

Oracle BI Publisher Best Practices for SaaS Environments

ORACLE WHITE PAPER | MAY 2016






Table of Contents

Introduction	1
Report Design	1
Report Creation Steps	1
Caching	6
Layout Design	8
Choose the right Template Type	8
BI Publisher Template	9
RTF Template	9
Use tables for pixel perfect positioning of data	9
Use Form Fields	10
Use Style Templates	11
Do not overcomplicate your Layout Template Design	11
Use XPATH	11
XPath Tuning	12
Layout Tables	12
Sorting and Grouping	12
Handling Large Outputs in PDF format	12
Data Model Design	14
Data Model Creation	14
Data Model Properties	14
What Dataset to use?	16
Leverage Database Capabilities	17



Data Caching	17
Triggers	17
Before Data Triggers	17
After Data Triggers	17
Schedule Triggers	17
Best Practices for SQL Dataset	17
Only Return the Data You Need	17
Use Column Aliases to Shorten XML File Length	18
Avoid Using Group Filters in Data Model. Instead enhance Your Query	18
Avoid PL/SQL function Calls in WHERE Clauses	18
Avoid Use of the System Dual Table	19
Avoid PL/SQL Calls at the Element Level	19
Avoid Including Multiple Data Sets	20
Avoid Nested Data Sets	20
Avoid In-Line Queries (as summary columns)	21
Avoid Excessive Parameter Bind Values	21
The number of columns selected in SQL should not exceed 100	22
Sorting should be done in the SQL, not during report formatting	22
Avoid having WHERE clause on large tables with only NOT IN or <>	22
SQL Query Tuning	22
Generate Explain Plan	23
Administration Page Configurations	26
Runtime Configurations	26





Memory Guard	27
Data Model	29
Scheduling & Delivery	30
How to decide whether to run a Report Online/Offline (Schedule)	30
Scheduler Database Management	31
Scheduling Tips	31
Report Customization	31
Scheduling a Custom Report	34
Viewing a Custom Report	34
Securing Custom Report	34
Troubleshooting Tips	34
Out of Memory Issue	34
Unusually slow throughput from RTF template	35
What logs to capture when report performs poorly on Server	35
How to debug or troubleshoot an RTF template design?	35
Stuck Job Scenario	35
Schedule Job Diagnostics	36
Process to initiate critical Administration changes on BI Publisher Server	36



Introduction

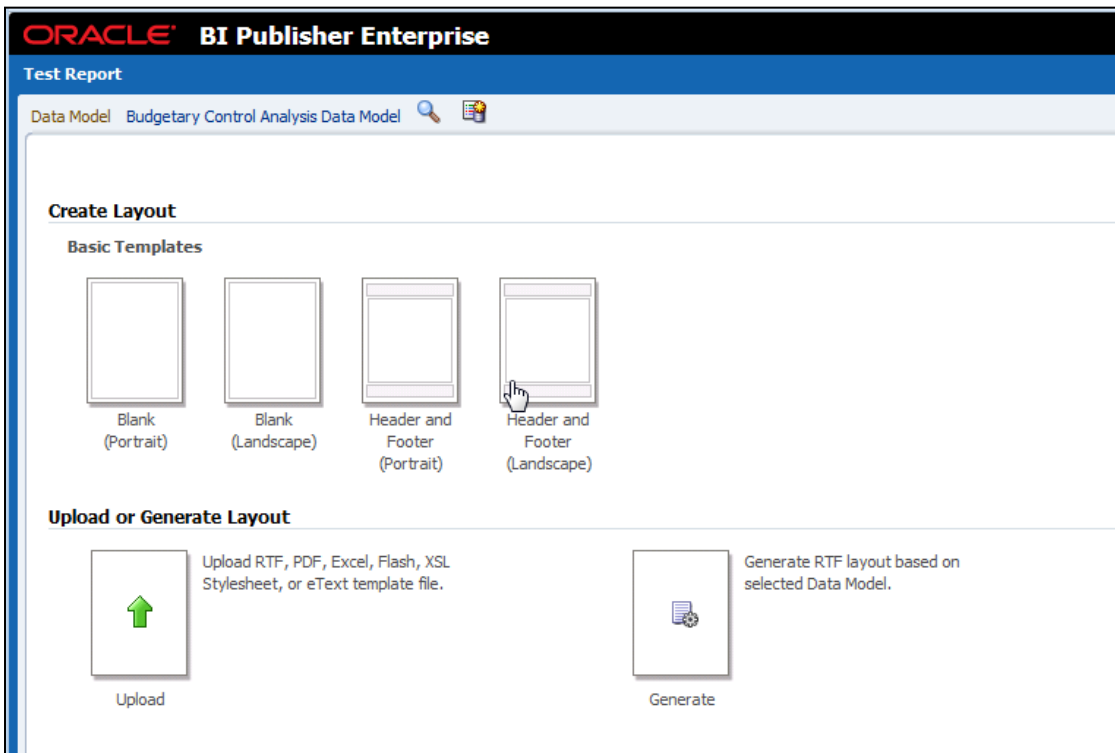
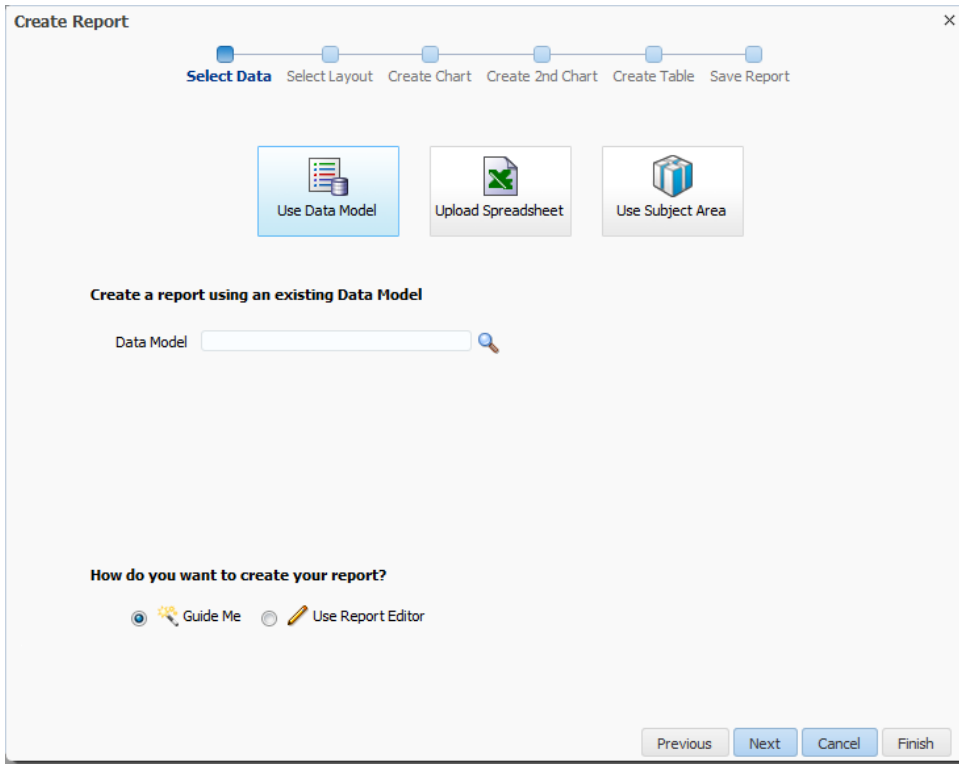
Oracle Business Intelligence Publisher is an enterprise reporting solution for creating highly formatted, pixel perfect reports. It offers a single solution environment to author, manage and deliver a variety of business documents. Today Oracle BI Publisher is available to customers with Oracle BI Suite Plus and Oracle BI Foundation, while it is also available standalone as Oracle BI Publisher Enterprise. Today, over 90 Oracle products are integrated with Oracle BI Publisher for their enterprise reporting requirements. Often these reports are required to process large amount of data and therefore it is very important that all products and applications and even the end customers should know the best practices of Oracle BI Publisher. This white paper provides a detailed list of best practices that are recommended by Oracle while generating reports using Oracle BI Publisher in a Software-as-a-Service (SaaS) environment.

Report Design

Report Creation Steps

A report author can create a report in Oracle BI Publisher following different paths. However, it is always recommended to create a data model before you work on the layout design. The steps can be defined as:

1. Create a Data Model (Refer to Data model section)
2. Create a Report and Layout(s)
 - a. Create a new report from Homepage “Create” action menu or from Global Header Menu, map to an existing data model and then add a layout

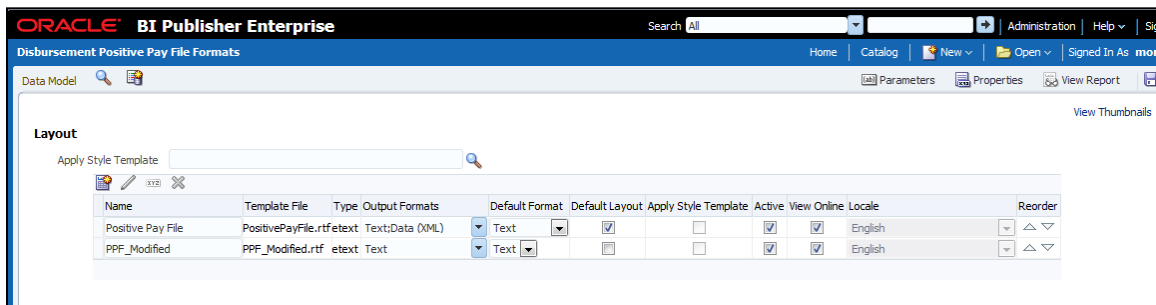
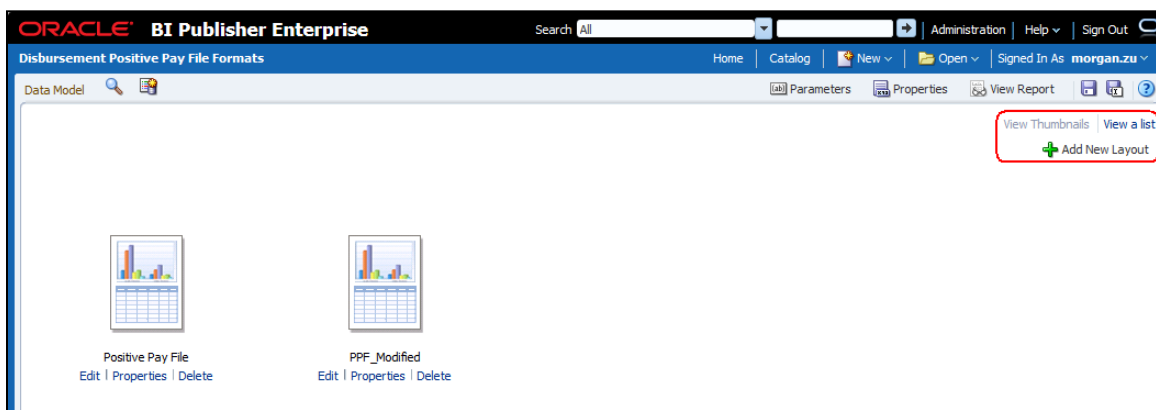


In this report editor page you can either create a new layout using Layout Editor or you can upload RTF, PDF, Excel, XSL or eText template file. Refer to Layout Design section of this document for the layout design related best practices.

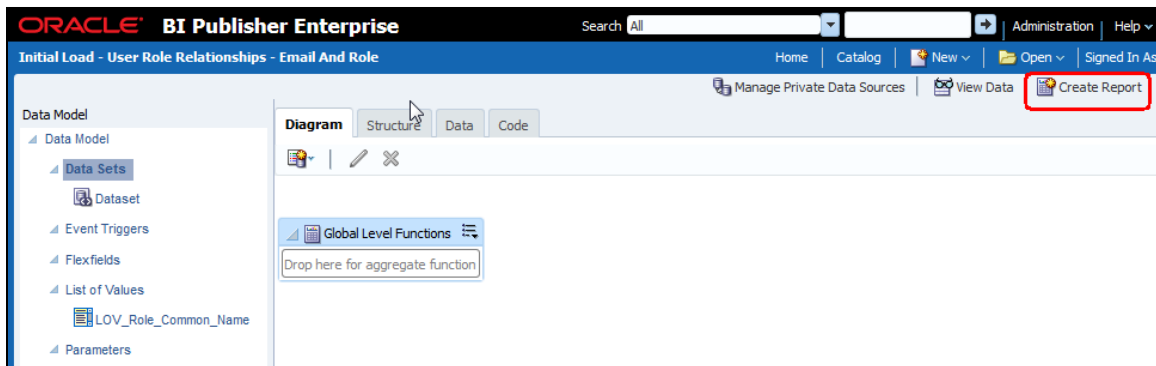
Important Note

Generate Layout option in Report Editor will automatically create a simple table-based RTF layout that includes all the fields in the data model. This will enable you to create a tabular report quickly but will require formatting adjustments. **This is not recommended for Production Layout creation.**

In the Report Editor, use the List View to set output formats, Default output format, Locale, etc.



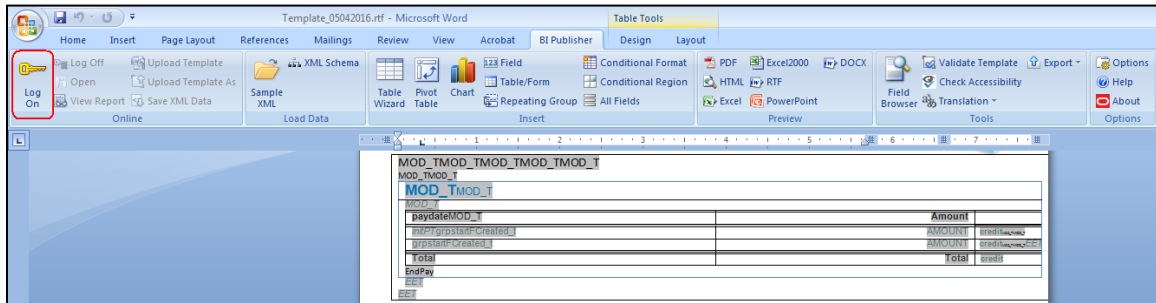
b. From Data Model Editor



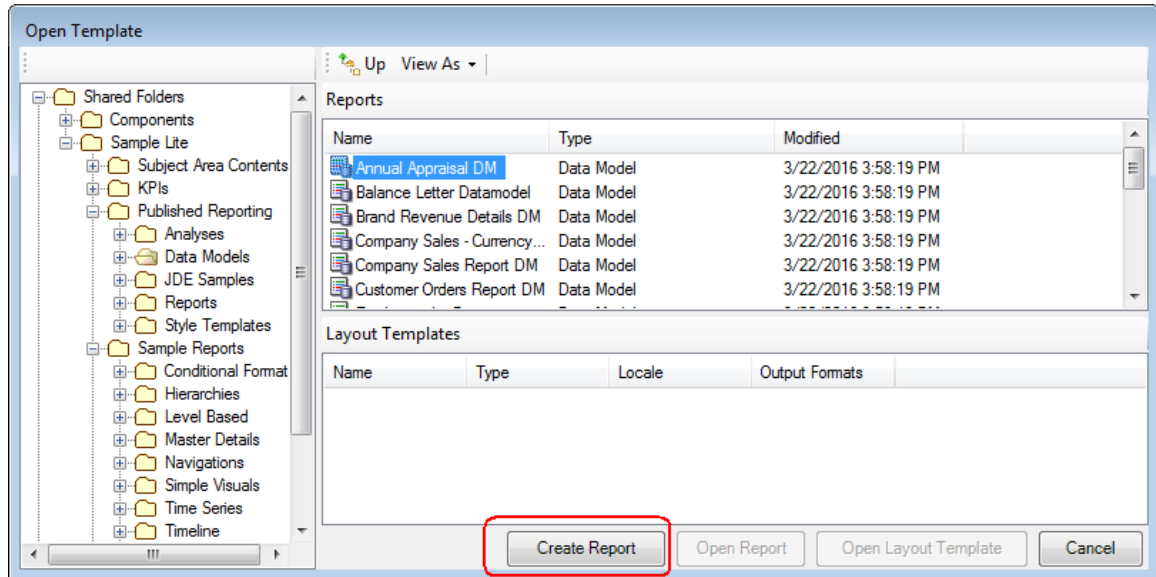
The steps to create report will follow similar step as in a. above

c. From Template Builder

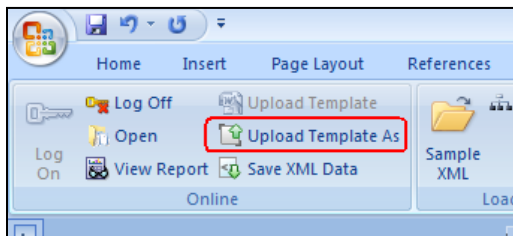
You can login from RTF or Excel Template Builder



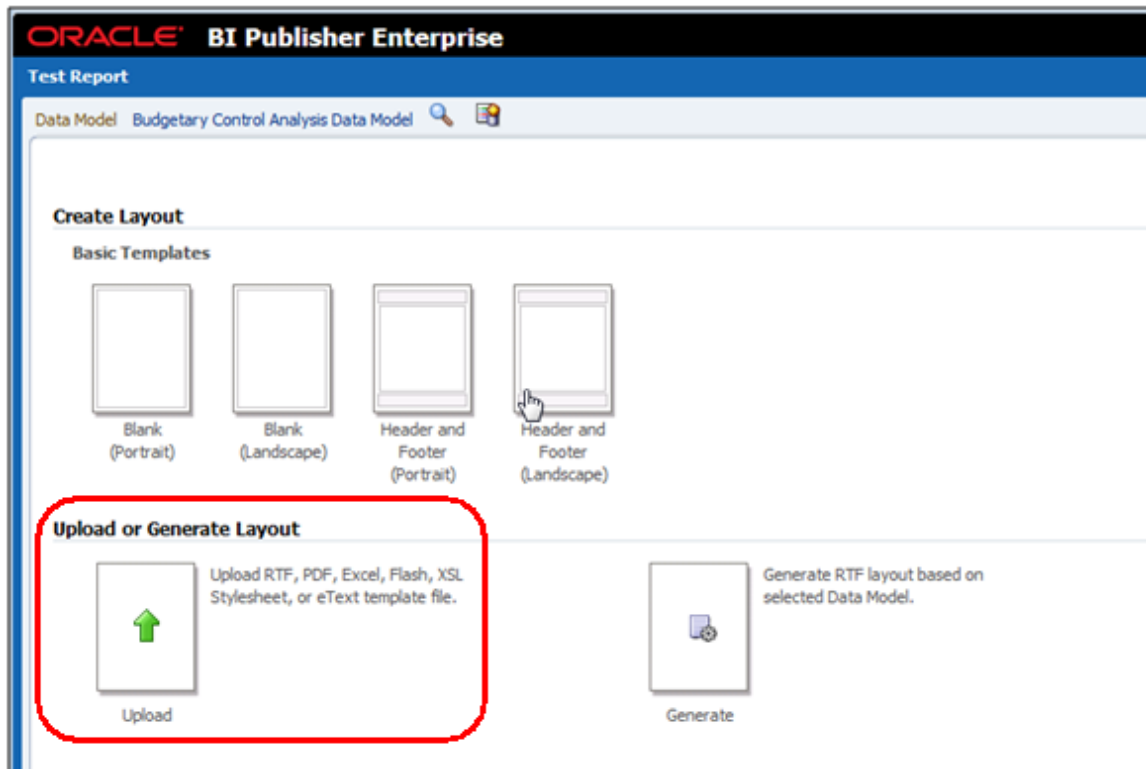
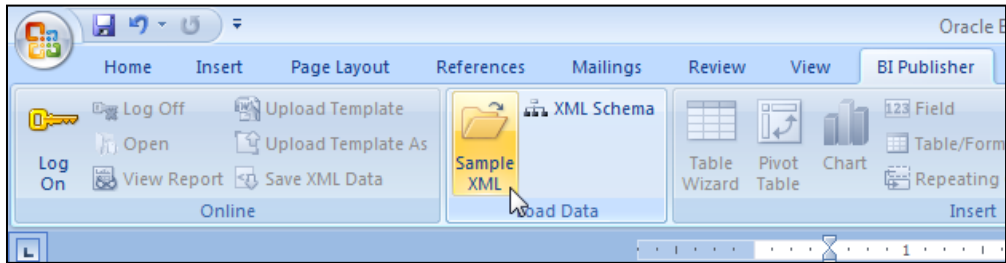
Select the data model from catalog and Create a new Report.



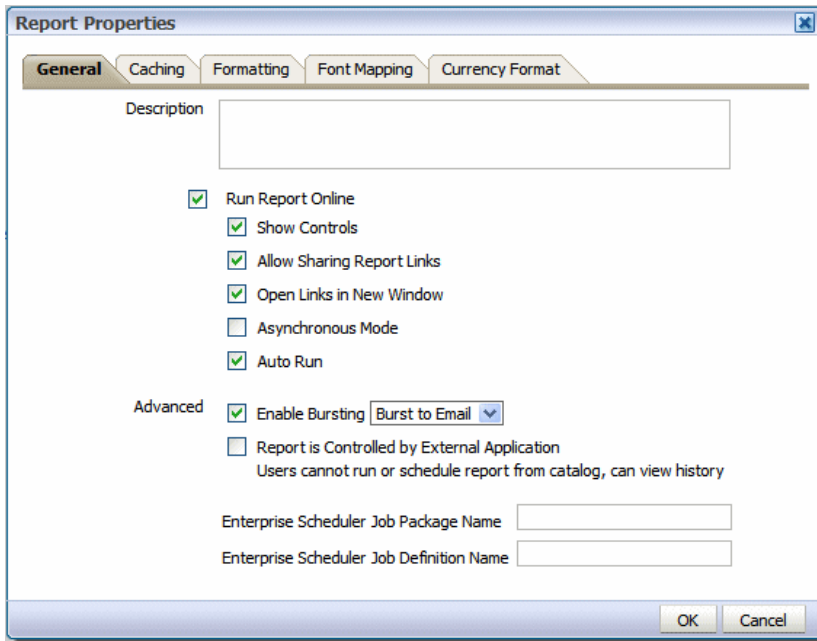
Once the report is created, you can work on the layout in Template builder and upload the template to server using "Upload Template As".



Alternatively, you can use a sample XML data, design the template without connecting to BI Publisher server and then upload the layout template from Report Editor Page.



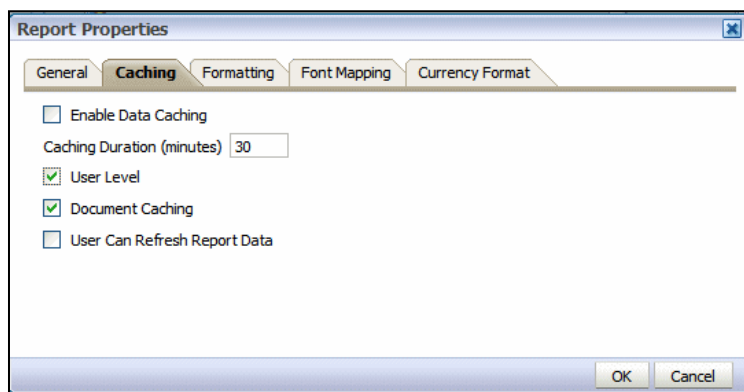
3. Configure Report Properties to control the report execution behavior –Schedule Only / Run Online and Schedule



It is recommended

- For long running reports, Run Report Online option should be unchecked so that the report will not be available to View online. Such reports can only be scheduled.
- In case long running reports are included as “Run Report Online” then **Auto Run** should be disabled to avoid running the report with incorrect parameters.
- For reports to be run through ESS only, enable “**Report is Controlled by External Application**”. The report cannot be run or scheduled from BI Publisher Catalog and can only be submitted by ESS scheduling user interface.

Caching



- **Enable Data Caching**

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

- **Enable Document Caching**

- Better performance
- Consumes more Temp storage space
- Scheduled reports do not use document cache

- **User Level**

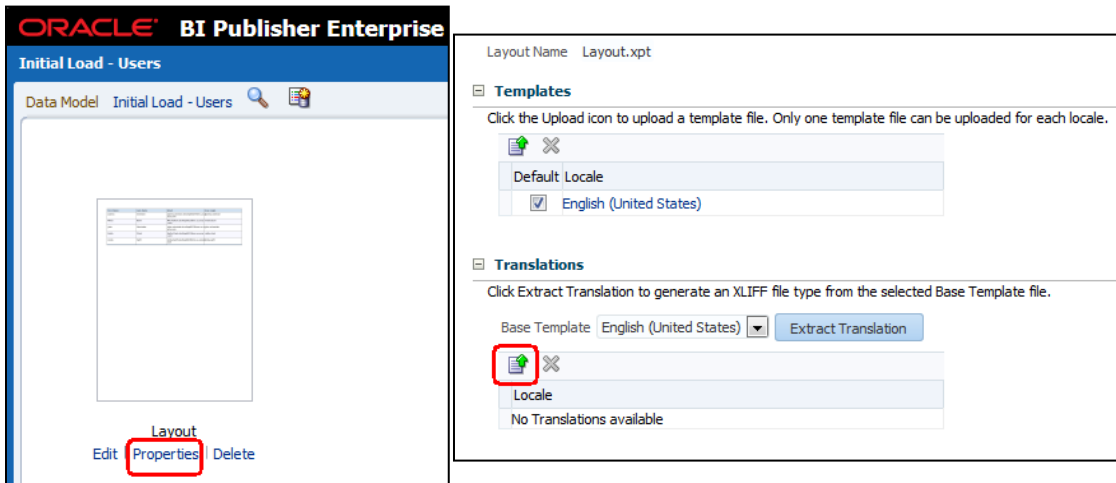
- Disable for better performance if data can be shared across users

- **Cache Duration**

- Default is 30 minutes. Once the time limit has expired, the next request for the same report generates a fresh data set.

4. Translation

If the report requires supporting multiple languages, then you can create translation files and upload from the Layout Properties.



Layout Design

Choose the right Template Type

Oracle BI Publisher offers a variety of layout design template. The table below highlights the scenarios when a specific layout template is recommended.

BI Publisher Template	<ul style="list-style-type: none"> • Web based Layout Editor – no client installation required • Best for Management Report – WYSIWIG experience • Interactive Output • Wide range of output (PDF, HTML, Excel, PPT, RTF, MHTML) • Not recommended for data size above 50 MB
RTF Template	<ul style="list-style-type: none"> • Easy to create using Template Builder MS Word Add-in • Extensible to use XSL code syntax within BI Publisher Code Syntax • Wide range of output (PDF, HTML, Excel, PPT, RTF, MHTML) • Complex layout design, formulae, calculations • For Barcodes, Packing Slips, Invoices, Checks, Complex Charts
Excel Template	<ul style="list-style-type: none"> • Excel format is the only output expected • The data in excel output expects to maintain the cell formatting for number and date fields • The column width in excel output should remain exactly as designed in template, creating a pixel perfect printable area in excel output • Use of native excel charts, functions, macros etc • Handle large number of columns in excel output • Not recommended for data size above 50 MB
PDF Template	<ul style="list-style-type: none"> • PDF forms with XML elements mapped to form fields. • Directly use Government Forms as Template • PDF output only • Use Acrobat Professional to Create/Edit Template • Not recommended for data size above 10 MB
E-Text Template	<ul style="list-style-type: none"> • Text output only – for electronic communication • Great for character delimited or fixed position docs (EFT & EDI)

	<ul style="list-style-type: none"> • RTF with table of statements to place fields and separators
Flash Template	<ul style="list-style-type: none"> • SWF files with BI Publisher data – for sophisticated interactivity • Create in Adobe Flex Builder
XSL Template	<ul style="list-style-type: none"> • Allows for third party tools and legacy solutions

Note:

- Excel templates use excel 97-2003 (.xls) workbook binary format and therefore has a limitation of 65K rows. A workaround to process more than 65K rows is to split the data into multiple sheets. Alternatively, we can generate a CSV (comma separated value) output to process more than 65K rows.
- There is no support for excel 2007 + (.xlsx) workbook binary format today for excel templates.
- Excel output can be generated from RTF, Excel and XSL template. Excel output with large amount of data consumes a large amount of memory irrespective of template type. It is not recommended to create excel output for reports with more than 50 MB of data.

BI Publisher Template

- Use Layout Grids to position components – similar to tables in RTF Templates
- Layout Grids can be nested
- Layout Grid Cell can be joined
- The minimum height of rows can be fixed
- DO NOT put **LARGE** tables or pivot tables into Layout Grids as they can affect Performance of the report.
- Interactivity works best for flat tables or simple nested master-detail data sets
- Unrelated queries or groups cannot interact
 - Disable interactions for unrelated queries in the **Configure Events** Dialog

RTF Template

Use tables for pixel perfect positioning of data

Use tables to control precisely where field data will appear in the generated document. Use fields in inline text only for form letters and where text with inline data fields is appropriate like form letters. Don't try to control the precise placement of fields with spaces or tabs.

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

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
TRX_NUMBER	TRANSACTION_DATE

Find Find Next

Text	Code
COUNTRY	<?COUNTRY?>
AS_OF_DATE	<?AS_OF_DATE?>
TRX_CURRENCY_CODE	<?TRX_CURRENCY_CODE?>
CF_INV_BAL_DISPLAY	<?CF_INV_BAL_DISPLAY?>
<?for-each:G_INVOICES?>	<?for-each:G_INVOICES?>
TRX_NUMBER	<?TRX_NUMBER?>
TRANSACTION_DATE	<?TRANSACTION_DATE?>
1,000.00	<?TRANS_AMOUNT?>
500.00	<?TRANS_AMOUNT_REMAINING?>
E	<?end for-each?>

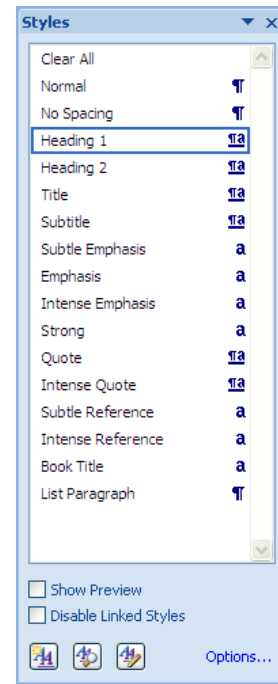
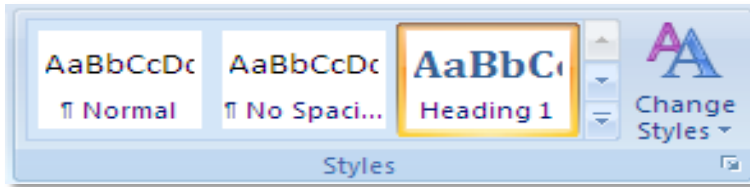
Edit

<?for-each:G_INVOICES?>

Update Refresh Show All Close Help

Use Style Templates

- Achieve consistency in the template and between templates

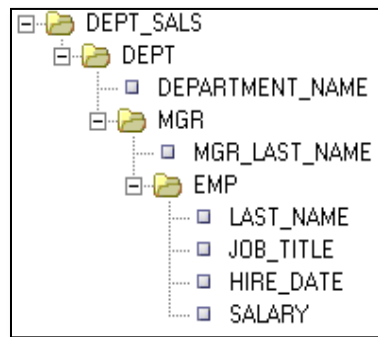


Do not overcomplicate your Layout Template Design

- Keep it easy to understand, debug and maintain
- 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 and reuse content
- Many calculations are better performed in the data model

Use XPATH

- BI Publisher uses XPATH to access data elements
- DEPARTMENT_NAME is inserted in the template 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



XPATH Tuning

- Start with the outer loops and outer most data access.

For example, if there is a nested for-each loop, the outer loop will access the complete XML tree while the inner loop will only traverse one branch of data and therefore 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.

Layout Tables

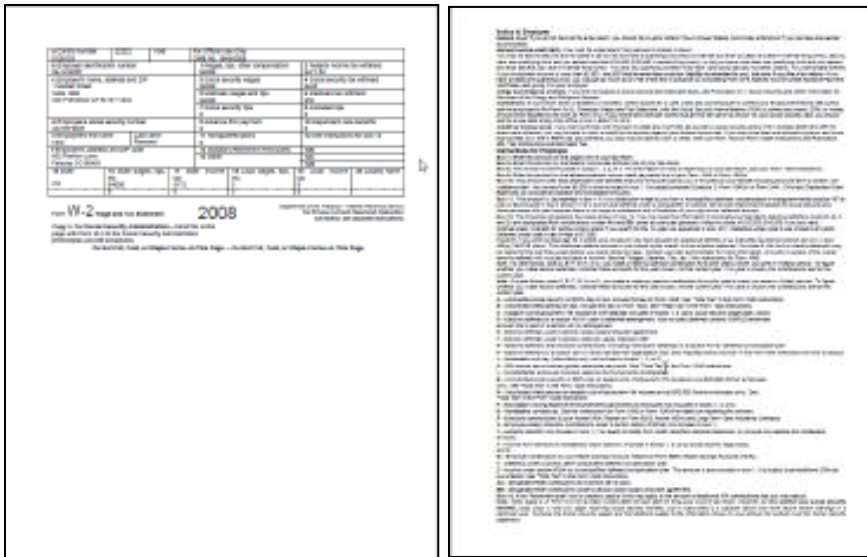
- LARGE tables that span hundreds of pages consume considerable server resources. If possible group the data and create a table inside each grouping.
- Do not nest LARGE table inside another table.

Sorting and Grouping

- Recommended to Sort data in the data model. Sorting in RTF template should be avoided.
- Grouping of data by use of syntax `<?for-each-group?>` will sort data by default in XSL engine which is efficient.
- Checking “Data already sorted” option in the Table Wizard of Word Template Builder will not re-sort data which will help improve performance.

Handling Large Outputs in PDF format

- Make use of “Reuse Static Content” for reports with alternate page with repeating fine prints



- Use zipped Output format when PDF output is extremely large to read or print as single document.

Layout

Apply Style Template

Name	Template File	Type	Output Formats	Default Format	Default Layout
ops salary	ops salary.rtf	rtf	<input checked="" type="checkbox"/> HTML;PDF;RTF;Excel <input checked="" type="checkbox"/> PowerPoint (*.pptx) <input type="checkbox"/> MHTML <input type="checkbox"/> PDF/A <input type="checkbox"/> PDF/X <input checked="" type="checkbox"/> Zipped PDFs <input type="checkbox"/> Formatted XML <input type="checkbox"/> Data (XML)	HTML	<input checked="" type="checkbox"/>

C:\Users\PKSHAR~1.ORA\AppData\Local\Temp\W2%20Report%20for%20Zipped%20Output_W2%20Form.zip\

File Edit View Favorites Tools Help

Add Extract Test

C:\Users\PKSHAR~1.ORA\AppData\Local\Temp\W2%20Rep

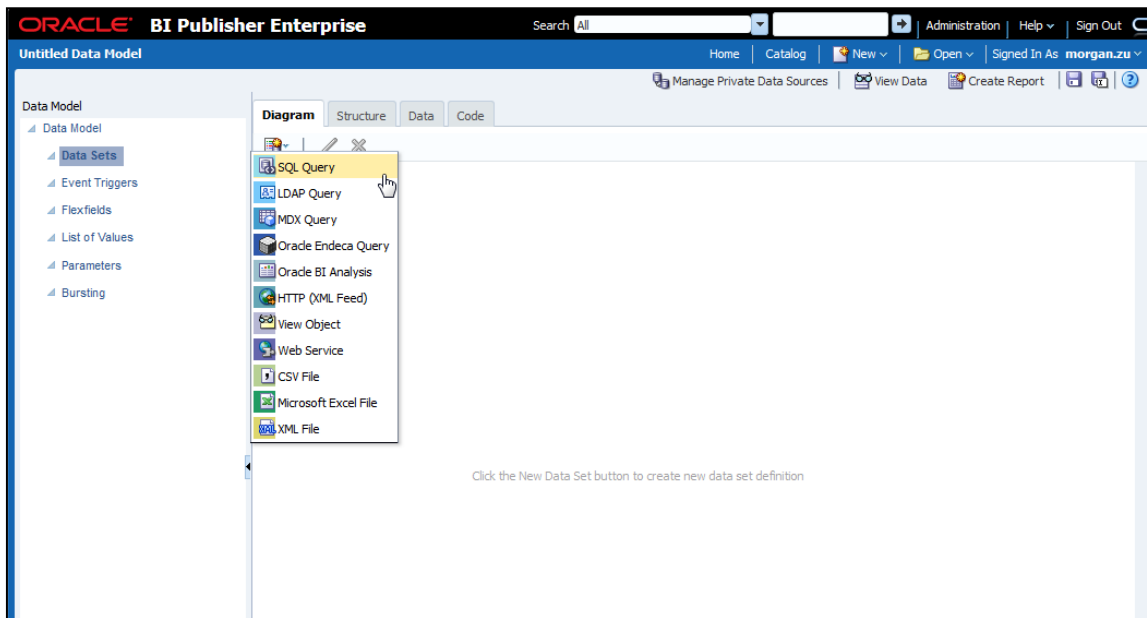
Name	From	To	In
	First Name, Last Name	First Name, Last Name	Document
Doc0001.pdf	Kelly, Rowland	Ellie, Smith	Doc0001.pdf
Doc0002.pdf	Mike, Simonz	Brandon, Espinoza	Doc0002.pdf
Doc0003.pdf	Joe, Thomas	Johnathan X Sir, Harris	Doc0003.pdf
Doc0004.pdf	Johnathan, Segel	Freeda, Nightingale	Doc0004.pdf

615 758	104 811	2014-09-29 05:06
2 935	407	2014-09-29 05:06
3 366	567	2014-09-29 05:06

Data Model Design

Data Model Creation

To Launch the Data Model Editor from Home Page, Click Data Model under Create Region or from Global Header Menu, click on New→ Data Model.



Data Model Properties


The Data Model properties can be seen as the first option in the Data Model Editor.

Oracle DB Default Package — Not applicable for SaaS

Database Fetch Size — Not recommended to edit in SaaS environment.

Query Time Out - applies to SQL query-based data models. If the SQL query is still processing when the time out value is met, the error "Failed to retrieve data xml." is returned. Enter a value in seconds. If you do not enter a value for this data model, the value defined in Administration page is used. The timeout defaults to 600 seconds.

IMPORTANT NOTE - WebLogic Server has a default time out of 600 seconds for each thread that spans for a request. When a query exceeds 600 seconds, WebLogic Server marks the thread as "Stuck". When the number of Stuck threads reaches 25, the server shuts down. To avoid this problem, ensure that your SQL execution time does not exceed the WebLogic Server thread time out.



Scalable Mode — Processing large data sets requires the use of large amounts of RAM. To prevent running out of memory, activate scalable mode for the data engine. In scalable mode, the data engine takes advantage of disk space when it processes the data. Setting this to ON will impact performance, but guard against out of memory errors. **For SaaS environments, Scalable Mode is managed by BI Publisher internally (except for interactive report and Data Model)**, and does not require any change by user.

Enable SQL Pruning - applies to Oracle Database queries only that use Standard SQL. SQL pruning enhances performance by fetching only the columns that are used in the report layout/template. Columns that are defined in the query but are not used in the report are not fetched. This improves query fetch size and reduces JDBC rowset memory. Oracle recommends enabling this property at individual data model level as this may give unexpected result if data is cached and if there are multiple layouts in a report accessing the same data model.

Note: Enable SQL Pruning is also a server-level property therefore by default the data model-level property is set to Instance Level to inherit the server or instance level setting. To turn SQL pruning on or off for this particular data model, select **On** or **Off** from the list.

Backup Data Source — If you have set up a backup database for this data source, select **Enable Backup Connection** to enable the option; then select it when you want BI Publisher to use the backup.

- To use the backup data source only when the primary is down, select **Switch to Backup Data Source when Primary Data Source is unavailable**. Note that when the primary data source is down, the data engine must wait for a response before switching to the backup.
- To always use the backup data source when executing this data model, select **Use Backup Data Source Only**. Using the backup database may enhance performance.

XML Output Options — These options define characteristics of the XML data structure.

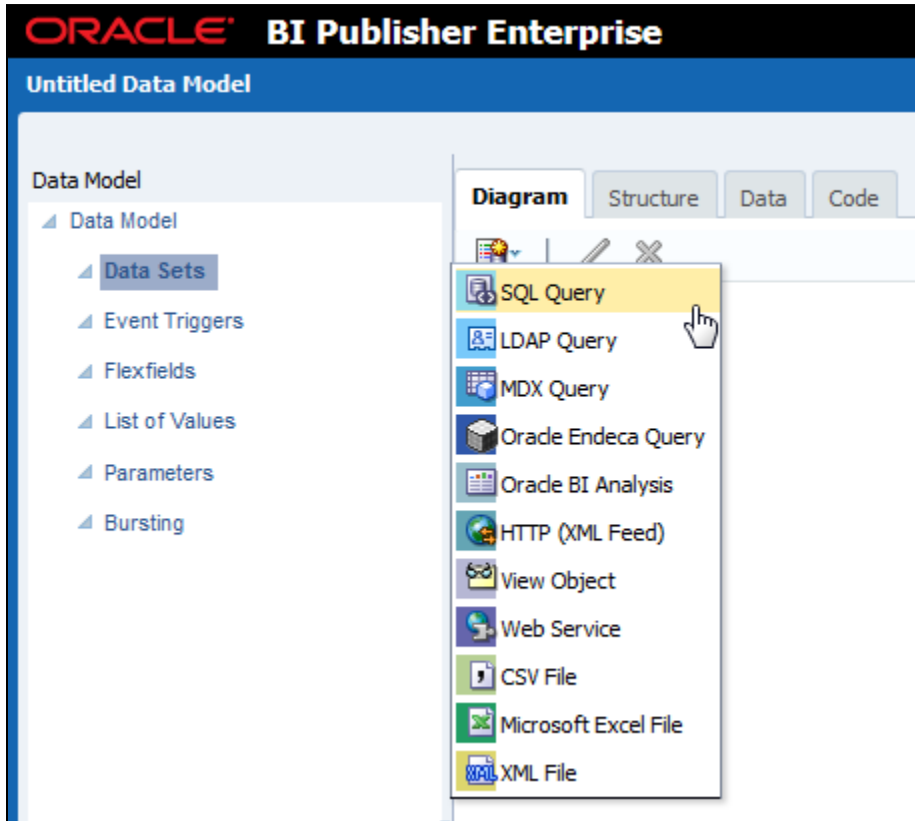
- **Include Parameter Tags** — If you define parameters for your data model, select this box to include the parameter values in the XML output file. Enable this option when you want to use the parameter value in the report.
- **Include Empty Tags for Null Elements** — Select this box to include elements with null values in your output XML data. When you include a null element, then a requested element that contains no data in your data source is included in your XML output as an empty XML tag as follows: <ELEMENT_ID>. For example, if the element MANAGER_ID contained no data and you chose to include null elements, it would appear in your data as follows: <MANAGER_ID />. If you do not select this option, no entry appears for MANAGER_ID.
- **Include Group List Tag** — (This property is for 10g backward compatibility and Oracle Report migration.) Select this box to include the rowset tags in your output XML data. If you include the group list tags, then the group list appears as another hierarchy within your data.
- **XML Tag Display** — Select whether to generate the XML data tags in upper case, in lower case, or to preserve the definition you supplied in the data structure.

IMPORTANT NOTE —

- Any changes to XML Output options can impact layouts that are built on the data model.
- **Include Empty Tags for Null Elements** is by **default unchecked**. Therefore, by default there will be no entry for any element that has null value. If your RTF template is not designed to handle the missing element, for example in case where dynamic cells are created in RTF template, we may see data displaying under incorrect columns. Therefore, we recommend enabling this feature to prevent misplaced data in report.


What Dataset to use?

BI Publisher supports a variety of data source types for creating data sets. These can be categorized into three general types:



- i. Data sets for which BI Publisher can retrieve metadata information from the source. For these data set types, the full range of data model editor functions is supported.
 - SQL Query
 - MDX Query
 - LDAP query
 - Microsoft Excel File
 - XML File
 - CSV
- ii. Data sets for which BI Publisher can retrieve column names and data type information from the data source but it cannot process or structure the data. For these data set types, only a subset of the full range of data model editor functions is supported.
 - Oracle BI Analysis
 - View Object
- iii. Data sets for which data is generated and structured at the source and no additional modifications can be applied by the data model editor
 - HTTP (XML Feed)
 - Web Service

SQL Query is the most recommended Data set in general. For Fusion Applications, it is recommended to use OTBI (i.e. BI Analysis) as data source to leverage the benefits of OTBI data security, performance optimizations, ease of use etc.



Leverage Database Capabilities

Leverage back-end system resources whenever possible

- Join data
- Filter & group data
- Perform expensive calculations & data transformations
- Sort data

Data Caching

Report Data and LOV caching can improve performance of a report significantly. The LOV caching should be considered when a report has multiple LOVs and without caching these LOVs may take few seconds to load.

LOV parameter caching is an option available in Data Model Editor, while Report Data cache option is available in Report Properties. Refer to Caching in Report Design best practices.

Triggers

Before Data Triggers

This is not recommended in SaaS environment.

After Data Triggers

This is not recommended in SaaS environment.

Schedule Triggers

- Data Model Designer creates Data Model with schedule trigger
- Triggers can be created and shared from a single data model
- Users create scheduled jobs and determine window of time to check for condition
- Reports execute when condition is true or are skipped

Best Practices for SQL Dataset

Only Return the Data You Need

Ensure that your query returns only the data you need for your reports. Returning excessive data risks OutOfMemory exceptions. For example, never simply return all columns as in:



SELECT * FROM EMPLOYEES;

Always avoid the use of *.

Three best practices for restricting the data returned are:

- i. Always select only the columns you need

For example:

```
SELECT DEPARTMENT_ID, DEPARTMENT_NAME FROM EMPLOYEES;
```

- ii. Use a WHERE clause and bind parameters whenever possible to restrict the returned data more precisely.

This example selects only the columns needed and only those that match the value of the parameter:

```
SELECT DEPARTMENT_ID, DEPARTMENT_NAME
```

```
FROM EMPLOYEES
```

```
WHERE DEPARTMENT_ID IN (:P_DEPT_ID)
```

- iii. The number of columns selected in SQL query but not used in report should not exceed 10 if SQL pruning is not enabled
- iv. Check all tables in SELECT list are properly joined to avoid Cartesian products

Use Column Aliases to Shorten XML File Length

The shorter the column name, the smaller the resulting XML file; the smaller the XML file the faster the system parses it. Shorten your column names using aliases to shorten I/O processing time and enhance report efficiency.

In this example, DEPARTMENT_ID is shortened to "id" and DEPARTMENT_NAME is shortened to "name":

```
SELECT DEPARTMENT_ID id, DEPARTMENT_NAME name FROM EMPLOYEES
```

```
WHERE DEPARTMENT_ID IN (:P_DEPT_ID)
```

Avoid Using Group Filters in Data Model. Instead enhance Your Query

Although the Data Model Group Filter feature enables you to remove records retrieved by your query, this process takes place in the middle tier, which is much less efficient than the database tier.

It is a better practice to remove unneeded records through your query using WHERE clause conditions instead.

Avoid PL/SQL function Calls in WHERE Clauses

PL/SQL function calls in the WHERE clause of the query can result in multiple executions. These function calls execute for each row found in the database that matches. Moreover, this construction requires PL/SQL to SQL context switching, which is inefficient.



As a best practice, avoid PL/SQL function calls in the WHERE clause; instead, join the base tables and add filters.

Avoid Use of the System Dual Table

Use of the system DUAL table for returning the sysdate or other constants is inefficient and should be avoided when not required.

For example, instead of:

```
SELECT DEPARTMENT_ID ID, (SELECT SYSDATE FROM DUAL) TODAYS_DATE FROM
DEPARTMENTS WHERE DEPARTMENT_ID IN (:P_DEPT_ID)
```

Consider:

```
SELECT DEPARTMENT_ID ID, SYSDATE TODAYS_DATE FROM DEPARTMENTS WHERE
DEPARTMENT_ID IN (:P_DEPT_ID)
```

Note that in the first example, DUAL is not required. You can access SYSDATE directly.

Avoid PL/SQL Calls at the Element Level

Package function calls at the element (within the group) or row level are not allowed; however you can include package function calls at the global element level because these functions are executed only once per data model execution request.

Example:

```
<dataStructure>
  <group name="G_order_short_text" dataType="xsd:string" source="Q_ORDER_ATTACH">
    <element name="order_attach_desc" dataType="xsd:string" value="ORDER_ATTACH_DESC"/>
    <element name="order_attach_pk" dataType="xsd:string" value="ORDER_ATTACH_PK"/>
    <element name="ORDER_TOTAL_FORMAT" dataType="xsd:string" value="
WSH_WSHRDPK_XMLP_PKG.ORDER_TOTAL_FORMAT "/> <!-- This is wrong should not be called
within group.-->
  </group>
  <element name="S_BATCH_COUNT" function="sum" dataType="xsd:double"
value="G_mo_number.pick_slip_number"/>
</dataStructure>
```

Avoid Including Multiple Data Sets

It can seem desirable to create one data model with multiple data sets to serve multiple reports, but this practice results in very poor performance. When a report runs, the data processor executes all data sets irrespective of whether the data is used in the final output.

For better report performance and memory efficiency, consider carefully before using a single data model to support multiple reports.

Avoid Nested Data Sets

The data model provides a mechanism to create parent-child hierarchy by linking elements from one data set to another. At run time, the data processor executes the parent query and for each row in the parent executes the child query. When a data model has many nested parent-child relationships slow processing can result.

A better approach to avoid nested data sets is to combine multiple data set queries into a single query using the WITH clause.

Following are some general tips about when to combine multiple data sets into one data set:

When the parent and child have a 1-to-1 relationship; that is, each parent row has exactly one child row, then merge the parent and child data sets into a single query.

When the parent query has many more rows compared to the child query. For example, an invoice distribution table linked to an invoice table where the distribution table has millions of rows compared to the invoice table. Although the execution of each child query takes less than a second, for each distribution hitting the child query can result in STUCK threads.

Example of when to use a WITH clause:

Query Q1:


```
SELECT DEPARTMENT_ID EDID,EMPLOYEE_ID EID,FIRST_NAME FNAME,LAST_NAME  
LNAME,SALARY SAL,COMMISSION_PCT COMMFROM EMPLOYEES
```

Query Q2:

```
SELECT DEPARTMENT_ID DID, DEPARTMENT_NAME DNAME, LOCATION_ID LOCFROM  
DEPARTMENTS
```

Combine these two queries into one using WITH clause as follows:

```
WITH Q1 as (SELECT DEPARTMENT_ID DID, DEPARTMENT_NAME DNAME, LOCATION_ID LOC  
FROM DEPARTMENTS),  
Q2 as (SELECT DEPARTMENT_ID EDID, EMPLOYEE_ID EID,FIRST_NAME FNAME,LAST_NAME  
LNAME,SALARY SAL,COMMISSION_PCT COMM  
FROM EMPLOYEES)  
SELECT Q1.*, Q2.*  
FROM Q1 LEFT JOIN Q2
```

ON Q1.DID=Q2.EDID

Avoid In-Line Queries (as summary columns)

In-line queries execute for each column for each row. For example, if a main query has 100 columns, and brings 1000 rows, then each column query executes 1000 times. Altogether, it is 100 multiplied by 1000 times. This is not scalable and cannot perform well. Avoid using in-line sub queries whenever possible.

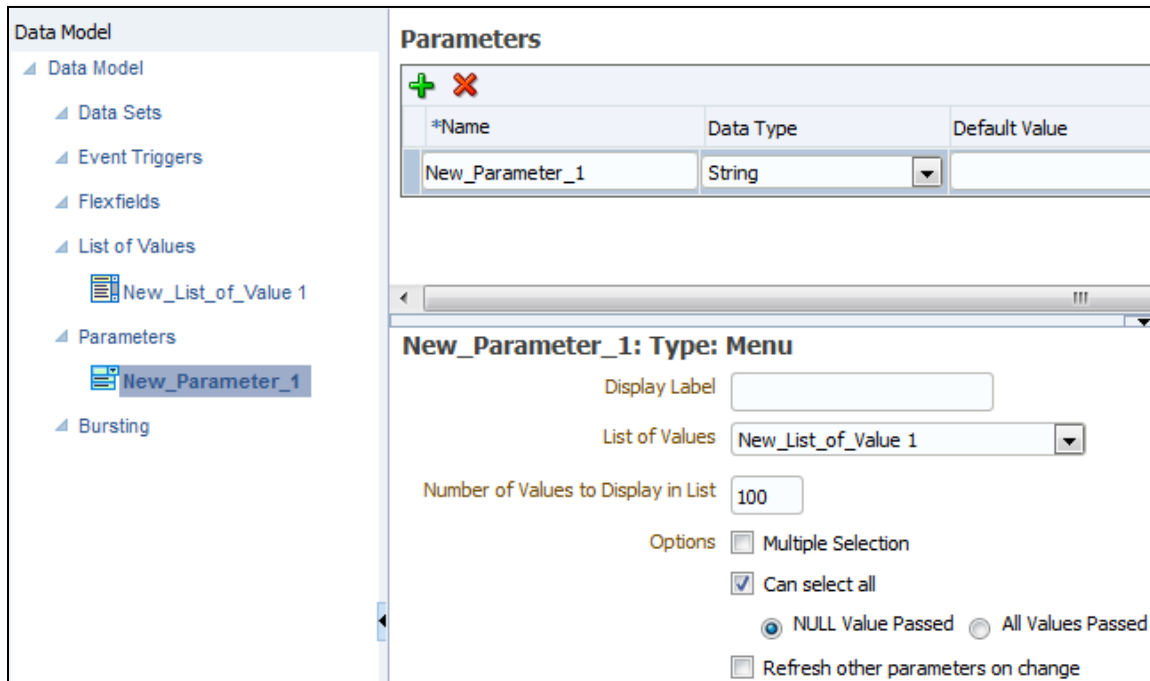
Avoid the following use of in-line queries. If this query returns only a few rows this approach may work satisfactorily; however, if the query returns 10000 rows, then each sub or inline query executes 10000 times and the query would likely result in Stuck threads.

```
SELECT
NATIONAL_IDENTIFIERS,NATIONAL_IDENTIFIER,
PERSON_NUMBER,
PERSON_ID,
STATE_CODE
FROM
(select pprd.person_id,(select REPLACE(national_identifier_number,'-') from per_
national_identifiers pni where pni.person_id = pprd.person_id and rownum<2)
national_identifiers,(select national_identifier_number from per_national
identifiers pni where pni.person_id = pprd.person_id and rownum<2) national_
identifier,(select person_number from per_all_people_f ppf
where ppf.person_id = pprd.person_id
and :p_effective_start_date between ppf.effective_start_date and ppf.effective_
end_date) PERSON_NUMBER
(Select hg.geography_code from hz_geographies hg
where hg.GEOGRAPHY_NAME = paddr.region_2
and hg.geography_type = 'STATE') state_code
```

Avoid Excessive Parameter Bind Values

Oracle database allows bind maximum of 1000 values per parameter. Binding a large number of parameter values is inefficient. Avoid binding more than 100 values to a parameter.

When you create a Menu type parameter, if your list of values may contain many values, ensure that if you enable both the "Multiple Selection" and "Can Select All" options, then also select NULL value passed to ensure too many values are not passed.



The number of columns selected in SQL should not exceed 100

We should restrict the SQL query to not fetch more than 100 columns as large number of columns will have adverse impact on report performance.

Sorting should be done in the SQL, not during report formatting

Sorting of data is highly recommended to be done at the time of data extraction and we should avoid sorting during report formatting to improve performance.

Avoid having WHERE clause on large tables with only NOT IN or <>

Such where clause on large data set will have adverse performance impact.

SQL Query Tuning

Query tuning is the most important step to improve performance of any report. Explain plan, SQL Monitoring, SQL Trace facility with TKPROF are the most basic performance diagnostic tools that can help to tune SQL statements in applications running against the Oracle Database.

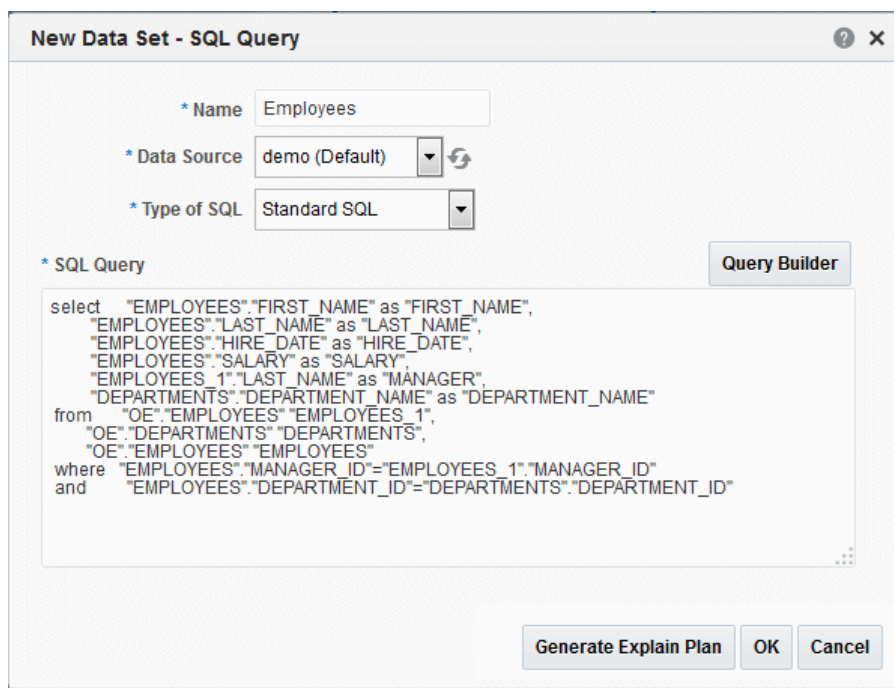
Oracle BI Publisher provides a mechanism to generate the explain plan and SQL monitoring reports and to enable SQL session trace. This functionality is applicable to SQL statements executing against Oracle Database only. Logical queries against BI Server or any other type of database are not supported.

Generate Explain Plan

You can generate an Explain plan at the data set level for a single query or at the report level for all queries in a report. For more information about interpreting the explain plan, see the Oracle Database SQL Tuning Guide.

Explain Plan for a Single Query

From the SQL data set Edit dialog you can generate an explain plan before actually executing the query. This will provide a best guess estimation of plan. The query will be executed binding with null values.



Click Generate Explain Plan on the Edit SQL Query dialog. Open the generated document in a text editor like Notepad or WordPad.

Explain Plan for Reports

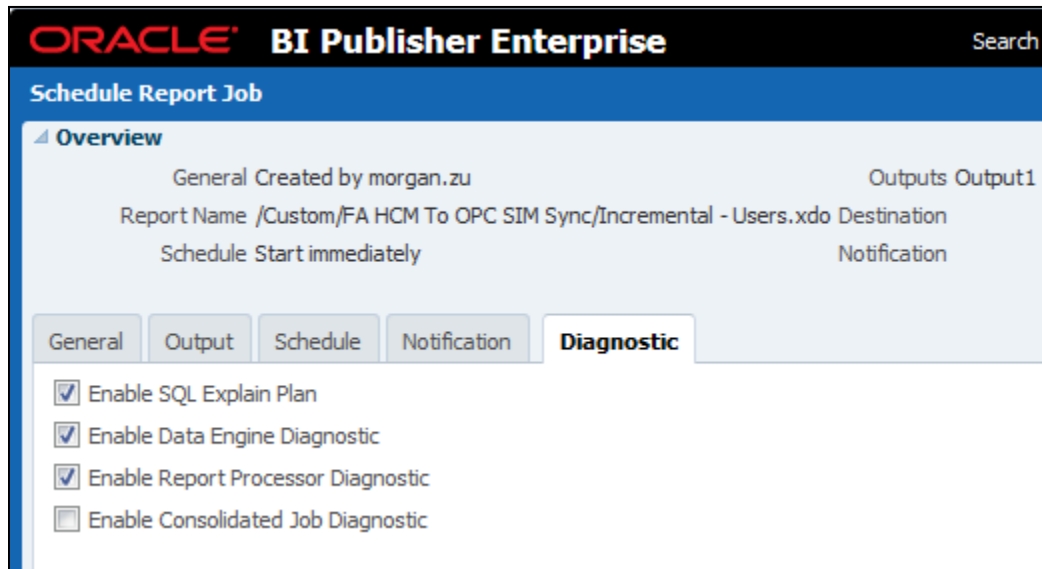
To generate an explain plan for a report, run the report through the Scheduler:

- On the New menu, select Report Job.

- Select the report to schedule then click the Diagnostics tab.

Note: You must have BI Administrator or BI Data Model Developer privileges to access the Diagnostics tab.

- Select one or more of the options to enable SQL Explain Plan, Data Engine Diagnostic and Report Processor Diagnostic. You can also chose to enable consolidated job diagnostic to create all the logs – Scheduler, Data Engine, Report Processor and Server log in one file.



- Submit the report.
- When the report finishes, go to the Report History page.
(From the Home page, under Browse/Manage, select Report Job History.)
- Select your report to view the details. Under Output & Delivery click Diagnostic Log to download the explain plan output.

C

Report Job History Home Catalog New ▼ Open

Last Refreshed Mon Aug 03, 2015 09:28:13 AM Western European Summer Time ↻

General Information

General Information

Report Job ID 1004
 Report Job Name Orders Job
 Owner
 Report Name New Orders
 Report Scope Public
 Report Job Schedule 8/2/15 4:39:42 PM WEST
 Active Start Date
 Active End Date
 Trigger Data Model
 Trigger Name
 Trigger Retry Limit
 Trigger Pause Time
 Trigger Parameters

No parameters available

Report Job Execution Information

Report Job Status ✔ Success
 Start Processing Time 8/2/15 4:39:44 PM WEST
 End Processing Time 8/2/15 4:39:54 PM WEST
 Time Elapsed 10.655 seconds

Output & Delivery

XML Data Diagnostic Log

Status

Output Name	Template	Format	Locale	Time Zone	Ca
Output1	Orders	HTML	English (United States)	[GMT+00:00] Casablanca	Gr

NOTE: The explain plan at Data Model level that executes a single query is recommended for report authors as it does not execute the report, while in case of schedule job diagnostic the explain plan is executed along with the report.

Guidelines for Tuning Queries

- Analyze the explain plan and identify high impact SQL statements.
- Add required filter conditions and remove unwanted joins.
- Avoid and remove FTS (full table scans) on large tables. Note that in some cases, full table scans on small tables are faster and improve query fetch.
- Use SQL hints to force use of proper indexes.
- Avoid complex sub-queries and use Global Temporary Tables where necessary.
- Use Oracle SQL Analytical functions for multiple aggregation.
- Avoid too many sub-queries in where clauses if possible. Instead rewrite queries with outer joins.
- Avoid group functions like HAVING and IN / NOT IN where clause conditions.
- Use CASE statements and DECODE functions for complex aggregate functions.



Administration Page Configurations

The Administration page of Oracle BI Publisher has several configurations under different sections such as Data Sources, Security Center, Delivery, System Maintenance, Runtime configuration and Integration.

In the SaaS environment, the following configurations **should never be changed** by Customers. These are provisioned for every customer and maintained by Oracle CloudOps. Any change in these configuration should go through Service Request process flow (explained under section “Process to initiate critical changes on BI Publisher Server”)


- a. Security Center
 - o Security Configuration
 - o Roles and Permissions
- b. System Maintenance
 - o Server Configurations
 - o Scheduler Configurations
- c. Integration
 - o Oracle BI Presentation Services

Also note that changes to above configuration require BI Publisher application restart.

Runtime Configurations

Runtime Configurations under following sections are recommended to be handled by Customers:

- a. Properties
 - i. PDF Output
 - ii. PDF Digital Signature
 - iii. PDF/A Output
 - iv. PDF/X Output
 - v. DOCX Output
 - vi. RTF Output
 - vii. HTML Output
 - viii. RTF Template
 - ix. PDF Form Templates
 - x. Flash
 - xi. CSV Output

- 
- xii. Excel 2007 Output
 - xiii. All Outputs
 - b. Font Mappings
 - c. Currency Formats

Runtime Configurations under following sections **should be changed with extreme CAUTION** as they will impact report and server performance and a **mistake may cause outage**.

- a. Properties
 - i. FO Processing
 - ii. Memory Guard
 - iii. Data Model

Memory Guard

Memory Guard settings are to enable administrator to restrict reports to use certain limits of JVM memory and if a report exceeds the limit, it will fail to execute. This will protect the BI Publisher server from some unwarranted extremely large report generation that may crash the server.

Maximum report data size for online reports

This property enables you to specify a maximum data size allowed for online report viewing. When you set a maximum data size, and if a user runs a report that exceeds this limit, he gets the following message:

"The report you are trying to run exceeds the data limit set for this server. Either re-run with parameters that reduce the data or schedule this report. Contact your Administrator if you have questions."


Maximum report data size for offline (scheduled) reports.

This feature enables you to specify a maximum data size allowed for scheduled reports. If the data generated is larger than the maximum setting, the report processing is ended. The scheduled report job fails with the following status message:

"Report data size exceeds the maximum limit (<nnn> bytes). Stopped processing."

Free memory threshold

This setting enables you to specify a minimum value for free JVM space. This enables you to control whether to run a report based on two factors: current usage and the size of the report data. This feature requires the setting of several properties that work together. You specify the threshold JVM space, the report maximum report size that will



be allowed when the JVM falls below the threshold, and the maximum wait time to pause the report to wait for more JVM free space to become available.

If the report data size exceeds the threshold, then the report is paused to wait for free memory to be available. The report will wait for the time specified in the property Maximum Wait Time for Free Memory to Come Back Above Threshold Value. If the free memory does not rise back above the minimum in the wait period specified, the report request is rejected.

Maximum report data size under the free memory threshold

Maximum single report data size allowed when free JVM memory is under the specified threshold value set in Free memory threshold. For example (assuming the default setting), if the data generated for a single report exceeds one-tenth of the value set for Free memory threshold, then processing is terminated. Therefore if the Free memory threshold is set to 100 MB and a single report data extract exceeds 10 MB, then the report processing is terminated.

This property takes effect only when Free memory threshold is set to be a positive value.

Minimum Time Span between Garbage Collection Runs

Set this value to avoid overrunning JVM garbage collection. The server enforces the minimum of 120 seconds, which means the value will be reset to 120 seconds if it falls below the minimum.

Maximum Wait Time for Free Memory to Come Back Above the Threshold

The maximum time in seconds that a run-report request will wait for free JVM memory to come back above the threshold value. This property value takes effect only when a positive value for Free memory threshold is specified.

If the free memory becomes available within the time specified, the request will proceed immediately to generate the document. If free memory is still below the threshold value after the time specified, the request is rejected. For online requests, the larger this property value, the longer the browser will wait for a request to run.

Process timeout for online report formatting

For online reports, the maximum time in seconds that a formatting process is allowed to run. If an online report formatting process exceeds the limit, the user receives the error message:

"Formatting time (nn seconds) exceeds the limit (nn seconds). Stopped processing."



Administration		Home	Catalog	Favorites ▼	Dashboards ▼	New ▼	On
Maximum number of rows	<input type="text"/>						25000
<input checked="" type="checkbox"/> All Outputs							
Hide version number in output	<input type="text"/>	▼					False
Use 11.1.1.5 compatibility mode	<input type="text"/>	▼					False
<input checked="" type="checkbox"/> Memory Guard							
Maximum report data size for online reports	<input type="text"/>						300MB
Maximum report data size for offline (scheduled) reports	<input type="text"/>						500MB
Free memory threshold	<input type="text"/>						500MB
Maximum report data size under the free memory threshold	<input type="text"/>						free_memory_threshold/10
Minimum time span between garbage collection runs	<input type="text"/>						300 (seconds)
Maximum wait time for free memory to come back above the threshold value	<input type="text"/>						30 (seconds)
Process timeout for online report formatting	<input type="text"/>						600 (seconds)
<input checked="" type="checkbox"/> Data Model							
Maximum data size limit for data generation	<input type="text"/>						500MB
Maximum sample data size limit	<input type="text"/>						1MB
Enable Data Model scalable mode	<input type="text"/>	▼					True
Enable Auto DB fetch size mode	<input type="text"/>	▼					True
DB fetch size	<input type="text"/>						20
SQL Query Timeout	<input type="text"/>						600 (seconds)
Enable Data Model diagnostic	<input type="text"/>	▼					False
Enable SQL session trace	<input type="text"/>	▼					False
Enable SQL Pruning	<input type="text"/>	▼					False

Note: Remember to be extremely cautious while changing these settings as they may negatively impact report and server performance.

Data Model

Maximum data size limit for data generation

Maximum XML data size in that can be generated from the execution of a data model. This setting applies to both online report requests and to requests submitted through the scheduler. When the size of the file generated by the data engine exceeds the value set for this property, the data engine terminates execution of the data model and throws an exception.

You can set the value in GB, MB, or KB. To turn this property off, enter 0 or a negative number.

Maximum sample data size limit

Maximum file size of a sample data file that can be uploaded to the data model editor.

Enable Data Model diagnostic

Setting this property to true will write data set details, memory, and SQL execution time information to the log file. Oracle recommends setting this property to true only for debugging purposes. When set to true, processing time is increased.

Enable SQL Session Trace

Setting this property to True writes a SQL session trace log to the database for every SQL query that is executed. The log can then be examined by a database administrator.

Oracle recommends that you turn this property on only in test and development environments.

Important: To enable this property, the user that you define for the database connection must be granted the Alter Session privilege on the database.

Scheduling & Delivery

As a part of sizing and capacity planning, we should always plan a separation of online report from batch/scheduled jobs by time and/or by server. Running both online and scheduled jobs concurrently on same server during peak business hours will cause performance issues and may upset the online report customers.

How to decide whether to run a Report Online/Offline (Schedule)

Running reports in interactive/online mode uses in-memory processing. Use the following guidelines for deciding when a report is appropriate for running online.

For Online / Interactive mode:

- When report output size is less than 50MB
Browsers do not scale when loading large volumes of data. Loading more than 50MB in the browser will slow down or possibly crash your session.
- Data model SQL Query time out is less than 600 seconds
Any SQL query execution that takes more than 600 seconds results in Stuck WebLogic Server threads. To avoid this condition schedule long-running queries. The Scheduler process uses its own JVM threads instead of Weblogic server threads. It is more efficient to schedule reports than run reports online.
- Total number of elements in the data structure is less than 500
When the data model data structure contains many data elements, the data processor must maintain the element values in memory; which may result in OutOfMemory exceptions. To avoid this condition, schedule these reports. For scheduled reports, the data processor uses temporary file system to store and process data.
- No CLOB or BLOB columns
Online processing holds the entire CLOB or BLOB columns in memory. You should schedule reports that include CLOB or BLOB columns.

Scheduler Database Management

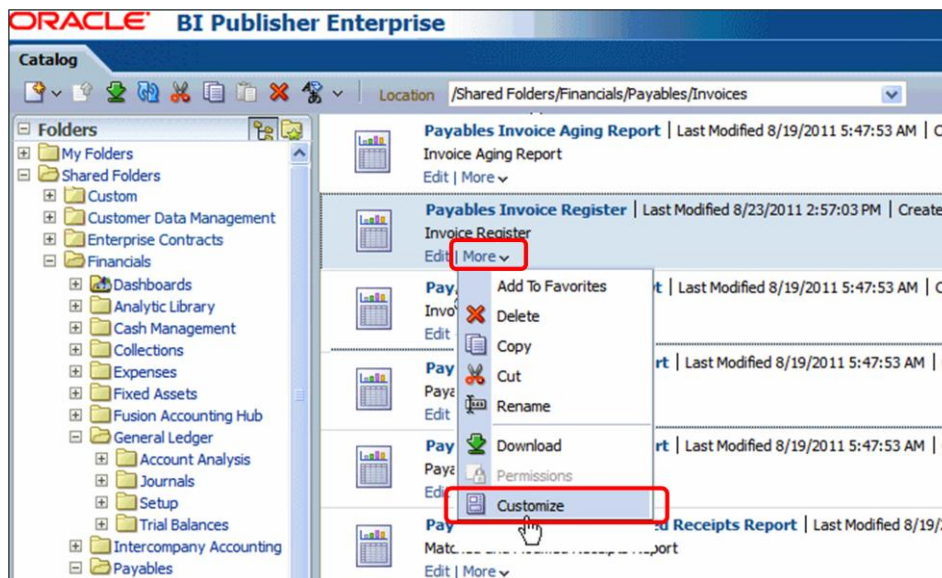
Turn off saving of XML data if very large reports are to run and growing database size is a concern

Scheduling Tips

- Use scheduler history page and hover over status detail field to monitor the job execution status (time taken, stage in which the job is currently at, which server runs the job, etc)
- Schedule the jobs to avoid maintenance window.
- Scatter the jobs to distribute the load uniformly over a period of time. Do not clutter all jobs in a short period of time.
- User should run critical scheduled jobs during off-hours when fewer resources are shared by other processes, or run them on a dedicated bi server where on-demand queries are running on a different server.
- Administrator can use Scheduler Diagnostics page to check the health of scheduling engine and to observe number of deliveries in different queue
- When a job is interrupted due to maintenance shutdown the job will be marked as failure when server is back up again. Manual resubmission is needed.

Report Customization

Several Oracle products that integrate with Oracle BI Publisher, ship canned or pre-packaged reports. Often there is a need to customize certain reports to fit to the organizational requirement. Oracle BI Publisher provides a report customization feature (only available through /xmlpserver URL) to help with customization.



When you select the Customize option for a report, BI Publisher creates a copy of the report in the "Custom" folder. This custom copy is linked internally to the original report. You can customize the custom copy of the report, leaving the original report intact. When users initiate a request to run the original report, whether from the BI Publisher catalog or through an application process, BI Publisher detects the customized version and runs your custom version instead.

The Customize feature provides the following benefits:

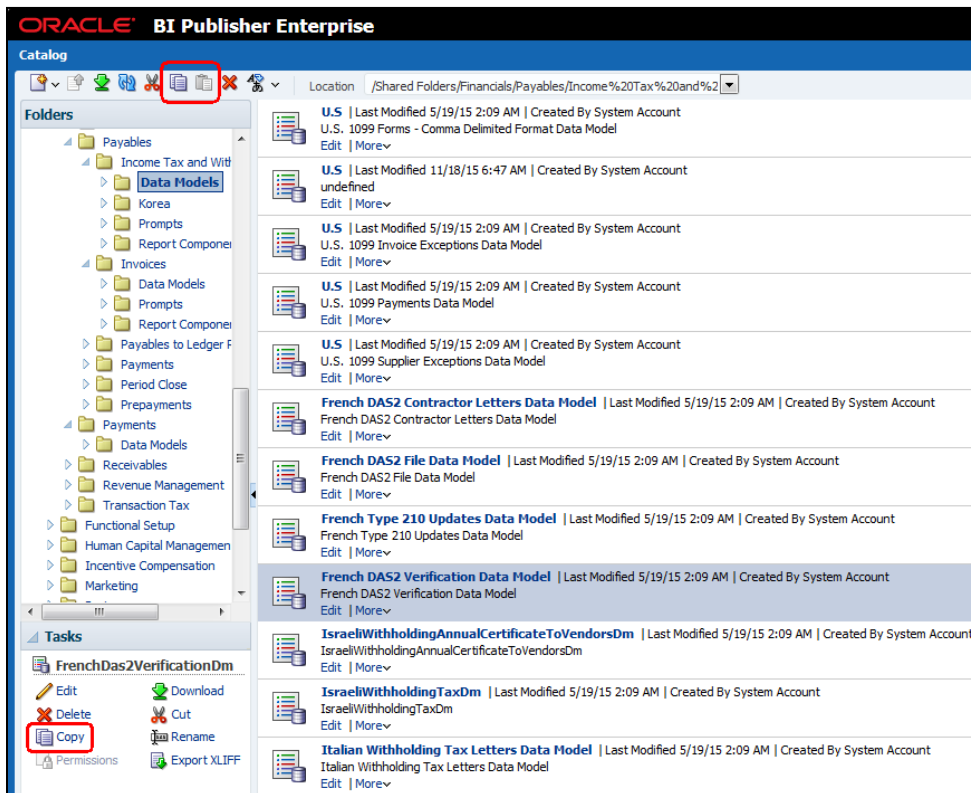
- Enables a customization process that mimics the convenience of an "in-place" customization. BI Publisher automatically creates the copy and the mapping.
- Removes the requirement to edit calling processes or applications to execute the custom report. Although you customize the copy, BI Publisher automatically sends all requests to run the original report to the custom copy instead.
- Removes the risk of patches overwriting your customizations of prepackaged Oracle reports.

Important Note:

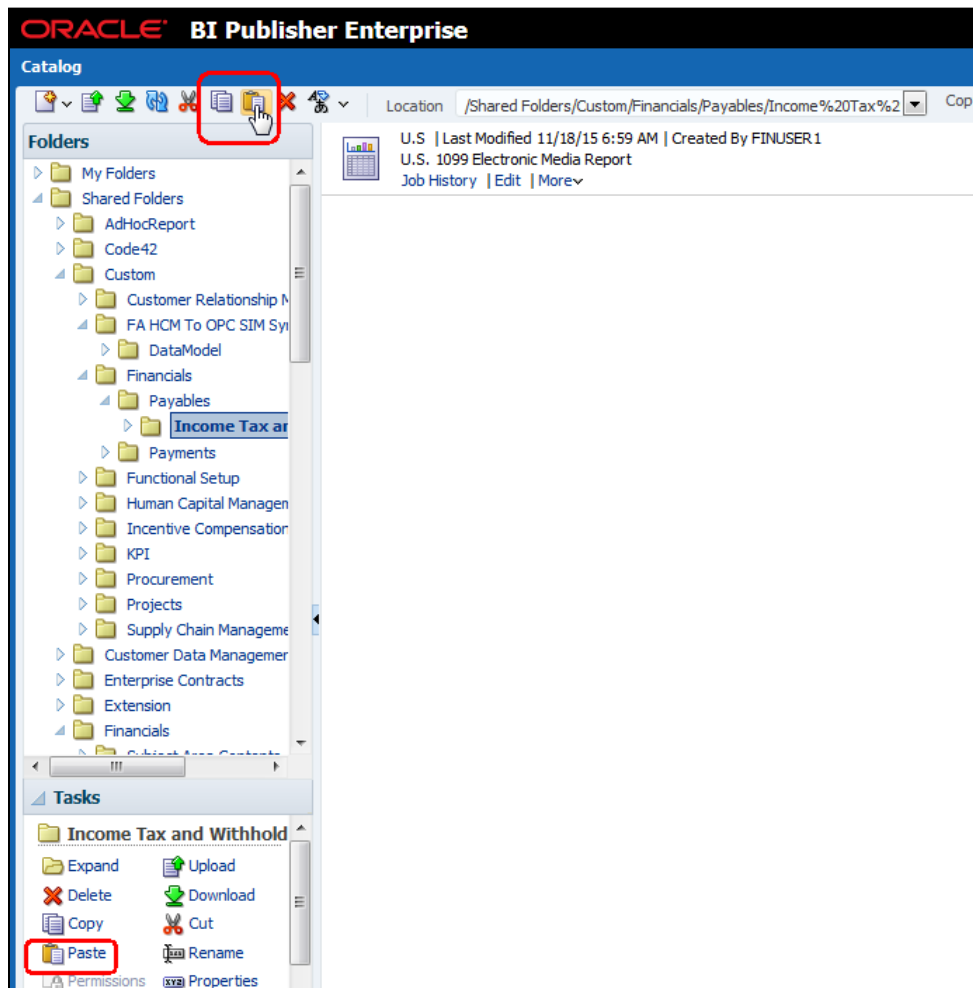
- The Customize option is **available only for reports**. The Customize option is not available for data models, style templates, or sub templates.

To customize data models, style templates or sub templates and insulate them from potential changes from patching, make a copy of the data model, style template, or sub-template and either rename it or place it in a custom directory. Ensure that you update any reports to point to the customized data model, style template, or sub-template.


To copy a data model (or Style template or sub template), select the data model and use copy icon from the set of icons above the Folders accordion pane or by clicking on "copy" from Tasks pane.



Once copied, the paste icon is activated in Folders Pane and Tasks Pane. Create a Data Model folder in the Custom Folder under the appropriate report folder and then paste the data model object. The Custom Report needs to point to this copied data model.



- **The security grants** applied to the original report and folder hierarchy are not copied to the report created in the Custom folder. You must manually apply the security settings for the reports and folders that are created by the Customize feature in the Custom folder.
- When creating Custom Data Model, follow the guidelines and best practices of Data Model Creation.



Scheduling a Custom Report

When you use the Customize feature to add a custom layout to an existing report, you do not need to edit the Oracle Enterprise Scheduler Services (ESS) job.

ESS jobs are not customizable for Oracle Cloud implementations. **Therefore, a custom report created new cannot be configured to run through ESS. Such custom report can only be run through Oracle BI Publisher scheduler user interface.**

Note: Remember to mark any long running (over 10 minutes) or large custom report to be run by Scheduler

Viewing a Custom Report

For reports appropriate to run online, you can run them on demand through the Reports and Analytics pane or from the Oracle Business Intelligence catalog.

Securing Custom Report

When you create a custom report you must secure it so that only users with appropriate privileges can run it.

Two options for securing the report:


- Use an existing role – This simply extends the security permissions of an existing role to the custom report. By default it inherits the permissions from the original report.
- Create a new custom role – This will require the steps of creating a custom role and then granting appropriate permissions to the role.

Troubleshooting Tips

Out of Memory Issue

Out of Memory is generally caused by large report processing in memory. This can be caused during data generation or during report generation. When such an error is seen, you can use Job Diagnostics feature in to check the SQL explain plan and data engine log first. If there are no out of memory error during data generation, then the out of memory could be caused during report generation.

- If the out of memory is caused by data, refer to the data model best practices and see if there is a way to improve the query or attribute related to DB fetch size etc.
- If the out of memory is caused by report generation, then check for best practices of Layout design to manage the template design better.
- If the issue persists, then probably the report requires more memory and therefore we should increase the heap size for the BI Publisher Server instance. Create a Service Request for Oracle Support to investigate this.



Out of Memory Exceptions can also occur if too many reports are being run concurrently. Check the configured value for number of threads used for scheduled reports and also the number of online concurrent requests. To change the number of threads in Scheduler Configurations, create a Service Request for Oracle Support.

Interactive features in reports also consume a large amount of memory and should be avoided against large data sets.

Unusually slow throughput from RTF template

If RTF template renders output extremely slow with large data set, for example if we observe a throughput less than 20 pages per second, then the possible areas to check will be:

- Excessive use of // to search of elements
- Use of preceding in the XPATH
- Very high number of column data being rendered (for example output in excel format with 70 columns)
- Use of “Repeating Header” attribute in data row

What logs to capture when report performs poorly on Server

- Bipublisher.log. If possible enable Trace 32 and then capture this log.
- Bi_server1.log
- JVM Thread Dump. Check if any thread with high CPU usage, WAIT, DEADLOCK or BLOCKED status
- Heap Dump. Helps to analyze Out of Memory error and memory leaks.

Create a Service Request for Oracle Support to help with capturing the above log files.

How to debug or troubleshoot an RTF template design?

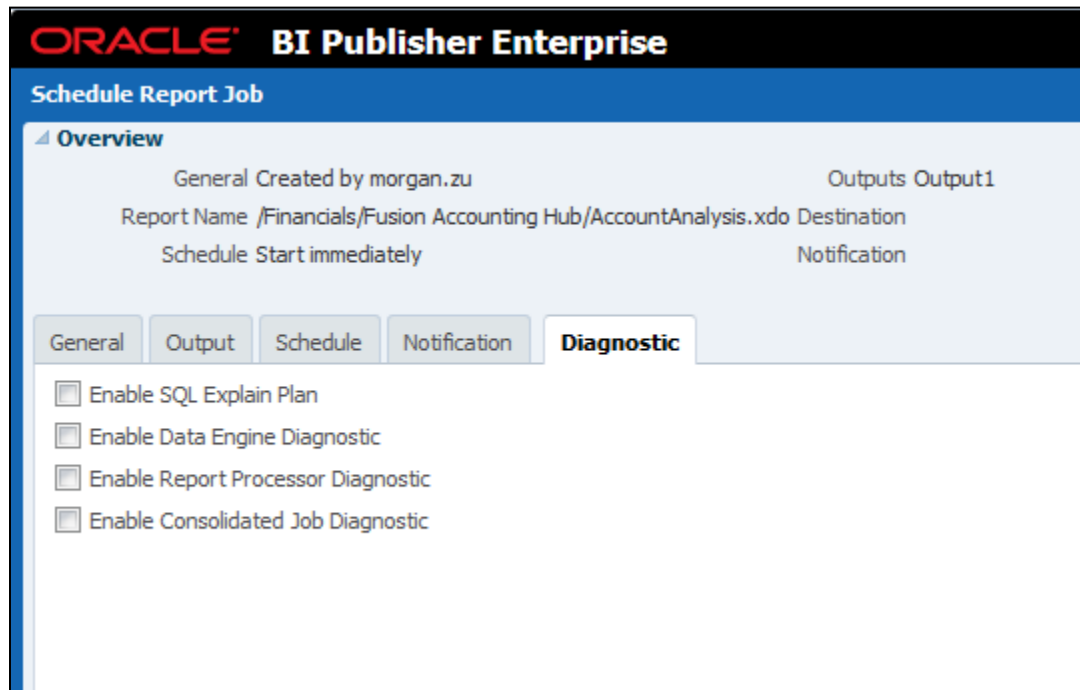
- Use Field Browser to view the entire RTF template code
- Add debug write statements (for e.g. <?\$varTest?>) and use font color (e.g. red or green) to print values of variables and calculations in the template or within a for-each loop.
- Take step by step approach to design a complex template in which first start with simple layout and calculation. Add one step of complexity and preview for error or correctness of data.
- If you are good at analyzing XSL-FO, then you can debug the XSL-FO generated by the RTF template

Stuck Job Scenario

When BIP job is somehow stuck in a running status, ESS will not launch the next run. This will cause entire recurring job to be stuck. The Administrator needs to go to ESS UI to remove the job

Schedule Job Diagnostics

BI Publisher has added diagnostics tab in the Schedule Job definition that allows user to view SQL Explain Plan, Data Engine Diagnostic log and Report Processor Diagnostic log. This feature has been introduced primarily to help report authors during custom report creation logs to diagnose report performance and other issues. This can also be used to troubleshoot production jobs.



Process to initiate critical Administration changes on BI Publisher Server





1. Open a Service Request to Oracle Support describing the change or problem
2. Support will open a Collab SR to engage CloudOps /PSR
3. CloudOps/PSR will perform the investigation on behalf of customer
4. Any configuration changes to be tested by CloudOps/PSR in Test environment.
5. If changes are approved by management, they will be applied to production SaaS environment
6. CollabSR will be updated with the changes
7. Customer Support will update Customer via Service Request



Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

CONNECT WITH US

-  blogs.oracle.com/oracle
-  facebook.com/oracle
-  twitter.com/oracle
-  oracle.com

Integrated Cloud Applications & Platform Services

Copyright © 2016, Oracle and/or its affiliates. All rights reserved. This document is provided *for* information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0116

Oracle BI Publisher Best Practices for SaaS Environments
May 2016
Author:
Contributing Authors:

