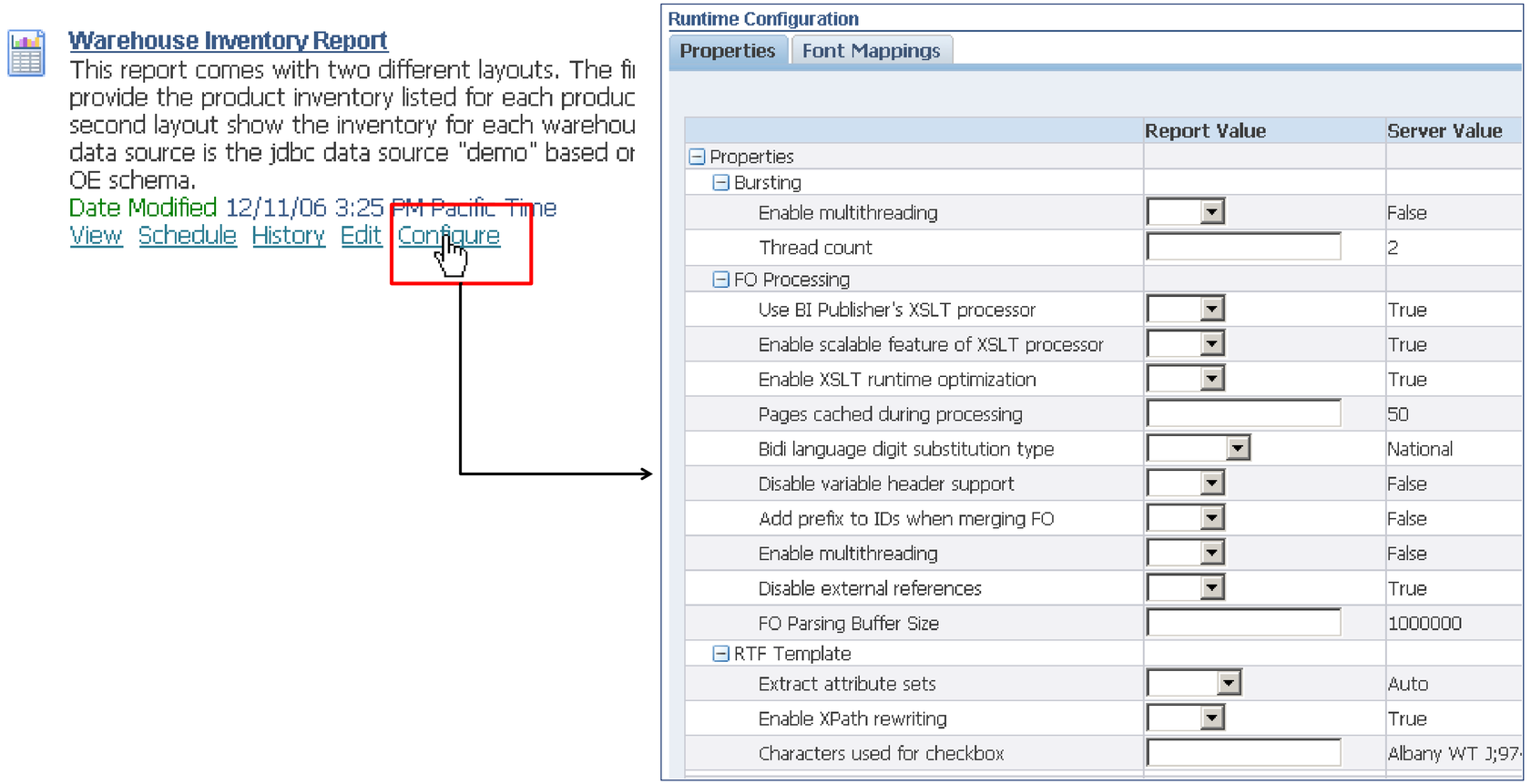
- Debug level*
- Scalable threshold (in bytes)*
- Cache expiration (in min)
- Cache size limit (# of datasets)
- Maximum cached reports



Server Configuration Options – Runtime Configuration Settings

- Bursting
  - Enable multithreading – *Default: False**
  - Thread count – *Default: 2**
- FO processing
  - Enable scalable feature of XSLT processor – *Default: False**
  - Enable multithreading – *Default: False**
  - FO Parsing Buffer Size – *Default: 1000000**
  - Use BI Publisher's XSLT processor – *Default: True*
  - Enable XSLT runtime optimization – *Default: True*
  - Pages cached during processing – *Default: 50*

Report-level Properties – Runtime Properties

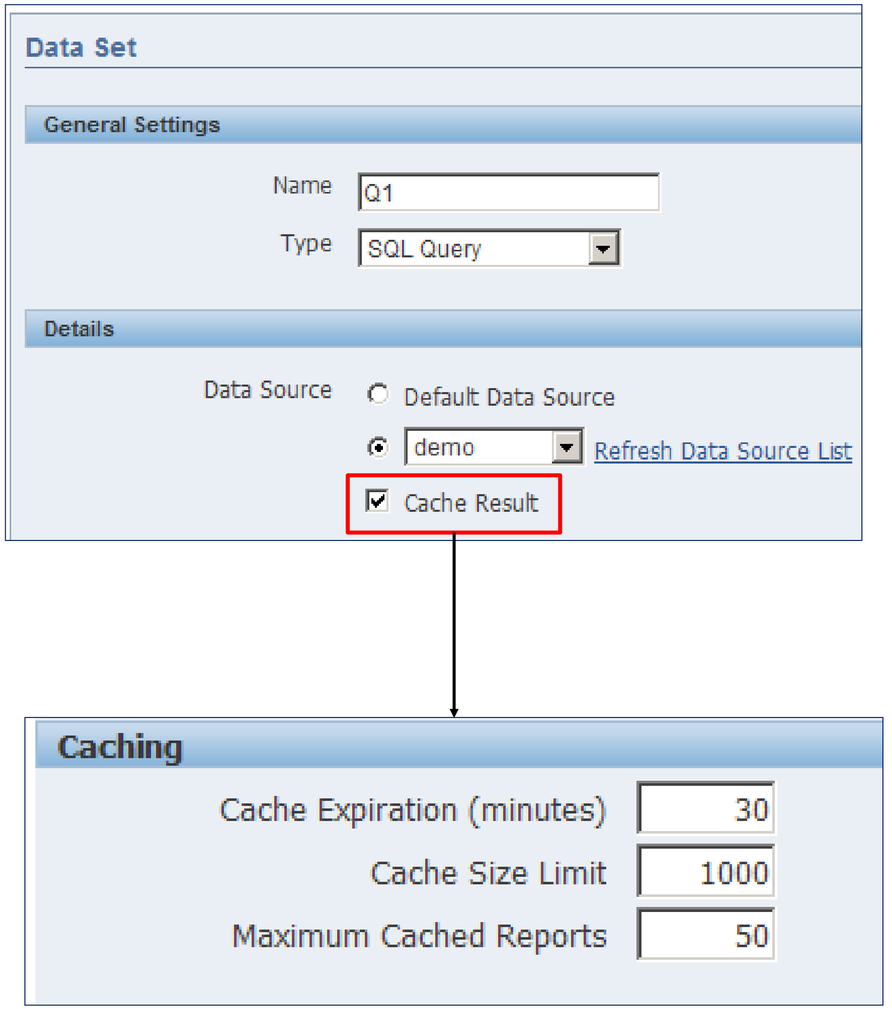


Report-level Properties – Data Model

- Set scalable mode property to “on” in Data Template

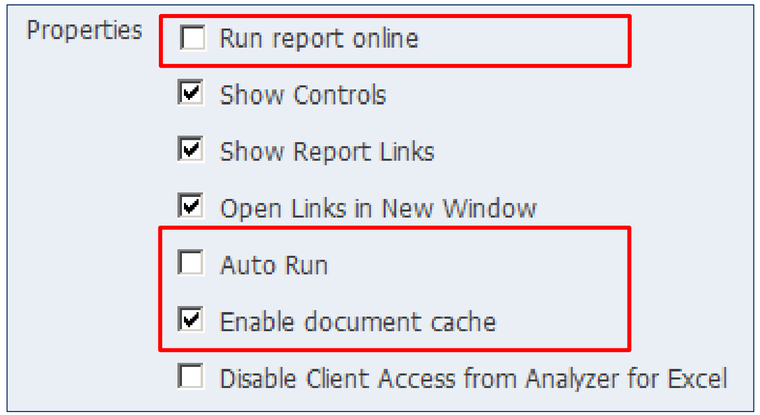
```
<properties>
    <property name="scalable_mode" value="on" />
</properties
```

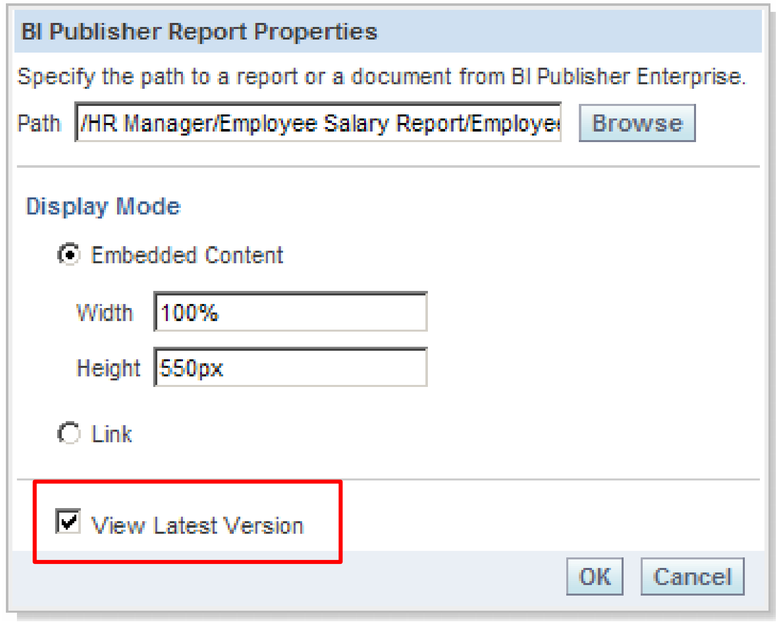
- Cache Result
  - Saves XML data for reuse up to the Cache Expiration threshold
  - Improved performance with different template/output format
  - Not recommended for real-time data



Report-level Properties – Viewing Reports Online

- For long running reports
  - Select ***Enable document cache***
  - Disable ***Auto Run***
  - Take advantage of scheduling and disable ***Run report online.***
  - **Tip:** If you are using OBIEE, publish the report to a Dashboard and select ***View Latest Version***.



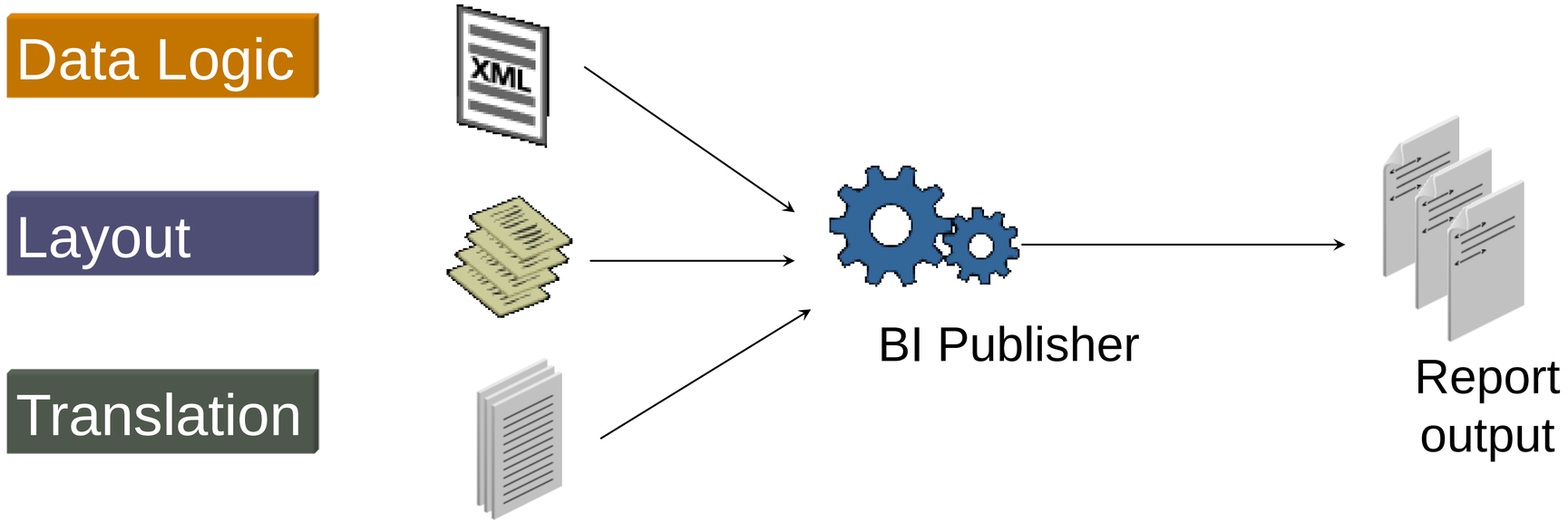


BI Publisher 10g Best Practices
Agenda

- Oracle BI Publisher Enterprise Overview
- System Configuration Guidelines
- BI Publisher Server Configuration Options
- Report Design Best Practices
  - Data Modeling Guidelines
  - Template Design Guidelines
- Q & A

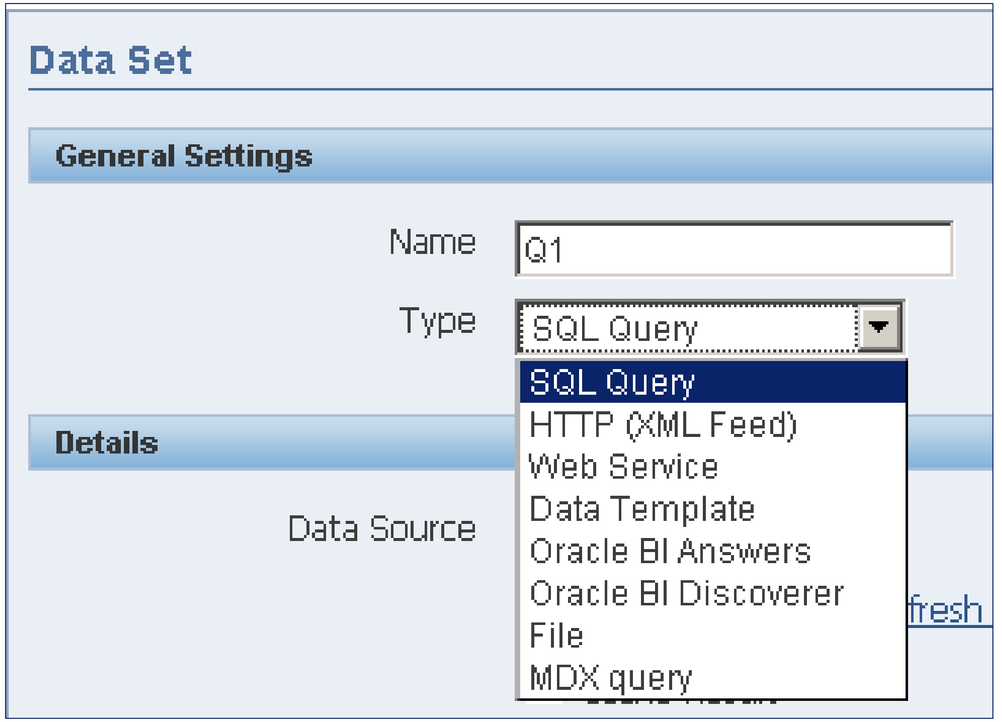
Oracle BI Publisher Enterprise – Report Architecture

- Separate data logic, layout & translation benefits
  - greater flexibility and reuse
  - fewer reports and better report maintenance
  - optimize data extraction and document generation process



Best Practices for XML Data
Data Extraction & Processing – Data Sources

- Data Source Types
  - File (XML)
  - Data Template
  - SQL Query
  - MDX Query
  - Web Service
  - Oracle BI Answers
  - Oracle BI Discoverer
  - HTTP (XML Feed)
- Connection Types
  - JDBC
  - JNDI (Recommended, can take advantage of connection pooling)

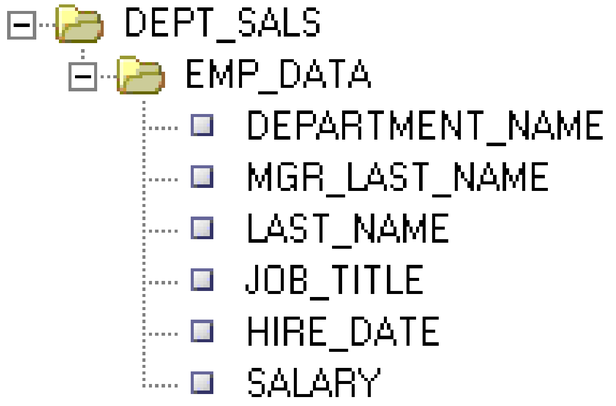


Best Practices for XML Data
Data Extraction & Processing – Data Template

- Join & structure data from different sources
- For large data sets
  - Push expensive joins to the database
  - Generate XML data optimized for document generation
    - Sort data (preferably at the source)
    - Structure data
    - Pre-calculate group-level and report-level aggregate functions
  - Take advantage of before report triggers

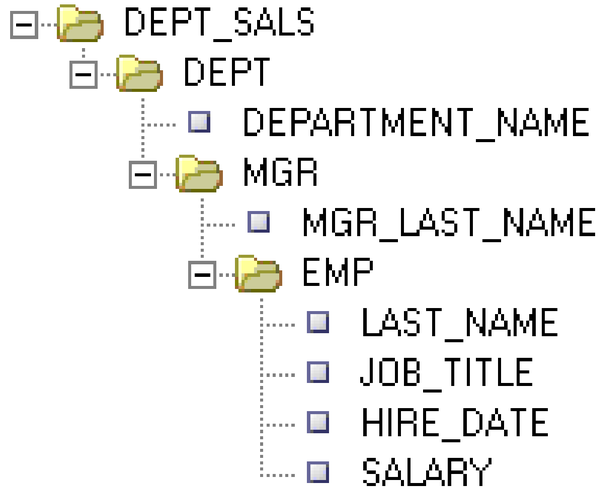
Best Practices for XML Data
Data Extraction & Processing – Flat XML

```
– <DEPT_SALS>
  – <EMP_DATA>
      <DEPARTMENT_NAME>Accounting</DEPARTMENT_NAME>
      <MGR_LAST_NAME>Kochhar</MGR_LAST_NAME>
      <LAST_NAME>Higgins</LAST_NAME>
      <JOB_TITLE>Accounting Manager</JOB_TITLE>
      <HIRE_DATE>1994-06-07T00:00:00.000-07:00</HIRE_DATE>
      <SALARY>12000</SALARY>
    </EMP_DATA>
  – <EMP_DATA>
      <DEPARTMENT_NAME>Accounting</DEPARTMENT_NAME>
      <MGR_LAST_NAME>Higgins</MGR_LAST_NAME>
      <LAST_NAME>Gietz</LAST_NAME>
      <JOB_TITLE>Public Accountant</JOB_TITLE>
      <HIRE_DATE>1994-06-07T00:00:00.000-07:00</HIRE_DATE>
      <SALARY>8300</SALARY>
    </EMP_DATA>
```



Best Practices for XML Data
Data Extraction & Processing – Hierarchical XML

```
– <DEPT_SALS>
  – <DEPT>
      <DEPARTMENT_NAME>Accounting</DEPARTMENT_NAME>
    – <MGR>
        <MGR_LAST_NAME>Higgins</MGR_LAST_NAME>
      – <EMP>
          <LAST_NAME>Gietz</LAST_NAME>
          <JOB_TITLE>Public Accountant</JOB_TITLE>
          <HIRE_DATE>1994-06-07T00:00:00.000-07:00</HIRE_DATE>
          <SALARY>8300</SALARY>
        </EMP>
      </MGR>
    – <MGR>
        <MGR_LAST_NAME>Kochhar</MGR_LAST_NAME>
      – <EMP>
          <LAST_NAME>Higgins</LAST_NAME>
          <JOB_TITLE>Accounting Manager</JOB_TITLE>
          <HIRE_DATE>1994-06-07T00:00:00.000-07:00</HIRE_DATE>
          <SALARY>12000</SALARY>
        </EMP>
      </MGR>
    </DEPT>
  – <DEPT>
```



Best Practices for XML Data
Data Extraction & Processing – Element Naming

- Avoid re-using the same name for element names. Sometimes this make sense – for example, `CITY` under `SHIPPING_ADDRESS` and `BILLING_ADDRESS`. But in many cases you are better off using different names e.g. `S_CITY` and `B_CITY`

- Use short element names to reduce XML file size

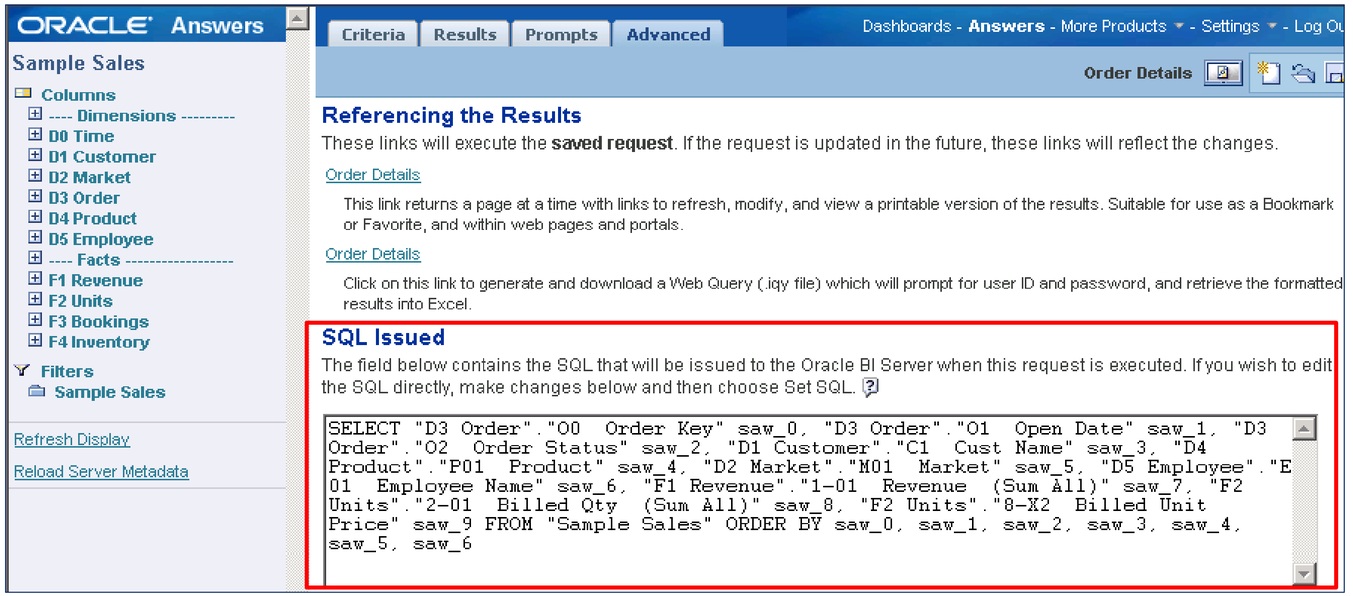
Best Practices for XML Data
NULL Elements & XML Attributes

- Avoid generating empty tags in the XML
  - Missing fields **do not** cause an issue with XSL – for example a print statement will just return an empty string
  - Aggregation over fields that contain empty tags **will fail** with a “not a number” exception

- Avoid using XML attributes, they are supported but are more complicated to use in the Template Builder for Word

Data Extraction & Processing
OBIEE Data Source

- If possible use SQL/ JDBC against BI Server instead of using an Answers Request as a data source
  - Tip: Copy the SQL in an Answers Request and paste into the Query Builder text area



BI Publisher 10g Best Practices
Agenda

- Oracle BI Publisher Enterprise Overview
- System Configuration Guidelines
- BI Publisher Server Configuration Options
- Report Design Best Practices
  - Data Modeling Guidelines
  - Template Design Guidelines
- Q & A

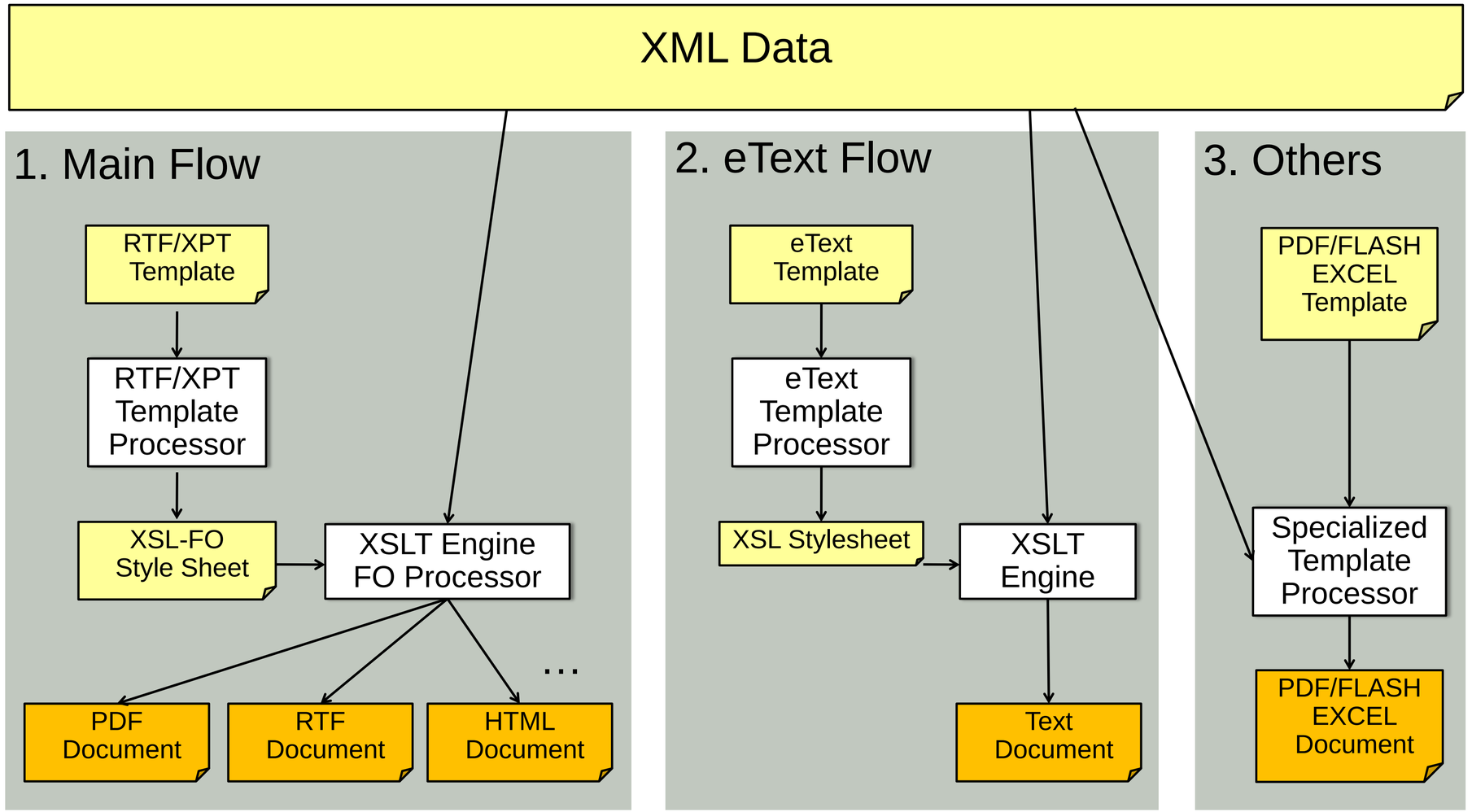
Document Formatting and Layout – What Template Type should I use? (1 of 2)

RTF Templates	•RTF marked up with XSL code or BI Publisher simplified XSL •Widest range of output (PDF, HTML, Excel, PPT, RTF, MHTML) •Greatest flexibility and functionality •Easy to create w/ BI Publisher Template Builder Add-in to Word
PDF Forms	•PDF forms with XML elements mapped to form fields. •Great to use with government forms •PDF output only •Create in Acrobat Professional
e-Text Templates	•RTF with table of statements to place fields and separators •Great for character delimited or fixed position docs (EFT & EDI) •Text output only •Create in any RTF editor (e.g. Word, OpenOffice)

Document Formatting and Layout
What Template Type should I use? (2 of 2)

Excel Templates	•Excel w/ data mapped to named cells & Excel and XSL formatting •Great for formatted, true Excel output and burst over sheets •Excel XLS output only •Create directly in Excel, good flexibility
Excel Analyzer Templates	•Auto generated EXCEL puts data in tabular format in sheet •Great to get data into Excel & Able to refresh data w/ params •Excel output only
Flash Templates	•SWF files embeded with BI Publisher data fields •Great for self contained, nearly unlimited interactive output •Create in Adobe Flex Builder
XSL Style Sheets	•Allows for third party tools and legacy solutions

Oracle BI Publisher Enterprise – Document Formatting Process



RTF Template Best Practices – Use Word Tables

- Use tables to control precisely where field data will be placed in the report

Grp:Supplier

Supplier	Supplier 1				
Address	1 Long Avenue				
Invoice Number	**Invoice Date**	**Currency**	**GL Date**	**Entered Amount**	**Accounted Amount**
Grp:Invoice1134922	01-Jan-2007	USD	01-Jan-2007	$100.00	$100.00End Invoice
			Total for Supplier: Supplier 1	$100.00	$100.00

End Supplier

Invoice Tax Summary

Tax Code	Entered Amt	Accounted Amt
Grp:TaxVAT 17.5%	$100.00	$100.00End Tax

Report Total	$100.00	$100.00

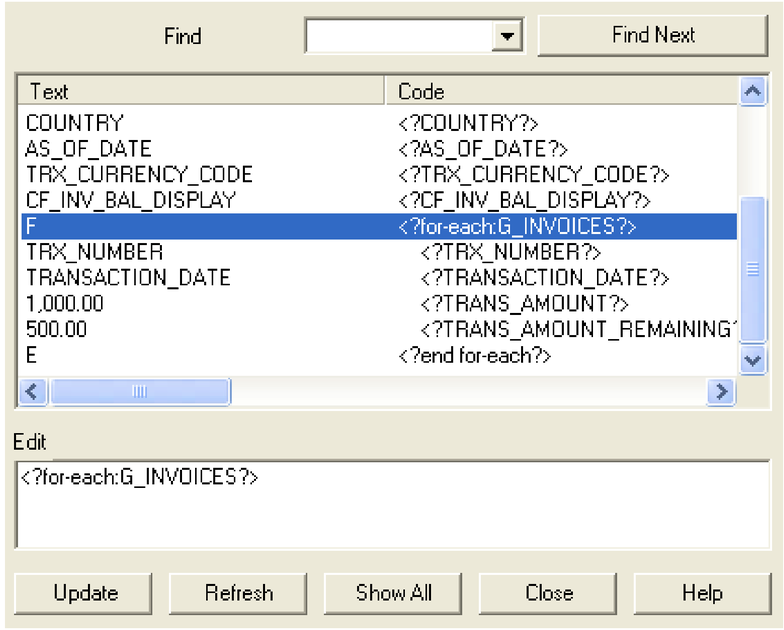
RTF Template Best Practices – Why Use Form Fields

- Keep the template clean
- Supported by the Template Builder *Field Browser*
- Can be colored or hidden to help understand the structure
- Caveat: Word header & footer don't allow form fields

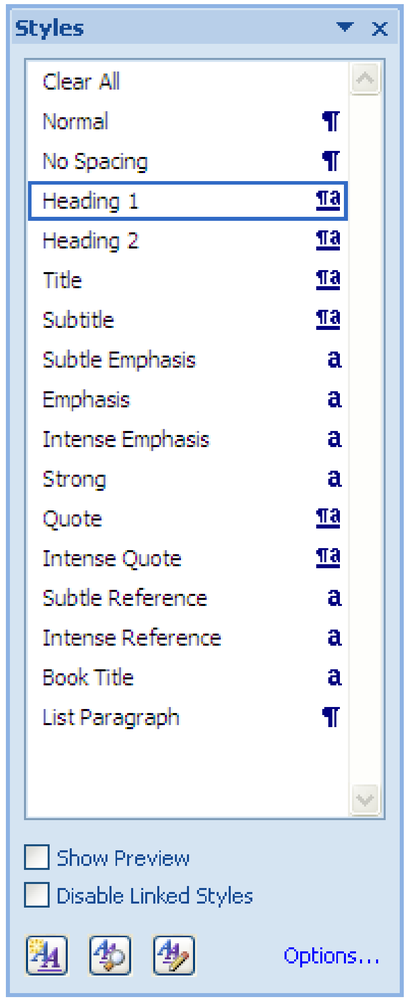
Invoice Number	Date
<?for-each:G_INVOICES?> TRX_NUMBER	TRANSACTION_DATE

Invoice Number	Date
F TRX_NUMBER	TRANSACTION_DATE

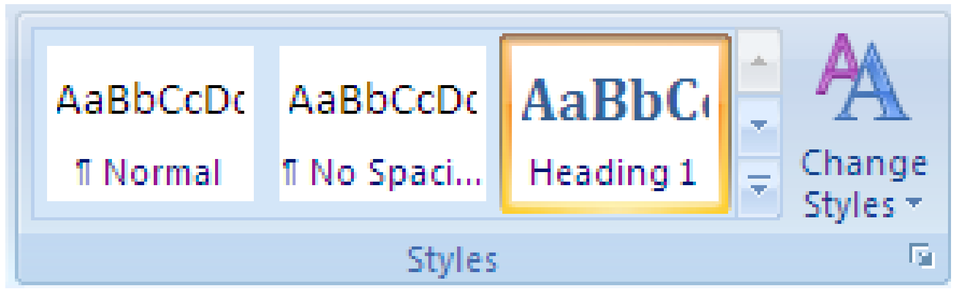
Please confirm the above information.



RTF Template Best Practices – Use Word Styles



- Achieve consistency in the template and between templates
- In 11g use Style Templates to control the template styles



RTF Template Best Practices – Don’t overcomplicate your template

- Keep it easy to understand, debug and maintain
- In general better to have different business documents in different templates
- Try to limit the logic in templates to simple if or loop statements
- Use sub templates to simplify documents if necessary
- Many calculations are better performed in the data template

Document Formatting and Layout
Sub Templates: Introduction

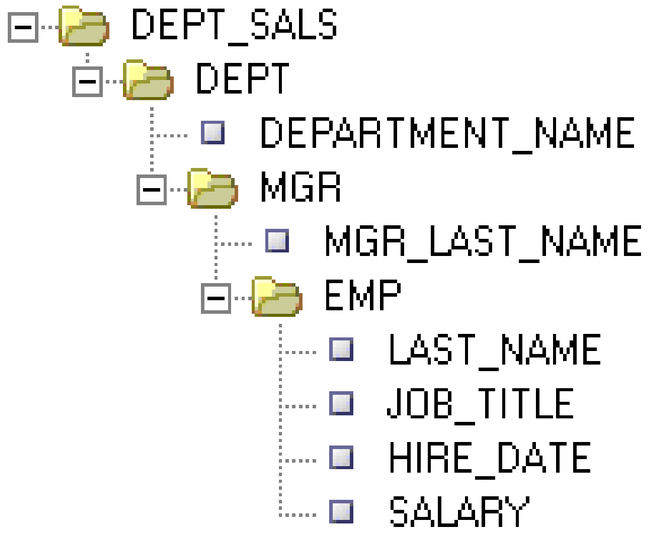
- What is a sub template?
  - A document that contains layout and or code that can be defined once and used multiple times
  - Multiple functions (called “templates” in XSL) or text segments can be defined in a single sub template
- Sub template types
  - XSL sub templates for code re-use or separation
  - RTF sub templates for layout re-use
  - Caveat: Use of RTF sub templates makes it hard to understand the template structure

Document Formatting and Layout
Sub Templates: Use Cases

- RTF sub templates
  - Re-use common layouts such as header/footer
  - Internationalized address block
- XSL sub templates
  - Transformations for complex chart requirements
  - Dynamically apply formatting to a portion of data (e.g. sub scripting / super scripting / chemical formula)
  - Print formatted XHTML data – convert to FO using sub template (as used in Oracle Contracts)

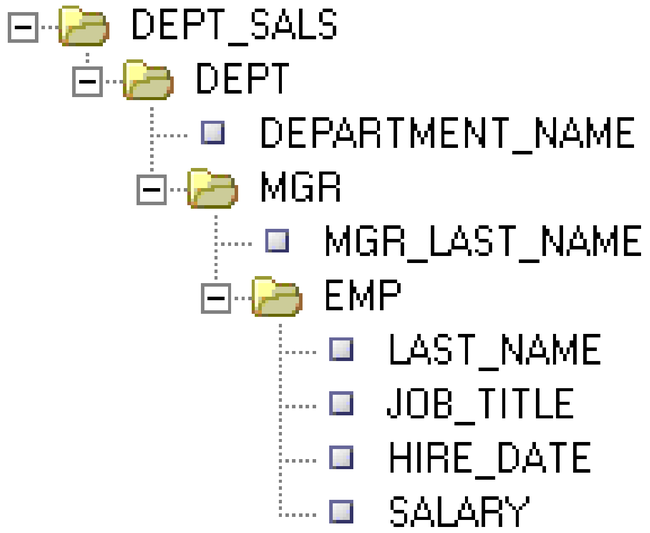
RTF Template Best Practices
Performance Considerations: XPATH Concepts

- BI Publisher uses XPATH to access data elements
- DEPARTMENT_NAME is inserted as <?DEPARTMENT_NAME?>
- <?DEPARTMENT_NAME?> is translated to the XPATH .//DEPARTMENT_NAME
- .//DEPARTMENT_NAME searches for DEPARTMENT_NAME in the complete sub-tree starting from current context



RTF Template Best Practices
Performance Considerations: XPATH Tuning

- For small documents the search time is negligible
- Large documents may not fit into memory and the search may require disk access
- For large datasets use the full relative path
- Instead of `<?for-each: DEPT?>` use `<?for-each:/DEPT_SALS/DEPT?>`
- Instead of `<?DEPARTMENT_NAME?>` use `<?./DEPARTMENT_NAME?>`

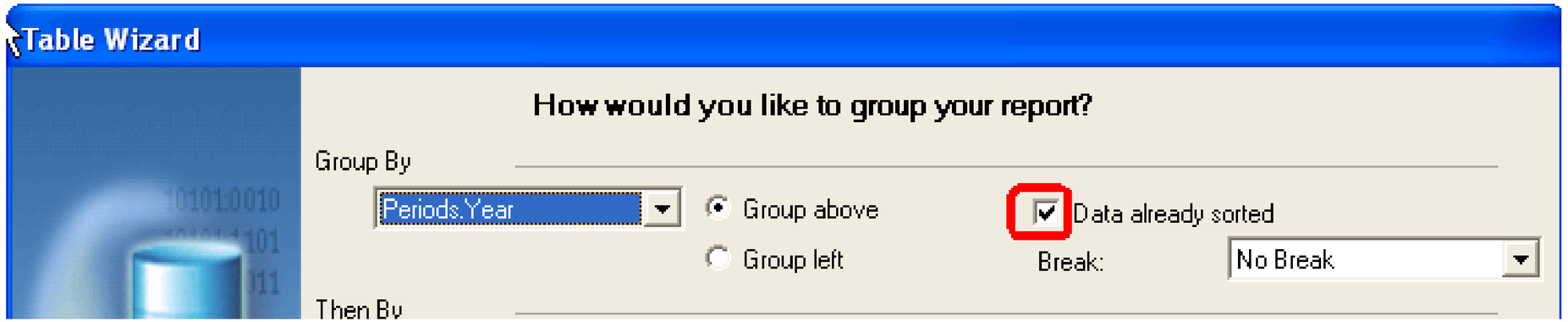


RTF Template Best Practices
Performance Considerations: Tuning

- XPATH Tuning
  - Start with the outer loops and outer most data access
  - Reducing full tree searches will provide bigger improvements than optimizing access in a small subset of the data
  - Fixing the XPATH in a single for-each loop may be all the performance tuning you need to do
- Tables
  - Large tables that span hundreds of pages consume considerable server resources. If possible group the data and create a table inside each grouping

RTF Template Best Practices – Performance Considerations: Grouping

- Better to Sort data in the data model
- Group By <?for-each-group?> will force the XSL engine to sort data
- Checking “Data already sorted” option in the Table Wizard will not re-sort data (good thing)



RTF Template Best Practices
Summary

- Use tables to place data fields precisely
- Use form fields
- Use word styles
- Don’t overcomplicate your templates
- Use sub templates for re-use and complex code
- Performance optimization
  - Optimize the XPATH
  - Focus optimization on outer loops
  - Sort in the data base and use “group-adjacent” in the template

BI Publisher 10g Best Practices
Agenda

- Oracle BI Publisher Enterprise Overview
- System Configuration Guidelines
- BI Publisher Server Configuration Options
- Report Design Best Practices
  - Data Modeling Guidelines
  - Template Design Guidelines
- Q & A

Resources

- Certification matrix http://download.oracle.com/docs/cd/E12844_01/doc/bip.1013/e12692/toc.htm
- High availability white paper www.oracle.com/technology/products/xml-publisher/docs/BIP_HA.pdf
- Sizing Spreadsheet for BI Publisher Enterprise 10g
  Metalink Note - 948841.1
- Sub-templates white paper http://www.oracle.com/technology/products/xml-publisher/docs/BIP-SubTemplate.pdf
- QUARTZ Scheduling (clustering) wiki page
  http://wiki.opensymphony.com/display/QRTZ1/ConfigJDBCJobStoreClustering
- CPU Ratings
  http://www.spec.org/cpu2006/results/cpu2006.html#SPECint_rate

For More Information

search.oracle.com

Oracle BI Publisher 

or

http://www.oracle.com/technology/products/xml-publisher/index.html


ORACLE®