

Reports Developer Release
6i Batch Registering Report
Information in WebDB

Technical White Paper
April, 2000

TABLE OF CONTENTS

Before you begin	1
Batch Registering report definition files in WebDB	2
Running RWCON60 to add report information to a SQL*PLUS script file.....	2
Registering report definition file information in WebDB.....	4
Batch Removing Report Packages From WebDB	5
Appendix A: Batch registering RWCON60 command line arguments.....	6
Appendix B: Batch registering PL/SQL load function.....	13

Reports Developer Release 6i: Batch Registering Reports in WebDB

Oracle Batch Registering enables you to extract and import information about the many reports you want to register (i.e., add access to) in Oracle WebDB all at one time. This makes it faster and easier for you to restrict access to reports that are published on the Web. Batch Registering runs on both UNIX and Windows NT platforms.

Batch Registering uses a SQL*PLUS script that is populated with information (i.e., report name, system and user parameters) from many report definition files. The script file creates a package in WebDB for each report specified in the script when it is run. The SQL*PLUS script contains a PL/SQL load function for each report definition file that you want to add access to in WebDB.

With Batch Registering, you do the following:

- Batch register report definition file information in WebDB. See "Batch Registering Report Definition Files in WebDB."
- Batch remove report definition file packages from WebDB. See "Batch Removing Report Definition Files from WebDB."

You need only one SQL*PLUS script file to batch register or remove an unlimited number of report definition files to and from WebDB.

BEFORE YOU BEGIN

Ensure that you have the following before you begin batch registering reports.

- WebDB Release 2.2 or later must be installed and configured with Reports Developer Security. Refer to the *Publishing Reports* manual, Chapter 5. "Controlling User Access to Reports" for more information.
- Reports Developer Release 6i must be installed if you plan to use the RWCON60 executable to create the SQL*PLUS script file.
- To run the SQL*PLUS scripts, SQL*PLUS must be installed on the machine where the SQL*PLUS script is run.
- Batch Registering should be run by the Reports Developer system administrator (i.e., a user account with the RW_ADMINISTRATOR role).
- Access to the Reports Servers and printers must be created in WebDB before batch registering reports. The user accounts, availability calendars, and any user parameter LOVs must be defined in WebDB. If information, such as the restricted Reports Servers and Printers, have not be added to WebDB, then an invalid packaged procedure may be generated for each report. Refer to the WebDB online help for more information about creating access to Reports Servers and printers.
- Evaluate the reports you want to register in WebDB. If reports have common parameter values, such as DESTYPE, DESFORMAT, SERVER values, run the RWCON60 executable for each report group with common values. Each time you run the RWCON60 executable, a load function for each report defined in the SOURCE parameter is appended to the SQL*PLUS file defined in the DEST parameter.

THE BATCH LOAD FUNCTION

To batch register report definition files, you run a SQL*PLUS script file that contains a PL/SQL load function for each report you want to add access to in WebDB. An example the SQL*PLUS script with the batch registering function is shown below:

```
SET SERVEROUTPUT ON
VAR STATUS NUMBER;
EXEC :STATUS := RWWWVREG.REGISTER_REPORT (P_NAME=>'Earnings',
P_OWNER=>'webdb', P_SERVERS=>'repserver,acct_server',
P_FILENAME=>'earnings.rdf', P_DESCRIPTION=>'restricted report',
P_PRIVILEGE=>'SCOTT,JABERS,ACCT', P_AVAILABILITY=>'production'
P_TYPES=>'Cache,printer)', P_FORMATS=>'HTMLCSS,PDF)',
P_PRINTERS=>'sales_printer,acct_printer',
P_PFORMTEMPLATE=>'public.finance_template', P_TRIGGER='Is begin IF
UPPER(DESTYPE) = 'PRINTER' AND EMPNAME = 'JABERS' THEN RETURN(TRUE);
ELSE RETURN(FALSE); END IF; end;');
```

This batch load function contains information about the report definition file that is needed to create a package in WebDB (i.e., the report name, who has access privileges to run the report, the server it can run on, the destination format and type).

See Appendix B, "Batch registering PL/SQL load function" for more information about the Batch load function.

BATCH REGISTERING REPORT DEFINITION FILES IN WEBDB

Batch registering is a two-step process:

1. Add report information to the SQL*PLUS script file. You can choose to:
 - create the SQL*PLUS script file using the batch load function. See Appendix B, "Batch registering PL/SQL load function" for more information about these parameters.
 - use the RWCON60 executable to batch populate the SQL*PLUS script file with information about the reports. See "Running RWCON60 to add report information to a SQL*PLUS script file" for more information.
2. Run the SQL*PLUS script to add the report information to WebDB. When this SQL*PLUS script is run, a PL/SQL package is created for each report specified. See "Registering report information in WebDB" for more information.
3. After batch registering your report definitions, you need to look at each report definition in WebDB, going into the parameter page for each one. For more information, see "Verifying registered report definition files."

Running RWCON60 to add report information to a SQL*PLUS script file

The RWCON60 executable is used to convert files. The first step of the batch registering report definition files is to create a SQL*PLUS script file that contains a PL/SQL load function for each report being registered with user-supplied information and information extracted from the report definition file. The RWCON60 executable can be used to help you create SQL*PLUS file. Each line in the SQL*PLUS script is PL/SQL function that represents the information for a single report.

If you prefer, you can create the SQL*PLUS script file yourself and manually add the PL/SQL load function parameters rather than running the RWCON60 executable. See Appendix B "Batch Registering PL/SQL Load Function" for more information.

Tip: Group the report definition files by common parameters. Every time you run the RWCON60 batch registering command, it will append a load function for each specified report to the destination SQL*PLUS file.

On the machine where the report definition files are located:

1. Install Reports Developer Release 6i, if you haven't already. Refer to the *Getting Started* manual for more installation information.
2. From the command line prompt, type the RWCON60 arguments needed to batch register the report definition files. See Appendix A "Batch registering RWCON60 command line arguments" for a description of the arguments.

Note: To successfully create a SQL*PLUS script file with the load functions, you will need to specify the DTYPE, STYPE, SOURCE, and DEST arguments. To create a functional package in WebDB, you will need to specify the P_SERVERS, P_PRIVILEGE, P_TYPES, P_FORMATS in addition to the arguments used to create the SQL*PLUS script file.

The following example:

```
rwcon60.exe dtype="register" stype="rdffile"
source="(security.rdf,earnings.rdf,acc_pay.rdf)" dest="(output.sql)"
p_owner="webdb" p_servers="(repserver,acct_server)"
p_description="restricted report" p_privilege="(SCOTT,JABERS,ACCT)"
p_availability="production" p_types="(Cache,printer)"
p_formats="(HTMLCSS,PDF)" p_printers="(sales_printer,acct_printer)"
p_pformTemplate="public.finance_template" p_trigger="Is begin IF
UPPER(DESTYPE) = 'PRINTER' AND EMPNAME = 'SMITH' THEN RETURN(TRUE); ELSE
RETURN(FALSE); END IF; end;"
```

would generate a SQL*PLUS script file called `output.sql` that contains the following load functions:

```
SET SERVEROUTPUT ON
VAR STATUS NUMBER;
EXEC :STATUS := RWWWVREG.REGISTER_REPORT (P_NAME=>'Security',
P_OWNER=>'webdb', P_SERVERS=>'repserver,acct_server',
P_FILENAME=>'security.rdf', P_DESCRIPTION=>'restricted report',
P_PRIVILEGE=>'SCOTT,JABERS,ACCT', P_AVAILABILITY=>'production'
P_TYPES=>'Cache,printer', P_FORMATS=>'HTMLCSS,PDF)',
P_PRINTERS=>'sales_printer,acct_printer
P_PFORMTEMPLATE=>'public.finance_template'
P_PARAMETERS=>'(P_LASTNAME)(P_SSN)', P_TRIGGER=>'Is begin IF UPPER(DESTYPE)
= 'PRINTER' AND EMPNAME = 'SMITH' THEN RETURN(TRUE); ELSE RETURN(FALSE);
END IF; end;');
EXEC :STATUS := RWWWVREG.REGISTER_REPORT (P_NAME=>'Earnings',
P_OWNER=>'webdb', P_SERVERS=>'repserver,acct_server',
P_FILENAME=>'earnings.rdf', P_DESCRIPTION=>'restricted report',
P_PRIVILEGE=>'SCOTT,JABERS,ACCT', P_AVAILABILITY=>'production'
P_TYPES=>'Cache,printer', P_FORMATS=>'HTMLCSS,PDF)',
P_PRINTERS=>'sales_printer,acct_printer',
P_PFORMTEMPLATE=>'public.finance_template', P_TRIGGER='Is begin IF
UPPER(DESTYPE) = 'PRINTER' AND EMPNAME = 'JABERS' THEN RETURN(TRUE);
ELSE RETURN(FALSE); END IF; end;');
EXEC :STATUS := RWWWVREG.REGISTER_REPORT (P_NAME=>'Acc_pay',
P_OWNER=>'webdb', P_SERVERS=>'repserver,acct_server',
P_FILENAME=>'acc_pay.rdf', P_DESCRIPTION=>'restricted report',
P_PRIVILEGE=>'SCOTT,JABERS,ACCT', P_AVAILABILITY=>'production'
P_TYPES=>'Cache,printer', P_FORMATS=>'HTMLCSS,PDF)',
p_printers=>'sales_printer,acct_printer',
P_PFORMTEMPLATE=>'public.finance_template' P_TRIGGER=>'Is begin IF
UPPER(DESTYPE) = 'PRINTER' AND EMPNAME = 'JABERS' THEN RETURN(TRUE);
ELSE RETURN(FALSE); END IF; end;');
```

3. Check the `Reports.log` file (typically written to the current working directory) for errors that may have occurred during the conversion process. If the `Reports.log` file cannot be found, then no errors occurred during the conversion.

4. Supply system and user parameter information as desired.

Notice that the first RWWWVREG function in the example above generated an additional parameter called P_PARAMETERS. This occurred because the `security.rdf` file contains two user-defined parameters:

```
P_PARAMETERS=>' (P_LASTNAME) (P_SSN) ' ,
```

You can define the default value, low and high values, or LOV for each user parameter if you want to restrict the values to a range of values or to a list of values. Similarly, if you want to restrict system parameters, such as COPIES, to restrict the number of copies an authorized user can print, you do so using the P_PARAMETERS parameter.

For example, the edited P_PARAMETERS argument might look like the following:

```
P_PARAMETERS=>' (P_LASTNAME, LOV=LASTNAME_LOV) (P_SSN) (COPIES,
DEFAULT=1, LOW=1, HIGH=2) '
```

which restricts the P_LASTNAME user parameter to values listed in the LASTNAME_LOV, requires a user-supplied P_SSN value to validate the P_LASTNAME value, and sets the default of the COPIES system parameter to print one copy by default and restricts the number of printed copies to a range from 1 to 2.

See Appendix B: "Batch registering PL/SQL load function " for more information about the P_PARAMETER parameter.

5. Save and close the `output.sql` file.

Registering report definition file information in WebDB

1. Run the SQL*PLUS script that contains the RWWWVREG functions. Start SQL*PLUS and log on to the WebDB schema that will own the packaged procedures. Type the following:

```
@ output.sql
```

Note: This script will create packages in WebDB that contain the values defined by the load function parameters for each report identified in the script.

2. If you want to check or edit the contents of a report's package, start WebDB and log on as the Reports Server system administrator (i.e., a user account with RW_ADMINISTRATOR privileges). At the main menu page, click **Administer, Reports Developer Security, and Report Definition File Access**. Find the Report Definition File Access you want to review by choosing the schema that owns it's package. See Appendix B: "Batch registering PL/SQL load function " for more information about the WebDB fields that corresponds to the load function parameters.

Verifying registered report definition files

After you register an RDF file in the WebDB security framework, add it to a WebDB site, and schedule and run the report from the site, you will get the following error:

```
Name "_wvw_rw_log_id_=AAADBKAACAAACzDAAD_wvw_rw_stime_=9615794" has invalid
character ' ' (followed by details of the DAD, Procedure and the URL).
```

To avoid this problem, when registering the report within the WebDB framework, choose the PARAMETERS option in the Manage Component screen and choose Save Parameters, even if the report has no user-defined parameters. If you are using the batch registration utility to register many reports within WebDB, then you will need to save the parameters explicitly for each report to avoid this error.

This problem has been registered as Bug 1238387 and will be fixed in an upcoming patch to Reports 6i. Please contact Oracle Support Services or refer to Metalink for the latest information on this issue.

BATCH REMOVING REPORT PACKAGES FROM WEBDB

This is a two-step process:

1. In a text editor, create a SQL*PLUS file (e.g., `rmv_rdfs.sql`) that contains the `RWWWVREG.DEREGISTER_REPORT` function for each report definition file package you want to remove from WebDB. The function looks like the following:

```
VAR STATUS NUMBER;  
EXEC :STATUS := RWWWVREG.DEREGISTER_REPORT (P_NAME=>'Security');  
EXEC :STATUS := RWWWVREG.DEREGISTER_REPORT (P_NAME=>'Earnings');  
EXEC :STATUS := RWWWVREG.DEREGISTER_REPORT (P_NAME=>'Acc_pay');
```

Where *P_NAME* is the name of the report definition file package you want to remove from WebDB.

2. Run the SQL*PLUS script file that contains the `RWWWVREG.DEREGISTER_REPORT` functions. Start SQL*PLUS and log on to the WebDB schema that owns the report definition file packages that you want to remove from WebDB. Type the following:

```
@ rmv_rdfs.sql
```

where *rmv_rdfs* is the name of the SQL*PLUS script that contains the remove functions.

APPENDIX A: BATCH REGISTERING RWCON60 COMMAND LINE ARGUMENTS

The following command line arguments are used to import information from many report definition files to a SQL*PLUS script file. This section describes only those arguments used to populated the load function. Refer the Report Builder online help for more information the RWCON60 executable and other arguments associated with it.

DTYPE

Description DTYPE is the format the reports or libraries are being converted to.

Syntax

```
[ DTYPE= ] { DATABASE | PLLDB | PLDFILE | PLLFILE | RDFFILE | REPFIL | REXFILE | TDFFILE | XML | REGISTER }
```

Values

DATABASE means that the converted reports will be stored in ORACLE.

PLLDB means that the converted PL/SQL libraries will be stored in the database.

PLDFILE means that the converted PL/SQL libraries will be stored in files in ASCII format.

PLLFILE means that the converted PL/SQL libraries will be stored in files containing source code and P-code.

RDFFILE means that the converted reports will be stored in one or more report definition files (files with an extension of .RDF).

REPFIL means that the converted reports will be stored in one or more binary runfiles (files with an extension of .REP).

REXFILE means that the converted reports will be stored in one or more text files (files with an extension of .REX).

TDFFILE means that the report will be converted to a template (.TDF file).

XML means that the destination report(s) are stored in one or more XML files (files with an extension of .XML).

REGISTER means that a SQL*PLUS script file is created with the RWWWVREG.REGISTER_REPORT load function for each report defined as the SOURCE. Each load function is populated with information about the report.

Default

REXFILE

Usage Notes

- When you try to create a .REP file using RWCON60, the source report's PL/SQL is automatically compiled. If there are compile errors, an error message is displayed and the .rep file is not created. To avoid this problem, make sure you compile the source report's PL/SQL using FileCompile before you try to create a .REP file.
- When converting a report to a template, only objects in the report's header and trailer pages, and the margin area are used in the template. Objects in the body are ignored.

STYPE

Description: STYPE is the format of the report(s) or libraries being converted.

Syntax

```
[ STYPE= ] { DATABASE | PLLDB | PLDFILE | PLLFILE | RDFFILE | REXFILE | XML }
```

Values

DATABASE means that the source report(s) are stored in ORACLE.

PLLDDB means that the source PL/SQL libraries are stored in the database.

PLDFILE means that the source PL/SQL libraries are stored in files in ASCII format.

PLLFILE means that the source PL/SQL libraries are stored in files containing source code and P-code.

RDFFILE means that the source report(s) are stored in one or more report definition files (files with an extension of .RDF).

REXFILE means that the source report(s) are stored in one or more text files (files with an extension of .REX).

XML means that the source report(s) are stored in one or more XML files (files with an extension of .XML).

Default

DATABASE

Usage Note

- When DTYPE=REGISTER, choose DATABASE, RDDFILE, REXFILE, or XML.

SOURCE

Description SOURCE is the report/library or list of reports/libraries being converted. RWCON60 requires that you specify a source report or library.

Syntax

```
[ SOURCE= ] { sname | ( sname1 , sname2 , . . . ) }
```

Values

Any valid report/library name or filename (e.g., payroll.rdf).

A list of valid report/library names or filenames enclosed by parentheses with a comma separating the names (e.g., (payroll,acct_pay,earnings)).

Usage Notes

- SQL wildcard characters (%) and _) may be used for reports or libraries that are stored in the database. For example, R% would fetch all reports stored in the database that begin with R. All reports that match will be converted.
- A list of report/library names or filenames must be enclosed in parentheses with commas separating the names. For example:

```
(qanda, test, dmast)
```

- Wildcard characters are invalid for reports/libraries stored in files (i.e., with extensions of .RDF, .REP, .REX, .PLD, .PLL).
- The argument(s) for this keyword may be operating system-specific.

- If you are converting reports/libraries stored in ORACLE and are using system-owned Report Builder tables, you may list reports/libraries of other users, for example:

```
SOURCE=(sday.qanda,dsmith.test,dmast.sal)
```

- If you are using user-owned Report Builder tables, reports/libraries from multiple users must be converted for each user individually.
- You must have created the reports/libraries, or have been granted access to the ones you did not create, in order to convert them. If no USERID is prefixed to the report/library name, the USERID is assumed to be the current user.
- If you are converting from database to file and the database report/library name is too long to be a filename, Report Builder prompts you to provide a filename of the correct length for your operating system.
- When DTYPE=REGISTER, you may only want to list report definition files with common parameters, such as destination types and formats, user access, and availability calendars.

DEST

Description DEST is the name(s) of the converted reports or libraries, if they are stored in the database, or the names of the files.

Syntax

```
[DEST=] {dname | (dname1,dname2,...) | pathname}
```

Values

Any valid report/library name or filename (e.g., qanda).

A list of valid report/library names or filenames enclosed by parentheses with a comma separating the names (e.g., (qanda,test,dmast)).

Default

If the DEST keyword is not specified, RWCON60 uses the following default names:

when DTYPE	DEST is
DATABASE or PLLDB	SOURCE
PLDFILE	SOURCE with the .PLD extension
PLLFILE	SOURCE with the .PLL extension
RDFFILE	SOURCE with the .RDF extension
REPFIL	SOURCE with the .REP extension
REXFILE	expdat.rex
REGISTER	name of the SQL*PLUS script file (e.g., output.sql)

Usage Notes

- A list of report/library names or filenames must be enclosed in parentheses with commas separating the names. For example:

```
(qanda,test,dmast)
```

- If you have more destination names than there are source names, the extra destination names are ignored. If you have fewer destination names than there are source names, default names will be used after the destination names run out.
- When DTYPE=REGISTER, multiple destinations are not required. If you list more than one SQL*PLUS script file as the DEST, only the first is recognized. Subsequent destinations will be ignored.
- The argument(s) for this keyword may be operating system-specific.

P_owner

Description P_OWNER is the WebDB schema that owns a report's package that is created when registering the report definition files specified in the SQL*PLUS script.

Syntax

```
[P_OWNER=] {"webdb" }
```

Values

Any valid WebDB schema name.

Default

The schema name that you are logged on to WebDB when you run the SQL*PLUS script file.

P_servers

Description P_SERVERS is the restricted Reports Server(s) the report has access to run to.

Syntax

```
[P_SERVERS=] {" (repserver1, repserver2, ... ) " }
```

Values

Any valid TNS name of the Reports Server (e.g., repserver).

A list of valid Reports Server TNS names enclosed by parentheses with a comma separating the names (e.g., (repserver, acct_server, sales_server)).

Default

none

Usage Note

- Access to the Reports Server(s) should already exist in WebDB.

P_name

Description P_NAME is the report name displayed in WebDB.

Syntax

```
[P_NAME=] {"sales_report" }
```

Values

Any report name.

Default

If P_NAME is not specified, the PL/SQL function is populated with the report definition file name.

Usage Notes

- Specify P_NAME only when you want to use the same report name for each report definition file being registered in WebDB. This argument is typically left blank.
- The report name cannot be prefaced with numeric characters (e.g., 401K_report is an invalid file name and my_401K_report is valid).

P_description

Description P_DESCRIPTION is text that provides additional information about the report.

Syntax

```
[P_DESCRIPTION=]{ "This is a restricted report" }
```

Values

Any text string.

Default

none

P_privilege

Description P_PRIVILEGE are the user(s) or role(s) who have access privileges to run the report(s) specified.

Syntax

```
[P_PRIVILEGE=]{ " ( SCOTT , JABERS , PMARTIN , . . . ) " }
```

Values

Any user name or role that WebDB can recognize (e.g., SCOTT).

A list of user account names or roles enclosed by parentheses with a comma separating the names (e.g., (SCOTT , JABERS , PMARTIN)).

Default

none

P_availability

Description P_AVAILABILITY is the name of the availability calendar that determines when the reports specified will be available for processing.

Syntax

```
[P_NAME=]{production}
```

Values

Any valid availability calendar name.

Default

none

Usage Note

- The availability calendar must exist in WebDB before running the SQL*PLUS script. If it does not, an invalid package may be created.

P_types

Description P_TYPES is the destination type(s) that the reports specified are restricted to output to.

Syntax

```
[P_TYPES=]{ "(CACHE,MAIL, . . .) " }
```

Values

Any valid destination type (e.g., CACHE).

A list valid destination types enclosed by parentheses with a comma separating the names (e.g., (CACHE,MAIL,PRINTER)).

Default

none

P_formats

Description P_FORMATS is the destination format(s) that the reports specified are restricted to output to.

Syntax

```
[P_FORMATS=]{ "(HTMLCSS,PDF, . . .) " }
```

Values

Any valid destination type (e.g., HTML).

A list valid destination types enclosed by parentheses with a comma separating the names (e.g., (HTMLCSS,PDF,RTF)).

Default

none

P_printers

Description P_PRINTERS is the printer(s) that the reports specified are restricted to print to.

Syntax

```
[P_PRINTERS=]{ "(PRT1,PRT2, . . .) " }
```

Values

Any valid printer (e.g., PRT1).

A list valid destination types enclosed by parentheses with a comma separating the names (e.g., (PRT1,PRT2,PRT3)).

Default

none

Usage Note

- Access to the Printer(s) should already exist in WebDB before running the SQL*PLUS script.

P_pformTemplate

Description P_PFORMTEMPLATE is name of the WebDB template that determines the style of the Runtime Parameter Form.

Syntax

```
[P_PFORMTEMPLATE=]{"public.finance_template1"}
```

Values

Any valid WebDB template name.

Default

none

P_trigger

Description P_TRIGGER is a PL/SQL function that is executed when parameter values are specified on the command line and when users accept the Runtime Parameter Form. The function must return a boolean value (TRUE or FALSE).

Syntax

```
[P_TRIGGER=]{"Is begin IF UPPER(DESTTYPE) = 'PRINTER' AND EMPNAME = 'SMITH'  
THEN RETURN(TRUE); ELSE RETURN(FALSE); END IF; end;"}
```

Values

Any valid PL/SQL function that returns a boolean value.

Default

none

APPENDIX B: BATCH REGISTERING PL/SQL LOAD FUNCTION

The following is the PL/SQL Load function and parameter descriptions.

```
Function Rwwwvreg.register_report(p_owner varchar2,
                                p_name varchar2,
                                p_servers varchar2,
                                p_filename varchar2,
                                p_description varchar2,
                                p_privileges varchar2,
                                p_availability varchar2,
                                p_types varchar2,
                                p_formats varchar2,
                                p_printers varchar2,
                                p_pdfformTemplate varchar2,
                                p_parameters varchar2,
                                p_trigger varchar2)

return number;
-- =0 : succeeded;
-- !=0 : failed;
```

PL/SQL Parameter	Description
P_OWNER	<p>Owner of the schema. The default is the current WebDB schema that you are logged on to when you start the SQL*PLUS script.</p> <p>Corresponds to the Owner field on the Report Name and Schema page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_OWNER=> 'WebDB'</p>
P_NAME	<p>The name used to identify the report in WebDB.</p> <p>Corresponds to the Report Name field on the Report Name and Schema page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_NAME=> 'Earnings'</p>
P_SERVERS	<p>The name of Report Server(s) that the report definition files defined in the P_FILENAME parameter have access privileges to run. Comma delimited.</p> <p>Corresponds to the Server field on the Report Name and Schema page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_SERVERS=> 'repserver,acct'</p> <p>Note: Access to the Reports Server(s) must already exist in WebDB.</p>
P_FILENAME	<p>The report definition file name that is being restricted.</p> <p>Corresponds to the Report Definition File field on the Report Name and Schema page of the Report Definition File Access wizard in WebDB.</p>

PL/SQL Parameter	Description
	<p>Example: P_FILENAME=>'earnings.rdf'</p>
P_DESCRIPTION	<p>A description of the report.</p> <p>Corresponds to the Description field on the Report Name and Schema page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_DESCRIPTION=>'restricted report'</p>
P_PRIVILEGE	<p>The user(s) or role(s) given access privileges to run the report definition files defined in the P_FILENAME parameter. Comma delimited.</p> <p>Corresponds to the Selected Users and Roles list on the Report Users and Roles page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_PRIVILEGE=>'SCOTT,JABERS,WEBDB'</p>
P_AVAILABILITY	<p>The name of the availability calendar that determines when the report definition file defined in the P_FILENAME parameter will be available for processing.</p> <p>Corresponds to the Availability Calendar Name field on the Availability Calendar page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_AVAILABILITY=>'production'</p> <p>Note: The availability calendar should already exist in WebDB.</p>
P_TYPES	<p>The destination type(s) that the report definition file defined in the P_FILENAME can be output to (e.g., cache, printer). Comma delimited.</p> <p>Corresponds to the Destination Types multiple select box on the Required Parameters page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_TYPES=>'CACHE,printer'</p>
P_FORMATS	<p>The destination format(s) that the report definition file defined in the P_FILENAME parameter can output to (e.g., HTML, PDF). Comma delimited.</p> <p>Corresponds to the Destination Formats multiple select box on the Required Parameters page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_FORMATS=>'HTMLCSS,PDF'</p>
P_PRINTERS	<p>The printer(s) that the report definition file defined in the P_FILENAME parameter can print to. Comma delimited.</p> <p>Corresponds to the Printers multiple select box on the Required Parameters page of the Report Definition File Access wizard in WebDB.</p>

PL/SQL Parameter	Description
	<p>Example: P_PRINTERS=>'sales_printer,acct_printer'</p> <p>Note: Access to the printer(s) should already exist in WebDB.</p>
P_PFORMTEMPLATE	<p>The parameter form template that determine the page style of the Runtime Parameter Form.</p> <p>Corresponds to the Parameter Form Template field on the Required Parameters page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_PFORMTEMPLATE=>'public.finance_template'</p>
P_PARAMETERS	<p>The user parameters default, high, and low values, or the LOV name.</p> <p>Note: The P_PARAMETERS parameter does not have a corresponding RWCON60 argument. If you want to batch import user parameter values (i.e., default, low, high, or LOV), you will need to edit the PL/SQL load function in the SQL*PLUS file for each report.</p> <p>Corresponds to the Parameter Name, LOV, Low Value, High Value field on the Optional Parameters page of the Report Definition File Access wizard in WebDB. Note that the default corresponds to value set in the Runtime Parameter Form for the specified parameter.</p> <p>Example: P_PARAMETERS=>' (P_LASTNAME , LOV=LASTNAME_LOV) (P_SSN) (COPIES , DEFAULT=1 ,LOW=1 ,HIGH=2) '</p> <p>where <i>P_LASTNAME</i>, <i>P_SSN</i>, and <i>COPIES</i> are parameter names <i>LOV</i> is the name of the list of values <i>DEFAULT</i> is the default value <i>LOW</i> is the low value in a range of values <i>HIGH</i> is the high value in a range of values</p>
P_TRIGGER	<p>The validation trigger written in PL/SQL that returns a boolean statement (e.g., true: succeeded or false: failed).</p> <p>Corresponds to the Trigger field on the Validation Trigger page of the Report Definition File Access wizard in WebDB.</p> <p>Example: P_TRIGGER=>'Is begin IF UPPER(DESTYPE) = ''PRINTER'' AND EMPNAME = ''SMITH'' THEN RETURN(TRUE); ELSE RETURN(FALSE); END IF; end;'</p>



Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
+1.650.506.7000
Fax +1.650.506.7200
<http://www.oracle.com>

Reports Developer Release 6i: Batch Registering Reports in
WebDB

Copyright © Oracle Corporation 2000
All Rights Reserved

This document is provided for informational purposes only,
and the information herein is subject to change
without notice. Please report any errors herein to
Oracle Corporation. Oracle Corporation does not provide
any warranties covering and specifically disclaims any
liability in connection with this document.

Oracle is a registered trademark.

All other company and product names mentioned are used
for identification purposes only and may be trademarks of
their respective owners.