

---

# EnterpriseOne Supply Chain Planning Sales and Operations Planning 8.12 Implementation Guide

---

**June 2006**

Copyright © 2006, Oracle. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

### **U.S. GOVERNMENT RIGHTS**

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are “commercial computer software” or “commercial technical data” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software–Restricted Rights (June 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee’s responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

### **Open Source Disclosure**

Oracle takes no responsibility for its use or distribution of any open source or shareware software or documentation and disclaims any and all liability or damages resulting from use of said software or documentation. The following open source software may be used in Oracle’s PeopleSoft products and the following disclaimers are provided.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>). Copyright © 1999-2000 The Apache Software Foundation. All rights reserved. THIS SOFTWARE IS PROVIDED “AS IS” AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# Contents

## Preface

<b>EnterpriseOne Sales and Operations Planning Preface.....</b>	<b>xv</b>
Related Documentation.....	xv
Obtaining Documentation Updates.....	xv
Ordering Printed Documentation.....	xv
Typographical Conventions and Visual Cues.....	xvi
Typographical Conventions.....	xvi
Visual Cues.....	xvi
Comments and Suggestions.....	xvii

## Chapter 1

<b>Getting Started with EnterpriseOne Sales and Operations Planning.....</b>	<b>1</b>
EnterpriseOne Sales and Operations Planning Overview.....	1
EnterpriseOne Sales and Operations Planning Automation Shell.....	2
EnterpriseOne Sales and Operations Planning Business Processes.....	2
Sales and Operations Planning Integrations.....	4
EnterpriseOne Supply Chain Business Modeler.....	4
EnterpriseOne Sales and Operations Planning Implementation.....	5

## Chapter 2

<b>Understanding Sales and Operations Planning.....</b>	<b>7</b>
Sales and Operations Planning Process.....	7
The Sales and Operations Planning Workflow.....	9
Sales and Operations Planning Workflow Overview.....	9
Current Planning Cycle.....	11
Plan Import.....	11
Assign Plans to Views.....	11
Demand Plan Review.....	11
Supply Review.....	13
Financial Reconciliation.....	14
Senior Management Meeting.....	15
Sales and Operations Planning User Interface.....	15

## Chapter 3

<b>Understanding Views.....</b>	<b>17</b>
Views Overview.....	17
View Permissions.....	18
View Manager.....	19
Display Types and View Formats.....	19
Data Views and Metrics.....	20
Actual Versus Plan Data.....	22
Plan Data Views.....	22
Demand Plan View.....	23
Production Plan View.....	24
Inventory Plan View.....	24
Rough Cut Capacity Plan View.....	25
Combined Data Views.....	26
Consolidated View.....	26
Financial View.....	27
Display Types.....	28
Table Display Type.....	29
Graph Display Type.....	29
View Formats.....	31
View Format Overview.....	32
Default Format.....	32
Comparison Format.....	32
Dashboard Format.....	33
Waterfall Format.....	34
Units of Measure.....	35
Dimension Properties.....	36
Dimension Properties Overview.....	36
Dimension Property Names and Values.....	36
Targets and Thresholds.....	37

## Chapter 4

<b>Configuring EnterpriseOne Sales and Operations Planning.....</b>	<b>39</b>
Understanding Configuring Sales and Operations Planning.....	39
User Administration.....	39
Logs.....	41
Planning Periods.....	42
Planning Cycles.....	42
Custom Metrics.....	43



Administering Users.....	45
Pages Used to Administer Users.....	46
Adding Users.....	46
Deleting Users.....	46
Editing User Data.....	47
Changing Passwords.....	47
Administering User Roles.....	47
Page used to Administer User Roles.....	47
Adding User Roles.....	47
Editing User Roles.....	48
Deleting User Roles.....	48
Managing Planning Periods.....	48
Pages Used to Manage Planning Periods.....	48
Adding Planning Periods.....	48
Managing Planning Cycles.....	49
Pages Used to Manage Planning Cycles.....	49
Adding Planning Cycles.....	49
Deleting Planning Cycles.....	49
Setting the Current Planning Cycle.....	49
Managing Approvals.....	49
Working with Logs.....	50
Pages Used to Work with Logs.....	50
Clearing Log Events.....	50
Adding Log Patterns.....	51
Clearing All Log Patterns.....	51

## Chapter 5

### **Managing the Sales and Operations Planning Process using the Plan Calendar.....53**

Understanding the Plan Calendar.....53

Working with the Plan Calendar.....54

    Pages used to Work with the Plan Calendar.....55

    Adding Actions to the Plan Calendar.....56

    Adding Milestones to the Plan Calendar.....56

    Adding Meetings to the Plan Calendar.....56

## Chapter 6

### **Importing Data.....59**

Understanding Importing Data.....59

Data Import Overview.....	59
Import Data Requirements.....	60
Refresh Command Syntax.....	61
Understanding Exporting Data.....	62
Importing Data Using the Sales and Operations Planning Connector.....	62
Windows Used to Import Data Using the Sales and Operations Planning Connector.....	63
Importing Data Using the Sales and Operations Planning Connector.....	63

## Chapter 7

<b>Working with Views.....</b>	<b>65</b>
Understanding Working with Views.....	65
Targets and Thresholds.....	66
Working with Views.....	67
Pages Used to Work with Views.....	67
Creating New Views.....	67
Running Existing Views.....	68
Assigning Plans to Views.....	68
Assigning View Permissions.....	69
Adding Custom Metrics to Views.....	69
Managing Targets for Relative Metrics.....	69
Page Used to Manage Targets for Relative Metrics.....	70
Adding New Targets for Relative Metrics.....	70
Editing Targets for Relative Metrics.....	70
Changing the Order of Targets for Relative Metrics.....	70
Managing Targets for Absolute Metrics.....	71
Page Used to Manage Targets for Absolute Metrics.....	71
Adding New Targets for Absolute Metrics.....	71
Changing the Planning Period Group for Absolute Metrics.....	71
Manage Thresholds for Absolute Metrics.....	72
Page Used to Manage Thresholds for Absolute Metrics.....	72
Inserting New Threshold Rules for Absolute Metrics.....	72
Editing Threshold Rules.....	73
Changing the Order of Thresholds for Absolute Metrics.....	73

## Chapter 8

<b>Working with Reports.....</b>	<b>75</b>
Understanding Reports.....	75
Net Change Reports.....	75

Metric Exception Reports.....	76
Report Execution.....	77
Managing Net Change Reports.....	78
Pages Used to Manage Net Change Reports.....	78
Creating Net Change Reports.....	78
Editing Net Change Reports.....	79
Running Net Change Reports.....	79
Managing Metric Exception Reports.....	80
Pages Used to Manage Metric Exception Reports.....	80
Creating Metric Exception Reports.....	80
Editing Metric Exception Reports.....	81
Running Metric Exception Reports.....	81

## Chapter 9

<b>Using the Sales and Operations Planning Automation Shell.....</b>	<b>83</b>
Understanding the Sales and Operations Planning Automation Shell.....	83
Sales and Operations Planning Automation Shell Syntax.....	83
Understanding Sales and Operations Planning Workflows.....	84
Starting the Sales and Operations Planning Automation Shell.....	86
Procedures Used to Start the Sales and Operations Planning Automation Shell.....	87
Automating Business Processes Using Batch Scripts.....	87
Understanding Batch Scripts.....	88
Windows Used to Run Batch Scripts.....	89
Working with Sales and Operations Planning Workflows.....	89
Page Used to Work with Sales and Operations Planning Workflows.....	90
Adding Custom Workflows to Sales and Operations Planning.....	90
Sales and Operations Planning Tcl Command Reference.....	90
Tcl Commands by Category.....	90
Sales and Operations Planning Tcl Commands.....	100
App Archive Command.....	100
App Backup Command.....	101
App Clear Command.....	101
App DatabaseExists Command.....	102
App FixJavaClasspath Command.....	102
App GetCompressMode command.....	102
App GetDBDiskFree Command.....	103
App GetInstallLocation Command.....	103
App GetVersion Command.....	104
App LoadPropertyNamees Command.....	104

App Restore Command.....	104
App RunBackground Command.....	105
App SetCompressMode Command.....	105
App WritePropertyNames Command.....	106
Calendar AddAction Command.....	106
Calendar AddMeeting Command.....	107
Calendar AddMilestone Command.....	108
Calendar AddToCalendarFromXml Command.....	109
Calendar Clear Command.....	109
Calendar DeleteAction Command.....	110
Calendar DeleteMeeting Command.....	110
Calendar DeleteMilestone Command.....	111
Calendar GetAction Command.....	111
Calendar GetMeeting Command.....	112
Calendar GetMilestone Command.....	112
Calendar GetAllActions Command.....	112
Calendar GetAllMeetings Command.....	113
Calendar GetAllMilestones Command.....	113
Calendar UpdateAction Command.....	114
Calendar UpdateMeeting Command.....	114
Calendar UpdateMilestone Command.....	115
Calendar WriteCalendarToXml Command.....	116
Channels AddChannel Command.....	116
Channels AddChannelsFromXml Command.....	117
Channels CheckExists Command.....	118
Channels DeleteAllChannels Command.....	118
Channels DeleteChannel Command.....	118
Channels GetChannelCodes Command.....	119
Channels GetChannelProperties Command.....	119
Channels GetChannelPropertyNames Command.....	120
Channels SetChannelPropertyNames Command.....	120
Channels SetChannelProperties Command.....	120
Channels StrictUpdateChannelsFromXml Command.....	121
Channels UpdateChannelsFromXml Command.....	122
Channels WriteChannelsToXml Command.....	122
CustomMetrics AddCustomMetric Command.....	123
CustomMetrics DeleteCustomMetric Command.....	124
CustomMetrics ExportCustomMetric Command.....	124
CustomMetrics ExportCustomMetrics Command.....	125
CustomMetrics GetValues Command.....	125

CustomMetrics ImportCustomMetrics Command.....	126
CustomMetrics SetValue Command.....	127
CustomWorkflows ExecuteProcedure Command.....	128
CustomWorkflows GetLogEvents Command.....	128
CustomWorkflows GetOutput Command.....	129
Cycles AddCycle Command.....	129
Cycles AddCyclesFromXml Command.....	130
Cycles AddPlanMappingFromXml Command.....	131
Cycles Contains Command.....	131
Cycles DeleteAllCycles Command.....	132
Cycles DeleteApprovedPlan Command.....	132
Cycles DeleteCycle Command.....	132
Cycles GetApprovedPlanCyclesInfo Command.....	133
Cycles GetCurrentCycleName Command.....	134
Cycles SetApprovedPlan Command.....	134
Cycles SetCurrentCycle Command.....	134
Cycles StrictUpdateCyclesFromXml Command.....	135
Cycles UpdateCyclesFromXml Command.....	136
Cycles WriteCyclesToXml Command.....	136
Cycles WritePlanMappingToXml Command.....	137
DemandPlan AddDemand Command.....	137
DemandPlan AddDemandFromXml Command.....	138
DemandPlan AddDemandFromXmlToCycle Command.....	138
DemandPlan AddDemandPlan Command.....	139
DemandPlan ContainsId Command.....	140
DemandPlan DeleteAllDemands Command.....	141
DemandPlan DeleteAllDemandsForCycle Command.....	141
DemandPlan DeleteDemand Command.....	141
DemandPlan GetDemandIds Command.....	142
DemandPlan GetDemandPlanNames Command.....	142
DemandPlan GetSupplyPlans Command.....	143
DemandPlan StrictUpdateDemandFromXml Command.....	144
DemandPlan StrictUpdateDemandFromXmlToCycle Command.....	144
DemandPlan UpdateDemand Command.....	145
DemandPlan UpdateDemandFromXml Command.....	146
DemandPlan UpdateDemandFromXmlToCycle Command.....	147
DemandPlan WriteDemandToXml Command.....	148
DemandPlan WriteDemandToXmlFromCycle Command.....	148
InventoryHistory AddInventoryHistory Command.....	149
InventoryHistory AddInventoryHistoryFromXml Command.....	149

InventoryHistory DeleteAllInventoryHistories Command.....	150
InventoryHistory DeleteInventoryHistory Command.....	150
InventoryHistory GetBeginningInventory Command.....	150
InventoryHistory WriteInventoryHistoryToXml Command.....	151
Items AddItem Command.....	151
Items AddItemChannelPrice Command.....	152
Items AddItemChannelPricesFromXml Command.....	152
Items AddItemHoldingCost Command.....	153
Items AddItemHoldingCostsFromXml Command.....	153
Items AddItemProductionCost Command.....	154
Items AddItemProductionCostsFromXml Command.....	154
Items AddItemsFromXml Command.....	155
Items CheckExists Command.....	156
Items DeleteAllItems Command.....	156
Items DeleteItem Command.....	156
Items GetItemCodes Command.....	157
Items GetItemProperties Command.....	157
Items GetItemPropertyNames Command.....	158
Items SetItemProperties Command.....	158
Items SetItemPropertyNames Command.....	158
Items StrictUpdateItemsFromXml Command.....	159
Items UpdateItemsFromXml Command.....	160
Items WriteItemChannelPricesToXml Command.....	160
Items WriteItemHoldingCostsToXml Command.....	161
Items WriteItemProductionCostsToXml Command.....	161
Items WriteItemsToXml Command.....	162
Locations AddLocation Command.....	162
Locations AddLocationsFromXml Command.....	162
Locations CheckExists Command.....	163
Locations DeleteAllLocations Command.....	163
Locations DeleteLocation Command.....	164
Locations GetLocationCodes Command.....	164
Locations GetLocationProperties Command.....	164
Locations GetLocationPropertyNames Command.....	165
Locations SetLocationProperties Command.....	165
Locations SetLocationPropertyNames Command.....	166
Locations StrictUpdateLocationsFromXml Command.....	167
Locations UpdateLocationsFromXml Command.....	167
Locations WriteLocationsToXml Command.....	168
Log ClearDbLog Command.....	168

Log ClearDbLogOnAndBeforeDate Command.....	169
Log ClearFileLog Command.....	169
Log GetDbLogEvents Command.....	170
Log GetFileLogEvents Command.....	170
Log GetLoggingStatus Command.....	171
Log LogEvent Command.....	171
Log MultiLogEvent Command.....	172
Log ResetLoggingSystem Command.....	172
Log StartDbLog Command.....	173
Log StartFileLog Command.....	173
Log StartLogging Command.....	173
Log StopAllLogging Command.....	174
Log StopLogging Command.....	174
MetricExceptionReports DeleteAllReports Command.....	175
MetricExceptionReports DeleteReport Command.....	175
MetricExceptionReports ExecuteReport Command.....	176
MetricExceptionReports ExportReports Command.....	176
MetricExceptionReports GetReportsInfo Command.....	177
MetricExceptionReports GetReportsStatus Command.....	177
MetricExceptionReports GetReportValues Command.....	178
MetricExceptionReports ImportReports Command.....	179
NetChangeReports DeleteAllReports Command.....	179
NetChangeReports DeleteReport Command.....	180
NetChangeReports ExecuteReport Command.....	180
NetChangeReports ExportReports Command.....	180
NetChangeReports GetReportsInfo Command.....	181
NetChangeReports GetReportsStatus Command.....	181
NetChangeReports GetReportValues Command.....	182
NetChangeReports ImportReports Command.....	183
Periods AddPeriod Command.....	183
Periods DeleteAllPeriods Command.....	184
Periods DeletePeriod Command.....	184
Periods ExportPeriods Command.....	185
Periods ImportPeriods Command.....	185
PlanMapping AddPlanMappingFromXml Command.....	186
PlanMapping WritePlanMappingToXml Command.....	186
ProductionHistory AddProductionHistory Command.....	187
ProductionHistory AddProductionHistoryFromXml Command.....	187
ProductionHistory DeleteAllProductionHistories Command.....	188
ProductionHistory DeleteProductionHistory Command.....	188

ProductionHistory WriteProductionHistoryToXml Command.....	189
ProductionPlan AddProduction Command.....	189
ProductionPlan AddProductionFromXml Command.....	190
ProductionPlan AddProductionFromXmlToCycle Command.....	190
ProductionPlan ContainsId Command.....	191
ProductionPlan DeleteAllProductions Command.....	192
ProductionPlan DeleteAllProductionsForCycle Command.....	192
ProductionPlan DeleteProduction Command.....	192
ProductionPlan GetProductionIds.....	193
ProductionPlan StrictUpdateProductionFromXml Command.....	193
ProductionPlan StrictUpdateProductionFromXmlToCycle Command.....	194
ProductionPlan UpdateProduction Command.....	195
ProductionPlan UpdateProductionFromXml Command.....	196
ProductionPlan UpdateProductionFromXmlToCycle Command.....	197
ProductionPlan WriteProductionToXml Command.....	197
ProductionPlan WriteProductionToXmlFromCycle Command.....	198
Resources AddPlannedCapacity Command.....	198
Resources AddPlannedCapacityFromXml Command.....	199
Resources AddRequiredCapacity Command.....	200
Resources AddRequiredCapacityFromXml Command.....	200
Resources AddRequiredCapacityFromXmlToCycle Command.....	201
Resources AddResource Command.....	202
Resources AddResourceMaximumCapacitiesFromXml Command.....	202
Resources AddResourceMaximumCapacity Command.....	203
Resources AddResourcesFromXml Command.....	203
Resources CheckExists Command.....	204
Resources DeletePlannedCapacity Command.....	205
Resources DeleteAllPlannedCapacityForCycle Command.....	205
Resources DeleteAllRequiredCapacities Command.....	205
Resources DeleteAllRequiredCapacitiesForCycle Command.....	206
Resources DeleteAllResources Command.....	206
Resources DeletePlannedCapacity Command.....	207
Resources DeleteRequiredCapacity Command.....	207
Resources DeleteResource Command.....	208
Resources GetPlannedCapacityValues Command.....	208
Resources GetResourceCodes Command.....	209
Resources GetResourceProperties Command.....	209
Resources GetResourcePropertyNames Command.....	209
Resources PlannedCapacityContainsId Command.....	210
Resources RequiredCapacityContainsId Command.....	210



Resources SetResourceProperties Command.....	211
Resources SetResourcePropertyNames Command.....	211
Resources StrictUpdatePlannedCapacityFromXml Command.....	212
Resources StrictUpdatePlannedCapacityFromXmlToCycle command.....	213
Resources StrictUpdateRequiredCapacityFromXml Command.....	213
Resources StrictUpdateRequiredCapacityFromXmlToCycle Command.....	214
Resources StrictUpdateResourcesFromXml Command.....	215
Resources UpdatePlannedCapacity Command.....	216
Resources UpdatePlannedCapacityFromXml Command.....	216
Resources UpdatePlannedCapacityFromXmlToCycle Command.....	217
Resources UpdateRequiredCapacity Command.....	218
Resources UpdateRequiredCapacityFromXml Command.....	219
Resources UpdateRequiredCapacityFromXmlToCycle Ccommand.....	220
Resources UpdateResourcesFromXml Command.....	221
Resources WritePlannedCapacityToXml Command.....	221
Resources WritePlannedCapacityToXmlFromCycle command.....	222
Resources WriteRequiredCapacityToXml Command.....	222
Resources WriteRequiredCapacityToXmlFromCycle Command.....	223
Resources WriteResourceMaximumCapacitiesToXml Command.....	224
Resources WriteResourcesToXml Command.....	224
SalesHistory AddSalesHistoryFromXml Command.....	224
SalesHistory AddSalesHistoryPrices Command.....	225
SalesHistory AddSalesHistoryUnits Command.....	226
SalesHistory DeleteAllSalesHistories Command.....	226
SalesHistory DeleteSalesHistoryPrices Command.....	227
SalesHistory DeleteSalesHistoryUnits Command.....	227
SalesHistory UpdateSalesHistoryFromXml Command.....	228
SalesHistory WriteSalesHistoryToXml Command.....	229
SupplyPlan AddSupplyPlan Command.....	229
SupplyPlan DeleteSupplyPlan Command.....	230
SupplyPlan GetSupplyPlanNames Command.....	230
UnitsOfMeasure AddItemUomFromXml Command.....	231
UnitsOfMeasure AddUom Command.....	232
UnitsOfMeasure Clear Command.....	232
UnitsOfMeasure DeleteUom Command.....	232
UnitsOfMeasure GetItemUom Command.....	233
UnitsOfMeasure GetUomList Command.....	234
UnitsOfMeasure SetItemUom Command.....	234
UnitsOfMeasure WriteItemUomToXml Command.....	234
Users AddUser Command.....	235

Users AddUsersFromXml Command.....	235
Users DeleteUser Command.....	236
Users IsAdmin Command.....	237
Users IsViewAdmin Command.....	237
Users SetPassword Command.....	238
Users StrictUpdateUsersFromXml Command.....	238
Users UpdateUsersFromXml Command.....	239
Users WriteUsersToXml Command.....	239
Views AddConsolidatedView Command.....	240
Views AddDemandPlanView Command.....	241
Views AddFinancialView Command.....	241
Views AddInventoryView Command.....	242
Views AddProductionPlanView Command.....	243
Views AddRoughCutCapacityView Command.....	243
Views DeleteAllViews Command.....	244
Views DeleteView Command.....	245
Views ExecuteAllViews Command.....	245
Views ExecuteView Command.....	245
Views ExportRelativeTargets Command.....	246
Views ExportTargets Command.....	246
Views ExportThresholds Command.....	247
Views ExportViews Command.....	247
Views GetCyclePlans Command.....	248
Views GetKeys Command.....	248
Views GetMetricValues Command.....	249
Views GetViewsStatus Command.....	250
Views ImportRelativeTargets Command.....	251
Views ImportTargets Command.....	251
Views ImportThresholds Command.....	252
Views ImportViews Command.....	253
Views InvalidateAllViews Command.....	253
Views SetPlansForViewCycle Command.....	253

## Chapter 10

<b>EnterpriseOne Sales and Operations Planning Glossary.....</b>	<b>257</b>
EnterpriseOne Sales and Operations Planning Glossary.....	257

<b>Index .....</b>	<b>261</b>
--------------------	------------

# EnterpriseOne Sales and Operations Planning Preface

This preface discusses:

- Related documentation.
- Typographical Conventions and Visual Cues.

---

**Note.** This Implementation Guide documents only page elements that require additional explanation. If a page element is not documented with the process or task in which it is used, then it either requires no additional explanation or is documented with the common elements for the section, chapter, or Implementation Guide.

---

## Related Documentation

This section discusses how to:

- Obtain documentation updates
- Order printed documentation

### Obtaining Documentation Updates

The EnterpriseOne Sales and Operations Planning Implementation Guide provides you with information about how to implement and use the EnterpriseOne Sales and Operations Planning system. Additional essential information describing deployment and supplemental third-party software options resides in the Supply Chain Planning Hardware and Software Requirements Guide. You should be familiar with the contents of this guide.

### Ordering Printed Documentation

You can order printed, bound volumes of the complete Oracle documentation that is delivered on the Implementation Guides CD-ROM. Oracle makes printed documentation available for each major release shortly after the software is shipped. Customers and partners can order the documentation in the following ways:

- Electronic mail: [appsdoc\\_us@oracle.com](mailto:appsdoc_us@oracle.com)
- FAX: 650-506-7200 Attn: Oracle EnterpriseOne Sales and Operations Planning Manager
- Postal Service:

Oracle EnterpriseOne Sales and Operations Planning Manager  
Oracle Corporation  
500 Oracle Parkway  
Redwood Shores, CA 94065  
USA

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

---

## Typographical Conventions and Visual Cues

This section discusses:

- Typographical conventions
- Visual cues

### Typographical Conventions

The following table contains the typographical conventions that are used in Implementation Guides:

Typographical Convention or Visual Cue	Description
" " (quotation marks)	Indicate chapter titles in cross-references and words that are used differently from their intended meanings.
{ } (curly braces)	Indicate a choice between two options in code syntax. Options are separated by a pipe (   ).
[ ] (square brackets)	Indicate optional items in code syntax.
Cross-references	Implementation Guides provide cross-references either following the heading "See Also" or on a separate line preceded by the word See. Cross-references lead to other documentation that is pertinent to the immediately preceding documentation.

### Visual Cues

Implementation Guides contain the following visual cues.

#### Notes

Notes indicate information that you should pay particular attention to as you work with the EnterpriseOne system.

---

**Note.** Example of a note.

---

A note that is preceded by Important! is crucial and includes information that concerns what you must do for the system to function properly.

---

**Note.** Example of an important note.

---

#### Warnings

Warnings indicate crucial configuration considerations. Pay close attention to warning messages.

---

**Note.** Example of a warning.

---

## Comments and Suggestions

Your comments are important to us. We encourage you to tell us what you like, or what you would like to see changed about Implementation Guides and other Oracle reference and training materials. Please send your suggestions to:

Oracle EnterpriseOne Sales and Operations Planning Documentation Manager

Oracle Corporation

500 Oracle Parkway

Redwood Shores, CA 94065

USA

Or email comments to: [appsdoc\\_us@oracle.com](mailto:appsdoc_us@oracle.com).

While we cannot guarantee to answer every email message, we will pay careful attention to your comments and suggestions.



# CHAPTER 1

## Getting Started with EnterpriseOne Sales and Operations Planning

This chapter provides an overview of Sales and Operations Planning and discusses:

- EnterpriseOne Sales and Operations Planning Business Processes.
- EnterpriseOne Sales and Operations Planning Integrations.
- EnterpriseOne Sales and Operations Planning Implementation.

---

### EnterpriseOne Sales and Operations Planning Overview

EnterpriseOne Sales and Operations Planning is a web-based, collaborative tool that helps businesses plan and facilitate their sales and operations planning process. It provides a bird's eye view of all relevant information, behind which are the precise, accurate details that can be tapped into on-demand. Multiple plans can be compared against each other and with the baseline using financial and/or operations metrics. Using Sales and Operations Planning, organizations can reach consensus on a single operating plan, which ensures that they will meet their long-term corporate goals.

Sales and Operations Planning focuses on the reconciliation and alignment of demand, supply, and financial plans; developing scenarios and enabling decision support; and managing the business process. Sales and Operations Planning assumes demand, supply, and financial plans have been developed elsewhere. Sales and Operations Planning leverages Demand Forecasting or Demand Consensus (DF/DC) for demand plans, Strategic Network Optimization (SNO) for supply plans, and Oracle Financial Planning for financial plans and budgets. However, since data is imported using standard XML files, any valid source of these plans may be used.

Sales and Operations Planning provides a simple mechanism to roll up the information provided by these products into an easy to analyze, single consolidated plan. With the help of Key Performance Indicators (KPIs), the sales and operations planning team can suggest different scenarios to manage out-of-tolerance conditions in metrics such as demand fill rate, inventory level, and capacity utilization. In addition, the sales and operations planning process can help an organization:

- Build teamwork and functional alignment.
- Agree on cross-functional strategies.
- Keep demand and supply in balance.
- Integrate operational and financial planning.
- Communicate one plan monthly to every manager in the business.

---

## EnterpriseOne Sales and Operations Planning Automation Shell

In addition to using the EnterpriseOne Sales and Operations Planning graphical user interface (GUI), you can perform functions using commands in the Sales and Operation Planning Automation Shell (SASH). The Sales and Operations Planning Automation Shell includes commands and subcommands for working with the following Sales and Operations Planning components:

- Calendars
- Custom Metrics
- Data Import
- Error Logs
- Reports
- Users
- Views

This command reference guide provides detailed information about each SASH command and subcommand, including syntax information and examples.

### See Also

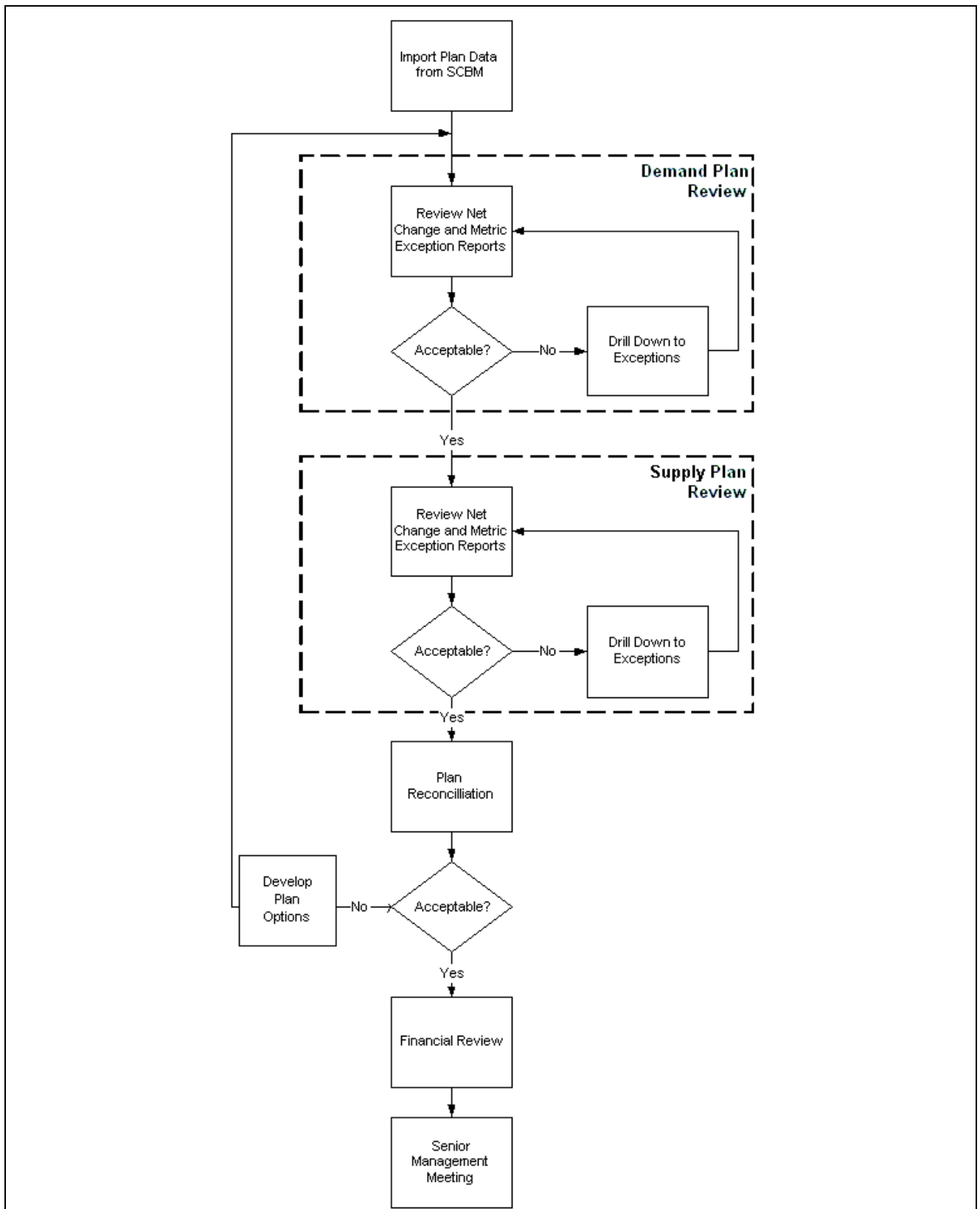
Using the EnterpriseOne Sales and Operations Planning Automation Shell

---

## EnterpriseOne Sales and Operations Planning Business Processes

Sales and Operations Planning supports the Planning and Performance Management business process. The following process flow illustrates this business process:





The Sales and Operations Planning business process

## Sales and Operations Planning Integrations

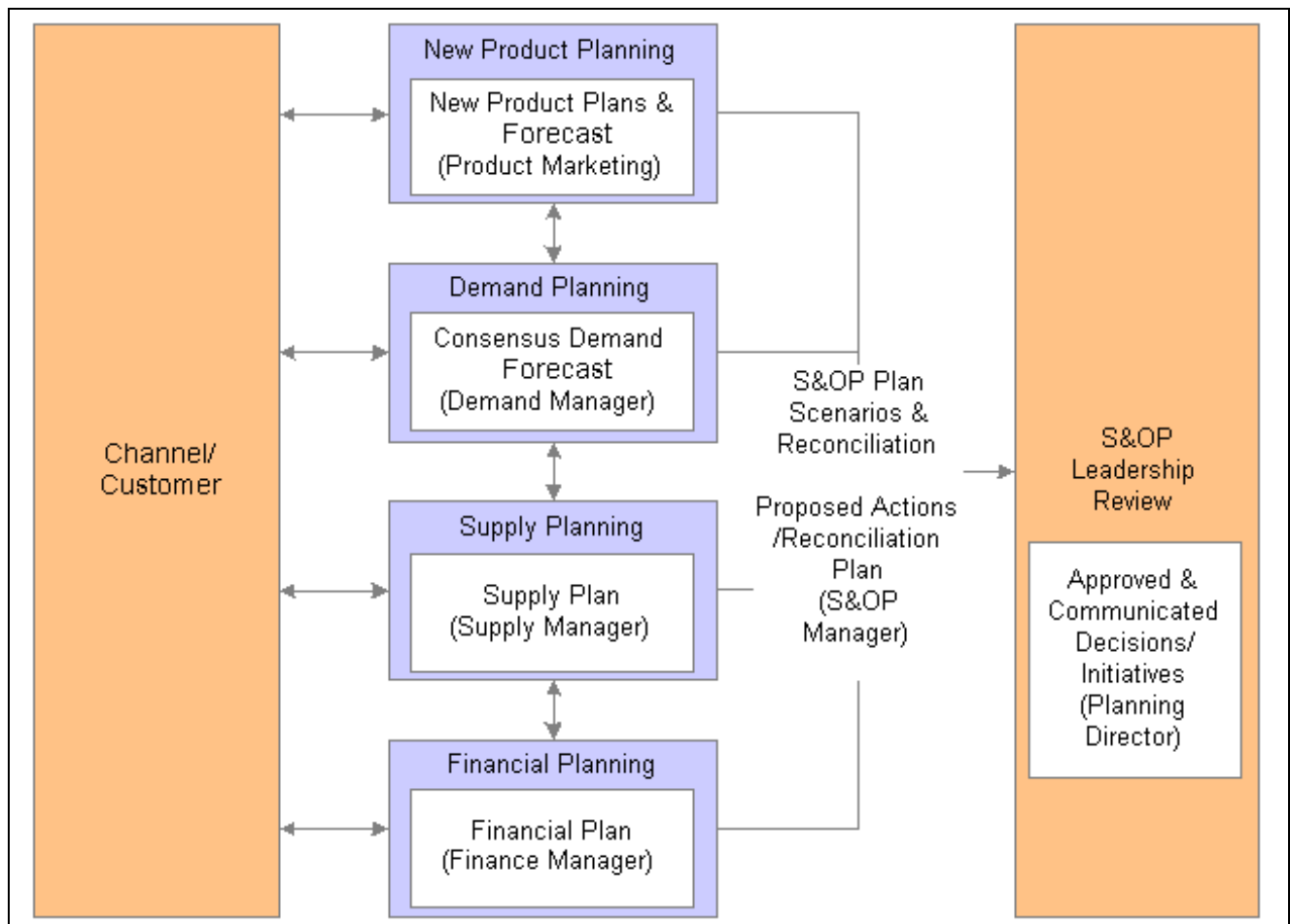
Sales and Operations Planning integrates with EnterpriseOne Supply Chain Business Modeler.

Integration considerations are discussed in the implementation chapters in this Integration Guide. Supplemental information about third-party application integrations is located on the Oracle Customer Connection web site.

## EnterpriseOne Supply Chain Business Modeler

EnterpriseOne Sales and Operations Planning integrates with EnterpriseOne Supply Chain Business Modeler, enabling seamless exchange of data.

The following graphic illustrates how data is exchanged with EnterpriseOne Supply Chain Business Modeler:



Data exchange between Sales and Operations Planning and Supply Chain Business Modeler

# EnterpriseOne Sales and Operations Planning Implementation

The EnterpriseOne Sales and Operations Planning implementation can be divided into the following phases:

- Configuring the application environment.
- Configuring the Plan Calendar.
- Importing plan data.
- Creating views.
- Defining reports.

## Configuring the Application Environment

The steps discussed in this section provide the information necessary to configure Sales and Operations Planning for optimal performance:

Step	Reference
Set up the web server.	“Installing Sales and Operations Planning”, <i>EnterpriseOne Sales and Operations Planning 8.12 Installation for Windows</i> .  “Installing Sales and Operations Planning”, <i>EnterpriseOne Sales and Operations Planning 8.12 Installation for UNIX</i> .
Add users.	“Administering Users”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> .

## Configuring the Plan Calendar

The steps discussed in this section provide the information necessary to configure the Sales and Operations Planning Plan Calendar:

Step	Reference
Add planning periods to the model.	“Managing Planning Periods”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> .
Add planning cycles to the model.	“Managing Planning Cycles”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> .

## Importing Plan Data

The steps discussed in this section provide the information necessary to import the latest plan data to Sales and Operations Planning:

Step	Reference
Import the latest plan data from SCBM.	“Importing Data Using the Sales and Operations Planning Connector”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> .

## Creating Views

The steps discussed in this section provide the information necessary to create Sales and Operations Planning data views and define targets and thresholds:

Step	Reference
Create the required data views.	“Creating New Views”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> .
Set your planning targets and thresholds.	“Managing Targets for Relative Metrics”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> . “Managing Targets for Absolute Metrics”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> . “Manage Thresholds for Absolute Metrics”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> .

## Defining Reports

The steps discussed in this section provide the information necessary to create Metric Exception and Net Change reports:

Step	Reference
Create Net Change reports.	“Managing Net Change Reports”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> .
Create Metric Exception reports.	“Managing Metric Exception Reports”, <i>Sales and Operations Planning 8.12 Implementation Guide</i> .

## CHAPTER 2

# Understanding Sales and Operations Planning

This chapter discusses:

- Sales and operations planning process.
- The Sales and Operations Planning workflow.
- Sales and Operations Planning user interface.

---

## Sales and Operations Planning Process

Sales and operations planning is a formal management process that creates a unified business plan based on a consensus between an organization's sales, manufacturing and financial functions. Collaboration among an organization's functional areas results in an integrated set of plans that all stakeholders understand and are committed to support.

Sales and operations planning is typically led by senior management. Sales and operations planning is most often done monthly, and is an iterative process. Results from one planning cycle are compared with the next to give senior management trends within their business. They evaluate time-phased projections for supply and demand, and ensure that the tactical plans in all business functions and geographies are aligned and in support of the company's strategy. In addition to senior management, each planning stage may involve many other roles, including the demand manager, sales and operations process leader, master planner, and so on.

With traditional "silo-based" companies, sales, manufacturing and financial functions often compete against each other. Without a single consolidated plan, small problems can quickly grow into much larger issues. Using Sales and Operations Planning reports and KPIs, these problems can be quickly identified. Plan over plan net change reports can be run against key metrics to identify the largest changes automatically. With the high-level differences identified, further detail and understanding of the underlying causes are analyzed through various views and reports. With the exceptions and issues identified, different supply scenarios can be developed.

The primary objectives of the sales and operations planning process are to:

- Ensure that plans are aligned with the company's strategic objectives.
- Ensure that the plans are realistic and meet customer requirements.
- Effectively manage change.
- Manage inventory, lead time, backlog, capacity, capital investment and demand.

A sales and operations planning process should also help to answer the following basic questions:

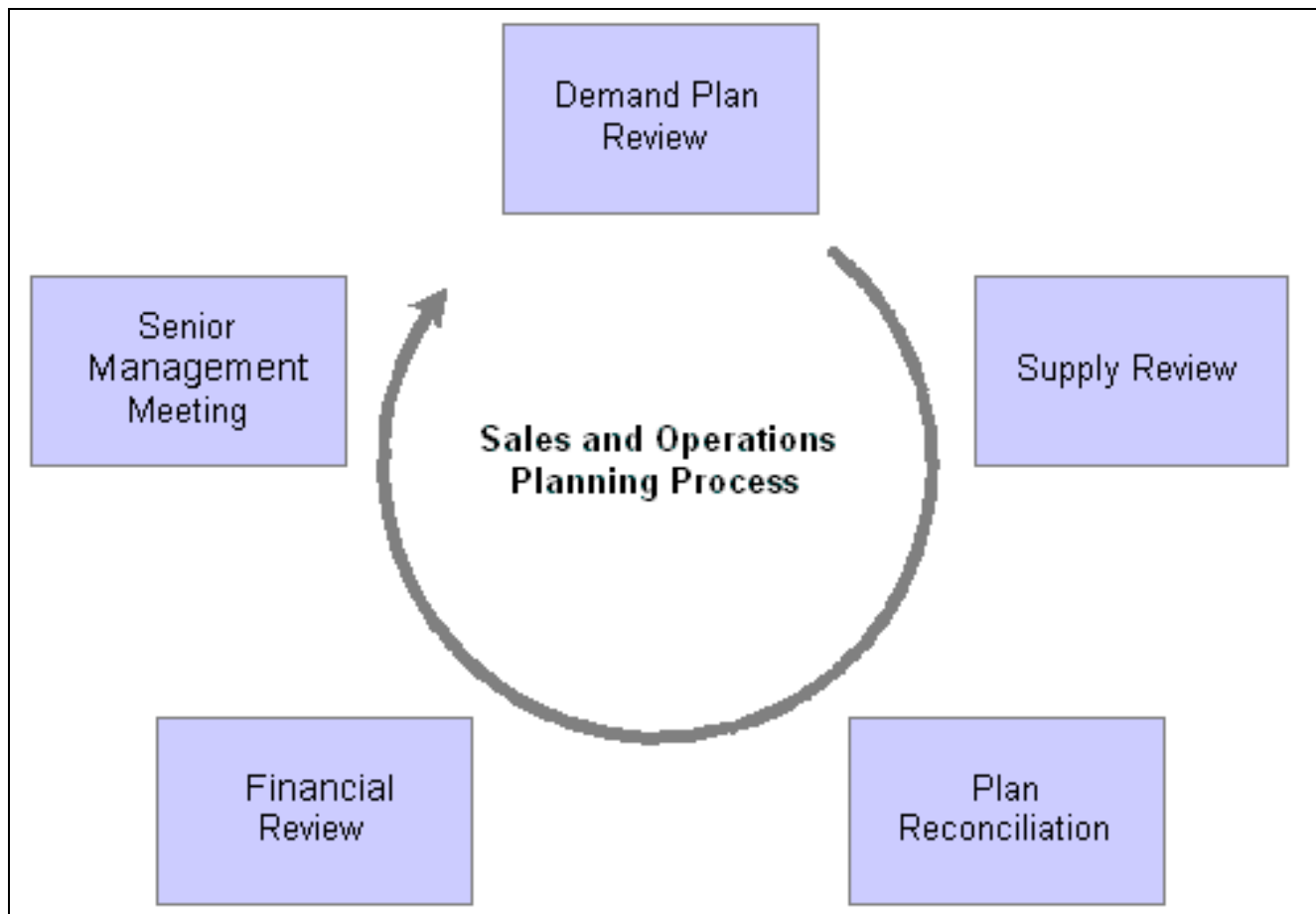
- How does projected demand compare to projected supply?
- What are the projected resource requirements to meet both service and cost targets?
- What actions are required to ensure the appropriate levels of resources are available when needed?

While the sales and operations planning process can differ greatly between organizations, there are specific elements that are common to virtually all processes. A typical sales and operations planning process can be described in three major steps:

1. Updating demand, supply, and financial plans with the latest plan data.
2. Analysis and reconciliation of the demand, supply, and financial plans.
3. Senior management meeting to review the consolidated plans.

However, in practice, different companies will have varying practices and time frames for the process, ranging from a high-level, strategic process to a more short-term, operational focus. Sales and Operations Planning's flexible, non-linear architecture allows you to map your existing sales and operations planning process to Sales and Operations Planning.

The following diagram illustrates the typical monthly sales and operations planning business process:



The sales and operations planning process

Sales and Operations Planning features a robust XML-based import mechanism which allows you to import data from other systems. Since Sales and Operations Planning does not own any of the data it uses, it's designed to work with - and not replace - your existing systems of record. For example, one of the requirements for the sales and operations planning process is a master production schedule. This schedule can come from a variety of sources, such as data from ERP, or from a capacity-constrained supply planning tool like EnterpriseOne Strategic Network Optimization.

---

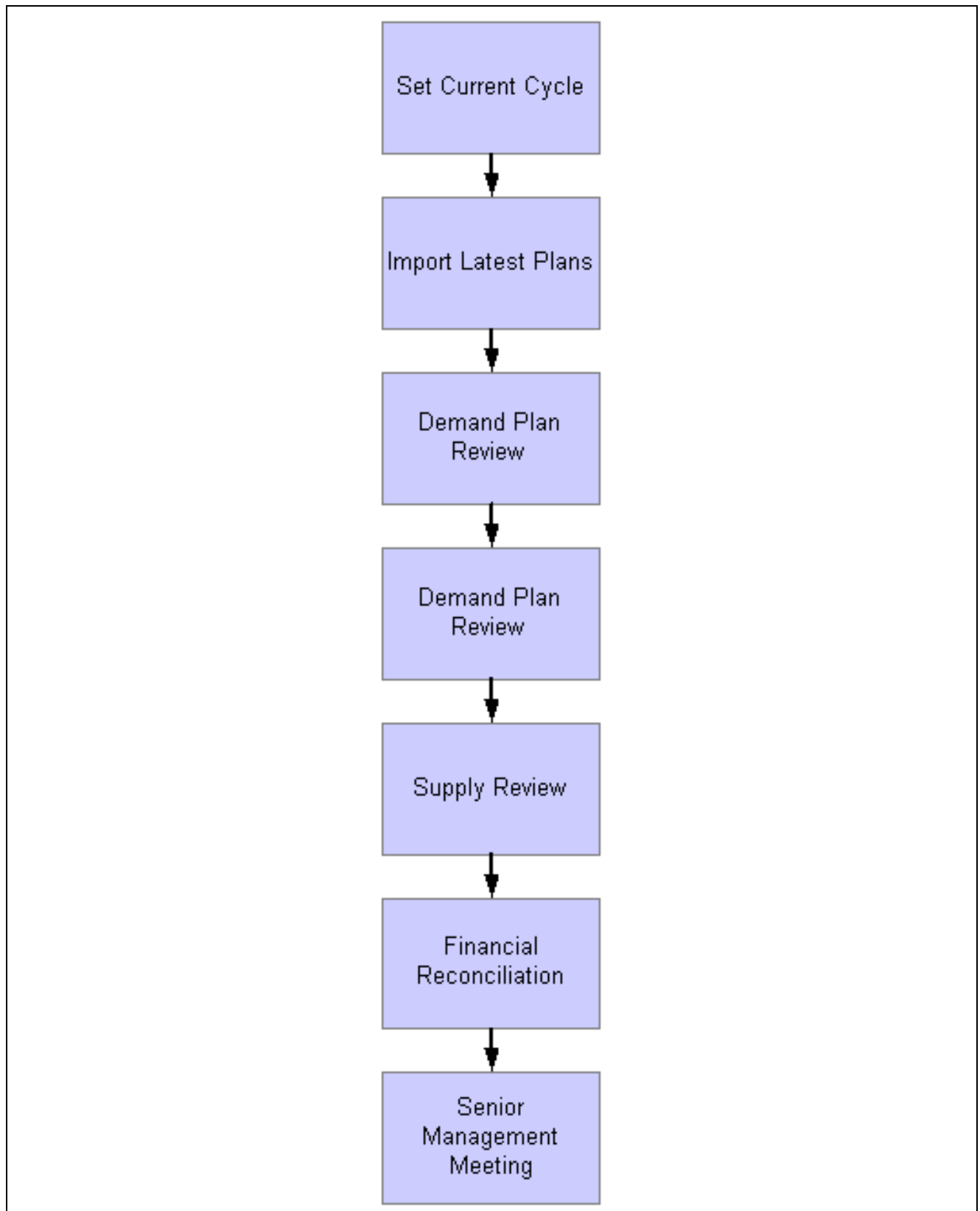
## The Sales and Operations Planning Workflow

This section discusses:

- The Sales and Operations Planning workflow.
- Current planning cycle.
- Plan import.
- Demand Plan review.
- Supply review.
- Financial reconciliation.
- Senior management meeting.

### Sales and Operations Planning Workflow Overview

The following business process diagram illustrates a typical monthly Sales and Operations Planning workflow:



Sales and operations planning workflow



## Current Planning Cycle

Each iteration of the monthly sales and operations planning process is called a planning cycle and is based on your calendar periods. You define these planning cycles to match your financial calendar. Organizing plans into planning cycles enables month-by-month plan comparisons and allows you to check if targets have been met.

The first step in sales and operations planning is to set the current planning cycle. When you set the new current cycle, Sales and Operations Planning automatically updates your views to display plan data from the new cycle.

### See Also

Managing Planning Periods

Managing Planning Cycles

## Plan Import

After setting the current planning cycle, the next step in the sales and operations planning process is to import your latest plan data. Plans are imported using the Sales and Operations Planning Connector from EnterpriseOne Supply Chain Business Modeler and are imported to the current plan. Alternately, you can use SASH to import plan data from other external sources.

### See Also

Importing Data Using the Sales and Operations Planning Connector

## Assign Plans to Views

Once you plans are imported into Sales and Operations Planning, you should assign them to data views. Data views access plans and display the data in a meaningful way, and can be manipulated to display what is most relevant to your business. For example, Sales and Operations Planning can aggregate data to the planning period, geographical location, or channel in which you are most interested. Each data view is designed for a different phase of the sales and operations planning process. For example, the Demand Plan view uses pre-defined metrics and graphical display options to help reach consensus on forecast demand.

### See Also

Assigning Plans to Views

Understanding Views

Creating New Views

## Demand Plan Review

Once Sales and Operations Planning contains updated plan data, the next step is to review the Demand Plan.

A Demand Plan is a company's agreed-upon forecast of future demand for its products. Demand Plans are based on historical information on sales, sales campaigns, or other initiatives that can affect demand. A demand plan enables you to create meaningful schedules, production plans, and financial plans to meet that demand. A Demand Plan is created during the demand review process. This review may only involve the marketing and sales departments, or may incorporate other stakeholders like customers. The demand review meeting typically involves reviewing:

- The baseline demand forecast.
- Changes in marketing plans.

- Changes in sales plans.
- New product introduction plans.

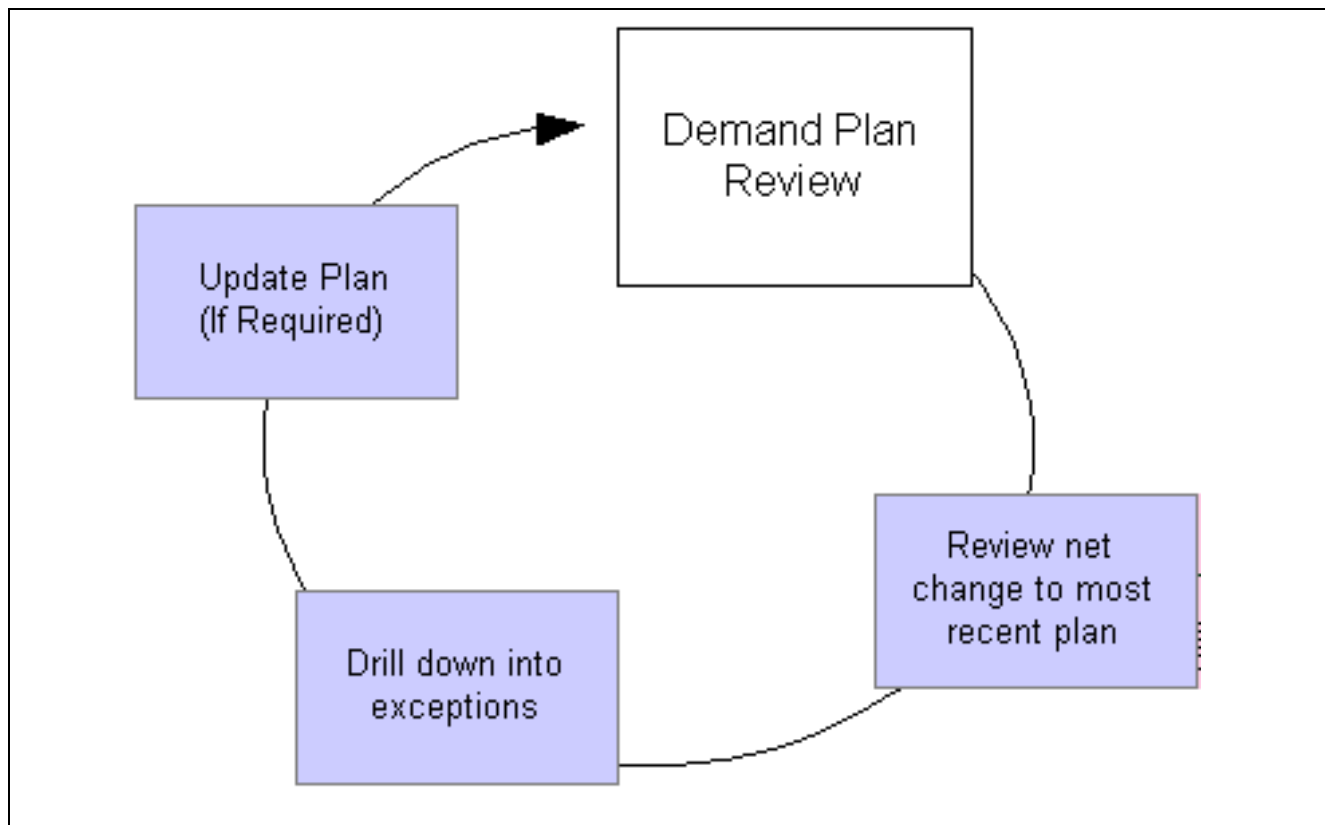
---

**Note.** If this is the first planning cycle, create a Demand Plan view that includes the metrics, dimensions and properties in which you're interested.

---

To help ensure that your organization is investing at the right level and producing optimal plans, you can create and work with multiple Demand Plans for comparison and “what if” scenarios. For example, you may choose to create optimistic and pessimistic plan options to determine how to plan future demand. When you have completed the plan comparison, use Sales and Operations Planning to approve the plan of your choice and then use the approved plan during your supply review and financial reconciliation.

The following diagram illustrates the Demand Plan review process:



Demand Plan review

Use the Demand Plan's Dashboard view format for a high-level view of your plan. This view format helps identify net changes to the most recent plan and your defined targets to understand any change in strengths, weaknesses, opportunities and threats. Compare the new plans to the current or previous plans using the Consolidated Plan view to see exactly what is changing in the demand and supply picture over time. This may include looking at year-to-date customer demand and current orders compared to forecasted revenue.

From the Dashboard you can drill-down to the Demand and Consolidated Plan views to identify the cause of any exceptions and potential constraints. Based on these exceptions, you may need to assign certain external tasks, such as generating a new demand plan based on updated assumptions and priorities. To help you organize and manage the sales and operations planning process, these external tasks and other important milestones can be captured using the Plan Calendar.

## See Also

Demand Plan view

Consolidated View

Managing the Sales and Operations Planning Process using the Plan Calendar

Dashboard Format

## Supply Review

Supply reviews determine how an organization plans to meet demand, based on its existing inventory and production capacity. The supply review meeting typically involves reviewing:

- Company's current situation, including inventories, opportunities and problems, and available capacity.
- Changes in production plans.
- Changes in inventory plans.
- Changes in capacity plans.
- Any supplier issues.
- New process and equipment.

The three pieces of the supply review that are displayed in Sales and Operations Planning are the Production Plan, Inventory Plan, and Rough Cut Capacity Plan.

---

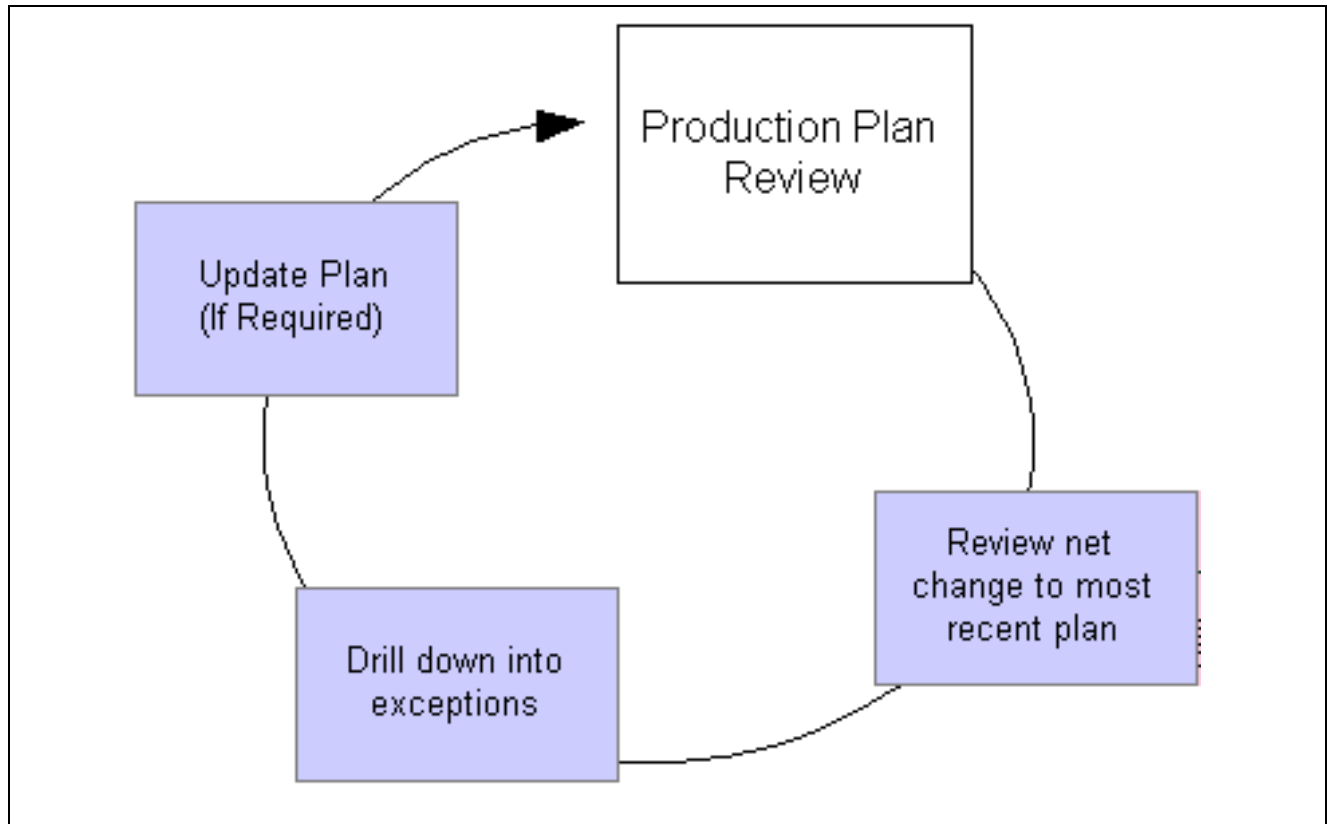
**Note.** If this is the first planning cycle, create Production Plan, Inventory Plan, and Rough Cut Capacity Plan views that include the metrics, dimensions and properties in which you are interested.

---

Use the Production Plan's Dashboard view format for a high-level view of your plan. This view format helps identify net changes to the most recent plan and your defined targets to understand any change in strengths, weaknesses, opportunities and threats. Once these exceptions have been identified, drill-down to more granular data views to understand their impact on key financial and operational measures.

From the Dashboard you can drill-down to the Inventory and Production Plan views to determine the plan feasibility. Based on the production plan review results, you may decide to revise the Demand or Rough Cut Capacity Plans to better reflect your current production capabilities. Alternately, the production plan may be updated to ensure that key demand is met.

The following diagram illustrates the Production Plan review process:



Production Plan review process

To help ensure that your organization is investing at the right level and producing optimal plans, you can create and work with multiple Supply Plans for comparison and “what if” scenarios. For example, you may choose to create optimistic and pessimistic plan options to determine how to plan future production and inventory. When you have completed the plan comparison, use Sales and Operations Planning to approve the plan of your choice and then use the approved plan during your financial reconciliation meeting.

### See Also

Production Plan View

Inventory Plan View

Rough Cut Capacity Plan View

Dashboard Format

## Financial Reconciliation

The next step in the Sales and Operations Planning workflow is to evaluate the Demand and Production Plans by comparing them against financial targets. The financial plan is the benchmark used by the Sales and Operations Planning process. Use the Financial Dashboard to compare your revenue, margin and other financial metrics against your targets. This high level view gives you an indication of how well the Demand and Production Plans meet your financial objectives.

Review the exception list to see if there are any out of tolerance metrics related to the financial performance that need to be reviewed and reconciled. Look for any changes to revenues, costs and budget from the previous cycle to ensure that your company’s performance is on target.

Depending on the outcome of the financial review and reconciliation process, you may need to change the Demand and Production Plans and continue iterating through the reconciliation process. Use Sales and Operations Planning views to identify gaps between supply and demand, and determine the cause of these gaps by reviewing material availability and capacity utilization.

### **See Also**

Financial View

## **Senior Management Meeting**

The senior management meeting is the final step in the sales and operations planning process. This meeting is an opportunity for all stakeholders to meet and:

- Review financial performance.
- Review identified issues and vulnerabilities.
- Review recommended solutions.
- Agree on final operations plan.
- Review strategies and assign actions required to bring operations in line with strategic direction.

Use the Plan Calendar to help you organize and manage the senior management meeting, assign action items and publish an agenda.

### **See Also**

Managing the Sales and Operations Planning Process using the Plan Calendar

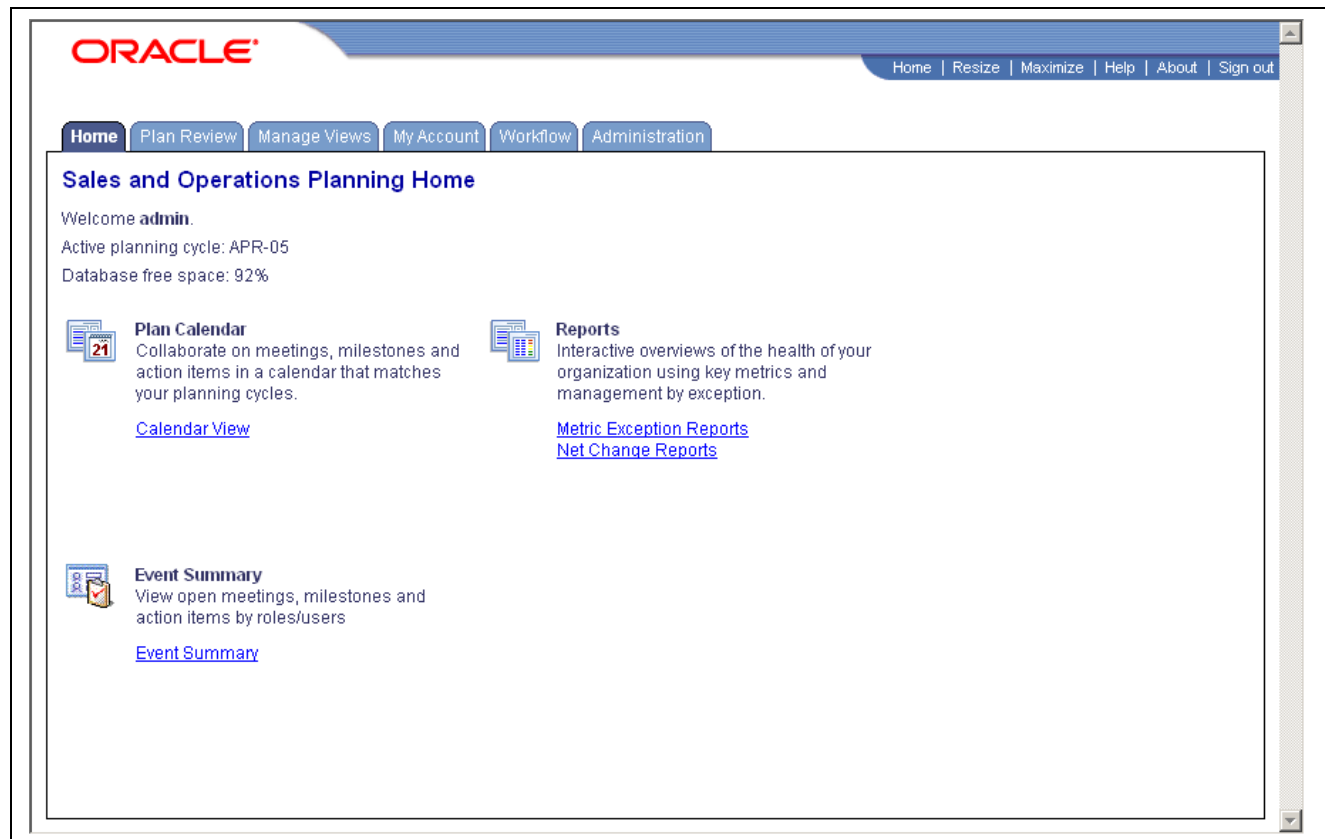
---

## **Sales and Operations Planning User Interface**

Sales and Operations Planning uses an intuitive, web-based user interface. Your particular Sales and Operations Planning desktop configuration depends on your account type and roles to which you are assigned. Normal users and view owners have access to Sales and Operations Planning functions found on the Home, Plan Review, Manage Views, My Account and Workflow pages. Administrators have access to these functions in addition to the administration tasks accessed on the Administration page.

Links to views that do not exist or for which users do not have permissions appear grayed.

This screen shows the components of the Sales and Operations Planning user interface:



Sales and Operations Planning User Interface

This table describes the various components of Sales and Operations Planning views:

Component	Description
Home	Navigates to the Sales and Operations Planning Home Page.
Resize	Repositions the application to fit your browser's current resolution and window size.
Maximize	Maximizes the Sales and Operations Planning view by removing the main menu tabs.
Help	Provides documentation about features and tasks.
Sign out	Closes the active page and returns to the sign in screen.
About	Displays the Sales and Operations Planning About screen, which shows the version and patch history.
Sales and Operations Planning main menu tabs	Provides access to all of the Sales and Operations Planning functions.

## CHAPTER 3

# Understanding Views

This chapter discusses:

- Views overview.
- Plan data views.
- Combined data views.
- View formats.
- Units of measure.
- User defined-metrics.
- Dimension properties.
- Targets and thresholds.

---

## Views Overview

This section provides an overview of Sales and Operations Planning views, and discusses:

- View Manager.
- Display types and view formats.
- Data views and metrics.
- Actual versus plan data.

Data views access plans and display the data in a meaningful way, and can be manipulated to display what you are most interested in seeing. Sales and Operations Planning allows you to create different plan types, and these plan types hold a particular kind of planning data. For example, Sales and Operations Planning can aggregate data to the planning period, geographical location, or channel in which you are most interested. However, a Supply Plan is a combination of inventory, production, and capacity and therefore cannot be displayed in a single data view. The view that is best able to represent a supply plan is the Consolidated view.

Sales and Operations Planning provides various data view types that highlight different aspects of your organization's performance. Data views allow the placement of time, product, location and channel dimensions on any axis of the graph. In addition, you can select how many series to appear in the graph by changing the row/column paging parameters. This paging feature works in both tabular and graphical displays.

Each data view displays different planning data and is used to identify different problems. For example, the Demand Plan view displays demand by product or product group, geographical region, sales channel, or any combination of these parameters. You can define multiple views of each type to help capture a particular set of data. By running Exception and Net Change reports on your views, you can focus your review and exceptions and then drill down to see your planning data in more detail and to identify potential problems.

Sales and Operations Planning views are grouped into two categories: plan data views and combined data views. Plan data views display data from one specific aspect of the sales and operations planning process. You can customize both axes of plan data views to display a combination of product, location, channel and resource data. The Demand, Production, and Inventory Plan are plan data views.

Combined data views take data from the different functional areas to get a complete picture of your organization. Combined data views are most useful during the reconciliation part of the sales and operations planning process, because they combine metrics from multiple data views. With combined data views the horizontal axis is fixed. The Financial and Consolidated views are combined data views.

## View Permissions

Since Sales and Operations Planning uses detailed financial data and forecasts, you may want to control who has access to each data view. For example, while a sales manager should have access to the Demand Plan views, you might want to restrict his or her access to the Financial and Consolidated views. You can allow or deny access to each data view individually, or by defining user roles. User roles allow administrators and view owners to maintain groups of users based on job role. Any number of user roles can be defined to logically categorize the users of the system.

View permissions are managed from the View Manager. Users can have access to a data view either through a user role to which they belong, or by explicitly assigning view permissions to the user. The following example shows a sample View Manager with the Item Location Demand view assigned to the Executive and Sales Manager roles:



The screenshot shows the Oracle View Manager interface. At the top, there is a navigation bar with the Oracle logo and links for Home, Resize, Maximize, Help, About, and Sign out. Below this is a secondary navigation bar with links for Home, Plan Review, Manage Views, My Account, Workflow, and Administration. The main content area is titled 'Manage Views > View Manager' and contains a table of views.

View Name	Type	Owner	Status
<a href="#">Branch Inventory by container type</a>	Inventory Plan	admin	Ready
<a href="#">Detailed Consolidated View</a>	Consolidated View	admin	Ready
<a href="#">Detailed Demand Review</a>	Demand Plan	admin	Ready
<a href="#">Detailed Inventory Plan</a>	Inventory Plan	admin	Ready
<a href="#">Detailed Production Plan</a>	Production Plan	admin	Ready
<b>Description:</b> <b>Dimensions:</b> Item Code, Location.City <b>System Metrics:</b> Unit Production, Production Cost <b>Custom Metrics:</b> <b>Units of Measure:</b> Each, Case, Litres <b>Roles:</b> Operations <b>Users:</b> admin			
<a href="#">National Brand Revenue Forecast</a>	Financial View	admin	Ready
<a href="#">Projected Financials - National Brand &amp; Channel</a>	Financial View	admin	Ready
<a href="#">Rough Cut Capacity</a>	Rough Cut Capacity Plan	admin	Ready
<a href="#">Test Financial Plan</a>	Financial View	admin	Ready

At the bottom of the table, there are buttons for 'Add', 'Plans', 'Custom Metrics', 'Permissions', 'Execute', and 'Delete'.

Example View Manager showing user roles



## See Also

Understanding Configuring Sales and Operations Planning

Administering Users

## View Manager

The View Manager displays a list of all defined views, the corresponding type, and the view owner. By expanding the folder beside the view name, you can display the description, configuration options and dimensions associated with each view. Use the View Manager to add, remove, and execute views, and to assign view permissions.

The following example illustrates the Sales and Operations Planning View Manager:



The View Manager

Clicking links navigates between data views to help you investigate plan details and fully understand potential problem areas. For example, the Demand Plan view may show that a particular brand has experienced a sharp decrease in demand month over month. To help understand why this is occurring, you can drill down from this brand-level to a more granular Demand Plan views to see if the decrease is consistent across all product families or is being caused by a sharp decrease in one particular family.

## Display Types and View Formats

To help you understand and interpret your planning data in the best way possible, Sales and Operations Planning provides many different display types and view formats.

A display type determines how you choose to represent planning data within a particular data view. Each view can be display as either a table or graph. The default display type for each data view is the table, and it displays actual and planned values for each metric, along with the ability to aggregate these values to the level in which you are most interested. The graph display type provide a flexible and intuitive visualization of plan data and performance indicators. Each Sales and Operations Planning view uses a different graph that best represents the data that you want to see. For example, the Demand Plan uses a bar graph to illustrate demand for each item by location or channel.

A view format determines what data you want to display. The default view format displays data as it is stored in the model without any comparison. This view format is useful for in-depth analysis of your planning data. The Comparison view formats compares planning data between two cycles, or between different plans. For example, you can compare the same plan between two planning cycles, or two different plan versions for the same cycle.

## Data Views and Metrics

The following table lists which metrics are available in each Sales and Operations Planning view. The Unit Based column indicates if a particular metric is based on the specific unit of measure and change based on which unit in which they are viewed. Some views use metrics that do not change with the unit of measure, for example Demand Fill Percentage.

The Metric Target Type column indicates the type of targets and thresholds that can be defined for this view.

**Note.** The metric calculations in the Metric Description column are only valid for plan data and not historical actuals. The metrics for actuals use imported values and not calculated.

Data View	Available Metrics	Metric Description	Unit Based?	Metric Target Type
Consolidated View	Unit Demand	The historical and forecast demand for the specified time period.	Yes	None
	Unit Production	The historical and forecast production for the specified time period.	Yes	None
	Inventory	The historical and forecast inventory levels for the specified time period.	Yes	None
	Shortage	The amount of demand in a specified time period that cannot be met by inventory levels.	Yes	None
	Demand Fill Percentage	The percentage of total demand that is satisfied in the actual period of demand.	No	Relative
Demand Plan	Unit Demand	The historical and forecast demand for the specified time period.	Yes	Absolute
	Revenue	The demand for the specified time period times price.	No	Absolute

Data View	Available Metrics	Metric Description	Unit Based?	Metric Target Type
Financial View	Unit Demand	The historical and forecast demand for the specified time period.	Yes	Absolute
	Average Selling Price	Revenue divided by demand for the specified time period.	Yes	Relative
	Revenue	The demand for the specified time period times price.	No	Absolute
	Cost of Goods Sold	The demand for the specified time period times production cost.	No	Relative
	Cost of Goods Sold per unit	The cost of goods sold divided by demand for the specified time period.	Yes	Absolute
	Margin	Revenue minus cost of goods sold.	No	Relative
	Margin Percentage	Margin divided by revenue minus 100%.	No	Relative
Inventory Plan	Inventory	Inventory levels at the end of the specified period.	Yes	Effective Dated
	Shortage	Inventory shortage (back orders) at the end of the specified period.	Yes	Effective Dated
	Holding Cost	Inventory level times holding cost.	No	Absolute
Production Plan	Unit Production	The historical and forecast production for the specified time period.	No	Absolute
	Production Cost	Unit production times production cost.	No	Absolute
Rough Cut Capacity Plan	Planned Capacity	The amount of planned capacity.	No	None
	Required Capacity	The demand for the specified time period.	No	None
	Deviation	Planned capacity minus required capacity.	No	None
	Utilization Percentage	Planned capacity divided by required capacity.	No	Relative

## See Also

Targets and Thresholds

Creating New Views

## Actual Versus Plan Data

Sales and Operations Planning data views display both plan and historical data, which can help you track your overall plan accuracy. For example, by comparing the forecast for a particular planning cycle to actual sales, you can see how well your organization estimated supply and demand.

Data views highlight historical data by shading the table cells gray. A horizontal bar indicates where the split between actuals and plan data occurs. However, the values associated with the planning period immediately after the horizon may represent both actual and forecast data.

**ORACLE** Home | Resize | Maximize | Help | About | Sign out

Home | **Plan Review** | Manage Views | My Account | Workflow | Administration

Plan Review > Demand Plan

View: Detailed Demand Review Format: Default

Demand Plan: Apr Baseline

**Detailed Demand Review** Rows: 1-3 of 3 Columns: 13-53 of 53 Display as: [Table Icon]

Period Group: Weekly Metric: Unit Demand Unit: Each Row: Channel Code Column: Time.Weekly [Update]

	2005-03-21	2005-03-28	2005-04-04	2005-04-11	2005-04-18	2005-04-25	2005-05-02	2005-05-09	2005-05-16	2005-05-23	2005-05-30
1001	7500.00	7500.00	7000.00	7000.00	7000.00	7000.00	8000.00	8000.00	8300.00	8500.00	9000.00
1002	1500.00	1500.00	5800.00	5800.00	5800.00	5800.00	6300.00	7000.00	7500.00	7500.00	8000.00
1003	620.00	620.00	4900.00	4900.00	4900.00	4900.00	5500.00	5500.00	5500.00	5500.00	5900.00

Item Code: 99001 Location.City: Atlanta

Demand plan showing actual and forecast planning data

In the Comparison and Waterfall views, there are special considerations when displaying historical planning data. Plan data in old cycles is not updated to represent actuals, since you would lose the ability to see how your plans change over time. By preserving plan data in the Comparison and Waterfall views, you can establish reasonable forecast lead times and understand how your plans change over time.

## Plan Data Views

This section discusses:

- The Demand Plan view.
- The Production Plan view.
- The Inventory Plan view.
- The Rough Cut Capacity Plan view.

## Demand Plan View

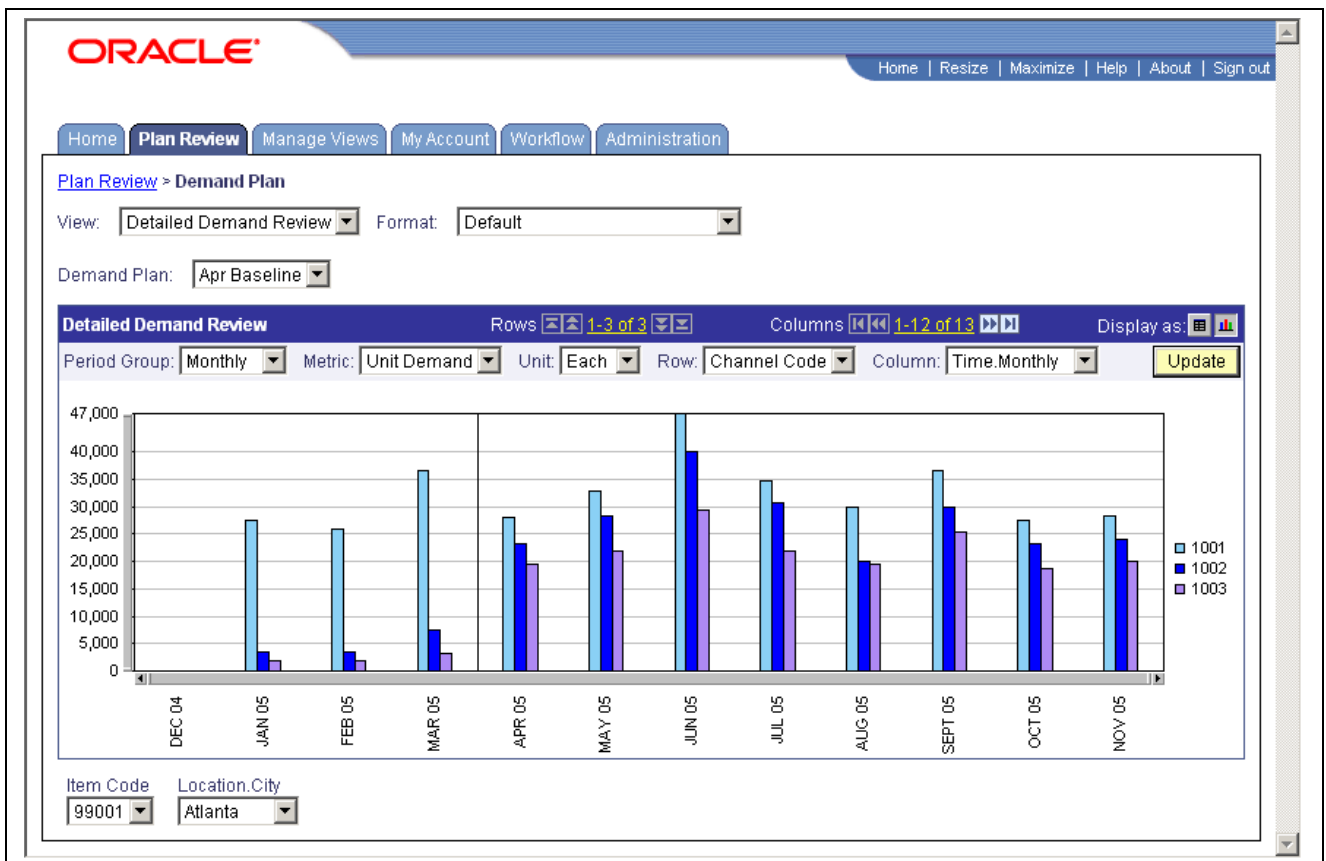
Use the Demand Plan view to display historical and forecast demand data. The Demand Plan view displays projected unit demand and revenue for the current planning cycle grouped by customer, item and location dimensions. This view can display plan data by week, month, quarter, or year.

Large and complex environments are represented in the Production Plan using simplified, aggregated views. However, even at simplified levels it can be difficult to interpret deviations and trends in tabular data. For this reason, the Demand Plan view supports graphical displays to provide a flexible and intuitive visualization of plan data and performance indicators. Use the Display As buttons to toggle between the table and graph display type.

The available Demand Plan metrics are:

- Revenue.
- Unit Demand.

The following example shows a sample Demand Plan view.



Example Demand Plan view

## Production Plan View

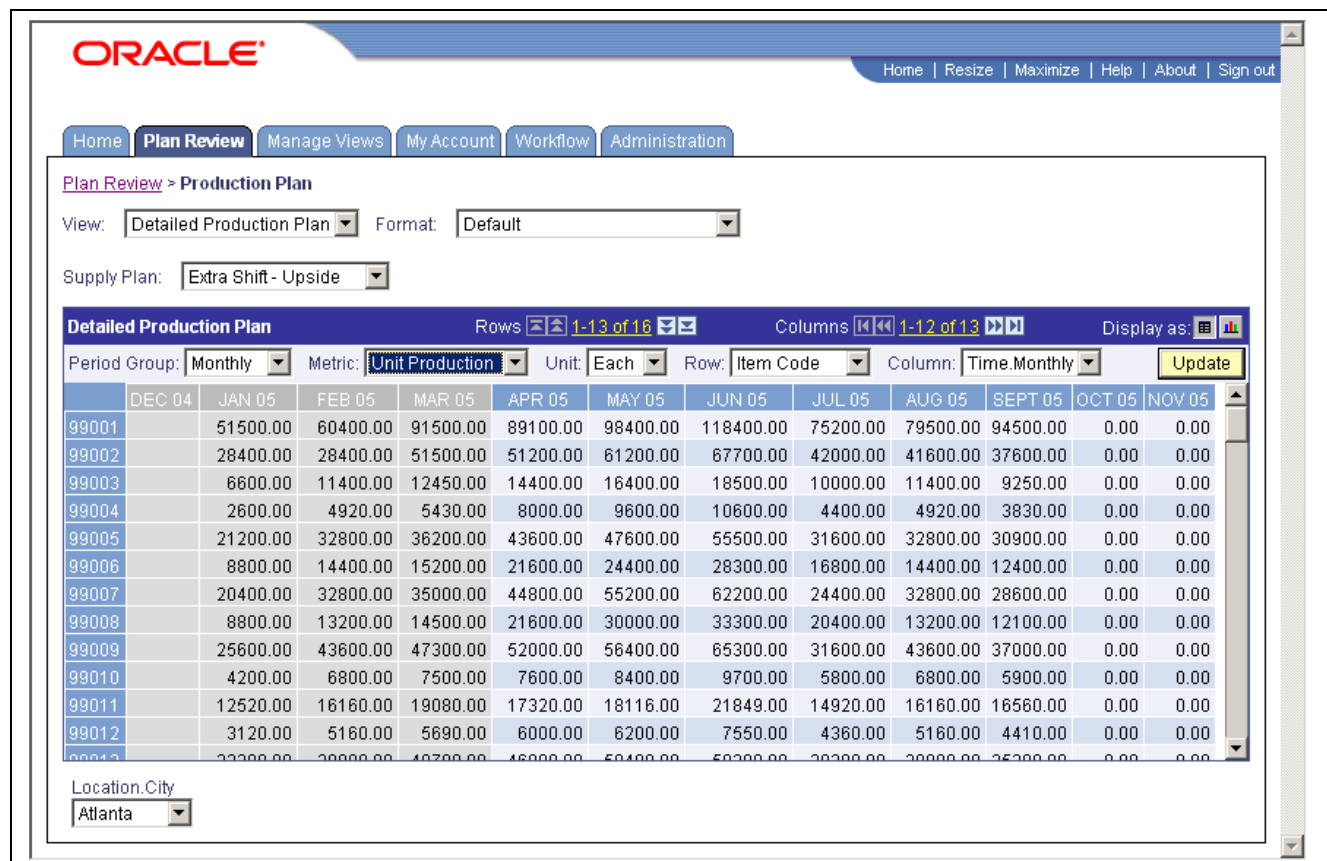
The Production Plan view displays the overall level of planned production for a given time period. The Production Plan is used during the supply review process to determine how an organization plans to meet demand based on its existing inventory and production capacity.

Large and complex environments are represented in the Production Plan using simplified, aggregated views. However, even at simplified levels it can be difficult to interpret deviations and trends in tabular data. For this reason, the Production Plan view supports graphical displays to provide a flexible and intuitive visualization of plan data and performance indicators. Use the Display As buttons to toggle between the table and graph display type.

The available Production Plan metrics are:

- Unit Production.
- Production Costs.

The following example shows a sample Production Plan view:



### Example Production Plan view

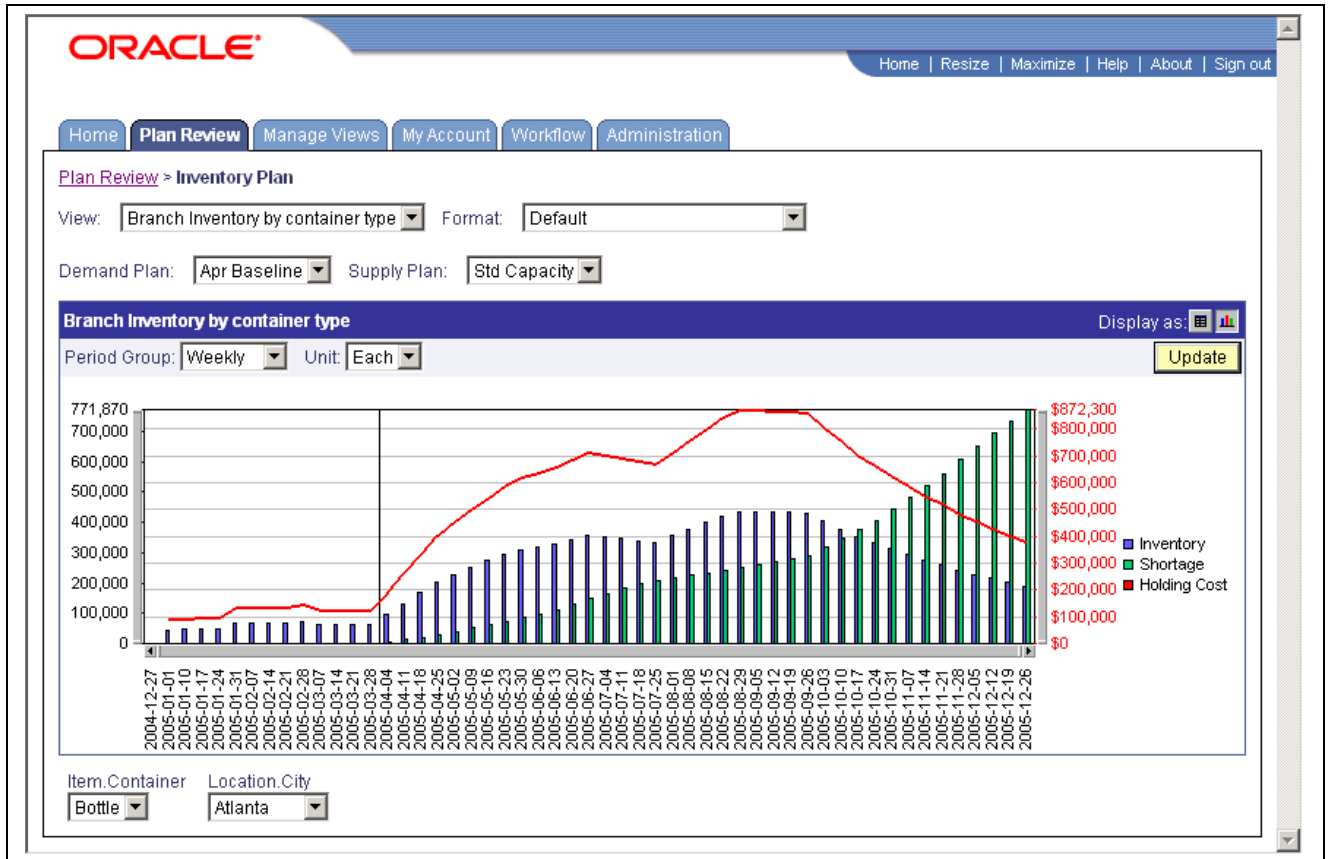
## Inventory Plan View

The Inventory Plan view displays product inventory levels for a given time period. Cells are shaded to highlight the start of the horizon, and you can view inventory levels by item or location. The Inventory Plan is used during the supply review process to determine how an organization plans to meet demand based on its existing inventory and production capacity.

The available Inventory Plan metrics are:

- Inventory.
- Shortage.
- Holding Cost.

The following example shows a sample Inventory Plan view:



Example Inventory Plan view

## Rough Cut Capacity Plan View

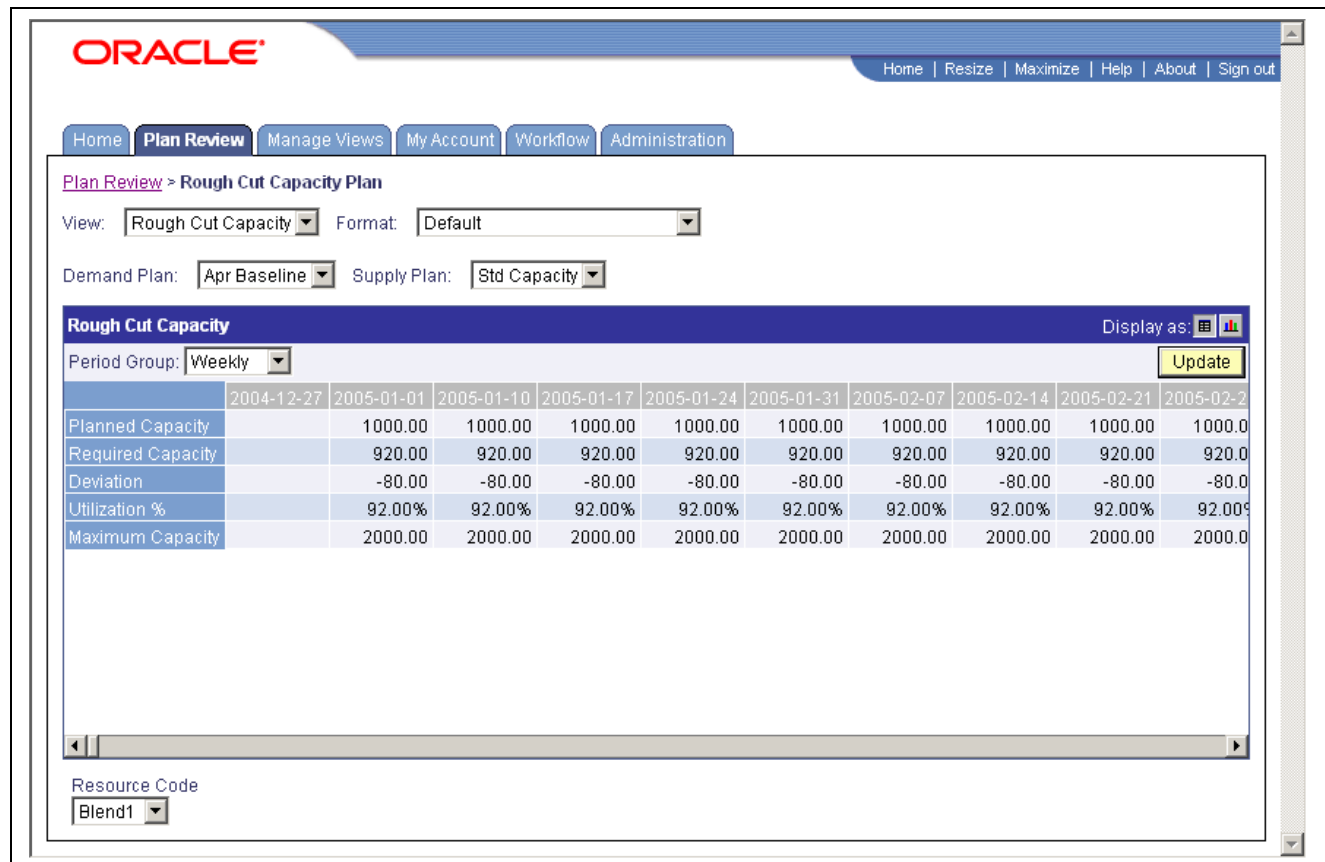
The Rough Cut Capacity Plan view displays your current and planned production capabilities for a given time period. Use this view during the reconciliation process to identify differences between planned production capacity, and your current resource utilization percentage.

The maximum capacity is the maximum feasible availability of a resource for the given time period, and is provided for reference only.

The available Rough Cut Capacity Plan metrics are:

- Planned Capacity.
- Required Capacity.
- Deviation.
- Utilization Percentage.
- Maximum Capacity.

The following example shows a sample Rough Cut Capacity Plan:



Example Rough Cut Capacity Plan view

## Combined Data Views

This section discusses:

- Consolidated view.
- Financial view.

### Consolidated View

The Consolidated data view displays your combined Demand, Production, and Inventory plans. This view is used during the reconciliation process to get a high-level view of all your plans.

Use the Consolidated view's Waterfall view format to see plan-over-plan changes and to identify trends. When viewing the Consolidated Plan using the Waterfall view format, numbers in brackets for the Inventory plan represent shortages.

**Note.** To use the Waterfall view format in the Consolidated data view, you should have two or more planning cycles worth of demand, production, and starting inventory data.

If any value is missing from the Consolidated Plan, then the demand fill percentage is not calculated.

The available Consolidated Plan metrics are:



- Demand.
- Production.
- Inventory.
- Shortage.
- Demand Fill Percentage.

The following example shows a sample Consolidated view:

**ORACLE**

Home | Resize | Maximize | Help | About | Sign out

Home | **Plan Review** | Manage Views | My Account | Workflow | Administration

Plan Review > Consolidated View

View: Declan's Consolidated Plan by Country Format: Default

Demand Plan: Apr Baseline Supply Plan: Std Capacity

**Declan's Consolidated Plan by Country** Display as:

Period Group: Weekly Unit: Case Update

	2005-03-14	2005-03-21	2005-03-28	2005-04-04	2005-04-11	2005-04-18	2005-04-25	2005-05-02	2005-05-09	2005-05-16
Demand	4735.00	4768.33	4768.33	4425.00	4425.00	4425.00	4425.00	6050.00	6275.00	6425.00
Production	4879.17	4879.17	4879.17	6158.33	6325.00	6408.33	6408.33	7008.33	7008.33	6966.67
Inventory	4879.17	4879.17	4879.17	6612.50	8512.50	10495.83	12479.17	13479.17	14395.83	15120.83
Shortage				0.00	0.00	0.00	0.00	41.67	225.00	408.33
Demand Fill %	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	99.31%	96.41%	93.64%

Item Major Group: Lager Location Location Groups: US branches

Example Consolidated view

## See Also

Demand Plan View

Production Plan View

Inventory Plan View

## Financial View

The Financial view displays key financial projections over time. Falling short or exceeding financial targets has a major impact on a company's operations and relationship with investors. Use the Financial view during the reconciliation process to evaluate your demand and production plans against budgeted targets such as revenue, cost, and margin.

Financial data is aggregated to the period group that you are currently viewing. When you click on a period group in the X-axis, you can drill down to view financial data for this time period, organized by other item, location, or channel attributes. For example, you can view margin by product group to help you identify which groups are not performing. From here you can drill-down for more detail.

The available Financial view metrics are:

- Unit.
- Average Selling Price.
- Revenue.
- Cost of Goods Sold per Unit.
- Cost of Goods Sold.
- Margin.
- Margin Percentage.

The screenshot shows the Oracle Financial View interface. At the top, there's a navigation bar with 'ORACLE' and links like Home, Resize, Maximize, Help, About, and Sign out. Below this is a sub-navigation bar with 'Home', 'Plan Review', 'Manage Views', 'My Account', 'Workflow', and 'Administration'. The main content area is titled 'Plan Review > Financial View'. It includes dropdowns for 'View: National Brand Revenue Forecast', 'Format: Default', and 'Demand Plan: Apr Upside'. Below these is a table titled 'National Brand Revenue Forecast' with columns for dates from 2005-03-14 to 2005-05-09. The table displays various financial metrics: Unit Demand, Average selling price, Revenue, COGS per unit, COGS, Margin, Margin %, and Adjusted Revenue. At the bottom, there's a dropdown for 'Item.Brand' set to 'Labatt'.

	2005-03-14	2005-03-21	2005-03-28	2005-04-04	2005-04-11	2005-04-18	2005-04-25	2005-05-02	2005-05-09	2005-05-16
Unit Demand	167380.00	167980.00	167980.00	229400.00	229400.00	229400.00	225400.00	293400.00	293400.00	293400.00
Average selling price	10.00	10.00	10.00	6.06	6.06	6.06	6.15	5.96	5.96	5.96
Revenue	1673800.00	1679800.00	1679800.00	1390600.00	1390600.00	1390600.00	1386600.00	1749400.00	1749400.00	1749400.00
COGS per unit	4.88	4.88	4.88	4.60	4.60	4.60	4.61	4.68	4.68	4.68
COGS	817070.00	819470.00	819470.00	1054300.00	1054300.00	1054300.00	1038300.00	1371950.00	1371950.00	1371950.00
Margin	856730.00	860330.00	860330.00	336300.00	336300.00	336300.00	348300.00	377450.00	377450.00	377450.00
Margin %	51.18%	51.22%	51.22%	24.18%	24.18%	24.18%	25.12%	21.58%	21.58%	21.58%
Adjusted Revenue	1482136.87	1690109.49	1856487.60	1989590.08	2088072.06	2323737.65	2512270.12	2663096.10	2783756.88	2922111.11

Example Financial view

## Display Types

This section describes Sales and Operations Planning display types, and discusses:

- Table display type.

- Graph display type.

## Table Display Type

The Table display type uses a tabular layout to present your plan data.

You select what data is displayed on the horizontal and vertical axes on the Dimensions page of the Add View wizard. The first two dimensions that you select display on the horizontal and vertical axes. Additional dimensions can be used to pivot the table. Pivot tables enable you to interact with views by rotating rows and columns to see different summaries of your plan data. Pivot tables are useful for comparing related totals, filter the data by displaying different pages, or display particular data in more detail.

For example, you can use the Demand Plan view to look at customer demand over time. Create a view to display time on the horizontal axis and your product families on the vertical axis. After identifying that demand for one particular product is down, you can exchange the product family vertical axis with your customers. The Demand Plan now displays demand for a single product for all customers. In this way you can use pivot tables to get more detail on your plan data.

ORACLE

Home | Resize | Maximize | Help | About | Sign out

Change Google Toolbar behavior and layout

Home | Plan Review | Manage Views | My Account | Workflow | Administration

Plan Review > Inventory Plan

View: Branch Inventory by container type Format: Default

Demand Plan: Apr Baseline Supply Plan: Std Capacity

Branch Inventory by container type Display as: [Table Icon] [Bar Icon]

Period Group: Weekly Unit: Each Update

	2005-03-14	2005-03-21	2005-03-28	2005-04-04	2005-04-11	2005-04-18	2005-04-25	2005-05-02	2005-05-09	2005-05-16	2005-05-23
Inventory	63910.00	63910.00	63910.00	95830.00	131150.00	166970.00	202790.00	227619.00	250748.00	272927.00	295106.00
Shortage				5500.00	13400.00	21300.00	29200.00	40000.00	50800.00	61900.00	73000.00
Holding Cost	127820.00	127820.00	127820.00	191660.00	262300.00	333940.00	405580.00	455238.00	501496.00	545854.00	589512.00

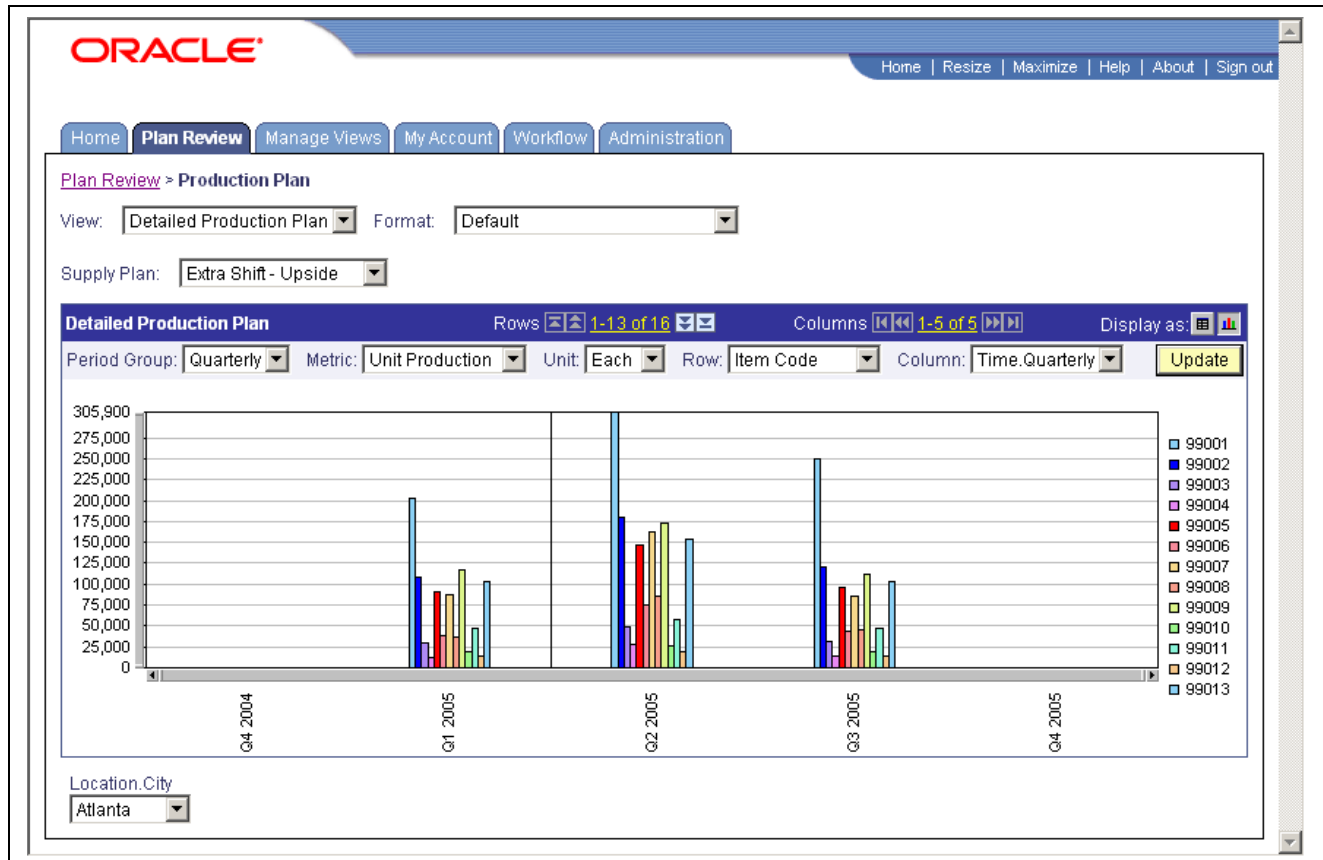
Item.Container: Bottle Location.City: Atlanta

Inventory Plan view using the Table display type

## Graph Display Type

The Graph display type uses a graphical layout to visualize plan data and performance indicators.

Sales and Operations Planning displays plans so that large and complex environments are represented in simplified, aggregated views. However, even at simplified levels it can be difficult to interpret deviations and trends in tabular data. For this reason, Sales and Operations Planning uses the graph display type to provide a flexible and intuitive visualization of plan data and performance indicators.



Production Plan using the Graph display type

Each Sales and Operations Planning view and display format uses a graph type that is most appropriate for the type of data being displayed. Clicking on Maximize allows you to remove the main navigation tabs to increase the graph area. Clicking Restore adds the navigation tabs. You can also set the granularity of the zoom and the area you are looking at by adjusting the size of the slider bar and dragging it across the screen.

The following table lists the graph type associated with each data view.

Data View	Display Format	Graph Type and Description
Consolidated View	Default	Combination line and bar graph with planning periods as the X axis. The bar graph displays inventory and shortage values, and the line graph displays demand and production.
	Compare to Last, Compare to Approved, Compare Current Plan Options, Compare Any Plans	Bar graph with two series representing the selected baseline and comparison metric.
	Waterfall	Bar graph that compares plan data from multiple planning cycles. Select which plan data series or metric you want to compare from the Metric drop-down box in the graph header.
Demand View	Default	Bar graph that you can customize to display any available dimension on the X- or Y-axis. Additional dimensions appear in combo boxes in the graph header.

Data View	Display Format	Graph Type and Description
	Compare to Approved, Compare Current Plan Options, Compare Any Plans	Line graph that compares the two series, whether they are from different plans in the same cycle or the same plan across multiple cycles.
Production View	Default	Bar graph that you can customize to display any available dimension on the X- or Y-axis. Additional dimensions appear in combo boxes in the graph header.
	Compare to Approved, Compare Current Plan Options, Compare Any Plans	Line graph that compares the two series, whether they are from different plans in the same cycle or the same plan across multiple cycles.
Inventory View	Default	Combination bar and line graph, with the bar graph representing inventory and shortage values, and the line graph representing inventory holding cost over time.
	Compare to Approved, Compare Current Plan Options, Compare Any Plans	Bar graph that compares inventory values between different plans from the same cycle, or compares the same plan from different cycles.
Rough Cut Capacity View	Default	Combination bar and line graph with the bar graph representing planned, maximum, and required capacity values, and the line graph representing the plan's utilization percentage.
	Compare to Approved, Compare Current Plan Options, Compare Any Plans	Bar graph that compares capacity values between different plans from the same cycle, or compares the same plan from different cycles.
Financial View	Default	Combination bar and line graph with the bar graph representing revenue, cost, and margin values. The Financial View graph has a double Y-axis so that revenue, cost and margin values can be plotted together with the resulting margin percentage overlaid as a line graph.
	Compare to Approved, Compare Current Plan Options, Compare Any Plans	Bar graph that compares revenue values between different plans from the same cycle, or compares the same plan from different cycles.

## View Formats

This section provides an overview of Sales and Operations Planning view formats, and discusses:

- The Default format.
- The Comparison format.
- The Dashboard format.

- The Waterfall format.

## View Format Overview

Sales and Operations Planning provides different ways to display data views. Each of these views highlights a different aspect of your plan data and is useful at different stages in your plan reviews.

Common to all view formats are display controls that configure how your plan data displays. These controls enable you to set the Period Group level, the metric to display, and the row and column dimensions. Display controls are context-sensitive and change depending on the data view you are using.

## Default Format

The default view format displays plan data in either tabular or graphical format. Unlike the other view formats, the default format does not make any comparisons between other plans or targets. Use the default view format to analyze and drill-down into your plan data to see it at the planning level in which you are most interested.

Sales and Operations Planning displays plans so that large and complex environments are represented in simplified, aggregated views. However, even at simplified levels it can be difficult to interpret deviations and trends in tabular data. For this reason, Sales and Operations Planning supports graphical displays to provide a flexible and intuitive visualization of plan data and performance indicators. Use the Display As buttons to toggle between the table and graph display type.

### See Also

Table Display Type

Graph Display Type

Working with Views

## Comparison Format

The Comparison view format compares key metrics between two planning cycles, and allows you to:

- Compare the current plan option to the approved plan from the previous planning cycle.
- Compare the current plan option to the last approved plan.
- Compare the current plan option to historical data and plans.
- Compare the current plan option to another current plan option.

All data views can be displayed using the Comparison format. The Comparison format displays the base and comparison values and calculates the absolute and percentage difference between the two values. Use the display controls to specify the metric that you want to compare and to set the base and comparison cycles.

---

**Note.** You must have at least two planning cycles worth of data to use the Comparison format.

---

When you create and execute a Net Change report, exceptions are linked to the comparison view format for the view with which they are associated. The following example shows a Demand Plan view using the Comparison format:

**ORACLE**

Home | Resize | Maximize | Help | About | Sign out

Home | **Plan Review** | Manage Views | My Account | Workflow | Administration

Plan Review > Demand Plan

View: **Detailed Demand Review** Format: **Compare Any Plans**

Base Cycle: **APR-05** Demand Plan: **Apr Upside**

Comparison Cycle: **MAR-05** Demand Plan: **Mar Baseline**

**Detailed Demand Review** Display as:

Period Group: **Monthly** Metric: **Unit Demand** Unit: **Each** **Update**

	JAN 05	FEB 05	MAR 05	APR 05	MAY 05	JUN 05	JUL 05	AUG 05	SEPT 05	OCT 05	NOV 05	DEC 05
Baseline	27600.00	26000.00	36600.00	32400.00	36000.00	47500.00	36000.00	35200.00	42500.00	34500.00	34400.00	43000.00
Comparison	27600.00	26000.00	35000.00	31600.00	33600.00	44600.00	31200.00	28400.00	35900.00	29200.00	28700.00	36400.00
Deviation	0.00	0.00	-1600.00	-800.00	-2400.00	-2900.00	-4800.00	-6800.00	-6600.00	-5300.00	-5700.00	-6600.00
Absolute Deviation	0.00	0.00	1600.00	800.00	2400.00	2900.00	4800.00	6800.00	6600.00	5300.00	5700.00	6600.00
% Deviation	0.00%	0.00%	-4.37%	-2.47%	-6.67%	-6.11%	-13.33%	-19.32%	-15.53%	-15.36%	-16.57%	-15.36%
Absolute % Deviation	0.00%	0.00%	4.37%	2.47%	6.67%	6.11%	13.33%	19.32%	15.53%	15.36%	16.57%	15.36%

Channel Code: **1001** Item Code: **99001** Location/ City: **Atlanta**

Demand Plan view using the Comparison view format

## Dashboard Format

The Demand Plan Dashboard view format provides a high-level view of plan data. The Dashboard compares plan data against your defined targets and thresholds to give you an indication of your plan's success or failure. For each available metric it displays a current and target value and the total and percentage deviation between these values. A status indicator shows whether or not the plan has achieved the defined targets.

Before using the Dashboard, you must set targets and thresholds for the metrics that you are viewing. If a metric does not have an associated target, click the Set Target link. Target and threshold values automatically aggregate to the Period Group for which your view is set. For example, if you have monthly product demand set at 10 units, this value translates into yearly demand of 120 units.

This view format helps identify net changes to the most recent plan and your defined targets to understand any change in strengths, weaknesses, opportunities, and threats. From the Dashboard you can drill down to identify the cause of any exceptions and potential constraints.

The following example shows a Demand Plan view using the Dashboard view format:

The screenshot shows the Oracle Plan Review interface. At the top, there's a navigation bar with links: Home, Resize, Maximize, Help, About, Sign out. Below this, a secondary navigation bar includes: Home, Plan Review (selected), Manage Views, My Account, Workflow, and Administration. The main content area is titled 'Plan Review > Demand Plan'. It features two dropdown menus: 'View: Detailed Demand Review' and 'Format: Dashboard'. Below these are 'Cycle: APR-05' and 'Demand Plan: Apr Upside'. A section titled 'Detailed Demand Review' contains a 'Period Group: Monthly' dropdown, a 'Unit: Each' dropdown, and an 'Update' button. A table displays metrics for 'Unit Demand' and 'Revenue'. The 'Unit Demand' row shows a current value of 32400.00, a target of 36800.00 (highlighted in red), a deviation of -4400.00, a percentage deviation of -11.96%, and a status of 'X'. The 'Revenue' row shows a current value of 324000.00, a target of 318000.00 (highlighted in red), a deviation of 6000.00, a percentage deviation of 1.89%, and a status of a green checkmark. At the bottom, there are four dropdown menus: 'Channel Code: 1001', 'Item Code: 99001', 'Location.City: Atlanta', and 'Time.Monthly: APR 05'.

Metric	Unit	Current	Target	Deviation	% Deviation	Status
Unit Demand	Each	32400.00	36800.00	-4400.00	-11.96%	X
Revenue		324000.00	318000.00	6000.00	1.89%	✓

Demand Plan view using the Dashboard view format

## See Also

Targets and Thresholds

## Waterfall Format

The Waterfall view format enables you to compare changes to plans from planning cycle to planning cycle. By comparing different plan versions, you can identify any changes since the last plan was adopted. Examining net change in plans is a key method used in Sales and Operations Planning to focus the data review process. In a Waterfall display, the rows of the table are reserved for the various plan versions that you are comparing and the columns are the different planning periods.

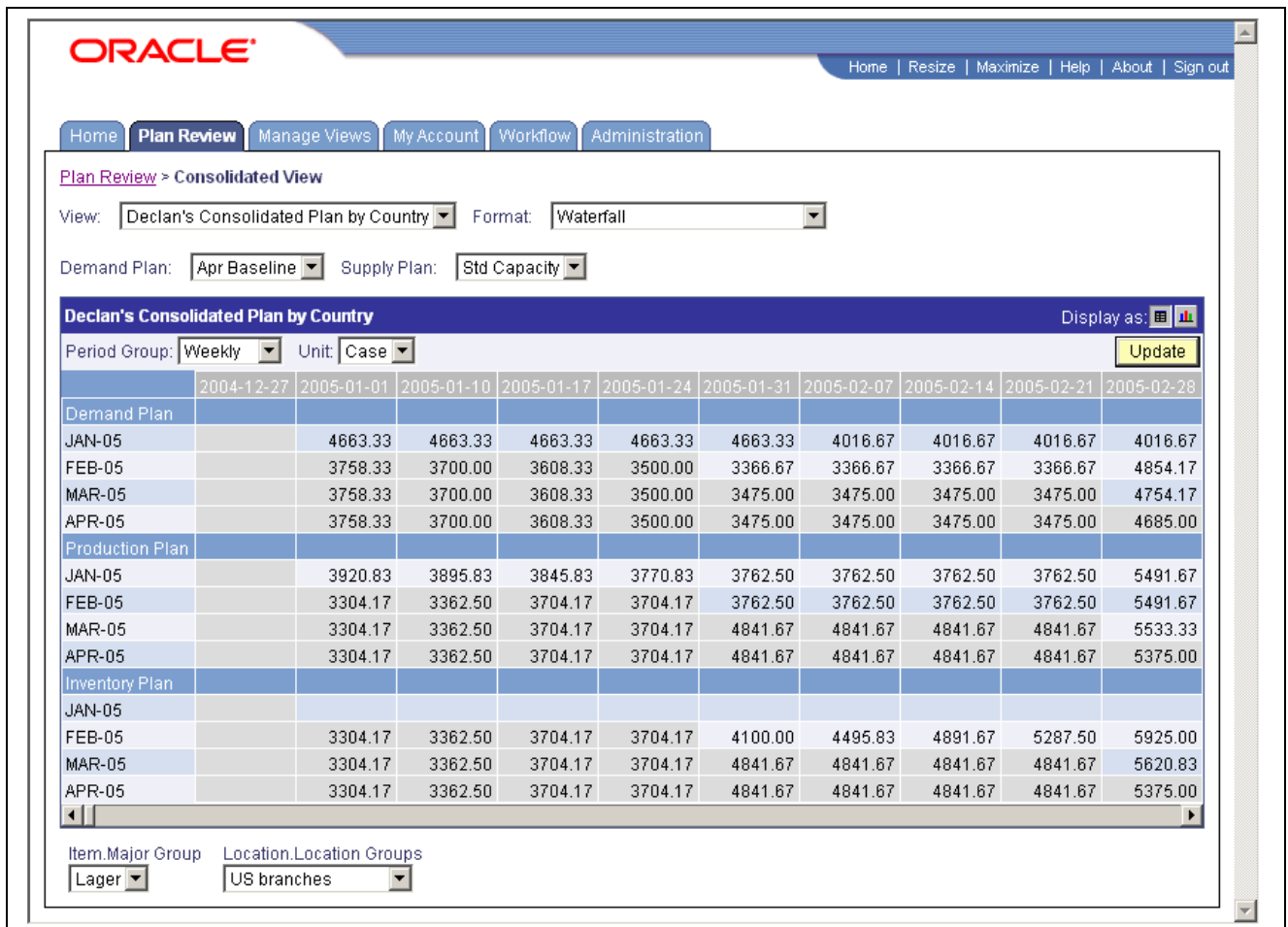
---

**Note.** The Waterfall format is available only in the Consolidated Data view.

---

The following example shows a Consolidated view using the Waterfall view format:





Consolidated Data view using the Waterfall view format

## Units of Measure

With Sales and Operations Planning you have the flexibility to display planning data in the unit of measure that is most relevant to you. Often the unit of measure changes depending on the granularity of your data. For example, when a brewery is reviewing its long-term forecasts it is most interested in the amount of beer being brewed and not the individual quantities of cans versus bottles.

Units of measure and their conversion rates are imported for each item in the model. Each imported item has an associated planning unit that represents how the item is packaged and sold at its most granular level. Sales and Operations Planning views dynamically aggregate units of measure for all items with pre-defined conversion rates. Having a conversion factor for units of measure allows companies that sell products in many different packages to aggregate items to the individual item level. For example, even though a soft drink manufacturer sells cans in cases of 6, 12 and 24, it uses 12 ounce cans as its planning unit.

By default, Sales and Operations Planning views display items using the Each unit of measure. Each always has a value of one item, and the each unit of measure is always valid for every aggregation point in the model. When viewing counts for multiple items, each represents the count of all items viewed. For example, 100 one liter bottles and 60 6-packs of cans is represented as 160 “each”.

You can also choose from the available options from the unit of measure drop-down list. Changing any unit of measure information results in all views needing to be re-executed. If you view items using a unit of measure without a defined conversion, the view displays “N/A” in those fields.

Targets for absolute metrics must be set using a single unit of measure. Sales and Operations Planning aggregates the targets to the chosen unit of measure, if possible. Views that do not have data for that unit of measure and without a conversion factor do not appear in the dashboard.

### See Also

Importing Data

Managing Targets for Absolute Metrics

UnitsOfMeasure.AddItemUomFromXml Command

---

## Dimension Properties

This section discusses:

- Dimension properties overview.
- Dimension property names and values.

### Dimension Properties Overview

The EnterpriseOne Sales and Operations Planning model defines the environment within which your business uses and maintains item, location, channel, resource and time information. Items represent a good or service. Locations represent a place where products are manufactured, shipped, or sold. Channels represent a set of companies, distributors, or individuals who participate in the movement of goods. Resources represent anything required to produce items.

EnterpriseOne Sales and Operations Planning provides multiple data views that enable you to display, organize, and manipulate model data by item, location, channel, and resource.

### Dimension Property Names and Values

Item, location, channel, and resource each have distinct properties that are used to describe them. Properties are name-value pairs. A property name describes what type of information is stored in a property—for example, region, country, or continent. All channels share the same property names. All locations share the same property names. All items share the same property names. However, individual items, locations, channels, and resources have a specific property value that is associated with each property name. For example, different locations might have the property values Boston, Denver, or Toronto associated with the property name city. Property names group data into aggregation levels while the property values associated with that name define the groups on that level. At the region level, groups can exist for Boston, Denver, and Toronto.

---

## Targets and Thresholds

One of the primary objectives of the sales and operations planning process is setting realistic and achievable targets. Targets are objectives that an organization is trying to achieve, and are set as part of the financial planning process. Targets may be financial, capacity-based, a service objective, and so on. Targets enable an organization to set goals and then track their progress at meeting these goals. Falling short or exceeding targets can have a major impact on a company's operations and relationships with its investors.

EnterpriseOne Sales and Operations Planning uses the “manage by exception” technique to enable you to quickly pinpoint those areas of concern within a plan. Exceptions are generated from the Dashboard view and using Net Change and Metric Exception reports when planning cycle data differs from budgeted targets.

Thresholds are a percentage of targets. You can set threshold percentages that represent “good” and “fair” results, which helps you to monitor varying levels of plan performance. Because thresholds are usually similar between product groups, locations, and so on, you can use wild cards to set thresholds quickly. Your threshold tolerance is also dependent on the relative importance of the customer, item, or location.

Relative metrics are based on other values. These are comparison metrics and are usually calculated as a percentage or rate. For example, you may have a goal of a business unit having a certain profitability, so you want each unit to have 40 percent profitability. Absolute metrics means actual value. For example, revenue is an absolute metric and represents the total income, in dollars, produced by a given source.



## CHAPTER 4

# Configuring EnterpriseOne Sales and Operations Planning

This chapter provides an overview of configuring EnterpriseOne Sales and Operations Planning, and discusses how to:

- Administer users.
- Administer roles.
- Manage planning periods.
- Manage planning cycles.
- Work with logs.

---

## Understanding Configuring Sales and Operations Planning

This section discusses:

- User administration.
- Logs.
- Planning periods.
- Planning cycles.
- Custom metrics.

### User Administration

Sales and Operations Planning supports multiple account types and user roles that allow administrators to protect sensitive information by controlling access to data views. When you create a Sales and Operations Planning user, you assign them to one of three account types: administrators, view owners, and users.

Administrators have access to the entire Sales and Operations Planning system, and can perform the tasks of users and view owners. In addition, they can also define periods and cycles, configure logging, set the user account type, and create and remove user accounts.

---

**Note.** Sales and Operations Planning is configured with a default administrator account that cannot be deleted. The account user name and password is admin. You should change the default password after logging in to the system for the first time. If you delete all user accounts, Sales and Operations Planning re-creates the default administration account with the original name and password.

---

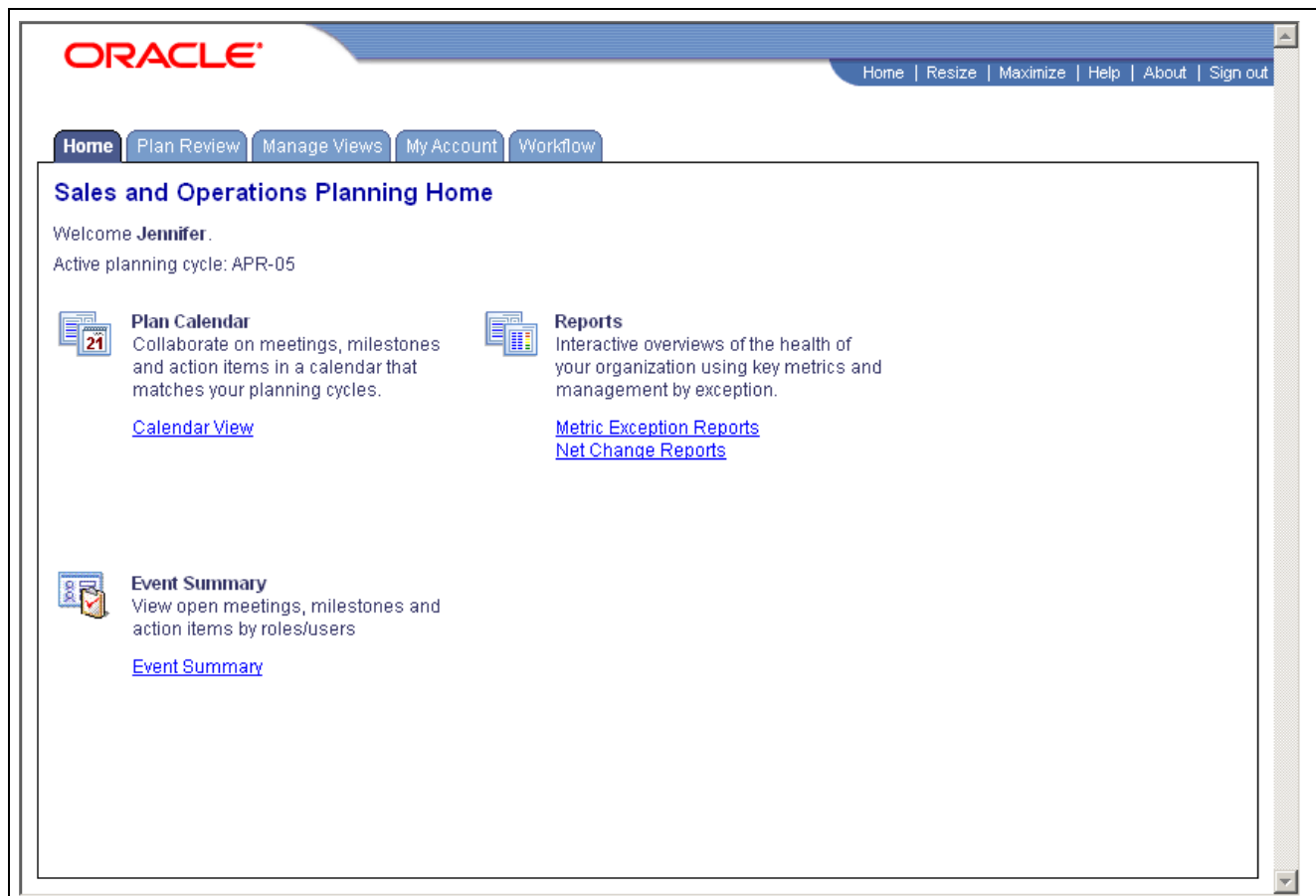
View owners can perform the tasks of users. In addition, they can also create and execute views, grant view access permissions to users, execute workflows not flagged as administrator-only, and manage target metrics for views that they own.

Normal users can access (but not modify) views for which they have been granted permissions, generate reports, and execute workflows not flagged as administrator-only.

Before users can access Sales and Operations Planning, they must have a user ID and password, both of which are assigned by the system administrator. To assign this information, the system administrator adds user accounts on the Users page. You can change your password in the My Account page.

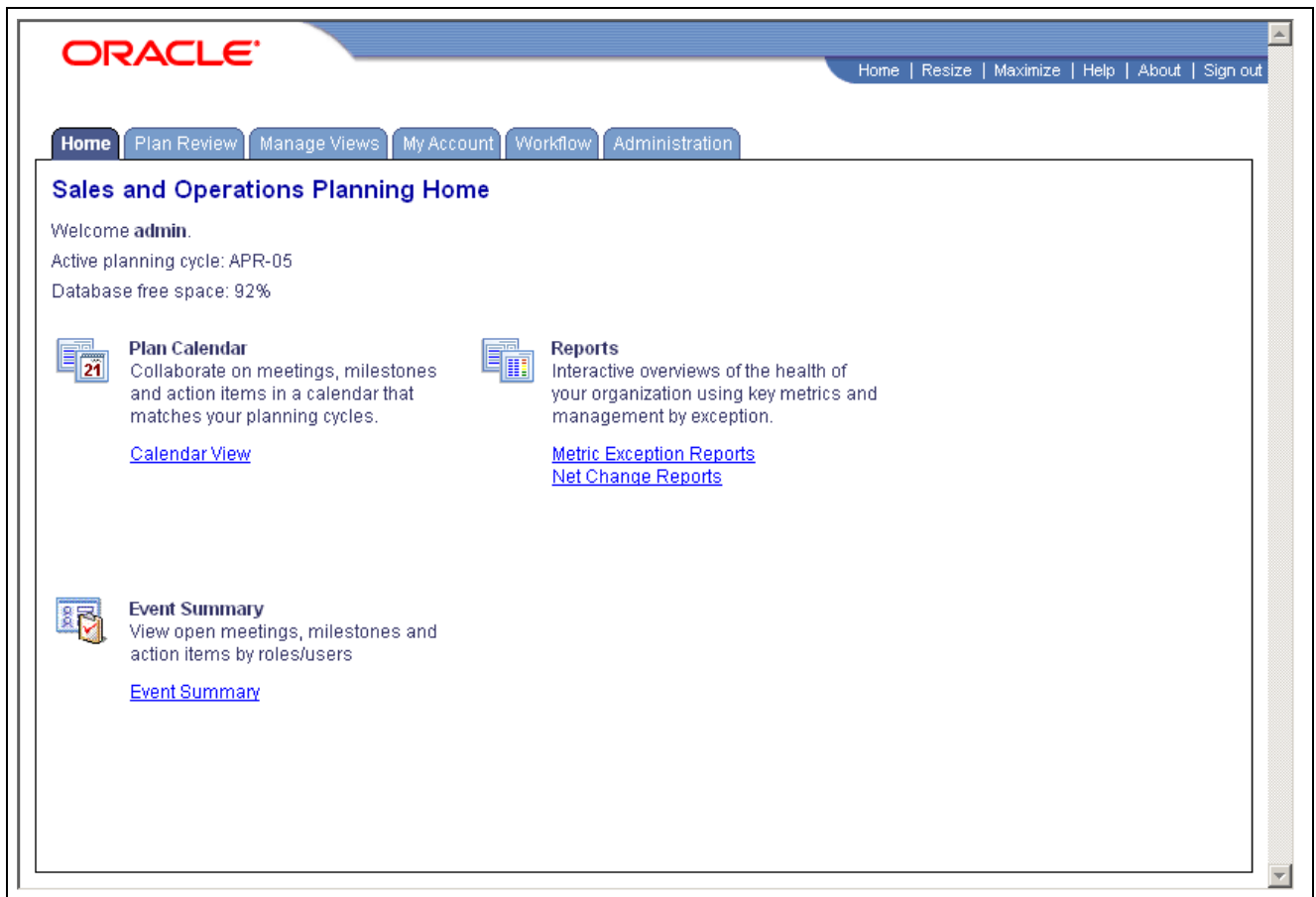
The Sales and Operations Planning desktop configuration depends on your account type. Normal users and view owners have access to Sales and Operations Planning functions found on the Home, Plan Review, Manage Views, My Account and Workflow pages. Administrators have access to these functions in addition to the administration tasks accessed on the Users, Calendar and Logging pages. Links to views that do not exist or for which users do not have permissions appear grayed.

The following example illustrates the home page configuration for normal users:



Sales and Operations Planning desktop for normal users

The following example illustrates the home page configuration for administrators:



Sales and Operations Planning desktop for Administrators

## User Roles

User roles enable administrators to manage high-level access security by setting permissions for specific groups of people with similar job functions. For example, you may want to create a role called assembly line managers, and give this role permissions to only access the Inventory and Production views. Using roles enables administrators to easily set up view permissions without having to set them for each individual user.

Sales and Operations Planning administrators can create as many roles as are needed and you can assign multiple roles to a user.

## Targets and Thresholds

Administrators and view owners have permissions to set targets and thresholds. Administrators can set targets and thresholds for any data view, and view owners can set targets for the views they have permissions to access.

## Reports

All users can create and view reports, as long as they have permissions to see the views that the report accesses.

## Logs

A log file is created whenever Sales and Operations Planning is launched. The log presents information about different categories of system events.

Events are placed into functional categories, and event types have one of three levels:

- Error

A critical problem has occurred. Error messages are most often associated with the environment, web server, and so forth. You should record error messages to enable the Technical Support department to resolve the problem.

- Warning

There was a non-critical problem with the application. For example, a user may have entered an incorrect password.

- Info

This type of event provides update on the current system state. All application activity is stored as an info message, and you can use it to determine the cause of an error.

By default, Sales and Operations Planning logs all error messages and application server warnings. These messages are written to the database and to the log file `\scp\vers_8.12\so-plan\logs\NexusServer.log`.

You can configure Sales and Operations Planning to filter the log for specific patterns. When you configure the log to display Info events, all Warning and Error events are automatically displayed. When you display Warnings, Error events are also displayed.

Category patterns enable you to filter the error log to display only messages that match a specified text string.

## Planning Periods

Planning periods enable you to configure Sales and Operations Planning to match your fiscal or operational calendar. Using the planning period interface in Sales and Operations Planning, you can map weekly time buckets to fiscal years, quarters, and months. This flexible approach enables you to accurately represent your corporate fiscal calendar. Depending on your business process, you can add planning periods to cover the entire planning horizon, or you can add periods at the beginning of each cycle.

Sales and Operations Planning uses weeks as its most discrete unit of measure. By manually entering planning periods, you determine how weeks fall into fiscal months, quarters, and years. This planning period configuration determines how Sales and Operations Planning displays aggregated data. You can change the start and end day of fiscal weeks to match your particular fiscal patterns.

## Planning Cycles

The sales and operations planning process usually follows a monthly cycle, with each of these cycles representing an iteration of the planning process. Each time you start a new planning iteration, you advance the system to a new planning cycle. Planning cycles enable you to store all data related to a specific planning cycle under a unique name. This makes it easy to access, organize, and compare multiple plan data snapshots.

To minimize business disruption and manage large volumes of data, the sales and operations planning process most often focuses on the net change between current plan data and previous planning cycles or longer-term targets. These planning cycles are typically aligned to a corporate fiscal calendar to ensure that operational plans can be easily compared with monthly and quarterly based financial plans and projections.

The first step in the sales and operations planning process is to set the current planning cycle. All imported planning data queries, views, and calculated metrics are automatically associated with this planning cycle.



## Custom Metrics

Sales and Operations Planning supports many frequently-used metrics to give you “out of the box” best practice process and data support for the sales and operations planning activities of a company. However, you may want to incorporate your own custom metrics in addition to those provided, to view data series’ of your choice alongside those data series that are standard components of the Sales and Operations Planning model.

Some examples of custom metrics include the number of inventory turns or the days of supply and plan costs. These examples are very commonly-used, but have very specific formula parameters that make these metrics individual to your business and are therefore not included in the default Sales and Operations Planning configuration.

There are no restrictions on how you choose to create custom metrics and calculate the data. For example, you might write a Tcl script that generates custom metric data every time you run a data import, or you might import these metrics from an external system. The Custom metric definition and associated data is imported to Sales and Operations Planning using the CustomMetricList.xml file and the Sales and Operations Planning Application Shell (SASH). Once created they can be reviewed, associated to views, and updated using the Sales and Operations Planning interface.

Use the Custom Metrics page to view the definition of each custom metric currently in the model. Expand the folder icon to view the custom metric type, the period group, dimensions and for a list of all views that use this metric. Clicking the data view name automatically opens the view.

The screenshot displays the Oracle Custom Metrics page. The top navigation bar includes the Oracle logo and links for Home, Resize, Maximize, Help, About, and Sign out. Below this is a secondary navigation bar with links for Home, Plan Review, Manage Views (selected), My Account, Workflow, and Administration. The main content area is titled 'Manage Views > Custom Metrics' and contains a table of custom metrics.

Name	Last Update
60 Day MAPE	2006-05-30 10:38:55
Customer Service	2006-05-30 10:38:55
<b>Type:</b> Global <b>Period Group:</b> Monthly <b>Dimensions:</b> Item.Brand, Channel.Channel Type <b>Assigned Views:</b> <b>Compatible Views:</b> <a href="#">Projected Financials - National Brand &amp; Channel</a> <a href="#">National Brand Revenue Forecast</a>	
Inventory Turns	2006-05-30 10:38:55
<b>Type:</b> Global <b>Period Group:</b> Monthly <b>Dimensions:</b> Item.Brand, Channel.Channel Type <b>Assigned Views:</b> <b>Compatible Views:</b> <a href="#">Projected Financials - National Brand &amp; Channel</a> <a href="#">National Brand Revenue Forecast</a>	
Adjusted Revenue	2006-05-30 10:38:55
<b>Type:</b> Demand Plan <b>Period Group:</b> Weekly <b>Dimensions:</b> Item.Brand <b>Assigned Views:</b> <a href="#">National Brand Revenue Forecast</a> <b>Compatible Views:</b> <a href="#">National Brand Revenue Forecast</a>	

Custom Metrics page

## Custom Metric Types

Sales and Operations Planning supports the Global, Cycle, Demand Plan, and Supply Plan custom metrics types. The custom metric you use depends on the type of planning data it uses:

<b>Global</b>	Use the Global custom metric type when there is a single data series for the entire application. For example, sales history exists in the model as a single data series used for all planning cycles and data views.
<b>Cycle</b>	Use the Cycle custom metric type when there is one data series per planning cycle. For example, supply or demand forecasts change with each planning cycle.
<b>Demand Plan</b>	Use the Demand Plan custom metric type when working with plan data that is specific to each Demand Plan. For example, a custom metric based on safety stock levels is specific to each Demand Plan.
<b>Supply Plan</b>	Use the Supply Plan custom metric type when working with plan data that is specific to each Supply Plan. For example, a custom metric based on the number of days of inventory cover is specific to each Supply Plan.

## Custom Metrics Definition

To add a custom metric to Sales and Operations Planning you must edit or create the CustomMetricList.xml file. The following example shows a sample CustomMetricList.xml file that contains a custom metric 95% Forecast:

```
<?xml version="1.0" encoding="utf-8"?>
<customMetricList>
<customMetric name="95% Forecast" type="Demand Plan" planningPeriodGroup="Weekly">
<header>
<element dimension="Item" propertyName="Brewer"/>
<element dimension="Location" propertyName="City"/>
<element dimension="Channel" propertyName="Company"/>
</header>
<seriesList cycle="Cycle 3" demandPlan="DP3.1">
<series>
<key field1="Labatt Ontario Breweries Limited" field2="Ottawa" field3="Liquor⇒
Control Board of Ontario"/>
<valueList startPeriod="2004-01-01">
<value>7</value>
<value>4</value>
<value>6</value>
</valueList>
</series>
</seriesList>
</customMetric>
</customMetricList>
```

<b>&lt;customMetric name&gt;</b>	Name of the custom metric that you are defining.
<b>&lt;customMetric type&gt;</b>	Custom metric type. Valid values are:

	<ul style="list-style-type: none"> <li>• Global.</li> <li>• Cycle.</li> <li>• Demand Plan.</li> <li>• Supply Plan.</li> </ul>
<b>&lt;customMetric planningPeriodGroup&gt;</b>	<p>Planning period to which this custom metric applies. Valid values are:</p> <ul style="list-style-type: none"> <li>• Weekly.</li> <li>• Monthly.</li> <li>• Quarterly.</li> <li>• Yearly.</li> </ul>
<b>&lt;header&gt;</b>	Defines the dimensions used for each field in the <key> element.
<b>&lt;seriesList&gt;</b>	Defines each data series in the XML file. The number of attributes within this element depends on the metric type. For Global custom metrics, no attributes are required. For Cycle custom metrics, you must specify the <cycle> attribute. For Demand Plan and Supply Plan metrics, you must specify the <cycle> and <demandPlan>/<supplyPlan> attribute.
<b>&lt;key&gt;</b>	Defines the dimension values for the custom metric key.
<b>&lt;valueList startPeriod&gt;</b>	Defines the start period for the specified custom metric. Each <valueList> element has one or more <values> child elements that contains metric data.

### See Also

Adding Custom Metrics to Views

---

## Administering Users

This section discusses how to:

- Add users.
- Give users administrator privileges.
- Delete users.
- Change passwords.

## Pages Used to Administer Users

Page Name	Navigation	Usage
User Manager	Administration, Manage Users.	Add, remove and edit user profiles.
Add User	Administration, Manage Users, Add.	Add a new user account to Sales and Operations Planning.
Edit User	Administration, Manage Users, Edit.	Edit an existing user account.

## Adding Users

Access the Manage Users page.

---

**Note.** You must have administrator privileges to access the Manage Users page.

---

The Manage Users page displays a list of all current Sales and Operations Planning accounts.

To add a user:

1. Click Add.
2. In the User ID field, type a unique identifier for the user.

The user ID can be a short form of the user's name or the user's role.

1. In the Full Name field, type the user's name.
2. In the Type Password field, type a password that the user types to access the Sales and Operations Planning system.

The password must be at least eight characters long, and it can consist of both numbers and letters.

1. In the Confirm Password field, retype the password.
2. In the Account Type drop-down list box, select the type of user account that you want to create.
3. In the Roles section, do the following:
  - In the Available list, select the user role category to which you want to add the new user.
  - Click the arrow button to add the user role to the Assigned list.

Repeat the process for each user role that you want to assign to the user.

4. Click Save.

## Deleting Users

Access the Manage Users page.

---

**Note.** You cannot delete the default administrator account.

---

To delete a user:

1. In the Select column, select the ID of the user that you want to remove.

2. Click Delete.

The selected user account is deleted from Sales and Operations Planning.

## Editing User Data

Access the Manage Users page.

1. In the Select column, select the ID of the user that you want to edit.
2. Click Edit.
3. Make the desired changes to the user account and then click Save.

## Changing Passwords

Access the Manage Users page.

To change a user password:

1. In the Select column, select the user account whose password you want to change.
2. Click Change Password.
3. In the Type Password field, type the password to access the Sales and Operations Planning system.
4. In the Confirm Password field, retype the new password.

---

## Administering User Roles

This section discusses how to:

- Add user roles.
- Edit user roles.
- Delete user roles.

## Page used to Administer User Roles

Page Name	Navigation	Usage
Manage Roles	Administration, Manage Roles.	Add, remove and edit user roles.
Add Role	Administration, Manage Roles, Add.	Add a role.

## Adding User Roles

Access the Manage Roles page.

1. In the Role ID field, type a descriptive role identifier.
2. Click Save.

The role is added to the Manage Roles page.

## Editing User Roles

Access the Manage Roles page.

1. In the Select column, select the role that you want to edit.
2. Click Edit.
3. Make the desired changes to the role and then click Save.

## Deleting User Roles

Access the Manage Roles page.

1. In the Select column, select the role that you want to delete.
2. Click Delete.

---

## Managing Planning Periods

This section discusses how to add a planning period to the planning horizon.

### Pages Used to Manage Planning Periods

Page Name	Navigation	Usage
Manage Planning periods	Administration, Manage Planning Periods	Work with planning periods.
Manage Planning periods	Administration, Manage Planning Periods, Delete	Delete a planning period from the planning horizon.

## Adding Planning Periods

Access the Manage Planning Periods page.

The Manage Planning Periods page displays all defined planning periods in your planning horizon. You can only add periods to the start or end of the horizon.

To add planning periods:

1. In the Start Date field, enter the first day of the new period, using the date format YYYY-MM-DD.

---

**Note.** When adding planning periods to the start of the planning horizon, the start dates must fall before the first period. Start dates for periods added to the end of the horizon should fall after the last period.

---

2. In the Month, Quarter, and Year fields, indicate to which fiscal month, quarter, and year the planning period belongs.
3. Click Add.

### See Also

Periods AddPeriod Command

---

## Managing Planning Cycles

This section discusses how to:

- Add planning cycles.
- Delete planning cycles.
- Set the current planning cycle.
- Manage approvals.

### Pages Used to Manage Planning Cycles

Page Name	Navigation	Usage
Manage Planning Cycles	Administration, Manage Planning Cycles	Work with planning cycles.

### Adding Planning Cycles

Access the Manage Planning Cycles page.

Planning cycles use planning periods to define the start date. Before creating planning cycles, ensure that you have defined all necessary planning periods.

To add a planning cycle:

1. In the Cycle Name field, enter a descriptive name for the planning cycle.
2. In the Start Date field, select the planning period that starts the cycle.
3. Click Add Cycle.

### Deleting Planning Cycles

Access the Manage Planning Cycles page.

---

**Warning!** Deleting a planning cycle also deletes all associated plan data.

---

To delete a planning cycle, click Delete Cycle. The planning cycle is removed from the system.

### Setting the Current Planning Cycle

Access the Manage Planning Cycles page.

To set the current planning cycle, click Set Current Cycle beside the appropriate cycle name.

### Managing Approvals

Access the Manage Planning Cycles page.

To manage approvals:

1. In the Select column, select the planning cycle for which you want to manage approvals.

2. Click Manage Approvals.

The Manage Approvals page appears.

3. Do one or more of the following:

- In the Approved Demand Plan drop-down list box, select the demand plan that you want approve for this planning cycle.
- In the Approved Supply Plan drop-down list box, select the supply plan that you want to approve for this planning cycle.

To remove an approval for a demand or supply plan, choose None.

4. Click Save.

---

## Working with Logs

This section discusses how to:

- Clear log events.
- Add log patterns.
- Clear all log patterns.

### Pages Used to Work with Logs

Page Name	Navigation	Usage
View Event Log	Administration, View Event Log	View the system log.
Configure Logging Patterns	Administration, Configure Logging	Set logging levels and configure logging patterns.

### Clearing Log Events

Access the View Event Log page.

To remove events from the system log:

1. In the Up to date field, enter a threshold date in the format YYYY-MM-DD.
2. Click Clear Events.

All events with a timestamp before the selected threshold are removed from the database and log file.

### See Also

Logs

Log ClearDbLog Command

Log ClearFileLog Command



## Adding Log Patterns

Access the Configure Logging Patterns page.

To add log patterns:

1. From the Level drop-down list box, select the logging level to which you want to add a pattern.
2. Type the text string that you want to use for filtering, and then click Add Category Pattern.

Use the asterisk to define a wildcard pattern. For example, web\* matches all events that start with web.

### See Also

Logs

Log LogEvent Command

## Clearing All Log Patterns

Access the Configure Logging Patterns page.

To clear all log patterns:

1. In the Up to date field, enter a threshold date in the format YYYY-MM-DD.
2. Click Clear Events.

All log events older than the specified date are removed from the Log Events page

### See Also

Logs

Log StopAllLogging Command



## CHAPTER 5

# Managing the Sales and Operations Planning Process using the Plan Calendar

This chapter provides an overview of the Plan Calendar.

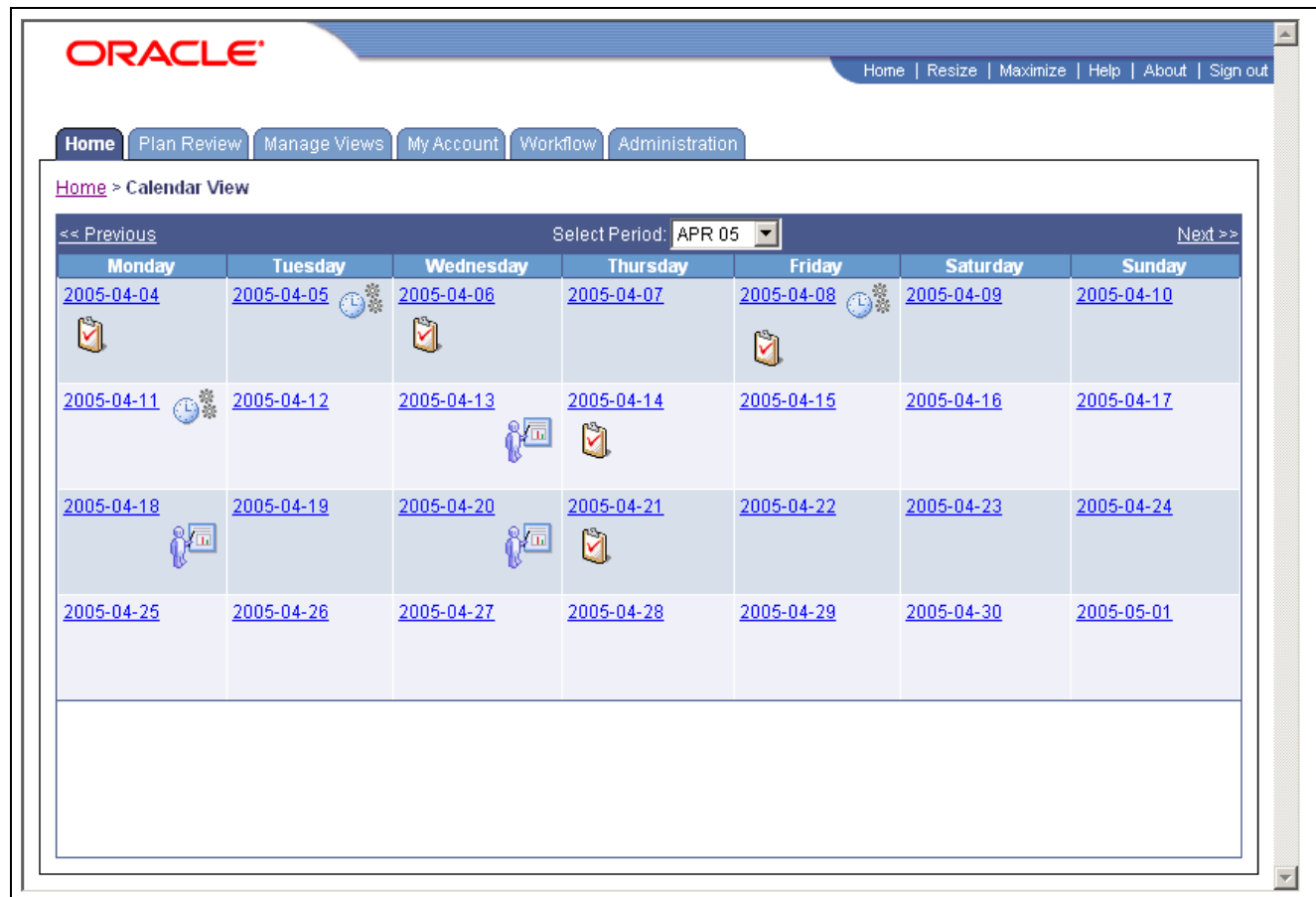
---

### Understanding the Plan Calendar

EnterpriseOne Sales & Operations Planning is a collaborative process where key stakeholders often determine it's relative success or failure. The sales and operations planning process also relies heavily on the dissection, management and sharing of information.

The Plan Calendar supports these planning and communication needs and can help your organization promote a common understanding of your business strategy, potential threats, required action items and significant milestones. Use the calendar's graphical interface to track process-related milestones, assign internal and external action items and manage meetings.

The following example shows the Sales and Operations Planning Plan Calendar:



Example Plan Calendar

Planning months may be longer or shorter than a calendar month. The Plan Calendar displays one planning month at a time. For example, if the current planning cycle is a five week period then the Plan Calendar displays the entire five weeks with the appropriate start and end dates.

Icons in the Plan Calendar represent action items, meetings and milestones. From the main Plan Calendar view you can drill down into a day to create, edit, or view items attached to that day.

## Event Summary

The Event Summary page is accessible from the Sales and Operations Planning home page, and displays open meetings, milestones and action items by roles or users. This page allows you to quickly check the status of important events in your Sales and Operations Planning process. You can sort this list by user name or role, and only those actions assigned to the particular role or user displays in the Task Summary list.

## See Also

Managing Planning Periods

# Working with the Plan Calendar

This section discusses how to:

- Add actions to the Plan Calendar.

- Add milestones to the Plan Calendar.
- Add meetings to the Plan Calendar.

## Pages used to Work with the Plan Calendar

Page Name	Navigation	Usage
Event Summary	Home, Event Summary	View open meetings, milestones and action items by roles or users.
Calendar View	Home, Calendar View	View the Plan Calendar.
Day Summary	From the Calendar View page, click on a date.	View plan details for a particular day and add actions, milestones and meetings.
Details	From the Day Summary page, click on an Action, Milestone or Meeting and then click Edit.	View, edit and delete actions, milestones and meetings.
Add Action	Home, Calendar ViewFrom the Select Period drop-down list box, choose the planning period to which you want to add the action. Click the date when the action will occur. Click the Add Action button.	Add an action to the Plan Calendar.
Add Meeting	Home, Calendar ViewFrom the Select Period drop-down list box, choose the planning period to which you want to add the meeting. Click the date when the meeting will occur. Click the Add Meeting button.	Add an meeting to the Plan Calendar.
Add Milestone	Home, Calendar ViewFrom the Select Period drop-down list box, choose the planning period to which you want to add the milestone. Click the date when the milestone will occur. Click the Add Milestone button.	Add an milestone to the Plan Calendar.

## Adding Actions to the Plan Calendar

Access the Add Action page.

Name	Specify the action name.
Due Date	Specify the date when the action is due. You can enter the date manually in the field or use the calendar tool.
Description	Type a meaningful description for the action. This field is optional.
Notes	Specify any supplemental information about the action. This field is optional.
Assigned To	Specify the person responsible for completing this action. This field is optional.
Status	Specify the action progress. Valid values are: <ul style="list-style-type: none"> <li>• Not Started.</li> <li>• In Progress.</li> <li>• Completed.</li> </ul>

## Adding Milestones to the Plan Calendar

Access the Add Milestones page.

Name	Specify the milestone name.
Due Date	Specify the date when the milestone is due. You can enter the date manually in the field or use the calendar tool.
Description	Type a meaningful description for the milestone. This field is optional.
Status	Specify the milestone progress. Valid values are: <ul style="list-style-type: none"> <li>• Not Started.</li> <li>• In Progress.</li> <li>• Completed.</li> </ul>

## Adding Meetings to the Plan Calendar

Access the Add Meetings page.

Name	Specify the meetings name.
Date	Specify the date when the meeting is due. You can enter the date manually in the field or use the calendar tool.

Description	Type a meaningful description for the meeting. This field is optional.
Agenda	Specify any agenda items to be discussed at this meeting. This field is optional.
Notes	Specify any supplemental information about the meeting. This field is optional.





## CHAPTER 6

# Importing Data

This chapter provides a overview of importing and exporting data, and discusses how to import data using the Sales and Operations Planning Connector.

---

## Understanding Importing Data

This section discusses:

- Data import overview.
- Import data requirements.
- Refresh command syntax.

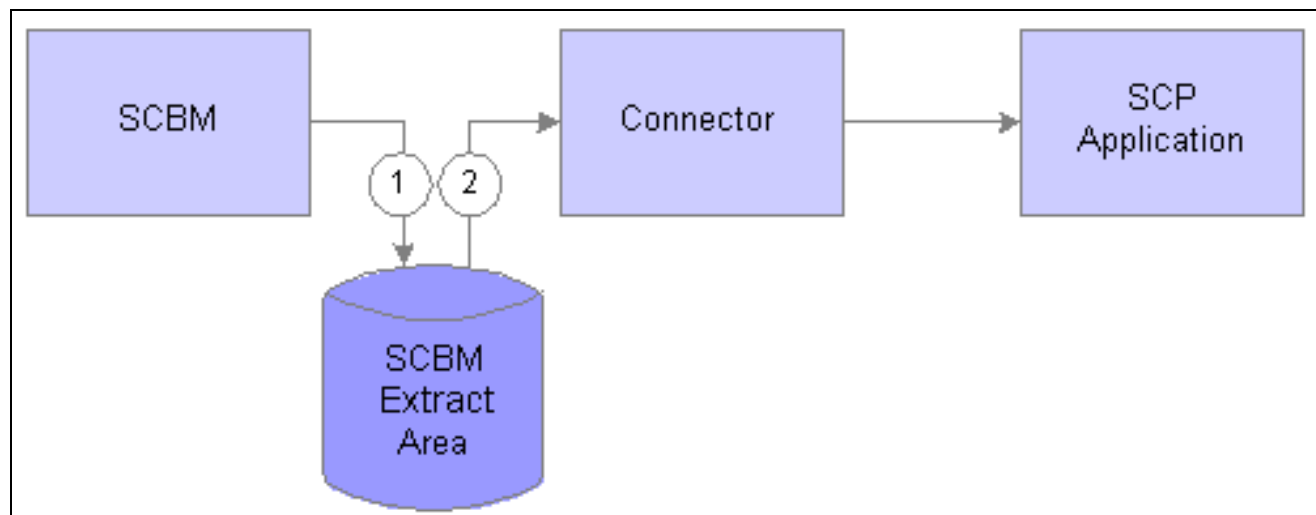
### Data Import Overview

The Sales and Operations Planning connector helps integrate Sales and Operations Planning with your ERP system and other EnterpriseOne Supply Chain Planning products. The Sales and Operations Planning Connector facilitates the data transfer from Supply Chain Business Modeler to Sales and Operations Planning. Planning data from Supply Chain Business Modeler is imported to Sales and Operations Planning, and then used to create and evaluate your plans.

You can use the Sales and Operations Planning Connector to update Sales and Operations Planning with data from Supply Chain Business Modeler. Before using the Refresh command, you must export your planning data from SCBM to an extract area. From there, use the Refresh command to copy Supply Chain Business Modeler data to one or more Supply Chain Planning products.

When refreshing data, you can choose the data granularity level. You can update the entire model or specific packages.

The following diagram illustrates the entire Refresh process:



The Sales and Operations Planning Connector Refresh Process

The Refresh command:

1. Copies Supply Chain Business Modeler model data to the Supply Chain Business Modeler extract area.
2. Transforms the data and imports the data to Sales and Operations Planning.

The Sales and Operations Planning Connector's Refresh command reads XML files. When refreshing data from Supply Chain Business Modeler to Sales and Operations Planning, you must include the following packages to create a model:

- Base
- EnterpriseForecast
- NetProductionRequirements
- BeginningInventory
- DemandPointHistory

## Import Data Requirements

Sales and Operations Planning requires data from Supply Chain Business Modeler's Demand, Strategic, and Tactical models. You should create and execute separate export scenarios for each Supply Chain Business Modeler model. After running the Supply Chain Business Modeler export scenarios, you must rename the files required by Sales and Operations Planning.

Supply Chain Business Modeler XML File	New File Name	Model Data Contained in this File
Demand model Base.xml	DemandBase.xml	Items, locations, channels and their properties, prices.
Demand model EnterpriseForecast.xml	EnterpriseForecast.xml	Demand plan.
Strategic model Base.xml	StrategicBase.xml	Costs.
Tactical model NetProductionRequirements.xml	NetProductionRequirements.xml	Production plan.

Supply Chain Business Modeler XML File	New File Name	Model Data Contained in this File
Tactical model BeginningInventory.xml	BeginningInventory.xml	Beginning inventory.
Tactical model DemandPointHistory.xml	DemandPointHistory.xml	Sales history.

## Config.xml File

Supply Chain Business Modeler supports multiple plans and forecasts. By default, the Sales and Operations Planning Connector imports all properties from the first plan and forecast in the Supply Chain Business Modeler. However, using a Config.xml file you can configure Sales and Operations Planning to import a subset of all properties from a specific plan and forecast.

## Refresh Command Syntax

To run the Sales and Operations Planning connector, use the following syntax:

```
model refresh scbmDataFolder cycle importList importMode
```

This table describes the parameters of the Refresh command:

Parameter	Description
<i>scbmDataFolder</i>	The Supply Chain Business Modeler extract area where XML files are stored. If the file name or path includes blanks, then it must be enclosed in quotation marks.
<i>cycle</i>	The planning cycle to update. If the cycle name includes blanks, then it must be enclosed in quotation marks.
<i>importList</i>	The Supply Chain Business Modeler packages to import to the model. Values are one or more of the following: <ul style="list-style-type: none"> <li>• -Items</li> <li>• -Locations</li> <li>• -Channels</li> <li>• -DemandPlan</li> <li>• -Production</li> <li>• -Price</li> <li>• -Cost</li> <li>• -BeginningInventory</li> <li>• -SalesHistory</li> </ul>
<i>importMode</i>	The update method. Values are: <ul style="list-style-type: none"> <li>• -Add: adds data that does not already exist in the model. Duplicate items are skipped.</li> <li>• -Update: updates data that exists in the model. Items that do not exist are skipped.</li> </ul>

The following example updates the entire Sales and Operations Planning model with data from the Data Extract directory:

```
model refresh "c:/scp/8.12/so-plan/plandata/Data Extract" "March 2005"  
-Items -Locations -Channels -DemandPlan -Production -Prices -Costs  
-BeginningInventory -Update
```

### See Also

Understanding Data for Importing into and Exporting from Supply Chain Business Modeler

---

## Understanding Exporting Data

Sales and Operations Planning does not own any data and therefore does not export data to Supply Chain Business Modeler. Any changes to your planning data must be done in the system of record. However, you can use the Sales and Operations Planning Automation Shell to export data for backup and reporting purposes.

Use the App Backup command to back up the Sales and Operations Planning model to XML files.

### See Also

Understanding the Sales and Operations Planning Automation Shell

App Backup Command

---

## Importing Data Using the Sales and Operations Planning Connector

This section discusses how to import data using the Sales and Operations Planning connector.

## Windows Used to Import Data Using the Sales and Operations Planning Connector

Page Name	Navigation	Usage
Windows	Navigate to the <path>\scp\8.12\so-plan\bin directory and enter the following command: SASH.bat < path/filenameWhere <i>path/filename</i> specifies the batch script that you want to run. path is the directory where the script is saved. filename is the name of the Tcl script. If the path or name of the script includes blanks, you must enclose the path and name in quotation marks (for example, SASH.bat “c:/so-plan/batch script.tcl”).	Access the Sales and Operations Planning connector in Windows.
UNIX	Navigate to the <path>\scp\8.12\so-plan\bin directory and enter the following command: SASH < path/filenameWhere <i>path/filename</i> specifies the batch script that you want to run. path is the directory where the script is saved. filename is the name of the Tcl script. If the path or name of the script includes blanks, you must enclose the path and name in quotation marks (for example, SASH.bat < “c:/so-plan/batch script.tcl”).	Access the Sales and Operations Planning connector in UNIX.

## Importing Data Using the Sales and Operations Planning Connector

Access the Command Prompt.

To import data using the Sales and Operations Planning connector:

1. From the Command Prompt, navigate to the Sales and Operations Planning \bin directory:

```
cd <install path>\scp\8.12\so-plan\bin
```

2. Type the following command:

```
SASH.bat
```

The Sales and Operations Planning Automation Shell starts.

3. Use the Connector's Refresh command:

```
model refresh scbmDataFolder cycle importlist importmode
```

## See Also

Refresh Command Syntax

## CHAPTER 7

# Working with Views

This chapter provides an overview of working with views, and discusses how to:

- Work with views.
- Manage targets for relative metrics.
- Manage targets for absolute metrics.
- Manage thresholds for absolute metrics.

---

## Understanding Working with Views

Use EnterpriseOne Sales and Operations Planning data views to analyze all aspects of your organization's performance, ensure that your plans are realistic and predict future performance. Use the Add View wizard to create any number of views based on one of the available view templates.

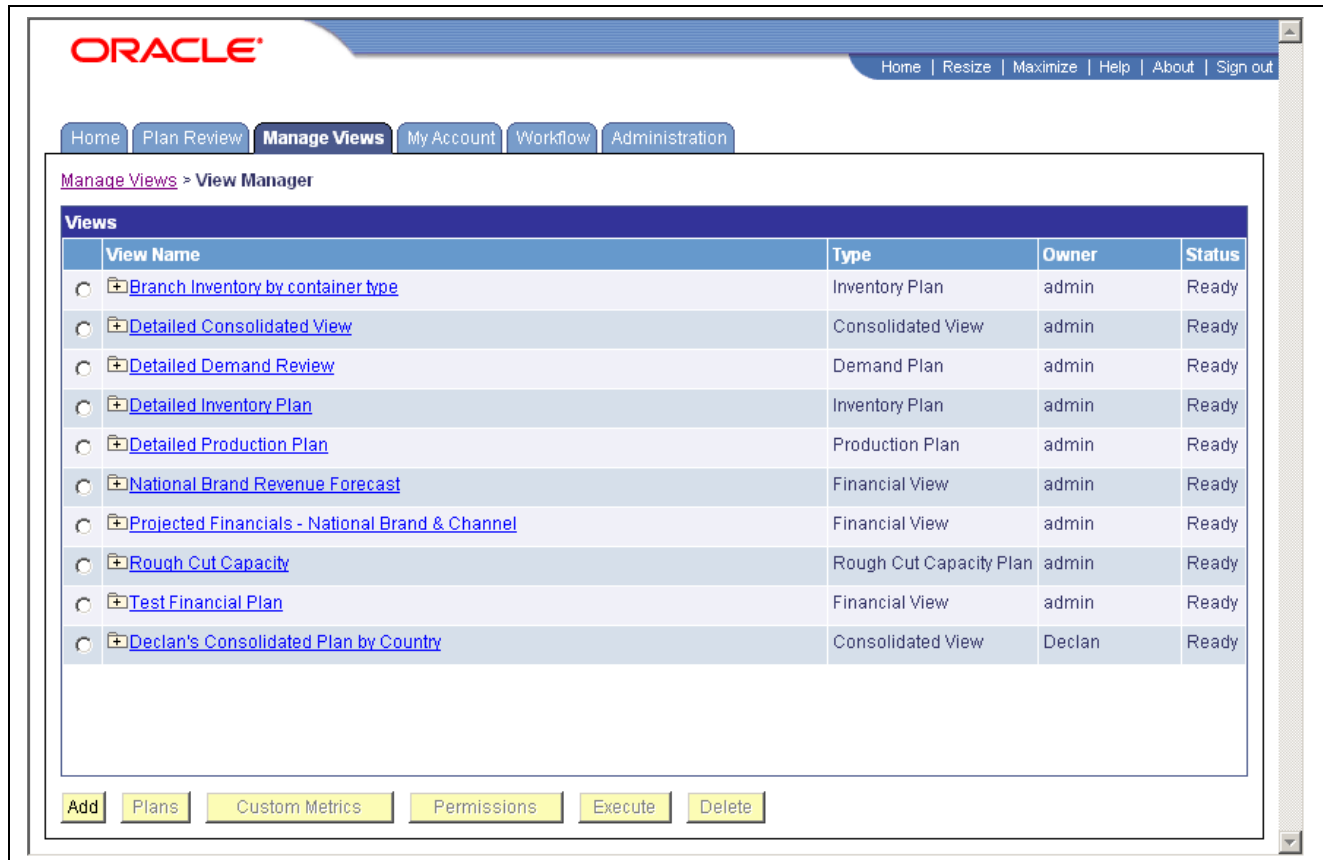
---

**Note.** Only users with Administrator or View Owner accounts can create new Sales and Operations Planning views.

---

The View Manager displays all defined views, along with the view owner and current status. From the View Manager you can add, delete and execute views. Clicking on a view's folder icon displays detailed view information like the specified configuration options, units of measure used, view dimensions and a description.

The following example shows a sample View Manager page:



The View Manager

Administrators can assign permissions

## Targets and Thresholds

One of the most important part of the planning process is setting realistic and achievable targets for an organization's planning metrics. Metrics are used to monitor and measure the performance of your business. In Sales and Operations Planning you can apply target and threshold rules to metrics in each data view. These rules enable you to manage your plans by exception and quickly identify those areas where targets are not being met. Each view must have its metric targets defined for a single planning period. For example, targets may be defined at the monthly or quarterly level, but not both.

Sales and Operations supports two types of targets: relative and absolute. Relative metrics are based on other values. These are comparison metrics and are usually calculated as a percentage or rate. For example, you may have a goal of a business unit having a certain profitability, so you want each unit to have 40 percent profitability. Absolute metrics means actual value. For example, revenue is an absolute metric and represents the total income, in dollars, produced by a given source.

A hierarchical order exists for the rules that are entered in the Manage Targets pages. The first rule on the list that applies to a specific dimension property is used when generating target and threshold metrics. For instance, if you have two margin percentage target rules for Item Code GNS in location Denver, Sales and Operations Planning uses the first defined rule.

When defining rules for relative metrics, you can use an asterisk (\*) to indicate a wildcard. Wildcards indicate that the rule applies to all elements within the dimension property. Wild cards enable you to quickly set targets for many elements at once. For example, a target rule for Item Code HRP in location \* applies to the item HRP at all locations in the model.



Thresholds are a percentage of targets. You can set threshold percentages that represent "good" and "fair" results, which helps you to monitor varying levels of plan performance. Because thresholds are usually similar between product groups, locations, and so on, you can use wild cards to set thresholds quickly.

### See Also

Understanding Views

---

## Working with Views

This section discusses how to:

- Create new views.
- Run existing views.
- Assign plans to views.
- Assign view permissions.

### Pages Used to Work with Views

Page Name	Navigation	Usage
View Manager	Log in to the EnterpriseOne Sales and Operations Planning system, click the Manage Views tab and then click View Manager.	Create, execute and launch views.
View Manager	From the View Manager page, click a view title.	Display a view.
View Manager	Click the radio button adjacent to the view that you want to delete and click Delete.	Delete a view.

### Creating New Views

Access the View Manager page.

Use the Add View wizard to quickly create a new Sales and Operations Planning view. This wizard guides you through the view creation process and allows you to select and configure the view type, and choose what data you want to display.

To create a new view:

1. Click Add.  
The View Name & Type page of the New View wizard appears.
2. In the View Name field, type a descriptive name for the data view.
3. In the View Type drop-down list box, select the view type on which this view is based.

4. Click Next to continue.

The Configuration page of the New View wizard appears.

---

**Note.** The contents of this page depends on the view type that this view is based on. For more information on the metrics available in each data view, see Understanding Data Views.

---

5. Select the metrics that you want to see in the view.

---

**Note.** You must select at least one metric for the view to be valid. If no metrics are selected, Sales and Operations Planning automatically adds all available metrics to the view.

---

6. Click Next to continue.

The Dimensions page of the New View wizard appears.

7. From the Available list, select the dimension that you want to include in the view.
8. Click the right arrow button to add the selected dimension to the view definition.
9. Click Next to continue.

The Description & Review page of the New View wizard appears.

10. You may optionally decide to add a description to your view.

Review the details of the view and then click Finish to save your changes.

## See Also

Understanding Views

## Running Existing Views

Access the View Manager page.

---

**Note.** Views must be run before you can view them.

---

To run a view, select the option for the view that you want to run, and then click Execute.

When you run a view, the View Manager displays the number of items in the view.

## Assigning Plans to Views

Access the View Manager page.

To assign plans to views:

1. In the Select column, select the view to which you want to assign plans.
2. Click Plans.
3. For each planning cycle, enable the plan option that you want to assign to the view.
4. Click Save.

---

**Note.** You must assign at least one plan to a view for it to be valid.

---

## Assigning View Permissions

Access the View Manager page.

To assign view permissions:

1. In the Select column, select the view to which you want to assign permissions.
2. Click Permissions.
3. In the Available Roles/Users field, select the role or user to which you want to grant view permissions.  
To select multiple entries, hold down the CTRL key while selecting users and roles.
4. Click the right arrow button to add the selected role or user to the Selected Roles/Users list.  
To remove a permission, select the permission in the Selected Roles/Users field and then click the left arrow button.
5. Click Save.

### See Also

Administering Users

## Adding Custom Metrics to Views

Access the View Manager page.

To add custom metrics to views:

1. In the Select column, select the view to which you want to add custom metrics.
2. Click Custom Metrics.
3. In the Custom Metrics page, select the custom metrics that you want to add to the view.
4. Click Save.

### See Also

Custom Metrics

---

## Managing Targets for Relative Metrics

This section discusses how to:

- Add new targets for relative metrics.
- Edit targets for relative metrics.
- Change the order of targets for relative metrics.

## Page Used to Manage Targets for Relative Metrics

Page Name	Navigation	Usage
Manage Targets for Relative Metrics	Manage Views, Manage Targets for Relative Metrics	Insert, delete, and edit relative metric target rules.
Manage Targets for Relative Metrics	Select the option for the rule that you want to delete and click Delete.	Delete a relative metric target rule.

## Adding New Targets for Relative Metrics

Access the Manage Targets for Relative Metrics page.

A hierarchical order exists for the rules that are entered in the Manage Targets pages. When generating target metrics, Sales and Operations Planning uses the first rule on the list that applies to a specific dimension property.

To add a new target:

1. In the Selection column, select the option for the target above which you want to add the new rule.

---

**Note.** If this is the first target added to the Manage Targets for Relative Metrics page, ignore this step and proceed to step 2.

---

2. In the Select View drop-down list box, select the data view on which to base the target.  
All data views that use relative metrics appear in this list.
3. In the Select Metric drop-down list box, select the metric for which you want to specify targets.
4. Click Add.
5. In the available dimensions drop-down list, select the elements to which you want to apply the new target.  
To apply the target to all elements within the dimension, select the wild card character (\*).
6. In the Target, Good Threshold, and Fair Threshold fields, specify the target and threshold values that you want to apply to this target.
7. Click Save.

## Editing Targets for Relative Metrics

Access the Manage Targets for Relative Metrics page.

To edit an existing targets:

1. In the Selection column, select the option for the rule that you want to edit, and then click Edit.
2. Modify the target value, and then click Save to save your changes.

## Changing the Order of Targets for Relative Metrics

Access the Manage Targets for Relative Metrics page.

To change the order of targets for relative metrics:

1. In the Select column, select the rule for which you want to change the order.
2. Click Move Up or Move Down to change the target order.

### See Also

Targets and Thresholds

---

## Managing Targets for Absolute Metrics

This section discusses how to:

- Add new targets for absolute metrics.
- Change the planning period group for absolute metrics.

### Page Used to Manage Targets for Absolute Metrics

Page Name	Navigation	Usage
Manage Targets for Absolute Metrics	Manage Views, Manage Targets for Absolute Metrics	Insert, delete and edit absolute metric targets.
Change Planning Period Group	Manage Views, Manage Targets for Absolute Metrics Click the planning period group value in the Unit Demand Targets field.	Change the current planning period group.

### Adding New Targets for Absolute Metrics

Access the Manage Targets for Absolute Metrics page.

To add new targets for absolute metrics:

1. In the Select View drop-down list box, select the data view on which to base the targets.  
All data views that use absolute metrics appear in this list. When you select a data view, the page updates to display all available dimensions for this view.
2. In the Select Metric drop-down list box, select the metric for which you want to specify target.
3. In the Unit drop-down list box, select the unit of measure that you want to use for the target.
4. For each dimension, specify the element to which you want to apply the target.
5. Click Save to save your changes.

### Changing the Planning Period Group for Absolute Metrics

Access the Change Planning Period Group page.

---

**Note.** For each view, metric targets can be defined for a single Planning Period Group only. Changing the Planning Period Group will remove any targets currently associated with this view.

---

To change the planning period group for absolute metrics:

1. In the Planning Period Group drop-down list box, select the planning group level at which you want to set metrics.
2. Click Save to save your changes.

---

## Manage Thresholds for Absolute Metrics

This section discusses how to:

- Insert new threshold rules for absolute metrics.
- Edit threshold rules.
- Change the order of thresholds for absolute metrics.

### Page Used to Manage Thresholds for Absolute Metrics

Page Name	Navigation	Usage
Manage Thresholds for Absolute Metrics	Manage Views, Manage Thresholds for Absolute Metrics	Insert, delete, and edit absolute metric thresholds.
Manage Thresholds for Absolute Metrics	Select the option for the rule that you want to delete and then click Delete.	Delete an absolute metric threshold.

### Inserting New Threshold Rules for Absolute Metrics

Access the Manage Thresholds for Absolute Metrics page.

To insert a new threshold rule:

1. In the Select View drop-down list box, select the data view on which to base the threshold rule.
2. In the Select Metric drop-down list box, select the metric for which you want to specify threshold rule.
3. Click Add.
4. For each dimension, specify the element to which you want to apply the threshold rule.
5. In the Good Threshold and Fair Threshold fields, specify the threshold values that you want to apply to this rule.

Thresholds represent a percentage of targets. Because thresholds are usually similar between product groups, locations, and so on, you can use wild cards to set thresholds quickly.

6. Click Save to save your changes.

## Editing Threshold Rules

Access the Manage Thresholds for Absolute Metrics page.

To edit a threshold rule:

1. Select the option for the rule that you want to edit, and then click Edit.
2. Modify the rule, and then click Save to save your changes.

## Changing the Order of Thresholds for Absolute Metrics

Access the Manage Thresholds for Absolute Metrics page.

To change the order of thresholds for absolute metrics:

1. In the Select column, select the rule for which you want to change the order.
2. Click Move Up or Move Down to change the rule order.

### See Also

Targets and Thresholds





## CHAPTER 8

# Working with Reports

This chapter provides an overview of reports, and discusses how to:

- Manage Net Change reports.
- Manage Metric Exception reports.

---

## Understanding Reports

EnterpriseOne Sales and Operations Planning reports enable you to automatically identify differences and potential issues between plans. Use reports to perform a relative comparison between any two planning cycles, or a fixed comparison of a given cycle against the aggregated baseline budget. After you identify these high-level differences, you can analyze plans in further detail through various Sales and Operations Planning views.

Reports are based on existing data views and are an effective method of managing exceptions and focusing plan review effort. Reports allow you to manage by exception, by only highlighting those areas of concern. Major changes in plan values often indicate that the plan assumptions need to be reviewed and updated.

Sales and Operations Planning provides two different types of reports:

- Net Change reports highlight changes between two planning cycles.
- Metric Exception reports highlight where one or more planning cycle differs from performance targets.

Net Change and Metric Exception reports are based on existing planning data views. This allows helps you maintain focus during the review process and allows stakeholders to generate reports that match their areas of responsibility.

---

**Note.** Before creating reports you must have at least one defined data view. Your defined reports are stored in the model and can be executed whenever required.

---

## Net Change Reports

Net Change reports highlight major changes in plan values from one cycle to another using the metrics available in the specified view. For example, a report based on the Demand Plan view highlights changes in demand forecasts between the compared cycles. When defining Net Change reports, you choose which metrics you want to compare.

The Net Change Report Manager displays a list of existing Net Change reports, the number of alerts within each report, and the last time each report was executed. From the Net Change Report Manager you can add, delete, edit and execute Net Change reports. For example, you may want to edit a net change report to update the planning cycles that it compares. You can view executed reports from the Net Change Report Manager by clicking on the links in the Report Name column.

Net Change reports are created using the Net Change Report wizard. Through this wizard you define the type of plan data that will be examined for changes over time, and select the two planning cycles that you want to compare and the length of the planning horizon.

The following example shows a sample Net Change report:

**ORACLE** Home | Resize | Maximize | Help | About | Sign out

Home | Plan Review | Manage Views | My Account | Workflow | Administration

Home > Net Change Report Manager > Report View

**Net Change Report Results: Demand Net Change**

Associated View: Detailed Demand Review Cycles: APR-05 - Apr Baseline, MAR-05 - Mar Baseline (Weekly) Metric: Unit Demand Unit: Case Periods: 2005-04-04 - 2005-12-26

**Result List** Rows 1-13 of 192

Sort By: Abs. % Difference Direction: Descending

Rank	Channel Code	Item Code	Location	City	% Difference	Abs. % Difference	Difference	Abs. Difference
1	1001	99013	Toronto		546.15	546.15	156.25	156.25
2	1001	99013	San Diego		546.15	546.15	156.25	156.25
3	1001	99013	Halifax		546.15	546.15	156.25	156.25
4	1001	99013	Atlanta		546.15	546.15	156.25	156.25
5	1001	99009	Atlanta		447.37	447.37	177.08	177.08
6	1001	99009	Toronto		447.37	447.37	177.08	177.08
7	1001	99009	San Diego		447.37	447.37	177.08	177.08
8	1001	99009	Halifax		447.37	447.37	177.08	177.08
9	1001	99015	Toronto		361.54	361.54	195.83	195.83
10	1001	99015	Halifax		361.54	361.54	195.83	195.83
11	1001	99015	Atlanta		361.54	361.54	195.83	195.83
12	1001	99015	San Diego		361.54	361.54	195.83	195.83
13	1003	99007	Atlanta		215.79	215.79	104.17	104.17

Example Net Change report

## See Also

Understanding Views

Working with Views

## Metric Exception Reports

Metric Exception reports highlight where planning cycles deviate from their pre-defined targets. Use Metric Exception reports to quickly see which plan areas are cause for concern, and to drill-down to view the exceptions in more detail.

**Note.** Before you can create any Metric Exception Reports, you must have defined your plan targets and thresholds.

The Metric Exception Report Manager displays a list of existing Metric Exception reports, the number of results within each report, and the last time the report was executed. From the Metric Exception Report Manager you can add, delete, edit and execute Metric Exception reports. For example, you may want to edit a net change report to update the planning cycles that it compares. You can view executed reports from the Metric Exception Report Manager by clicking on the link in the Report Name column.

The following example shows a sample Metric Exception report:

The screenshot shows the Oracle Metric Exception Report Manager interface. The top navigation bar includes links for Home, Plan Review, Manage Views, My Account, Workflow, and Administration. The main content area displays the 'Metric Exception Report Results: Margin % Alerts' for the 'Projected Financials - National Brand & Channel' view, covering the period range of APR 05 - DEC 05. The metrics are set to 'Margin %'. A 'Result List' table is shown with 12 rows of data. The table columns are Rank, Plan, Channel, Channel Type, Item, Brand, Metric, Unit, Actual, Target, Deviation, % Deviation, and Status. The status column uses color-coded icons: red 'X' for negative deviations and yellow bars for positive deviations.

Rank	Plan	Channel	Channel Type	Item	Brand	Metric	Unit	Actual	Target	Deviation	% Deviation	Status
1	Apr Baseline	Beer Store		Labatt		Margin %	Each	-361.62	60.00	-421.62	-702.71	X
2	Apr Baseline	Beer Store		Newcastle		Margin %	Each	-324.61	60.00	-384.61	-641.02	X
3	Apr Baseline	Beer Store		Sleeman		Margin %	Each	-419.30	60.00	-479.30	-798.84	X
4	Apr Baseline	Beer Store		Upper Canada		Margin %	Each	-389.89	60.00	-449.89	-749.81	X
5	Apr Baseline	Commercial		Labatt		Margin %	Each	57.79	60.00	-2.21	-3.69	■
6	Apr Baseline	Commercial		Sleeman		Margin %	Each	57.31	60.00	-2.69	-4.49	■
7	Apr Baseline	Commercial		Upper Canada		Margin %	Each	57.46	60.00	-2.54	-4.23	■
8	Apr Baseline	Grocery		Labatt		Margin %	Each	-367.52	60.00	-427.52	-712.53	X
9	Apr Baseline	Grocery		Newcastle		Margin %	Each	-332.89	60.00	-392.89	-654.82	X
10	Apr Baseline	Grocery		Sleeman		Margin %	Each	-409.09	60.00	-469.09	-781.82	X
11	Apr Baseline	Grocery		Upper Canada		Margin %	Each	-410.48	60.00	-470.48	-784.14	X
12	Apr Upside	Beer Store		Labatt		Margin %	Each	-365.28	60.00	-425.28	-708.80	X

Example Metric Exception report

## See Also

Targets and Thresholds

Managing Targets for Relative Metrics

Managing Targets for Absolute Metrics

Creating New Views

## Report Execution

When you run a report, Sales and Operations Planning compares planning data in the chosen cycles. Because data is frequently updated, you should run reports frequently to ensure that they contain current data. You must run a report before you can view it. Reports that have not been run appear gray in the Report Manager. You can view executed reports in the Report Manager by clicking the corresponding links in the Report Name column.

---

**Note.** If executing a report does not generate any alerts then the report name remains grayed.

---

## Managing Net Change Reports

This section discusses how to:

- Create Net Change reports.
- Edit Net Change reports.
- Run Net Change reports.

### Pages Used to Manage Net Change Reports

Page Name	Navigation	Usage
Net Change Report Manager	Home, Net Change Reports	Work with Net Change reports.
Net Change Report Manager	Click the report that you want to view.	View a Net Change report.
Net Change Report Manager	Select the option for the report that you want to delete and then click Delete.	Delete a Net Change report.
Report View	In the Sort By drop-down list box, select the metric on which you want to sort. In the Direction drop-down list box, choose either Ascending or Descending.	Sort a Net Change report.

## Creating Net Change Reports

Access the Net Change Report Manager page.

Use the Sales and Operations Planning Net Change Report wizard to define your report parameters. This wizard guides you through the report generation process, and prompts you to provide the required information.

To create a Net Change report:

1. Click Add.

The Base View Selection page of the Add New Report wizard appears.

2. In the Report Name field, type a descriptive name for the Net Change report.
3. In the View Name drop-down list box, select the view type that is the basis of the report.
4. Click Next to continue.

The Data Selection page appears.

5. In the Plan Metric drop-down list box, select the measurement criteria on which you want to base the report.

Sales and Operations Planning populates the Plan Metric list box with the metrics available in the selected data view. For example, a report based on a Demand view can use revenue or unit-demand as plan metrics.

6. In the Unit drop-down list box, select the unit of measure that you want the report to use.
7. In the Period Group drop-down list box, select the planning period granularity.
8. Click Next to continue.

The Comparison Parameters page appears.

9. In the Select the Baseline Plans section, select the planning cycle and supply plan that you want to use as the baseline for your net change report.
10. In the Select the Period Range section, select the planning period range for the net change report.
11. In the Select the Comparison Plans section, select the planning cycle and supply plan that you want to compare to the baseline.
12. Click Next to continue.

The Review and Completion page appears.

13. Review the report details and click Finish to complete defining the report.

### See Also

Understanding Reports

Units of Measure

## Editing Net Change Reports

Access the Net Change Report Manager page.

To edit an existing report:

1. Select the option for the report that you want to edit, and then click Edit.

The Add New Report wizard appears.

2. Modify the report, and then click Finish to save your changes.

### See Also

Creating Net Change Reports

## Running Net Change Reports

Access the Net Change Report Manager page.

---

**Note.** Reports must be run before you can view them.

---

To run a report, select the option for the report that you want to run, and then click Execute.

When you run a report, the Net Change Report Manager displays the number of report results and the date when the report was last run.

## Managing Metric Exception Reports

This section discusses how to:

- Create Metric Exception reports.
- Edit Metric Exception reports.
- Run Metric Exception reports.

### Pages Used to Manage Metric Exception Reports

Page Name	Navigation	Usage
Metric Exception Report Manager	Home, Metric Exception Reports	Work with Metric Exception reports.
Metric Exception Report Manager	Click the report that you want to view.	View a Metric Exception report.
Metric Exception Report Manager	Click the radio button adjacent to the report that you want to delete and then click Delete.	Delete a Metric Exception report.
Report View	In the Sort By drop-down list box, select the metric on which you want to sort.	Sort a Metric Exception Report.

## Creating Metric Exception Reports

Access the Metric Exception Report Manager page.

To create a Metric Exception report:

1. Click Add.

The Base View Selection page of the Add New Report wizard appears.

2. In the Report Name field, type a descriptive name for your net change report.
3. In the View Name drop-down list box, select the view type that is the basis of the report.
4. Click Next to continue.

The Data Selection page appears. Sales and Operations Planning populates the Data Selection page with the metrics available in the selected data view. For example, a report based on a Demand view can use revenue or unit-demand as plan metrics.

5. Select the measurement criteria on which you want to base the report. You must select at least one metric to include in the report.
6. In the Period Group drop-down list box, choose a planning period granularity.
7. Click Next to continue.

The Period Range Selection page appears.

8. In the Starting Period drop-down list box, choose the report's beginning planning period.

9. In the Ending Period drop-down list box, choose the report's ending planning period.
10. Click Next to continue.

The Review and Completion page appears.

11. Review the report details and click Finish to generate the Metric Exception report.

## Editing Metric Exception Reports

Access the Metric Exception Report Manager page.

To edit an existing report:

1. Select the option for the report that you want to edit, and then click Edit.

The Add New Report wizard appears.

2. Modify the report, and then click Finish to save your changes.

### See Also

Creating Metric Exception Reports

## Running Metric Exception Reports

Access the Metric Exception Report Manager page.

---

**Note.** Metric Exception reports must be run before you can view them.

---

To run a report, select the option for the report that you want to edit, and then click Execute.

When you run a report, the Metric Exception Report Manager displays the number of report results and the date and time when the report was last run.





## CHAPTER 9

# Using the Sales and Operations Planning Automation Shell

This chapter provides overviews of the Sales and Operations Planning Automation Shell and custom workflows, and discusses how to:

- Start the Sales and Operations Planning Automation Shell.
- Work with Sales and Operations Planning workflows.
- Automate business processes using batch scripts.
- Sales and Operations Planning Tcl command reference

---

## Understanding the Sales and Operations Planning Automation Shell

The Sales and Operations Planning Shell (SASH) extends the Tool Command Language (Tcl) shell with Sales and Operations Planning commands. These commands enable you to run Sales and Operations Planning functions from the SASH shell command prompt and to create Tcl scripts that automate your business processes.

### Sales and Operations Planning Automation Shell Syntax

The following syntax rules apply to all Sales and Operations Planning commands and parameters:

- Command and subcommand names are case-sensitive.  
Each command and subcommand begins with an uppercase character, but it can contain both lowercase and uppercase characters, as specified by the command syntax.
- If a directory path, extract area, data folder, scenario, model, or calendar name includes spaces, enclose the name in quotation marks when you use it as a parameter value.  
For example, you must enclose the following path in quotation marks if you use it as a parameter value:  
`c:/SCP Plans/My Data`
- When you specify a date and time as a parameter value, use the following format: yyyy-mm-ddTHH:MM:SS.  
The T separating the date and time must be uppercase. The time must be specified in 24-hour format. For example, to specify March 31, 2005 at 6:00 p.m. as a parameter value, specify: `2005-03-31T18:00:00`
- When you specify a path as a parameter value, use forward slashes (/) as path separators.  
The Sales and Operations Planning automation shell does not recognize single backslashes as path separators. For example, use the following parameter value to specify a directory path:  
`c:/directory1/directory2`

## Understanding Sales and Operations Planning Workflows

Using the Sales and Operations Planning Workflow Manager, you can run customized Tcl scripts from within the Sales and Operations Planning browser interface. This gives you the flexibility to add custom functionality and to automate commonly-performed Sales and Operations Planning tasks.

For example, you may want to create a Tcl procedure to:

- Import plans.
- Trigger an external workflow.
- Extract data for generating reports in an external application.
- Automatically set up and configure a new planning cycle.

View and execute custom workflows from the Custom Workflow Manager, which is accessed by clicking the Workflow tab. All user types can access the Custom Workflow Manager page. Tcl procedures with the adminOnly flag set to true may only be run by Administrators and are hidden to all other users.

**Note.** To view any changes made to procedureList.xml in the Custom Workflow Manager you must stop and re-start the Sales and Operations Planning server.

The following example shows a sample Custom Workflow Manager:

**ORACLE**

Home | Resize | Maximize | Help | About | Sign out

Home | Plan Review | Manage Views | My Account | **Workflow** | Administration

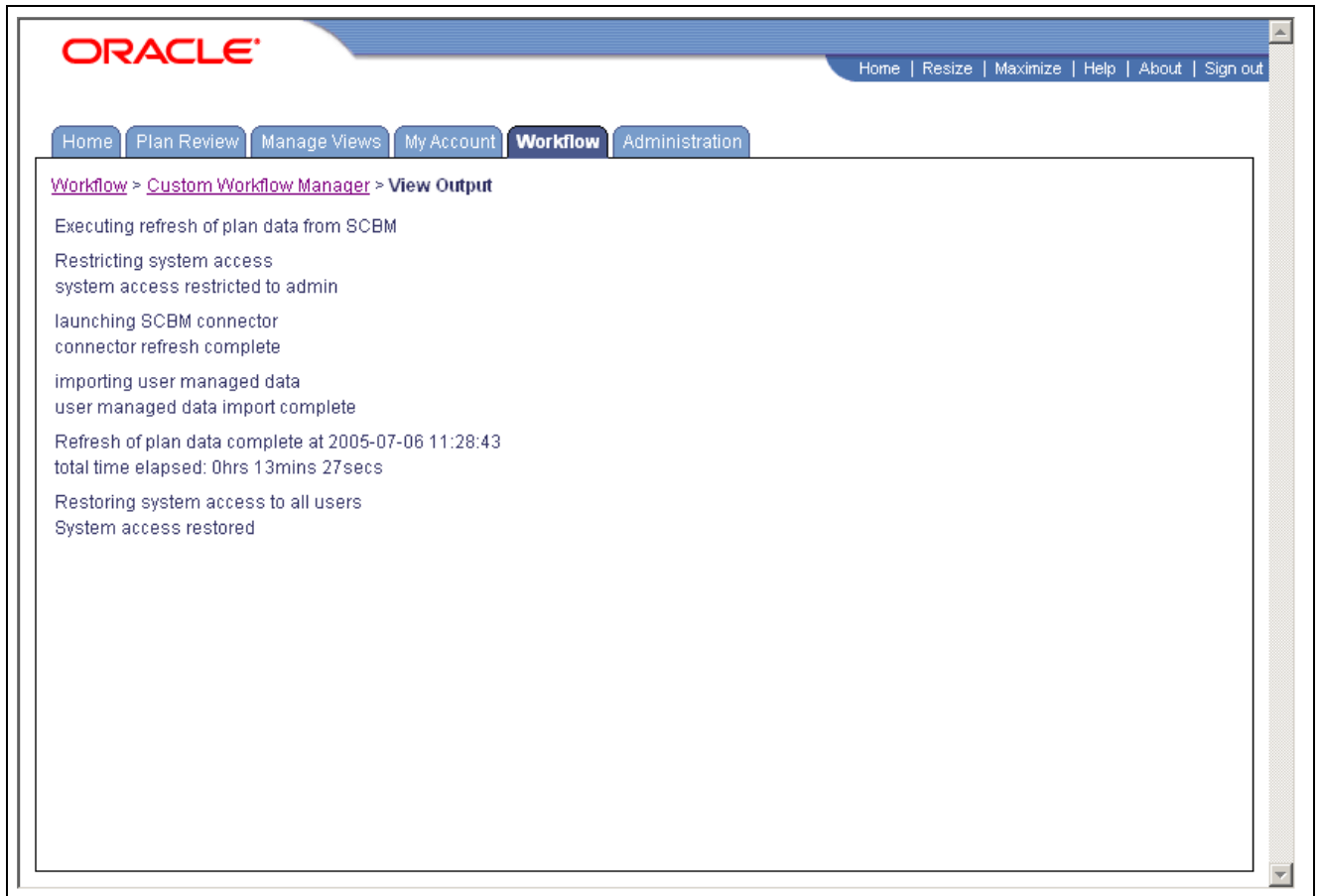
[Workflow](#) > Custom Workflow Manager

Procedure Name	Last Executed On	Status
Run system backup	2006-05-30 11:30:31	Executed
Restore last backup		Executing
Run month end processing	2006-05-30 11:30:42	Failed
Run an on-demand refresh of planning data from SCBM	2006-05-30 11:30:45	Executed
Restrict system access	2006-05-30 11:30:49	Failed
Restore system access	2006-05-30 11:30:51	Failed
Update adjusted revenue forecast		Executing

Execute View Log View Output

Custom Workflow Manager

The following example shows sample output from a workflow:



Sample Output from a Custom Workflow

Custom workflows are stored in the `<install path>\scp\8.12\so-plan\scripts` directory. Workflows are registered with Sales and Operations Planning using the `procedureList.xml` configuration file. This XML file is read by Sales and Operations Planning at startup and contains:

- Tcl procedure name.
- Procedure description.
- TCL file name that executes when running the procedure.
- Administration flag that determines if the procedure can be executed by all users or only administrators.

To add a procedure to the Custom Workflow Manager, you must edit the `procedureList.xml` file. The following example shows a sample `procedureList.xml` file that contains two Tcl procedures.

```

<?xml version="1.0" encoding="utf-8"?>
<procedureList>

  <procedure>
    <name>Import plan data</name>
    <description>Imports new plan data.</description>
    <scriptFileName>import_plan.tcl</scriptFileName>
    <adminOnly>true</adminOnly>
  </procedure>

  <procedure>

```

```
<name>New Planning Cycle</name>
<description>Add a new planning cycle and set it as current.</description>
<scriptFileName>new_cycle.tcl</scriptFileName>
<adminOnly>true</adminOnly>
</procedure>
```

<b>&lt;name&gt;</b>	Name of the Tcl procedure, as it will appear in the Custom Workflow Manager page.
<b>&lt;description&gt;</b>	Description of the Tcl procedure, as it will appear in the Custom Workflow Manager page .
<b>&lt;scriptFileName&gt;</b>	Tcl file name and extension to run when executing the procedure. The Tcl file must be located in the \scp\8.12\so-plan\scripts directory.
<b>&lt;adminOnly&gt;</b>	Administrator flag. Set to True if the Tcl procedure should be only run by administrators; otherwise false.

### See Also

Adding Custom Workflows to Sales and Operations Planning

Working with Sales and Operations Planning Workflows

---

## Starting the Sales and Operations Planning Automation Shell

This section lists the procedures used to start the Sales and Operations Planning Automation Shell.

## Procedures Used to Start the Sales and Operations Planning Automation Shell

Page Name	Navigation	Usage
Windows	Navigate to the <path>\scp\8.12\so-plan\bin directory and enter the following command: SASH.bat < path/filenamewhere path/filename specifies the batch script that you want to run. The path is the directory where the script is saved. The filename is the name of the Tcl script. If the path or name of the script includes blanks, you must enclose the path and name in quotation marks (for example, SASH.bat < c:/so-plan/batch script.tcl).	Run a batch script from the Sales and Operations Planning Automation Shell in Windows.
UNIX	Navigate to the <path>/scp/8.12/so-plan/bin directory and enter the following command: SASH < path/filenamewhere path/filename specifies the batch script that you want to run. The path is the directory where the script is saved. The filename is the name of the Tcl script. If the path or name of the script includes blanks, you must enclose the path and name in quotation marks (for example, SASH.bat < c:/so-plan/batch script.tcl).	Run a batch script from the Sales and Operations Planning Automation Shell in UNIX.

---

## Automating Business Processes Using Batch Scripts

This section provides an overview of batch scripts and discusses how to create batch scripts.

## Understanding Batch Scripts

In addition to entering commands interactively in the Sales and Operations Planning Automation Shell, you can create and run batch scripts of Sales and Operations Planning Automation Shell and other Tcl commands. Batch scripts enable you to link multiple functions and scenarios, and automate business processes. For example, you can create a script that does the following:

- Extracts data from a supply chain management system.
- Imports the data into Supply Chain Business Modeler.
- Refreshes the data in Supply Chain Business Modeler models.
- Exports the data to Sales and Operations Planning.

Because Sales and Operations Planning Automation Shell commands are an extension of the Tcl programming language, you can include both Sales and Operations Planning Automation Shell and other Tcl commands in batch scripts that you run from the Sales and Operations Planning Automation Shell. However, if you want to specify a script as a pre- or post- script in an import or export scenario, the script cannot include Sales and Operations Planning Automation Shell commands and must be saved with the .tcl extension in the extract area's script folder.

You can run scripts from the Sales and Operations Planning Automation Shell prompt, or use scheduling tools and utilities to run batch scripts automatically. For example, you can create a batch script that imports planning data into Sales and Operations Planning, and schedule the script to run monthly.

---

**Note.** The # symbol in Tcl scripts indicates a comment. The Sales and Operations Planning Automation Shell ignores lines that begin with # so that you can annotate scripts.

---

## Windows Used to Run Batch Scripts

Page Name	Navigation	Usage
	<p>Navigate to the <i>path</i>\scp\8.12\so-plan\bin directory and enter the following command:</p> <p>SASH.bat  <code>&lt;path/filename</code>Where <i>path/filename</i> specifies the batch script that you want to run. <i>path</i> is the directory where the script is located. <i>filename</i> is the name of the Tcl script. If the path or name of the script includes blanks, you must enclose the path and name in quotation marks. For example, SASH.bat  <code>&lt; "c:/so-plan/batch script.tcl".</code></p>	Run a batch script from the Sales and Operations Planning Automation Shell in Windows.
UNIX	<p>Navigate to the <i>path</i>\scp\8.12\so-plan\bin directory and enter the following command:</p> <p>SASH  <code>&lt;path/filename</code>Where <i>path/filename</i> specifies the batch script that you want to run. <i>path</i> is the directory where the script is saved. <i>filename</i> is the name of the Tcl script. If the path or name of the script includes blanks, you must enclose the path and name in quotation marks. For example, startSASH.bat  <code>&lt; "c:/so-plan/batch script.tcl"</code></p>	Run a batch script from the Sales and Operations Planning Automation Shell in UNIX.

## Working with Sales and Operations Planning Workflows

This section explains how to add custom workflows to Sales and Operations Planning.

## Page Used to Work with Sales and Operations Planning Workflows

Page Name	Navigation	Usage
Custom Workflow Manager	Home, Workflow, Custom Workflow Manager.	Work with custom Tcl procedures.
Custom Workflow Manager	Home, Workflow, Custom Workflow Manager, Execute.	Run a custom workflow.
Custom Workflow Manager	Home, Workflow, Custom Workflow Manager, View Log.	View any errors or messages generated by the selected Tcl procedure.
Custom Workflow Manager	Home, Workflow, Custom Workflow Manager, View Output.	View the output of the selected Tcl procedure.

## Adding Custom Workflows to Sales and Operations Planning

To add a custom workflow to Sales and Operations Planning:

1. Copy the Tcl procedure file that you want to add to Sales and Operations Planning to the <path>\scp\8.12\so-plan\scripts directory.
2. Open procedureList.xml in a text editor.
3. Add a new <procedure> root element, and <name>, <description>, <scriptFileName> and <adminOnly> child elements.
4. Stop and re-start the Sales and Operations Planning server.

The new Tcl procedure is added to the Custom Workflow Manager page.

---

## Sales and Operations Planning Tcl Command Reference

This section lists:

- Tcl commands by category.
- Sales and Operations Planning Tcl commands.

### Tcl Commands by Category

The following topics list the Sales and Operations Planning Automation Shell commands by category and provide links from within each category to detailed command information.

#### Application Commands

App Archive command

App Backup command



App Clear command  
App DatabaseExists command  
App FixJavaClassPath command  
App GetCompressMode command  
App GetDBDiskFree command  
App GetInstallLocation command  
App GetVersion command  
App LoadPropertyNames command  
App Restore command  
App SetCompressMode command  
App RunBackground command  
App WritePropertyNames command

### **Calendar Commands**

Calendar AddAction command  
Calendar AddMeeting command  
Calendar AddMilestone command  
Calendar AddToCalendarFromXml command  
Calendar Clear command  
Calendar DeleteAction command  
Calendar DeleteMeeting command  
Calendar DeleteMilestone command  
Calendar GetAction command  
Calendar GetMeeting command  
Calendar GetMilestone command  
Calendar GetAllActions command  
Calendar GetAllMeetings command  
Calendar GetAllMilestones command  
Calendar UpdateAction command  
Calendar UpdateMeeting command  
Calendar UpdateMilestone command  
Calendar WriteCalendarToXml command

**Channels Commands**

Channels AddChannel command

Channels AddChannelsFromXml command

Channels CheckExists command

Channels DeleteAllChannels command

Channels DeleteChannel command

Channels GetChannelCodes command

Channels GetChannelProperties command

Channels GetChannelPropertyNames command

Channels SetChannelPropertyNames command

Channels SetChannelProperties command

Channels StrictUpdateChannelsFromXml command

Channels UpdateChannelsFromXml command

Channels WriteChannelsToXml command

**CustomMetrics Commands**

CustomMetrics AddCustomMetric command

CustomMetrics DeleteCustomMetric command

CustomMetrics ExportCustomMetric command

CustomMetrics ExportCustomMetrics command

CustomMetrics GetValues command

CustomMetrics ImportCustomMetrics command

CustomMetrics SetValues command

**CustomWorkflows Commands**

CustomWorkflows ExecuteProcedure command

CustomWorkflows GetLogEvents command

CustomWorkflows GetOutput command

**Cycles Commands**

Cycles AddCycle command

Cycles AddCyclesFromXml command

Cycles AddPlanMappingFromXml command

Cycles Contains command

Cycles DeleteAllCycles command

Cycles DeleteApprovedPlan command  
Cycles DeleteCycle command  
Cycles GetApprovedPlanCyclesInfo command  
Cycles GetCurrentCycleName command  
Cycles SetApprovedPlan command  
Cycles SetCurrentCycle command  
Cycles StrictUpdateCyclesFromXml command  
Cycles UpdateCyclesFromXml command  
Cycles WriteCyclesToXml command  
Cycles WritePlanMappingToXml command

### **DemandPlan Commands**

DemandPlan AddDemand command  
DemandPlan AddDemandFromXml command  
DemandPlan AddDemandFromXmlToCycle command  
DemandPlan AddDemandPlan command  
DemandPlan ContainsId command  
DemandPlan DeleteAllDemands command  
DemandPlan DeleteAllDemandsForCycle command  
DemandPlan DeleteDemand command  
DemandPlan GetDemandIds command  
DemandPlan GetDemandPlanNames command  
DemandPlan GetSupplyPlans command  
DemandPlan StrictUpdateDemandPlanFromXml command  
DemandPlan StrictUpdateDemandPlanFromXmlToCycle command  
DemandPlan UpdateDemand command  
DemandPlan UpdateDemandFromXml command  
DemandPlan UpdateDemandFromXmlToCycle command  
DemandPlan WriteDemandToXml command  
DemandPlan WriteDemandToXmlFromCycle command

### **InventoryHistory Commands**

InventoryHistory AddInventoryHistory command  
InventoryHistory AddInventoryHistoryFromXml command

InventoryHistory DeleteAllInventoryHistories  
InventoryHistory DeleteInventoryHistory  
InventoryHistory GetBeginningInventory  
InventoryHistory WriteInventoryHistoryToXml

## **Items Commands**

Items AddItem command  
Items AddItemChannelPrice command  
Items AddItemChannelPricesFromXml command  
Items AddItemHoldingCost command  
Items AddItemHoldingCostsFromXml command  
Items AddItemProductionCost command  
Items AddItemProductionCostsFromXml command  
Items AddItemPrice command  
Items CheckExists command  
Items DeleteAllItems command  
Items DeleteItem command  
Items GetItemCodes command  
Items GetItemProperties command  
Items GetItemPropertyNameNames command  
Items SetItemProperties command  
Items SetItemPropertyNameNames command  
Items StrictUpdateItemsFromXml command  
Items UpdateItemsFromXml command  
Items WriteItemChannelPricesToXml command  
Items WriteItemHoldingCostsToXml command  
Items WriteItemProductionCostsToXml command  
Items WriteItemsToXml command

## **Locations Commands**

Locations AddLocation command  
Locations AddLocationsFromXml command  
Locations CheckExists command  
Locations DeleteAllLocations command

Locations DeleteLocation command  
Locations GetLocationCodes command  
Locations GetLocationProperties command  
Locations GetLocationPropertyNames command  
Locations SetLocationProperties command  
Locations SetLocationPropertyNames command  
Locations StrictUpdateLocationsFromXml command  
Locations UpdateLocationsFromXml command  
Locations WriteLocationsToXml command

### **Log Commands**

Log ClearDbLog command  
Log ClearDbLogOnAndBeforeDate command  
Log ClearFileLog command  
Log GetDbLogEvents command  
Log GetFileLogEvents command  
Log GetLoggingStatus command  
Log LogEvent command  
Log MultiLogEvent command  
Log ResetLoggingSystem command  
Log StartDbLog command  
Log StartFileLog command  
Log StartLogging command  
Log StopAllLogging command  
Log StopLogging command

### **MetricExceptionReports Commands**

MetricExceptionReports DeleteAllReports command  
MetricExceptionReports DeleteReport command  
MetricExceptionReports ExecuteReport command  
MetricExceptionReports ExportReports command  
MetricExceptionReports GetReportsInfo command  
MetricExceptionReports GetReportsStatus command  
MetricExceptionReports GetReportValues command

MetricExceptionReports ImportReports command

### **NetChangeReports Commands**

NetChangeReports DeleteAllReports command

NetChangeReports DeleteReport command

NetChangeReports ExecuteReport command

NetChangeReports ExportReports command

NetChangeReports GetReportsInfo command

NetChangeReports GetReportsStatus command

NetChangeReports GetReportValues command

NetChangeReports ImportReports command

### **Periods Commands**

Periods AddPeriod command

Periods DeleteAllPeriods command

Periods DeletePeriod command

Periods ExportPeriods command

Periods ImportPeriods command

### **PlanMapping Commands**

PlanMapping AddPlanMappingFromXml command

PlanMapping WritePlanMappingToXml command

### **ProductionHistory Commands**

ProductionHistory AddProductionHistory command

ProductionHistory AddProductionHistoryFromXml command

ProductionHistory DeleteAllProductionHistories command

ProductionHistory DeleteProductionHistory command

ProductionHistory WriteProductionHistoryToXml command

### **ProductionPlan Commands**

ProductionPlan AddProduction command

ProductionPlan AddProductionFromXml command

ProductionPlan AddProductionFromXmlToCycle command

ProductionPlan ContainsId command

ProductionPlan DeleteAllProductions command

ProductionPlan DeleteAllProductionsForCycle command

ProductionPlan DeleteProduction command  
ProductionPlan GetProductionIds command  
ProductionPlan StrictUpdateProductionFromXml command  
ProductionPlan StrictUpdateProductionFromXmlToCycle command  
ProductionPlan UpdateProduction command  
ProductionPlan UpdateProductionFromXml command  
ProductionPlan UpdateProductionFromXmlToCycle command  
ProductionPlan WriteProductionToXml command  
ProductionPlan WriteProductionToXmlFromCycle command

### **Resources Commands**

Resources AddPlannedCapacity command  
Resources AddPlannedCapacityFromXml command  
Resources AddPlannedCapacityFromXmlToCycle command  
Resources AddRequiredCapacity command  
Resources AddRequiredCapacityFromXml command  
Resources AddRequiredCapacityFromXmlToCycle command  
Resources AddResource command  
Resources AddResourceMaximumCapacitiesFromXml command  
Resources AddResourceMaximumCapacity command  
Resources AddResourcesFromXml command  
Resources CheckExists command  
Resources DeleteAllPlannedCapacity command  
Resources DeleteAllPlannedCapacityForCycle command  
Resources DeleteAllRequiredCapacities command  
Resources DeleteAllRequiredCapacitiesForCycle command  
Resources DeleteAllResources command  
Resources DeletePlannedCapacity command  
Resources DeleteRequiredCapacity command  
Resources DeleteResource command  
Resources GetPlannedCapacityValues command  
Resources GetResourceCodes command  
Resources GetResourceProperties command

Resources GetResourcePropertyNames command  
Resources PlannedCapacityContainsId command  
Resources RequiredCapacityContainsId command  
Resources SetResourceProperties command  
Resources SetResourcePropertyNames command  
Resources StrictUpdatePlannedCapacityFromXml command  
Resources StrictUpdatePlannedCapacityFromXmlToCycle command  
Resources StrictUpdateRequiredCapacityFromXml command  
Resources StrictUpdateRequiredCapacityFromXmlToCycle command  
Resources StrictUpdateResourcesFromXml command  
Resources UpdatePlannedCapacity command  
Resources UpdatePlannedCapacityFromXml command  
Resources UpdatePlannedCapacityFromXmlToCycle command  
Resources UpdateRequiredCapacity command  
Resources UpdateRequiredCapacityFromXml command  
Resources UpdateRequiredCapacityFromXmlToCycle command  
Resources UpdateResourcesFromXml command  
Resources WritePlannedCapacityToXml command  
Resources WritePlannedCapacityToXmlFromCycle command  
Resources WriteRequiredCapacityToXml command  
Resources WriteRequiredCapacityToXmlFromCycle command  
Resources WriteResourceMaximumCapacitiesToXml command  
Resources WriteResourcesToXml command

### **SalesHistory Commands**

SalesHistory AddSalesHistoryFromXml command  
SalesHistory AddSalesHistoryPrices command  
SalesHistory Add SalesHistoryUnits command  
SalesHistory DeleteAllSalesHistories command  
SalesHistory DeleteSalesHistoryPrices command  
SalesHistory DeleteSalesHistoryUnits command  
SalesHistory UpdateSalesHistoryFromXml command  
SalesHistory WriteSalesHistoryToXml command



**SupplyPlan Commands**

SupplyPlan AddSupplyPlan command

SupplyPlan DeleteSupplyPlan command

SupplyPlan GetSupplyPlanNames command

**UnitsOfMeasure Commands**

UnitsOfMeasure AddItemUomFromXml command

UnitsOfMeasure AddUom command

UnitsOfMeasure Clear command

UnitsOfMeasure DeleteUom command

UnitsOfMeasure GetItemUom command

UnitsOfMeasure GetUomList command

UnitsOfMeasure SetItemUom command

UnitsOfMeasure WriteItemUomToXml command

**Users Commands**

Users AddUser command

Users AddUsersFromXml command

Users DeleteUser command

Users IsAdmin command

Users IsViewAdmin command

Users SetPassword command

Users StrictUpdateUsersFromXml command

Users UpdateUsersFromXml command

Users WriteUsersToXml command

**Views Commands**

Views AddConsolidatedView command

Views AddDemandPlanView command

Views AddFinancialView command

Views AddInventoryView command

Views AddProductionPlanView command

Views AddRoughCutCapacityView command

Views DeleteAllViews command

Views DeleteView command

Views ExecuteAllViews  
 Views ExecuteView command  
 Views ExportRelativeTargets command  
 Views ExportTargets command  
 Views ExportThresholds command  
 Views ExportViews command  
 Views GetCyclePlans command  
 Views GetKeys command  
 Views GetMetricValues command  
 Views GetViewsStatus command  
 Views ImportRelativeTargets command  
 Views ImportTargets command  
 Views ImportThresholds command  
 Views ImportViews command  
 Views InvalidateAllViews command  
 Views SetPlansForViewCycle command

---

## Sales and Operations Planning Tcl Commands

The following are Sales and Operations Planning Tcl commands and subcommands:

### App Archive Command

#### Syntax

*App\_Archive directoryName*

#### Description

Use the App Archive command to archive all model data to the specified directory.

#### Parameters

Parameter	Description
<i>directoryName</i>	Name and path of the directory where you want to archive model data. If the path includes blanks, then it must be enclosed in quotation marks.

#### Example

The following example archives the Sales and Operations Planning model to the `c:\scp\modelbackup` directory:

```
App_Archive c:/scp/modelbackup
```

## App Backup Command

### Syntax

```
App_Backup path
```

### Description

Use the Backup command to back up the Sales and Operations Planning model to XML files.

### Parameters

Parameter	Description
<i>path</i>	The destination directory where the backup XML files are written. If the path includes blanks, then it must be enclosed in quotation marks.

### Example

The following example backs up your database to a series of XML files in the directory `/scp/8.12/so-plan/plandata`:

```
App_Backup /scp/8.12/so-plan/plandata
```

### See Also

Understanding Exporting Data

## App Clear Command

### Syntax

```
App_Clear deleteUsers
```

### Description

Use the App Clear command to clear the model and remove all users.

### Parameters

Parameter	Description
<i>deleteUsers</i>	Boolean indicating whether or not user information should be removed from the model. Valid values are: <ul style="list-style-type: none"><li>• 1 — user information is removed from the model.</li><li>• 0 — user information is kept in the model.</li></ul>

### Example

The following example clears the model and users:

```
App_Clear 1
```

## App DatabaseExists Command

### Syntax

```
App_DatabaseExists databaseName
```

### Description

Use the DatabaseExists command to determine if the specified database is present.

### Parameters

Parameter	Description
<i>databaseName</i>	The database name. If the database name contains blanks, then it must be enclosed in quotation marks.

### Returns

True, if the specified database exists; otherwise, False.

### Example

The following example searches for the database SOPplandb:

```
App_DatabaseExists SOPplandb
```

## App FixJavaClasspath Command

### Syntax

```
App_FixJavaClasspath classpath
```

### Description

Use the FixJavaClasspath command to convert “;” separators to “:” so that the classpath is readable on UNIX platforms.

### Parameters

Parameter	Description
<i>classpath</i>	The path where Java is installed on the Sales and Operations Planning server.

### Example

The following example fixes the classpath so that it is readable on UNIX platforms:

```
App_FixJavaClasspath $ROOT/bin;$JAVA_HOME/bin;$JAVA_HOME/bin/classic
```

## App GetCompressMode command

### Syntax

```
App_GetCompressMode
```

**Description**

Use the GetCompressMode command to retrieve the current XML file compress mode.

**Returns**

This command returns:

- 1 — XML compress mode is enabled.
- 2 — XML compress mode is disabled.

**Example**

The following example retrieves the current XML compress mode:

```
App_GetCompressMode
```

**See Also**

App SetCompressMode command

## App GetDBDiskFree Command

**Syntax**

```
App_GetDBDiskFree
```

**Description**

Use the GetDBDiskFree command to return the available free space in the current database as a percentage of the total database size.

**Returns**

The amount of available database space as a percentage of the total database size.

## App GetInstallLocation Command

**Syntax**

```
App_GetInstallLocation
```

**Description**

Use the GetInstallLocation command to return the Sales and Operations Planning install path.

**Returns**

A string containing the install path for Sales and Operations Planning.

**Example**

The following example returns the current Sales and Operations Planning install location:

```
App_GetInstallLocation
```

## App GetVersion Command

### Syntax

```
App_GetVersion
```

### Description

Use the GetVersion command to display the current Sales and Operations Planning version.

### Returns

A string containing the application version.

### Example

The following example returns the current Sales and Operations Planning version:

```
App_GetVersion
```

## App LoadPropertyNames Command

### Syntax

```
App_LoadPropertyNames filename
```

### Description

Use the LoadPropertyNames command to load model property names from an XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing the model properties. If the file name includes blanks, then it must be enclosed in quotation marks.

### Example

This example loads the property names from the filePropertyNames.xml.

```
App_LoadPropertyNames c:/scp/8.12/so-plan/plandata/PropertyNames.xml
```

## App Restore Command

### Syntax

```
App_Restore path restoreUsers
```

### Description

Use the Restore command to update the contents of the database using the specified XML files.

## Parameters

Parameter	Description
<i>path</i>	Location of the XML files used to update the database. If the path includes blanks, then it must be enclosed in quotation marks.
<i>restoreUsers</i>	Determines whether or not the Users database should be updated. Possible values are: 1– The update overwrites the contents of the Users database. 0– The update does not overwrite the contents of the Users database.

## Example

The following example updates the database using XML files found at `c:/scp/8.12/so-plan/plandata` and overwrites the Users database.

```
App_Restore c:/scp/8.12/so-plan/plandata 1
```

## App RunBackground Command

### Syntax

```
App_RunBackground commandName
```

### Description

Use the RunBackground command to start an executable from a the Tcl shell without opening a command prompt window.

### Parameters

Parameter	Description
<i>commandName</i>	Name of the command to be run as a background process.

## Example

The following command launches the Sales and Operations Planning Application Server as a background process:

```
App_RunBackground c:/scp/8.12/so-plan/bin/startAppServer.bat
```

## App SetCompressMode Command

### Syntax

```
App_SetCompressMode compress
```

### Description

Use the SetCompressMode command to configure Sales and Operations Planning to compress all output XML files using gzip. Compressed XML files use the `.xml.gz` extension.

## Parameters

Parameter	Description
<i>compress</i>	Boolean value indicating whether or not output XML files are compressed. Values are: <ul style="list-style-type: none"> <li>• True — output XML files are compressed as they are written.</li> <li>• False — output XML files are not compressed.</li> </ul>

## Example

The following example configures Sales and Operations Planning to compress XML files:

```
App_SetCompressMode 1
```

## App WritePropertyNamees Command

### Syntax

```
App_WritePropertyNamees filename
```

### Description

Use the WritePropertyNamees command to write model properties to an XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where model properties are written. If the path includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes model properties to the file PropertyNames.xml.

```
App_WritePropertyNamees /scp/8.12/so-plan/plandata/PropertyNames.xml
```

## Calendar AddAction Command

### Syntax

```
Calendar_AddAction dueDate name description notes assignedTo status
```

### Description

Use the AddAction command to add a new action to the planning calendar.

When adding actions to the planning calendar, they should have a unique date/name combination.



## Parameters

Parameter	Description
<i>dueDate</i>	Date when the action is due, in the format YYYY-MM-DD.
<i>name</i>	Action name, as it appears in the planning calendar. If the action name includes blanks, then it must be enclosed in quotation marks. This parameter is required.
<i>description</i>	Action description. If the description includes blanks, then it must be enclosed in quotation marks. This parameter is optional; use empty quotation marks (" ") to create an action with no description.
<i>notes</i>	Comments or additional details that you want to assign to the action. If the note include blanks, then it must be enclosed in quotation marks.
<i>assignedTo</i>	Sales and Operations Planning user to which the action is assigned. This parameter is required; you must assign actions to a defined Sales and Operations Planning user.
<i>status</i>	Current action item progress. Valid values are: <ul style="list-style-type: none"> <li>• Completed.</li> <li>• Not Started.</li> <li>• In Progress.</li> </ul> This parameter is required.

## Example

The following example adds an action Approve S&OP Plan to the planning calendar:

```
Calendar_AddAction 2006-09-13 "Approve S&OP Plan" "Select the approved plan option
set for the current planning cycle and publish approval." "" Bradley "Not Started"
```

## Calendar AddMeeting Command

### Syntax

```
Calendar_AddMeeting date name description agenda notes status
```

### Description

Use the AddMeeting command to add a new meeting to the planning calendar.

When adding a new meeting to the planning calendar, it must have a unique date/name combination. Otherwise, the first meeting on that date is overwritten.

## Parameters

Parameter	Description
<i>date</i>	Meeting date, in the format YYYY-MM-DD.
<i>name</i>	Meeting name, as it appears in the planning calendar. If the meeting name includes blanks, then it must be enclosed in quotation marks.
<i>description</i>	A detailed description for the new meeting. If the description includes blanks, then it must be enclosed in quotation marks. This parameter is optional; use empty quotation marks (“”) to create a meeting with no description.
<i>agenda</i>	A list of agenda items for the created meeting. If the agenda includes blanks, then it must be enclosed in quotation marks. This parameter is optional; use empty quotation marks (“”) to create a meeting with no agenda.
<i>notes</i>	A list of optional notes for the created meeting. If the notes includes blanks, then it must be enclosed in quotation marks. This parameter is optional; use empty quotation marks (“”) to create a meeting with no notes.
<i>status</i>	Current meeting progress. Valid values are: <ul style="list-style-type: none"> <li>• Completed.</li> <li>• Not Started.</li> <li>• In Progress.</li> </ul> This parameter is required.

## Example

The following example adds a meeting Senior Management Plan Approval to the planning calendar:

```
Calendar_AddMeeting 2006-09-13 "Senior Management Plan Approval"
"Meeting to discuss the current sales and operations plan with management"
"" "" "Not Started"
```

## Calendar AddMilestone Command

### Syntax

```
Calendar_AddMilestone dueDate name
description status
```

### Description

Use the AddMilestone command to add a new project milestone to the planning calendar.

When adding milestones to the planning calendar, they must have a unique date/name combination. Otherwise, the first milestone on that date is overwritten.

## Parameters

Parameter	Description
<i>dueDate</i>	Date when the milestone is due, in the format YYYY-MM-DD.
<i>name</i>	Milestone name as it appears in the planning calendar. If the milestone name includes blanks, then it must be enclosed in quotation marks.
<i>description</i>	Milestone description. If the description includes blanks, then it must be enclosed in quotation marks.
<i>status</i>	Current milestone progress. Valid values are: <ul style="list-style-type: none"> <li>• Completed.</li> <li>• Not Started.</li> <li>• In Progress.</li> </ul>

## Example

The following example adds a milestone Demand Plan Review to the planning calendar:

```
Calendar_AddMilestone 2005-15-06 "Demand Plan Review" "Quarterly demand plan
review meeting" "Not Started"
```

## Calendar AddToCalendarFromXml Command

### Syntax

```
Calendar_AddToCalendarFromXML filename
```

### Description

Use the AddToCalendarFromXml command to add new calendar actions, milestones and meetings to the model using an input XML file. Calendar items that are already present in the model are skipped.

### Parameters

Parameter	Description
<i>filename</i>	File name and path of the input XML file. If the path includes blanks, then it must be enclosed in quotation marks.

## Example

The following example updates the Sales and Operations Planning calendar with the contents of the file Calendar\_Update.xml:

```
Calendar_AddToCalendarFromXml c:/scp/8.12/so-plan/plandata/Calendar_Update.xml
```

## Calendar Clear Command

### Syntax

```
Calendar_Clear
```

## Description

Use the Clear command to remove all actions, milestones and meetings from the planning calendar.

## Example

The following example removes all events from the planning calendar:

```
Calendar_Clear
```

## Calendar DeleteAction Command

### Syntax

```
Calendar_DeleteAction dueDate name
```

### Description

Use the DeleteAction command to remove the specified action from the planning calendar.

### Parameters

Parameter	Description
<i>dueDate</i>	Date when the action that you want to delete is due, in the format YYYY-MM-DD.
<i>name</i>	Action name that you want to delete, as it appears in the planning calendar. If the action name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example deletes the action Approve S&OP Plan from the planning calendar:

```
Calendar_DeleteAction 2006-09-13 "Approve S&OP Plan"
```

## Calendar DeleteMeeting Command

### Syntax

```
Calendar_DeleteMeeting date name
```

### Description

Use the DeleteMeeting command to remove the specified meeting from the planning calendar.

### Parameters

Parameter	Description
<i>date</i>	Date of the meeting that you want to delete from the planning calendar, in the format YYYY-MM-DD.
<i>name</i>	Name of the meeting that you want to delete, as it appears in the planning calendar. If the meeting name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example removes a meeting Senior Management Plan Approval from the planning calendar:

```
Calendar_AddMeeting 2006-09-13 "Senior Management Plan Approval"
```

## Calendar DeleteMilestone Command

### Syntax

```
Calendar_DeleteMilestone dueDate name
```

### Description

Use the DeleteMilestone command to remove a project milestone from the planning calendar.

### Parameters

Parameter	Description
<i>dueDate</i>	Date when the milestone that you want to delete is due, in the format YYYY-MM-DD.
<i>name</i>	Name of the milestone that you want to delete, as it appears in the planning calendar. If the milestone name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example removes a milestone Demand Plan Review from the planning calendar:

```
Calendar_DeleteMilestone 2006-09-13 "Demand Plan Review"
```

## Calendar GetAction Command

### Syntax

```
Calendar_GetAction dueDate name
```

### Description

Use the GetAction command to retrieve from the planning calendar details for the specified action.

### Parameters

Parameter	Description
<i>dueDate</i>	Date when the action for which you want to retrieve is due, in the format YYYY-MM-DD.
<i>name</i>	Name of the action that you want to retrieve from the planning calendar.

## Example

The following example retrieves details for the action Approve S&OP Plan:

```
Calendar_GetAction 2006-09-13 "Approve S&OP Plan"
```

## Calendar GetMeeting Command

### Syntax

```
Calendar_GetMeeting date name
```

### Description

Use the GetMeeting command to retrieve from the planning calendar details for the specified meeting.

### Parameters

Parameter	Description
<i>date</i>	Meeting date for which you want to retrieve details, in the format YYYY-MM-DD.
<i>name</i>	Name of the meeting for which you want to retrieve details, as it appears in the planning calendar. If the meeting name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example retrieves details for the meeting Senior Management Plan Approval:

```
Calendar_GetMeeting 2006-09-13 "Senior Management Plan Approval"
```

## Calendar GetMilestone Command

### Syntax

```
Calendar_GetMilestone dueDate name
```

### Description

Use the GetMilestone command to retrieve from the planning calendar details for the specified milestone.

### Parameters

Parameter	Description
<i>dueDate</i>	Date when the milestone for which you want to retrieve is due, in the format YYYY-MM-DD.
<i>name</i>	Name of the milestone for which you want to retrieve details, as it appears in the planning calendar. If the milestone name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example retrieves details for the milestone Demand Plan Review:

```
Calendar_GetMilestone 2006-09-13 "Demand Plan Review"
```

## Calendar GetAllActions Command

### Syntax

```
Calendar_GetAllActions
```

**Description**

Use the GetAllActions command to generate a list all actions in the planning calendar, sorted by date and name.

**Returns**

A paired list of actions and dates in the format “YYYY-MM-DD:<action name>”.

**Example**

The following example returns a list of all plan calendar actions:

```
Calendar_GetAllActions
```

## Calendar GetAllMeetings Command

**Syntax**

```
Calendar_GetAllMeetings
```

**Description**

Use the GetAllMeetings command to generate a list of all meetings in the planning calendar, sorted by date and name.

**Returns**

A paired list of meetings and dates in the format “YYYY-MM-DD:<meeting name>”.

**Example**

The following example returns a list of all plan calendar meetings:

```
Calendar_GetAllMeetings
```

## Calendar GetAllMilestones Command

**Syntax**

```
Calendar_GetAllMilestones
```

**Description**

Use the GetAllMilestones command to generate a list of all milestones in the planning calendar, sorted by date and name.

**Returns**

A paired list of milestones and dates in the format “YYYY-MM-DD:<milestone name>”.

**Example**

The following example returns a list of all plan calendar milestones:

```
Calendar_GetAllMilestones
```

## Calendar UpdateAction Command

### Syntax

```
Calendar_UpdateAction oldDueDate oldName
dueDate name description notes
assignedTo status
```

### Description

Use the UpdateAction command to move or edit an action in the plan calendar.

### Parameters

Parameter	Description
<i>oldDueDate</i>	Original due date of the action to be updated, in the format YYYY-MM-DD.
<i>oldName</i>	Name of the action to be updated. If the action name includes blanks, then it must be enclosed in quotation marks.
<i>dueDate</i>	New due date for the updated action, in the format YYYY-MM-DD.
<i>name</i>	New action name. If the action name includes blanks, then it must be enclosed in quotation marks.
<i>description</i>	New action description. If the description includes blanks, then it must be enclosed in quotation marks.
<i>notes</i>	Any supplemental information that you want to include with this action. If the note includes blanks, then it must be enclosed in quotation marks.
<i>assignedTo</i>	Person to which the action is assigned.
<i>status</i>	Current action progress. Valid values are: <ul style="list-style-type: none"> <li>• Completed.</li> <li>• Not Started.</li> <li>• In Progress.</li> </ul>

### Example

The following example reschedules an action Review Demand Plan to September 13, 2005:

```
Calendar_UpdateAction 2005-09-06 " Review Demand Plan" 2005-09-13
"Review Demand Plan" "Review quarterly demand plan in advance of the senior
management meeting." "Re-scheduled from September 6th" "Nicholas McCully"
"Not Started"
```

## Calendar UpdateMeeting Command

### Syntax

```
Calendar_UpdateMeeting oldDate oldName
newDate name description agenda
notes
```



## Description

Use the UpdateMeeting command to move or edit a meeting in the plan calendar.

## Parameters

Parameter	Description
<i>oldDate</i>	Date of the meeting to be updated, in the format YYYY-MM-DD.
<i>oldName</i>	Name of the meeting to be updated. If the meeting name includes blanks, then it must be enclosed in quotation marks.
<i>newDate</i>	New due date for the updated meeting, in the format YYYY-MM-DD.
<i>name</i>	New meeting name. If the meeting name includes blanks, then it must be enclosed in quotation marks.
<i>description</i>	New meeting description. If the description includes blanks, then it must be enclosed in quotation marks.
<i>agenda</i>	New meeting agenda. If the agenda includes blanks, then it must be enclosed in quotation marks.
<i>notes</i>	Any supplemental information that you want to include with this meeting. If the note includes blanks, then it must be enclosed in quotation marks.

## Example

The following example reschedules a meeting Demand Plan Review Meeting to September 21, 2005:

```
Calendar_UpdateMeeting 2005-09-13 "Demand Plan Review Meeting" 2005-09-21
"Demand Plan Review Meeting" "Review demand plan with senior management."
"9 am - 12 pm" "Re-scheduled from September 13th"
```

## Calendar UpdateMilestone Command

### Syntax

```
Calendar_UpdateMilestone oldDueDate oldName
dueDate name description status
```

### Description

Use the UpdateMilestone command to move or edit a milestone in the plan calendar.

## Parameters

Parameter	Description
<i>oldDueDate</i>	Original due date of the milestone to be updated, in the format YYYY-MM-DD.
<i>oldName</i>	Name of the milestone to be updated. If the action name includes blanks, then it must be enclosed in quotation marks.
<i>dueDate</i>	New due date for the updated milestone, in the format YYYY-MM-DD.
<i>name</i>	New milestone name. If the milestone name includes blanks, then it must be enclosed in quotation marks.
<i>description</i>	New milestone description. If the description includes blanks, then it must be enclosed in quotation marks.

## Example

The following example reschedules a milestone Sales Forecasts Due from August 23rd, 2005 to September 1st, 2005:

```
Calendar_UpdateMilestone 2005-08-23 "Sales Forecasts Due" 2005-09-01 "Sales
Forecasts Due" ""
```

## Calendar WriteCalendarToXml Command

### Syntax

```
Calendar_WriteCalendarToXml filename
```

### Description

Use the WriteCalendarToXml command to write the contents of the plan calendar to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	File name and path to which the plan calendar contents are written. If the path includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes the contents of the plan calendar to the file Calendar.xml:

```
Calendar_WriteCalendarToXml "c:/plan_calendar.xml"
```

## Channels AddChannel Command

### Syntax

```
Channels_AddChannel channelCode channelName
```

### Description

Use the AddChannel command to add a new channel to the model.

## Parameters

Parameter	Description
<i>channelCode</i>	Code for the channel that you want to add to the model.
<i>channelName</i>	Name of the channel that you want to add to the model. If the channel name includes blanks, then it must be enclosed in quotation marks.

## Example

In the following example, the channel Liquor Control Board of Ontario is added to the model.

```
Channels_Add Channel LCBO "Liquor Control Board of Ontario"
```

## Channels AddChannelsFromXml Command

### Syntax

```
Channels_AddChannelsFromXML filename errors
skippedChannelCodes
```

### Description

Use the AddChannelsFromXml command to add new channels to the model using an input XML file. Channels that are already present in the model are skipped.

## Parameters

Parameter	Description
<i>filename</i>	File name and path of the input XML file. If the path includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered when adding channels.
<i>skippedChannelCodes</i>	List object containing codes of channels that were not added to the model.

## Example

The following example adds channel data to the model from the specified XML file:

```
# prepare list objects for receiving errors and skipped channels
set errorsPtr [ListObject]
set skippedChannelsPtr [ListObject]
# Add the Channel data
Channels_AddChannelsFromXml c:/scp/8.12/so-plan/plandata/ChannelList.xml
$errorsPtr $skippedChannelsPtr
# Return any errors or skipped channels
ListObject_Get $errorsPtr
ListObject_Get $skippedChannelsPtr
```

## Channels CheckExists Command

### Syntax

```
Channels_CheckExists channelName
```

### Description

Use the CheckExists command to verify that the specified channel exists in the model.

### Parameters

Parameter	Description
<i>channelName</i>	Name of the channel that you want to verify exists in the database.

### Returns

An error, if the specified channel name does not exist.

### Example

The following example searches the model for the Liquor Control Board of Ontario channel, which has the code LCBO:

```
Channels_CheckExists LCBO
```

## Channels DeleteAllChannels Command

### Syntax

```
Channels_DeleteAllChannels
```

### Description

Use the DeleteAllChannels command to delete all channels from the model.

### Example

The following example deletes all channels from the Sales and Operations Planning model:

```
Channels_DeleteAllChannels
```

## Channels DeleteChannel Command

### Syntax

```
Channels_DeleteChannel channelCode
```

### Description

Use the DeleteChannel command to delete the specified channel from the model.

## Parameters

Parameter	Description
<i>channelCode</i>	Code for the channel that you want to delete from the model.

## Example

The following example deletes the channel LCBO from the model:

```
Channels_DeleteChannel LCBO
```

## Channels GetChannelCodes Command

### Syntax

```
Channels_GetChannelCodes
```

### Description

Use the GetChannelCodes command to retrieve the codes for all of the channels in the model.

### Returns

A list of all channels in the model.

## Example

The following example returns a list of all channels that are defined in the Sales and Operations Planning model:

```
Channels_GetChannelCodes
```

## Channels GetChannelProperties Command

### Syntax

```
Channels_GetChannelProperties channelCode
```

### Description

Use the GetChannelProperties command to retrieve channel properties using the specified channel code.

## Parameters

Parameter	Description
<i>channelCode</i>	Code for the channel with properties that you want to retrieve.

## Returns

A paired list of properties and their associated values.

## Example

The following example retrieves properties for the Liquor Control Board of Ontario channel:

```
Channels_GetChannelProperties LCBO
```

This Tcl command returns:

```
{Company {Liquor Control Board of Ontario}} {{Phone Nr.} {1-800-ONT-LCBO }}
{Web site http://www.lcbo.com}
```

## Channels GetChannelPropertyNames Command

### Syntax

```
Channels_GetChannelPropertyNames
```

### Description

Use the GetChannelPropertyNames command to retrieve a list of your model's channel property attributes.

### Returns

A list your model's channel property attributes.

### Example

The following example retrieves all defined channel properties:

```
Channels_GetChannelPropertyNames
```

## Channels SetChannelPropertyNames Command

### Syntax

```
Channels_SetChannelPropertyNames channelCode
propertyNames
```

### Description

Use the SetChannelPropertyNames command to set the names for the specified channel.

### Parameters

Parameter	Description
<i>channelCode</i>	Code for the channel with properties that you want to set.
<i>propertyNames</i>	Vector containing a list of property names. Channels that exist in the model will keep property values for any property names that were already present in the model.

### Example

The following example sets the name for the channel code LCBO:

```
Channels_SetChannelPropertyNames LCBO "Liquor Control Board of Ontario"
```

## Channels SetChannelProperties Command

### Syntax

```
Channels_SetChannelProperties channelCode
propertyList
```

## Description

Use the SetChannelProperties command to set property values for the channel with the specified code.

## Parameters

Parameter	Description
<i>channelCode</i>	Code for the channel with properties that you want to set.
<i>propertyList</i>	Vector containing paired properties that you want to set for the specified channel.

## Example

The following example sets properties for the channel c1:

```
# Set Channel properties
Channels_SetChannelPropertyNames {AccountNo Name}
Channels_AddChannel c1 ""
Channels_SetChannelProperties c1 {{Name CostMart}}
```

# Channels StrictUpdateChannelsFromXml Command

## Syntax

```
Channels_StrictUpdateChannelsFromXml filename
invalidChannelCodes
```

## Description

Use the StrictUpdateChannelsFromXml command to do a strict update of your model channel data using the specified XML file. If a channel in the XML file does not exist in the model then that entry is ignored and its code is added to the list of invalid channel codes.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing channel data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidChannelCodes</i>	List object containing channels that were not added to the model. Only channels that already exist in your model are updated.

## Example

The following example performs a strict update of the model's channel data using the specified XML file:

```
# prepare list objects for receiving errors and invalid channel codes
set errorsPtr [ListObject]
set invalidChannelsPtr [ListObject]
# Add the Channel data
```

```
Channels_StrictUpdateChannelsFromXML c:/scp/8.12/so-plan/plandata/
ChannelList.xml $errorsPtr $invalidChannelsPtr
# Return any errors or skipped items
ListObject_Get $errorsPtr
ListObject_Get $invalidChannelsPtr
```

## Channels UpdateChannelsFromXml Command

### Syntax

```
Channels_UpdateChannelsFromXml filename errors
addedChannelCodes
```

### Description

Use the UpdateChannelsFromXml command to update model channel data using the specified XML file. This command updates channels that are already present in the model, and creates channels that are not present in the model.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing channel data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedChannelCodes</i>	List object containing the channels that were added to the model.

### Example

The following example updates model channel data from the specified XML file:

```
# prepare list objects for receiving errors and added channel codes
set errorsPtr [ListObject]
set addedChannelsPtr [ListObject]
# Add channel data
Channels_UpdateChannelsFromXML c:/scp/8.12/so-plan/plandata/ChannelList.xml
$errorsPtr $addedChannelsPtr
# Return any errors or added items
ListObject_Get $errorsPtr
ListObject_Get $addedChannelsPtr
```

## Channels WriteChannelsToXml Command

### Syntax

```
Channels_WriteChannelsToXml filename
```

### Description

Use the WriteChannelsToXml command to write all model channels to the specified XML file.



## Parameters

Parameter	Description
<i>filename</i>	The XML file name and path where model channel data is written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes all model channels to the file ChannelList.xml.

```
Channels_WriteChannelsToXml c:/scp/8.12/so-plan/plandata/ChannelList.xml
```

## CustomMetrics AddCustomMetric Command

### Syntax

```
CustomMetrics_AddCustomMetric name type  
planningPeriodGroup dimensions
```

### Description

Use the AddCustomMetric command to add a user-defined metric to the Sales and Operations Planning model.

### Parameters

Parameter	Description
<i>name</i>	Name of the custom metric that you want to add to the model.
<i>type</i>	Metric type. Valid values are: <ul style="list-style-type: none"> <li>• Global.</li> <li>• Cycle.</li> <li>• Demand Plan.</li> <li>• Supply Plan.</li> </ul>
<i>planningPeriodGroup</i>	Granularity of the planning period. Valid values are: <ul style="list-style-type: none"> <li>• Weekly.</li> <li>• Monthly.</li> <li>• Quarterly.</li> <li>• Yearly.</li> </ul>
<i>dimensions</i>	Vector containing the dimension and level at which this metric is stored. Valid values are: <ul style="list-style-type: none"> <li>• Item.</li> <li>• Location.</li> <li>• Channel.</li> <li>• Resource.</li> </ul>

## Example

The following example adds a global custom metric Brewed Quantity at the monthly level, using the Item and Location.City dimensions:

```
CustomMetrics_AddCustomMetric Global Monthly {{Item, }}{Location, City}}
```

## See Also

Custom Metrics

Adding Custom Metrics to Views

## CustomMetrics DeleteCustomMetric Command

### Syntax

```
CustomMetrics_DeleteCustomMetric customMetricName
```

### Description

Use the DeleteCustomMetric command to delete the specified custom metric from the Sales and Operations Planning model.

### Parameters

Parameter	Description
<i>customMetricName</i>	Name of the custom metric that you want to delete from the model. If the metric name contains blanks, then it must be enclosed in quotation marks.

## Example

The following example deletes the custom metric Brewed Quantity:

```
CustomMetrics_DeleteCustomMetric "Brewed Quantity"
```

## See Also

Custom Metrics

## CustomMetrics ExportCustomMetric Command

### Syntax

```
CustomMetrics_ExportCustomMetric fileName customMetricName
```

### Description

Use the ExportCustomMetric command to write an existing custom metric to the specified XML file.

## Parameters

Parameter	Description
<i>fileName</i>	The XML file name and path where the specified custom metric is written. If the file name includes blanks, then it must be enclosed in quotation marks.  <b>Note.</b> If the specified XML file already exists, the ExportCustomMetric command will overwrite it.
<i>customMetricName</i>	The custom metric that you want to export.

## Example

The following example exports the custom metric Brewed Quantity to the file BrewedQuantityMetric.xml:

```
CustomMetrics_ExportCustomMetric c:/scp/8.12/so-plan/plandata/BrewedQuantityMetric.xml
"Brewed Quantity"
```

## CustomMetrics ExportCustomMetrics Command

### Syntax

```
CustomMetrics_ExportCustomMetrics fileName
```

### Description

Use the ExportCustomMetrics command to write all defined custom metrics to the specified XML file.

### Parameters

Parameter	Description
<i>fileName</i>	The XML file name and path where the specified custom metric is written. If the file name includes blanks, then it must be enclosed in quotation marks.  <b>Note.</b> If the specified XML file already exists, the ExportCustomMetric command will overwrite it.

## Example

The following example exports all custom metrics to the file Metrics.xml:

```
CustomMetrics_ExportCustomMetrics c:/scp/8.12/so-plan/plandata/Metrics.xml
```

## CustomMetrics GetValues Command

### Syntax

```
CustomMetrics_GetValues name cycleName
demandPlan supplyPlan key
```

### Description

Use the GetValues command to retrieve the values of a custom metric.

**Note.** The number of required parameters depends on the metric type. For global metrics, only the name and key parameters are required. For cycle-specific metrics, the name, cycleName and key parameters are required. For demand and supply plan metrics, provide the name, cycle name, demandPlan/supplyPlan and key parameters.

Use an empty string as placeholders for unneeded parameters.

## Parameters

Parameter	Description
<i>name</i>	Name of the custom metric for which you want to retrieve values. If the custom metric name includes blanks, then it must be enclosed in quotation marks.
<i>cycleName</i>	Cycle for which you want to retrieve custom metrics. For global metrics, use an empty string (" ") for this parameter.
<i>demandPlan</i>	Demand plan for which you want to retrieve custom metrics values. If the demand plan name includes blanks, then it must be enclosed in quotation marks. For global or cycle-specific metrics, use an empty string (" ") for this parameter.
<i>supplyPlan</i>	Supply plan for which you want to retrieve custom metrics values. If the supply plan name includes blanks, then it must be enclosed in quotation marks. For global or cycle-specific metrics, use an empty string (" ") for this parameter.
<i>key</i>	Vector containing item and location values.

## Returns

A vector of custom metrics values.

## Example

The following example retrieves the value for the global metric 95% Forecast:

```
CustomMetrics_GetValues "95% Forecast" " " " " " {SportMart Toronto}
```

## See Also

Custom Metrics

# CustomMetrics ImportCustomMetrics Command

## Syntax

```
CustomMetrics_ImportCustomMetrics filename errors
```

## Description

Use the ImportCustomMetrics command to add all custom metrics from the specified XML file. Any custom metric that already exists in the model is changed.

## Parameters

Parameter	Description
<i>filename</i>	The XML file name and path from which you want to import custom metrics. If the file name includes blanks, then it must be enclosed in quotation marks.

Parameter	Description
<i>errors</i>	List object containing any errors encountered during the custom metric import.

## Example

The following example imports custom metrics from the file CustomMetrics.xml:

```
CustomMetrics_ImportCustomMetrics c:/scp/8.12/so-plan/plandata/CustomMetrics.xml
```

## CustomMetrics SetValues Command

### Syntax

```
CustomMetrics_SetValues name cycleName demandPlan supplyPlan key
```

### Description

Use the SetValues command to set the values of a custom metric.

**Note.** The number of required parameters depends on the metric type. For global metrics, only the name and key parameters are required. For cycle-specific metrics, the name, cycleName and key parameters are required. For demand and supply plan metrics, provide the name, cycle name, demandPlan/supplyPlan and key parameters.

Use an empty string as placeholders for unneeded parameters.

### Parameters

Parameter	Description
<i>name</i>	Name of the custom metric for which you want to set values. If the custom metric name includes blanks, then it must be enclosed in quotation marks.
<i>cycleName</i>	Cycle for which you want to set custom metrics. For global metrics, use an empty string (" ") for this parameter.
<i>demandPlan</i>	Demand plan for which you want to set custom metrics values. If the demand plan name includes blanks, then it must be enclosed in quotation marks. For global or cycle-specific metrics, use an empty string (" ") for this parameter.
<i>supplyPlan</i>	Supply plan for which you want to set custom metrics values. If the supply plan name includes blanks, then it must be enclosed in quotation marks. For global or cycle-specific metrics, use an empty string (" ") for this parameter.
<i>key</i>	Custom metric key, consisting of item and location values.

## Example

The following example sets the value for the global metric 95% Forecast:

```
CustomMetrics_SetValues "95% Forecast" "" "" "" Location.City
```

## See Also

Custom Metrics

# CustomWorkflows ExecuteProcedure Command

## Syntax

```
CustomWorkflows_ExecuteProcedure procedureName
```

## Description

Use the ExecuteProcedure command to launch a Sales and Operations Planning custom workflow.

## Parameters

Parameter	Description
<i>procedureName</i>	Path and file name of the procedure to be executed. If the file name contains blanks it must be enclosed in quotation marks.  <b>Note.</b> To run a custom workflow from SASH, it must be added to the procedureList.xml file.

## Example

The following example executes the workflow SimpleProcedure.tcl:

```
CustomWorkflows_ExecuteProcedure c:/scp/8.12/so-plan/scripts/SimpleProcedure.tcl
```

## See Also

Adding Custom Workflows to Sales and Operations Planning  
Working with Sales and Operations Planning Workflows

# CustomWorkflows GetLogEvents Command

## Syntax

```
CustomWorkflows_GetLogEvents procedureName
```

## Description

Use the GetLogEvents command to retrieve all log events for the specified Tcl command.

---

**Note.** You must execute a procedure before you can retrieve log events.

---

## Parameters

Parameter	Description
<i>procedureName</i>	Path and file name of the procedure for which you want to retrieve log events. If the file name contains blanks it must be enclosed in quotation marks.

## Example

The following example retrieves log events from the workflow SimpleProcedure.tcl:

```
CustomWorkflows_GetLogEvents c:/scp/8.12/so-plan/scripts/SimpleProcedure.tcl
```

## See Also

Working with Sales and Operations Planning Workflows

# CustomWorkflows GetOutput Command

## Syntax

```
CustomWorkflows_GetOutput procedureName
```

## Description

Use the GetOutput command to retrieve the output from the specified Tcl command.

---

**Note.** You must execute a procedure before you can view its output.

---

## Parameters

Parameter	Description
<i>procedureName</i>	Path and file name of the procedure for which you want to retrieve log events. If the file name contains blanks it must be enclosed in quotation marks.

## Example

The following example retrieves the output from the workflow SimpleProcedure.tcl:

```
CustomWorkflows_GetOutput c:/scp/8.12/so-plan/scripts/SimpleProcedure.tcl
```

## See Also

Working with Sales and Operations Planning Workflows

# Cycles AddCycle Command

## Syntax

```
Cycles_AddCycle name startDate
```

## Description

Use the AddCycle command to add a planning cycle to the model.

## Parameters

Parameter	Description
<i>name</i>	Name of the planning cycle to be added. If the cycle name contains blanks it must be enclosed in quotation marks.
<i>startDate</i>	Cycle start date, in the format YYYY-MM-DD.

## Example

The following example adds a cycle Q4 2006 with a start date of January 1st, 2006 to the model:

```
Cycles_AddCycle "Q4 2006" 2006-01-01
```

## See Also

Managing Planning Cycles

## Cycles AddCyclesFromXml Command

### Syntax

```
Cycles_AddCyclesFromXml filename errors skippedCycleCodes
```

### Description

Use the AddCyclesFromXml command to add new planning cycles to the model using an input XML file. Cycles items that are already present in the model are skipped.

## Parameters

Parameter	Description
<i>filename</i>	File name and path of the input XML file. If the path includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered when adding cycles.
<i>skippedCycleCodes</i>	List object containing codes of planning cycles that were not added to the model.

## Example

The following example adds planning cycles to the model from the specified XML file:

```
# prepare list objects for receiving errors and skipped cycles
set errorsPtr [ListObject]
set skippedCyclesPtr [ListObject]
# Add the planning cycles
Cycles_AddCyclesFromXml c:/scp/8.12/so-plan/plandata/Cycles.xml
$errorsPtr $skippedCyclesPtr
# Return any errors or skipped cycles
ListObject_Get $errorsPtr
ListObject_Get $skippedCyclesPtr
```



## Cycles AddPlanMappingFromXml Command

### Syntax

```
Cycles_AddPlanMappingFromXml filename errors
```

### Description

Use the AddPlanMappingFromXml command to add new plan mappings to the model using an input XML file. Plan mappings allow you to assign which Demand and Supply plans are approved for each planning cycle, and can be viewed from the Manage Planning Cycles page.

Plan mappings that are already present in the model are skipped.

### Parameters

Parameter	Description
<i>filename</i>	File name and path of the input XML file. If the path includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered when adding plan mappings.

### Example

The following example adds plan mappings to the model from the specified XML file:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Add the plan mappings
Cycles_AddPlanMappingsFromXml c:/scp/8.12/so-plan/plandata/Planmappings.xml
$errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## Cycles Contains Command

### Syntax

```
Cycles_Contains cycleName
```

### Description

Use the Contains command to determine if the specified cycle exists in the model.

### Parameters

Parameter	Description
<i>cycleName</i>	Name of the planning cycle that you want to verify in the model. If the cycle name contains blanks it must be enclosed in quotation marks.

### Returns

True, if the cycle exists; otherwise, False.

**Example**

The following example checks the model for the planning cycle Q4 2005:

```
Cycles_Contains "Q4 2005"
```

**Cycles DeleteAllCycles Command****Syntax**

```
Cycles_DeleteAllCycles
```

**Description**

Use the DeleteAllCycles command to remove all planning cycles from the Sales and Operations Planning model.

**Example**

The following example removes all planning cycles from the model:

```
Cycles_DeleteAllCycles
```

**Cycles DeleteApprovedPlan Command****Syntax**

```
Cycles_DeleteApprovedPlan cycleName
```

**Description**

Use the DeleteApprovedPlan command to remove approved plans from the specified planning cycle.

**Parameters**

Parameter	Description
<i>cycleName</i>	Name of the planning cycle from which you want to remove approved plans. If the cycle name contains blanks, then it must be enclosed in quotation marks.

**Example**

The following example deletes approved plans from the planning cycle Q4 2006:

```
Cycles_DeleteApprovedPlan "Q4 2006"
```

**Cycles DeleteCycle Command****Syntax**

```
Cycles_DeleteCycle name
```

**Description**

Use the DeleteCycle command to remove the specified planning cycle from the Sales and Operations Planning model.

## Parameters

Parameter	Description
<i>name</i>	Planning cycle that you want to delete from the model. If the cycle name contains blanks, it must be enclosed in quotation marks.

## Example

The following example removes the planning cycle Q4 2006 from the model:

```
Cycles_DeleteCycle "Q4 2006"
```

## Cycles GetApprovedPlanCyclesInfo Command

### Syntax

```
Cycles_GetApprovedPlanCyclesInfo cycleName
```

### Description

Use the GetApprovedPlanCyclesInfo command to get information on the approved plans for the specified cycle.

### Parameters

Parameter	Description
<i>cycleName</i>	Name of the planning cycle for which you want to retrieve approved plan information. If the cycle name contains blanks it must be enclosed in quotation marks.

### Returns

A list containing:

- Cycle name.
- Cycle start date.
- Approved demand plan name.
- Name of the user who approved the demand plan.
- Date and time when the demand plan was approved.
- Approved production plan name.
- Name of the user who approved the production plan.
- Date and time when the demand plan was approved.

## Example

The following example retrieves approved plan information for the planning cycle Q4 2005:

```
Cycles_GetApprovedPlanCyclesInfo "Q4 2005"
```

## Cycles GetCurrentCycleName Command

### Syntax

```
Cycles_GetCurrentCycleName
```

### Description

Use the GetCurrentCycleName command to retrieve the name of the active cycle.

### Example

The following example retrieves the name of the active cycle:

```
Cycles_GetCurrentCycleName
```

## Cycles SetApprovedPlan Command

### Syntax

```
Cycles_SetApprovedPlan cycleName ProductionName userName approvalTime
```

### Description

Use the SetApprovedPlan command to set the approved plans for the specified cycle.

### Parameters

Parameter	Description
<i>cycleName</i>	Name of the planning cycle for which you want to set the approved plans. If the cycle name contains blanks it must be enclosed in quotation marks.
<i>ProductionName</i>	Name of the production plan that you want to approve. If the production plan contains blanks, then it must be enclosed in quotation marks.
<i>userName</i>	Name of the user that you want to associate with the plan approval. If the user name contains blanks, then it must be enclosed in quotation marks.
<i>approvalTime</i>	Approval timestamp, in the format HH:MM:SS.

### Example

The following example approves the production plan Q4 Approved in the cycle Q4 2005, and assigns the plan approver to Nicholas James:

```
Cycles_SetApprovedPlan "Q4 2005" "Q4 Approved" "Nicholas James" 17:00:00
```

## Cycles SetCurrentCycle Command

### Syntax

```
Cycles_SetCurrentCycle cycleName
```

### Description

Use the SetCurrentCycle command to set the current planning cycle.

## Parameters

Parameter	Description
<i>cycleName</i>	Planning cycle that you want to set as current. If the cycle name contains blanks it must be enclosed in quotation marks.

## Example

The following example sets the cycle Q4 2005 as the current planning cycle:

```
Cycles_SetCurrentCycle "Q4 2005"
```

## Cycles StrictUpdateCyclesFromXml Command

### Syntax

```
Cycles_StrictUpdateCyclesFromXml filename errors invalidCycleCodes
```

### Description

Use the StrictUpdateCyclesFromXml command to do a strict update of your model planning cycle data using the specified XML file. If a cycle in the XML file does not exist in the model then that entry is ignored, and its added to the list of invalid planning cycle codes.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing planning cycles used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidCycleCodes</i>	List object containing planning cycles that were not added to the model. Only cycles that already exist in your model are updated.

## Example

The following example performs a strict update of the model's planning cycle data using the specified XML file:

```
# prepare list objects for receiving errors and invalid cycle codes
set errorsPtr [ListObject]
set invalidCyclesPtr [ListObject]
# Add planning cycles
Cycles_StrictUpdateCyclesFromXml c:/scp/8.12/so-plan/plandata/
Cycles.xml $errorsPtr $invalidCyclesPtr
# Return any errors or skipped planning cycles
ListObject_Get $errorsPtr
ListObject_Get $invalidCyclesPtr
```

## Cycles UpdateCyclesFromXml Command

### Syntax

```
Cycles_UpdateCyclesFromXml filename errors addedCycleCodes
```

### Description

Use the UpdateCyclesFromXml command to update planning cycles using data from an XML file. This command updates planning cycles that exist in the model, and adds those cycles that are not present.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing planning cycle data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedCycleCodes</i>	List object containing planning cycles that were added to the model.

### Example

The following example updates the model using planning cycle data from the specified XML file:

```
# prepare list objects for receiving errors and added cycle codes
set errorsPtr [ListObject]
set addedCycleCodesPtr [ListObject]
# Update planning cycles
Cycles_UpdateCyclesFromXml c:/scp/8.12/so-plan/plandata/
Cycles.xml $errorsPtr $addedCycleCodesPtr
# Return any errors or added cycles
ListObject_Get $errorsPtr
ListObject_Get $addedCycleCodesPtr
```

## Cycles WriteCyclesToXml Command

### Syntax

```
Cycles_WriteCyclesToXml filename
```

### Description

Use the WriteCyclesToXml command to write all defined planning cycle information to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where cycle data is written. If the path includes blanks, then it must be enclosed in quotation marks.

### Example

The following example writes planning cycle information to the file PlanningCycles.xml:

```
Cycles_WriteCyclesToXml c:/planningdata/PlanningCycles.xml
```

## Cycles WritePlanMappingToXml Command

### Syntax

```
Cycles_WritePlanMappingToXml filename
```

### Description

Use the WritePlanMappingToXml command to write the current plan mappings to the specified XML file. Plan mappings allow you to assign which Demand and Supply plans are approved for each planning cycle, and can be viewed from the Manage Planning Cycles page.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where plan mappings are written. If the path includes blanks, then it must be enclosed in quotation marks.

### Example

The following example writes the model plan mappings to the file Mappings.xml:

```
Cycles_WritePlanMappingToXml c:/scp/8.12/so-plan/plandata/Mappings.xml
```

## DemandPlan AddDemand Command

### Syntax

```
DemandPlan_AddDemand itemCode locationCode channelCode periodStartDate values
```

### Description

Use the AddDemandcommand to add a demand plan time series to the model.

### Parameters

Parameter	Description
<i>itemCode</i>	Item code for the demand plan entry being added to the model.
<i>locationCode</i>	Location code for the demand plan entry being added to the model.
<i>channelCode</i>	Channel code for the demand plan entry being added to the model.
<i>periodStartDate</i>	Date that specifies the starting period of the values parameter, in the format YYYY-MM-DD.
<i>values</i>	Vector containing demand plan values. The demand plan is updated with these values, starting at the period start date.

### Example

The following example updates the demand plan for Brown Ale\_341ml at the San Diego distribution center for the Hotels and Restaurants channel, starting with the planning period 2004-09-13:

```
DemandPlan_AddDemand 99011 "San Diego" 1001 2004-09-13 {40 65 20 40}
```

This command adds four planning periods.

### See Also

Demand Plan View

Working with Views

## DemandPlan AddDemandFromXml Command

### Syntax

```
DemandPlan_AddDemandFromXml filename errors skippedDemandPlanCodes
```

### Description

Use the AddDemandFromXml command to add a demand plan to the current cycle from the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing demand plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the demand plan update.
<i>skippedDemandPlanCodes</i>	List object containing codes for those demand plans that were not added to the model.

### Example

The following example adds a demand plan to the model from the specified XML file:

```
# prepare list objects for receiving errors and skipped demand plan codes
set errorsPtr [ListObject]
set skippedDemandPlansPtr [ListListObject]
# Add demand plan data
DemandPlan_AddDemandFromXml c:/scp/8.12/so-plan/plandata/DemandPlan.xml
$errorsPtr $skippedDemandPlansPtr
# Return any errors or skipped demand plans
ListObject_Get $errorsPtr
ListListObject_Get $skippedDemandPlansPtr
```

### See Also

Demand Plan View

Working with Views

## DemandPlan AddDemandFromXmlToCycle Command

### Syntax

```
DemandPlan_AddDemandFromXmlToCycle filename errors skippedDemandPlanCodes cycleName
```



## Description

Use the `AddDemandFromXmlToCycle` command to add a demand plan to a specified cycle from an XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing demand plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the demand plan update.
<i>skippedDemandPlanCodes</i>	List object containing codes for those demand plans that were not added to the model.
<i>cycleName</i>	Name of the cycle to update. This parameter is optional; if <i>cycleName</i> is not specified, the current cycle is updated.

## Example

The following example adds a demand plan from an XML file to the specified cycle:

```
# prepare list objects for receiving errors and skipped demand plan codes
set errorsPtr [ListObject]
set skippedDemandPlansPtr [ListObject]
# Add demand plan data
DemandPlan_AddDemandFromXmlToCycle c:/scp/8.12/so-plan/plandata/
DemandPlan.xml $errorsPtr $skippedDemandPlansPtr "Cycle 1"
# Return any errors or skipped demand plans
ListObject_Get $errorsPtr
ListObject_Get $skippedDemandPlansPtr
```

## See Also

Demand Plan View

Working with Views

# DemandPlan AddDemandPlan Command

## Syntax

```
DemandPlan_AddDemandPlan cycleName demandPlanName
```

## Description

Use the `AddDemandPlan` command to add a demand plan to the specified planning cycle.

## Parameters

Parameter	Description
<i>cycleName</i>	Name of the planning cycle to which you want to add the demand plan. If the cycle name contains spaces it must be enclosed in quotation marks.
<i>demandPlanName</i>	Name of the demand plan to add to the model. Demand plan names with spaces it must be enclosed in quotation marks.

## Example

The following example adds the demand plan Q4 Final to the cycle Q4 2006:

```
DemandPlan_AddDemandPlan "Q4 2006" "Q4 Final"
```

## DemandPlan ContainsId Command

### Syntax

```
DemandPlan_ContainsId itemCode locationCode
channelCode
```

### Description

Use the ContainsId command to checks if the model contains a demand plan entry for the specified item, location, and channel.

### Parameters

Parameter	Description
<i>itemCode</i>	Item code of the demand plan entry for which you're searching.
<i>locationCode</i>	Location code of the demand plan entry for which you're searching.
<i>channelCode</i>	Channel code of the demand plan entry for which you're searching.

### Returns

True if the specified demand plan entry exists; otherwise False.

### Example

The following example checks for a demand plan entry for Brown Ale\_341ml at the San Diego distribution center for the Hotels and Restaurants channel:

```
DemandPlan_ContainsId 99011 "San Diego" 1001
```

This command returns:

```
1
```

### See Also

Demand Plan View

## DemandPlan DeleteAllDemands Command

### Syntax

```
DemandPlan_DeleteAllDemands
```

### Description

Use the DeleteAllDemands command to remove all demand plan data from the current cycle.

### See Also

Demand Plan View

Working with Views

## DemandPlan DeleteAllDemandsForCycle Command

### Syntax

```
DemandPlan_DeleteAllDemandsForCycle cycleName
```

### Description

Use the DeleteAllDemandsForCycle command to remove the specified cycle's demand plan data from the model for the specified cycle.

### Parameters

Parameter	Description
<i>cycleName</i>	Name of the cycle all demand plan data is removed from. If the cycle name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example removes all demand plan data from the cycle March 2005.

```
DemandPlan_DeleteAllDemandsForCycle "MARCH 2005"
```

### See Also

Demand Plan View

Working with Views

## DemandPlan DeleteDemand Command

### Syntax

```
DemandPlan_DeleteDemand itemCode locationCode channelCode
```

### Description

Use the DeleteDemand command to remove a specified a demand plan entry from the current cycle.

## Parameters

Parameter	Description
<i>itemCode</i>	Item code for the demand plan entry to be deleted from the model.
<i>locationCode</i>	Location code for the demand plan entry to be deleted from the model.
<i>channelCode</i>	Channel code for the demand plan entry to be deleted from the model.

## Example

The following example deletes the corresponding demand plan entry Brown Ale\_341ml at the San Diego distribution center in the Hotels and Restaurants channel:

```
DemandPlan_DeleteDemand 99011 "San Diego" 1001
```

## See Also

Demand Plan View

Working with Views

## DemandPlan GetDemandIds Command

### Syntax

```
DemandPlan_GetDemandIds demandPlanName
```

### Description

Use the GetDemandIds command to retrieve the demand identifiers from the specified demand plan. Each identifier consists of an item code and its corresponding location and channel.

### Parameters

Parameter	Description
<i>demandPlanName</i>	Name of the demand plan for which you want to retrieve identifiers. If the demand plan contains blanks it must be enclosed in quotation marks.

### Returns

A list of item codes and their corresponding location and channel.

### Example

The following command retrieves all time series codes from the demand plan “Q4 Final”:

```
DemandPlan_GetDemandIds "Q4 Final"
```

## DemandPlan GetDemandPlanNames Command

### Syntax

```
DemandPlan_GetDemandPlanNames cycleName
```

## Description

Use the `GetDemandPlanNames` command to retrieve a list of all demand plans from the specified planning cycle.

## Parameters

Parameter	Description
<i>cycleName</i>	Cycle name for which you want to retrieve all demand plans. If the cycle name contains blanks it must be enclosed in quotation marks.

## Returns

A string containing the demand plans associated with the specified cycle.

## Example

The following example retrieves the demand plans for the cycle Q4 Final:

```
DemandPlan_GetDemandPlanNames "Q4 Final"
```

# DemandPlan GetSupplyPlans Command

## Syntax

```
DemandPlan_GetSupplyPlans cycleName demandPlanName
```

## Description

Use the `GetSupplyPlanNames` command to retrieve a list of all supply plans from the specified planning cycle and demand plan.

## Parameters

Parameter	Description
<i>cycleName</i>	Cycle name for which you want to retrieve all supply plans. If the cycle name contains blanks it must be enclosed in quotation marks.
<i>demandPlanName</i>	Demand plan that is associated with the supply plans that you want to retrieve. If the demand plan name contains blanks it must be enclosed in quotation marks.

## Returns

A list of all supply plans that are associated with the specified cycle and demand plan.

## Example

The following example gets supply plans for the planning cycle Q4 Final and demand plan Projected Q4 Demand:

```
DemandPlan_GetSupplyPlans "Q4 Final" "Projected Q4 Demand"
```

## DemandPlan StrictUpdateDemandFromXml Command

### Syntax

```
DemandPlan_StrictUpdateDemandFromXml filename errors
invalidDemandPlanCodes
```

### Description

Use the StrictUpdateDemandFromXml command to do a strict update of your model demand plan data using the specified XML file. If a demand plan in the XML file does not exist in the model then that entry is ignored, and its added to the list of invalid demand plan codes.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing demand plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidDemandPlanCodes</i>	List object containing demand plans that were not added to the model. Only demand plans that already exist in your model are updated.

### Example

The following example performs a strict update of the model's demand plan data using the specified XML file:

```
# prepare list objects for receiving errors and invalid demand plan codes
set errorsPtr [ListObject]
set invalidDemandPlanPtr [ListObject]
# Add demand plan data
DemandPlan_StrictUpdateDemandFromXml c:/scp/8.12/so-plan/plandata/
DemandPlan.xml $errorsPtr $invalidDemandPlanPtr
# Return any errors or skipped demand plans
ListObject_Get $errorsPtr
ListObject_Get $invalidDemandPlanPtr
```

### See Also

Demand Plan View  
Working with Views

## DemandPlan StrictUpdateDemandFromXmlToCycle Command

### Syntax

```
DemandPlan_StrictUpdateDemandFromXmlToCycle filenameerrors invalidDemandPlanCodes
cycleName
```

### Description

Use the StrictUpdateDemandFromXmlToCycle command to do a strict update of a cycle's demand plan data using the specified XML file. If a demand plan in the XML file does not exist in the model, then that entry is ignored, and its added to the list of invalid demand plan codes.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing demand plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidDemandPlanCodes</i>	List object containing demand plans that were not added to the model. Only demand plans that already exist in your model are updated.
<i>cycleName</i>	Cycle to be updated. If the cycle name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example performs a strict update of the specified planning cycle's demand plan data:

```
# prepare list objects for receiving errors and invalid demand plan codes
set errorsPtr [ListObject]
set invalidDemandPlanPtr [ListObject]
# Add demand plan data
DemandPlan_StrictUpdateDemandFromXmlToCycle c:/scp/8.12/so-plan/plandata/
/DemandPlan.xml $errorsPtr $invalidDemandPlanPtr "Cycle 1"
# Return any errors or skipped demand plans
ListObject_Get $errorsPtr
ListObject_Get $invalidDemandPlanPtr
```

## See Also

Demand Plan View  
Working with Views

## DemandPlan UpdateDemand Command

### Syntax

```
DemandPlan_UpdateDemand itemCode locationCode channelCode values
```

### Description

Use the UpdateDemand command to update a demand plan in the model. The passed in values do not have to overlap with the existing values in the model.

## Parameters

Parameter	Description
<i>itemCode</i>	Item code for the demand plan entry to be updated.
<i>locationCode</i>	Location code for the demand plan entry to be updated.
<i>channelCode</i>	Channel code for the demand plan entry to be updated.
<i>periodStartDate</i>	Date that specifies the starting period for the new values, in the format YYYY-MM-DD.
<i>values</i>	Vector containing new demand values for the corresponding item, location, channel and planning period. Sales and Operations Planning updates as many planning periods as there are values.

## Example

The following example updates a demand plan for Brown Ale\_341ml at the San Diego distribution center in the Hotels and Restaurants channel:

```
DemandPlan_UpdateDemand 99011 "San Diego" 1001 2004-09-13 40
```

## See Also

Demand Plan View  
Working with Views

## DemandPlan UpdateDemandFromXml Command

### Syntax

```
DemandPlan_UpdateDemandFromXml filename errors addedDemandPlanCodes
```

### Description

Use the UpdateDemandFromXml command to update a demand plan using data from an XML file. This command updates demand plans that exist in the model, and adds those demand plans that are not present.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing demand plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedDemandPlanCodes</i>	List object containing demand plans that were added to the model.

## Example

The following example updates the model using demand plan data from the specified XML file:

```
# prepare list objects for receiving errors and added demand plan codes
set errorsPtr [ListObject]
set addedDemandPlanPtr [ListObject]
```



```
# Update demand plan data
DemandPlan_UpdateDemandFromXml c:/scp/8.12/so-plan/plandata/
DemandPlan.xml $errorsPtr $addedDemandPlanPtr
# Return any errors or added demand plans
ListObject_Get $errorsPtr
ListObject_Get $addedDemandPlanPtr
```

### See Also

Demand Plan View  
Working with Views

## DemandPlan UpdateDemandFromXmlToCycle Command

### Syntax

```
DemandPlan_UpdateDemandFromXmlToCycle filename errors addedDemandPlanCodes
```

### Description

Use the UpdateDemandFromXmlToCycle command to update a demand plan for a specific cycle using data from an XML file. This command updates demand plans that exist in the model, and it adds those demand plans that are not present.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing demand plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedDemandPlanCodes</i>	List object containing demand plans that were added to the model.
<i>cycleName</i>	Name of the cycle to update.

### Example

The following example updates the model's demand plan data for the specified planning cycle:

```
# prepare list objects for receiving errors and invalid demand plan codes
set errorsPtr [ListObject]
set addedDemandPlanPtr [ListObject]
# Add demand plan data
DemandPlan_UpdateDemandFromXmlToCycle c:/scp/8.12/so-plan/plandata/
DemandPlan.xml $errorsPtr $addedDemandPlanPtr "Cycle 1"
# Return any errors or added demand plans
ListObject_Get $errorsPtr
ListObject_Get $addedDemandPlanPtr
```

### See Also

Demand Plan View  
Working with Views

## DemandPlan WriteDemandToXml Command

### Syntax

```
DemandPlan_WriteDemandToXml filename
```

### Description

Use the WriteDemandToXml command to write the demand plan to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where the demand plan is written. If the file name included blanks, then it must be enclosed in quotation marks.

### Example

The following example writes the demand plan to the file DemandPlan.xml:

```
DemandPlan_WriteDemandToXml c:/scp/8.12/so-plan/plandata/DemandPlan.xml
```

### See Also

Demand Plan View

Working with Views

## DemandPlan WriteDemandToXmlFromCycle Command

### Syntax

```
DemandPlan_WriteDemandToXmlFromCycle filename cycleName
```

### Description

Use the WriteDemandToXml command to write the demand plan for a specific cycle to an XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where the demand plan is written. If the file name included blanks, then it must be enclosed in quotation marks.
<i>cycleName</i>	Name of the cycle from which you want to write demand data.

### Example

The following example writes the demand plan to the file DemandPlan.xml for the cycle March 2005:

```
DemandPlan_WriteDemandToXmlFromCycle c:/scp/8.12/so-plan/plandata/
DemandPlan.xml "March 2005"
```

### See Also

Demand Plan View

Working with Views

## InventoryHistory AddInventoryHistory Command

### Syntax

```
InventoryHistory_AddInventoryHistory key date values
```

### Description

Use the AddInventoryHistory command to add inventory history values to the model.

### Parameters

Parameter	Description
<i>key</i>	Inventory History key, consisting of a vector containing an item and location value.
<i>date</i>	Start date for the inventory history, in the format YYYY-MM-DD.
<i>values</i>	Vector containing the inventory history values that you want to add to the model. Sales and Operations Planning updates as many planning periods as there are values. For example, if the <i>values</i> parameter contains seven entries, a inventory history value is added to seven planning periods, starting with the specified start date.

### Example

The following example adds an inventory history value for the period 2005–09–13 for the item RES\_2 at location Toronto:

```
InventoryHistory_AddInventoryHistory {RES_2 Toronto} 2005-09-13 {400}
```

## InventoryHistory AddInventoryHistoryFromXml Command

### Syntax

```
InventoryHistory_AddInventoryHistoryFromXml filename errors
```

### Description

Use the AddInventoryHistoryFromXml command to add inventory history values from the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing inventory histories that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.

### Example

The following example adds inventory history values from the file InventoryHistory.xml:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Add inventory history values
```

```
InventoryHistory_AddInventoryHistoryFromXml c:/scp/8.12/so-plan/plandata/
InventoryHistory.xml $errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## InventoryHistory DeleteAllInventoryHistories Command

### Syntax

```
InventoryHistory_DeleteAllInventoryHistories
```

### Description

Use the DeleteAllInventoryHistories command to delete all inventory history values from the model.

## InventoryHistory DeleteInventoryHistory Command

### Syntax

```
InventoryHistory_DeleteInventoryHistory key
```

### Description

Use the DeleteInventoryHistory command to remove the specified inventory history value from the model.

### Parameters

Parameter	Description
<i>key</i>	Inventory History key, consisting of a vector containing an item and location value.

### Example

The following example deletes the inventory history for the item RES\_2 at location Toronto:

```
InventoryHistory_DeleteInventoryHistory {RES_2 Toronto}
```

## InventoryHistory GetBeginningInventory Command

### Syntax

```
InventoryHistory_GetBeginningInventory cycleStartDate key
```

### Description

Use the GetBeginningInventory command to retrieve the specified beginning inventory value from the model.

### Parameters

Parameter	Description
<i>cycleStartDate</i>	Start date for the planning cycle from which you want to retrieve beginning inventory, in the format YYYY-MM-DD.
<i>key</i>	Inventory History key, consisting of a vector containing an item and location value.

## Example

The following example gets beginning inventory values for the planning period beginning June 15, 2005, for the item RES\_2 at location Toronto:

```
InventoryHistory_GetBeginningInventory 2005-06-15 {RES_2 Toronto}
```

## InventoryHistory WriteInventoryHistoryToXml Command

### Syntax

```
InventoryHistory_WriteInventoryHistoryToXml filename
```

### Description

Use the WriteInventoryHistoryToXml command to write all inventory history values to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where inventory history values are written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes all inventory history values to the XML file InventoryHistory.xml:

```
InventoryHistory_WriteInventoryHistoryToXml c:/scp/8.12/so-plan/plandata/
InventoryHistory.xml
```

## Items AddItem Command

### Syntax

```
Items_AddItem itemCode itemName itemDescription
```

### Description

Use the AddItem command to add an item to the model.

### Parameters

Parameter	Description
<i>itemCode</i>	Code for the item that you want to add to the model.
<i>itemName</i>	Name for the item that you want to add to the model.
<i>itemDescription</i>	Detailed description of the new item. If the item description contains blanks, then it must be enclosed in quotation marks.

## Example

The following example adds an item Wheat Ale\_355ml to the model:

```
Items_AddItem 99014 "Wheat Ale 355ml" Can
```

## Items AddItemChannelPrice Command

### Syntax

```
Items_AddItemChannelPrice itemCode channelCode period price
```

### Description

Use the Items\_AddItemChannelPrice command to add an item price for the specified period.

### Parameters

Parameter	Description
<i>itemCode</i>	Code for the item to which you want to add an item price.
<i>channelCode</i>	Code for the channel to which you want to add an item price.
<i>period</i>	Start date for the new item price, in the YYYY-MM-DD format. This cost value remains in effect until the next defined period.
<i>price</i>	The item price in dollars.

### Example

The following example adds the price for the item Brown Ale 355ml to the channel LCBO:

```
Items_AddItemChannelPrice 99012 LCBO 2005-10-04 5.85
```

## Items AddItemChannelPricesFromXml Command

### Syntax

```
Items_AddItemChannelPricesFromXml filename errors
```

### Description

Use the AddItemChannelPricesFromXml command to add item prices from the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing item prices to be added to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.

### Example

The following example prepares the list objects needed to receive errors and skipped items, and then imports items from the specified XML file:

```
# prepare list objects for receiving errors and skipped items
set errorsPtr [ListObject]
set skippedItemsPtr [ListObject]
# Perform the import
```

```

Items_AddItemsFromXml c:/scp/8.12/so-plan/plandata/ItemList.xml $errorsPtr
$skippedItemsPtr
# Return any errors or skipped items
ListObject_Get $errorsPtr
ListObject_Get $skippedItemsPtr

```

## Items AddItemHoldingCost Command

### Syntax

```
Items_AddItemHoldingCost itemCode locationCode period cost
```

### Description

Use the AddItemHoldingCost command to add the cost associated with holding an item in inventory at a particular location.

### Parameters

Parameter	Description
<i>itemCode</i>	Code for the item to which you want to add production costs.
<i>locationCode</i>	Location where the holding cost is being applied.
<i>period</i>	Start date for the holding cost, in the format YYYY-MM-DD. This cost value remains in effect until the next defined period.
<i>cost</i>	Cost, in dollars, associated with keeping the item in inventory at this location.

### Example

The following example adds the holding cost for the item Wheat Ale 355ml at the San Diego location:

```
Items_AddItemHoldingCost 99014 "San Diego" 2005-06-15 90
```

## Items AddItemHoldingCostsFromXml Command

### Syntax

```
Items_AddItemHoldingCostsFromXml filename errors
```

### Description

Use the AddItemHoldingCostsFromXml command to add holding costs to the model from the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing holding cost data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.

## Example

The following example adds item holding costs to the model from the specified XML file:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Add item holding costs
items_AddHoldingCostsFromXml c:/scp/8.12/so-plan/plandata/HoldingCosts.xml
$errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## Items AddItemProductionCost Command

### Syntax

```
Items_AddItemProductionCost itemCode locationCode period cost
```

### Description

Use the AddItemProductionCost command to add to the model the cost associated with producing an item at a particular location.

### Parameters

Parameter	Description
<i>itemCode</i>	Code for the item to which you want to add production costs.
<i>locationCode</i>	Location code where the production cost is applied.
<i>period</i>	Start date for the new production cost, in format YYYY-MM-DD. This cost value remains in effect until the next defined period.
<i>cost</i>	Cost, in dollars, associated with producing the item at this location.

### Example

The following example adds the production cost for the item Brown Ale 355ml at the Halifax location:

```
Items_AddItemProductionCost 99012 Halifax 2005-06-15 110
```

## Items AddItemProductionCostsFromXml Command

### Syntax

```
Items_AddItemProductionCostsFromXml filename errors
```

### Description

Use the AddItemProductionCostsFromXml command to add the cost associated with producing an item from the specified XML file.



## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing production cost data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.

## Example

The following example adds item production costs to the model from the specified XML file:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Add item production costs
items_AddProductionCostsFromXml c:/scp/8.12/so-plan/plandata/
ProductionCosts.xml $errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## Items AddItemsFromXml Command

### Syntax

```
Items_AddItemsFromXml filename errors skippedItemCodes
```

### Description

Use the AddItemsFromXml command to add items from the specified XML file to the model. Items that are already present in the model are skipped.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing items to be added to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>skippedItemCodes</i>	List object containing codes of items that were not added to the model.

## Example

The following example adds items to the model from the specified XML file:

```
# prepare list objects for receiving errors and skipped item codes
set errorsPtr [ListObject]
set skippedItemCodesPtr [ListObject]
# Add items
items_AddItemsFromXml c:/scp/8.12/so-plan/plandata/ItemList.xml $errorsPtr
$skippedItemCodesPtr
# Return any errors or skipped item codes
ListObject_Get $errorsPtr
```

```
ListObject_Get $skippedItemCodesPtr
```

## Items CheckExists Command

### Syntax

```
Items_CheckExists itemCode
```

### Description

Use the CheckExists command to verify that the specified item exists in the model.

### Parameters

Parameter	Description
<i>itemCode</i>	Code for the item that you want to verify exists in the model.

### Returns

An error, if the specified item name does not exist.

### Example

The following example searches the model for the Brown Ale 355ml item:

```
Items_CheckExists 99012
```

## Items DeleteAllItems Command

### Syntax

```
Items_DeleteAllItems
```

### Description

Use the DeleteAllItems command to delete all items from the model.

### Example

The following example deletes all items from the Sales and Operations Planning model:

```
Items_DeleteAllItems
```

## Items DeleteItem Command

### Syntax

```
Items_DeleteItem itemCode
```

### Description

Use the DeleteItem command to delete an item from your model.

## Parameters

Parameter	Description
<i>itemCode</i>	Code for the item that you want to delete from the model.

## Example

The following example deletes the item Brown Ale 355ml from the model:

```
Items_DeleteItem 99012
```

## Items GetItemCodes Command

### Syntax

```
Channels_GetItemCodes
```

### Description

Use the GetItemCodes command to retrieve the codes for all of the items in the model.

### Returns

A list of all item codes in the model.

## Example

The following example retrieves all item codes from the Sales and Operations Planning model:

```
Items_GetItemCodes
```

## Items GetItemProperties Command

### Syntax

```
Items_GetItemProperties itemCode
```

### Description

Use the GetItemProperties command to retrieve item properties using the specified item code.

### Parameters

Parameter	Description
<i>itemCode</i>	Code for the item with properties that you want to retrieve.

### Returns

A paired list of properties and their associated values.

## Example

The following example retrieves properties for the Brown Ale 355ml item:

```
Items_GetItemProperties 99012
```

This Tcl command returns:

```
{Description {Can}} {{Item Family} {Brown Ale }} {Item Group Ale}
{Alcohol Content {5%}}
```

## Items GetItemPropertyNames Command

### Syntax

```
Items_GetItemPropertyNames
```

### Description

Use the GetItemPropertyNames command to retrieve a list of your model's item property attributes.

### Returns

A list your model's item property attributes.

### Example

The following example retrieves a list of all property names from the Sales and Operations Planning model:

```
Items_GetItemPropertyNames
```

## Items SetItemProperties Command

### Syntax

```
Items_SetItemProperties itemCode propertyList
```

### Description

Use the SetItemProperties command to set property values for the item with the specified code.

### Parameters

Parameter	Description
<i>itemCode</i>	Code for the item with properties that you want to set.
<i>propertyList</i>	Vector of paired properties that you want to apply to the specified item.

### Example

The following example sets values for the Height and Weight properties, for item b1:

```
Items_SetItemProperties b1 {{Height 100} {Weight 200}}
```

## Items SetItemPropertyNames Command

### Syntax

```
Items_SetItemPropertyNames itemCode propertyNames
```

### Description

Use the SetItemPropertyNames command to set item property names.

## Parameters

Parameter	Description
<i>itemCode</i>	Code for the item with properties that you want to set.
<i>propertyNames</i>	Vector containing a list of property names.

## Example

The following example adds the Height and Weight properties to item BER\_2:

```
Items_SetItemPropertyNames BER_2 {Height Weight}
```

## Items StrictUpdateItemsFromXml Command

### Syntax

```
Items_StrictUpdateItemsFromXml filename invalidItemCodes
```

### Description

Use the StrictUpdateItemsFromXml command to do a strict update of your model item data using the specified XML file. If a item in the XML file does not exist in the model then that entry is ignored and its code is added to the list of invalid item codes.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing item data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidItemCodes</i>	List object containing items that were not added to the model. Only items that already exist in your model are updated.

## Example

The following example performs a strict update of the model using item data from the specified XML file:

```
# prepare list objects for receiving errors and invalid item codes
set errorsPtr [ListObject]
set invalidItemCodesPtr [ListObject]
# Update item data
Inventory_StrictUpdateItemsFromXml c:/scp/8.12/so-plan/plandata/ItemList.xml
$errorsPtr $invalidItemCodesPtr
# Return any errors or invalid item codes
ListObject_Get $errorsPtr
ListObject_Get $invalidItemCodesPtr
```

## Items UpdateItemsFromXml Command

### Syntax

```
Items_UpdateItemsFromXml filename errors addedItemCodes
```

### Description

Use the UpdateItemsFromXml command to update model item data using the specified XML file. This command updates items that are already present in the model, and creates items that are not present in the model.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing item data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedItemCodes</i>	List object containing the items that were added to the model.

### Example

The following example performs an update of the model using item data from the specified XML file:

```
# prepare list objects for receiving errors and added item codes
set errorsPtr [ListObject]
set addedItemCodesPtr [ListObject]
# Update item data
Inventory_UpdateItemsFromXml c:/scp/8.12/so-plan/plandata/ItemList.xml
$errorsPtr $addedItemCodesPtr
# Return any errors or added item codes
ListObject_Get $errorsPtr
ListObject_Get $addedItemCodesPtr
```

## Items WriteItemChannelPricesToXml Command

### Syntax

```
Items_WriteItemChannelPricesToXml filename
```

### Description

Use the WriteItemChannelPricesToXml command to write the item prices to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	XML file name and path where item prices is written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes item prices to the file ItemList.xml:

```
Items_WriteItemChannelPricesToXml ItemList.xml
```

## Items WriteItemHoldingCostsToXml Command

### Syntax

```
Items_WriteItemHoldingCostsToXml filename
```

### Description

Use the WriteItemHoldingCostsToXml command to write the costs associated with holding an item in inventory to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	XML file name and path where holding costs are written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes holding cost data to the file ItemList.xml:

```
Items_WriteItemHoldingCostsToXml ItemList.xml
```

## Items WriteItemProductionCostsToXml Command

### Syntax

```
Items_WriteItemProductionCostsToXml filename
```

### Description

Use the WriteItemProductionCostsToXml command to write the costs associated with producing an item to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	XML file name and path where production costs is written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes production cost data to the file ItemList.xml:

```
Items_WriteItemProductionCostsToXml ItemList.xml
```

## Items WriteItemsToXml Command

### Syntax

```
Items_WriteItemsToXml filename
```

### Description

Use the WriteItemsToXml command to write all model items to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	The XML file where model item data is written. If the file name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example writes all model items to the file ItemList.xml:

```
Items_WriteItemsToXml c:/scp/8.12/so-plan/plandata/ItemList.xml
```

## Locations AddLocation Command

### Syntax

```
Locations_AddLocation locationCode locationName
```

### Description

Use the AddLocation command to add a location to the model.

### Parameters

Parameter	Description
<i>locationCode</i>	Code of the location that you want to add to the model.
<i>locationName</i>	Name of the location that you want to add to the model.

### Example

The following example adds a location, San Diego, to the model:

```
Items_AddLocation "San Diego" "San Diego Distribution Center"
```

## Locations AddLocationsFromXml Command

### Syntax

```
Locations_AddLocationsFromXml filename errors skippedLocationCodes
```

### Description

Use the AddLocationsFromXml command to add new locations to the model using an input XML file. Locations that are already present in the model are skipped.



## Parameters

Parameter	Description
<i>filename</i>	File name and path of the input XML file. If the path includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered when adding locations.
<i>skippedLocationCodes</i>	List object containing codes of locations that were not added to the model.

## Example

The following example adds items to the model from the specified XML file:

```
# prepare list objects for receiving errors and skipped location codes
set errorsPtr [ListObject]
set skippedLocationCodesPtr [ListObject]
# Add locations
DemandPlan_AddLocationsFromXml c:/scp/8.12/so-plan/plandata/Locations.xml
$errorsPtr $skippedLocationCodesPtr
# Return any errors or skipped location codes
ListObject_Get $errorsPtr
ListObject_Get $skippedLocationCodesPtr
```

## Locations CheckExists Command

### Syntax

```
Locations_CheckExists locationCode
```

### Description

Use the CheckExists command to verify that the specified location exists in the model.

### Parameters

Parameter	Description
<i>locationCode</i>	Name of the location that you want to verify exists in the model.

### Returns

An error if the specified location does not exist.

### Example

The following example searches the model for the Atlanta distribution center:

```
Locations_CheckExists Atlanta
```

## Locations DeleteAllLocations Command

### Syntax

```
Locations_DeleteAllLocations
```

## Description

Use the DeleteAllLocations command to delete all locations from the model.

## Example

The following example deletes all locations from the Sales and Operations Planning model:

```
Locations_DeleteAllLocations
```

## Locations DeleteLocation Command

### Syntax

```
Locations_DeleteLocation locationCode
```

### Description

Use the DeleteLocation command to delete a location from your model.

### Parameters

Parameter	Description
<i>locationCode</i>	Code for the location that you want to remove from the model.

## Example

The following example deletes the location San Diego from the model:

```
Locations_DeleteLocation "San Diego"
```

## Locations GetLocationCodes Command

### Syntax

```
Locations_GetLocationCodes
```

### Description

Use the GetLocationCodes command to retrieve the codes for all of the locations in the model.

### Returns

A list of all location codes in the model.

## Example

The following example retrieves all location codes from the Sales and Operations Planning model:

```
Locations_GetLocationCodes
```

## Locations GetLocationProperties Command

### Syntax

```
Locations_GetLocationProperties locationCode
```

## Description

Use the `GetLocationProperties` command to retrieve location properties using the specified location code.

## Parameters

Parameter	Description
<i>locationCode</i>	Code for the location with properties that you want to retrieve.

## Returns

A paired list of location properties and their associated values.

## Example

The following example retrieves properties for the San Diego Location:

```
Locations_GetLocationProperties "San Diego"
```

This Tcl command returns:

```
{Name {San Diego Distribution Center}} {{Location Groups} {US Branches }}  
{Branch Type {DC}} {Country {United States}}
```

# Locations GetLocationPropertyNames Command

## Syntax

```
Locations_GetLocationPropertyNames
```

## Description

Use the `GetLocationPropertyNames` command to retrieve a list of your model's location property attributes.

## Returns

A list your model's location property attributes.

## Example

The following example retrieves all location property names from the Sales and Operations Planning model:

```
Locations_GetLocationPropertyNames
```

# Locations SetLocationProperties Command

## Syntax

```
Locations_SetLocationProperties locationCode propertyList
```

## Description

Use the `SetLocationProperties` command to set property values for the location with the specified code.

## Parameters

Parameter	Description
<i>locationCode</i>	Code for the location with properties that you want to set.
<i>propertyList</i>	Vector of paired properties that you want to apply to the specified location.

## Example

The following example adds a location L1 and sets its properties:

```
# Add the location
Locations_AddLocation "L1" ""

# Set location property names
Locations_SetLocationPropertyNames {City Province Country}

# Set location properties
Locations_SetLocationProperties L1 {{City Waterloo} {Province Ontario}
{Country Canada}}

# Test to see if they were set correctly; it returns:
# {City Waterloo} {Province Ontario} {Country Canada}
Locations_GetLocationProperties L1
```

## Locations SetLocationPropertyNames Command

### Syntax

```
Locations_SetLocationPropertyNames locationCode propertyNames
```

### Description

Use the SetLocationPropertyNames command to set location property names.

### Parameters

Parameter	Description
<i>locationCode</i>	Code for the location with properties that you want to set.
<i>propertyNames</i>	Vector containing a list of property names.

## Example

The following example adds the Height, Weight, and Color properties to location TOR2:

```
Locations_SetLocationPropertyNames TOR2 {Height Weight Color}
```

## Locations StrictUpdateLocationsFromXml Command

### Syntax

```
Items_StrictUpdateLocationsFromXml filename invalidLocationCodes
```

### Description

Use the StrictUpdateLocationsFromXml command to do a strict update of your model location data using the specified XML file. If a location in the XML file does not exist in the model then that entry is ignored and its code is added to the list of invalid location codes.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing location data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidLocationCodes</i>	List object containing locations that were not added to the model. Only locations that already exist in your model are updated.

### Example

The following example performs a strict update of the model using location data from the specified XML file:

```
# prepare list objects for receiving errors and invalid location codes
set errorsPtr [ListObject]
set invalidLocationCodesPtr [ListObject]
# Update location data
Locations_StrictUpdateLocationsFromXml c:/scp/8.12/so-plan/plandata/
Locations.xml $errorsPtr $invalidLocationCodesPtr
# Return any errors or invalid location codes
ListObject_Get $errorsPtr
ListObject_Get $invalidLocationCodesPtr
```

## Locations UpdateLocationsFromXml Command

### Syntax

```
Locations_UpdateLocationsFromXml filename errors addedLocationCodes
```

### Description

Use the UpdateLocationsFromXml command to update model location data using the specified XML file. This command updates locations that are already present in the model, and it creates locations that are not present in the model.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing location data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedLocationCodes</i>	List object containing the locations that were added to the model.

## Example

The following example performs an update of the model using location data from the specified XML file:

```
# prepare list objects for receiving errors and added location codes
set errorsPtr [ListObject]
set addedLocationCodesPtr [ListObject]
# Update location data
Locations_UpdateLocationsFromXml c:/scp/8.12/so-plan/plandata/Locations.xml
$errorsPtr $addedLocationCodesPtr
# Return any errors or added location codes
ListObject_Get $errorsPtr
ListObject_Get $addedLocationCodesPtr
```

## Locations WriteLocationsToXml Command

### Syntax

```
Locations_WriteLocationsToXml filename
```

### Description

Use the WriteLocationsToXml command to write all model locations to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	File name and path of the XML file where location data is written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes all model locations to the file LocationList.xml:

```
Locations_WriteLocationsToXml c:/scp/8.12/so-plan/plandata/LocationList.xml
```

## Log ClearDbLog Command

### Syntax

```
Log_ClearDbLog
```

## Description

Use the ClearDbLog command to delete all log entries.

## Example

The following example clears all log entries:

```
Log_ClearDbLog
```

## See Also

Logs

# Log ClearDbLogOnAndBeforeDate Command

## Syntax

```
Log_ClearDbLogOnAndBeforeDate date
```

## Description

Use the ClearDbLogOnAndBeforeDate command to remove all log entries on and before the specified date.

## Parameters

Parameter	Description
<i>Date</i>	Date, in the format YYYY-MM-DD, before which all log entries are removed.

## Example

The following example deletes all log entries on or before October 4, 2003:

```
Log_ClearDbLogOnAndBeforeDate 2003-10-04
```

## See Also

Logs

# Log ClearFileLog Command

## Syntax

```
Log_ClearFileLog filename
```

## Description

Use the ClearFileLog command to delete the specified log file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the log file to be cleared. If the log file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example clears the NexusServer.log file:

```
Log_ClearFileLog NexusServer.log
```

## See Also

Logs

## Log GetDbLogEvents Command

### Syntax

```
Log_GetDbLogEvents
```

### Description

Use the GetDbLogEvents command to retrieve a list of log events from the Sales and Operations Planning database.

### Returns

A vector containing the log file contents.

## See Also

Logs

## Log GetFileLogEvents Command

### Syntax

```
Log_GetFileLogEvents filename
```

### Description

Use the GetFileLogEvents command to retrieve a list of log events from the specified file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the log file to retrieve events from. If the log file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example retrieves events from the SOP Import.log file.

```
Log_GetFileLogEvents "SOP Import.log"
```

## See Also

Logs



## Log GetLoggingStatus Command

### Syntax

```
Log GetLoggingStatus
```

### Description

Use the GetLoggingStatus command to return the current system logging status.

### Returns

A vector containing all defined logging categories.

### See Also

Logs

## Log LogEvent Command

### Syntax

```
Log_LogEvent level category message
```

### Description

Use the LogEvent command to write an event to the error log.

### Parameters

Parameter	Description
<i>level</i>	Event level associated with the error message. Valid values are: <ul style="list-style-type: none"><li>• INFO.</li><li>• WARNING.</li><li>• ERROR.</li></ul>
<i>category</i>	Category to which the log event is applied. Categories enable you to group log events using meaningful descriptions. For example, you may want to write events to the application log with a time stamp, notifying you of the progress of your integration script.
<i>message</i>	Event message to write to the log. If the event message includes blanks, then it must be enclosed in quotation marks.

### Example

The following example writes a message to the event log:

```
Log_LogEvent INFO integration "Demand data import successful"
```

### See Also

Logs

## Log MultiLogEvent Command

### Syntax

```
Log_MultiLogEvent level categories  
message
```

### Description

Use the MultiLogEvent command to write a single message to the log file under multiple categories.

### Parameters

Parameter	Description
<i>level</i>	Event level associated with the error message. Valid values are: <ul style="list-style-type: none"><li>• INFO.</li><li>• WARNING.</li><li>• ERROR.</li></ul>
<i>categories</i>	Vector containing categories that the log event is applied to. Categories enable you to group log events using meaningful descriptions.
<i>message</i>	The event message to write to the log.

### Example

The following example creates two logging categories:

```
set categories [list category4 category5]  
Log_MultiLog INFO $categories testing multilog
```

These commands produce the following log results:

```
2004-10-26 11:55:43 INFO category4->testing multilog  
2004-10-26 11:55:43 INFO category5->testing multilog
```

### See Also

Logs

## Log ResetLoggingSystem Command

### Syntax

```
Log_ResetLoggingSystem
```

### Description

Use the ResetLoggingSystem command to stop logging for all levels and categories.

### See Also

Logs

## Log StartDbLog Command

### Syntax

```
Log_StartDbLog
```

### Description

Use the StartDbLog command to start logging events to the database.

### See Also

Logs

## Log StartFileLog Command

### Syntax

```
Log_StartFileLog filename
```

### Description

Use the StartFileLog command to start logging events to the specified log file.

### Parameters

Parameter	Description
<i>filename</i>	Log file name used to start logging. This file is located in the \so-plan\vers_8.12\logs directory. If the log file name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example starts logging to the file SOP Import.log:

```
Log_StartLogFile "SOP Import.log"
```

### See Also

Logs

## Log StartLogging Command

### Syntax

```
Log_StartLogging level category
```

### Description

Use the StartLogging command to enable logging for the specified level and category.

## Parameters

Parameter	Description
<i>level</i>	Logging level to enable. Valid values are: <ul style="list-style-type: none"><li>• INFO.</li><li>• WARNING.</li><li>• ERROR.</li></ul>
<i>category</i>	Text string pattern used to filter the error log. Use an asterisk (*) to enable wildcard matches. Only events that match the specified category are written to the error log.

## Example

The following example enables event logging for error messages matching the pattern Nexus\*:

```
Log_StartLogging ERROR Nexus*
```

## See Also

Logs

## Log StopAllLogging Command

### Syntax

```
Log_StopAllLogging
```

### Description

Use the StopAllLogging command to remove all defined logging levels and categories from the database.

## See Also

Logs

## Log StopLogging Command

### Syntax

```
Log_StopLogging level category
```

### Description

Use the StopLogging command to disable logging for the specified level and category.

## Parameters

Parameter	Description
<i>level</i>	Logging level that you want to stop. Valid values are: <ul style="list-style-type: none"> <li>• INFO.</li> <li>• WARNING.</li> <li>• ERROR.</li> </ul>
<i>category</i>	The text string pattern that you want to stop logging.

## Example

The following example disables event logging of error messages matching the pattern Nexus\*:

```
Log_StopLogging ERROR Nexus*
```

## See Also

Logs

# MetricExceptionReports DeleteAllReports Command

## Syntax

```
MetricExceptionReports_DeleteAllReports
```

## Description

Use the DeleteAllReports command to remove all generated Metric Exception reports from the model.

## See Also

Metric Exception Reports  
Working with Reports

# MetricExceptionReports DeleteReport Command

## Syntax

```
MetricExceptionReports_DeleteReport reportName
```

## Description

Use the DeleteReport command to delete the specified Metric Exception report from the model.

## Parameters

Parameter	Description
<i>reportName</i>	Name of the report that you want to delete from the model.

## Example

The following example removes the report Brewer Total Report:

```
MetricExceptionReports_DeleteReport "Brewer Total Report"
```

**See Also**

Metric Exception Reports  
Working with Reports

## MetricExceptionReports ExecuteReport Command

**Syntax**

```
MetricExceptionReports_ExecuteReport reportName
```

**Description**

Use the ExecuteReport command to run the specified Metric Exception report.

**Parameters**

Parameter	Description
<i>reportName</i>	Name of the report that you want to execute.

**Example**

The following example executes the report Brewer Total Report:

```
MetricExceptionReports_ExecuteReport "Brewer Total Report"
```

**See Also**

Metric Exception Reports  
Working with Reports

## MetricExceptionReports ExportReports Command

**Syntax**

```
MetricExceptionReports_ExportReports filename
```

**Description**

Use the ExportReports command to write all Metric Exception reports in the model to the specified XML file.

**Parameters**

Parameter	Description
<i>filename</i>	Name and path of the XML file where the exported reports are written. If the file name includes blanks, then it must be enclosed in quotation marks.

**Example**

The following command writes model reports to the file MetricExceptionReports.xml.

```
MetricExceptionReports_ExportReports c:/scp/8.12/so-plan/plandata/  
MetricExceptionReports.xml
```

**See Also**

Metric Exception Reports  
Working with Reports

## MetricExceptionReports GetReportsInfo Command

**Syntax**

```
MetricExceptionReports_GetReportsInfo
```

**Description**

Use the GetReportsInfo command to get information for all reports in the model.

**Returns**

A list containing:

- Report name.
- View name associated with the report.
- View status (true if the view has entries; otherwise false).
- Number of Metric Exception reports.
- Timestamp indicating when the report was last executed.

**Example**

The following example returns a list of all reports in the Sales and Operations Planning model:

```
MetricExceptionReports_GetReportsInfo
```

## MetricExceptionReports GetReportsStatus Command

**Syntax**

```
MetricExceptionReports_GetReportStatus
```

**Description**

Use the GetReportStatus command to retrieve information on all Metric Exception reports defined in the Sales and Operations Planning model.

**Returns**

A report that includes:

- Report name.
- Related view.
- Number of exceptions contained in the report.
- Time of the last execution.

**Example**

The following example retrieves information on all Metric Exception reports defined in the model:

```
MetricExceptionReports_GetReportStatus
```

## See Also

Metric Exception Reports

Working with Reports

# MetricExceptionReports GetReportValues Command

## Syntax

```
MetricExceptionReports_GetReportValues reportName sortType includeFair
startRow endRow
```

## Description

Use the GetReportValues command to retrieve values from the specified Metric Exception report.

## Parameters

Parameter	Description
<i>reportName</i>	Name of the report from which you want to retrieve values. If the report name contains blanks, then it must be enclosed in quotation marks.
<i>sortType</i>	Type of sort to perform. Valid values are: <ul style="list-style-type: none"> <li>• By Key.</li> <li>• By Percentage Deviation.</li> </ul>
<i>includeFair</i>	Boolean value indicating if fair values should be included in the report. Set to True to include items with fair status; otherwise, only items with Poor status are included.
<i>startRow</i>	Start row index.
<i>endRow</i>	End row index.

## Returns

A list of Metric Exception report entries, each of which contains:

- Report key with the metric name appended.
- Actual value.
- Target value.
- Deviation.
- Percentage deviation.
- Metric target status.

## Example

The following example retrieves the values from Report 1:

```
# Get report values
MetricExceptionReports_GetReportValues "Report 1" "% Difference" 1 0 10
```



## MetricExceptionReports ImportReports Command

### Syntax

```
MetricExceptionReports_ImportReports filename errors skippedReports
```

### Description

Use the ImportReports command to add Metric Exception report definitions from the specified XML file. Metric Exception report definitions that are already present in the model are skipped.

### Parameters

Parameter	Description
<i>filename</i>	File name and path of the input XML file. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered while importing reports.
<i>skippedReports</i>	List object containing reports that were not added to the model.

### Example

The following example imports reports from the specified XML file:

```
# prepare list objects for receiving errors and skipped reports
set errorsPtr [ListObject]
set skippedReportsPtr [ListObject]
# Import report definitions
MetricExceptionReports_ImportReports c:/scp/8.12/so-plan/plandata/
MetricExceptionReports.xml $errorsPtr $skippedReportsPtr
# Return any errors or skipped reports
ListObject_Get $errorsPtr
ListObject_Get $skippedReportsPtr
```

### See Also

Metric Exception Reports  
Working with Reports

## NetChangeReports DeleteAllReports Command

### Syntax

```
NetChangeReports_DeleteAllReports
```

### Description

Use the DeleteAllReports command to remove all generated Net Change reports from the model.

### See Also

Net Change Reports  
Working with Reports

## NetChangeReports DeleteReport Command

### Syntax

```
NetChangeReports_DeleteReport reportName
```

### Description

Use the DeleteReport command to delete the specified Net Change report from the model.

### Parameters

Parameter	Description
<i>reportName</i>	The report that you want to delete from the model.

### Example

The following example removes the report Brewer Total Report:

```
NetChangeReports_DeleteReport "Brewer Total Report"
```

### See Also

Net Change Reports  
Working with Reports

## NetChangeReports ExecuteReport Command

### Syntax

```
NetChangeReports_ExecuteReport reportName
```

### Description

Use the ExecuteReport command to run the specified Net Change report.

### Parameters

Parameter	Description
<i>reportName</i>	Name of the report that you want to execute.

### Example

The following example executes the report Brewer Total Report:

```
NetChangeReports_ExecuteReport "Brewer Total Report"
```

### See Also

Net Change Reports  
Working with Reports

## NetChangeReports ExportReports Command

### Syntax

```
NetChangeReports_ExportReports filename
```

## Description

Use the ExportReports command to write all Net Change reports in the model to the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where the exported reports are written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following command writes model reports to the file NetChangeReports.xml.

```
NetChangeReports_ExportReports c:/scp/8.12/so-plan/plandata/NetChangeReports.xml
```

## See Also

Net Change Reports  
Working with Reports

# NetChangeReports GetReportsInfo Command

## Syntax

```
NetChangeReports_GetReportsInfo
```

## Description

Use the GetReportsInfo command to retrieve information on all NetChangeReports stored in the model.

## Returns

A list containing:

- Report name.
- View name associated with the report.
- View status (true if the view has entries; otherwise false).
- Number of Metric Exception reports.
- Timestamp indicating when the report was last executed.

# NetChangeReports GetReportsStatus Command

## Syntax

```
NetChangeReports_GetReportStatus
```

## Description

Use the GetReportStatus command to retrieve information on all Net Change reports defined in the model.

## Returns

A report that includes:

- Report name
- Related view
- Number of exceptions contained in the report
- Time of the last execution.

### See Also

Net Change Reports  
Working with Reports

## NetChangeReports GetReportValues Command

### Syntax

```
NetChangeReports_GetReportValues reportName sortColumn isDescending
startRow endRow
```

### Description

Use the GetReportValues command to retrieve values from the specified Net Change report.

### Parameters

Parameter	Description
<i>reportName</i>	Name of the report from which you want to retrieve values. If the report name contains blanks, then it must be enclosed in quotation marks.
<i>sortColumn</i>	The report column on which you want to sort. Valid values are: <ul style="list-style-type: none"> <li>• % Difference.</li> <li>• Abs. % Difference.</li> <li>• Value Difference.</li> <li>• Abs. Value Difference.</li> </ul>
<i>isDescending</i>	True if the sort order is descending; otherwise False.
<i>startRow</i>	Start row index.
<i>endRow</i>	End row index.

### Example

The following example creates a report, Report 1, and retrieves the report values.

```
# Add the Report Definition
NetChangeReports_AddDefinition "Report 1" view1 "Cycle 1" dem1 "" "Cycle 2"
dem1 "" "Monthly" "Nov 2003" "Apr 2004" "Unit Demand" ""

# Execute the report
NetChangeReports_Execute "Report 1"

# Get report values
NetChangeReports_GetReportValues "Report 1" "% Difference" 1 0 10
```

## NetChangeReports ImportReports Command

### Syntax

```
NetChangeReports_ImportReports filename errors skippedReports
```

### Description

Use the ImportReports command to add Net Change report definitions from the specified XML file. Net Change report definitions that are already present in the model are skipped.

### Parameters

Parameter	Description
<i>filename</i>	File name and path of the input XML file. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered while importing reports.
<i>skippedReports</i>	List object containing reports that were not added to the model.

### Example

The following example imports reports from the specified XML file:

```
# prepare list objects for receiving errors and skipped reports
set errorsPtr [ListObject]
set skippedReportsPtr [ListObject]
# Import report definitions
NetChangeReports_ImportReports c:/scp/8.12/so-plan/plandata/
NetChangeReports.xml $errorsPtr $skippedReportsPtr
# Return any errors or skipped reports
ListObject_Get $errorsPtr
ListObject_Get $skippedReportsPtr
```

### See Also

Net Change Reports  
Working with Reports

## Periods AddPeriod Command

### Syntax

```
Periods_AddPeriod startDate fiscalMonth fiscalQuarter fiscalYear
```

### Description

Use the AddPeriod command to add a calendar period to the model.

## Parameters

Parameter	Description
<i>startDate</i>	Calendar period start date, in the format YYYY-MM-DD.
<i>fiscalMonth</i>	Fiscal month to associate with the calendar period.
<i>fiscalQuarter</i>	Fiscal quarter to associate with the calendar period.
<i>fiscalYear</i>	Fiscal year to associate with the calendar period.

## Example

The following example adds the January 7, 2005, calendar period to the model:

```
Periods_AddPeriod 2005 "Jan 2005" "March 2005" 2005
```

## See Also

Managing Planning Periods

## Periods DeleteAllPeriods Command

### Syntax

```
Periods_DeleteAllPeriods
```

### Description

Use the DeleteAllPeriods command to remove all calendar periods from the model.

## See Also

Managing Planning Periods

## Periods DeletePeriod Command

### Syntax

```
Periods_DeletePeriod startDate
```

### Description

Use the DeletePeriod command to remove from the model the period with the specified start date.

## Parameters

Parameter	Description
<i>startDate</i>	Start date of the period to delete from the model, in the format YYYY-MM-DD.

## Example

The following example removes the period starting on June 15, 2005, from the model:

```
Periods_DeletePeriod 2005-06-15
```

**See Also**

Managing Planning Periods

## Periods ExportPeriods Command

**Syntax**`Periods_ExportPeriods filename`**Description**

Use the ExportPeriods command to write calendar period data to the specified file.

**Parameters**

Parameter	Description
<i>filename</i>	File name and path of the destination XML file. If the file name includes blanks, then it must be enclosed in quotation marks.

**Example**

The following example writes model calendar periods to the file PeriodsList.xml:

```
Periods_ExportPeriods PeriodsList.xml
```

**See Also**

Managing Planning Periods

## Periods ImportPeriods Command

**Syntax**`Periods_ImportPeriods filename`**Description**

Use the ImportPeriods command to import calendar period data from the specified file.

**Parameters**

Parameter	Description
<i>filename</i>	File name and path of the XML file containing calendar period data. If the file name includes blanks, then it must be enclosed in quotation marks.

**Example**

The following example imports model calendar periods from the file PeriodsList.xml:

```
Periods_ImportPeriods PeriodsList.xml
```

**See Also**

Managing Planning Periods

## PlanMapping AddPlanMappingFromXml Command

### Syntax

```
PlanMapping_AddPlanMappingFromXml filename errors
```

### Description

Use the AddPlanMappingFromXml command to add new plan mappings to the model using an input XML file. Plan mappings determine which plans are assigned to each Sales and Operations Planning views, and can be viewed from the Assign Plans page.

Plan mappings that are already present in the model are skipped.

### Parameters

Parameter	Description
<i>filename</i>	File name and path of the input XML file. If the path includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered when adding plan mappings.

### Example

The following example adds plan mappings to the model from the specified XML file:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Add the plan mappings
PlanMapping_AddPlanMappingsFromXml c:/scp/8.12/so-plan/plandata/Planmappings.xml
$errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## PlanMapping WritePlanMappingToXml Command

### Syntax

```
PlanMapping_WritePlanMappingToXml filename
```

### Description

Use the WritePlanMappingToXml command to write the current plan mappings to the specified XML file. Plan mappings determine which plans are assigned to each Sales and Operations Planning views, and can be viewed from the Assign Plans page.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where plan mappings are written. If the path includes blanks, then it must be enclosed in quotation marks.



## Example

The following example writes the model plan mappings to the file Mappings.xml:

```
PlanMappings_WritePlanMappingToXml c:/scp/8.12/so-plan/plandata/Mappings.xml
```

## ProductionHistory AddProductionHistory Command

### Syntax

```
ProductionHistory_AddProductionHistory key date values
```

### Description

Use the AddProductionHistory command to add Production history values to the model.

### Parameters

Parameter	Description
<i>key</i>	Production history key, consisting of a vector containing an item and location value.
<i>date</i>	Start date for the production history that you want to retrieve, in the format YYYY-MM-DD.
<i>values</i>	Vector containing the production history values that you want to add to the model. Sales and Operations Planning updates as many planning periods as there are values. For example, if the <i>values</i> parameter contains seven entries, a production history value is added to seven planning periods, starting with the specified start date.

## Example

The following example adds an production history value for the period 2005–09–13 for the item RES\_2 at location Toronto:

```
ProductionHistory_AddProductionHistory {RES_2 Toronto} 2005-09-13 {400}
```

## ProductionHistory AddProductionHistoryFromXml Command

### Syntax

```
ProductionHistory_AddProductionHistoryFromXml filename errors
```

### Description

Use the AddProductionHistoryFromXml command to add production history values from the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing production histories that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.

## Example

The following example adds production history values from the file ProductionHistory.xml:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Add production history values
ProductionHistory_AddProductionHistoryFromXml c:/scp/8.12/so-plan/plandata
/ProductionHistory.xml $errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## ProductionHistory DeleteAllProductionHistories Command

### Syntax

```
ProductionHistory_DeleteAllProductionHistories
```

### Description

Use the DeleteAllProductionHistories command to delete all production history values from the model.

## ProductionHistory DeleteProductionHistory Command

### Syntax

```
ProductionHistory_DeleteProductionHistory key
```

### Description

Use the DeleteProductionHistory command to remove the specified production history value from the model.

### Parameters

Parameter	Description
<i>key</i>	Production history key, consisting of a vector containing an item and location value.

## Example

The following example deletes the production history for the item RES\_2 at location Toronto:

```
ProductionHistory_DeleteProductionHistory {RES_2 Toronto}
```

## ProductionHistory WriteProductionHistoryToXml Command

### Syntax

```
ProductionHistory_WriteProductionHistoryToXml filename
```

### Description

Use the WriteProductionHistoryToXml command to write all production history values to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where production history values are written. If the file name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example writes all production history values to the XML file ProductionHistory.xml:

```
ProductionHistory_WriteProductionHistoryToXml c:/scp/8.12/so-plan/plandata
/ProductionHistory.xml
```

## ProductionPlan AddProduction Command

### Syntax

```
ProductionPlan_AddProduction itemCode locationCode
channelCode periodStartDate values
```

### Description

Use the AddProduction command to add a production plan time series to the model.

### Parameters

Parameter	Description
<i>itemCode</i>	Item code for the production plan entry being added to the model.
<i>locationCode</i>	Location code for the production plan entry being added to the model.
<i>channelCode</i>	Channel code for the production plan entry being added to the model.
<i>periodStartDate</i>	Date that specifies the starting period of the values parameter, in the format YYYY-MM-DD.
<i>values</i>	Vector containing new values associated with the corresponding item, location, channel and planning period. Sales and Operations Planning updates as many planning periods as there are values.

### Example

The following example adds a production plan value for Brown Ale\_341ml at the San Diego distribution center in the Hotels and Restaurants channel:

```
ProductionPlan_UpdateProduction 99011 "San Diego" 1001 2004-09-13 40
```

## ProductionPlan AddProductionFromXml Command

### Syntax

```
ProductionPlan_AddProductionFromXml filename errors skippedProductionCodes
```

### Description

Use the AddProductionFromXml command to add a production plan to the current cycle from the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing production plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the production plan update.
<i>skippedProductionCodes</i>	List object containing codes for those production plans that were not added to the model.

### Example

The following example adds production plan data to the model from the specified XML file:

```
# prepare list objects for receiving errors and skipped production plan codes
set errorsPtr [ListObject]
set skippedProductionCodesPtr [ListObject]
# Add production plan data
ProductionPlan_AddProductionFromXml c:/scp/8.12/so-plan/plandata/
Production.xml $errorsPtr $skippedProductionCodesPtr
# Return any errors or skipped production plan codes
ListObject_Get $errorsPtr
ListObject_Get $skippedProductionCodesPtr
```

## ProductionPlan AddProductionFromXmlToCycle Command

### Syntax

```
ProductionPlan_AddProductionFromXmlToCycle filename errors
skippedProductionCodes cycleName
```

### Description

Use the AddProductionFromXmlToCycle command to add a production plan to a specified cycle from an XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing production plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the production plan update.
<i>skippedProductionCodes</i>	List object containing codes for those production plans that were not added to the model.
<i>cycleName</i>	Name of the cycle to update. This parameter is optional; if cycleName is not specified, the current cycle is updated.

## Example

The following example adds production plan data to the specified planning cycle:

```
# prepare list objects for receiving errors and skipped production plan codes
set errorsPtr [ListObject]
set skippedProductionCodesPtr [ListObject]
# Add production plan data
ProductionPlan_AddProductionFromXmlToCycle c:/scp/8.12/so-plan/plandata
/Production.xml $errorsPtr $skippedProductionCodesPtr "Cycle 1"
# Return any errors or skipped production plan codes
ListObject_Get $errorsPtr
ListObject_Get $skippedProductionCodesPtr
```

## ProductionPlan ContainsId Command

### Syntax

```
ProductionPlan_ContainsId itemCode locationCode channelCode
```

### Description

Use the ContainsId command to check if the model contains a production plan entry for the specified item, location and channel.

### Parameters

Parameter	Description
<i>itemCode</i>	Item code of the production plan entry for which you're searching.
<i>locationCode</i>	Location code of the production plan entry for which you're searching.
<i>channelCode</i>	Channel code of the production plan entry for which you're searching.

### Returns

True, if the production plan entry exists; otherwise, False.

## Example

The following example checks for a production plan entry for Brown Ale\_341ml at the San Diego distribution center for the Hotels and Restaurants channel:

```
ProductionPlan_ContainsId 99011 "San Diego" 1001
```

This command returns:

```
1
```

## ProductionPlan DeleteAllProductions Command

### Syntax

```
ProductionPlan_DeleteAllProductions
```

### Description

Use the DeleteAllProductions command to remove all production plan data from the current cycle.

### Example

The following example deletes all production plan data from the current cycle:

```
ProductionPlan_DeleteAllProductions
```

## ProductionPlan DeleteAllProductionsForCycle Command

### Syntax

```
ProductionPlan_DeleteAllProductionsForCycle cycleName
```

### Description

Use the DeleteAllProductionsForCycle command to remove the specified cycle's production plan data from the model for the specified cycle.

### Parameters

Parameter	Description
<i>cycleName</i>	Name of the cycle all production plan data is removed from. If the cycle name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example removes all production plan data from the cycle March 2005:

```
ProductionPlan_DeleteAllProductionsForCycle "March 2005"
```

## ProductionPlan DeleteProduction Command

### Syntax

```
ProductionPlan_DeleteProduction itemCode  
locationCode channelCode
```

## Description

Use the DeleteProduction command to remove a specified a production plan entry from the current cycle.

## Parameters

Parameter	Description
<i>itemCode</i>	Item for the corresponding production plan entry to be deleted from the model.
<i>locationCode</i>	Location code for the corresponding production plan entry to be deleted from the model.
<i>channelCode</i>	Channel code for the corresponding production plan entry to be deleted from the model.

## Example

The following example deletes the production plan for Brown Ale\_341ml at the San Diego distribution center in the Hotels and Restaurants channel:

```
ProductionPlan_DeleteProduction 99011 "San Diego" 1001
```

## ProductionPlan GetProductionIds

### Syntax

```
Production_GetProductionIds supplyPlanName
```

### Description

Use the GetProductionIds command to retrieve the production identifiers from the specified supply plan. Each identifier consists of an item code and its corresponding location.

### Parameters

Parameter	Description
<i>supplyPlanName</i>	Supply Plan for which you want to retrieve production identifiers. If the supply plan name includes blanks, then it must be enclosed in quotation marks.

### Returns

A vector containing paired item codes and their corresponding location.

### Example

The following example returns production identifiers from the supply plan Q2 Supply Plan:

```
Production_GetProductionIds "Q2 Supply Plan"
```

## ProductionPlan StrictUpdateProductionFromXml Command

### Syntax

```
ProductionPlan_StrictUpdateProductionFromXml filename errors  
invalidProductionCodes
```

## Description

Use the `StrictUpdateProductionFromXml` command to do a strict update of your model production plan data using the specified XML file. If a production plan in the XML file does not exist in the model then that entry is ignored, and its added to the list of invalid production plan codes.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing production plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidProductionCodes</i>	List object containing production plans that were not added to the model. Only production plans that already exist in your model are updated.

## Example

The following example performs a strict update of the model using production plan data from the specified XML file:

```
# prepare list objects for receiving errors and invalid production plan codes
set errorsPtr [ListObject]
set invalidProductionCodesPtr [ListObject]
# Update production plan data
ProductionPlan_StrictUpdateProductionFromXml c:/scp/8.12/so-plan/plandata
/Production.xml $errorsPtr $invalidProductionCodesPtr
# Return any errors or invalid production plan codes
ListObject_Get $errorsPtr
ListObject_Get $invalidProductionCodesPtr
```

## ProductionPlan StrictUpdateProductionFromXmlToCycle Command

### Syntax

```
ProductionPlan_StrictUpdateProductionFromXmlToCycle filename errors
invalidProductionCodes cycleName
```

### Description

Use the `StrictUpdateProductionFromXmlToCycle` command to do a strict update of a cycle's production plan data using the specified XML file. If a production plan in the XML file does not exist in the model, then that entry is ignored, and its added to the list of invalid production plan codes.



## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing production plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidProductionCodes</i>	List object containing production plans that were not added to the model. Only production plans that already exist in your model are updated.
<i>cycleName</i>	Cycle to be updated. If the cycle name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example updates the model's production plan data for the specified planning cycle:

```
# prepare list objects for receiving errors and invalid production plan codes
set errorsPtr [ListObject]
set invalidProductionCodesPtr [ListObject]
# Update production plan data
ProductionPlan_StrictUpdateProductionFromXmlToCycle c:/scp/8.12/so-plan/
plandata/Production.xml $errorsPtr $invalidProductionCodesPtr "Cycle 1"
# Return any errors or invalid production plan codes
ListObject_Get $errorsPtr
ListObject_Get $invalidProductionCodesPtr
```

## ProductionPlan UpdateProduction Command

### Syntax

```
ProductionPlan_UpdateProduction itemCode locationCode channelCode values
```

### Description

Use the UpdateProduction command to update a production plan in the model. The passed in values do not have to overlap with the existing values in the model.

## Parameters

Parameter	Description
<i>itemCode</i>	Item code for the production plan entry to be updated.
<i>locationCode</i>	Location code for the production plan entry to be updated.
<i>channelCode</i>	Channel code for the production plan entry to be updated.
<i>periodStartDate</i>	Date that specifies the starting period for the new values, in the format YYYY-MM-DD.
<i>values</i>	Vector containing production plan values. The production plan is updated with these values, starting at the period start date. Sales and Operations Planning updates as many planning periods as there are values.

## Example

The following example updates the production plan for Brown Ale\_341ml at the San Diego distribution center for the Hotels and Restaurants channel, starting with the planning period 2004-09-13:

```
ProductionPlan_UpdateProduction 99011 "San Diego" 1001 2004-09-13 {40 65 20 40}
```

This command updates four planning periods.

## ProductionPlan UpdateProductionFromXml Command

### Syntax

```
ProductionPlan_UpdateProductionFromXml filename errors addedProductionCodes
```

### Description

Use the UpdateProductionFromXml command to update a production plan using data from an XML file. This command updates production plans that exist in the model, and adds those production plans that are not present.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing production plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedProductionCodes</i>	List object containing production plans that were added to the model.

## Example

The following example updates the model using production plan data from the specified XML file:

```
# prepare list objects for receiving errors and added production plan codes
set errorsPtr [ListObject]
set addedProductionCodesPtr [ListObject]
# Update production plan data
ProductionPlan_UpdateProductionFromXml c:/scp/8.12/so-plan/plandata
```

```

/Production.xml $errorsPtr $addedProductionCodesPtr
# Return any errors or added production plan codes
ListObject_Get $errorsPtr
ListObject_Get $addedProductionCodesPtr

```

## ProductionPlan UpdateProductionFromXmlToCycle Command

### Syntax

```

ProductionPlan_UpdateProductionFromXmlToCycle filenameerrors
addedProductionCodes

```

### Description

Use the UpdateProductionFromXmlToCycle command to update a production plan for a specific cycle using data from an XML file. This command updates production plans that exist in the model, and adds those production plans that are not present.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing production plan data used to update your model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedProductionCodes</i>	List object containing production plans that were added to the model.
<i>cycleName</i>	Cycle to be updated.

### Example

The following example updates the model for a planning cycle using production plan data from the specified XML file:

```

# prepare list objects for receiving errors and added production plan codes
set errorsPtr [ListObject]
set addedProductionCodesPtr [ListObject]
# Update production plan data
ProductionPlan_UpdateProductionFromXmlToCycle c:/scp/8.12/so-plan/plandata
/Production.xml $errorsPtr $addedProductionCodesPtr "Cycle 1"
# Return any errors or added production plan codes
ListObject_Get $errorsPtr
ListObject_Get $addedProductionCodesPtr

```

## ProductionPlan WriteProductionToXml Command

### Syntax

```

ProductionPlan_WriteProductionToXml filename

```

### Description

Use the WriteProductionToXml command to write the production plan to the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where the production plan is written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes the production plan to the file Production.xml:

```
ProductionPlan_WriteProductionToXml c:/scp/8.12/so-plan/plandata/Production.xml
```

## ProductionPlan WriteProductionToXmlFromCycle Command

### Syntax

```
ProductionPlan_WriteProductionToXmlFromCycle filename cycleName
```

### Description

Use the WriteProductionToXml command to write the production plan for a specific cycle to an XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where the production plan is written. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>cycleName</i>	Cycle from which you want to write production data. If the cycle name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes the production plan to the file Production.xml for the cycle March 2005:

```
ProductionPlan_WriteProductionToXmlFromCycle c:/scp/8.12/so-plan/plandata
/Production.xml "March 2005"
```

## Resources AddPlannedCapacity Command

### Syntax

```
Resources_AddPlannedCapacity supplyPlanName resourceCode periodStartDate
values
```

### Description

Use the AddPlannedCapacity command to add capacity to the specified resource in the specified supply plan.

## Parameters

Parameter	Description
<i>supplyPlanName</i>	Supply plan for which you want to add planned capacity. If the supply plan name includes blanks, then it must be enclosed in quotation marks.
<i>resourceCode</i>	Code for the resource to which you want to add planned capacity.
<i>periodStartDate</i>	Start date for the period in which you want to add planned capacity, in the format YYYY-MM-DD.
<i>values</i>	Vector of planned capacity values.

## Example

The following example adds planned capacity to the resource RES2:

```
# Add planned capacity to the resource RES2
Resources_AddPlannedCapacity "Q4 Projected Supply" RES2 2005-06-18
{90 90 120 140 90}
```

## Resources AddPlannedCapacityFromXml Command

### Syntax

```
Resources_AddPlannedCapacityFromXml filename errors skippedPlannedCapacityCodes
```

### Description

Use the AddPlannedCapacityFromXml command to add planned capacity from the specified XML file. Planned capacity entries that already exist in the model are skipped.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing planned capacity to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>skippedPlannedCapacityCodes</i>	List object containing codes for planned capacity that were not added to the model.

## Example

The following example adds planned capacity to the model from the specified XML file:

```
# prepare list objects for receiving errors and skipped capacity codes
set errorsPtr [ListObject]
set skippedPlannedCapacityCodesPtr [ListObject]
# Add planned capacity
Resources_AddPlannedCapacityFromXml c:/scp/8.12/so-plan/plandata/
capacity.xml $errorsPtr $skippedPlannedCapacityCodesPtr
# Return any errors or skipped capacity codes
ListObject_Get $errorsPtr
```

```
ListObject_Get $skippedPlannedCapacityCodesPtr
```

## Resources AddRequiredCapacity Command

### Syntax

```
Resources_AddRequiredCapacity demandPlanName resourceCode periodStartDate values
```

### Description

Use the AddRequiredCapacity command to add required capacity for the specified resource to the specified demand plan.

### Parameters

Parameter	Description
<i>demandPlanName</i>	Demand plan to which you want to add required capacity. If the demand plan name contains blanks, then it must be enclosed in quotation marks.
<i>resourceCode</i>	Code for the resource to which you want to add capacity. If the resource name contains blanks, then it must be enclosed in quotation marks.
<i>periodStartDate</i>	Start date for the period in which you want to add required capacity, in the format YYYY-MM-DD.
<i>values</i>	Vector of required capacity values.

### Example

The following example adds required capacity to the resource RES2:

```
# Add required capacity to the resource RES2
Resources_AddRequiredCapacity "Q4 Projected Demand" RES2 2005-06-18
{105 90 110 160 90}
```

## Resources AddRequiredCapacityFromXml Command

### Syntax

```
Resources_AddRequiredCapacityFromXml filename errors skippedRequiredCapacityCodes
```

### Description

Use the AddRequiredCapacityFromXml command to add required capacity from the specified XML file. Required capacity entries that already exist in the model are skipped.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing required capacity that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>skippedRequiredCapacityCodes</i>	List object containing codes for required capacity that were not added to the model.

## Example

The following example adds required capacity to the model from the specified XML file:

```
# prepare list objects for receiving errors and skipped capacity codes
set errorsPtr [ListObject]
set skippedRequiredCapacityCodesPtr [ListObject]
# Add required capacity
Resources_AddRequiredCapacityFromXml c:/scp/8.12/so-plan/plandata/
requiredcapacity.xml $errorsPtr $skippedRequiredCapacityCodesPtr
# Return any errors or skipped capacity codes
ListObject_Get $errorsPtr
ListObject_Get $skippedRequiredCapacityCodesPtr
```

## Resources AddRequiredCapacityFromXmlToCycle Command

### Syntax

```
Resources_AddRequiredCapacityFromXmlToCycle filename errors skippedRequiredCapacityCodes
cycleName
```

### Description

Use the AddRequiredCapacityFromXmlToCycle command to add required capacity from the specified XML file. Required capacity entries that already exist in the model are skipped.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing required capacity that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>skippedRequiredCapacityCodes</i>	List object containing codes for required capacity that were not added to the model.
<i>cycleName</i>	Cycle to which you want to add required capacity. If the cycle name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example adds required capacity from an XML file to the specified cycle:

```
# prepare list objects for receiving errors and skipped required capacity codes
set errorsPtr [ListObject]
set skippedRequiredCapacityPtr [ListObject]
# Add required capacity data
Resources_AddRequiredCapacityFromXmlToCycle c:/scp/8.12/so-plan/plandata
/requiredcapacity.xml $errorsPtr $skippedDemandPlansPtr "Cycle 1"
# Return any errors or skipped capacity codes
ListObject_Get $errorsPtr
ListObject_Get $skippedDemandPlansPtr
```

## Resources AddResource Command

### Syntax

```
Resources_AddResource code name
```

### Description

Use the AddResource command to add a resource to the Sales and Operations Planning model.

### Parameters

Parameter	Description
<i>code</i>	Code of the resource that you add to the model. If the resource code contains blanks, then it must be enclosed in quotation marks.
<i>name</i>	Name of the resource that you want to add to the model. If the resource name contains blanks, then it must be enclosed in quotation marks.

### Example

The following example adds the resource Resource\_2 to the model:

```
Resources_AddResource RES2 Resource_2
```

## Resources AddResourceMaximumCapacitiesFromXml Command

### Syntax

```
Resources_AddResourceMaximumCapacitiesFromXml filename errors
```

### Description

Use the AddResourceMaximumCapacitiesFromXml command to add resource maximum capacities from the specified XML file.



## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing resource maximum capacities that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.

## Example

The following example adds resource maximum capacities from the file ResourceMaximumCapacities.xml:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Add maximum resource capacities
Resources_AddResourceMaximumCapacitiesFromXml c:/scp/8.12/so-plan/plandata
/ResourceMaximumCapacities.xml $errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## Resources AddResourceMaximumCapacity Command

### Syntax

```
Resources_AddResourceMaximumCapacity resourceCode period maximumCapacity
```

### Description

Use the AddResourceMaximumCapacity command to add maximum capacity to the specified resource.

### Parameters

Parameter	Description
<i>resourceCode</i>	Code for the resource to which you want to add capacity. If the resource name contains blanks, then it must be enclosed in quotation marks.
<i>period</i>	Start date for the period in which you want to add required capacity, in the format YYYY-MM-DD.
<i>maximumCapacity</i>	New maximum capacity for the specified resource.

## Example

The following example adds maximum capacity to the resource RES2 for the planning period beginning on March 1st, 2005:

```
Resources_AddResourceMaximumCapacity RES2 2005-03-01 130
```

## Resources AddResourcesFromXml Command

### Syntax

```
Resources_AddResourcesFromXml filename errorsskippedResourceCodes
```

## Description

Use the `AddResourcesFromXml` command to add resources to the model from the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing resources that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>skippedResourceCodes</i>	List object containing resources that were not added to the model.

## Example

The following example adds resources from the file `resources.xml`:

```
# prepare list objects for receiving errors and skipped resource codes
set errorsPtr [ListObject]
set skippedRequiredResourceCodesPtr [ListObject]
# Add required capacity data
Resources_AddResourcesFromXml c:/scp/8.12/so-plan/plandata/resources.xml
$errorsPtr $skippedRequiredResourceCodesPtr
# Return any errors or skipped resource codes
ListObject_Get $errorsPtr
ListObject_Get $skippedRequiredResourceCodesPtr
```

## Resources CheckExists Command

### Syntax

```
Resources_CheckExists resourceCode
```

### Description

Use the `CheckExists` command to determine if the specified resource exists in the model.

### Parameters

Parameter	Description
<i>resourceCode</i>	Code for the resource that you want to verify exists in the model. If the resource name contains blanks, then it must be enclosed in quotation marks.

## Example

The following example checks the model for the resource `RES_2`:

```
Resources_CheckExists RES_2
```

## Resources DeletePlannedCapacity Command

### Syntax

```
Resources_DeletePlannedCapacity supplyPlanName
```

### Description

Use the DeletePlannedCapacity command to delete all planned capacities in the specified supply plan.

### Parameters

Parameter	Description
<i>supplyPlanName</i>	Supply plan for which you want to delete planned capacity. If the supply plan name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example deletes all planned capacity values from the supply plan Projected Supply:

```
Resources_DeletePlannedCapacity "Projected Supply"
```

## Resources DeleteAllPlannedCapacityForCycle Command

### Syntax

```
Resources_DeleteAllPlannedCapacityForCycle cycleName supplyPlanName
```

### Description

Use the DeleteAllPlannedCapacity command to delete all planned capacities in the specified cycle and supply plan.

### Parameters

Parameter	Description
<i>cycleName</i>	Cycle from which you want to delete planned capacity. If the cycle name includes blanks, then it must be enclosed in quotation marks.
<i>supplyPlanName</i>	Supply plan for which you want to delete planned capacity. If the supply plan name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example deletes all planned capacity values from the supply plan Projected Supply in the planning cycle Q4 2005:

```
Resources_DeleteAllPlannedCapacityForCycle "Q4 2005" "Projected Supply"
```

## Resources DeleteAllRequiredCapacities Command

### Syntax

```
Resources_DeleteAllRequiredCapacities demandPlanName
```

## Description

Use the DeleteAllRequiredCapacities command to delete all required capacities from the specified demand plan and the current cycle.

## Parameters

Parameter	Description
<i>demandPlanName</i>	Demand plan from which you want to delete required capacity. If the demand plan name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example deletes all required capacities from the demand plan Projected Demand:

```
Resources_DeleteAllRequiredCapacities "Projected Demand"
```

## Resources DeleteAllRequiredCapacitiesForCycle Command

### Syntax

```
Resources_DeleteAllRequiredCapacitiesForCycle cycleName demandPlanName
```

### Description

Use the DeleteAllRequiredCapacitiesForCycle command to delete all required capacities from the specified demand plan and cycle.

### Parameters

Parameter	Description
<i>cycleName</i>	Cycle from which you want to delete required capacity. If the cycle name includes blanks, then it must be enclosed in quotation marks.
<i>demandPlanName</i>	Demand plan from which you want to delete required capacity. If the demand plan name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example deletes all required capacities from the demand plan Projected Demand and planning cycle Q4 2005:

```
Resources_DeleteAllRequiredCapacitiesForCycle "Projected Demand" "Q4 2005"
```

## Resources DeleteAllResources Command

### Syntax

```
Resources_DeleteAllResources
```

### Description

Use the DeleteAllResources command to delete all resources from the model.

## Example

The following example deletes all resources from the Sales and Operations Planning model:

```
Resources_DeleteAllResources
```

## Resources DeletePlannedCapacity Command

### Syntax

```
Resources_DeletePlannedCapacity supplyPlanName resourceCode
```

### Description

Use the DeletePlannedCapacity command to delete planned capacity values from the specified resource and supply plan.

### Parameters

Parameter	Description
<i>supplyPlanName</i>	Supply plan from which you want to delete planned capacity values. If the supply plan name contains blanks, then it must be enclosed in quotation marks.
<i>resourceCode</i>	Code for the resource from which you want to delete planned capacity.

## Example

The following example deletes all planned capacity values from the resource RES\_2 in the supply plan Projected Supply:

```
Resources_DeletePlannedCapacity "Projected Supply" RES_2
```

## Resources DeleteRequiredCapacity Command

### Syntax

```
Resources_DeleteRequiredCapacity demandPlanName resourceCode
```

### Description

Use the DeleteRequiredCapacity command to delete required capacity values from the specified resource and demand plan.

### Parameters

Parameter	Description
<i>demandPlanName</i>	Demand Plan from which you want to delete required capacity. If the demand plan name contains blanks, then it must be enclosed in quotation marks.
<i>resourceCode</i>	Code for the resource from which you want to delete required capacity.

## Example

The following example deletes all required capacity values from the resource RES\_3 and demand plan Projected Demand:

```
Resources_DeleteRequiredCapacity "Projected Demand" RES_3
```

## Resources DeleteResource Command

### Syntax

```
Resources_DeleteResource resourceCode
```

### Description

Use the DeleteResource command to remove the specified resource from the model.

### Parameters

Parameter	Description
<i>resourceCode</i>	Code for the resource that you want to remove from the model.

### Example

The following example deletes the resource RES\_2 from the model:

```
Resources_DeleteResource RES_2
```

## Resources GetPlannedCapacityValues Command

### Syntax

```
Resources_GetPlannedCapacityValues supplyPlanName resourceCode
```

### Description

Use the GetPlannedCapacityValues command to retrieve a resource's capacity values in the specified supply plan. Each identifier consists of an item code and its corresponding location.

### Parameters

Parameter	Description
<i>supplyPlanName</i>	Supply plan from which you want to retrieve planned capacity identifiers. If the supply plan name includes blanks, then it must be enclosed in quotation marks.
<i>resourceCode</i>	Code for the resource for which you want to retrieve planned capacity values.

### Returns

A vector containing paired item codes and their corresponding planned capacity.

### Example

The following example returns capacity values for item 9015 from the supply plan Q2 Supply Plan:

```
Resources_GetPlannedCapacityValuess "Q2 Supply Plan" 9015
```

## Resources GetResourceCodes Command

### Syntax

```
Resources_GetResourceCodes
```

### Description

Use the GetResourceCodes command to retrieve all resource codes from the model.

### Example

The following example retrieves all resources codes from the Sales and Operations Planning model:

```
Resources_GetResourceCodes
```

## Resources GetResourceProperties Command

### Syntax

```
Resources_GetResourceProperties resourceCode
```

### Description

Use the GetResourceProperties command to retrieve properties for the specified resource.

### Parameters

Parameter	Description
<i>resourceCode</i>	Resource for which you want to retrieve properties. If the resource code contains blanks, then it must be enclosed in quotation marks.

### Returns

A list of properties for the specified resource.

### Example

The following example retrieves properties for the resource RES\_2:

```
Resources_GetResourceProperties RES_2
```

## Resources GetResourcePropertyNames Command

### Syntax

```
Resources_GetResourcePropertyNames
```

### Description

Use the GetResourcePropertyNames command to retrieve the names of all resources that are stored in the model.

### Returns

A list of resource names.

## Example

The following example returns a list of resources in the Sales and Operations Planning model:

```
Resources_GetResourcePropertyNames
```

## Resources PlannedCapacityContainsId Command

### Syntax

```
Resources_PlannedCapacityContainsId supplyPlanName resourceCode
```

### Description

Use the PlannedCapacityContainsId command to determine if the specified resource contains planned capacity values.

### Parameters

Parameter	Description
<i>supplyPlanName</i>	Supply plan for which you want to check for capacity values. If the supply plan name includes blanks, then it must be enclosed in quotation marks.
<i>resourceCode</i>	Resource that you want to check for planned capacity values.

### Returns

True if the specified resource contains capacity values; otherwise false.

### Example

The following example checks the resource RES\_2 for planned capacity values in the supply plan Projected Supply :

```
Resource_PlannedCapacityContainsId "Projected Supply" RES_2
```

## Resources RequiredCapacityContainsId Command

### Syntax

```
Resources_RequiredCapacityContainsId demandPlanName resourceCode
```

### Description

Use the RequiredCapacityContainsId command to determine if the specified resource contains required capacity values.



## Parameters

Parameter	Description
<i>demandPlanName</i>	Demand plan for which you want to check for capacity values. If the demand plan name includes blanks, then it must be enclosed in quotation marks.
<i>resourceCode</i>	Resource that you want to check for planned capacity values.

## Returns

True if the specified resource contains capacity values; otherwise false.

## Example

The following example checks the resource RES\_2 for planned capacity values in the demand plan Projected Demand :

```
Resources_RequiredCapacityContainsId "Projected Demand" RES_2
```

## Resources SetResourceProperties Command

### Syntax

```
Resources_SetResourceProperties code properties
```

### Description

### Parameters

Parameter	Description
<i>code</i>	Code for the resource to which you want to set properties. If the resource name contains blanks, then it must be enclosed in quotation marks.
<i>properties</i>	A vector of paired properties that you want to apply to the specified resource.

## Example

The following example sets the item color property to Blue for the resource RES\_2:

```
Resources_SetResourceProperty RES_2 {ItemColor Blue}
```

## Resources SetResourcePropertyNames Command

### Syntax

```
Resources_SetResourcePropertyNames propertyNames
```

### Description

Use the SetResourcePropertyNames command to define the valid resource names in the model.

## Parameters

Parameter	Description
<i>propertyName</i>	A vector of property names. Any property name that already exists in the model is not updated.

## Example

The following example adds the properties SKU and, Item\_Color to the model:

```
Resources_SetResourcePropertyNames {SKU Item_Color}
```

## Resources StrictUpdatePlannedCapacityFromXml Command

### Syntax

```
Resources_StrictUpdatePlannedCapacityFromXml filename errors invalidPlannedCapacityCodes
```

### Description

Use the StrictUpdatePlannedCapacityFromXml command to perform a strict update of planned capacity from the specified XML file. If a planned capacity value from the XML file does not exist in the model, then that entry is ignored and added to the list of invalid planned capacity ids.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing planned capacity values that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidPlannedCapacityCodes</i>	List object containing planned capacity codes that were not added to the model. Only planned capacity codes that already exist in your model are updated.

## Example

The following example performs a strict update of the model using planned capacity codes from the specified XML file:

```
# prepare list objects for receiving errors and invalid planned capacity codes
set errorsPtr [ListObject]
set invalidPlannedCapacityCodesPtr [ListObject]
# Update planned capacity codes
Resources_StrictUpdatePlannedCapacityFromXml c:/scp/8.12/so-plan/plandata
/plannedcapacity.xml $errorsPtr $invalidPlannedCapacityCodesPtr
# Return any errors or invalid production plan codes
ListObject_Get $errorsPtr
ListObject_Get $invalidPlannedCapacityCodesPtr
```

## Resources StrictUpdatePlannedCapacityFromXmlToCycle command

### Syntax

```
Resources_StrictUpdatePlannedCapacityFromXmlToCycle filename errors
invalidPlannedCapacityCodes cycleName
```

### Description

Use the StrictUpdatePlannedCapacityFromXml command to perform a strict update of planned capacity from the specified XML file to the specified planning cycle. If a planned capacity value from the XML file does not exist in the model, then that entry is ignored and added to the list of invalid planned capacity ids.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing planned capacity values that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidPlannedCapacityCodes</i>	List object containing planned capacity codes that were not added to the model. Only planned capacity codes that already exist in your model are updated.
<i>cycleName</i>	Cycle to which you want to update planned capacity codes. If the cycle name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example performs a strict update of the model using planned capacity codes from the specified XML file:

```
# prepare list objects for receiving errors and skipped planned capacity codes
set errorsPtr [ListObject]
set skippedPlannedCapacityPtr [ListObject]
# Add planned capacity data
Resources_AddPlannedCapacityFromXmlToCycle c:/scp/8.12/so-plan/plandata
/plannedcapacity.xml $errorsPtr $skippedPlannedCapacityPtr "Cycle 1"
# Return any errors or skipped capacity codes
ListObject_Get $errorsPtr
ListObject_Get $skippedPlannedCapacityPtr
```

## Resources StrictUpdateRequiredCapacityFromXml Command

### Syntax

```
Resources_StrictUpdateRequiredCapacityFromXml filenameerrors invalidRequiedCapacityCodes
```

### Description

Use the StrictUpdateRequiredCapacityFromXml command to perform a strict update of required capacity from the specified XML file. If a required capacity value from the XML file does not exist in the model, then that entry is ignored and added to the list of invalid required capacity ids.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing required capacity values that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidRequiredCapacityCodes</i>	List object containing required capacity codes that were not added to the model. Only required capacity codes that already exist in your model are updated.

## Example

The following example performs a strict update of the model using required capacity codes from the specified XML file:

```
# prepare list objects for receiving errors and invalid required capacity codes
set errorsPtr [ListObject]
set invalidRequiredCapacityCodesPtr [ListObject]
# Update required capacity codes
Resources_StrictUpdateRequiredCapacityFromXml c:/scp/8.12/so-plan/plandata
/requiredcapacity.xml $errorsPtr $invalidRequiredCapacityCodesPtr
# Return any errors or invalid required capacity codes
ListObject_Get $errorsPtr
ListObject_Get $invalidRequiredCapacityCodesPtr
```

## Resources StrictUpdateRequiredCapacityFromXmlToCycle Command

### Syntax

```
Resources_StrictUpdateRequiredCapacityFromXmlToCycle filenameerrors
invalidRequiredCapacityCodes cycleName
```

### Description

Use the StrictUpdateRequiredCapacityFromXmlToCycle command to perform a strict update of required capacity from the specified XML file to the specified planning cycle. If a required capacity value from the XML file does not exist in the model, then that entry is ignored and added to the list of invalid required capacity ids.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing required capacity values that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidRequiredCapacityCodes</i>	List object containing required capacity codes that were not added to the model. Only required capacity codes that already exist in your model are updated.
<i>cycleName</i>	Cycle to which you want to update required capacity codes. If the cycle name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example performs a strict update of the model using required capacity codes from the specified XML file:

```
# prepare list objects for receiving errors and skipped required capacity codes
set errorsPtr [ListObject]
set skippedRequiredCapacityPtr [ListObject]
# Add required capacity data
Resources_AddRequiredCapacityFromXmlToCycle c:/scp/8.12/so-plan/plandata
/requiredcapacity.xml $errorsPtr $skippedRequiredCapacityPtr "Cycle 1"
# Return any errors or skipped required capacity codes
ListObject_Get $errorsPtr
ListObject_Get $skippedRequiredCapacityPtr
```

## Resources StrictUpdateResourcesFromXml Command

### Syntax

```
Resources_StrictUpdateResourcesFromXml filename errors invalidResourceCodes
```

### Description

Use the StrictUpdateResourcesFromXml command to perform a strict update of resources from the specified XML file. If a resource from the XML file does not exist in the model, then that entry is ignored and added to the list of invalid resources.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing resources that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidResourceCodes</i>	List object containing resources that were not added to the model. Only resources that already exist in your model are updated.

## Example

The following example performs a strict update of the model with resources from the specified XML file:

```
# prepare list objects for receiving errors and invalid resources
set errorsPtr [ListObject]
set invalidResourceCodesPtr [ListObject]
# Update resource codes
Resources_StrictUpdateResourcesFromXml c:/scp/8.12/so-plan/plandata/
resources.xml $errorsPtr $invalidResourceCodesPtr
# Return any errors or invalid required capacity codes
ListObject_Get $errorsPtr
ListObject_Get $invalidResourceCodesPtr
```

## Resources UpdatePlannedCapacity Command

### Syntax

```
Resources_UpdatePlannedCapacity supplyPlanName resourceCode periodStartDate values
```

### Description

Use the UpdatePlannedCapacity command to update a resource's planned capacity in specified supply plan. The passed in values do not have to overlap with the existing values in the model.

### Parameters

Parameter	Description
<i>supplyPlanName</i>	Supply plan for which you want to update planned capacity. If the supply plan name includes blanks, then it must be enclosed in quotation marks.
<i>resourceCode</i>	Code for the resource to which you want to update planned capacity. If the resource name contains blanks, then it must be enclosed in quotation marks.
<i>periodStartDate</i>	Start date for the period in which you want to update planned capacity, in the format YYYY-MM-DD.
<i>values</i>	Vector containing planned capacity values. Sales and operations updates as many planning periods as there are values, starting with the specified period start date.

### Example

The following example updates the resource RES\_2 required capacity values in the demand plan Forecast Demand:

```
Resources_UpdatePlannedCapacity Forecast Demand RES_2 2005-07-15
{100 100 125 120 100}
```

## Resources UpdatePlannedCapacityFromXml Command

### Syntax

```
Resources_UpdatePlannedCapacityFromXml filename errors addedPlannedCapacityCodes
```

## Description

Use the `UpdatePlannedCapacityFromXml` command to update a resource's planned capacity in specified supply plan from the specified XML file. The passed in values do not have to overlap with the existing values in the model.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing planned capacity values that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedPlannedCapacityCodes</i>	List object containing codes for planned capacity values that were added to the model.

## Example

The following example performs an update of the model with resources from the specified XML file:

```
# prepare list objects for receiving errors and added planned capacity codes
set errorsPtr [ListObject]
set addedPlannedCapacityCodesPtr [ListObject]
# Update planned capacity codes
Resources_UpdatePlannedCapacityFromXml c:/scp/8.12/so-plan/plandata
/plannedcapacity.xml $errorsPtr $addedPlannedCapacityCodesPtr
# Return any errors or invalid required capacity codes
ListObject_Get $errorsPtr
ListObject_Get $addedPlannedCapacityCodesPtr
```

## Resources UpdatePlannedCapacityFromXmlToCycle Command

### Syntax

```
Resources_UpdatePlannedCapacityFromXmlToCycle filename errors
addedPlannedCapacityCodes cycleName
```

### Description

Use the `UpdatePlannedCapacityFromXmlToCycle` command to update a resource's planned capacity for the specified planning cycle from the specified XML file. The passed in values do not have to overlap with the existing values in the model.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing planned capacities that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedPlannedCapacityCodes</i>	List object containing codes for planned capacity values that were added to the model.
<i>cycleName</i>	Cycle to which you want to add planned capacity. If the cycle name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example performs an update of the model using planned capacity codes from the specified XML file:

```
# prepare list objects for receiving errors and added planned capacity codes
set errorsPtr [ListObject]
set addedPlannedCapacityPtr [ListObject]
# Add planned capacity data
Resources_UpdatePlannedCapacityFromXmlToCycle c:/scp/8.12/so-plan/plandata
/plannedcapacity.xml $errorsPtr $addedPlannedCapacityPtr "Cycle 1"
# Return any errors or skipped required capacity codes
ListObject_Get $errorsPtr
ListObject_Get $addedPlannedCapacityPtr
```

## Resources UpdateRequiredCapacity Command

### Syntax

```
Resources_UpdateRequiredCapacity demandPlanName resourceCode periodStartDate values
```

### Description

Use the UpdateRequiredCapacity command to update a resource's required capacity in specified demand plan. The passed in values do not have to overlap with the existing values in the model.



## Parameters

Parameter	Description
<i>demandPlanName</i>	Demand plan to which you want to update required capacity. If the demand plan name includes blanks, then it must be enclosed in quotation marks.
<i>resourceCode</i>	Code for the resource to which you want to update required capacity. If the resource name contains blanks, then it must be enclosed in quotation marks.
<i>periodStartDate</i>	Start date for the period in which you want to update required capacity, in the format YYYY-MM-DD.
<i>values</i>	Vector containing required capacity values. Sales and operations updates as many planning periods as there are values, starting with the specified period start date.

## Example

The following example updates the resource RES\_2 required capacity values in the demand plan Forecast Demand:

```
Resources_UpdateRequiredCapacity Forecast Demand RES_2 2005-07-15
{90 90 120 120 90}
```

## Resources UpdateRequiredCapacityFromXml Command

### Syntax

```
Resources_UpdateRequiredCapacityFromXml filename errors addedRequiredCapacityCodes
```

### Description

Use the UpdateRequiredCapacityFromXml command to update a resource's required capacity in specified supply plan from the specified XML file. The passed in values do not have to overlap with the existing values in the model.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing required resource capacities that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedRequiredCapacityCodes</i>	List object containing codes for required capacity values that were added to the model.

## Example

The following example performs an update of the model with resources from the specified XML file:

```
# prepare list objects for receiving errors and updated required capacity codes
set errorsPtr [ListObject]
set updatedRequiredCapacityCodesPtr [ListObject]
# Update required capacity codes
```

```
Resources_UpdateRequiredCapacityFromXml c:/scp/8.12/so-plan/plandata
/requiredcapacity.xml $errorsPtr $updatedRequiredCapacityCodesPtr
# Return any errors or invalid required capacity codes
ListObject_Get $errorsPtr
ListObject_Get $updatedRequiredCapacityCodesPtr
```

## Resources UpdateRequiredCapacityFromXmlToCycle Ccommand

### Syntax

```
Resources_UpdateRequiredCapacityFromXmlToCycle filename errors
addedRequiredCapacityCode cycleName
```

### Description

Use the UpdateRequiredCapacityFromXmlToCycle command to update a resource's required capacity for the specified planning cycle from the specified XML file. The passed in values do not have to overlap with the existing values in the model.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing required capacity values that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedRequiredCapacityCodes</i>	List object containing codes for required capacity values that were added to the model.
<i>cycleName</i>	Cycle to which you want to add required capacity. If the cycle name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example performs an update of the model using required capacity codes from the file RequiredCapacity.xml:

```
# prepare list objects for receiving errors and added required capacity codes
set errorsPtr [ListObject]
set addedRequiredCapacityPtr [ListObject]
# Add required capacity data
Resources_UpdateRequiredCapacityFromXmlToCycle c:/scp/8.12/so-plan/plandata
/RequiredCapacity.xml $errorsPtr $addedRequiredCapacityPtr "Cycle 1"
# Return any errors or skipped required capacity codes
ListObject_Get $errorsPtr
ListObject_Get $addedRequiredCapacityPtr
```

## Resources UpdateResourcesFromXml Command

### Syntax

`Resources_UpdateResourcesFromXml filename errors addedResourceCodes`

### Description

Use the UpdateResourcesFromXml command to update resources in the model with values from the specified XML file. The passed in values do not have to overlap with the existing values in the model.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing resources that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedResourceCodes</i>	List object containing codes for resources that were added to the model.

### Example

The following example updates the model with resources from the file Resources.xml:

```
# prepare list objects for receiving errors and added resource codes
set errorsPtr [ListObject]
set addedResourceCodesPtr [ListObject]
# Update required capacity codes
Resources_UpdateResourcesFromXml c:/scp/8.12/so-plan/plandata/Resources.xml
$errorsPtr $addedResourceCodesPtr
# Return any errors or added resource codes
ListObject_Get $errorsPtr
ListObject_Get $addedResourceCodesPtr
```

## Resources WritePlannedCapacityToXml Command

### Syntax

`Resources_WritePlannedCapacityToXml filename supplyPlanName`

### Description

Use the WritePlannedCapacityToXml command write a supply plan's planned capacity values to the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where the planned capacity values are written. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>supplyPlanName</i>	Supply plan from which you want to write planned capacity. If the supply plan name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes planned capacity values from the supply plan Projected Supply to the file PlannedCapacity.xml:

```
Resources_WritePlannedCapacityToXml c:/scp/8.12/so-plan/plandata/
PlannedCapacity.xml "Projected Supply"
```

## Resources WritePlannedCapacityToXmlFromCycle command

### Syntax

```
Resources_WritePlannedCapacityToXmlFromCycle filename cycleNamesupplyPlanName
```

### Description

Use the WritePlannedCapacityToXml command write a supply plan's planned capacity values for the specified planning cycle to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where the planned capacity values are written. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>cycleName</i>	Cycle from which you want to write planned capacity values. If the cycle name includes blanks, then it must be enclosed in quotation marks.
<i>supplyPlanName</i>	Supply plan from which you want to write planned capacity values. If the supply plan name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes planned capacity values from the supply plan Projected Supply to the file PlannedCapacity.xml:

```
Resources_WritePlannedCapacityToXmlFromCycle PlannedCapacity.xml "Q4 2005"
"Projected Supply"
```

## Resources WriteRequiredCapacityToXml Command

### Syntax

```
Resources_WriteRequiredCapacityToXml filename supplyPlanName
```

## Description

Use the WriteRequiredCapacityToXml command to write a supply plan's required capacity values to the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where the required capacity values are written. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>supplyPlanName</i>	Supply plan from which you want to write required capacity values. If the supply plan name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes required capacity values from the supply plan Projected Supply to the file PlannedCapacity.xml:

```
Resources_WriteRequiredCapacityToXml c:/scp/8.12/so-plan/plandata/
RequiredCapacity.xml "Projected Supply"
```

## Resources WriteRequiredCapacityToXmlFromCycle Command

### Syntax

```
Resources_WriteRequiredCapacityToXmlFromCycle filename cycleNamesupplyPlanName
```

### Description

Use the WriteRequiredCapacityToXmlFromCycle command to write a supply plan's required capacity values for the specified planning cycle to an XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing required capacity values that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>cycleName</i>	Cycle from which you want to write required capacity values. If the cycle name includes blanks, then it must be enclosed in quotation marks.
<i>supplyPlanName</i>	Supply plan from which you want to write required capacity values. If the supply plan name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes required capacity values from the supply plan Projected Supply to the file PlannedCapacity.xml

```
Resources_WriteRequiredCapacityToXmlFromCycle c:/scp/8.12/so-plan/plandata
/RequiredCapacity.xml "Q4 2005" "Projected Supply"
```

## Resources WriteResourceMaximumCapacitiesToXml Command

### Syntax

```
Resources_WriteResourceMaximumCapacitiesToXml filename
```

### Description

Use the WriteResourceMaximumCapacitiesToXml command to write the model's maximum capacity values to the specified Xml file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where you want to write maximum capacities. If the file name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example writes maximum capacity values for all resources in the model to the file MaximumCapacities.xml:

```
Resources_WriteResourceMaximumCapacitiesToXml c:/scp/8.12/so-plan/plandata
/MaximumCapacities.xml
```

## Resources WriteResourcesToXml Command

### Syntax

```
Resources_WriteResourcesToXml filename
```

### Description

Use the WriteResourcesToXml command to write a list of all resources found in the model to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where resource data is written. If the file name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example writes all resources in the model to the file Resources.xml:

```
Resources_WriteResourcesToXml c:/scp/8.12/so-plan/plandata/Resources.xml
```

## SalesHistory AddSalesHistoryFromXml Command

### Syntax

```
SalesHistory_AddSalesHistoryFromXml filename errors UnitSkippedCodes PriceSkippedCodes
```

## Description

Use the `AddSalesHistoryFromXml` command to add new sales history values from the specified XML file. Sales history values that already exist in the model are not overwritten.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing sales history values that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>UnitSkippedCodes</i>	List object that will receive all unit code entries that were not added to the model.
<i>PriceSkippedCodes</i>	List object that will receive all price codes for entries that were not added to the model.

## Example

The following example adds sales history vales to the model from the file `SalesHistory.xml`:

```
# Prepare list objects for receiving errors and skipped sales history codes
set errorsPtr [ListObject]
set skippedUnitPtr [ListListObject]
set skippedPricePtr [ListListObject]

# Add sales history data
SalesHistory_AddSalesHistoryFromXml c:/scp/8.11.1/so-plan/plandata/SalesHistory.xml
$errorsPtr $skippedUnitPtr $skippedPricePtr

# Return any errors or skipped codes
ListObject_Get $errorsPtr
ListListObject_Get $skippedUnitPtr
ListListObject_Get $skippedPricePtr
```

## SalesHistory AddSalesHistoryPrices Command

### Syntax

```
SalesHistory_AddSalesHistoryPrices key date prices
```

### Description

Use the `AddSalesHistoryPrices` command to the model, starting at the specified planning period.

## Parameters

Parameter	Description
<i>key</i>	Sales history key, consisting of a vector containing an item and location value.
<i>date</i>	Start date for the period to which you want to add sales history values, in the format YYYY-MM-DD.
<i>prices</i>	Vector containing the sales history prices that you want to add to the model. Sales and Operations Planning updates as many planning periods as there are values. For example, if the <i>prices</i> parameter contains seven entries, a sales history price is added to seven planning periods, starting with the specified start date.

## Example

The following example adds sales history prices for five planning periods starting June 15, 2005, for the item RES\_2 and location Toronto:

```
SalesHistory_AddSalesHistoryPrices {RES_2 Toronto} 2005-06-15
{12 12 13.50 13.50 14}
```

## SalesHistory AddSalesHistoryUnits Command

### Syntax

```
SalesHistory_AddSalesHistoryUnits key date units
```

### Description

### Parameters

Parameter	Description
<i>key</i>	Sales history key, containing an item and location value.
<i>date</i>	Start date for the period to which you want to add sales history units, in the format YYYY-MM-DD.
<i>units</i>	Vector containing the sales history units that you want to add to the model. Sales and Operations Planning updates as many planning periods as there are values. For example, if the <i>units</i> parameter contains seven entries, a sales history unit value is added to seven planning periods, starting with the specified start date.

## Example

The following example adds sales history units for five planning periods starting June 15, 2005, for the item RES\_2 and location Toronto:

```
SalesHistory_AddSalesHistoryUnits {RES_2 Toronto} 2005-06-15 {60 70 65 50 45}
```

## SalesHistory DeleteAllSalesHistories Command

### Syntax

```
SalesHistory_DeleteAllSalesHistories
```



## Description

Use the DeleteAllSalesHistories command to remove all sales history data from the model.

## SalesHistory DeleteSalesHistoryPrices Command

### Syntax

```
SalesHistory_DeleteSalesHistoryPrices item location channel
```

### Description

Use the DeleteSalesHistoryPrices command to deletes the sales history price values corresponding to the specified item, location and channel codes.

### Parameters

Parameter	Description
<i>item</i>	Item code for the sales history price that you want to delete from the model. If the item code contains blanks, then it must be enclosed in quotation marks.
<i>location</i>	Location code for the sales history price that you want to delete from the model. If the location code contains blanks, then it must be enclosed in quotation marks.
<i>channel</i>	Channel code for the sales history price that you want to delete from the model. If the channel code contains blanks, then it must be enclosed in quotation marks.

### Example

The following example removes sales history prices for the resource RES\_2 at the Toronto Weston location and channel Retail:

```
SalesHistory_DeleteSalesHistoryPrices RES_2 "Toronto Weston" Retail
```

## SalesHistory DeleteSalesHistoryUnits Command

### Syntax

```
SalesHistory_DeleteSalesHistoryUnits itemlocation channel
```

### Description

Use the DeleteSalesHistoryPrices command to deletes the sales history unit values corresponding to the specified item, location and channel codes.

## Parameters

Parameter	Description
<i>item</i>	Item code for the sales history units that you want to delete from the model. If the item code contains blanks, then it must be enclosed in quotation marks.
<i>location</i>	Location code for the sales history units that you want to delete from the model. If the location code contains blanks, then it must be enclosed in quotation marks.
<i>channel</i>	Channel code for the sales history units that you want to delete from the model. If the channel code contains blanks, then it must be enclosed in quotation marks.

## Example

The following example removes sales history unit values for the resource RES\_2 at the Toronto Weston location and channel Retail:

```
SalesHistory_DeleteSalesHistoryUnit RES_2 "Toronto Weston" Retail
```

## SalesHistory UpdateSalesHistoryFromXml Command

### Syntax

```
SalesHistory_UpdateSalesHistoryFromXml filename errors UnitSkippedCodes
PriceSkippedCodes
```

### Description

Use the UpdateSalesHistoryFromXml command to update sales history data from the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing sales history values that you want to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>UnitSkippedCodes</i>	List object that will receive all unit code entries that were not added to the model.
<i>PriceSkippedCodes</i>	List object that will receive all price codes for entries that were not added to the model.

## Example

The following example updates the model with sales history vales from the file SalesHistory.xml:

```
# Prepare list objects for receiving errors and skipped sales history codes
set errorsPtr [ListObject]
set skippedUnitPtr [ListListObject]
set skippedPricePtr [ListListObject]

# Update sales history data
SalesHistory_UpdateSalesHistoryFromXml c:/scp/8.12/so-plan/plandata/SalesHistory.xml
```

```

$errorsPtr $skippedUnitPtr $skippedPricePtr

# Return any errors or skipped codes
ListObject_Get $errorsPtr
ListListObject_Get $skippedUnitPtr
ListListObject_Get $skippedPricePtr

```

## SalesHistory WriteSalesHistoryToXml Command

### Syntax

```
SalesHistory_WriteSalesHistoryToXml filename
```

### Description

Use the WriteSalesHistoryToXml command to write all sales history values found in the model to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where sales history values is written. If the file name includes blanks, then it must be enclosed in quotation marks.

### Example

The following example writes all sales history values in the model to the file SalesHistory.xml:

```
SalesHistory_WriteSalesHistoryToXml c:/plandata/SalesHistory.xml
```

## SupplyPlan AddSupplyPlan Command

### Syntax

```
SupplyPlan_AddSupplyPlan cycleName demandPlanName supplyPlanName
```

### Description

Use the AddSupplyPlan command to add a supply plan to the specified cycle and associated with the specified demand plan.

## Parameters

Parameter	Description
<i>cycleName</i>	Cycle to which you want to add a supply plan. If the cycle name includes blanks, then it must be enclosed in quotation marks.
<i>demandPlanName</i>	Name of the Demand plan to which you want to associate the supply plan. If the demand plan name includes blanks, then it must be enclosed in quotation marks.
<i>supplyPlanName</i>	Name of the Supply plan that you want to add to the model. If the supply plan name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example adds a supply plan Optimistic Supply to the planning cycle Q4 2005:

```
SupplyPlan_AddSupplyPlan "Q4 2005" "Optimistic Demand" "Optimistic Supply"
```

## SupplyPlan DeleteSupplyPlan Command

### Syntax

```
SupplyPlan_DeleteSupplyPlan cycleName supplyPlanName
```

### Description

Use the DeleteSupplyPlan command to remove the specified supply plan from the model.

## Parameters

Parameter	Description
<i>cycleName</i>	Cycle from which you want to remove a supply plan. If the cycle name includes blanks, then it must be enclosed in quotation marks.
<i>supplyPlanName</i>	Name of the Supply plan that you want to remove from the model. If the supply plan name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example removes the Supply plan Optimistic Supply from the planning cycle Q4 2005:

```
SupplyPlan_DeleteSupplyPlan "Q4 2005" "Optimistic Supply"
```

## SupplyPlan GetSupplyPlanNames Command

### Syntax

```
SupplyPlan_GetSupplyPlanNames cycleName
```

### Description

Use the GetSupplyPlanNames command to retrieve all supply plans from the specified planning cycle.

## Parameters

Parameter	Description
<i>cycleName</i>	Cycle from which you want to retrieve all associated supply plans. If the cycle name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example retrieves supply plan names from the planning cycle Q4 2005:

```
SupplyPlan_GetSupplyPlanNames "Q4 2005"
```

## UnitsOfMeasure AddItemUomFromXml Command

### Syntax

```
UnitsOfMeasure_AddItemUomFromXml filename errors
```

### Description

Use the AddItemUomFromXml command to add item unit of measure conversion factors from the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing unit of measure conversion factors that you want to add to the model. If the path includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered when adding unit of measure conversion factors.

## Example

The following example adds unit of measure conversion factors from the file ItemUom.xml:

```
# prepare list objects for receiving errors
errorsPtr [ListObject]

# Add the unit of measure data
UnitsOfMeasure_AddItemUomFromXml c:/scp/8.12/so-plan/plandata/ItemUom.xml
$errorsPtr

# Return any errors
ListObject_Get $errorsPtr
```

## See Also

Units of Measure

## UnitsOfMeasure AddUom Command

### Syntax

```
UnitsOfMeasure_AddUom UomName
```

### Description

Use the AddUom command to add a unit of measure to the Sales and Operations Planning model.

### Parameters

Parameter	Description
<i>UomName</i>	Name of the unit of measure that you want to add to the model. If the unit of measure contains blanks, then it must be enclosed in quotation marks.

### Example

The following example adds the unit of measure Pallet to the Sales and Operations Planning Model:

```
UnitsOfMeasure_AddUom Pallet
```

### See Also

Units of Measure

## UnitsOfMeasure Clear Command

### Syntax

```
UnitsOfMeasure_Clear
```

### Description

Use the Clear command to remove all custom units of measure from the model.

### Example

The following example removes all units of measure values from the Sales and Operations Planning model:

```
UnitsOfMeasure_Clear
```

### See Also

Units of Measure

## UnitsOfMeasure DeleteUom Command

### Syntax

```
UnitsOfMeasure_DeleteUom UomName
```

### Description

Use the DeleteUom command to remove the specified unit of measure from the Sales and Operations Planning model.

## Parameters

Parameter	Description
<i>UomName</i>	Name of the unit of measure value that you want to remove from the Sales and Operations Planning model. If the unit of measure contains blanks, then it must be enclosed in quotation marks.

## Example

The following example removes the Pallet unit of measure from the Sales and Operations Planning model:

```
UnitsOfMeasure_DeleteUom Pallet
```

## See Also

Units of Measure

## UnitsOfMeasure GetItemUom Command

### Syntax

```
UnitsOfMeasure_GetItemUom itemCode UomName
```

### Description

Use the GetItemUom command to retrieve the conversion factor for the specified unit of measure to the standard planning unit of measure.

## Parameters

Parameter	Description
<i>itemCode</i>	Item for which you want to retrieve the unit of measure conversion factor.
<i>UomName</i>	Name of the unit of measure for which you want to retrieve the conversion factor. If the unit of measure contains blanks, then it must be enclosed in quotation marks.

## Returns

A string containing the factor used to convert between the specified unit of measure and the standard planning unit of measure.

## Example

The following example retrieve the factor used to convert the Pallet unit of measure to the standard unit of measure, for the item 9002:

```
UnitsOfMeasure_GetItemUom 9002 Pallet
```

## See Also

Units of Measure

## UnitsOfMeasure GetUomList Command

### Syntax

```
UnitsOfMeasure_GetUomList
```

### Description

Use the GetUomList command to retrieve a sorted list of all units of measure in the Sales and Operations Planning database.

## UnitsOfMeasure SetItemUom Command

### Syntax

```
UnitsOfMeasure_SetItemUom itemCode UomName  
factor
```

### Description

Use the SetItemUom command to add a new unit of measure and corresponding conversion factor for the specified item.

### Parameters

Parameter	Description
<i>itemCode</i>	Code for the item to which you want to add a new unit of measure. If the item code contains blanks, then it must be enclosed in quotation marks.
<i>UomName</i>	Name for the new unit of measure, as it will appear in the Sales and Operations Planning.
<i>factor</i>	Factor used to convert between the standard planning unit of measure and the new unit of measure. The specified factor acts as a multiplier. For example, if the new unit of measure is a palette that can hold 36 items, the factor would be 36. Use decimal values to represent instances where the new unit of measure is smaller than the standard planning units.

### Example

The following example adds the Palette unit of measure with a conversion factor of 36, to the item 9002:

```
UnitsOfMeasure_SetItemUom 9002 Palette 36
```

### See Also

Units of Measure

## UnitsOfMeasure WriteItemUomToXml Command

### Syntax

```
UnitsOfMeasure_WriteItemUomToXml filename
```



## Description

Use the `WriteItemUomToXml` command to write item units of measure and their associated conversion factors to the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	File name and path where the item units of measures are written. If the file name contains blanks, then it must be enclosed in quotation marks.

## Example

The following example writes unit of measure values for all items in the Sales and Operations Planning model to the file `ItemUom.xml`:

```
UnitsOfMeasure_WriteItemUomToXml "c:/plan data/ItemUom.xml"
```

## Users AddUser Command

### Syntax

```
Users_AddUser userId fullName
```

### Description

Use the `AddUser` command to add a user to the database.

### Parameters

Parameter	Description
<i>id</i>	ID of the new user.
<i>fullName</i>	Full name of the new user. If the name contains blanks, then it must be enclosed in quotation marks.

## Example

The following example adds a new user, Nicholas James to the database:

```
Users_AddUser njames "Nicholas James"
```

### See Also

Adding Users

## Users AddUsersFromXml Command

### Syntax

```
Users_AddUsersFromXml filename errors skippedUserIds
```

### Description

User the `AddUsersFromXml` command to add users from the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing users to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>skippedUserIds</i>	List object containing ids for those users that were not added to the model.

## Example

The following example adds users to the model from the specified XML file:

```
# prepare list objects for receiving errors and skipped user codes
set errorsPtr [ListObject]
set skippedUserCodesPtr [ListObject]
# Add users
Users_AddUsersFromXml c:/scp/8.12/so-plan/plandata/Users.xml $errorsPtr
$skippedUserCodesPtr
# Return any errors or skipped user codes
ListObject_Get $errorsPtr
ListObject_Get $skippedUserCodesPtr
```

## See Also

Adding Users

## Users DeleteUser Command

### Syntax

```
Users_DeleteUser userId
```

### Description

User the DeleteUser command to remove a user from the database.

### Parameters

Parameter	Description
<i>userId</i>	ID of the user to delete from the database.

## Example

The following example removes a user, Nicholas James, from the database:

```
Users_DeleteUser njames
```

## See Also

Deleting Users

## Users IsAdmin Command

### Syntax

```
Users_IsAdmin userId
```

### Description

Use the IsAdmin command to determine whether or not the specified user has Administrator privileges.

### Parameters

Parameter	Description
<i>userId</i>	ID of the user that you want to check for administrator privileges.

### Returns

True if the specified user has administrator; otherwise False.

### Example

The following example checks to see if the user Nicholas James has administrator:

```
Users_IsAdmin njames
```

### See Also

Administering Users

## Users IsViewAdmin Command

### Syntax

```
Users_IsViewAdmin userId
```

### Description

Use the IsViewAdmin command to determine whether or not the specified user has View Administrator privileges.

### Parameters

Parameter	Description
<i>userId</i>	ID of the user that you want to check for View Administrator privileges.

### Returns

True if the specified user has administrator; otherwise False.

### Example

The following example checks to see if the user Nicholas James has View Administrator privileges:

```
Users_IsViewAdmin njames
```

**See Also**

Administering Users

## Users SetPassword Command

**Syntax**

```
Users_SetPassword userId password
```

**Description**

User the SetPassword command to set the password for the specified user.

**Parameters**

Parameter	Description
<i>userId</i>	Id of the user whose password you want to set.
<i>password</i>	New user password.

**Example**

The following example sets the password for the user Nicholas James:

```
Users_SetPassword njames password123
```

**See Also**

Changing Passwords

## Users StrictUpdateUsersFromXml Command

**Syntax**

```
Users_StrictUpdateUsersFromXml filename errors  
invalidUserIds
```

**Description**

Use the StrictUpdateUsersFromXml command to do a strict update of user information from the specified XML file. If a user in the XML file does not exist in the model then that entry is ignored and the id is added to the list of invalid users.

**Parameters**

Parameter	Description
<i>filename</i>	Name and path of the XML file containing users to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>invalidUserIds</i>	List object containing user Ids that were not added to the model. Only user Ids that already exist in your model are updated.

## Example

The following example performs a strict update of the model using user data from the specified XML file:

```
# prepare list objects for receiving errors and invalid user codes
set errorsPtr [ListObject]
set invalidUserCodesPtr [ListObject]
# Update user data
Users_StrictUpdateUsersFromXml c:/scp/8.12/so-plan/plandata/Users.xml
$errorsPtr $invalidUserCodesPtr
# Return any errors or invalid user codes
ListObject_Get $errorsPtr
ListObject_Get $invalidUserCodesPtr
```

## Users UpdateUsersFromXml Command

### Syntax

```
Users_UpdateUsersFromXml filename errors invalidUserIds
```

### Description

Use the UpdateUsersFromXml command to do an update of user information from the specified XML file. If a user in the XML file does not exist in the model, then it is added.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file containing users to add to the model. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.
<i>addedUserIds</i>	List object containing user Ids that were added to the model.

## Example

The following example updates the model using user data from the specified XML file:

```
# prepare list objects for receiving errors and added user codes
set errorsPtr [ListObject]
set addedUserCodesPtr [ListObject]
# Update user data
Users_StrictUpdateUsersFromXml c:/scp/8.12/so-plan/plandata/Users.xml
$errorsPtr $addedUserCodesPtr
# Return any errors or added user codes
ListObject_Get $errorsPtr
ListObject_Get $addedUserCodesPtr
```

## Users WriteUsersToXml Command

### Syntax

```
Users_WriteUsersToXml filename
```

## Description

Use the WriteProductionToXml command to write user information to the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where the user data is written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes your user data to the file Users.xml:

```
Users_WriteUsersToXml c:/scp/8.12/so-plan/plandata/Users.xml
```

## Views AddConsolidatedView Command

### Syntax

```
Views_AddConsolidatedView viewName ownerId description dimensions uOm
```

### Description

Use the AddConsolidatedView command to add the specified Consolidated view to the Sales and Operations Planning model.

### Parameters

Parameter	Description
<i>viewName</i>	Name of the Consolidated view that you want to create. If the view name includes blanks, then it must be enclosed in quotation marks.
<i>ownerId</i>	The Sales and Operations Planning user ID to which the view is assigned. The user ID must already exist in the model.
<i>description</i>	A detailed description of the Consolidated view. If the description includes blanks, then it must be enclosed in quotation marks.
<i>dimensions</i>	A list of data dimensions used by the view. Each dimension in the view definition must be enclosed in curly brackets.
<i>uOm</i>	A list of units of measure that you want to make available in the view. Each unit of measure must be enclosed in curly brackets.

## Example

The following example adds a Consolidated view to the Sales and Operations Planning model:

```
Views_AddConsolidatedView "Consolidated View By Country" admin "Consolidated Plan by country and major selling channel" {{Location Country}} {{each}}
```

## Views AddDemandPlanView Command

### Syntax

```
Views_AddDemandPlanView viewName ownerId description dimensions metrics uOm
```

### Description

Use the AddDemandPlanView command to add the specified Demand Plan to the Sales and Operations Planning model.

### Parameters

Parameter	Description
<i>viewName</i>	Name of the Demand Plan view that you want to create. If the view name includes blanks, then it must be enclosed in quotation marks.
<i>ownerId</i>	The Sales and Operations Planning user ID to which the view is assigned. The user ID must already exist in the model.
<i>description</i>	A detailed description of the Demand Plan view. If the description includes blanks, then it must be enclosed in quotation marks.
<i>dimensions</i>	A list of data dimensions used by the view. Each dimension in the view definition must be enclosed in curly brackets.
<i>metrics</i>	<p>The system metrics that you want to apply to the view. Each metric in the view definition must be enclosed in curly brackets.</p> <p>This parameter is optional; if no metrics are selected, then all applicable metrics are assigned to the view.</p> <p><b>Note.</b> Not all system metrics are compatible with all Demand Plan views.</p>
<i>uOm</i>	A list of units of measure that you want to make available in the view. Each unit of measure must be enclosed in curly brackets.

### Example

The following example adds a Demand Plan view to the Sales and Operations Planning model:

```
Views_AddDemandPlanView "Detailed Demand Review" admin "" {{Channel Code} {Item Code}
{Location.city}} {{{}} {{each} {cases} {litres}}}
```

## Views AddFinancialView Command

### Syntax

```
Views_AddFinancialView viewName ownerId description dimensions uOm
```

### Description

Use the AddFinancialView command to add the specified Financial View to the Sales and Operations Planning model.

## Parameters

Parameter	Description
<i>viewName</i>	Name of the Financial view that you want to create. If the view name includes blanks, then it must be enclosed in quotation marks.
<i>ownerId</i>	The Sales and Operations Planning user ID to which the view is assigned. The user ID must already exist in the model.
<i>description</i>	A detailed description of the Financial view. If the description includes blanks, then it must be enclosed in quotation marks.
<i>dimensions</i>	A list of data dimensions used by the view. Each dimension in the view definition must be enclosed in curly brackets.
<i>uOm</i>	A list of units of measure that you want to make available in the view. Each unit of measure must be enclosed in curly brackets.

## Example

The following example adds a Financial view to the Sales and Operations Planning model:

```
Views_AddFinancialView "Projected Financials" admin "Projected financials and costs by
brand and major selling channel." {{Channel.Channel Type} {Item.Brand}} {{each}}
```

## Views AddInventoryView Command

### Syntax

```
Views_AddInventoryView viewName ownerId description dimensions uOm
```

### Description

Use the AddInventoryView command to add the specified Inventory view to the Sales and Operations Planning model.

### Parameters

Parameter	Description
<i>viewName</i>	Name of the Inventory view that you want to create. If the view name includes blanks, then it must be enclosed in quotation marks.
<i>ownerId</i>	The Sales and Operations Planning user ID to which the view is assigned. The user ID must already exist in the model.
<i>description</i>	A detailed description of the Inventory view. If the description includes blanks, then it must be enclosed in quotation marks.
<i>dimensions</i>	A list of data dimensions used by the view. Each dimension in the view definition must be enclosed in curly brackets.
<i>uOm</i>	A list of units of measure that you want to make available in the view. Each unit of measure must be enclosed in curly brackets.



## Example

The following example adds an Inventory view to the Sales and Operations Planning model:

```
Views_AddInventoryView "Branch Inventory by Type" admin "" {{Channel Code} {Item Code}
{Location.City}} {{each} {case} {litres}}
```

## Views AddProductionPlanView Command

### Syntax

```
Views_AddProductionPlanView viewName ownerId description dimensions metrics uOm
```

### Description

Use the AddProductionPlanView to add the specified Production Plan view to the Sales and Operations Planning model.

### Parameters

Parameter	Description
<i>viewName</i>	Name of the Production Plan view that you want to create. If the view name includes blanks, then it must be enclosed in quotation marks.
<i>ownerId</i>	The Sales and Operations Planning user ID to which the view is assigned. The user ID must already exist in the model.
<i>description</i>	A detailed description of the Production Plan view. If the description includes blanks, then it must be enclosed in quotation marks.
<i>dimensions</i>	A list of data dimensions used by the view. Each dimension in the view definition must be enclosed in curly brackets.
<i>metrics</i>	The system metrics that you want to apply to the view. Each metric in the view definition must be enclosed in curly brackets. This parameter is optional; if no metrics are selected, then all applicable metrics are assigned to the view. <b>Note.</b> Not all system metrics are compatible with all Production Plan views.
<i>uOm</i>	A list of units of measure that you want to make available in the view. Each unit of measure must be enclosed in curly brackets.

## Example

The following example adds a Production Plan view to the Sales and Operations Planning model:

```
Views_AddProductionPlanView "Detailed Production Plan" admin "" {{Item Code}
{Location.City}} {{}} {{each} {case} {litres}}
```

## Views AddRoughCutCapacityView Command

### Syntax

```
Views_AddRoughCutCapacityView viewName ownerId description dimensions metrics uOm
```

## Description

Use the `AddRoughCutCapacityView` command to add the specified Rough Cut Capacity view to the Sales and Operations Planning model.

## Parameters

Parameter	Description
<i>viewName</i>	Name of the Rough Cut Capacity view that you want to create. If the view name includes blanks, then it must be enclosed in quotation marks.
<i>ownerId</i>	The Sales and Operations Planning user ID to which the view is assigned. The user ID must already exist in the model.
<i>description</i>	A detailed description of the Rough Cut Capacity view. If the description includes blanks, then it must be enclosed in quotation marks.
<i>dimensions</i>	A list of data dimensions used by the view. Each dimension in the view definition must be enclosed in curly brackets.
<i>metrics</i>	The system metrics that you want to apply to the view. Each metric in the view definition must be enclosed in curly brackets. This parameter is optional; if no metrics are selected, then all applicable metrics are assigned to the view. <b>Note.</b> Not all system metrics are compatible with all Production Plan views.
<i>uOm</i>	A list of units of measure that you want to make available in the view. Each unit of measure must be enclosed in curly brackets.

## Example

The following example adds a Rough Cut Capacity view to the Sales and Operations Planning model:

```
Views_AddRoughCutCapcityView "Rough Cut Capacity" admin "A comparison of planned and
required capacity" {{Resource Code}} {{Planned Capacity} {Required Capacity}
{Deviation} {Utilization %} {Maximum Capacity}} {{each} {case}}
```

## Views DeleteAllViews Command

### Syntax

```
Views_DeleteAllViews
```

### Description

Use the `DeleteAllViews` command to remove all defined planning views from the Sales and Operations Planning model.

### Example

The following example deletes all views from the Sales and Operations Planning model:

```
Views_DeleteAllViews
```

### See Also

Working with Views

## Views DeleteView Command

### Syntax

```
Views_DeleteView viewName
```

### Description

Use the DeleteView command to remove the specified Sales and Operations Planning view from the model.

### Parameters

Parameter	Description
<i>viewName</i>	Name of data view that you want to remove from the model. If the description includes blanks, then it must be enclosed in quotation marks.

### Example

The following example removes the view Detailed Demand Plan from the model:

```
Views_DeleteView "Detailed Demand Plan"
```

## Views ExecuteAllViews Command

### Syntax

```
Views_ExecuteAllViews
```

### Description

Use the ExecuteAllViews command to execute all defined Sales and Operations Planning views. This command is useful when performing batch updates to the model.

### Example

The following example executes all Sales and Operations Planning views:

```
Views_ExecuteAllViews
```

## Views ExecuteView Command

### Syntax

```
Views_ExecuteView viewName
```

### Description

Use the ExecuteView command to execute and update the specified Sales and Operations Planning data view.

### Parameters

Parameter	Description
<i>viewName</i>	Name of the data view that you want to execute. If the view name contains blanks, then it must be enclosed in quotation marks.

## Example

The following example executes the view Item Inventory:

```
Views_ExecuteView "Item Inventory"
```

## See Also

Working with Views

## Views ExportRelativeTargets Command

### Syntax

```
Views_ExportRelativeTargets filename
```

### Description

Use theExportRelativeTargets command to write all threshold values found in the Sales and Operations Planning model to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where threshold values are written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes all threshold values to the file RelativeData.xml:

```
Views_ExportRelativeData c:/scp/8.12/so-plan/plandata/RelativeData.xml
```

## See Also

Working with Views

## Views ExportTargets Command

### Syntax

```
Views_ExportTargets filename
```

### Description

Use the ExportTargets command to write all target values found in the Sales and Operations Planning model to the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where target values are written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes all target values to the file Targets.xml:

```
Views_ExportTargets c:/scp/8.12/so-plan/plandata/Targets.xml
```

## See Also

Working with Views

# Views ExportThresholds Command

## Syntax

```
Views_ExportThresholds filename
```

## Description

Use the ExportThresholds command to write all threshold values found in the Sales and Operations Planning model to the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where threshold values are written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes all target values to the file Thresholds.xml:

```
Views_ExportThresholds c:/scp/8.12/so-plan/plandata/Thresholds.xml
```

## See Also

Working with Views

# Views ExportViews Command

## Syntax

```
Views_ExportViews filename
```

## Description

Use the ExportViews command to export all Sales and Operations Planning view definitions to the specified XML file.

---

**Note.** The ExportViews command does not export any model data.

---

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file where view definitions are written. If the file name includes blanks, then it must be enclosed in quotation marks.

## Example

The following example writes your view definitions to the file Views.xml:

```
Views_ExportViews c:/scp/8.12/so-plan/plandata/Views.xml
```

## See Also

Working with Views

## Views GetCyclePlans Command

### Syntax

```
Views_GetCyclePlans viewName
```

### Description

Use the GetCyclePlans command to retrieve a list of all planning cycles, all demand and supply plans associated with each cycle, and whether or not each cycle and plan is associated with the specified data view.

## Parameters

Parameter	Description
<i>viewName</i>	Data view that you want to validate against the list of all cycles and plans. If the view name contains blanks, then it must be enclosed in quotation marks.

## Returns

A vector containing a list of all planning cycles, the supply and demand plans associated with each cycle, and a true/false flag indicating if the specified view is associated with each plan.

## Example

The following example checks the view Total Margin against all cycles and plans in the model:

```
Views_GetCyclePlans "Total Margin"
```

## See Also

Working with Views

## Views GetKeys Command

### Syntax

```
Views_GetKeys viewName cycleName demandPlan supplyPlan
```

## Description

Use the GetKeys command to get a list of all valid keys (item and location pairs) in a view for the specified cycle, Demand and Supply plan.

## Parameters

Parameter	Description
<i>viewName</i>	Name of the view from which you want to return a list of valid keys.
<i>cycleName</i>	Planning cycle for which you want to retrieve a list of valid keys. This parameter is optional; if no cycle is specified, the current planning cycle is used.
<i>demandPlan</i>	Name of the associated Demand plan. If the Demand plan name contains blanks, it must be enclosed in quotation marks.
<i>supplyPlan</i>	Name of the associated Supply plan. If the Supply plan name contains blanks, it must be enclosed in quotation marks.

## Returns

Returns a vector of valid item/location pairs.

## Example

The following example retrieves item and location values from the Detailed Inventory Plan view:

```
Views_GetKeys "Detailed Inventory Plan" "" "Apr Baseline" "Std Capacity"
```

## Views GetMetricValues Command

### Syntax

```
Views_GetMetricValues viewName metric planningPeriodGroup cycleName  
demandPlan supplyPlan uOm key
```

### Description

Use the GetMetricValues command to retrieve metric values for the specified data view, planning cycle and Demand and Supply Plan.

## Parameters

Parameter	Description
<i>viewName</i>	Name of the Sales and Operations Planning view from which you want to return metric values.
<i>metric</i>	Name of the planning metric for which you want to retrieve values.
<i>planningPeriodGroup</i>	Granularity of the planning period. Valid values are: <ul style="list-style-type: none"> <li>• Weekly.</li> <li>• Monthly.</li> <li>• Quarterly.</li> <li>• Yearly.</li> </ul>
<i>cycleName</i>	Planning cycle for which you want to retrieve metric values. This parameter is optional; if no cycle is specified, the current planning cycle is used.
<i>demandPlan</i>	Name of the associated Demand plan. If the Demand plan name contains blanks, it must be enclosed in quotation marks.
<i>supplyPlan</i>	Name of the associated Supply plan. If the Supply plan name contains blanks, it must be enclosed in quotation marks.
<i>uOm</i>	The unit of measure in which you want to display the results. The unit of measure must be defined for the specified metric. <b>Note.</b> The default unit of measure is “each”
<i>key</i>	Vector containing an item and location value.

## Returns

A list of metric values.

## Example

The following example gets monthly values for the metric Total Inventory in the view Detailed Inventory Plan:

```
Views_GetMetricValues "Detailed Inventory Plan" "Total Inventory" Monthly "Q4 2006"
"Apr Baseline" "Std Capacity" each {RES_2 Toronto}
```

## Views GetViewsStatus Command

### Syntax

```
Views_GetViewsStatus userId
```

### Description

Use the GetViewStatus command to retrieve the status of all views created by the specified user.



## Parameters

Parameter	Description
<i>userId</i>	ID of the user for which you want to retrieve the view status.

## Returns

A structured list of the view name, type, isActive (boolean), owner, description and definition.

## Example

The following example returns the status of all views created by the user Njames:

```
Views_GetViewsStatus njames
```

## Views ImportRelativeTargets Command

### Syntax

```
Views_ImportRelativeTargets filename errors
```

### Description

Use the ImportRelativeTargets command to import relative metric values from the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file from where you want to import relative target definitions. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.

## Example

The following example imports relative target values from the file RelativeTargets.xml:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Update relative targets
Views_ImportRelativeTargets c:/scp/8.12/so-plan/plandata/RelativeTargets.xml
$errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## Views ImportTargets Command

### Syntax

```
Views_ImportTargets filename errors
```

## Description

Use the ImportTargets command to import metric target values from the specified XML file.

## Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file from where you want to import metric target definitions. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.

## Example

The following example imports target values from the file Targets.xml:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Update target data
Views_ImportTargets c:/scp/8.12/so-plan/plandata/Targets.xml $errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## Views ImportThresholds Command

### Syntax

```
Views_ImportThresholds filename errors
```

### Description

Use the ImportThresholds command to import metric threshold values from the specified XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file from where you want to import metric threshold definitions. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>errors</i>	List object containing any errors encountered during the update.

## Example

The following example imports threshold values from the file Threshold.xml:

```
# prepare list objects for receiving errors
set errorsPtr [ListObject]
# Update threshold data
Views_ImportThresholds c:/scp/8.12/so-plan/plandata/Threshold.xml $errorsPtr
# Return any errors
ListObject_Get $errorsPtr
```

## Views ImportViews Command

### Syntax

```
Views_ImportViews filename executeIfActive
```

### Description

Use the ImportViews command to import data views from the specified XML file. If views stored in the XML file already exist in the model, then those views are replaced by the views from the XML file.

### Parameters

Parameter	Description
<i>filename</i>	Name and path of the XML file from where you want to import view definitions. If the file name includes blanks, then it must be enclosed in quotation marks.
<i>executeIfActive</i>	Boolean flag indicating whether or not you want to execute active views when they are imported to the model.

### Example

The following example imports and executes active views from the file Views.xml:

```
Views_ImportViews c:/scp/8.12/so-plan/plandata/Views.xml
```

## Views InvalidateAllViews Command

### Syntax

```
Views_InvalidateAllViews
```

### Description

Use the InvalidateAllViews command to invalidate all views in the Sales and Operations Planning model. This command is useful to turn off all view updates while performing a batch update of model data.

### Example

The following example invalidates all Sales and Operations Planning views:

```
Views_InvalidateAllViews
```

## Views SetPlansForViewCycle Command

### Syntax

```
Views_SetPlansForViewCycle viewName cycleName plans
```

### Description

Use the SetPlansForViewCycle command to assign plans to the specified data view.

## Parameters

Parameter	Description
<i>viewName</i>	Data view to which you want to assign plans. If the view name contains blanks, then it must be enclosed in quotation marks.
<i>cycleName</i>	Cycle for which you want to assign plans. If the cycle name contains blanks, then it must be enclosed in quotation marks.
<i>plans</i>	Vector containing cycles and plans that you want to assign to the specified view.

## Example

The following example assigns the plans dem1 and pro1 to the view cycle 1:

```
# Set item, location & Channel properties
Items_SetItemPropertyNames {Height Weight Color}
Locations_SetLocationPropertyNames {City}
Channels_SetChannelPropertyNames {Name}

# Add the item, location and Channel
Items_AddItem b1 "velotruck"
Locations_AddLocation l1 "TorontoStore"
Channels_AddChannel c1 "CostMart on Dufferin/Bloor"

# Set item properties for item b1
Items_SetItemProperties b1 {{Height 1000.00} {Weight 200} {Color Black}}
Locations_SetLocationProperties l1 {{City Toronto}}
Channels_SetChannelProperties c1 {{Name CostMart}}

# Add Views
Views_AddView "Production Plan" view1 admin "This view has three dimensions"
{{Item Height} {Item Weight} {Location City}} {} {{}}

# Add the Period
Periods_AddPeriod "2003-12-01" "Dec 2003" "Q4 2003" "2003"
Periods_AddPeriod "2004-01-01" "Jan 2004" "Q1 2004" "2004"
Periods_AddPeriod "2004-02-01" "Feb 2004" "Q1 2004" "2004"
Periods_AddPeriod "2004-03-01" "Mar 2004" "Q1 2004" "2004"
Periods_AddPeriod "2004-04-01" "Apr 2004" "Q1 2004" "2004"

# Add the cycle
Cycles_AddCycle "cycle 1" "2004-01-01"

# Add Demand Plan
DemandPlan_AddDemandPlan "cycle 1" dem1

# Add Supply Plan
SupplyPlan_AddSupplyPlan "cycle 1" dem1 pro1

Cycles_SetCurrentCycle "cycle 1"
```

```
# Add the Time Series
DemandPlan_AddDemand dem1 "b1" "l1" "c1" "2004-01-01" {1 2 3 4}
ProductionPlan_AddProduction pro1 "b1" "l1" "2004-01-01" {1 2 3 4}

# Set Plans for view
Views_SetPlansForViewCycle "Detailed Inventory Plan" "JAN-06" {"Demand Below" "Std Capacity"
{"Expected Demand" "Std Capacity"}} {"Demand Above" "Std Capacity"}}
```

**See Also**

Working with Views



## CHAPTER 10

# EnterpriseOne Sales and Operations Planning Glossary

This chapter provides definitions of commonly-used Sales and Operations Planning terms.

---

## EnterpriseOne Sales and Operations Planning Glossary

The follow table contains glossary items used in Sales and Operations Planning:

Average Selling Price	The average price for which an item is sold. Prices for individual items are imported. When viewing this metric at an aggregated level, Sales and Operations Planning averages the price for all items.
Comparison Format	A view format used to compare key metrics between two planning cycles. All data views can be displayed using the Comparison format.
Consolidated View	A data view that displays the combined Demand, Production, and Inventory plans. This plan is used during the reconciliation process to get a high-level view of all your plans.
Cost of Goods Sold	The total cost of the products or goods that a company sells to generate revenue.
Cost of Goods Sold per Unit	The per unit cost of the products or goods that a company sells to generate revenue. The cost of goods sold per unit is imported by item and location.
Dashboard Format	<p>A view format used to compare plan data against your defined targets and thresholds to give you an indication of your plan's success or failure.</p> <p>For each available metric, the Dashboard displays a current and target value and the total and percentage deviation between these values. A status indicator shows whether or not the plan has achieved the defined targets.</p>

Default Format	<p>A view format that displays your plan data in tabular layout. The Table format is the default data view.</p> <p>You can pivot tables by rotating rows and columns to see different summaries of your plan data.</p>
Demand Fill Percentage	<p>The percentage of total demand that is being met by production and inventory. The demand fill percentage is calculated using unit demand minus production and inventory.</p>
Demand Plan	<p>A Sales and Operations Planning view that displays projected unit demand and revenue for the current planning cycle grouped by customer, item and location dimensions.</p>
Deviation	<p>The absolute difference between required and planned capacity.</p>
Financial View	<p>A Sales and Operations Planning view that displays key financial data over time. Use the Financial view during the reconciliation process to evaluate your demand and production plans against budgeted targets such as revenue, cost, and margin.</p>
Holding Cost	<p>The cost required to hold an item in inventory for the specified period of time. Holding costs are imported.</p>
Inventory	<p>All items held in stock at the end of a period for future sale. Inventory is calculated using starting inventory plus production minus demand.</p>
Inventory Plan	<p>A Sales and Operations Planning view that displays product inventory levels for a given time period. The Inventory Plan is used during the supply review process to determine how an organization plans to meet demand based on its existing inventory and production capacity.</p>
Margin	<p>The gross profit for an organization, product line, item, or other category. Margin is calculated as net sales minus the cost of goods sold.</p>
Margin Percentage	<p>The gross profit for an organization, product line, location, or other category represented as a percentage.</p>
Maximum Capacity	<p>The highest feasible capacity for a resource or group of resources, for a given time period. Maximum capacity is imported.</p>
Metric Exception Report	<p>A report that highlights where one or more planning cycle differs from performance targets. Use Metric Exception reports to quickly see which plan areas are cause for concern, and to drill-down to view the exceptions in more detail.</p>



Net Change Report	A report that highlights major changes in plan values from one cycle to another using the metrics available in the specified view. For example, a report based on the Demand Plan view highlights changes in demand forecasts between the compared cycles. When defining Net Change reports, you choose which metrics you want to compare.
Plan Calendar	A Sales and Operations Planning page that enables you to track process-related milestones, assign internal and external action items and manage meetings.
Planned Capacity	The current available capacity of a resource or group of resources, for a given time period. Planned capacity is imported.
Planning Cycle	A Sales and Operations Planning iteration. Planning cycles are typically one month in duration, with each of these cycles representing iterations of the planning process. Using planning cycles you can access, organize, and compare multiple plan data snapshots.
Planning Period	User-defined time periods that maps weekly time buckets to fiscal years, quarters, and months.
Production Costs	The costs incurred producing an item. Production costs is calculated using the cost per item times the number of items.
Production Plan	A Sales and Operations Planning view that displays the overall level of planned production for a given time period. The Production Plan is used during the supply review process to determine how an organization plans to meet demand based on its existing inventory and production capacity.
Required Capacity	The amount of capacity required to meet demand.  This metric is imported.
Revenue	The earnings attributed to an item, product family, or organization before any costs or expenses are deducted. Revenue is calculated using the average selling price times unit demand.
Rough Cut Capacity Plan	A Sales and Operations Planning view that displays the current production capabilities for a given time period. Use this view during the reconciliation process to identify differences between planned and required production capacity, and your current production utilization percentage.
Sales and Operations Planning Automation Shell (SASH)	A command line interface that enables you to perform Sales and Operations Planning functions using Tcl commands.

Sales and Operations Planning Connector	A utility that facilitates the data transfer from Supply Chain Business Modeler to Sales and Operations Planning. Planning data from Supply Chain Business Modeler is imported to Sales and Operations Planning, and then used to create and evaluate plans.
Shortage	The amount of demand that cannot be met by current supply. Shortage is calculated using unit demand minus production and inventory.
Unit Demand	The projected volume of sales for a given time period. Unit demand is imported.
Unit Production	The number of items produced during a given time period. Unit production is imported.
Utilization Percentage	The percentage of a resource's total available capacity that is used to meet demand during a given time period. Utilization percentage is calculated using required capacity as a percentage of available capacity.
View Manager	A Sales and Operations Planning page that displays a list of all defined views, the corresponding type, and the number of rows in each view. You can use the View Manager to add, remove and execute views.
Waterfall Format	<p>A view format available in the Consolidated data view. Use the Waterfall format to identify changes by comparing multiple plan versions.</p> <p>In a Waterfall display, the rows of the table are reserved for the various plan versions that you are comparing and the columns are the different planning periods.</p>

# Index

## A

- absolute metrics 37, 66
  - adding rules 71
  - changing order of thresholds 73
  - changing planning period group 71
  - inserting threshold rules 72
- action items 53
  - adding to Plan Calendar 56
- Add Action page 56
- Add Meetings page 56
- Add Milestone page 56
- Add New Report wizard 79, 80, 81
- Add View wizard 65, 67
- AddAction Tcl command 106
- AddChannel Tcl command 116
- AddChannelsFromXml Tcl command 117
- AddConsolidatedView Tcl command 240
- AddCustomMetric Tcl command 123
- AddCycle Tcl command 129
- AddCyclesFromXml Tcl command 130
- AddDemand Tcl command 137
- AddDemandFromXml Tcl command 138
- AddDemandFromXmlToCycle Tcl command 138
- AddDemandPlan Tcl command 139
- AddDemandPlanView Tcl command 241
- AddFinanicalView Tcl command 241
- adding
  - action items to Plan Calendar 56
  - custom metrics to views 69
  - log patterns 51
  - meetings to Plan Calendar 56
  - milestones to Plan Calendar 56
  - planning cycles 49
  - planning periods 5, 48
  - target rules for absolute metrics 71
  - target rules for relative metrics 70
  - users 5, 46
  - workflows 90
- AddInventoryHistory Tcl command 149
- AddInventoryHistoryFromXml Tcl command 149
- AddInventoryView Tcl command 242
- AddItemUomFromXml Tcl command 231
- AddMeeting Tcl command 107
- AddMilestone Tcl command 108
- AddPlanMappingFromXml Tcl command 131, 186
- AddPlannedCapacity Tcl command 198
- AddPlannedCapacityFromXml Tcl command 199
- AddProductionHistory Tcl command 187
- AddProductionHistoryFromXml Tcl command 187
- AddProductionPlanView command 243
- AddRequiredCapacity Tcl command 200
- AddRequiredCapacityFromXml Tcl command 200
- AddRequiredCapacityFromXmlToCycle Tcl command 201
- AddResource Tcl command 202
- AddResourceMaximumCapacitiesFromXml Tcl command 202
- AddResourceMaximumCapacity Tcl command 203
- AddResourcesFromXml Tcl command 203
- AddSalesHistoryFromXml Tcl command 224
- AddSalesHistoryPrices Tcl command 225
- AddSalesHistoryUnits Tcl command 226
- AddSupplyPlan Tcl command 229
- AddToCalendarFromXml Tcl command 109
- AddUom Tcl command 232
- administering users 39, 45
- administrator desktop 39
- App Backup command 62
- App GetVersion Tcl command 104
- Application commands
  - Archive 100
  - Backup 101
  - Clear 101
  - DatabaseExists 102
  - FixJavaClasspath 102
  - GetDBDiskFree 103
  - GetInstallLocation 103

- GetVersion 104
- LoadPropertyNames 104
- Restore 104
- RunBackground 105
- WritePropertyNames 106
- Application Commands
  - GetCompressMode 102
  - SetCompressMode 105
- application environment 5
- approvals
  - managing 49
- Archive Tcl commands 100
- assigning
  - permissions to views 69
  - plans to views 68

**B**

- Backup Tcl command 101
- Base package 59
- batch scripts
  - example 88
  - running 89
  - understanding 88
- BeginningInventory package 59
- budgets 14
- business processes 2

**C**

- Calendar commands
  - AddAction 106
  - AddMeeting 107
  - AddMilestone 108
  - Clear 109
  - DeleteAction 110
  - DeleteMeeting 110
  - DeleteMilestone 111
  - GetAction 111
  - GetAllActions 112
  - GetAllMeetings 113
  - GetMeeting 112
  - GetMilestone 112
  - UpdateAction 114
  - UpdateMeeting 114
  - UpdateMilestone 115
  - WriteCalendarToXml 116
- Calendar Commands
  - AddToCalendarFromXml 109
- category patterns 41
- Change Planning Period Group page 71

- changing
  - order of thresholds 73
  - passwords 47
- channels 36
- Channels commands
  - AddChannel 116
  - AddChannelsFromXml 117
  - CheckExists 118
  - DeleteAllChannels 118
  - DeleteChannel 118
  - GetChannelCodes 119
  - GetChannelProperties 119
  - GetChannelPropertyNames 120
  - SetChannelProperties 120
  - SetChannelPropertyNames 120
  - StrictUpdateChannelsFromXml 121
  - UpdateChannelsFromXml 122
  - WriteChannelsToXml 122
- Channels SetChannelPropertyNames Tcl command 120
- CheckExists Tcl command 118, 204
- Clear Tcl command 101, 109, 232
- clearing
  - log events 50
  - log patterns 51
- combined data views 17
- comparing planning cycles 75
- Comparison format 32
- Config.xml 61
- Configure Logging Patterns page 51
- configuring
  - application environment 5
  - data import 61
  - data views 6
  - Plan Calendar 5
  - planning cycles 42
  - Sales and Operations Planning 39
- Configuring Log Patterns page 51
- Connector Refresh command 59
- Consolidated data view 26, 34
- Contains Tcl command 131
- ContainsId Tcl command 140
- cost of good sold per unit metric 27
- creating
  - batch scripts 88
  - data views 6, 67
  - Metric Exception reports 6, 80
  - Net Change reports 6, 78
- custom metrics 43
  - adding to views 69

- Cycle 44
- Demand Plan 44
- Global 44
- Supply Plan 44
- custom workflows 89
  - adding 90
  - executing 90
  - viewing logs 90
  - viewing output 90
- CustomMetrics commands
  - AddCustomMetric 123
  - DeleteCustomMetric 124
  - ExportCustomMetric 124
  - ExportCustomMetrics 125
  - GetValues 125
  - ImportCustomMetrics 126
  - SetValues 127
- CustomWorkflows commands
  - ExecuteProcedure 128
  - GetLogEvents 128
  - GetOutput 129
- Cycles command
  - WritePlanMappingToXml 137
- Cycles commands
  - AddCycle 129
  - AddCyclesFromXml 130
  - AddPlanMappingFromXml 131
  - Contains 131
  - DeleteAllCycles 132
  - DeleteApprovedPlan 132
  - DeleteCycle 132
  - GetApprovedPlanCyclesInfo 133
  - GetCurrentCycleName 134
  - SetApprovedPlan 134
  - SetCurrentCycle 134
  - StrictUpdateCyclesFromXml 135
  - UpdateCyclesFromXml 136
  - WriteCyclesToXml 136

## D

- Dashboard
  - view format 13
- Dashboard view format 11, 33
- data
  - dimension properties 36
  - exporting 62
  - import overview 59
  - import requirements 60
- data views
  - adding custom metrics 69
  - assigning permissions 69
  - assigning plans 68
  - combined 17
  - Comparison 32
  - Consolidated 26
  - creating 6, 67
  - Dashboard 33
  - default format 32
  - Demand Plan 23
  - Financial 27
  - Inventory Plan 24
  - overview 17, 65
  - plan 22
  - Production Plan 24
  - running 68
  - view formats 31
  - view permissions 18
  - Waterfall format 34
- DataBaseExists Tcl command 102
- Default format 32
- default password 39
- defining
  - planning periods 42
  - targets and thresholds 6
- DeleteAction Tcl command 110
- DeleteAllChannels Tcl command 118
- DeleteAllCycles Tcl command 132
- DeleteAllDemands Tcl command 141
- DeleteAllDemandsForCycle Tcl command 141
- DeleteAllPlannedCapacityForCycle command 205
- DeleteAllRequiredCapacities Tcl command 205
- DeleteAllRequiredCapacitiesForCycle 206
- DeleteAllResources Tcl command 206
- DeleteAllSalesHistories Tcl command 226
- DeleteAllViews Tcl command 244
- DeleteApprovedPlan Tcl command 132
- DeleteChannel Tcl command 118
- DeleteCustomMetric Tcl command 124
- DeleteCycle command 132
- DeleteDemand Tcl command 141
- DeleteInventoryHistory Tcl command 150
- DeleteMeeting Tcl command 110
- DeleteMilestone Tcl command 111
- DeletePlannedCapacity Tcl command 205, 207

- DeleteProductionHistory Tcl
  - command 188
- DeleteRequiredCapacity Tcl
  - command 207
- DeleteResource Tcl command 208
- DeleteSalesHistoryPrices Tcl
  - command 227
- DeleteSalesHistoryUnits Tcl
  - command 227
- DeleteSupplyPlan Tcl command 230
- DeleteUom Tcl command 232
- DeleteView Tcl command 245
- deleting
  - planning cycles 49
  - user accounts 46
- demand fill percentage metric 26
- Demand Forecasting 1
- demand metric 26
- Demand Plan
  - data view 23
  - review 11
  - review meeting 7
- Demand Plan view 17
- DemandPlan commands
  - AddDemand 137
  - AddDemandFromXml 138
  - AddDemandFromXmlToCycle 138
  - AddDemandPlan 139
  - ContainsId 140
  - DeleteAllDemands 141
  - DeleteAllDemandsForCycle 141
  - DeleteDemand 141
  - GetDemandIds 142
  - GetDemandPlanNames 142
  - GetSupplyPlans 143
  - StrictUpdateDemandFromXml 144
  - StrictUpdateDemandFromXmlToCycle 144
  - UpdateDemand 145
  - UpdateDemandFromXml 146
  - UpdateDemandFromXmlToCycle 147
  - WriteDemandToXml 148
  - WriteDemandToXmlFromCycle 148
- desktop configuration 39
- dimension properties 36
- display controls 32
- display types 28
  - Graph 29
  - Table 29
- documentation
  - obtaining updates xv

- ordering printed xv

## E

- editing
  - Metric Exception reports 81
  - Net Change reports 79
  - target rules for relative metrics 70
  - threshold rules 73
- EnterpriseForecast package 59
- error logs 41
- events, clearing log 50
- example batch script 88
- Exception reports 17
- exceptions 33
- ExecuteAllViews Tcl command 245
- ExecuteProcedure Tcl command 128
- ExecuteView Tcl command 245
- executing
  - custom workflows 90
  - data views 68
  - reports 77, 79
- ExportCustomMetric Tcl command 124
- ExportCustomMetrics Tcl command 125
- exporting data 62
- ExportRelativeData Tcl command 246
- ExportTargets Tcl command 246
- ExportThresholds Tcl command 247
- ExportViews Tcl command 247

## F

- fair threshold level 66
- Financial data view 27
- financial reconciliation 14
- fiscal calendars 42
- FixJavaClasspath Tcl command 102

## G

- GetAction Tcl command 111
- GetAllActions Tcl command 112
- GetAllMeetings command 113
- GetAllMilestones command 113
- GetApprovedPlanCyclesInfo Tcl
  - command 133
- GetBeginningInventory Tcl
  - command 150
- GetChannelCodes Tcl command 119
- GetChannelProperties Tcl command 119
- GetChannelPropertyNames Tcl command
  - 120

- GetCompressMode Tcl command 102
- GetCurrentCycleName command 134
- GetCyclePlans Tcl command 248
- GetDBDiskFree Tcl command 103
- GetDemandIds Tcl command 142
- GetDemandPlanNames Tcl command 142
- GetInstallLocation Tcl command 103
- GetItemUom Tcl command 233
- GetKeys Tcl command 248
- GetLogEvents Tcl command 128
- GetMeeting Tcl command 112
- GetMetricValues Tcl command 249
- GetMilestone Tcl command 112
- GetOutput Tcl command 129
- GetPlannedCapacityValues Tcl command 208
- GetProductionIds Tcl command 193
- GetReportsInfo Tcl command 177, 181
- GetReportValues Tcl command 178, 182
- GetResourceCodes Tcl command 209
- GetResourceProperties Tcl command 209
- GetResourcePropertyNames Tcl command 209
- GetSupplyPlanNames Tcl command 230
- GetSupplyPlans Tcl command 143
- GetUomList Tcl command 234
- GetValues Tcl command 125
- GetViewsStatus Tcl command 250
- good threshold level 66
- Graph display type 28, 29

## H

- holding cost metric 24

## I

- icons, Plan Calendar 53
- implementing Sales and Operations Planning 5
- ImportCustomMetrics Tcl command 126
- importing
  - data 1, 7, 59
  - plans 5, 11
  - Sales and Operations Planning Connector 62
- ImportRelativeTargets Tcl command 251
- ImportTargets Tcl command 251
- ImportThresholds Tcl command 252
- ImportViews Tcl command 253
- info messages 41

- inserting threshold rules 72
- integrations
  - Supply Chain Business Modeler 4
- InvalidateAllViews Tcl command 253
- inventory metric 24, 26
- Inventory Plan 13, 24
- InventoryHistory AddInventoryHistory command 149
- InventoryHistory
  - AddInventoryHistoryFromXml command 149
- InventoryHistory
  - DeleteAllInventoryHistories command 150
- InventoryHistory DeleteInventoryHistory command 150
- InventoryHistory GetBeginningInventory command 150
- InventoryHistory
  - WriteInventoryHistoryToXml command 151
- Items AddItem Tcl command 151
- Items AddItemChannelPrice Tcl command 152
- Items AddItemChannelPricesFromXml Tcl command 152
- Items AddItemHoldingCost Tcl command 153
- Items AddItemHoldingCostsFromXml Tcl command 153
- Items AddItemProductionCost Tcl command 154
- Items AddItemProductionCostsFromXml Tcl command 154
- Items AddItemsFromXml Tcl command 155
- Items CheckExists Tcl command 156
- Items commands
  - AddItem 151
  - AddItemChannelPrice 152
  - AddItemChannelPricesFromXml 152
  - AddItemHoldingCost 153
  - AddItemHoldingCostsFromXml 153
  - AddItemProductionCost 154
  - AddItemProductionCostsFromXml 154
  - AddItemsFromXml 155
  - CheckExists 156
  - DeleteAllItems 156
  - DeleteItem 156
  - GetItemCodes 157

- GetItemProperties 157
- GetItemPropertyNames 158
- SetItemProperties 158
- SetItemPropertyNames 158
- StrictUpdateItemsFromXml 159
- UpdateItemsFromXml 160
- WriteItemChannelPricesToXml 160
- WriteItemHoldingCostsToXml 161
- WriteItemProductionCostsToXml 161
- WriteItemsToXml 162
- Items DeleteAllItems Tcl command 156
- Items DeleteItem Tcl command 156
- Items GetItemCodes Tcl command 157
- Items GetItemProperties Tcl command 157
- Items GetItemPropertyNames Tcl command 158
- Items SetItemProperties Tcl command 158
- Items SetItemPropertyNames Tcl command 158
- Items StrictUpdateItemsFromXml Tcl command 159
- Items UpdateItemsFromXml Tcl command 160
- Items WriteItemChannelPricesToXml Tcl command 160
- Items WriteItemHoldingCostsToXml Tcl command 161
- Items WriteItemProductionCostsToXml Tcl command 161
- Items WriteItemsToXml Tcl command 162

**K**

- Key Performance Indicators 1

**L**

- LoadPropertyNames Tcl command 104
- locations 36
- Locations AddLocation Tcl command 162
- Locations AddLocationsFromXml Tcl command 162
- Locations CheckExists Tcl command 163
- Locations commands
  - AddLocation 162
  - AddLocationsFromXml 162

- CheckExists 163
- DeleteAllLocations 163
- DeleteLocation 164
- GetLocationCodes 164
- GetLocationProperties 164
- GetLocationPropertyNames 165
- SetLocationProperties 165
- SetLocationPropertyNames 166
- StrictUpdateLocationsFromXml 167
- UpdateLocationsFromXml 167
- WriteLocationsToXml 168
- Locations DeleteAllLocations Tcl command 163
- Locations DeleteLocation Tcl command 164
- Locations GetLocationCodes Tcl command 164
- Locations GetLocationProperties Tcl command 164
- Locations GetLocationPropertyNames Tcl command 165
- Locations SetLocationProperties Tcl command 165
- Locations SetLocationPropertyNames Tcl command 166
- Locations StrictUpdateLocationsFromXml Tcl command 167
- Locations UpdateLocationsFromXml Tcl command 167
- Locations WriteLocationsToXml Tcl command 168
- Log ClearDbLog Tcl command 168
- Log ClearDbLogOnAndBeforeDate Tcl command 169
- Log ClearFileLog Tcl command 169
- Log commands
  - ClearDbLog 168
  - ClearDbLogOnAndBeforeDate 169
  - ClearFileLog 169
  - GetDbLogEvents 170
  - GetFileLogEvents 170
  - GetLoggingStatus 171
  - LogEvent 171
  - MultiLogEvent 172
  - ResetLoggingSystem 172
  - StartDbLog 173
  - StartFileLog 173
  - StartLogging 173
  - StopAllLogging 174
  - StopLogging 174



- Log GetDbLogEvents Tcl command 170
- Log GetFileLogEvents Tcl command 170
- Log GetLoggingStatus Tcl command 171
- Log LogEvent Tcl command 171
- Log MultiLogEvent Tcl command 172
- log patterns 51
- Log StartDbLog Tcl command 173
- Log StartFileLog Tcl command 173
- Log StartLogging Tcl command 173
- Log StopAllLogging Tcl command 174
- Log StopLogging Tcl command 174
- logs 41
  - adding log patterns 51
  - category patterns 41
  - clearing events 50
  - error messages 41
  - info messages 41
  - viewing workflow 90
  - warning messages 41
  - working with 50

## M

- Manage Planning Cycles page 49
- Manage Planning Periods page 48
- Manage Targets for Absolute Metrics page 71, 72, 73
- Manage Targets for Relative Metrics page 70
- Manage Users page 45, 46, 47
- managing
  - approvals 49
  - Net Change reports 78
  - planning cycles 49
  - planning periods 48
  - sales and operations planning process 53
  - targets for absolute metrics 71
  - targets for relative metrics 69
  - thresholds for absolute metrics 72
- margin per unit metric 27
- margin percentage metric 27
- meetings 53
  - adding to Plan Calendar 56
- Metric Exception Report Manager 81
- Metric Exception Report Manager page 80, 81
- Metric Exception reports 75, 76
  - creating 6, 80
  - editing 81
  - running 81
- MetricExceptionReports commands
  - DeleteAllReports 175
  - DeleteReport 175
  - ExecuteReport 176
  - ExportReports 176
  - GetReportsInfo 177
  - GetReportsStatus 177
  - GetReportValues 178
  - ImportReports 179
- MetricExceptionReports DeleteAllReports Tcl command 175
- MetricExceptionReports DeleteReport Tcl command 175
- MetricExceptionReports ExecuteReport Tcl command 176
- MetricExceptionReports ExportReports Tcl command 176
- MetricExceptionReports GetReportsStatus Tcl command 177
- MetricExceptionReports ImportReports Tcl command 179
- metrics
  - absolute 37
  - adding target rules for absolute metrics 71
  - adding targets for relative metrics 70
  - cost of goods sold per unit 27
  - custom 43
  - demand 26
  - demand fill percentage 26
  - editing relative metric target rules 70
  - holding cost 24
  - inserting absolute metric threshold rules 72
  - inventory 24, 26
  - margin 27
  - margin per unit 27
  - production 26
  - production costs 24
  - relative 37
  - revenue 23, 26
  - revenue per unit 27
  - shortage 24, 26
  - unit demand 23, 26
  - unit production 24
- milestones 53
  - adding to Plan Calendar 56

**N**

- Net Change Report Manager 75
- Net Change Report Manager page 78, 79
- Net Change Report wizard 75, 78
- Net Change reports 17, 75
  - creating 6, 78
  - editing 79
  - running 79
- NetChangeReports commands
  - DeleteAllReports 179
  - DeleteReport 180
  - ExecuteReport 180
  - ExportReports 180
  - GetReportsInfo 181
  - GetReportsStatus 181
  - GetReportValues 182
  - ImportReports 183
- NetChangeReports DeleteAllReports Tcl command 179
- NetChangeReports DeleteReport Tcl command 180
- NetChangeReports ExecuteReport Tcl command 180
- NetChangeReports ExportReports Tcl command 180
- NetChangeReports GetReportsStatus Tcl command 181
- NetChangeReports ImportReports Tcl command 183
- NetProductionRequirements package 59

**O**

- overviews
  - Consolidated view 26
  - data import 59
  - data views 65
  - Demand Plan 23
  - dimension properties 36
  - Financial view 27
  - importing data 59
  - Inventory Plan 24
  - Metric Exception reports \t See 76
  - Production Plan 24
  - reports 75
  - Sales and Operations Planning 1
  - Sales and Operations Planning
    - Automation Shell 83
  - sales and operations planning process 7
  - view formats 31

- views 17

**P**

- packages
  - Base 59
  - BeginningInventory 59
  - EnterpriseForecast 59
  - NetProductionRequirements 59
- passwords
  - changing 47
  - default 39
- patterns
  - adding log 51
  - clearing log 51
- Performance Management business process 2
- period groups 71
- Periods AddPeriod Tcl command 183
- Periods commands
  - AddPeriod 183
  - DeleteAllPeriods 184
  - DeletePeriod 184
  - ExportPeriods 185
  - ImportPeriods 185
- Periods DeleteAllPeriods Tcl command 184
- Periods DeletePeriod Tcl command 184
- Periods ExportPeriods Tcl command 185
- Periods ImportPeriods Tcl command 185
- permissions
  - assigning to views 69
- Plan Calendar 15, 53, 54
  - adding actions 56
  - adding meetings 56
  - adding milestones 56
  - configuring 5
  - understanding 53
- plan data views 17, 22
- plan data, importing 59
- PlanMapping Tcl commands
  - AddPlanMappingFromXml 186
  - WritePlanMappingToXml 186
- PlannedCapacityContainsId Tcl command 210
- planning cycles 42
  - adding 5, 49
  - comparing using Net Change reports 75
  - deleting 49
  - managing 49
  - managing approvals 49

- setting 11
- setting current 49
- planning horizon 48
- planning periods 42
  - adding 5, 48
  - managing 48
- plans
  - assigning to views 11, 68
  - importing 5, 11
- production costs metric 24
- production metric 26
- production plan
  - review 13
- Production Plan
  - review meeting 7
- Production Plan data view 24
- ProductionHistory
  - AddProductionHistoryFromXml
    - command 187
- ProductionHistory
  - DeleteAllProductionHistories
    - command 188
- ProductionHistory
  - DeleteProductionHistory command 188
- ProductionHistory Tcl commands
  - AddProductionHistory 187
- ProductionHistory
  - WriteProductionHistoryToXml
    - command 189
- ProductionPlan AddProduction Tcl
  - command 189
- ProductionPlan AddProductionFromXml
  - Tcl command 190
- ProductionPlan
  - AddProductionFromXmlToCycle
    - Tcl command 190
- ProductionPlan commands
  - AddProduction 189
  - AddProductionFromXml 190
  - AddProductionFromXmlToCycle 190
  - ContainsId 191
  - DeleteAllProductions 192
  - DeleteAllProductionsForCycle 192
  - DeleteProduction 192
  - GetProductionIds 193
  - StrictUpdateProductionFromXml 193
  - StrictUpdateProductionFromXmlToCycle 194
  - UpdateProduction 195
  - UpdateProductionFromXml 196
  - UpdateProductionFromXmlToCycle 197

- WriteProductionToXml 197
- WriteProductionToXmlFromCycle 198
- ProductionPlan ContainsId Tcl command 191
- ProductionPlan DeleteAllProductions Tcl command 192
- ProductionPlan
  - DeleteAllProductionsForCycle Tcl command 192
- ProductionPlan DeleteProduction Tcl command 192
- ProductionPlan
  - StrictUpdateProductionFromXml
    - Tcl command 193
- ProductionPlan
  - StrictUpdateProductionFromXmlToCycle
    - Tcl command 194
- ProductionPlan UpdateProduction Tcl command 195
- ProductionPlan
  - UpdateProductionFromXml Tcl command 196
- ProductionPlan
  - UpdateProductionFromXmlToCycle Tcl command 197
- ProductionPlan WriteProductionToXml Tcl command 197
- ProductionPlan
  - WriteProductionToXmlFromCycle
    - Tcl command 198
- products 36

## R

- recording milestones and action items 53
- Refresh command
  - syntax 61
- relative metrics 37, 66
  - adding rules 70
  - changing order of target rules 70
  - editing target rules 70
- Report Manager 77
- reports 75
  - creating 6
  - creating Metric Exception 80
  - creating Net Change 78
  - editing 79
  - editing Metric Exception 81
  - Exception 17
  - executing 77
  - executing Metric Exception 81

- Metric Exception 76
  - Net Change 17
  - overview 75
  - running 79
  - RequiredCapacityContainsId Tcl
    - command 210
  - ResetLoggingSystem Tcl command 172
  - resources 36
  - Resources command
    - AddPlannedCapacityFromXml 199
  - Resources commands
    - AddPlannedCapacity 198
    - AddRequiredCapacity 200
    - AddRequiredCapacityFromXml 200
    - AddRequiredCapacityFromXmlToCycle 201
    - AddResource 202
    - AddResourceMaximumCapacitiesFromXml 202
    - AddResourceMaximumCapacity 203
    - AddResourcesFromXml 203
    - CheckExists 204
    - DeleteAllPlannedCapacityForCycle 205
    - DeleteAllRequiredCapacities 205
    - DeleteAllRequiredCapacitiesForCycle 206
    - DeleteAllResources 206
    - DeletePlannedCapacity 205, 207
    - DeleteRequiredCapacity 207
    - DeleteResource 208
    - GetPlannedCapacityValues 208
    - GetResourceCodes 209
    - GetResourceProperties 209
    - GetResourcePropertyNamees 209
    - PlannedCapacityContainsId 210
    - RequiredCapacityContainsId 210
    - SetResourceProperties 211
    - SetResourcePropertyNamees 211
    - StrictUpdatePlannedCapacityFromXml 212
    - StrictUpdatePlannedCapacityFromXmlToCycle 213
    - StrictUpdateRequiredCapacityFromXml 213
    - StrictUpdateRequiredCapacityFromXmlToCycle 214
    - StrictUpdateResourcesFromXml 215
    - UpdatePlannedCapacity 216
    - UpdatePlannedCapacityFromXml 216
    - UpdatePlannedCapacityFromXmlToCycle 217
    - UpdateRequiredCapacity 218
    - UpdateRequiredCapacityFromXml 219
    - UpdateRequiredCapacityFromXmlToCycle 220
    - UpdateResourcesFromXml 221
    - WritePlannedCapacityToXml 221
    - WritePlannedCapacityToXmlFromCycle 222
    - WriteRequiredCapacityToXml 222
    - WriteRequiredCapacityToXmlFromCycle 223
    - WriteResourceMaximumCapacitiesToXml 224
    - WriteResourcesToXml 224
  - Restore Tcl command 104
  - revenue metric 23, 26
  - revenue per unit 27
  - reviews
    - Demand Plan 11
    - Production Plan 13
  - Rough Cut Capacity Plan 13
  - rules
    - adding for absolute metrics 71
    - inserting new threshold rules for absolute metrics 72
  - rules, targets and thresholds 66
  - RunBackground Tcl command 105
  - running
    - batch scripts 89
    - data views 68
    - Metric Exception reports 81
- S**
- Sales and Operations Planning 39
    - business processes 2
    - Connector 59
    - data import requirements 60
    - implementation 5
    - integrations 4
    - objectives 7
    - overview 1
    - senior management meeting 7
    - user interface 15
    - workflow 9
  - Sales and Operations Planning Automation
    - Shell 1, 83
      - batch scripts 88
      - syntax 83
    - Tcl commands 90
  - Sales and Operations Planning Automation
    - Shell (SASH) 62
  - Sales and Operations Planning Automation
    - Shell Sales and Operations Planning
      - Automation Shell
        - starting 86
  - Sales and Operations Planning
    - Connector 62
  - sales and operations planning process
    - financial reconciliation 14
    - overview 7
    - senior management meeting 15

SalesHistory AddSalesHistoryPrices  
     command 225  
 SalesHistory AddSalesHistoryUnits  
     command 226  
 SalesHistory commands  
     AddSalesHistoryFromXml 224  
     DeleteAllSalesHistories 226  
     DeleteSalesHistoryPrices 227  
     DeleteSalesHistoryUnits 227  
     UpdateSalesHistoryFromXml 228  
     WriteSalesHistoryToXml 229  
 SASH 1, 83  
     starting 62, 86  
     syntax 83  
 SASH commands  
     App Backup 62  
     refresh 61  
 SCBM 4, 5  
     Demand model 60  
     importing plan data 59  
     Strategic model 60  
     Tactical model 60  
 senior management meeting 15  
 SetApprovedPlan Tcl command 134  
 SetChannelProperties Tcl command 120  
 SetCompressMode Tcl command 105  
 SetCurrentCycle Tcl command 134  
 SetItemUom Tcl command 234  
 SetPlansForViewCycle Tcl command 253  
 SetResourceProperties Tcl command 211  
 SetResourcePropertyNames Tcl  
     command 211  
 setting  
     current cycle 11  
     current planning cycle 49  
     targets and thresholds 6, 33, 37, 66  
     thresholds 70  
 SetValues Tcl command 127  
 shortage metric 24, 26  
 starting  
     Sales and Operations Planning  
         Automation Shell 86  
         Sales and Operations Planning Connector  
             62  
 Strategic Network Optimization 1  
 StrictUpdateChannelsFromXml Tcl  
     command 121  
 StrictUpdateCyclesFromXml  
     command 135

StrictUpdateDemandFromXml Tcl  
     command 144  
 StrictUpdateDemandFromXmlToCycle Tcl  
     command 144  
 StrictUpdatePlannedCapacityFromXml Tcl  
     command 212  
 StrictUpdatePlannedCapacityFromXmlToCycle  
     Tcl command 213  
 StrictUpdateRequiredCapacityFromXml  
     Tcl command 213  
 StrictUpdateRequiredCapacityFromXmlToCycle  
     Tcl command 214  
 StrictUpdateResourcesFromXml Tcl  
     command 215  
 Supply Chain Business Modeler  
     importing plans 5  
     integrations 4  
 SupplyPlan AddSupplyPlan  
     command 229  
 SupplyPlan DeleteSupplyPlan  
     command 230  
 SupplyPlan GetSupplyPlanNames  
     command 230  
 syntax of SASH commands 83

## T

Table display type 28, 29  
 targets 37  
     adding for absolute metrics 71  
     adding for relative metrics 70  
     changing order 70  
     displaying in Dashboard 33  
     editing 70  
     setting 6  
 Tcl commands 83, 88  
     AddConsolidatedView 240  
     AddCyclesFromXml 130  
     AddDemand 137  
     AddDemandFromXml 138  
     AddDemandFromXmlToCycle 138  
     AddDemandPlanView 241  
     AddFinancialView 241  
     AddInventoryView 242  
     AddItem 151  
     AddItemChannelPrice 152  
     AddItemChannelPricesFromXml 152  
     AddItemHoldingCost 153  
     AddItemHoldingCostsFromXml 153  
     AddItemProductionCost 154  
     AddItemProductionCostsFromXml 154

- AddItemsFromXml 155
- AddLocation 162
- AddLocationsFromXml 162
- AddMeeting 107
- AddPeriod 183
- AddPlanMappingFromXml 131, 186
- AddProductionPlanView 243
- AddRequiredCapacity 200
- AddRoughCutCapacityView 243
- AddToCalendarFromXml 109
- AddUser 235
- AddUsersFromXml 235
- App Archive 100
- App Backup 101
- App Clear 101
- App DatabaseExists 102
- App FixJavaClasspath 102
- App GetCompressMode 102
- App GetDBDiskFree 103
- App GetInstallLocation 103
- App GetVersion 104
- App LoadPropertyNames 104
- App Restore 104
- App RunBackground 105
- App SetCompressMode 105
- App WritePropertyNames 106
- Calendar AddAction 106
- Calendar AddMilestone 108
- Calendar Clear 109
- Calendar GetAllActoions
  - command 112
- Calendar GetAllMeetings 113
- Calendar GetAllMilestones 113
- Calendar UpdateAction 114
- Calendar UpdateMeeting 114
- Calendar UpdateMilestone 115
- Calendar WriteCalendarToXml 116
- Channels AddChannel 116
- ChannelsAddChannelsFromXml 117
- CheckExists 118, 156, 163
- ClearDbLog 168
- ClearDbLogOnAndBeforeDate 169
- ClearFileLog 169
- ContainsId 140
- CustomMetrics AddCustomMetrics
  - command 123
- CustomMetrics
  - DeleteCustomMetric 124
- CustomMetrics GetValues 125
- CustomMetrics
  - ImportCustomMetrics 126
  - CustomMetrics SetValues 127
- CustomWorkflows
  - ExecuteProcedure 128
- CustomWorkflows GetLogEvents 128
- CustomWorkflows GetOutput 129
- Cycles AddCycle 129
- Cycles Contains 131
- Cycles
  - GetApprovedPlanCyclesInfo 133
- Cycles GetCurrentCycleName 134
- Cycles SetApprovedPlan 134
- Cycles SetCurrentCycle 134
- Cycles WriteCyclesToXml 136
- DeleteAction 110
- DeleteAllChannels 118
- DeleteAllCycles 132
- DeleteAllDemands 141
- DeleteAllDemandsForCycle 141
- DeleteAllItems 156
- DeleteAllLocations 163
- DeleteAllPeriods 184
- DeleteAllReports 175, 179
- DeleteApprovedPlan 132
- DeleteChannel 118
- DeleteCycle 132
- DeleteDemand 141
- DeleteItem 156
- DeleteLocation 164
- DeleteMeeting 110
- DeleteMilestone 111
- DeletePeriod 184
- DeleteReport 175, 180
- DeleteUser 236
- DeleteView 245
- DemandPlan AddDemandPlan 139
- DemandPlan GetDemandIds 142
- DemandPlan
  - GetDemandPlanNames 142
- DemandPlan GetSupplyPlans 143
- ExecuteAllViews 245
- ExecuteReport 176, 180
- ExportCustomMetric 124
- ExportCustomMetrics 125
- ExportPeriods 185
- ExportReports 176, 180
- GetAction 111
- GetChannelCodes 119
- GetChannelProperties 119

- GetChannelPropertyNames 120
- GetDbLogEvents 170
- GetFileLogEvents 170
- GetItemCodes 157
- GetItemProperties 157
- GetItemPropertyNames 158
- GetKeys 248
- GetLocationCodes 164
- GetLocationProperties 164
- GetLocationPropertyNames 165
- GetLoggingStatus 171
- GetMeeting 112
- GetMetricValues 249
- GetMilestone 112
- GetReportsStatus 177, 181
- GetViewsStatus 250
- ImportPeriods 185
- ImportRelativeTargets 251
- ImportReports 179, 183
- ImportTargets 251
- ImportThresholds 252
- ImportViews 253
- InvalidateAllViews 253
- InventoryHistory
  - AddInventoryHistory 149
- InventoryHistory
  - AddInventoryHistoryFromXml 149
- InventoryHistory
  - DeleteAllInventoryHistories 150
- InventoryHistory
  - DeleteInventoryHistory 150
- InventoryHistory
  - GetBeginningInventory 150
- InventoryHistory
  - WriteInventoryHistoryToXml 151
- IsAdmin 237
- IsViewAdmin 237
- listing by category 90
- Log ResetLoggingSystem 172
- LogEvent 171
- MetricExceptionReports
  - GetReportsInfo 177
- MetricExceptionReports
  - GetReportValues 178
- MultiLogEvent 172
- NetChangeReports
  - GetReportsInfo 181
- NetChangeReports
  - GetReportValues 182
- ProductionHistory
  - AddProductionHistory 187
- ProductionHistory
  - AddProductionHistoryFromXml 187
- ProductionHistory
  - DeleteAllProductionHistories 188
- ProductionHistory
  - DeleteProductionHistory 188
- ProductionHistory
  - WriteProductionHistoryToXml 189
- ProductionPlan AddProduction 189
- ProductionPlan AddProductionFromXml 190
- ProductionPlan
  - AddProductionFromXmlToCycle 190
- ProductionPlan ContainsId 191
- ProductionPlan DeleteAllProductions 192
- ProductionPlan
  - DeleteAllProductionsForCycle 192
- ProductionPlan DeleteProduction 192
- ProductionPlan GetProductionIds 193
- ProductionPlan
  - StrictUpdateProductionFromXml 193
- ProductionPlan
  - StrictUpdateProductionFromXmlToCycle 194
- ProductionPlan UpdateProduction 195
- ProductionPlan
  - UpdateProductionFromXml 196
- ProductionPlan
  - UpdateProductionFromXmlToCycle 197
- ProductionPlan WriteProductionToXml 197
- ProductionPlan
  - WriteProductionToXmlFromCycle 198
- RequiredCapacityContainsId 210
- Resources AddPlannedCapacity 198
- Resources
  - AddPlannedCapacityFromXml 199
- Resources
  - AddRequiredCapacityFromXml 200
- Resources
  - AddRequiredCapacityFromXmlToCycle 201

- Resources AddResource 202
- Resources
  - AddResourceMaximumCapacitiesFromXml 202
- Resources
  - AddResourceMaximumCapacity 203
- Resources
  - AddResourcesFromXml 203
- Resources CheckExists 204
- Resources
  - DeleteAllPlannedCapacityForCycle 205
- Resources
  - DeleteAllRequiredCapacities 205
- Resources
  - DeleteAllRequiredCapacitiesForCycle 206
- Resources DeleteAllResources 206
- Resources DeletePlannedCapacity 205, 207
- Resources
  - DeleteRequiredCapacity 207
- Resources DeleteResource 208
- Resources
  - GetPlannedCapacityValues 208
- Resources GetResourceCodes 209
- Resources GetResourceProperties 209
- Resources
  - GetResourcePropertyNames 209
- Resources
  - PlannedCapacityContainsId 210
- Resources SetResourceProperties 211
- Resources
  - SetResourcePropertyNames 211
- Resources
  - StrictUpdateRequiredCapacityFromXml 213
- Resources
  - StrictUpdateRequiredCapacityFromXmlToCycle 214
- Resources
  - StrictUpdateResourcesFromXml 215
- Resources
  - UpdatePlannedCapacity 216
- Resources
  - UpdatePlannedCapacityFromXml 216
- Resources
  - UpdatePlannedCapacityFromXmlToCycle 217
- Resources
  - UpdateRequiredCapacity 218
- Resources
  - UpdateRequiredCapacityFromXml 219
- Resources
  - UpdateRequiredCapacityFromXmlToCycle 220
- Resources
  - UpdateResourcesFromXml 221
- Resources
  - WritePlannedCapacityToXml 221
- Resources
  - WritePlannedCapacityToXmlFromXml 222
- Resources
  - WriteRequiredCapacityToXml 222
- Resources
  - WriteRequiredCapacityToXmlFromCycle 223
- Resources WriteResourcesToXml 224
- SalesHistory
  - AddSalesHistoryFromXml 224
- SalesHistory
  - AddSalesHistoryPrices 225
- SalesHistory
  - AddSalesHistoryUnits 226
- SalesHistory
  - DeleteAllSalesHistories 226
- SalesHistory
  - DeleteSalesHistoryPrices 227
- SalesHistory
  - DeleteSalesHistoryUnits 227
- SalesHistory
  - UpdateSalesHistoryFromXml 228
- SalesHistory
  - WriteSalesHistoryToXml 229
- SetChannelProperties 120
- SetChannelPropertyNames 120
- SetItemProperties 158
- SetItemPropertyNames 158
- SetLocationProperties 165
- SetLocationPropertyNames 166
- SetPassword 238
- StartDbLog 173
- StartFileLog 173
- StartLogging 173
- StopAllLogging 174
- StopLogging 174
- StrictUpdateChannelsFromXml 121
- StrictUpdateCyclesFromXml 135
- StrictUpdateDemandFromXml 144
- StrictUpdateDemandFromXmlToCycle 144
- StrictUpdateItemsFromXml 159
- StrictUpdateLocationsFromXml 167
- StrictUpdatePlannedCapacityFromXml 212
- StrictUpdatePlannedCapacityFromXmlToCycle 212
- StrictUpdateUsersFromXml 238
- SupplyPlan AddSupplyPlan 229
- SupplyPlan DeleteSupplyPlan 230



- SupplyPlan GetSupplyPlanNames 230
  - UnitsOfMeasure
    - AddItemUomFromXml 231
    - UnitsOfMeasure AddUom 232
    - UnitsOfMeasure Clear 232
    - UnitsOfMeasure DeleteUom 232
    - UnitsOfMeasure GetItemUom 233
    - UnitsOfMeasure GetUomList 234
    - UnitsOfMeasure SetItemUom 234
    - UnitsOfMeasure
      - WriteItemUomToXml 234
    - UpdateChannelsFromXml 122
    - UpdateCyclesFromXml 136
    - UpdateDemand 145
    - UpdateDemandFromXml 146
    - UpdateDemandFromXmlToCycle 147
    - UpdateItemsFromXml 160
    - UpdateLocationsFromXml 167
    - UpdateUsersFromXml 239
    - ViewExportRelativeTargets 246
    - Views DeleteAllViews 244
    - Views ExecuteView 245
    - Views ExportTargets 246
    - Views ExportThresholds 247
    - Views ExportViews 247
    - Views GetCyclePlans 248
    - Views SetPlansForViewCycle 253
    - WriteChannelsToXml 122
    - WriteDemandToXml 148
    - WriteDemandToXmlFromCycle 148
    - WriteItemChannelPricesToXml 160
    - WriteItemHoldingCostsToXml 161
    - WriteItemProductionCostsToXml 161
    - WriteItemsToXml 162
    - WriteLocationsToXml 168
    - WritePlanMappingToXml 137
    - WriteResourceMaximumCapacitiesToXml 224
    - WriteUsersToXml 239
  - threshold rules
    - adding 72
    - editing 73
  - thresholds 37, 70
    - changing order 73
    - displaying in Dashboard 33
  - typographical conventions xvi
- U**
- unit demand metric 23, 26
  - unit production metric 24
  - units of measure 35
  - UnitsOfMeasure AddItemUomFromXml
    - command 231
  - UnitsOfMeasure AddUom command 232
  - UnitsOfMeasure Clear command 232
  - UnitsOfMeasure DeleteUom
    - command 232
  - UnitsOfMeasure GetItemUom
    - command 233
  - UnitsOfMeasure GetUomList
    - command 234
  - UnitsOfMeasure SetItemUom
    - command 234
  - UnitsOfMeasure WriteItemUomToXml
    - command 234
  - UpdateAction Tcl command 114
  - UpdateChannelsFromXml Tcl command 122
  - UpdateCyclesFromXml command 136
  - UpdateDemand Tcl command 145
  - UpdateDemandFromXml Tcl command 146
  - UpdateDemandFromXmlToCycle Tcl
    - command 147
  - UpdateMeeting Tcl command 114
  - UpdateMilestone Tcl command 115
  - UpdatePlannedCapacity Tcl
    - command 216
  - UpdatePlannedCapacityFromXml Tcl
    - command 216
  - UpdatePlannedCapacityFromXmlToCycle
    - Tcl command 217
  - UpdateRequiredCapacity Tcl
    - command 218
  - UpdateRequiredCapacityFromXml Tcl
    - command 219
  - UpdateRequiredCapacityFromXmlToCycle
    - Tcl command 220
  - UpdateResourcesFromXml Tcl
    - command 221
  - UpdateSalesHistoryFromXml Tcl
    - command 228
  - user administration 39, 45
  - user interface 15
  - user roles 19
  - users
    - adding 5, 46
    - changing passwords 47
    - deleting accounts 46
  - Users AddUser Tcl command 235

- Users AddUsersFromXml Tcl command 235
- Users commands
  - AddUser 235
  - AddUsersFromXml 235
  - DeleteUser 236
  - IsAdmin 237
  - IsViewAdmin 237
  - SetPassword 238
  - StrictUpdateUsersFromXml 238
  - UpdateUsersFromXml 239
  - WriteUsersToXml 239
- Users DeleteUser Tcl command 236
- Users IsAdmin Tcl command 237
- Users IsViewAdmin Tcl command 237
- Users SetPassword Tcl command 238
- Users StrictUpdateUsersFromXml Tcl command 238
- Users UpdateUsersFromXml Tcl command 239
- Users WriteUsersToXml Tcl command 239

## V

- View Event Log page 50
- view formats 31
  - Comparison 32
  - Dashboard 11, 13, 33
  - Default 32
  - Waterfall 34
- View Manager 17, 19, 65, 68
- View Manager page 67
- view permissions 18, 19
- viewing
  - workflow logs 90
  - workflow output 90
- viewing exceptions 33
- views
  - assigning permissions 69
  - assigning plans 11
  - Comparison formats 32
  - Consolidated 26
  - creating 67
  - Dashboard format 33
  - Demand Plan 23
  - display controls 32
  - Financial 27
  - Inventory Plan 24
  - overviews 17
  - Production Plan 24

- Rough Cut Capacity Plan 13
  - view formats 31
  - Waterfall format 34
- Views commands
  - AddConsolidatedView command 240
  - AddDemandPlanView 241
  - AddFinancialView 241
  - AddInventoryView 242
  - AddProductionPlanView 243
  - AddRoughCutCapacityView 243
  - DeleteAllViews 244
  - DeleteView 245
  - ExecuteAllViews 245
  - ExecuteView 245
  - ExportRelativeData 246
  - ExportTargets 246
  - ExportThresholds 247
  - ExportViews 247
  - GetCyclePlans 248
  - GetKeys 248
  - GetMetricValues 249
  - GetViewsStatus 250
  - ImportRelativeTargets 251
  - ImportTargets 251
  - ImportThresholds 252
  - ImportViews 253
  - InvalidateAllViews 253
  - SetPlansForViewCycle 253
- visual cues xvi

## W

- warning messages 41
- Waterfall format 34
- web server 5
- wildcard patterns 51
- wizards
  - Add New Report 79, 80, 81
  - Add View 67
  - Net Change Report 78
- workflows 89
  - adding 90
  - executing 90
  - viewing logs 90
  - viewing output 90
- WriteCalendarToXml Tcl command 116
- WriteChannelsToXml Tcl command 122
- WriteCyclesToXml Tcl command 136
- WriteDemandToXml Tcl command 148
- WriteDemandToXmlFromCycle Tcl command 148

WriteInventoryHistoryToXml Tcl  
     command 151  
 WriteItemUomToXml Tcl command 234  
 WritePlanMappingToXml Tcl  
     command 137, 186  
 WritePlannedCapacityToXml Tcl  
     command 221  
 WritePlannedCapacityToXmlFromCycle  
     Tcl command 222  
 WriteProductionHistoryToXml Tcl  
     command 189  
 WritePropertyNames Tcl command 106  
 WriteRequiredCapacityToXml Tcl  
     command 222  
 WriteRequiredCapacityToXmlFromCycle  
     Tcl command 223  
 WriteResourceMaximumCapacitiesToXml  
     Tcl command 224  
 WriteResourcesToXml Tcl command 224  
 WriteSalesHistoryToXml Tcl  
     command 229

## X

XML files 59  
     Base.xml 60  
     BeginningInventory.xml 60  
     Config.xml 61  
     EnterpriseForecast.xml 60  
     NetProduction Requirements.xml 60

