

TempWorks Software

*TempWorks Software WebCenter fx
Reporting Technical Reference*

White Paper

Date: January 2008



Contents

- Purpose and intended audience*..... 3
- A word of caution* 3
- Required software* 3
- Data source standards* 4
- Data sets*..... 4
- Parameters* 5
- Writing stored procedures using OriginId and OriginTypeId values* 5
- Naming conventions and recommendations*..... 7
- Stored procedure permissions*..... 8
- Adding finished reports to WebCenter*..... 8
- Creating a consistent look and feel*..... 8

PURPOSE AND INTENDED AUDIENCE

This document is intended to serve as a guide for creating custom reports compatible with the TempWorks WebCenter product. It is not designed to guide every step of the report development process, but rather a guide to how to create reports that will interface correctly with WebCenter's reporting system.

This document is tailored to meet the needs of IT professionals desiring to create custom WebCenter reports using Reporting Services. The level of detail provided assumes that the reader possesses a reasonable familiarity with Microsoft SQL Server Reporting Services, the SQL language, and the TempWorks database schema. Persons lacking in any of these areas will likely need more detailed information than what is provided here.

A WORD OF CAUTION

Be aware that by creating reports, you are allowing various outside parties to view data from within your database. If proper procedures are not followed, it is possible to create data security vulnerabilities that could allow unauthorized parties to view confidential or privileged information. If you are uncertain about any step along the way, please contact the TempWorks support department for assistance.

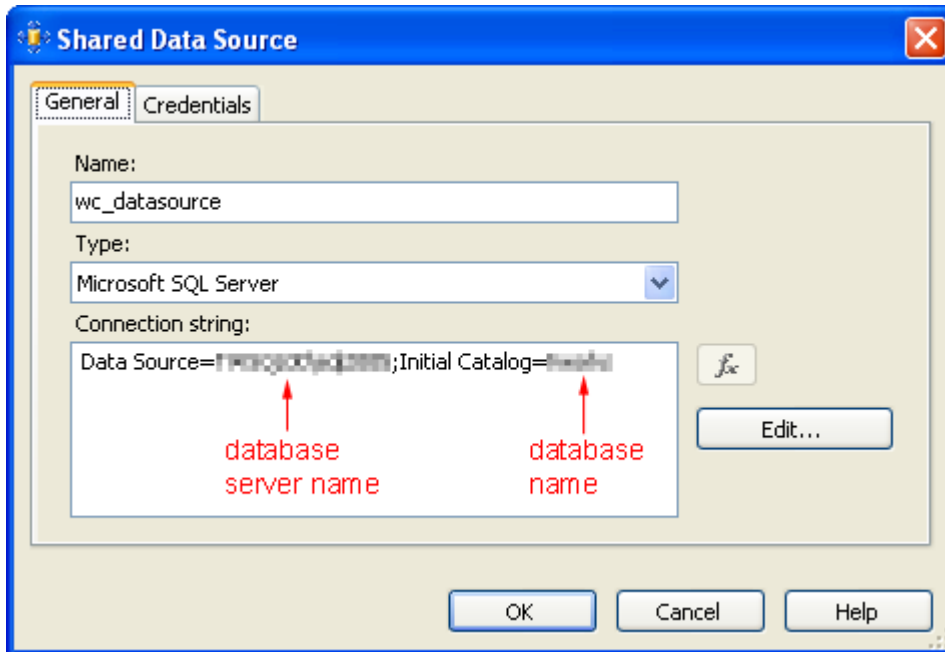
REQUIRED SOFTWARE

Two main pieces of software are needed to effectively create reports.

- graphical report designer:
 - Microsoft Visual Studio 2005 (more powerful, not free)
<http://msdn2.microsoft.com/en-us/vstudio/default.aspx>
 - SQL Server 2005 Express Edition Toolkit (less powerful, free download)
<http://www.microsoft.com/downloads/details.aspx?familyid=3C856B93-369F-4C6F-9357-C35384179543&displaylang=en>
- SQL server query interface:
 - Microsoft SQL Server Management Studio

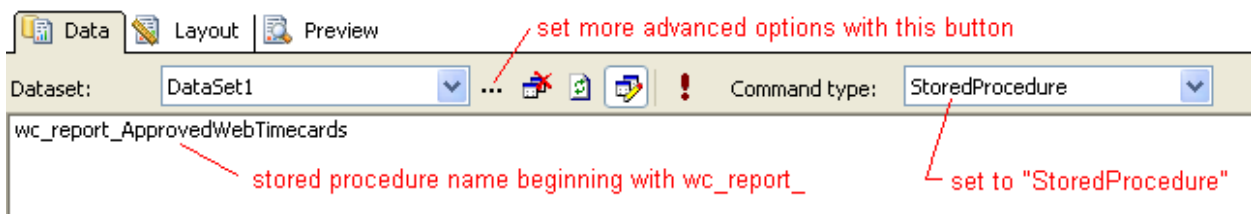
DATA SOURCE STANDARDS

WebCenter reporting uses a standard **shared data source** named **wc_datasource** for all report querying. You will need to create this data source and properly link it to your database. The data source should use **Windows Authentication (Integrated Security)** for credentials.



DATA SETS

Once a data source has been created properly, data sets can be created that will query against the data source. Although it is theoretically possible to create the query for a report as a part of the RDL file, we strongly recommend segregating the report and the query by using a stored procedure. Not only does this allow for much more complex queries, but it also makes it much easier to troubleshoot and update the queries behind reports after the fact.



PARAMETERS

WebCenter reporting requires two specific parameters to be used in order to function correctly, called **OriginId** and **OriginTypeId**. Setting up the parameters incorrectly will not only cause problems when the report is run, but **it can also provide a security vulnerability**, potentially allowing one customer or vendor to see data associated with a different customer. It is the responsibility of the report creator to ensure that the parameters are configured correctly and that they are properly used in the query or stored procedure to limit the data being returned. Here is a table of a typical set of parameters:

Parameter Name	Prompt Text	Data Type	Hidden?	Default Value	Required
StartDate	Start Date	DateTime	No	None	No
EndDate	End Date	DateTime	No	None	No
OriginId	OriginId	Integer	Yes*	None	Yes
OriginTypeId	OriginTypeId	Integer	Yes*	None	Yes

* Leaving the OriginId and OriginTypeId parameters unhidden will create large security vulnerabilities.

Note that the StartDate and EndDate parameters are not required and can be omitted completely if they are not needed for a report. WebCenter will function correctly with these parameters missing. Additional parameters can be added as required. There are no arbitrary limits on the quantity of parameters or their configuration, so long as the OriginId and OriginTypeId are always used and are configured correctly.

WRITING STORED PROCEDURES USING ORIGINID AND ORIGINTYPEID VALUES

WebCenter uses the OriginId and OriginTypeId values to determine what information to display to users. It is important that the same system is used correctly when creating queries for WebCenter reports. The OriginTypeId determines what type of contact is logged in, usually a customer contact or a vendor contact. The OriginId determines what data the person should be associated with by linking to the ID in the Contact table.

The OriginTypeId values used in WebCenter are:

- 1 employee
- 3 customer contact
- 12 vendor contact
- 18 serviceRep
- 36 applicant

Note that only two of these OriginTypeIds are valid for reporting; the customer contact and the vendor contact. WebCenter is not designed to handle reporting for any other roles. If you are interested in writing custom servicerep reports, please inquire about TempWorks Central, which is designed exactly for this purpose.

When a report is being run, the query driving the report needs to perform several joins to determine what data to display. If you were writing a report to display order information to customers, you would first need to verify that the OriginTypeId corresponds to a customer. Next, you would need to perform a series of joins starting with the OriginId to find orders associated with that contact's corresponding customer record. Here are some examples of joins:

To find orders for a customer contact:

```
-- the OriginId would normally be passed in as a parameter, but we
-- will create it manually for the purpose of this exercise.
declare @OriginId int
select @OriginId = 123

-- gather a list of related customers/depts to make sure nothing is missed
declare @CustomerId int
select @CustomerId = CustomerId from ContactRoot where Id = @OriginId

select o.*
from Order_ o
  inner join CustomerRoot c
    on o.CustomerId = c.CustomerId
  inner join dbo.wc_GetCustomerChildTree(@CustomerId) gct
    on c.CustomerId = gct.CustomerId
```

To find web timecards for a customer contact:

```
-- the OriginId would normally be passed in as a parameter, but we
-- will create it manually for the purpose of this exercise.
declare @OriginId int
select @OriginId = 665

-- a list of related customers/depts to make sure nothing is missed
declare @CustomerId int
select @CustomerId = CustomerId from ContactRoot where Id = @OriginId

select wt.*
from wc_Timecards wt
  inner join CtxnsRoot c
    on wt.CtxnsId = c.ID
  inner join AssignmentRoot a
    on c.ItemId = a.ItemID
  inner join Order_Root o
    on a.OrderID = o.OrderID
  inner join dbo.wc_GetCustomerChildTree(@CustomerId) gct
    on o.CustomerId = gct.customerId
```

To find employees associated (by assignment data) with a vendor contact:

```
-- the OriginId would normally be passed in as a parameter, but we
-- will create it manually for the purpose of this exercise.
declare @OriginId int
select @OriginId = 40

select e.*
from Employee e
inner join AssignmentRoot a on e.Aident = a.Aident
inner join ContactRoot c on a.CompanyIdent = c.CompanyIdent
where c.Id = @OriginId
```

Each actual query should be encompassed within an IF block that ensures only the correct OriginTypeId will be able to run the query, since each query will be performing joins differently. By doing this, it is possible to create reports that will run for more than one role, such as an assignment report that works both for customer contacts as well as vendor contacts. For example, a single report's stored procedure could contain both of the following IF blocks:

```
if @OriginTypeId = 3
begin
    --this block would contain a query designed for a customer contact
end

if @OriginTypeId = 12
begin
    --this block would contain a query designed for a vendor contact
end
```

NAMING CONVENTIONS AND RECOMMENDATIONS

Stored procedure names should generally be prefixed with **wc_report_** to allow them to be identified easily. When developing custom reports, it is good practice to add another prefix specific to your company's name. This will help identify custom work during upgrades and troubleshooting. The initials or an abbreviation of the company name should be sufficient. At TempWorks, we often use **tw_** in a similar manner. Be concise yet logical when naming a stored procedure and report. For the benefit of others, try to use the name to describe what the report does.

A poor stored procedure name: **NewReportJeff**

A good stored procedure name: **mp_wc_report_PastDuelInvoicePayments**

Similar matters apply when naming the actual RDL file. Our standard is to make the stored procedure name closely match the RDL file's name, which should be user-friendly and descriptive.

STORED PROCEDURE PERMISSIONS

Because WebCenter uses a centrally-authenticated user to access the database, no actual permissions are required on stored procedures. Note that this is not the case for most stored procedures outside of the WebCenter product.

ADDING FINISHED REPORTS TO WEBCENTER

Once a report is finished and has been properly tested and inspected, it needs to be uploaded and added to WebCenter. The first step is to actually upload the report's RDL file to the server. Enterprise customers should be able to upload the RDL file using the SQL Server Report Manager, but hosted customers do not have access to this tool for security reasons. If you are a hosted customer or likewise do not have access to the Report Manager, please contact the TempWorks support department for assistance: support@tempworks.com

Once the file has been uploaded, it still needs to be added to the list of reports in WebCenter. This is done from an administrative login by accessing the **add reports** link on the **control panel** page. Select Reporting Services as the report type, and follow the instructions in the application.

CREATING A CONSISTENT LOOK AND FEEL

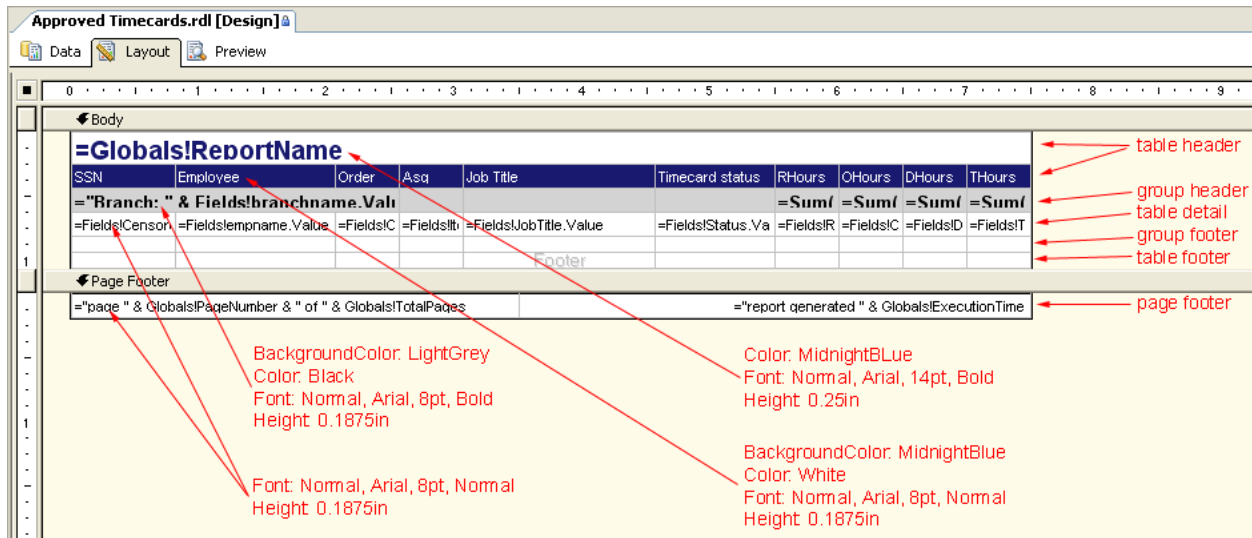
When creating custom reports, we recommend matching the general style and theme used in the standard reports provided with WebCenter. Here is how to achieve this look when starting from scratch:

Report properties:

- Set margins to **0.5in** on all four sides
- Set the GridSpacing to **0.0625in** (half of Visual Studio's default of **0.125in**)
- Check that both the **page size** and the **interactive size** are set to either **8.5in, 11in** or **11in, 8.5in**

The typical report uses a single table object with a width of 7.5in (7.5in plus two 0.5in margins fills an 8.5" wide page). The table object is placed within the Body section of the report, leaving no space or margin at the edges of the design area (margins are automatically added to the report when it is rendered). No changes need to be made to the table's default properties.

Add a row to the header of the table to display the report name. If necessary, add a group to the table to group the results. The standard WebCenter reports are formatted as follows:



Most of the text has been reduced to 8pt to allow more information to fit on the report. Group headers have a LightGrey background. The header at the top of the table object uses MidnightBlue 14pt font, and the table column headers use a MidnightBlue background with white 8pt font. The group and table footers are not hidden (to provide some spacing between group iterations). The page footer, containing page counts and a timestamp, is the same as the detail line.

To display the page number/count and a timestamp at the bottom of each page, create two boxes in the page footer containing the following expressions:

="page " & Globals!PageNumber & " of " & Globals!TotalPages	="report generated " & Globals!ExecutionTime
---	--

Add a top border property (using default style settings) for both of these boxes to create the dividing line effect. It will render at the bottom of each page like this:

page 1 of 1 report generated 5/18/2007 11:21:55 AM

Make sure to enable an upper border element for both of the boxes to create the line separating the footer from the main report content.