SAP BusinessObjects

SAP BusinessObjects Profitability and Cost Management Console User Guide

- SAP BusinessObjects Profitability and Cost Management 7.5

2010-05-28

SAP

2010-05-28

Contents

History of changes

The current version of this document is version 2.2. The following table provides an overview of the most important changes to prior and current versions of this document:

Version	Important Changes
Version 1.0	Initial version of the *SAP BusinessObjects Console User Guide* document for SAP BusinessObjects Profitability and Cost Management 7.5.
Version 1.1	The `EXECUTEDLROUTINES` command is added.
Version 1.2	A note is added to clarify the use of the **Load Stored Values** command. The section “Console Error Messages” is added for the list of Console Error Messages. Option is added to specify model server when creating a new model. Commands are added for new Work Manager functionality to execute processes as part of a user workflow. The topic “Work Manager commands” is added for new Work Manager functionality to execute processes as part of a user workflow.
Version 1.3	In section “Introduction to Profitability and Cost Management Console commands”, the example parameter value =IncludeTraceback is removed as it is no longer used. Option **Erase Data** is added to the command for executing data load routines, to specify whether to replace data values for Version and Period. Information is added for `username` command, advising that Console does not support Single Sign On.

Version	Important Changes
Version 2.0	This version is only applicable to SAP BusinessObjects Profitability and Cost Management 7.5 SP07 or higher. To improve clarity, a number of topics have been renamed and reorganised. A new section “Console command parameters” has been created. This contains information on Console commands for both Console Wizard and command-line use. Option **Target Sum** is added to the command type **Execute Data Load Routines**. Option **Timeout (minutes)** is added to the command type **Delete Model**. The following new commands have been added: **Load Model**, **Unload Model**, **Copy Model**, **Rename Model**, **Model Description**, **Model Server**, **Model Enabled**, **Audit Enabled**, **Command File**, **Logoff**, **Unique Model ID**. The previous **Pause** and **Wait** commands have been extended and clarified.
Version 2.1	This version is only applicable to SAP BusinessObjects Profitability and Cost Management 7.5 SP08 or higher. Amendments have been applied to clarify the terminology, and to clarify the functionality provided by Console and Console Wizard. The section of “Console comand parameters” is now Chapter 5 and the topic is renamed “Console comands available”. An example is added in the topic “Command File (`cmdfile`)”. Further information is added in the topic “Logoff (`logoff)`”. “Console Error Messages” is updated. The formatting of command parameters is revised in Chapter 5 “Console commands available”. A tip for using `|more` to view commands at the command line, is added in “Working with Console and its commands”.
Version 2.2	This version is only applicable to SAP BusinessObjects Profitability and Cost Management 7.5 SP09 or higher. Corrections have been made to the parameter descriptions for the following commands: Export Model to File. Export Results, Import XML File into Model, Run Data Bridge Import, Pause Processing for a While, Write a Message. The new **Sort Dimension** command has been added.

Introduction to the Console application

The Console utility is available to extend the usability of the SAP BusinessObjects Profitability and Cost Management applications. It is a client application that enables tasks to be run from a console, so no user interaction is required.

Using command-line statements, you can establish “jobs” (or scheduled tasks) to run commands against the various applications at a time of your choice.

You may create the required commands manually, or alternatively, use the Console Wizard utility, which is provided to assist you with generating and correctly formatting Console commands and command files.

2.1 Running Console

The Console utility is initiated by running the PCMConsole program file at the command prompt (in Windows systems this is sometimes called the system prompt or the DOS prompt).

The Console program is located by default in the `C:\Program Files\SAP BusinessObjects\PCM`. This location can be changed at installation time, if required.

The program is run in conjunction with a series of commands and parameters that are executed on the Profitability and Cost Management model server. Commands can be run:

- from the command line or
- from within a batch file.

Working with Console and its commands

The various functions of Console are called by command-line statements that are passed to the program when you run it. The command-line statements contain commands and related parameters that can assign values to model variables and pass values to model functions.

Commands have the following properties:

- Unless stated otherwise, each command can be used multiple times in a single Console process.
- The commands are executed in sequence, so their order is significant.
- Commands can be specified in either upper or lower case, or a mixture of both.
- All commands in an executable are checked to be valid at startup, before execution of any commands.

A model is automatically closed at the end of a successful Console process, provided no other users are connected.

If Console encounters an error while executing a command, it generates an error text containing a specific number reflecting the command type. The number for each command type is listed in “Console error messages”.

For details of all commands and parameters available in Console, refer to the chapter “Console commands available”.

Tip:
To display at the command prompt, a list of Console commands and parameters:

- Navigate to the directory where the Console program is located, then type `PCMCONSOLE /?`.
- To avoid the output scrolling past the command prompt buffer, add `|more` after the command, and this enables you to control viewing the output.

Related Topics

- Console commands available
- Console error messages

3.1 Constructing Console command-line statements

When formatting command statements, the commands and their parameters are separated by an equals sign (=). For example, the command `username` is shown with the parameter value User1 as: `username=User1`.

A command-line statement can contain any number of commands and parameters — they are executed in sequence. For example, to open and calculate a model called BIKES, you create a command-line statement to run PCMConsole with the commands `username`, `password`, `open` and `calculate` as follows:

```
PCMCONSOLE username=User1 password=pass open=BIKES calculate
```

Commands and parameters that contain long file names or spaces must be fully enclosed within double quotes, for example:

```
PCMCONSOLE username=User1 password=pass "open=My Bikes Model" calculate
```

Note:
The Console Wizard utility assists with the process of constructing command-line statements. For more information, see the chapter "Using the Console Wizard".

Related Topics

- Using the Console Wizard

3.2 Command files

Instead of passing many commands as a continuous single line, you can create a command file. This is a text file that contains either a list of commands or a single command. If the command file contains multiple commands, each one should be on a separate line. It is a good idea to use a command file for a complicated sequence of commands, or if if these commands are to be run multiple times, as part of a regular scheduled process, for example.

To call a command file, pass the file name as a parameter to Console with the `cmdfile` command. For more information, see the topic "Command File".

The Console Wizard utility assists with the process of creating command files. For more information, see the chapter "Using the Console Wizard".

Note:
If your environment requires the password to be encrypted within a command file then this will need to be generated by using the **Encrypted Password** function in the Console Wizard.

Example: CMDFILE

```
PCMCONSOLE "CMDFILE=C:\consolefile.ini"
```

Result: Console runs the commands contained within the file called `consolefile.ini` existing in the root of drive C.

So if the `consolefile.ini` file contains:

```
username=user
password=pass
open=Bikes
```

```
calculate
export=c:\BikesExport.xml,Overwrite,Results
```

then the model Bikes will be opened, calculated, and exported to the xml file `BikesExport.xml`.

Related Topics

- Command File (cmdfile)
- Encrypted Password (encryptedpassword)
- Using the Console Wizard

3.3 Work Manager commands

You can create a Console command file that is used as a template by Work Manager to enable user workflows to execute processes. Work Manager generates a temporary command file from the template and executes the Console job. The Work Manager server process must have write access to the directory where the Console `.ini` template file is located as it creates the temporary file in the same location.

The following commands are used to communicate with Work Manager:

- `wmconnect`

  Description: This command connects Console to Work Manager. Both the commands for a user name and a password (or encrypted password) are required before this, and must be valid login credentials.

  Note:
  This is not a required command, but it is useful in situations where you wish to ensure that you have established a connection, for example, before creating a model.

- `wmsendevent`=<WorkManagerEvent>

  Description: This command sends the event name to the Work Manager process instance, to enable its progess to be tracked.

Related Topics

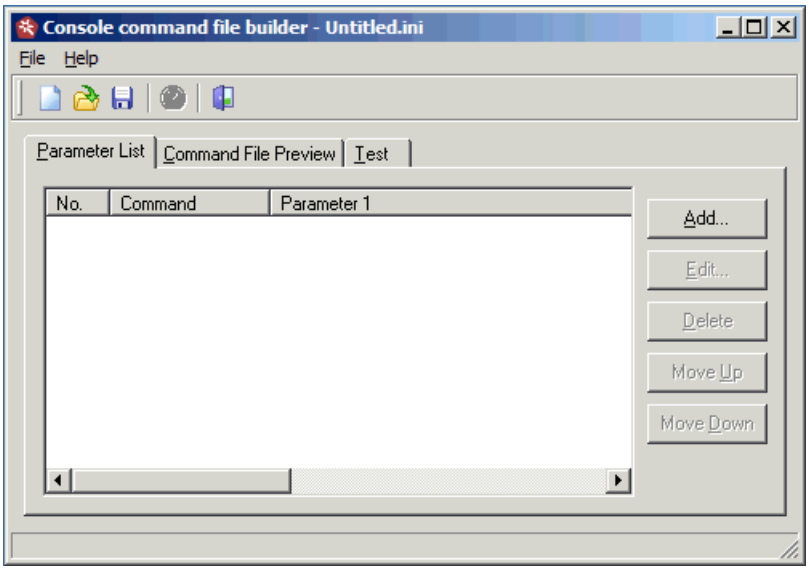
- Command files
- Connect to Work Manager (wmconnect)
- Send Work Manager Event (wmsendevent)

Using the Console Wizard

The Console Wizard is a quick and easy way to create your console job. You start the Console Wizard from the **Start** menu:

Start > Programs > SAP BusinessObjects > Profitability and Cost Management > Tools > Console Wizard.

When launched, the main screen appears as in the picture below:



When first initiated, the Console Wizard displays an empty new initialization file (*.ini) on the "Parameter List" tab for you to start to define. If this is not the first time you have opened the wizard, you will see the definition for the previous file that was in use.

The Console Wizard allows you to create command files, to create, edit and review parameters in a command file, and to run a command file in order to test it.

Related Topics

- Command files

4.1 Creating a new command file

In the Console Wizard you can create a new command file for console command parameters.

Click the **Create a new script file** icon, or navigate to the **File > New** menu item.

If any existing parameters are displayed, this clears them from the "Parameter List" and the "Command File Preview" tabs.

4.2 Defining console command parameters

A console file is a series of Profitability and Cost Management commands in the form of parameters for the Console application to execute.

On the "Parameter List" tab, you can **Add**, **Edit** or **Delete** commands, using the buttons or the right-click context menu. Once you have more than one command, you can alter their order of execution by moving them up or down as required, using the **Move Up** or **Move Down** buttons or the context menu.

4.2.1 To add a parameter to the command file

Note:
If creating a Console job for exporting, you are advised to define in advance, an export specification file. It this is not specified, then the Console Wizard will insert the default value of ALL tables, which may take an exceptionally long time to execute.

1. On the **Parameter List** tab, click **Add**.
   The "Command properties" dialog box displays. This allows a command type to be chosen and added to the **Parameter List**.
2. Select a **Command Type** from the list.
   The required parameter fields will be displayed. For each **Command Type**, you may need to specify the qualifying details, or some information concerning the Profitability and Cost Management system, for example, the name of the model or logon details. For further details, refer to the relevant command topic in “Console commands available”.
3. Enter any additional details as required in the displayed fields.
   This information is added to the command as parameters.
4. Click **OK** to add the command to the **Parameter List**.

The command will appear in the list, together with any parameters that you have specified.

Related Topics

- Console commands available

4.2.2 To preview the command file

- Click the **Command File Preview** tab to view the Console Wizard command file. Use this screen to check the correct order of precedence, and the correct contents for the command file before you attempt to use the file.

4.3 Testing the command file

It is recommended that you test the execution of the command file before you deposit the file on your model server for unattended execution.

The Console Wizard provides you with the ability to test your command file on the "Test" tab.

You need to ensure that the filepath to `PCMConsole.exe` (the Console Wizard application) displays in the **Console location** box. You determine the location of the Console Wizard application by browsing for the file. Its default location is in the `C:\Program Files\SAP BusinessObjects\PCM` directory, but this can be changed (and indeed has changed between versions of the software).

4.3.1 To test the command file

1. The **Command Line arguments** field displays a reference to the command file that you are creating. By default this is called `Untitled.ini`, but it is recommended that you save the file with a more meaningful name by selecting **File** from the menu bar and clicking **Save**.
2. Test the operation of the command file by clicking **Test Now**, to run the command file. (Output can be directed to the current screen by selecting the **Direct Console output to list below** option. Otherwise, a command window opens displaying status messages from the console job).
3. Once you are satisfied that there are no errors, and that the command script does what you expect it to, you can save the job by selecting **File** from the menu bar and clicking **Save**.
4. Create a shortcut to the command script using the **Create Shortcut** button.

5. The **Create Shortcut** button allows you to save a shortcut to the console job for easy initiation later. This shortcut can also be placed into applications such as the Windows Task Scheduler (**Start > Settings > Control Panel > Scheduled Tasks**) to be executed at a time of your choosing.

Console commands available

There are Console commands available for different functional categories, as shown in the summary tables below. For each specific command, refer to its individual topic for details of the command's options and its operation. Each topic describes how to use the command both in Console Wizard and as a command-line parameter.

Table 5-1: User login

Command name	Command	Purpose
Set Username	`username`	Sets the Profitability and Cost Management user account whose credentials will be used to perform actions. This command is usually required for all Console executables.
Set Password	`password`	Specifies the password for the user account. A username must also be set when this command is used.
Encrypted Password	`encryptedpassword`	Specifies the password for the user account in an encrypted format. A username must also be set when this command is used.
Logoff	`logoff`	Completely severs any client connection in the running system for the required logged in user(s) and logs them off.

Table 5-2: Model connection

Command name	Command	Purpose
Open Model	`open`	Opens an existing model, using the user account specified.
Close Model	`closemodel`	Closes the currently open model.

Command name	Command	Purpose
Load Model	`loadmodel`	Creates a connection to an existing model and keeps it open, regardless of other models being connected to during the Console run.
Unload Model	`unloadmodel`	Closes a connection with a specified model, opened by the `loadmodel` command.

Table 5-3: Model administration

Command name	Command	Purpose
Create a New Model	`newmodel`	Creates a new model.
Delete Model	`delete`	Deletes a model.
Copy Model	`copymodel`	Copies a model.
Rename Model	`renamemodel`	Renames a model.
Model Description	`modeldescription`	Amends the model description.
Model Server	`modelserver`	Changes the model server.
Model Enabled	`modelenabled`	Sets the model status.
Audit Enabled	`auditenabled`	Enables or disables auditing for the model.
Unique Model ID	`uniquemodelid`	Changes the model property Unique Model ID (UMID).
Sort Dimension	`sortdimension`	Sorts dimension items in a model hierarchy into ascending or descending order.

Table 5-4: Import

Command name	Command	Purpose
Import XML file into model	`import`	Imports an XML file into a specified model.

Command name	Command	Purpose
Run Databridge Import	`databridgeimport`	Executes a Data Bridge Control File (`*.CTF`) to import data into a specified model.
Execute Data Loader Routines	`executedlroutines`	Initiates a procedure for either Microsoft SQL Server or Oracle (depending on your installation) that automatically executes Data Loader routines in the correct order.
Load Volume Data	`loadvolumedata`	Loads Transactional Activity Driver data volumes into a Transactional Costing model and generates Activity Driver values.

Table 5-5: Export

Command name	Command	Purpose
Export model to file	`export`	Exports all the model's tables to an XML or CSV (comma separated values) text file.
Export To Database	`exporttodb`	Exports model results to a database table.
Export Results	`exportresults`	Exports model results to SAP BusinessObjects “Universe” tables.

Table 5-6: Calculation

Command name	Command	Purpose
Calculate model	`calculate`	Performs a calculation on the model that is currently open.
Automatic Calculation	`automaticcalculation`	Switches automatic calculation on or off for a model.
Load Stored Values	`loadstoredvalues`	Clears existing results and loads any cross-model rule values from associated models.

Command name	Command	Purpose
Lock Results	`lockresults`	Locks the specified version and period combination.
Unlock Results	`unlockresults`	Unlocks the specified version and period combination.
Clear Locked Results	`clearlockedresults`	Removes results for the specified version and period combination.
Calculate Transactional Costs	`calculatetransationalcosts`	Calculates the final unit rates and appends them to the `PPHV_UNITRATE` table, and also populates specific results tables.
Update Version Period Calculation List	`updatevpcalculationlist`	Adds or marks for deletion a Versions/Periods combination in the `PP_HVVERSIONPERIODCALCLIST` table, which holds a list of Versions/Periods combinations for which transactional costing calculations must take place.
Clear Version Period Calculation List	`clearvpcalculationlist`	Marks for deletion all Versions/Periods combinations in the `PP_HVVERSIONPERIODCALCLIST` table, which holds a list of Versions/Periods combinations for which transactional costing calculations must take place.
Load TC Calculation Only	`loadtccalconly`	Performs transactional costing mappings and recalculates Activity Driver values from the transactional Activity Driver volumes without loading any data from the `PPLOAD_HVACTIVITYDRIVER` table.

Table 5-7: Work Manager

Command name	Command	Purpose
Connect to Work Manager	`wmconnect`	Connects Console to Work Manager.

Command name	Command	Purpose
Send Work Manager Event	`wmsendevent`	Sends the event to the Work Manager process instance, to enable its progess to be tracked.
Set Work Manager Id	`wfid`	Not user specified.
Delete Command File After Use	`deletecmdfile`	Not user specified.

Table 5-8: Console control

Command name	Command	Purpose
Command file	`cmdfile`	Creates a placeholder for the contents of a specified command file.
Comment	`;`	Allows a comment to be inserted in a command file.
Log all events to text file	`log`	Specifies a name and location for retaining a log of the command parameters processed by the Console Wizard.
Pause Processing for a While	`pause`	Pauses execution of the command file parameters until the specified number of minutes has elapsed.
Prefix Date and Time	`prefixdate`	Places a date and time stamp before each command executed as part of the command file.
Wait, Keep Process Running for Certain Time	`wait`	Keeps a model open for the specified number of minutes.
Write a message	`msg`	Displays the message in the DOS box in which the command is run.

Related Topics

- Working with Console and its commands
- Using the Console Wizard

5.1 Audit Enabled (`auditenabled`)

Description:

- Specifies whether auditing is switched on for the model. This is only applicable if global auditing has been enabled via the SAP BusinessObjects Profitability and Cost Management Configuration Wizard.

Usage:

- `auditenabled`=<modelName>,<newStatus>

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<modelName>	Required	Name of existing model.
<newStatus>	Required	The value YES switches auditing on for the model. The value NO switches auditing off for the model.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Audit Enabled**.
2. Enter the name of the model.
3. Select the **Audit Enabled** checkbox to switch on auditing. If left unselected, auditing will be switched off.

5.2 Automatic Calculation (`automaticcalculation`)

Description:

- This command switches automatic calculation on or off in the currently open model.

Usage:

- `automaticcalculation`=[newValue]

The following parameter can be used with this command:

Parameter	Required / Optional	Description
[newValue]	Optional	Determines whether Automatic Calculation is switched on for the model. The value ON switches Automatic Calculation on. The default setting is OFF.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Automatic Calculation**.
2. Select the checkbox **Auto Calculation On**, to switch on automatic calculation for the model.

5.3 Calculate Model (`calculate`)

Description:

- Performs a calculation on the model that is currently open. It calculates all of the rule values required to produce Cost Object or Activity costs at leaf level for a model.

Usage:

- `calculate`

There are no parameters associated with this command.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Calculate Model**.

Example: Command-line statement to calculate a model

```
PCMCONSOLE username=user password=pass
    open=Model1 calculate
```

Result: Opens and calculates a model.

5.4 Calculate Transactional Costs (`calculatetransactionalcosts`)

Description:

- The Transactional Costing model must be open.
- Calculates the final unit rates and appends them to the `PPHV_UNITRATE` table, and also populates results tables, depending on the load ID parameter.
- If the Load ID contains a negative value, then the latest load ID for the active model will be used, otherwise the supplied load ID will be used. If the load ID contains the specific value of -1000, then

summarized detail cost value results will be output to the PPR_HVDETAILCOST table. For any other load ID value, the PPR_HVBREAKDOWN and PPR_HVBREAKDOWNDETAIL tables will be populated with detailed cost value results. (For further details refer to the *Profitability and Cost Management Transactional Costing User Guide*.)

Usage:

- calculatetransactionalcosts=<loadId>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<loadId>	Required	The Load ID is either the specific value of -1000 if you wish to output summarized results, or any other number to output detailed results. Other effective load IDs are either the specific number of a load ID that has been used to load Transactional Costing data, or alternatively, any negative number to use the latest load for the active model.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Calculate Transactional Costs**.
2. Enter the Load ID.

Example: Load ID containing specific value of -1000

```
PCMCONSOLE username=user password=pass
    open=TC_model calculatetransactionalcosts=-1000
```

Result: Calculates the final unit rates in a Transactional Costing model, using the latest load ID available from the PP_FASTLOAD table, appends them to the PPHV_UNITRATE table, and also outputs summarized detail cost value results to the PPR_HVDETAILCOST table.

Example: Load ID containing a non-specific negative value

```
PCMCONSOLE username=user password=pass
    open=TC_model calculatetransactionalcosts=-1
```

Result: Calculates the final unit rates in a Transactional Costing model, using the latest load ID available from the PP_FASTLOAD table, appends them to the PPHV_UNITRATE table, and also outputs detailed cost value results to the PPR_HVBREAKDOWN and PPR_HVBREAKDOWNDETAIL tables.

5.5 Clear Locked Results (clearlockedresults)

Description:

- Removes results for the specified locked Version and Period combination.

Usage:

- `clearlockedresults`=<version>,<period>

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<version>	Required	Version name.
<period>	Required	Period name.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Clear Locked Results**.
2. Enter the required **Version**.
3. Enter the required **Period**.

5.6 Clear Version Period Calculation List (`clearvpcalculationlist`)

Description:

- Optional command.
- The Transactional Costing model must be open.
- Marks for deletion all Versions/Periods combinations in the `PP_HVVERSIONPERIODCALCLIST` table, which holds a list of Versions/Periods combinations for which transactional costing calculations must take place. (For further details refer to the *Profitability and Cost Management Transactional Costing User Guide*.)

Usage:

- `clearvpcalculationlist`

There are no parameters associated with this command.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Clear Version Period Calculation List**.

5.7 Close Model (`closemodel`)

Description:

- Removes the connection to the currently open model. The model may remain open on the server if other users are still connected to it, otherwise it will be closed.

Usage:

- `closemodel`

There are no parameters associated with this command.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Close Model**.

5.8 Command File (`cmdfile`)

Description:

- Creates a placeholder for the contents of a specified command file. (See the topic “Command files” for further information).
- Commands in a command file are executed in the order in which they appear, except for the `cmdfile` command, which is the first operation to be executed. Its action is to insert the commands contained in the specified command file in an ordered list within the ordered list of commands to be executed sequentially.

Usage:

- `cmdfile`=<fileName>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<fileName>	Required	Location of an existing command file.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Command File**.
2. Enter the location of an existing command file.

Example:

Contents of `command file1.ini`:

```
username=User1
password=password
open=Model1
```

Contents of `command file2.ini`:

```
cmdfile=C:\command file1.ini
export=export Model1.csv,Replace,C:\my export.esp,ExcludeRulesValues,COMMA,DEFAULT
ALIAS,2,ANSI
```

When Console executes the script in a command file, its first action is to prepare the executable. To execute the `command file2.ini` command file, it first inserts the contents of the `command file1.ini` specified in the `cmdfile` command, as follows:

```
PCMCONSOLE username=User1 password=password open=Model1
 "export=export Model1.csv,Replace,C:\my export.esp,ExcludeRulesValues,COMMA,DEFAULT
ALIAS,2,ANSI"
```

Now the executable is ready, Console executes each of the commands in sequence to complete the following operations:

- User1 is logged in using the supplied password.
- Model1 is opened.
- Model1 is exported according to the export specification file.

Related Topics

- Command files

5.9 Comment (;)

Description:

- Used to start a single line comment.

Usage:

- In the command-line, comments are preceded by a semi-colon (;).

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Comment**.
2. Enter the required comment.

5.10 Connect to Work Manager (wmconnect)

Description:

- Connects Console to Work Manager. Both the user name and password (or encrypted password) commands are required before this, and must be valid login credentials.

Tip:
This is useful in situations where you wish to ensure that you have established a connection, for example, before creating a model.

Usage:

- `wmconnect`

There are no parameters associated with this command.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Connect to Work Manager**.

Related Topics

- Work Manager commands

5.11 Copy Model (`copymodel`)

Description:

- Description: Creates a copy of the specified model. All the items, values and “Books” created in the existing model are reproduced in the duplicate.

  Note:
  The new model name must be unique and must not contain any disallowed characters.

Usage:

- `copymodel`=<srcModel>, <destModel>

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<srcModel>	Required	Name of the existing model.
<destModel>	Required	The new model name.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Copy Model**.
2. Enter the **Source Model Name**.
3. Enter the **Destination Model Name**.

5.12 Create a New Model (`newmodel`)

Description:

- Creates a new model of the specified type with the specified name.

Usage:

- `newmodel`=<modelName>,[turnOnAudit],[modelType],[modelServer]

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<modelName>	Required	The name of the model to be created.
[turnOnAudit]	Optional	The value YES switches on auditing for the new model when it is created. The default setting is NO.
[modelType]	Optional	The model type values are: • Profitability and Costing = PCMMODEL (default) • Objectives and Metrics = OMMODEL • Transactional Costing = TCMODEL
[modelServer]	Optional	Parameter for the name of the server on which to create the model, for example if the secondary model server is required. **Note:** If no server name is specified (or the name is invalid / unreachable), this defaults to the default model server.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Create a New Model**.
2. Enter the **New Model Name**.
3. Select the **Audit Model** checkbox to switch on audit. This is only applicable if global auditing has been enabled via the Profitability and Cost Management Configuration Wizard.
4. In the **Model Type** list, select from the options of Profitability and Costing, Objectives and Metrics or Transactional Costing. If Model Type is not specified, a Profitability and Costing model is created by default.
5. Enter the **Model Server**, if required. This gives you the option to specify a server on which to create the model, for example the secondary model server. If no server name is specified (or the name is invalid / unreachable), this defaults to the default model server.

Example:

```
PCMCONSOLE username=user password=pass
    newModel=Model open=Model import=c:\exports\MyExport.xml calculate
```

Result: Creates a new model, then imports and calculates the new model.

5.13 Delete Model (`delete`)

Description:

- Deletes the specified model, if it exists, and is not currently being used. Optionally specifies whether to delete associated audit files and layouts.
- If the delete cannot occur because the model is still open for example, then the Console will continually retry for the specified number of timeout minutes before reporting a failure to delete. If the model closes during the waiting time, then it is deleted and the command file continues normally.

Usage:

- `delete`=<modelName>,[removeAudit],[removeViews],[timeoutValue]

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<modelName>	Required	The name of the model to be deleted.
[removeAudit]	Optional	The value YES deletes all audit records when deleting the model. The default value is NO.
[removeViews]	Optional	The value YES deletes all associated layouts when deleting the model. The default value is NO.
[timeoutValue]	Optional	Parameter for the number of minutes for which Console should continue to retry deleting the model. The default value is 0.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Delete Model**.
2. Enter the **Model Name to delete**.
3. To specify a number of minutes for which Console should continue to retry deleting the model, select a number in **Timeout (minutes)**.
4. Select the **Delete Audit** checkbox to remove associated audit files.
5. Select the **Delete Views** checkbox to remove associated views.

Example:

```
PCMCONSOLE username=user password=pass
    delete=Model1
```

Result: Deletes a model named Model1.

5.14 Encrypted Password (encryptedpassword)

Description:

- Sets the password in an encrypted format for the Profitability and Cost Management user account already specified. This may be used as an alternative to Set Password in an environment where it is preferred that user passwords are not stored as plain text. A username must also be set when this parameter is used (See “Set username”).
- For use in the command-line, first generate the password in an encrypted format, by using the Console Wizard function.

Usage:

- `encryptedpassword` =<password>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<password>	Required	The encrypted version of the password.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Encrypted Password**.
2. Enter the **Password** in plain text. Console Wizard encrypts the password that you have entered and displays this in the **Parameter List**.

Related Topics

- Set Username (username)

5.15 Execute Data Loader Routines (executedlroutines)

Description:

- The model must be open.

- For users setting up a large model, this command initiates the “Execute Data Loader Routines” procedure for Microsoft SQL Server or Oracle, depending on your installation. This procedure executes the Structural, Alias and Value Load procedures/packages in the correct order, without having to run them individually. (For further information, refer to the *SAP BusinessObjects Profitability and Cost Management Database Guide for Microsoft SQL Server* or the *SAP BusinessObjects Profitability and Cost Management Database Guide for Oracle*.)

Usage:

- `executedlroutines`=<loadId>,<replaceData>,<delta>,<target>,[erase],[sum]

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<loadId>	Required	This is the load identifier (load ID) of the individual data load from `PP_FASTLOAD`.
<replaceData>	Required	This specifies whether to replace attribute hierarchy settings. The value YES removes all existing attribute mappings and replaces with new data. The value NO appends new attribute hierarchy mappings without changing existing data.
<delta>	Required	Values allowed are YES or NO. The value YES specifies to load delta data.
<target>	Required	Values allowed are YES or NO. The value YES specifies to load target data.
[erase]	Optional	The supplied value specifies whether to replace data values for Version and Period: • 0 = Add new data items and overwrite any existing data in the model with the values from the load table. Non-matching data is left unchanged. • 1 = If data already exists in the model for the same Version and Period as in the load table, the data values in the model for this Version / Period combination are removed and replaced with the load table data. • 2 = If data for this load table already exists in the model, then the data values are removed for all Versions and Periods, before the load table values are inserted. The default value is 0.
[sum]	Optional	This parameter sums data for corresponding values. • YES = If the load table contains duplicate records for a data value, then the values are summed together with any values already existing in the model, and this new total value is loaded into the model. • NO = The procedure adds new data items and overwrites any existing data in the model with the values from the load table. Non-matching data is left unchanged. Duplicate key items are marked as duplicates and not loaded. The default value is NO.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Execute Data Loader Routines**.
2. Enter the **Load ID** of an individual data load.
3. Select the **Replace Data** checkbox to remove all existing attribute mappings and replace with new data. If unselected, new attribute hierarchy mappings are appended without changing existing data.
4. Select the **Delta Data** checkbox to load delta data.
5. Select the **Target Data** checkbox to load target data.

6. **Erase Data** specifies whether to replace data values for Version and Period. Select an option from the list (if no selection is made, the default value of 0 is set):
   - **0 - Append new data values**: Add new data items and overwrite any existing data in the model with the values from the load table. Non-matching data is left unchanged .
   - **1 - Erase data for same Version/Period first**: If data already exists in the model for the same Version and Period as in the load table, the data values in the model for this Version / Period combination are removed and replaced with the load table data.
   - **2 - Erase data for all Version/Period first**: If data for this load table already exists in the model, then the data values are removed for all Versions and Periods, before the load table values are inserted.
7. Select the **Target Sum** checkbox to sum data for corresponding values.
   - If the checkbox is selected and the load table contains duplicate records for a data value, then the values are summed together with any values already existing in the model, and this new total value is loaded into the model.
   - If the checkbox is left unselected, the procedure adds new data items and overwrites any existing data in the model with the values from the load table. Non-matching data is left unchanged. Duplicate key items are marked as duplicates and not loaded.

Example:

```
PCMCONSOLE username=user password=pass
    open=PCMmodel executedlroutines=1000,YES,NO,NO
```

Result: Runs the “Execute Data Loader Routines” procedure for Load ID 10001. It removes all existing attribute mappings and then loads all the new data. No delta data or target data is loaded.

5.16 Export Model to File (`export`)

Description:

- Exports the currently open model to either an XML file or comma-separated values (CSV) file as specified.

Note:
You are advised to define a specific export specification file, beacuse the default of ALL tables may take an exceptionally long time.

Usage:

- `export`=<fileName>,[appendData],[exportSpec], [includeRules],[delimiter],[aliasName],[precision],[unicode]

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<fileName>	Required	A valid filename and path location for the file to be exported (`.xml` or `.csv` file type).
[appendData]	Optional	Specify whether to replace any file that may already exist that has the same filename as the current export, or to append the current export to the existing file. Values allowed are APPEND, REPLACE or OVERWRITE (default). The file can be appended to by the use of APPEND, or by using a predefined `*.ESP` specification file (an export specification is defined within the **Model > Import/Export** screens in the Model Builder application).
[exportSpec]	Optional	Use settings from a predefined export specification file `(.ESP)` to determine which elements to export, or export ALL structure, data, security, and results. If no `(.ESP)` is supplied, the default is ALL_TABLES, which exports ALL tables. **Note:** Exporting all results tables may take a long time for large models.
[includeRules]	Optional	Specify whether to include any result values that are generated by rules. Values allowed are EXCLUDERULESVALUES (default) or INCLUDERULESVALUES. **Note:** If you have already selected a specific ESP file, rule-based values can be selected within it. This option is only appropriate when you have not selected a specific ESP file.
[delimiter]	Optional	Specify a particular delimiter for the export file to use. If a delimiter is not specified, the default is COMMA.
[aliasName]	Optional	This must be an existing Profitability and Cost Management Alias name that is available in the model being exported. The deafult is DEFAULT ALIAS, which selects the value DEFAULT ALIAS.
[precision]	Optional	Set the number of decimal places in exported values. The default value is 12.
[unicode]	Optional	Choose to export data to a CSV file as either ANSI or Unicode. Values allowed are ANSI (default) or UNICODE. This allows you to export model data containing Unicode characters such as Asian symbols, for example.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Export Model to File**.

2. In **Export to file**, specify the file name to export to and whether it is an XML file. If you add the filetype `.xml`, then an XML file will be generated, otherwise it will default to a comma-separated values (CSV) file.
3. Select the **Append data to this file** checkbox to append this export to any existing file. The default is to overwrite existing files of the same name.
4. If the **Include Rules Values** checkbox is selected, then any values that have been calculated by rules will also be exported; otherwise only input data values, Activity Costs and Cost Object Costs will be exported.
5. In **Export specification file**, you can select the location of an export specification file (*.ESP) that defines the export parameters.

   Note:
   You are advised to define a specific export specification file, because the default of ALL_TABLES exports EVERYTHING.

6. By default, a comma is used to delimit exported fields. You can use **Delimiter** to specify a particular delimiter for the export file to use. This option is only relevant for CSV file exports.
7. In **Alias Name**, you can specify a Data Alias name if the exported data is to appear as that alternative name.
8. You can specify the number of decimal places for exported values in **Decimal Precision**. The default value is 2.
9. You can choose export data to a CSV file as either ANSI or Unicode. If the **Unicode File** checkbox is selected, this option allows you to export model data containing Unicode characters such as Asian symbols, for example.

Example:

```
PCMCONSOLE username=user password=pass
    open=Model "export=c:\MyExport.xml,Overwrite,All"
```

Result: Exports everything from a model, overwriting any existing file.

Example:

```
PCMCONSOLE username=user password=pass
    open=Model
    "export=c:\AnExport.csv,Append,C:\ExportSpec.esp,ExcludeRulesValues,TAB,EPO
    CODE"
```

Result: Exports only those items which are included in a specification file. (CSV format, excluding rules, tab delimited file, using data alias EPO CODE)

5.17 Export Results (`exportresults`)

Description:

- Exports results data in the currently open model to SAP BusinessObjects “Universe” tables. You may optionally select the location of an export specification file (`*.ESP`) that determines which results

tables to export. If an export specification file is not listed, all the model's results tables will be exported to “Universe” tables.

Note:
You are advised to define a specific export specification file, since the default of ALL_TABLES will export EVERYTHING.

Usage:

- `exportresults`=<exportSpec>,[aliasName]

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<exportSpec>	Required	The name of the export specification file (`*.ESP`) is optional. If an export specification file is not supplied, the default is ALL_TABLES, which exports all the model's results tables to “Universe” tables.
[aliasName]	Optional	This must be an existing Profitability and Cost Management Alias name that is available in the model being exported. The deafult is DEFAULT ALIAS, which selects the value DEFAULT ALIAS.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Export Results**.
2. Enter the location of an export specification file (`*.ESP`) that determines which results tables to export.

   Note:
   You are advised to define a specific export specification file, because the default of ALL tables will export EVERYTHING.

5.18 Export to Database (`exporttodb`)

Description:

- This command exports results data in the currently open model to a database table. The export specification file (`*.ESP`) is optional. If an export specification file is not listed, then all the models results tables will be exported to database.

Note:
See the "Model Statistics" screen in Model Builder for guidance on the number of possible calculations (and hence the size of the data) that can be exported during this process.

Usage:

- `exporttodb`=[fileName]

The following parameter can be used with this command:

Parameter	Required / Optional	Description
[fileName]	Optional	You may optionally provide the location of an export specification file (`*.ESP`) that determines which tables and dimensions to export.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Export To Database**.
2. You may optionally select the location of an export specification file (`*.ESP`) that determines which tables and dimensions to export.

5.19 Import XML File into Model (`import`)

Description:

- Selects an existing XML model file to be imported into the currently open model. XML files imported into a model with existing structure and data will only append additional information. Optionally specifies whether to allow new structure to be created that does not already exist in the model.

Usage:

- `import`=<fileName>,[createStructure]

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<fileName>	Required	Name of an existing XML model file.
[createStructure]	Optional	The value YES allows new structure to be created that does not already exist in the model (this is the default). The value NO does not allow new structure to be created.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Import XML File into Model** .
2. Enter the name of an existing XML model file.
3. Select the **Allow creation of structure** checkbox to allow new structure to be created that does not already exist in the model.

Example:

```
PCMCONSOLE username=user password=pass
    delete=Model1 newModel=Model2 import=c:\exports\MyExport.xml
```

Result: Deletes a model, creates a new model, and imports structure and data.

5.20 Load Model (`loadmodel`)

Description:

- Creates a connection to an existing model using the account specified, for the duration of the Console run.
- Enables the Console executable to run with connections to multiple models (unlike the `open` command) beacuse `loadmodel` allows Console to have connections with more than one model at a time. This is useful when a Console run is dealing with more than one model, to ensure that each model is kept open.

Usage:

- `loadmodel`=<modelName>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<modelName>	Required	Name of existing model.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Load Model**.
2. Enter the name of the model to connect to.

5.21 Load Stored Values (`loadstoredvalues`)

Description:

- Clears existing results and loads any cross-model rule values from associated models.

  There are two types of cross model values:

  - Models that contain dynamic values - This type of model automatically detects and reloads any changed values in dependent models.
  - Models that contain stored model values (Non-dynamic) - This type of model contains a static set of data from dependent models, so that changed values are not detected or reflected in the master model.

In the case of refreshing stored model values in a master model, the `loadstoredvalues` command can be executed to remove any calculated results, and then obtain an update of all values in any dependent models that are required.

Note:
Because `loadstoredvalues` clears any existing (non-locked) results in a model, it is recommended to run `loadstoredvalues` BEFORE you run the `calculate` command. This applies particularly to models that contain rules that use **Load Stored Values**.

Usage:

- `loadstoredvalues`

There are no parameters associated with this command.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Load Stored Values**.

Example:

```
PCMCONSOLE USERNAME=<user>
PASSWORD=<password>
OPEN =<modelname>
LOADSTOREDVALUES
```

5.22 Load TC Calculation Only (`loadtccalconly`)

Description:

- The Transactional Costing model must be open.
- Performs transactional costing mappings and recalculates Activity Driver values from the transactional Activity Driver volumes without loading any data from the `PPLOAD_HVACTIVITYDRIVER` table.
- The load ID parameter is the load ID of the individual data load. If a negative value is entered, the maximum load ID is obtained from the `PP_LOADFAST` table for the active model. If you pass in a specific load ID, it will process data only where a load ID matches. (For further details refer to the *Profitability and Cost Management Transactional Costing User Guide*.)

Usage:

- `loadtccalconly` =<loadId>

The following parameters can be used with this command:

Parameter	Required / Optional	Description
<loadId>	Required	The load ID of the individual data load.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Load TC Calculation Only**.
2. Enter the load ID of the individual data load.

5.23 Load Volume Data (`loadvolumedata`)

Description:

- Loads transactional Activity Driver data volumes into the currently open Transactional Costing model. It uses the supplied load ID, and generates Activity Driver values. (For further details refer to the *Profitabiltiy and Cost Management Transactional Costing User Guide*.)
- If the load ID is a negative value such as -1, then the maximum load ID for the active model is used, which is the very latest load.

Usage:

- `loadvolumedata`=<loadId>

The following parameters can be used with this command:

Parameter	Required / Optional	Description
<loadId>	Required	The supplied load identifier (load ID) of the individual data load that correlates to the load ID in the `PPLOAD_HVACTIVITYDRIVERVOLUME` staging table.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Load Volume Data**.
2. Enter the supplied load identifier (load ID) of the individual data load that correlates to the load ID in the `PPLOAD_HVACTIVITYDRIVERVOLUME` staging table.

Example:

```
PCMCONSOLE username=user password=pass
    open=TC_model loadvolumedata=-1
```

Result: Imports Transactional Costing data into a model, using the latest load ID available from the `PP_FASTLOAD` table, and generates Activity Driver values from the transactional Activity Driver volumes.

5.24 Lock Results (`lockresults`)

Description:

- Locks the specified Version and Period combination.

Usage:

- `lockresults`=<version>,<period>

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<version>	Required	Version name.
<period>	Required	Period name.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Lock Results**.
2. Enter the required **Version**.
3. Enter the required **Period**.

5.25 Log All Events to Text File (`log`)

Description:

- This command produces a text file of recorded actions that took place during the last issued series of commands (command line or file). The filename and location must be specified and be available (that is, the user must have “write” permissions to the specified location). This command can be specified anywhere in the command file or command line and will apply to all actions. In addition, any errors encountered during the processing of the command file will be recorded in this log file.

Usage:

- `log`=<fileName>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<fileName>	Required	Specifies a name and location for retaining a log of the command parameters processed by the Console Wizard.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Log All Events to Text File**.
2. Enter a filename and path location to write the log file.

Example:

```
PCMCONSOLE username=user password=pass
    open=Model databridgeimport=c:\ADBImportspec.ctf prefixdate
    log=c:\MyLog.txt
```

Result: Imports data into an existing model using a Data Bridge control file, and records the time taken.

5.26 Logoff (`logoff`)

Description:

- Completely severs any client connection in the running system for the required logged in user(s) and logs them off. The following message appears in the client instance: `Your session has been manually terminated by an authoritative user`. The functionality provided by this command is the same as **Logoff** in the User Monitor utility. For further information, see “User Monitor” in the *SAP BusinessObjects Profitability and Cost Management Administrator's Guide*.

  Note:
  To execute any further commands after `logoff`, ensure that a different user has been logged in for the Console executable than any user being logged off.

Usage:

- `logoff`=<subcommand>,[subcommandParam]

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<subcommand>	Required	Logs off user(s) depending on the supplied parameters: • ALLUSERS - logs off all logged-on users in the system • USER - logs off the specified user • GROUP - logs off all logged-on users in the specified group • MODEL - logs off all users currently logged into the specified model
[subcommandParam]	Optional	Specify either a username, user group name or model name, depending on the selected subcommand. This parameter is not required for ALLUSERS.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Logoff**.
2. In the **Logoff Type** list:
   - to log off all logged-on users in the system, select **All Users**.
   - to log off a specified user, select **Specific User**, then enter the **User Name**.
   - to log off all logged-on users in a specified group, select **Specific Group of Users**, then enter the **Group Name**.
   - to log off all users currently logged into a specified model, select **All Users of a Specific Model**, then enter the **Model Name**.

5.27 Model Description (`modeldescription`)

Description:

- Specify or amend an existing description that is displayed against the model in Model Builder.

Usage:

- `modeldescription`=<modelName>,<description>

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<modelName>	Required	The name of the model.
<description>	Required	New description of model.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Model Description**.

2. Enter the **Model Name**.
3. Enter the **Model Description**.

5.28 Model Enabled (`modelenabled`)

Description:

- Specifies whether the model is enabled or disabled. A model cannot be opened if disabled.

Usage:

- `modelenabled`=<modelName>,<newStatus>

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<modelName>	Required	The name of the model.
<newStatus>	Required	The value YES enables the model. The value NO disables the model.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Model Enabled**.
2. Enter the name of the model.
3. Select the **Model Enabled** checkbox to enable the model. If left unselected, the model will be disabled.

5.29 Model Server (`modelserver`)

Description:

- Changes the model server on which the model runs.

Usage:

- `modelserver`=<modelName>,<server>

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<modelName>	Required	The name of the model.
<server>	Required	The name of the model server.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Model Server**.
2. Enter the **Model Name**.
3. In the **Model Server Name**, enter the name of the model server to change the model to.

5.30 Open Model (`open`)

Description:

- Mandatory command before the following commands:
  - `AutomaticCalculation`
  - `Calculate`
  - `CalculateTransactionalCosts`
  - `ClearVPCalculationList`
  - `ExecuteDLRoutines`
  - `Export`
  - `ExportResults`
  - `ExportView`
  - `Import`
  - `LoadTCCalcOnly`
  - `LoadVolumeData`
  - `UpdateVPCalculationList`
- Opens an existing model (unless already opened with another user) and creates a connection to it using the account specified.
- Only one model at a time can have a connection to a Console executable using the `open` command. If a second `open` command is issued in the same executable, the first model is automatically closed before the second model is opened.

Usage:

- `open`=<modelName>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<modelName>	Required	Name of the model to be opened.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Open Model**.
2. Enter the name of the model.

Example:

```
PCMCONSOLE username=user password=pass
    open=Model1 calculate
```

Result: Opens a model named Model1.

5.31 Pause Processing for a While (`pause`)

Description:

- Pauses execution of the command file parameters until the specified number of minutes has elapsed. The execution of a command can be delayed by placing a pause command in the console script.
- The pause is executed with immediate effect (that is, the execution of the script is halted at the point when the `pause` command is issued).

Usage:

- `pause`=<minutes>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
[minutes]	Optional	Number of minutes to pause before executing the next command.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Pause Processing for a While**.
2. Enter the number of minutes to pause before executing the next command.

5.32 Prefix Date and Time (`prefixdate`)

Description:

- Places a date and time stamp before each command executed as part of the command file. This will be reported via the console screen (DOS prompt or command box) or recorded in the log file, if one has been specified (see the “Log all events to text file” command).
- This command can be issued at any position in the command file or command line, but applies to all actions (that is, all actions will be prefixed).

Usage:

- `prefixdate`

There are no parameters associated with this command.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Prefix Date and Time**.

Example:

```
PCMCONSOLE username=user password=pass
    open=Model databridgeimport=c:\ADBImportspec.ctf prefixdate
    log=c:\MyLog.txt
```

Result: Imports data into an existing model using a Data Bridge control file, and records the time taken.

Related Topics

- Log All Events to Text File (log)

5.33 Rename Model (`renamemodel`)

Description:

- Renames the specified model.

  Note:
  The new model name must be unique and not contain any disallowed characters.

- If the rename cannot occur because the model is still open for example, then the Console will continually retry for the specified number of timeout minutes before reporting a failure to rename. If the model closes during the waiting time, then it is renamed and the command file continues normally.

Usage:

- `renamemodel`=<srcModel>,<destModel>,[timeoutValue]

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<srcModel>	Required	The existing name of the model to be renamed.
<destModel>	Required	The new name for the model.
[timeoutValue]	Optional	The number of minutes for which Console should continue to retry renaming the model. The default value is 0.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Rename Model**.
2. In **Source Model Name**, enter the existing name of the model.
3. In **Destination Model Name**, enter the new name for the model.
4. In **Timeout (minutes)**, select the number of minutes for which Console should continue to retry renaming the model.

5.34 Run Data Bridge Import (`databridgeimport`)

Description:

- This command executes a Data Bridge Control File (`*.CTF`) to import data into the specified model. The full path to the file is required unless the file is in the current working directory.
- Optionally specifies whether to allow new structure to be created that does not already exist in the model.

Note:
If more control of the data import is required, then please refer to the *Data Bridge User Guide* for information about the DBCONS Data Bridge automation utility.

Usage:

- `databridgeimport`=<fileName>,[allowCreate]

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<fileName>	Required	Name of (`*.CTF`) file to import.
[allowCreate]	Optional	The value YES allows new structure to be created that does not already exist in the model (this is the default) . The value NO does not allow new structure to be created.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Run Data Bridge Import**.
2. Enter the name of the (`*.CTF`) file to import.
3. Select the **Allow creation of structure** checkbox to allow new structure to be created that does not already exist in the model.

Example:

```
PCMCONSOLE username=user password=pass
    open=Model databridgeimport=c:\ADBImportspec.ctf prefixdate
    log=c:\MyLog.txt
```

Result: Imports data into an existing model using a Data Bridge control file, and records the time taken.

5.35 Send Work Manager Event (`wmsendevent`)

Description:

- Sends the event name to the Work Manager process instance, to enable its progess to be tracked.

Usage:

- `wmsendevent`=<eventName>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<eventName>	Required	Event name.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Send Work Manager Event**.
2. Enter the name of the **Work Manager Event**.

Related Topics

- Work Manager commands

5.36 Set Password (`password`)

Description:

- Sets the password for the Profitability and Cost Management user account already specified . A username must also be set when this parameter is used (See “Set username”). See “Encrypted Password” for use in a secure environment.

Usage:

- `password`=<password>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<password>	Required	Password

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Set Password**.
2. Enter the **Password**.

Related Topics

- Encrypted Password (encryptedpassword)
- Set Username (username)

5.37 Set Username (`username`)

Description:

- Sets the Profitability and Cost Management user account whose credentials will be used to perform actions (such as opening a model). A password must be set for this user.

  Note:
  Single Sign On is not available for Console. If you are working in a BusinessObjects Enterprise environment, you are not able to use the login credentials of an imported BusinessObjects Enterprise user, and you should create a valid Profitability and Cost Management user. Alternatively, you may convert an imported BusinessObjects Enterprise user by performing **Reset Password** in Model Builder (via **Tools** > **Security** > **Users and Groups**). This sets a Profitability and Cost Management password, enabling the user name to be used in Console also.

Usage:

- `username`=<userName>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<userName>	Required	The user name

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Set Username**.
2. Enter the **User Name**.

Example:

```
PCMCONSOLE username=User1 password=pass open=BIKES
```

Result: Opens a model with a user named User1.

5.38 Sort Dimension (sortdimension)

Description:

- Sorts dimension items in a model hierarchy into ascending or descending alphabetical order.

Usage:

- `sortdimension`=<Dimension>,<DataAlias>,<SortDirection>

The following parameters can be used with this command:

Parameters	Required/Optional	Description
<Dimension>	Required	The dimension to be sorted for this command
<DataAlias>	Required	The data alias by which to sort the dimension **Note:** You cannot sort by the EPO CODE alias
<SortDirection>	Required	The direction in which to sort the dimensions, ascending order for example. Allowed values are ASC and DESC.

In the Console Wizard, in the "Command properties" dialog box, proceed as follows:

1. Select **Sort Dimension** in the **Command Type** list.
2. Enter the dimension you want to sort in the **Dimension** box.
3. Enter the data alias by which to sort in the **Data Alias** box.
4. Select either **Sort Ascending** or **Sort Descending** to specify the sort order.

5.39 Unique Model ID (`uniquemodelid`)

Description:

- Changes the model property Unique Model ID (UMID). This can be used to ensure that users can track and locate data that originates from the Financial Information Management application, even after a model has been renamed, copied, imported or exported.

Usage:

- `uniquemodelid`=<id>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<id>	Required	The Unique Model ID.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Unique Model ID**.
2. Enter the Unique Model ID (UMID).

5.40 Unload Model (`unloadmodel`)

Description:

- Removes the connection to the specified model. The model may remain open on the server if other users are still connected to it, otherwise it will be closed.
- Use the `unloadmodel` command to remove a connection that was created with the `loadmodel` command.

Usage:

- `unloadmodel`=<modelName>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<modelName>	Required	Name of model with which to break connection.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Unload Model**.

2. Enter the **Model Name**.

Related Topics

• Load Model (loadmodel)

5.41 Unlock Results (unlockresults)

Description:

• Unlocks the specified locked Version and Period combination.

Usage:

• unlockresults=<version>,<period>

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<version>	Required	Version name.
<period>	Required	Period name.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Unlock Results**.
2. Enter the required **Version**.
3. Enter the required **Period**.

5.42 Update Version Period Calculation List (updatevpcalculationlist)

Description:

• The Transactional Costing model must be open.

• Adds or marks for deletion a Versions/Periods combination in the PP_HVVERSIONPERIODCALCLIST table, which holds a list of Versions/Periods combinations for which transactional costing calculations must take place. (For further details refer to the *Profitability and Cost Management Transactional Costing User Guide*.)

Usage:

• updatevpcalculationlist=<version>, <period>,<addTo>

The following parameters can be used with this command:

Parameters	Required / Optional	Description
<version>	Required	Version name.
<period>	Required	Period name.
<addTo>	Required	Allowed values are ADD or DELETE, to specify whether to add or mark for deletion.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Update Version Period Calculation List**.
2. Enter the required **Version**.
3. Enter the required **Period**.
4. Select either the **Add** or **Delete** radio button.

5.43 Wait Keep Process Running (`wait`)

Description:

- Keeps a model open for the specified number of minutes. When the script terminates without this command, it would cause the model to close. This can be used to wait a certain time before terminating, and is useful to keep a model open for a period of time (up to 24 hours).
- Irrespective of at which point the `wait` command is issued; the waiting begins at the end of a console command set. For example, if a `calculate` command came after a `wait` command, but the calculation took an hour, the `wait` command would not commence until after the hour-long calculation.

Usage:

- `wait`=<minutes>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
<minutes>	Required	Number of minutes to keep Console running before shutting down.

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Wait, Keep Process Running for Certain Time**.
2. Enter the number of minutes to keep Console running before shutting down.

Example:

```
PCMCONSOLE username=user password=pass
    open=Model calculate wait=240
```

Result: Opens a model, calculates it, and holds the model open for 4 hours.

5.44 Write a Message (msg)

Description:

- Displays a text message in the DOS box in which the command is run.
- The message appears as a line of text between information automatically displayed by Console.

Usage:

- msg=<message>

The following parameter can be used with this command:

Parameter	Required / Optional	Description
[message]	Optional	The message text must be a string starting and ending with double quotes. The string must not include a double quote (use single quotes instead).

In Console Wizard, in the "Command properties" dialog, proceed as follows:

1. In the **Command Type** list, select **Write a Message**.
2. Enter the message to be displayed.

Console error messages

Console error text messages are generated in the format `ERROR - n` where the number `n` indicates the command type of the failed instruction, as listed in the table.

Number	Command name
0	None
1	Set Username
2	Set Password
3	Encrypted Password
4	Delete Model
5	Open Model
6	Create a New Model
7	Create a New Model
8	Send Work Manager ID
9	Send Work Manager Event
10	Import XML File into Model
11	Calculate Model
12	Export Model to File
13	Log all Events to Text File
14	Export to Database
15	Command File
16	Wait, Keep Process Running
17	Write a Message
18	Pause Processing for a While
19	Load Stored Values
20	Prefix Date and Time
21	Run Data Bridge Import
22	Automatic Calculation

Number	Command name
23	Export Results
24	Lock Results
25	Unlock Results
26	Clear Locked Results
27	Load Volume Data
28	Calculate Transactional Costs
29	Update Version Period Calculation List
30	Clear Version Period Calculation List
31	Load Transactional Cost Calculation Only
32	Execute Data Loader Routines
33	Connect to Work Manager
34	Delete Command File After Use
35	Copy Model
36	Model Description
37	Model Server
38	Rename Model
39	Close Model
40	Load Model
41	Unload Model
42	Audit Enabled
43	Model Enabled
44	Unique Model ID
45	Logoff

More Information

Information Resource	Location
SAP BusinessObjects product information	http://www.sap.com
SAP Help Portal	Select http://help.sap.com > SAP BusinessObjects. You can access the most up-to-date documentation covering all SAP BusinessObjects products and their deployment at the SAP Help Portal. You can download PDF versions or installable HTML libraries. Certain guides are stored on the SAP Service Marketplace and are not available from the SAP Help Portal. These guides are listed on the Help Portal accompanied by a link to the SAP Service Marketplace. Customers with a maintenance agreement have an authorized user ID to access this site. To obtain an ID, contact your customer support representative.
SAP Service Marketplace	http://service.sap.com/bosap-support > Documentation • Installation guides: https://service.sap.com/bosap-instguides • Release notes: http://service.sap.com/releasenotes The SAP Service Marketplace stores certain installation guides, upgrade and migration guides, deployment guides, release notes and Supported Platforms documents. Customers with a maintenance agreement have an authorized user ID to access this site. Contact your customer support representative to obtain an ID. If you are redirected to the SAP Service Marketplace from the SAP Help Portal, use the menu in the navigation pane on the left to locate the category containing the documentation you want to access.
Developer resources	https://boc.sdn.sap.com/ https://www.sdn.sap.com/irj/sdn/businessobjects-sdklibrary
SAP BusinessObjects articles on the SAP Community Network	https://www.sdn.sap.com/irj/boc/businessobjects-articles These articles were formerly known as technical papers.

Information Resource	Location
Notes	https://service.sap.com/notes These notes were formerly known as Knowledge Base articles.
Forums on the SAP Community Network	https://www.sdn.sap.com/irj/scn/forums
Training	http://www.sap.com/services/education From traditional classroom learning to targeted e-learning seminars, we can offer a training package to suit your learning needs and preferred learning style.
Online customer support	http://service.sap.com/bosap-support The SAP Support Portal contains information about Customer Support programs and services. It also has links to a wide range of technical information and downloads. Customers with a maintenance agreement have an authorized user ID to access this site. To obtain an ID, contact your customer support representative.
Consulting	http://www.sap.com/services/bysubject/businessobjectsconsulting Consultants can accompany you from the initial analysis stage to the delivery of your deployment project. Expertise is available in topics such as relational and multidimensional databases, connectivity, database design tools, and customized embedding technology.

Index