
AIMMS Tutorial for Professionals - Data Management

This file contains only one chapter of the book. For a free download of the complete book in pdf format, please visit www.aimms.com

Copyright © 1993–2011 by Paragon Decision Technology B.V. All rights reserved.

Paragon Decision Technology B.V.	Paragon Decision Technology Inc.	Paragon Decision Technology Pte.
Schipholweg 1	500 108th Avenue NE	Ltd.
2034 LS Haarlem	Ste. # 1085	80 Raffles Place
The Netherlands	Bellevue, WA 98004	UOB Plaza 1, Level 36-01
Tel.: +31 23 5511512	USA	Singapore 048624
Fax: +31 23 5511517	Tel.: +1 425 458 4024	Tel.: +65 9640 4182
	Fax: +1 425 458 4025	

Email: info@aimms.com
WWW: www.aimms.com

AIMMS is a registered trademark of Paragon Decision Technology B.V. IBM ILOG CPLEX and sc CPLEX is a registered trademark of IBM Corporation. GUROBI is a registered trademark of Gurobi Optimization, Inc. KNITRO is a registered trademark of Ziena Optimization, Inc. XPRESS-MP is a registered trademark of FICO Fair Isaac Corporation. MOSEK is a registered trademark of Mosek ApS. WINDOWS and EXCEL are registered trademarks of Microsoft Corporation. $\text{T}_{\text{E}}\text{X}$, $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$, and $\text{A}_{\text{M}}\text{S}_{\text{L}}\text{A}_{\text{T}}\text{E}_{\text{X}}$ are trademarks of the American Mathematical Society. LUCIDA is a registered trademark of Bigelow & Holmes Inc. ACROBAT is a registered trademark of Adobe Systems Inc. Other brands and their products are trademarks of their respective holders.

Information in this document is subject to change without notice and does not represent a commitment on the part of Paragon Decision Technology B.V. The software described in this document is furnished under a license agreement and may only be used and copied in accordance with the terms of the agreement. The documentation may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from Paragon Decision Technology B.V.

Paragon Decision Technology B.V. makes no representation or warranty with respect to the adequacy of this documentation or the programs which it describes for any particular purpose or with respect to its adequacy to produce any particular result. In no event shall Paragon Decision Technology B.V., its employees, its contractors or the authors of this documentation be liable for special, direct, indirect or consequential damages, losses, costs, charges, claims, demands, or claims for lost profits, fees or expenses of any nature or kind.

In addition to the foregoing, users should recognize that all complex software systems and their documentation contain errors and omissions. The authors, Paragon Decision Technology B.V. and its employees, and its contractors shall not be responsible under any circumstances for providing information or corrections to errors and omissions discovered at any time in this book or the software it describes, whether or not they are aware of the errors or omissions. The authors, Paragon Decision Technology B.V. and its employees, and its contractors do not recommend the use of the software described in this book for applications in which errors or omissions could threaten life, injury or significant loss.

This documentation was typeset by Paragon Decision Technology B.V. using $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ and the LUCIDA font family.

Chapter 14

Data Management

In this chapter, you will learn how to manage your model data using *cases* and/or *datasets*. Such management is typically based on using menu commands. You will also write a procedure to generate cases automatically during an AIMMS session. These cases are then viewed and compared in a multiple case overview.

This chapter

14.1 Storing the solution in a case

A *case* is a set of data values at an instant in time and contains the values of a subset of all model identifiers. Such a subset is referred to as a *case type*. The default case type is the set of all identifiers. Cases enable you to save intermediate data values for inspection at a later moment. You can also use a case to continue your work during a later AIMMS session.

What is a case?

Following an iteration of the rolling horizon process, initiated by pressing the **Run Next** button, you can save both your input and the solution values in a new case by executing the following steps:

Creating a case

- ▶ select the **Save Case as...** command from the **Data** menu,
- ▶ specify 'Solution After First Roll' (without the quotes) in the 'Name' edit field, and
- ▶ press the **Save** button (see Figure 14.1).



Figure 14.1: Creating your first case

The following commands close and re-open your AIMMS project. Then, by loading the case you have just saved, you will have incorporated all your current data values. Please follow these instructions:

Loading a case

- ▶ change to the default page menubar by setting the current page to **Edit** mode,
- ▶ select the **Close Project** command from the **File** menu,
- ▶ open the project again,
- ▶ select the **Load Case** submenu from the **Data** menu,
- ▶ select the **as Active...** command,
- ▶ select the 'Solution After First Roll' entry from the list box, and
- ▶ press the **Load** button (see Figure 14.2).

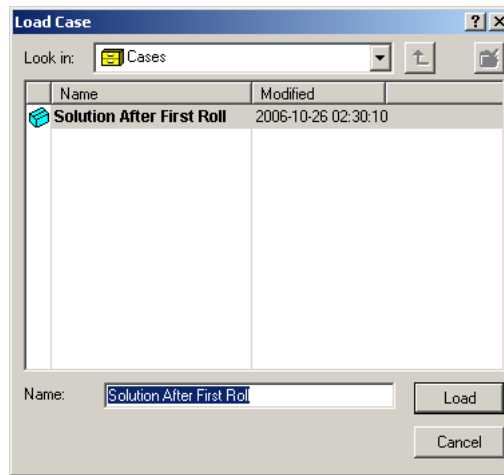


Figure 14.2: Loading your first case

In AIMMS, all the data that you are currently working with are referred to as the *active case*. The name of the currently active case is displayed in the status bar at the bottom of the AIMMS window as shown in Figure 14.3.

The active case

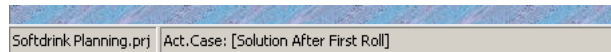


Figure 14.3: Part of the AIMMS status bar

14.2 Saving holidays and vacations in a dataset

A *dataset* contains the data values of a subset of all model identifiers at a particular instance. Such a subset is referred to as a *data category* and typically contains model identifiers associated with a particular functional aspect of your application.



What is a dataset?

Note that datasets and data categories are similar to cases and case types. The major difference is that a case type can include one or more data categories and, as a consequence, cases can be built up from multiple datasets. Such advanced use of case types and data categories is not illustrated in this tutorial but can be found in Chapter 17 of *The User's Guide*.

Cases versus datasets

Data categories are managed using the **Data Management Setup** tool. To create a data category describing the vacation weeks and official holidays, you should perform the following actions:

Creating a data category ...

- ▶ press the **Data Management Setup** tool button  on the toolbar,
- ▶ select the 'Data categories' node in the data management setup tree,
- ▶ press the **New Data Category** button  on the toolbar,
- ▶ specify 'Absentee Data' as the name for the data category, and
- ▶ press the *Enter* key to register this name.

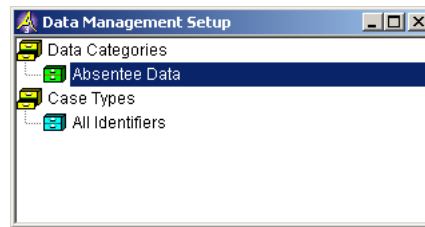

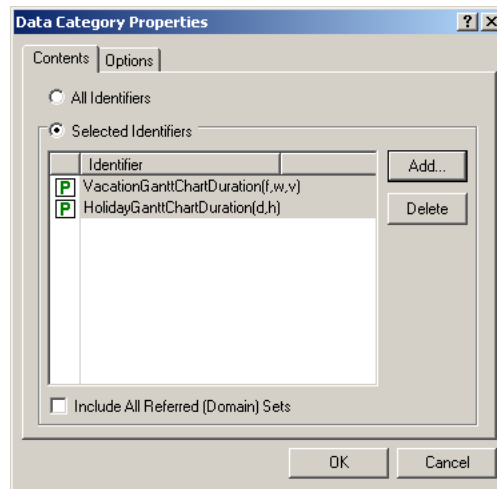


Figure 14.4: The data management setup tree

To specify which model identifiers are to be stored in the new dataset you need to take the following actions:

... and specifying its contents

- ▶ select the 'Absentee Data' node if necessary,
- ▶ press the **Properties** button  on the toolbar,
- ▶ press the **Add** button on the **Contents** tab of the **Data Category Properties** dialog box,
- ▶ select `HolidayGanttChartDuration` and `VacationGanttChartDuration` using the *Ctrl* key (both located inside the Absentee Overview of The User Interface),
- ▶ press the **OK** button (you can compare your screen to the dialog box in Figure 14.5), and
- ▶ press the **OK** button again.



Figure 14.5: The **Contents** tab of the **Dataset Properties** dialog box

Next, you should open the *Absentee Overview* page in **User** mode, and specify the vacation weeks and official holidays as listed in Table 14.1 by clicking on the two Gantt charts. *Specifying a dataset ...*

Eindhoven	Vacation Weeks		Official Holidays
	Haarlem	Zwolle	
week 27, 2000	week 30, 2000	week 29, 2000	Dec 25, 2000
week 28, 2000	week 31, 2000	week 30, 2000	Dec 26, 2000
week 29, 2000	week 32, 2000	week 31, 2000	Jan 1, 2001
week 30, 2000	week 33, 2000	week 32, 2000	Apr 15, 2001
week 31, 2000	week 34, 2000	week 33, 2000	Apr 16, 2001
week 32, 2000	week 35, 2000	week 34, 2000	Apr 30, 2001
week 33, 2000	week 36, 2000	week 35, 2000	May 5, 2001
week 34, 2000	week 37, 2000	week 36, 2000	May 24, 2001
week 50, 2000	week 50, 2000	week 50, 2000	Jun 3, 2001
week 51, 2000	week 51, 2000	week 51, 2000	Jun 4, 2001
week 52, 2000	week 52, 2000	week 52, 2000	
week 7, 2001	week 9, 2001	week 8, 2001	
week 8, 2001	week 10, 2001	week 9, 2001	
week 9, 2001	week 11, 2001	week 10, 2001	
week 10, 2001	week 12, 2001	week 11, 2001	

Table 14.1: Vacation weeks and official holidays

To save the holiday and vacation data you have just specified, perform the following actions: ... and saving it

- ▶ press the **Data Manager** button  on the toolbar,
- ▶ select the 'Datasets' entry from the data tree,
- ▶ select the 'Absentee Data' data category,
- ▶ open this data category by double clicking on the icon,
- ▶ press the **New Dataset** button  on the toolbar,
- ▶ specify 'Holiday and Vacation Data' as the name of the new dataset,
- ▶ press the *Enter* key, and
- ▶ select the **Save into Selected Node** command from the **Data** menu on your page.

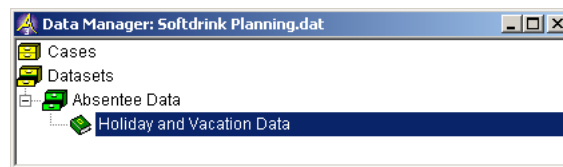


Figure 14.6: The data management tree

Alternatively, you could have saved the dataset using the **Save Dataset** or **Save Dataset as ...** menu commands from the **Data** menu.

To load the above dataset automatically during project startup, you first need to specify a procedure that loads the dataset. Please create a model section named Data Management directly underneath the section Softdrink Planning Menubar. In this section, declare a procedure named LoadHolidayAndVacationDataset. Local to this procedure you should create an element parameter DatasetReference with **Range** attribute AllDatasets (see Figure 14.7). Next, you should specify the following execution statements as its **Body** attribute:

Writing a loading procedure ...

```
if ( not DatasetFind('Absentee Data', "Holiday and Vacation Data", DatasetReference) )
then DialogMessage( "Error loading holiday and vacation data." );
    return;
endif;

DatasetLoadCurrent( 'Absentee Data', DatasetReference, 0 );
```

To obtain an explanation of the above predefined AIMMS functions, you can open *The Function Reference* from within AIMMS by first placing the text cursor on a particular function name and then selecting the **Help On** command from the right-mouse pop-up menu.

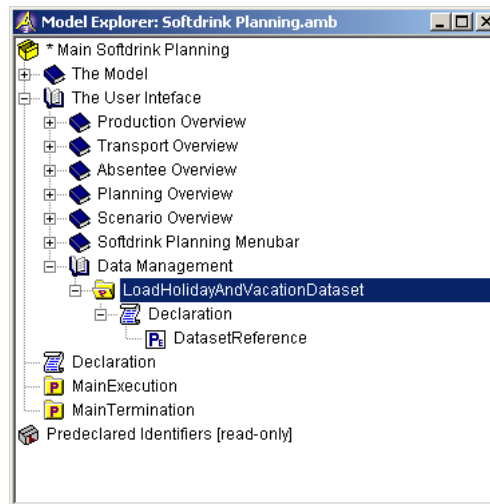


Figure 14.7: The procedure `LoadHolidayAndVacationDataset` in the model tree

To make the procedure `LoadHolidayAndVacationDataset` the startup procedure, you should follow the same steps used when you specified a startup page at the end of Chapter ?? . The corresponding **Options** dialog box is shown in Figure 14.8.

... to become the startup procedure

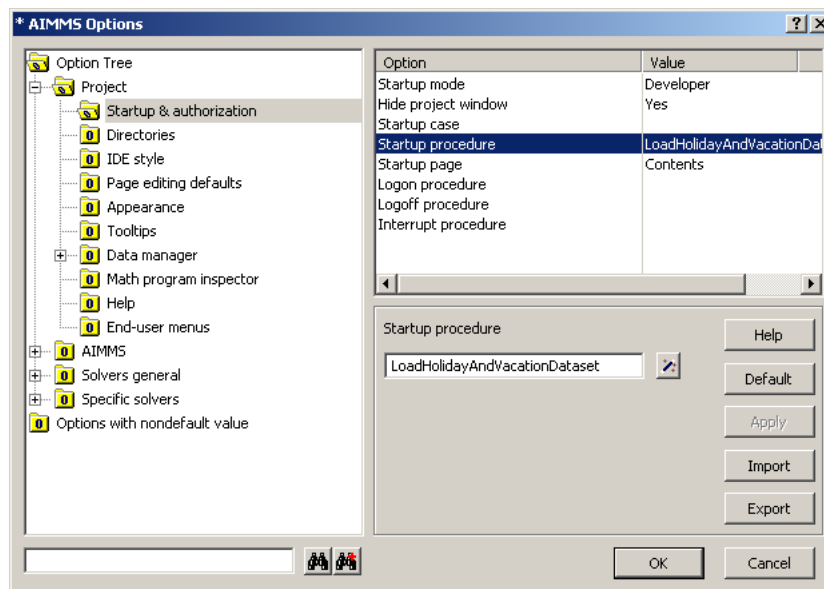


Figure 14.8: The AIMMS Options dialog box

14.3 Automatic case generation

In this section, you will first build your own procedure that automatically generates cases. After this, you will develop an experiment in which you will study the effect of the length of the planning horizon on the total cost of running the company. Finally, you will create a multiple case object to view and compare the results of this investigation.

This section

In a typical ‘What If’ experiment, you want to study the output of your model as a result of changes in data input. You can perform such an experiment through an interactive session. If the experiment is extensive and/or requires a great deal of CPU time, an alternative approach is to write a procedure to execute the entire experiment. It is then important to save the results in cases that are generated as the experiment evolves. The following paragraphs will show you how to construct an extensive experiment using an automatic case saving procedure.

*‘What If’
experiments*

The total cost of running the company will be the output of an experiment in which the length of the planning horizon is changed from 4 to 10 weeks. Please create a Data Management Declarations declaration section underneath the Data Management section in the model tree (see Figure 14.9) and declare the following identifiers in this declaration section:

*Declaring
required
identifiers*

```

ELEMENT PARAMETER:
  identifier  : CurrentPeriod
  range      : Periods

PARAMETER:
  identifier  : TotalCostInCurrentPeriod
  unit       : $
  definition :
    sum[ s, ScenarioProbability(s) * (
      sum[ (f,p), FixedCostDueToLevelChange *
        ProductionLineLevelChange(f,p,CurrentPeriod) ] +
      sum[ f,   UnitProductionCost(f) * Production(f,CurrentPeriod) ] +
      sum[ l,   UnitStockCost(l) * Stock(l,CurrentPeriod,s) ] +
      sum[ (f,c), UnitTransportCost(f,c) * Transport(f,c,CurrentPeriod,s) ] ) ]

PARAMETER:
  identifier  : AccumulatedTotalCostInCurrentPeriod
  unit       : $

```

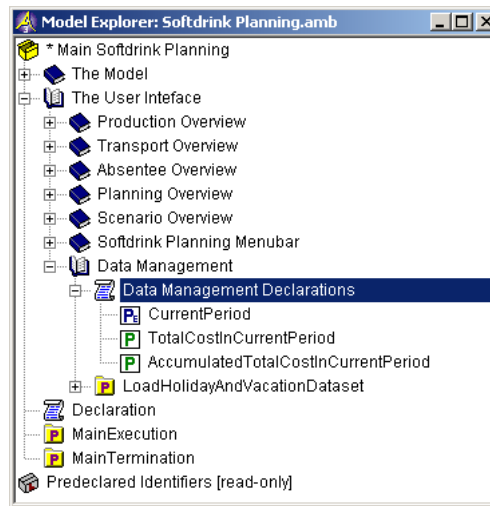




Figure 14.9: The Data Management Declarations section

To create a case that contains only a single identifier, namely *AccumulatedTotalCost*, you have to perform the following actions:

Creating a new case type ...

- ▶ press the **Data Management Setup** tool button  on the toolbar,
- ▶ select the 'Case Types' node,
- ▶ select the 'All Identifiers' case type (see Figure 14.10),
- ▶ press the **New Case Type** button  on the toolbar,
- ▶ specify 'Accumulated Total Cost' as the name for the case type, and
- ▶ press the *Enter* key to register this name.

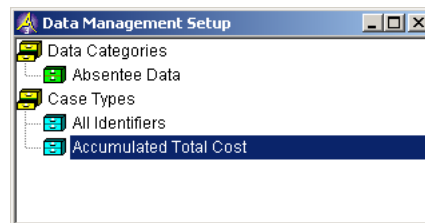



Figure 14.10: The Data Management Setup tree

To specify that only a single identifier is to be stored in 'Accumulated Total Cost' case types, please execute the following steps:

... and specifying its contents

- ▶ select the 'Accumulated Total Cost' case type if necessary,
- ▶ press the **Properties** button  to open the **Case Type Properties** dialog box,

- ▶ select the **Contents** tab,
- ▶ press the **Add** button,
- ▶ select the parameter `AccumulatedTotalCostInCurrentPeriod` (see also Figure 14.11), and
- ▶ press the **OK** button twice.

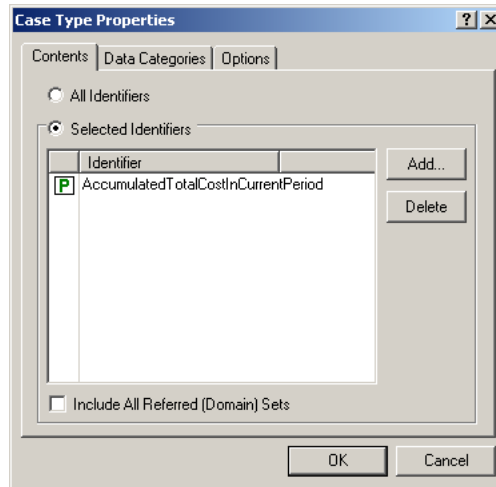


Figure 14.11: The **Contents** tab in the **Case Type Properties** dialog box

Next you need to create a procedure called `SaveCase(CaseName)` as shown in Figure 14.12. Use the **Argument** wizard to declare `CaseName` as a string parameter with property 'Input'. In addition, declare a local element parameter `CaseReference` with **Range** attribute `AllCases`. The **Body** attribute of the new procedure should be entered as follows:

*Building a
SaveCase
procedure ...*

```

if ( not CaseFind( CaseName, CaseReference ) ) then
  if ( not CaseCreate( CaseName, CaseReference ) ) then
    DialogMessage( "Error creating case." );
    return;
  endif;
endif;

CaseSetCurrent( CaseReference );
CaseSave( 0 );

```

As noted previously, you can find explanations of predefined AIMMS functions in *The Function Reference*.

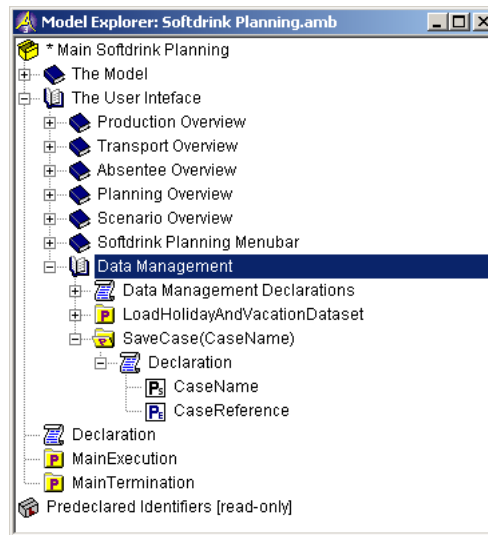


Figure 14.12: The SaveCase procedure in the model tree

Finally, you are now ready to specify the procedure `RunExperiment` in the Data Management section as shown in Figure 14.13. The contents of this procedure are extensive, but should be mostly self-explanatory. Note the use of the previously specified `SaveCase` procedure inside the following **Body** attribute:

... and specifying the experiment

```

CurrentDefaultCaseType      := 'Accumulated Total Cost';
NumberOfPeriodsInPlanningInterval := 4;

repeat "outer-loop"

  MovePlanningIntervalToStartOfCalendar;
  AccumulatedTotalCostInCurrentPeriod := 0;
  CurrentPeriod := FirstPeriodInPlanningInterval;

  while ( LastWeekInPlanningInterval < LastWeekInCalendar ) do "inner-loop"

    RollHorizonOnce;
    AccumulatedTotalCostInCurrentPeriod += TotalCostInCurrentPeriod;
    PageRefreshAll;
    break "inner-loop" when ( LeastCostPlan.ProgramStatus <> 'Optimal' );

  endwhile;

  if ( LeastCostPlan.ProgramStatus <> 'Optimal' ) then
    AccumulatedTotalCostInCurrentPeriod := 0;
  else
    for ( t | t > FirstPeriodInPlanningInterval ) do
      CurrentPeriod := t;
      AccumulatedTotalCostInCurrentPeriod += TotalCostInCurrentPeriod;
    endfor;
  endif;

  SaveCase( FormatString( "Length %n", NumberOfPeriodsInPlanningInterval ) );

```

```

break "outer-loop" when ( NumberOfPeriodsInPlanningInterval = 10 );

NumberOfPeriodsInPlanningInterval += 1;

endrepeat;

CurrentDefaultCaseType := '';

```

The completed Data Management section of the model tree should be as shown in Figure 14.13.

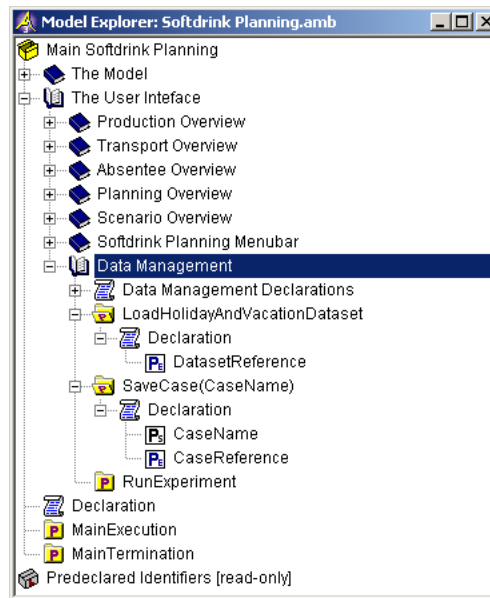


Figure 14.13: The final Data Management section

Execution of the above experiment may take a while, depending on the speed of your computer. However, before executing the experiment, you should first comment out the `halt` with part of the `solve` statement in the procedure `SolveLeastCostPlan`. This line is useful to give an appropriate error message when solving the model for one particular period, but we don't want the experiment to stop running upon finding a non-optimal solution for a certain period. The `break "inner-loop"` statement takes care of such situations in the `RunExperiment` procedure. To comment out this block, please do the following:

Preparing to run the experiment

- ▶ locate the `SolveLeastCostPlan` procedure, by using the **Find** function of AIMMS,
- ▶ select the 3 lines of the `halt` clause of the `solve` statement, as illustrated in Figure 14.14,
- ▶ open the right-mouse pop-up menu and select **Comment Block**, and
- ▶ add a semicolon after the `SolveLeastCostPlan` statement.

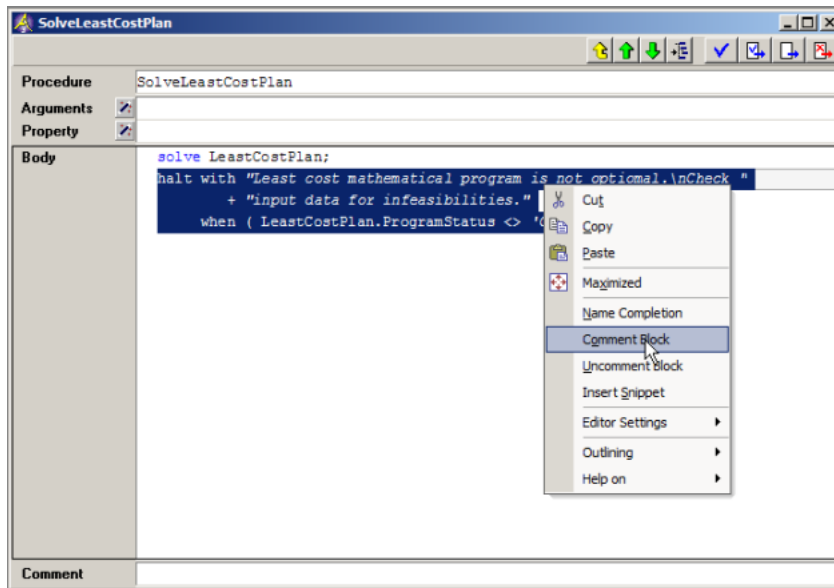


Figure 14.14: Commenting out the halt clause

To initiate the actual experiment, you should perform the following actions:

- ▶ select the RunExperiment procedure node in the model tree, and
- ▶ select the **Run Procedure** command from the right-mouse pop-up menu.

Running the experiment

The run could produce a number of warnings about the model being infeasible or unbounded. This is caused by some subproblems in the inner loop of the experiment having become insolvable. You can ignore these warnings for this tutorial.

After the experiment is complete, several cases should have been created in the case tree as shown in Figure 14.15. If you did not run the experiment, you can import the cases from the 'Cases and Datasets' directory in your project directory.

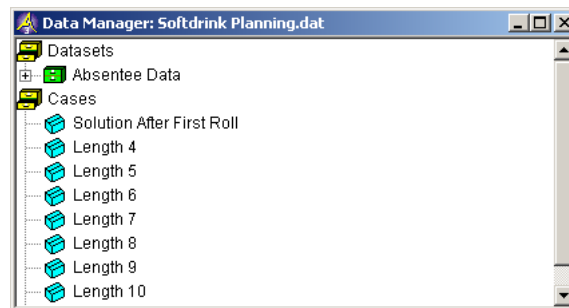



Figure 14.15: The case tree after the experiment

You are now in a position to create a table that displays the value of the parameter `AccumulatedTotalCost` for every case that has been generated during the experiment by executing the following steps:

Creating a table
...

- ▶ create a new page at the bottom of the page tree,
- ▶ enter 'Multiple Case Overview' as its name,
- ▶ press the *Enter* key to register the name,
- ▶ open the new page in edit mode,
- ▶ press the **New Table** button  on the toolbar,
- ▶ draw a rectangle on the page, and
- ▶ select the parameter `AccumulatedTotalCostInCurrentPeriod`.

To transform this table into a *multiple case object*, you should do the following:

... into a
multiple case
table

- ▶ open the **Table Properties** dialog box of the table object,
- ▶ select the **Table** tab if necessary,
- ▶ check the 'Multiple Case Object' checkbox (see Figure 14.16), and
- ▶ press the **OK** button.

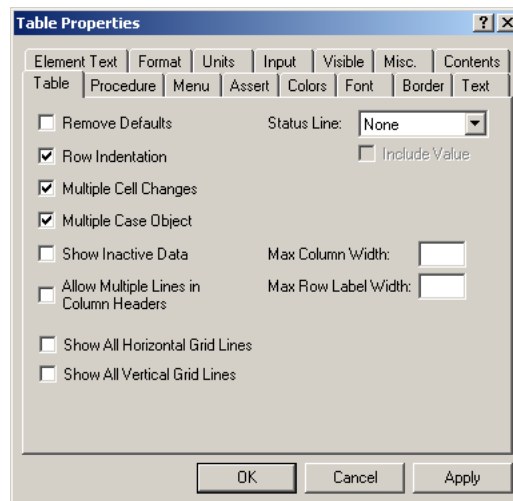




Figure 14.16: Creating a multiple case table

The table should have been extended with an empty column. To specify the multiple case selection, you should perform the following steps:

Specifying the
case selection

- ▶ press the **Page User Mode** button  on the toolbar,
- ▶ select the **Multiple Cases...** command from the **Data** menu,
- ▶ select the cases 'Length 4' through 'Length 10' from the left listbox in the **Select Multiple Cases** dialog box,

- ▶ press the **Move Right** button  to transfer the selected cases to the right listbox (see Figure 14.17), and
- ▶ press the **OK** button.

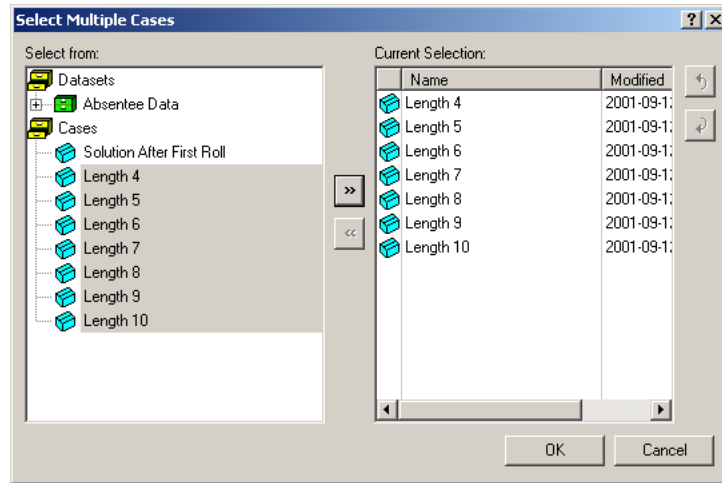


Figure 14.17: The **Select Multiple Cases** dialog box

Having specified the multiple case selection, AIMMS will automatically load the required data from the cases and complete the table as in Figure 14.18.

Viewing the result

	Length 4	Length 5	Length 6	Length 7	Length 8	Length 9	Length 10
Accumulated Total Cost In Current Period	2071568		2074931		1990153	2033005	1978466

Figure 14.18: A table displaying data for multiple cases

It is interesting to note that some entries in the table are left blank reflecting the fact that one of the subproblems in the "inner loop" of the experiment became insolvable. It is also interesting to note that the overall total cost does not decrease monotonically as the number of periods in the planning horizon increases. The experiment would seem to indicate that the number of periods should be greater than 10.