Product Information

This document applies to IBM Cognos TM1 Version 10.2.0 and may also apply to subsequent releases.

Licensed Materials - Property of IBM

© Copyright IBM Corporation 2007, 2013.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
## Contents

**Introduction** .......................................................... vii

**Chapter 1. Introduction to TM1 Development** .......... 1
- Understanding Multidimensionality .................................. 1
- Durables Cube .................................................................. 1
- Your Role as Developer .................................................. 3
- TM1 Object Naming Conventions ..................................... 3

**Chapter 2. Creating Cubes and Dimensions** .......... 7
- Designing Cubes ........................................................... 7
  - Selecting the Number of Dimensions ............................... 7
  - Consolidating Detail Using Dimension Hierarchies ................. 9
  - Types of Elements ..................................................... 12
  - Element Attributes .................................................. 13
  - Designing Cubes - Summary ....................................... 16
- Creating Dimensions ..................................................... 16
  - Creating Dimensions Using the Dimension Editor Window ........ 17
  - Modifying a Dimension ................................................. 18
  - Managing the Display of Elements in the Dimension Editor .......... 22
  - Creating Dimensions Using Dimension Worksheets ............... 24
  - Using Named Hierarchy Levels with TM1 Dimensions ............... 29
- Creating Cubes ........................................................... 31
  - Ordering Dimensions in a Cube .................................... 31
  - Creating a Cube ....................................................... 32
  - Optimizing the Order of Dimensions in a Cube ................... 33
  - Editing Cube Properties ............................................. 34

- Creating Pick Lists ..................................................... 34
  - Pick List Usage Notes ................................................ 34
  - Pick List Types ....................................................... 35
  - Creating Pick Lists with Element Attributes ........................ 36
  - Creating Pick Lists with Control Cubes ............................ 36
  - Null Values in Pick Lists .......................................... 38
  - Pick List Order of Precedence ..................................... 39

- Replicating Cubes between Servers .................................. 39
  - Cube Relationships .................................................. 39
  - Server Relationships ................................................. 39

**Chapter 3. Advanced Calculations for Business Data** .......... 41
- Overview of Cube Rules ............................................... 41
  - Guidelines for Writing TM1 Rules Statements .................... 42
    - General Considerations .......................................... 42
    - Syntax for Describing the Area .................................... 42
    - Syntax for Formulas ............................................... 43
    - Using Cube References ............................................ 45
    - Arranging Rules Statements ..................................... 46
    - Specifying Different Rules at the N: and C: Levels ............... 47
    - Bypassing Rules ................................................... 48
    - Qualifying Element Names ....................................... 48
  - Rules Editor and Rules Worksheets ................................ 48
    - Creating Rules Worksheets ....................................... 48
    - Saving Rules Worksheets .......................................... 49
  - Rules and Dimension Consolidations ................................ 49
    - Order of Calculation ............................................... 49
    - Overriding C: Level Elements with Rules ........................ 50
    - Stacking Rules ...................................................... 51

© Copyright IBM Corp. 2007, 2013
Chapter 4. Organizing Objects in TM1 Applications ................................................................. 71

Chapter 5. Importing Data with Processing Worksheets ......................................................... 93
Data Reservation modes ........................................... 136
Understanding Data Reservation behavior with other TM1 features ................ 138
  Data Reservations and Security .................................. 138
  Data Reservations and the TM1 user interface .................... 138
  Data Reservations and Data spreading ............................ 138
  Data Reservations and Turbolntegrator processes and chores .......... 139
  Data Reservations and Sandboxes ................................. 140
  Data Reservations, Sandboxes, and Job Queue .................... 140
Enabling Data Reservation .......................................... 140
  Enabling Data Reservation for cubes .............................. 141
  DataReservationMode property ................................. 141
  Enabling user groups to manage Data Reservations ................ 142
Monitoring Data Reservations ....................................... 143
  Displaying Data Reservation cell status in Server Explorer ........ 143
  Using TM1 Top to monitor threads waiting for Data Reservations .. 144
  Using the Audit Log to monitor Data Reservation events .......... 144
  Using Turbolntegrator and API functions to monitor Data Reservations . 145
Managing Data Reservations with Turbolntegrator functions ...................... 145
  CubeDataReservationAcquire .................................. 145
  CubeDataReservationRelease .................................. 146
  CubeDataReservationReleaseAll ................................ 147
  CubeDataReservationGet ....................................... 148
  CubeDataReservationGetConflicts ................................ 149
Managing Data Reservations with TM1 API functions ............................... 150
  TM1DataReservationAcquire .................................. 151
  TM1DataReservationRelease .................................. 151
  TM1DataReservationReleaseAll ................................ 152
  TM1DataReservationGetAll .................................... 153
  TM1DataReservationGetConflicts ................................ 154
  TM1DataReservationValidate ................................... 154
API error codes for data reservations ................................ 155

Appendix A. Supported Microsoft Excel Functions - TM1 Web .................. 157
Supported Functions .............................................. 157
  Date and Time Functions ....................................... 157
  Financial Functions .......................................... 158
  Information Functions ....................................... 158
  Logical Functions .......................................... 159
  Lookup and Reference Functions ................................ 159
  Math and Trigonometric Functions ................................ 160
  Text and Data Functions ..................................... 162
  Statistical Functions ....................................... 163

Appendix B. Unsupported Microsoft Excel Functions - TM1 Web ............... 167
Database and List Management Functions .................................. 167
  Date and Time Functions ....................................... 167
  Financial Functions .......................................... 168
  Information Functions ....................................... 169
  Lookup and Reference Functions ................................ 170
  Math and Trigonometric Functions ................................ 170
  Statistical Functions ....................................... 171
  Text and Data Functions ..................................... 172

Notices ......................................................... 173

Index ......................................................... 177
Introduction

This document is intended for use with IBM® Cognos® TM1®.

This manual describes how to create and maintain objects on the IBM Cognos TM1 server, administer security, and develop TM1 applications. The guide also describes aspects of TM1 security.

Finding information

To find IBM Cognos product documentation on the web, including all translated documentation, access one of the IBM Cognos Information Centers (http://pic.dhe.ibm.com/infocenter/cogic/v1r0m0/index.jsp). Release Notes are published directly to Information Centers, and include links to the latest technote and APARs.

Samples disclaimer

The Samples Outdoors Company, GO Sales, any variation of the Great Outdoors name, and Planning Sample depict fictitious business operations with sample data used to develop sample applications for IBM and IBM customers. These fictitious records include sample data for sales transactions, product distribution, finance, and human resources. Any resemblance to actual names, addresses, contact numbers, or transaction values is coincidental. Other sample files may contain fictional data manually or machine generated, factual data compiled from academic or public sources, or data used with permission of the copyright holder, for use as sample data to develop sample applications. Product names referenced may be the trademarks of their respective owners. Unauthorized duplication is prohibited.

Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products.

This product does not currently support accessibility features that help users with a physical disability, such as restricted mobility or limited vision, to use this product.

Forward-looking statements

This documentation describes the current functionality of the product. References to items that are not currently available may be included. No implication of any future availability should be inferred. Any such references are not a commitment, promise, or legal obligation to deliver any material, code, or functionality. The development, release, and timing of features or functionality remain at the sole discretion of IBM.
Chapter 1. Introduction to TM1 Development

This section provides an introduction to the concept of multidimensionality and describes some common responsibilities of developers using IBM Cognos TM1.

Understanding Multidimensionality

With IBM Cognos TM1, you can create multidimensional databases that provide business and finance managers instant meaning from complex, dynamic business models.

To understand multidimensionality, consider the example of the Vice President of Sales for a retail company who wants to analyze product sales across a retail chain that operates in the United States and Canada. Each retail store records the unit sales, dollar sales, and discounts for the durable consumer products.

The sales are analyzed by product, scenario (actual versus budget), region, measures (units, dollar sales, and discounts), and week. What we have identified is a five-dimensional model. The dimensions identify how the data is organized or how the types of data are tracked.

In TM1, the sales analysis can reside in one or more multidimensional structures called cubes. A collection of cubes forms a database. Each data point in a cube is identified by one element in each dimension of the cube; for example, actual dollar sales of dryers during the second week of January in the Boston store. TM1 cubes must contain no less than two and no more than 256 dimensions.

Durables Cube

In the following diagram, each dimension in the Durables cube is represented by a vertical line segment. The elements within the dimension are represented by unit intervals.
Suppose you are the Vice President of Sales, and you need to quickly compare the performance of products and stores to identify the winning strategies and trouble spots. Using TM1 multidimensional views, you can create an unlimited number of ad hoc queries.

In the following example, you can quickly compare actual versus budgeted dollar sales across weeks. The region is Boston and the product is a dryer model.

By rearranging the view, you can compare dollar sales for the dryer model across all regions.
Later sections of this book describe how you can use TM1 to reconfigure views and drill down into your multi-dimensional data to satisfy your analysis requirements.

Your Role as Developer

As an IBM Cognos TM1 developer, your responsibilities fall into four major tasks:

- Design and create the cubes that hold business analyses.
- Decide where to store the cubes so they can be shared across the organization.
- Import data into the cubes from transactional systems and other data sources.
- Create formulas that perform calculations, such as average prices, currency conversions, and price/earning ratios.

You must have access rights to the TM1 data in order to perform these tasks. Typically, your TM1 administrator is responsible for setting access rights. The following section describes the distinction between a local and a remote server, and lists the access rights you need to perform tasks on a remote server.

TM1 Object Naming Conventions

As a developer, you are responsible for creating and naming many objects in the IBM Cognos TM1 system. TM1 enforces some restrictions on naming while other guidelines offer “best practice” guidance. Observe the following conventions when naming TM1 objects.

Although some of these characters are not specifically reserved, it is a best practice to avoid the use of these special characters in the majority of cases when naming objects and elements. See “Element names and MDX expressions” on page 4 for specific information about naming elements.

Table 1. Special characters to avoid using in object and element names

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>’</td>
<td>apostrophe</td>
</tr>
<tr>
<td>*</td>
<td>asterisk</td>
</tr>
<tr>
<td>@</td>
<td>at sign - see “Object names in TM1 rules” on page 5</td>
</tr>
</tbody>
</table>
Table 1. Special characters to avoid using in object and element names (continued)

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\</td>
<td>back-slash</td>
</tr>
<tr>
<td>:</td>
<td>colon</td>
</tr>
<tr>
<td>,</td>
<td>comma</td>
</tr>
<tr>
<td>{</td>
<td>curly brace - see “The curly brace in object names.”</td>
</tr>
<tr>
<td>&quot;</td>
<td>double-quote</td>
</tr>
<tr>
<td>!</td>
<td>exclamation mark - see “Object names in TM1 rules” on page 5.</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater-than</td>
</tr>
<tr>
<td>&lt;</td>
<td>less-than</td>
</tr>
<tr>
<td>-</td>
<td>minus sign - in element names. See “Element names and MDX expressions.”</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>+</td>
<td>plus sign - in element names. See “Element names and MDX expressions.”</td>
</tr>
<tr>
<td>?</td>
<td>question-mark</td>
</tr>
<tr>
<td>;</td>
<td>semicolon</td>
</tr>
<tr>
<td>/</td>
<td>slash</td>
</tr>
<tr>
<td>~</td>
<td>tilde - see “Object names in TM1 rules” on page 5.</td>
</tr>
</tbody>
</table>

Specifically-reserved characters per component

The following characters are explicitly reserved for the following components and should never be used when naming objects in these contexts:

- TM1 Architect reserves these characters:
  \ / : * ? " < > | }
- TM1 Server reserves these characters in these objects: Cube, Dimension, Subset, View, Process, Chores.
  \ / : * ? " < > | ' ; ,
- For process variable name, the identifier cannot contain any special characters except for:
  AllowableChars[] = ".$\_";

The curly brace in object names

It is a best practice to avoid the use of the right curly brace (}) as the first character in any user-created TM1 object name. TM1 control object names always begin with the right curly brace. If a user-created object name begins with a right curly brace, the object becomes hidden if the Display Control Objects parameter is turned off.

Element names and MDX expressions

Do not use + or - as the first character of an element name. Although only the first element in a subset when slicing to active form cannot use + or - as the first character in the element name, in order to prevent encountering this problem it is best to never use + or - as the first character of an element name.
Although all the other characters available for use in element names are technically not restricted, it is best practice to avoid the special characters listed in the previous table when naming elements.

An element name can contain a right square bracket ( ] ) but when an element name containing this character is referenced in an MDX expression, the character needs to be escaped by doubling it. For example, an element named Array[N] Elements, can be referred to in an MDX expression as [Array[N]] Elements.

**Object names in TM1 rules**

Although technically allowed, it is a best practice to avoid using these special characters in object names because they may conflict when used in a rules expression. This guideline protects you if the objects or elements ever become part of a rule statement where those special characters are not permitted.

- For example, | \ ~ @ \ / : * ? " < > are all often found in rules statements and should not be used in object names.
- The @ is technically not restricted, however it is a best practice to avoid using the @ character in object names or element names because the @ character is also a string comparison operator in TM1 rules. If you reference any object with a name that contains the @ character in rules, the object name must be enclosed in single quotation marks. For example, a dimension named products@location must be referenced as 'products@location' in rules. Escaping the name with quotation marks does not work in all cases, so it is best to avoid the use of @ in all cases when naming objects.
- Escaping the special character using quotation marks does not work for ! or in certain rule expressions.
- The exclamation point ! character should not be used in object names because it is also used in rules expressions. For example:
  ```cub
  DB('MarketExchange',!market,!date)
  ```

**Maximum string length for data directory and object names**

The entire string represented by the combination of the IBM Cognos TM1 server data directory name and the object name is limited to 256 bytes. For example, if your data directory is C:\Financial data\TM1\ (22 bytes), object names are limited to 234 bytes, inclusive of a file extension such as .cub or .rux.

Some TM1 objects, such as views, subsets, and applications, are stored in sub-directories of the TM1 server data directory. In this case, the 256 byte limit is applied to the combination of the TM1 server data directory, the sub-directory, and the object name.

**Case sensitivity**

Object names are not case sensitive. For example, the dimension name actvsbud is equivalent to ActVsBud.

**Spaces in object names**

Spaces are allowed in all object names, but spaces are ignored by the IBM Cognos TM1 server. The TM1 server considers the dimension name Act Vs Bud to be equivalent to ActVsBud (or actvsbud).
**User names**

User names that include reserved characters cannot save private objects.
Chapter 2. Creating Cubes and Dimensions

IBM Cognos TM1 stores business data in cubes. This section describes how to create cubes and their building blocks, dimensions.

Note: All tasks described in this section require TM1 Perspectives or TM1 Architect. You cannot create cubes, create/edit dimensions, or establish replications with TM1 Client.

Designing Cubes

TM1 stores your business analyses in cubes. Each cell in a cube contains a measure that you are tracking in an analysis. A cube can store data against one or more measures.

You form a cube with dimensions, which identify how to organize the data or the measures you want to track. One element in each dimension identifies the location of a cell in a cube.

The following example cube contains three dimensions: Product, Measures, and Month. Each measure, such as Sales, is organized or dimensioned by a product and a month. For example, the cell value 30000 represents the sales of Sedan-1 in the month of January (Jan).

TM1 treats all dimensions the same way, whether they contain elements that identify measures or describe how you organize the measures.

Selecting the Number of Dimensions

Every cube has at least two dimensions and a maximum of 256 dimensions. For example, a two-dimensional cube is best suited as a lookup table when you want to calculate values in other cubes that have more dimensions. For example, you can convert the local currency amounts to a reporting currency by using a two-dimensional cube that stores the exchange rates. You retrieve the rate using the TM1 rule.
The number of dimensions in a cube depends mostly on the *dimensionality* of your data. Consider the accounts in a Profit and Loss Statement.

### Profit and Loss Statement (in thousands)

<table>
<thead>
<tr>
<th>Year Ending 31 Dec, 2002</th>
</tr>
</thead>
<tbody>
<tr>
<td>Net sales</td>
</tr>
<tr>
<td>Direct costs</td>
</tr>
<tr>
<td>Direct labor</td>
</tr>
<tr>
<td>Gross Profit</td>
</tr>
<tr>
<td>Salaries</td>
</tr>
<tr>
<td>Payroll</td>
</tr>
<tr>
<td>Electricity</td>
</tr>
<tr>
<td>Rent</td>
</tr>
<tr>
<td>Depreciation</td>
</tr>
</tbody>
</table>

If you want to examine how the revenue and expenses vary by their factors, you would need to divide the accounts into two groups.

- **Accounts above the Gross Profit line**, such as Net sales and Direct costs, which you can dimension by product, region, scenario (Actual versus Budget), and months.
- **Accounts below the Gross Profit line**, such as Payroll, Electricity, and Rent, which you can dimension by region, scenario (Actual versus Budget), and months, but not product. You cannot directly attribute the overheads to products, so you cannot analyze them at the same level of detail.

This difference in dimensionality suggests two cubes:

- Five-dimensional cube for the accounts above the Gross Profit line
- Four-dimensional cube for the accounts below the Gross Profit line

**Five-dimensional Cube**
The following diagram represents the dimensions and elements in the five-dimensional cube. Each dimension is represented by a vertical line segment. The elements within the dimension are represented by unit intervals.
**Four-dimensional Cube**

The following diagram represents the dimensions and elements in the four-dimensional cube. Each dimension is represented by a vertical line segment. The elements within the dimension are represented by unit intervals.

---

**Consolidating Detail Using Dimension Hierarchies**

The data you import into a cube provides a snapshot of your business at a specific level of detail. For example, you might import the weekly or monthly sales data for products by city. The dimension elements that identify these data points are simple or leaf-level elements in each dimension: sales for one week, one product, one city.

By using dimension hierarchies, you can easily aggregate numeric data into categories that are meaningful in your analyses. Each category corresponds to an
aggregation of detail for two or more elements in a dimension. For example, you can create quarterly elements that sum monthly sales amounts. In TM1, elements that represent aggregations are called consolidated elements or consolidations.

The following diagram shows three levels of consolidation for elements of a Region dimension. The cities provide the lowest level of detail (Level 0). The cities roll up into state consolidations (Level 1), which roll up into regional consolidations, which finally roll up into the Eastern USA consolidation (Level 3).

**Navigating through a Dimension Hierarchy**
A dimension hierarchy provides a navigation path for analyzing data at different levels of detail. Drilling down means navigating to greater levels of detail along one dimension. For example, as you drill down from New England in the above Region dimension, you access the underlying data for two states and then four cities. Drilling up means navigating to summary levels in dimensions.

**Using Weights to Express Consolidations**
Weight factors determine the contribution of an element to a consolidation. To express that the Connecticut sales total is a sum of Hartford and New Haven, you assign a default weight factor of 1.0 to both Hartford and New Haven.

You can also consolidate elements by subtracting the values associated with the elements. For example, you can express Net Profit as Market Value - Acquisition Value. You would assign a weight factor of 1.0 to Market Value and -1.0 to Acquisition Value. The following table presents four weighting examples of consolidation.
Creating Multiple Roll-ups in a Dimension

You can roll up low-level numeric data, such as sales and units, in several ways by creating multiple hierarchies in a dimension. By creating multiple roll-ups in a dimension, you can reduce the number of dimensions, and the number of empty cells in a cube.

In the following example, Hartford, a simple element in the Region dimension, rolls up along two paths: geographic and management.

Geographic Roll-up:

Dimension: Region

<table>
<thead>
<tr>
<th>Level 3</th>
<th>Level 2</th>
<th>Level 1</th>
<th>Level 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eastern USA</td>
<td>New England</td>
<td>Connecticut</td>
<td>Hartford</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>New Haven</td>
</tr>
</tbody>
</table>
Management Roll-up:

**Dimension: Region**

<table>
<thead>
<tr>
<th>Level 3</th>
<th>Level 2</th>
<th>Level 1</th>
<th>Level 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hartford</td>
<td>New Haven</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sales Rep</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sales Manager</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sales VP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Creating Complex Calculations**

TM1 aggregates the consolidations you create within dimensions by the weight factors you assign. To create ratios between the elements or multiply the values associated with the elements, you must create a complex calculation, called a rule. Here are two calculations that require TM1 rules:

- Gross Margin = (Gross Profit / Net Sales) x 100
- Sales = (Price x Units)

You can also use rules to tally the elements. However, rules take longer to process than consolidations, especially in large or sparse cubes. Sparse cubes have a high percentage of empty cells.

IBM® Cognos® TM1® also supports rule-aware aggregate processing. This feature is visible primarily on IBM® Cognos® Business Intelligence® (BI) clients. In previous versions of Cognos TM1 the default aggregation could not be computed by the Cognos BI server when applied to Cognos TM1 rule-calculated cells. In previous releases, Cognos BI reports returned error cells as a result of default aggregation applied to Cognos TM1 rule-calculated cells. With rule-aware aggregation in place, the aggregation numbers are reported based on the semantics of the Cognos TM1 rules.

The following limitations are still in place when computing default aggregation against rule-calculated cells:

- Multi-dimensional tuple sets will not be possible to aggregate against calculated cells unless they are a result of crossjoin of single dimension member sets. The aggregation result in this case will continue to be an error cell.
- Rule-aware aggregation cannot be applied to MDX calculated members, which means calculated members are not supported in the aggregated member set, nor in aggregation context. In this case the aggregation over calculated cells still results in an error cell.
- Aggregated cells have to be associated with the same rule so that the system can re-use this rule for the aggregation result. The scope of the rule needs to be general enough to include UDC (Use Defined Consolidation) elements belonging to the dimensions of the aggregated member set.

For a comprehensive explanation of rules, see the IBM Cognos TM1 Rules Guide.

**Types of Elements**

So far, you've learned about simple or base-level elements that you can roll up to consolidations. TM1 supports three types of elements:
Element Description

**Numeric**
Identify the lowest-level detail in a dimension. In a cube that contains only numbers, TM1 defines all the lowest-level elements as numeric.

**Consolidated**
Aggregations of lower-level detail. For example, you could use the 1Quarter element in a time dimension to sum the sales amounts for the first three months of the year.

**String**
Stores text strings in cells. To include a string in a cell in a cube, the element from the last dimension defining the cell must be a string element. TM1 treats string elements that occur in any dimension other than the last one as numeric elements.

The typical use for string elements is in a two-dimensional cube that converts codes in an input file to formal element names. For example, if you want to convert account codes to account names.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numeric</td>
<td>Identify the lowest-level detail in a dimension. In a cube that contains only numbers, TM1 defines all the lowest-level elements as numeric.</td>
</tr>
<tr>
<td>Consolidated</td>
<td>Aggregations of lower-level detail. For example, you could use the 1Quarter element in a time dimension to sum the sales amounts for the first three months of the year.</td>
</tr>
<tr>
<td>String</td>
<td>Stores text strings in cells. To include a string in a cell in a cube, the element from the last dimension defining the cell must be a string element. TM1 treats string elements that occur in any dimension other than the last one as numeric elements. The typical use for string elements is in a two-dimensional cube that converts codes in an input file to formal element names. For example, if you want to convert account codes to account names.</td>
</tr>
</tbody>
</table>

**Element Attributes**
The elements identify data in a cube, and the element attributes describe the elements themselves.

You can use attributes to:
- List features of elements. For example, the square footage of a store and the engine size of a car model.
- Provide alternative names, or aliases. For example, descriptive names of general ledger accounts and the local language versions of product names.
- Control the display format for the numeric data. Typically, you select a display format for the measures you track in a cube.

You can select elements by attribute value in the Subset Editor. You can also display element names in TM1 dialog boxes using their aliases.

To create attributes and assign attribute values, use the Attributes Editor.

**Descriptive Attributes**
The following table shows sample attributes that describe car models.

<table>
<thead>
<tr>
<th>Element</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sedan 1</td>
<td>Horsepower (Numeric) 190</td>
</tr>
<tr>
<td>Sedan 2</td>
<td>Horsepower (Numeric) 140</td>
</tr>
<tr>
<td>Sedan 3</td>
<td>Horsepower (Numeric) 120</td>
</tr>
<tr>
<td>Sedan 4</td>
<td>Horsepower (Numeric) 180</td>
</tr>
<tr>
<td>Sedan 5</td>
<td>Horsepower (Numeric) 140</td>
</tr>
</tbody>
</table>

**Alias Attributes**
The following table shows German, Spanish, and French versions of English furniture names.
**Display Format Attributes**

The **Cube Viewer** window displays numeric data in the formats shown in the following table.

<table>
<thead>
<tr>
<th>Format Name</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Currency</td>
<td>Numbers appear with a currency symbol and the specified number of decimal places (Precision). TM1 uses the currency symbol defined in the Microsoft Windows Regional Settings dialog box.</td>
<td>$90.00</td>
</tr>
<tr>
<td>General</td>
<td>Numbers appear with a specified number of decimal places (Precision).</td>
<td>-90</td>
</tr>
<tr>
<td>Percentage</td>
<td>Numbers appear as percentages, with a specified number of decimal places (Precision).</td>
<td>90.00%</td>
</tr>
<tr>
<td>Scientific</td>
<td>Numbers appear in exponential form, with a specified number of decimal places (Precision).</td>
<td>9.0e+001</td>
</tr>
<tr>
<td>Date</td>
<td>Numbers appear as a date string, 1=January 1, 1900. There are a number of date formats available.</td>
<td>March 31, 2002</td>
</tr>
<tr>
<td>Time</td>
<td>Numbers appear as a time string. There are a number of time formats available.</td>
<td>19:53:30 A</td>
</tr>
<tr>
<td>Comma</td>
<td>Places commas in the appropriate places in large numbers.</td>
<td>1,000,000</td>
</tr>
<tr>
<td>Custom</td>
<td>A user-defined format.</td>
<td>Custom</td>
</tr>
</tbody>
</table>

Using the **Attributes Editor** window, you can select a display format for every element in each dimension in a cube. However, it is recommended that you select display formats only for one dimension, the measures you track in a cube. You can also select a format in the **Cube Viewer** window that applies to cells whose elements do not have a display format defined.

TM1 determines which display format to use in the **Cube Viewer** window, as follows:

**Procedure**

1. TM1 first checks the elements in the column dimension for display formats. If dimensions are stacked, TM1 checks from the bottom upward.
2. If no format is found, TM1 checks the elements in the row dimension for display formats. If dimensions are stacked, TM1 checks from right to left.

3. If no format is found, TM1 checks the title elements for display formats. The elements are inspected from right to left.

4. If no format is found, TM1 applies the format for the current view.
   To ensure that TM1 applies the format for the cube measures, position the dimension containing the measures as the bottommost column dimension.

**Setting Display Formats for Rows or Columns**

You might want to format the numbers in a single column or row. For example, numbers in a column or row that contain two decimal places might be better represented as whole numbers with no decimal points. To format the numbers in a single column or row, use the Attribute Editor.

The Month dimension displays in the column of the view. Any display format you assign to a column dimension overrides the display format you select for the row dimension.

For the first task, let's make sure that no formatting is applied to the Year element of the Month dimension. Follow these steps.

**Procedure**

1. Open the Format view of SalesPriorCube.
2. In the Server Explorer, expand the SalesPriorCube cube so that you can see its dimensions.
3. Right-click the Month dimension, and select Edit Element Attributes. The Attribute Editor opens.
   Note that there are no Format attribute values for the Month dimension. You can now be sure that any formats you set for the row dimension will take precedence in the Cube Viewer (or In-Spreadsheet Browser).
4. Click the Cancel to close the Attribute Editor.
   For this task, follow the steps to format the numbers in the Units row as whole numbers with zero (0) decimal places.
5. In the Server Explorer, right-click the Account1 dimension, and select Edit Element Attributes. The Attribute Editor opens.
6. Click the cell at the intersection of the Units element row and the Format column.
7. Click the Format button. The Number Format dialog box opens.
8. Select the Comma category, enter 0 in the Precision box, and click OK.
9. Click OK in the Attributes Editor dialog box.
10. Click Recalculate to recalculate the Format view. The Units values now display as whole numbers without decimal points.

**Attributes versus Elements**

When you want to list multiple attributes values for a single element, consider creating additional elements or additional dimensions. For example, the exterior color is an attribute of car models. The red models often outsell the other color models. If you create one element per car and another dimension with elements for
each color, you can use TM1 to track car sales by color. If you combine sales into a single model, you might lose valuable detail.

Consider another example. In the car models table, there is an attribute category for engine configuration. Each car has a single engine configuration, such as V-8. If any sedans are available in more than one engine configuration, consider creating one element per engine configuration.

Designing Cubes - Summary

Here are guidelines to use when you design cubes:

Procedure

1. List the measures you want to track in your business analysis. Examples of measures include sales amounts, units sold, expenses, acquisition values, and campaign costs.
2. Determine how you want to organize or dimension the measures. In most analyses, you track measures over time.
   - What is the base time interval: days, weeks, months?
   - Is there a geographic dimension?
   - Do the measures vary by customer and product?
   - Is there a scenario dimension (actual versus budget)?
3. Determine how you want to consolidate the dimension elements.
4. Create a list of attributes you want to associate with the elements of the cube. Examples of attributes include store square footage, customer IDs, and local language versions of element names.
5. Define the display formats for the measures in your cubes. For example, define Gross Margin as a percentage and Sales as a currency amount.

Creating Dimensions

When you create a dimension, you identify the leaf-level elements that comprise the dimension and, optionally, any hierarchies (consolidations) within the dimension.

There are four ways to create dimensions:

- **Dimension Editor** - Add elements as well as create and rearrange consolidations while you design dimensions. For more information, see "Creating Dimensions Using the Dimension Editor Window" on page 17.
- **TurboIntegrator** - Import element names from an ASCII, ODBC, cube view, or dimension subset source. Simultaneously create multiple dimensions and establish consolidation within those dimensions. For more information, see the IBM Cognos TM1 TurboIntegrator Guide.

**Note**: TM1 requires DataDirect drivers to access an Oracle ODBC source on Solaris or AIX®. These drivers are not supplied with TM1 and must be acquired separately.

- **Importing Data into a New Cube** - Use TurboIntegrator to map input rows from a data source to a cube. Then identify the input columns that supply the cell values and the elements that identify the cell location. For more information about this process, see the IBM Cognos TM1 TurboIntegrator Guide.
Creating Dimensions Using the Dimension Editor Window

This section walks you through the steps for creating a simple Area dimension using the Dimension Editor window. Assume that the hierarchy for the Area dimension includes the consolidated New England element and three simple elements, Connecticut, Massachusetts and Vermont.

Procedure

1. In the Tree pane of the Server Explorer, select Dimensions beneath the server that you want to contain the dimension.

2. Click Dimensions, Create New Dimension.
   The Dimension Editor opens.
   You can now add elements to the dimension.

3. Click Edit, Insert Element or click Insert Sibling.
   The Dimension Element Insert dialog box opens.

4. To add a consolidated element, do the following:
   • Type New England in the Insert Element Name field.
   • Select Consolidated from the Element Type list.
   • Click Add.
   • Click OK.
   New England now appears as the first element of the dimension, which is a consolidated element. Now let's add three child elements to the New England element. As a result, New England becomes the parent element of the three child elements.

5. Select the New England element.

6. Click Edit, Insert Child or click Insert Child.
   The Dimension Element Insert dialog box opens. TM1 displays a Parent name of New England, indicating that any elements you create will be the children of New England.

7. In the Insert Element Name field, type Connecticut and click Add.

8. In the Insert Element Name field, type Massachusetts and click Add.

9. In the Insert Element Name field, type Vermont and click Add.
   The dialog box now contains three children of New England, each with a default weight of 1.

10. Click OK.
    The Dimension Editor shows the new elements as children of New England.

11. Click Dimension, Save or click Save.
    The Dimension Save As dialog box opens.

12. Enter a dimension name and click Save.
    Dimension names can have a maximum of 256 characters. Always use descriptive dimension names.
    The new dimension displays in the list of dimensions on the server.
Modifying a Dimension

After creating a dimension, you can make the following modifications:

- Add siblings to existing elements.
- Add children to existing elements.
- Rearrange the hierarchy structure, such as repositioning elements within consolidations.
- Delete elements from the dimension.
- Delete elements from consolidations.
- Edit element properties, such as changing the weight of an element within a consolidation.
- Rearrange the order of elements in the dimension.

Adding Siblings to Existing Elements

Follow these steps to add siblings to an existing element in the Dimension Editor.

Procedure

1. Right-click the element to which you want to add siblings and select Edit Dimension Structure.
2. Click Edit, Insert Sibling.
   The Dimension Element Insert dialog box opens.
3. Enter the name of the first sibling in the Insert Element Name field.
4. If applicable, enter an Element Weight.
   The default element weight is 1.
5. Click Add.
6. Repeat steps 3 through 5 for each sibling you want to add.
7. Click OK.
   TM1 adds the new elements as siblings of the element you selected in step 1.

Adding Children to Existing Elements

Follow these steps to add children to existing elements in the Dimension Editor.

Procedure

1. Right-click the element to which you want to add siblings and select Edit Dimension Structure.
   If you add children to a simple (leaf-level) element, the element automatically becomes a consolidated element.
2. Click Edit, Insert Child.
   The Dimension Element Insert dialog box opens.
3. Enter the name of the first child in the Insert Element Name field.
4. If applicable, enter an Element Weight.
   The default element weight is 1.
5. Click Add.
6. Repeat steps 3 through 5 for each child you want to add.
7. Click OK.
   TM1 adds the new elements as children of the element you selected in step 1.

Rearranging the Dimension Hierarchy

Follow these steps to change the position of elements within the dimension hierarchy.
Procedure
1. In the **Dimension Editor**, select the elements you want to move.
   - To select a single element, click the element.
   - To select multiple adjacent element, click the first element, hold down SHIFT, and click the last element. You can also press CTRL-A to select all visible elements.
   - To select multiple non-adjacent elements, hold down CTRL, and click each element.
2. Drag and drop the elements to their new location in the dimension hierarchy.
   As you drag the elements, the cursor changes to indicate where TM1 will drop the elements. Also, the status bar displays a message indicating where TM1 will drop the elements.
   You can also cut and paste elements to rearrange the dimension hierarchy.

Deleting Elements from a Dimension
Follow these steps to delete elements from a dimension.

Procedure
1. Select the elements you want to delete.
   - To select a single element, click the element.
   - To select multiple adjacent element, click the first element, hold down SHIFT, and click the last element.
   - To select multiple non-adjacent elements, hold down CTRL, and click each element.
   - To select elements by hierarchy level, attribute value, or spelling pattern, see the IBM Cognos TM1 User Guide.
2. Click **Edit, Delete Element**.
   A confirmation dialog box displays that lists the dimension name and asks if you are sure you want to delete the object that you have selected. Click **Yes** to proceed with the deletion, click **No** or **Cancel** to cancel the deletion.

Deleting Elements from a Consolidation
Follow these steps to delete elements from a consolidation.

Procedure
1. Select the elements you want to delete.
   - To select a single element, click the element.
   - To select multiple adjacent element, click the first element, hold down SHIFT, and click the last element.
   - To select multiple non-adjacent elements, hold down CTRL, and click each element.
2. Click **Edit, Delete Element from Consolidation** or click **Delete**.
   A confirmation dialog box displays that lists the dimension name and asks if you are sure you want to delete the object that you have selected. Click **Yes** to proceed with the deletion, click **No** or **Cancel** to cancel the deletion.
   TM1 deletes the element from the consolidation, and keeps any other instances of the element in the dimension.

Note: If you define the element only within the consolidation, TM1 deletes the element from the dimension as well.
Editing Element Properties
You can edit the element properties to assign a new weight to an element of a consolidation, or to change the element type of a leaf-level element.

**Note:** You cannot change the element type of consolidated elements, and you cannot assign an element weight to any instance of an element that is not a member of a consolidation.

**Procedure**
1. Select the element.
2. Click **Edit, Element Properties.**
   - The **Dimension Element Properties** dialog box opens.
3. If necessary, select a new element type from the **Element Type** list.
4. If necessary, enter a new **Element Weight**.
5. Click **OK**.

Setting the Order of Elements in a Dimension
TM1 lets you set the order of elements in a dimension to determine the index value for each element in a dimension. The first element in a dimension has an index value of 1, the second element has an index value of 2, and so on.

Set the order of elements in a dimension is an important feature because many TM1 functions (worksheet, rules, and TurboIntegrator) reference the element index values.

**Note:** If you change the order of elements in a dimension, any functions that reference element index values return new and possibly unexpected values.

**Procedure**
1. Order the elements as you want them to appear in the dimension.
   - You can use the sort options and drag-and-drop functionality of the **Dimension Editor** to alter the order of elements.
2. Click the **Set Dimension Order** button.
3. Click **Dimension, Save.**
   - When the sorting property of the dimension is set to Automatic, TM1 prompts you to change the sorting property to Manual.
   - When the sorting property of the dimension is set to Manual, TM1 inserts any elements you added to the dimension wherever you manually positioned them in the **Dimension Editor**.
4. Click **Yes** to save the new dimension order and set the dimension sorting property to Manual.
   - You can set the order of elements even when the Dimension Editor displays only a subset of all dimension elements. For example, if you have a large dimension, you might want to alter and set the order of just a few elements. Be aware that when you set the order of elements with just a subset of elements displayed in the **Dimension Editor**, the entire dimension is affected.
   - The following example shows how setting the order of elements when working with a subset affects the entire dimension in Subset Editor.
   - For simplicity, this sample dimension contains ten elements with single-letter names, but the concept illustrated in this example applies to larger, more complex dimensions.
   - a
5. Now if you select the elements c, d, and g then click Edit, Keep, the Dimension Editor contains only the selected subset of elements.

6. Now, you decide that you want to change the order of these three elements. You want d to be the first element, and c to be the last element.

7. Now that the elements appear in the order you want, click Set Dimension.

   Order

   The order of elements for the entire dimension is now set. If you look at the entire dimension, you see that it opens in Dimension Editor as follows:

   • i
   • a
   • b
   • d
   • g
   • c
   • e
   • f
   • h
   • i

   For the example, when you set the order of elements for a subset, the new order affects the entire dimension in the following way:

   • The subset of elements that was active when the element order was set appear with shaded icons.
   • The first subset element maintains its position in the dimension relative to its nearest predecessor.
   • In the example, element d is the first element in the subset when the order of elements was set.
   • Element b is the nearest predecessor, exclusive of subset elements, to d in the dimension, so d now follows b in the dimension structure.
   • The other subset elements appear in the dimension structure maintaining their position relative to the first element in the subset.

Setting the Order of Dimension Elements from the Server Explorer:

You can also set the order of dimension elements from the Server Explorer without opening the Dimension Editor. You can also select a sorting property for the dimension from these three automatic sort orders:

• Name
• Level
• Hierarchy
Procedure
1. Right-click the dimension in the Server Explorer.
2. Click Set Elements Order.
   The Dimension Element Ordering dialog box opens.
3. Select a sort type.

<table>
<thead>
<tr>
<th>Sort Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic</td>
<td>Enables the Automatic Sort By options: Name, Level, and Hierarchy.</td>
</tr>
<tr>
<td>Manual</td>
<td>Orders elements as they currently exist in the dimension structure and sets the dimension sorting property to Manual.</td>
</tr>
</tbody>
</table>

4. If you select the Manual sort type, skip to step 7.
5. Select an Automatic Sort By option.

<table>
<thead>
<tr>
<th>Sort By</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Sorts elements alphabetically</td>
</tr>
<tr>
<td>Level</td>
<td>Sorts elements by hierarchy level.</td>
</tr>
<tr>
<td>Hierarchy</td>
<td>Sorts elements according to the dimension hierarchy.</td>
</tr>
</tbody>
</table>

6. If applicable, select a Sort Direction.
7. Click OK.

Results
You have now set the order of the dimension elements. When you open the dimension, you will see the elements in order according to the Sort By option you specified in step 5.

Managing the Display of Elements in the Dimension Editor
The Dimension Editor includes several features that let you manage the way elements display. For example, when you work with large dimensions it might be helpful to display only the elements of a certain hierarchy level, or you might want to view the elements in alphabetical order.

When you alter the display of elements in the Dimension Editor, you do not change the dimension structure, you simply change the way TM1 presents the elements in the window.

The upcoming sections describe how to manage the display of elements in the Dimension Editor.

Keeping Elements
Follow these steps to display only the selected elements in the Dimension Editor.
Procedure
1. Select the elements you want to display.
   - To select a single element, click the element.
   - To select multiple adjacent elements, click the first element, hold down SHIFT, and click the last element.
   - To select multiple non-adjacent elements, hold down CTRL, and click each element.

2. Click **Edit, Keep** or click **Keep**.
   The **Dimension Editor** displays only the elements you selected.

Hiding Elements
Follow these steps to hide selected elements in the Dimension Editor.

Procedure
1. Select the elements you want to hide.
   - To select a single element, click the element.
   - To select multiple adjacent elements, click the first element, hold down SHIFT, and click the last element.
   - To select multiple non-adjacent elements, hold down CTRL, and click each element.

2. Click **Edit, Hide** or click **Hide**.
   The Dimension Editor hides the elements you selected. All other elements remain visible.

Sorting Elements Alphabetically
You can sort the elements in the Dimension Editor in ascending or descending alphabetical order.

<table>
<thead>
<tr>
<th>Sort Order</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ascending Alphabetical</td>
<td>Click <strong>Edit, Sort By, Ascending</strong> or click <strong>Sort Ascending</strong></td>
</tr>
<tr>
<td>Descending Alphabetical</td>
<td>Click <strong>Edit, Sort By, Descending</strong> or click <strong>Sort Descending</strong></td>
</tr>
</tbody>
</table>

Sorting Elements by Index Value
You can sort the elements in the **Dimension Editor** in ascending or descending order according to the index value.

<table>
<thead>
<tr>
<th>Sort Order</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ascending Index Value</td>
<td>Click <strong>Edit, Sort By, Index Ascending</strong> or click <strong>Sort By Index, Ascending</strong></td>
</tr>
<tr>
<td>Descending Index Value</td>
<td>Click <strong>Edit, Sort By, Index Descending</strong> or click <strong>Sort By Index, Descending</strong></td>
</tr>
</tbody>
</table>
Sorting Elements by Hierarchy
You can also sort elements as they appear in the dimension hierarchy.

<table>
<thead>
<tr>
<th>Sort Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>As they appear in the dimension hierarchy</td>
<td>Click Edit, Sort By, Hierarchy or click Hierarchy Sort</td>
</tr>
</tbody>
</table>

Viewing Elements by Alias
If you defined an alias attribute for a dimension, you can view the elements by their aliases in the Dimension Editor.

The dimensions in the TM1 Sample Data directory have aliases defined for French and German, so you can view all element names by their foreign language equivalents.

Procedure
1. Open the Month dimension in the Dimension Editor.
2. Click View, Toolbars, Alias to display the Alias toolbar.
   The Alias toolbar contains two objects: a Use Aliases toggle button to turn the display of aliases on or off, and a Select Alias list from which you can select an alias.
3. Select Monat (German for 'Month') from the Select Alias list.
4. Click the Use Aliases button.

Results
By default, TM1 does not display the aliases. The Dimension Editor now displays all elements by their German aliases.

Creating Dimensions Using Dimension Worksheets
A dimension worksheet is a modified Microsoft Excel spreadsheet in which you list elements and hierarchical relationships for one dimension. TM1 saves the worksheet data in two files: dimension worksheet (.dimname.xdi) and compiled file (.dimname.dim).

- When you create a dimension using the Dimension Editor window or TurboIntegrator, TM1 writes only to the dimname.dim (compiled) file.
- When you modify a dimension using the Dimension Editor, TM1 saves your changes to the .dim file. If an .xdi file exists for the dimension, TM1 prompts you to save changes to the dimension worksheet. If you decline to update the dimension worksheet, the dimension structure in the .dim file will differ from that in the .xdi file.
- To ensure that TM1 has access to all of your dimension changes, create and maintain dimensions with dimension worksheets or the Dimension Editor. Do not mix methods. This can lead to data loss.

Keeping Your Data Synchronized
TM1 saves the dimension worksheet files to the first directory listed in the Local Server Data Directory field in the Options dialog box. Therefore, TM1 might save your dimension worksheet (.xdi) to a different directory than your dimension file (.dim).
Note: Be very careful when editing dimensions using worksheets. There are several ways in which data can be lost if you do not take proper precautions. The following examples show two ways you can lose dimension data.

- On Monday, you edit a dimension worksheet for the Account1 dimension, and save the dimension. On Tuesday, another administrator uses the Dimension Editor to make changes to the dimension. The changes she makes are not propagated to the dimension worksheet file. On Wednesday, you make another change to the dimension using the out-of-date dimension worksheet. Your changes overwrite the changes made by the other administrator.

- Two administrators decide to update a dimension on the same server using two different dimension worksheets. This is very dangerous. TM1 can overwrite the changes very easily. Use one set of dimension worksheets. Try to avoid having more than one .xdi file for any dimension.

To avoid confusion and possible data loss, use precautions when editing dimension worksheets. We recommend using one of the following procedures.

- Use the remote server data directory to store the dimension worksheets.
- Use a special directory to store your dimension worksheets.

Using the Remote Server Data Directory:

To edit the dimension worksheets in the server data directory on the remote server, follow these steps.

Procedure
1. Set the Local Server Data Directory in the Options dialog box to the data directory used by your remote IBM Cognos TM1 server.
2. Turn off the option Connect to Local Server on Startup.
3. Verify that any local servers running on your computer are shut down.
4. Log in to the remote TM1 server.
5. Make all of the modifications to your dimensions using the worksheet files in the remote server data directory.
6. Click TM1, Dimension Worksheets, Save in Excel.
   TM1 writes both the .dim file and the .xdi file to the remote server data directory.

Using a Worksheet Directory:

By using a special worksheet directory, you can sharply restrict access to your dimension worksheets. This can be very useful in security-conscious installations. To set up a worksheet directory, follow these steps.

Procedure
1. Create a worksheet directory somewhere on your file system.
2. Move all of your worksheet files (.xdi for dimensions, and .xru for rules) to the worksheet directory.
3. Set the Local Server Data Directory in the Options dialog box to the worksheet directory.
4. Connect to the remote server that contains the dimension you want to update, and any other servers you want.
5. Make all of the modifications to your dimensions using the worksheet files in the worksheet directory.
6. Click TM1, Dimension Worksheets, Save in Excel.
   The Select Server Name dialog box opens.
7. Select the server on which you want to save the compiled dimension.
8. Click OK.

**Creating Dimension Worksheets**
Follow these steps to create a dimension using dimension worksheets.

**Procedure**
1. Click TM1, Dimension Worksheets, New in Excel.
   The Create a Dimension dialog box opens.
2. In the top box, enter a name for the dimension as follows:
   - To create a dimension on your local server, type the name of the dimension. For example: Product.
   - To create a dimension on a remote server, type the server name, a colon, and then the dimension name. For example, sales:Product creates the Product dimension on the sales server.

   **Note:** You must be the TM1 administrator to create a dimension on a remote server.
3. Click OK.
   A blank dimension worksheet opens in Excel.
Filling Out Dimension Worksheets

The following worksheet defines the structure for a Month dimension.

The following sections describe how to define simple and consolidated elements in a dimension worksheet.

Defining Simple Elements:

Begin filling out the dimension worksheet by defining the dimension’s simple (leaf-level) elements, starting in row 1.

Procedure

1. In column A, specify the element type.
   • For numeric elements, type N.
   • For string elements, type S.
   In the example, all elements are numeric.
2. In column B, type the name of an element.
Defining Consolidated Elements:

After defining the simple elements in the dimension, you can define the consolidated elements.

Procedure
1. In an empty row below the last simple element, type C in column A.
2. Type the name of the consolidated element in column B.
3. In subsequent rows, type the names of the consolidated element's children in column B.
   
   For example, the following excerpt from a dimension worksheet shows the consolidated element 1 Quarter defined as the consolidation of the children Jan, Feb, and Mar.

   ![Excel Spreadsheet Screenshot]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>1 Quarter</td>
</tr>
<tr>
<td>15</td>
<td>Jan</td>
</tr>
<tr>
<td>16</td>
<td>Feb</td>
</tr>
<tr>
<td>17</td>
<td>Mar</td>
</tr>
</tbody>
</table>

Weighting Elements in a Consolidation:

Use column C to weight the elements in a consolidation. To calculate the Gross Margin, for example, you subtract Variable Costs from Sales. To express this calculation, assign the weight -1 to the Variable Costs element in the consolidation. The following worksheet shows the weighting in the Account1 dimension.
Saving Dimension Worksheets
To save the dimension worksheet and compile the dimension, click TM1, Dimension Worksheets, Save in Excel. TM1 updates the dimension worksheet file (dimname.xdi) and creates the compiled dimension file (dimname.dim).

Note: Be sure not to use File, Save in Excel. This action saves only the .xdi file; TM1 does not compile the dimension and save the .dim file.

Using Named Hierarchy Levels with TM1 Dimensions
You can assign your own custom names to the hierarchy levels of the TM1 dimension by using the }HierarchyProperties control cube. The named levels you create can then be used when you externally access TM1 data with IBM Cognos Report Studio, MDX statements or other MDX OLAP tools. You can also assign a default member for the dimension.

For example, instead of using the generic hierarchy level names of level000, level001, level002, you could assign names that describe the levels in a Customers dimension as shown in the following table.

<table>
<thead>
<tr>
<th>TM1 Dimension Levels</th>
<th>Example Named Dimension Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>level000</td>
<td>All</td>
</tr>
<tr>
<td>level001</td>
<td>State</td>
</tr>
<tr>
<td>level002</td>
<td>City</td>
</tr>
<tr>
<td>level003</td>
<td>Individual</td>
</tr>
</tbody>
</table>

Configuring Named Levels
Use the }HierarchyProperties control cube to configure named levels.

For more details about the }HierarchyProperties control cube, see the section about control cubes in the IBM Cognos TM1 Operation Guide.
Procedure

1. In TM1 Architect, click the View menu and select Display Control Objects.
2. In the navigation pane, click to expand the Cubes node.
3. Double-click the }HierarchyProperties control cube.
   The }HierarchyProperties control cube opens.
4. Click the title dimension list to select the dimension for which you want to assign named levels.
5. In the defaultMember cell, enter an existing element name to set as the default member for this dimension.
   The element name you enter here may filter the dimension when TM1 data is retrieved from an external application like IBM Cognos Report Studio.
   Enter the name of the top element in the dimension hierarchy so all the dimension elements are retrieved by default.
   For example, set World as the default member for the Region dimension.
6. In the level000 to level020 cells, enter your own custom name for each hierarchy level that exists in the dimension.

   Note: Named levels are limited to a length of 255 single-byte characters. For details, see "String Length Limit for Named Levels."
7. After configuring named levels, you must do one of the following to apply the changes:
   • Restart the IBM Cognos TM1 server, or
   • Run the RefreshMdxHierarchy function in a TurboIntegrator process. For details, see "Using the RefreshMdxHierarchy TurboIntegrator Function with Named Levels."

Results

Note: Changes to element names or dimension structure are not automatically detected by the named levels feature. If your dimension changes, you must first manually update the named level assignments in the }HierarchyProperties control cube and then either restart the TM1 server or run the RefreshMdxHierarchyTurboIntegrator function to update the MDX hierarchies in the TM1 server.

String Length Limit for Named Levels

Named levels are limited to a length of 255 single-byte characters. Even though the }HierarchyProperties control cube supports long strings, MDX statements can return an error with named levels longer than 255 characters.

If you enter a named level with a length greater than 255 single-byte characters, TM1 displays an error when the server starts up:

4648 ERROR 2008-06-27 13:50:04,532 TM1.Hierarchy User-defined level name modified from ...

Using the RefreshMdxHierarchy TurboIntegrator Function with Named Levels

After configuring or editing the named levels in the }HierarchyProperties control cube, use the RefreshMdxHierarchy function to update the MDX hierarchies in the TM1 server without requiring you to restart the server.
**Procedure**

1. Create a new TI process.
2. Enter the `RefreshMdxHierarchy` function on the ProLog tab using the following format:
   
   ```plaintext
   RefreshMdxHierarchy(dimensionName)
   ```
   
   where the optional parameter, `dimensionName`, can either specify a specific dimension to update or can be blank to update all dimensions.

   For example, to update all dimensions:
   ```plaintext
   RefreshMdxHierarchy('');
   ```

   To update only the customers dimension:
   ```plaintext
   RefreshMdxHierarchy('customers');
   ```

3. Run the TI process.

---

**Creating Cubes**

You can always create cubes on your local server. You must be the TM1 administrator to create cubes on remote servers.

There are two ways to create cubes:

- **Empty Cube** - Select from a list of existing dimensions in the Creating Cube window to create a new cube with no data.
- **External Data Sources** - Use TurboIntegrator to identify and map dimensions and data from an external data sources to a new or existing cube.

This section documents creating cubes in the Creating Cube window. For information about creating cubes in TurboIntegrator, see the IBM Cognos TM1 TurboIntegrator Guide.

**Ordering Dimensions in a Cube**

Dimensions in a cube have an order that you select when you create a cube. The order you select can affect system performance, so you should give some consideration to the order of dimensions before creating a cube.

As a first step toward ordering dimensions, divide the dimensions into two groups: sparse and dense dimensions. A dense dimension has a high percentage of values for its elements. You can estimate the density by answering this question: If one element in the dimension has a value, keeping the elements of the other dimensions constant, what is the probability that the other elements in the dimension have values?

For example, if you have a budget in January for a given account and region, you probably also have a value for the remaining months. Therefore, the Month dimension is probably dense. Similarly, if you have a budget value for a given month, account, and region, you probably also have an actual value, making ActVsBud a dense dimension.

However, in a worldwide sales cube, you probably do not sell every product in every region. Therefore, you would treat Product and Region as sparse dimensions.

We generally recommend that you order the dimensions as follows: smallest sparse to largest sparse, followed by smallest dense to largest dense. However, some flexibility is required. For example, it is probably better to put a very small, dense...
dimension such as ActVsBud that has only two or three elements before a very large but sparse dimension, such as Product, which might have thousands of elements.

Creating a Cube

Follow these steps to create a cube.

**Procedure**

1. Open the Server Explorer.
2. In the Tree pane, select Cubes beneath the server on which you want to create the cube.
3. Click Cubes, Create New Cube.
   The Creating Cube dialog box opens. The Available Dimensions box on the left lists the dimensions stored on the server.
4. Type a cube name in the Cube Name field.

   **Note:** If you do not type a name, TM1 names the new cube Unnamed.

5. In the Available Dimensions box, double-click the name of the dimension you want to use as the first dimension in the new cube.
   The dimension name moves to the Dimensions in new cube box.
   You can also use the button to move selected names from the Available Dimensions box to the Dimensions in new cube box. To select multiple adjacent names, click and drag across the names. To select multiple non-adjacent names, hold down CRTL, and click each name.
6. Repeat the selection process for all the dimensions you want to include in the new cube. You must select at least two dimensions. The maximum number of dimensions is 256.
7. Using the up and down arrows, rearrange the dimensions if necessary. To remove a dimension from the list, double-click the dimension name.
8. If you want to specify the cube properties, click Properties. If you do not want to assign cube properties, skip to step 13.
   The Cube Properties dialog box opens.
   From here, you can set a Measures and Time dimension for the cube, and specify if the cube is loaded automatically or on demand.

   **Note:** OLE DB for OLAP clients may include provisions for referencing Measures and Time dimensions. TM1 does not reference Measures and Time dimensions, but does allow you to set these properties for other OLAP clients that may access the cube.
9. To set a Measures dimension, select a dimension from the Measures Dimension list.
10. To set a Time dimension, select a dimension from the Time Dimension list.
11. Specify how to load the cube:
    • To load the cube into server memory only when a client requests cube data, select the Load On Demand box.
    • To automatically load the cube into memory when the server starts, clear the Load On Demand box.
12. Click OK to save the properties and return to the Creating Cube dialog box.
13. Click Create Cube to create the cube.
The Server Explorer window opens. The new cube displays in alphabetical order in the Cubes list in the Tree pane.

Optimizing the Order of Dimensions in a Cube

If you're not extremely familiar with your business data, it's possible to specify an order of dimensions during cube creation that results in less than optimal performance. Similarly, it's possible for the distribution of data in a cube to change over time, making the order of dimensions specified during cube creation less than ideal. To address these issues, TM1 includes a feature that lets you optimize the order of dimensions in a cube, thereby consuming less memory and improving performance.

When you optimize the order of dimensions in a cube, TM1 does not change the actual order of dimensions in the cube structure. TM1 does change the way dimensions are ordered internally on the server, but because the cube structure is not changed, any rules, functions, or applications referencing the cube remain valid.

As you change the order of dimensions, you can instantly view a report detailing the impact your changes have on cube memory consumption.

For the following reasons, you should optimize the order of dimensions in a cube only in a development environment while you are trying to determine optimal cube configuration:

- Significant memory resources are required for the IBM Cognos TM1 server to reconfigure the order of dimensions in a cube. During the re-ordering process, the temporary RAM on the TM1 server increases by a factor of two for the cube that you are re-ordering. For example, a 50 MB cube requires 100 MB of RAM to reconfigure.
- Re-ordering puts a read lock on the server, locking all user requests while the re-order is performed.

Note: You must be a member of the ADMIN group to optimize the order of dimensions in cubes. The optimization option is only available for cubes on remote servers; you cannot optimize the order of dimensions in cubes on a local server. Also, when you optimize the order of dimensions in a cube, you should not move the string dimensions from the last position, nor move the string dimensions to the last position.

Procedure

1. In the Tree pane of the Server Explorer, select the cube you want to optimize.
2. Click Cube, Re-order Dimensions.
   The Cube Optimizer dialog box opens.
3. Select a dimension in the New Order of Dimensions list box.
4. Click the up or down arrows to change the order of the dimension in the cube.
5. Click Test.
   Note the value next to the Percent Change label. If this value is negative, the new order of dimensions consumes less memory and is therefore more efficient.
6. Repeat steps 3 through 5 until you achieve the most efficient ordering of dimensions.
7. Click OK.
Editing Cube Properties

TM1 allows you to set cube properties that specify measures and time dimensions used by OLE DB for OLAP applications, and that determine whether a cube loads automatically or on demand. Usually, you set these cube properties when you create a cube, but you can edit the properties any time.

Editing Measures and Time Dimension

OLE DB for OLAP client applications include provisions for measures and time dimensions. Even though TM1 clients do not include such provisions, you can use TM1 to set measures and time dimensions for cubes that you access by OLE DB for OLAP clients.

Procedure

1. Select the cube in the Tree pane of the Server Explorer.
2. Click Cube, Properties.
   The Cube Properties dialog box opens.
3. Select a measures dimension in the Measures Dimension list.
4. Select a time dimension in the Time Dimension list.
5. Click OK.

Editing the Load on Demand Property

By default, Cognos TM1 loads all cubes into memory when a server starts. While this allows fast access to Cognos TM1 data, it can consume significant server resources. If your server contains infrequently accessed cubes, you can conserve resources by setting those cubes to load only when a client attempts to access the cube data.

Procedure

1. Select the cube in the Tree pane of the Server Explorer.
2. Click Cube, Properties.
   The Cube Properties dialog box opens.
3. Specify how to load the cube the cube:
   • Select the Load On Demand box to load the cube on demand.
   • Clear the Load On Demand box to automatically load the cube when the server starts.
4. Click OK.

Creating Pick Lists

A pick list is a list of valid values for a specific element or cube cell. When an administrator defines a pick list for an element or a cell, a drop-down menu containing the defined values is available in the specified cell when browsing a cube in any of the TM1 clients.

Values in cells containing a pick list are validated; a user must select one of the predefined values for the cell. If a user attempts to enter a value that is not valid for the cell, an error appears indicating that only values from the pick list can be entered in the cell.

Pick List Usage Notes

You should be aware of the following requirements and behaviors when using pick lists.
• Cell edits applied through data spreading operations and TurboIntegrator processes are **not** validated. Edits applied through either of these methods can result in cell values that do not conform to valid pick list values. Data spreading can be applied to cells containing pick lists only through the data spreading dialog boxes; data spreading shortcuts cannot be used in cells containing pick lists.

• When defining a pick list that contains numeric values, you must use the Cultural Invariant style, which uses a period (.) as a decimal separator. The Cultural Invariant style is equivalent to English style.

• When using pick lists with Excel 2007, you **must** have Excel 2007 Service Pack 2 installed if you want to use pick lists in conjunction with automatic calculation mode. If you are running Excel 2007 without Service Pack 2, you should set Excel's calculation mode to manual. Using automatic calculation in the absence of Service Pack 2 can result in access errors on cells containing pick lists.

• Do not use double quotation marks in a pick list value that may be viewed in the TM1 Web Cube Viewer. In TM1 Web, pick list values that contain double quotation marks prevent display of the contents of the pick list and may make the view unusable. Double quotation marks in pick lists work correctly in TM1 Websheets.

• All pick lists in TM1 Web automatically contain a selectable null value. To make null values available in pick lists in TM1 Architect and TM1 Perspectives, you must explicitly define a null value in a static pick list. Null values cannot be used in dimension or subset pick lists in TM1 Architect and TM1 Perspectives.

**Pick List Types**

You can create three types of pick lists: static, subset, and dimension.

**Static Pick Lists**

A static pick list is composed of a comma-delimited list of values using the syntax `static:value1:value2:value3:value4`.

For example, `static:red:orange:yellow:green` results in a pick list containing the values red, orange, yellow, and green.

To include a null value at the beginning or in the middle of a static pick list, use two consecutive colons without intervening characters in the pick list definition. For example, `static::value1:value2::value3:value4` results in a pick list with a null value before value1. Similarly, `static:value1:value2::value3:value4` results in a pick list with a null value between value2 and value3.

To include a null value at the end of a static pick list, insert a colon without a following value at the end of the pick list definition. For example, `static:value1:value2::value3:value4:` results in a pick list with a null value following value4.

**Subset Pick Lists**

A subset pick list contains values corresponding to all elements of a named subset. If the members of the subset change, the values available in the pick list change correspondingly.

A subset pick list is defined using the syntax `subset:dimension_name:subset_name`.

For example, `subset:Products:Winter` results in a pick list containing all elements from the Winter subset of the Products dimension.
Dimension Pick Lists
A dimension pick list contains values corresponding to all elements of a
dimension. If the members of the dimension change, the values available in the
pick list change correspondingly.

A dimension pick list is defined using the syntax `dimension:dimension_name`.

For example, `dimension:Months` results in a pick list containing all elements from
the Months dimension.

Creating Pick Lists with Element Attributes
The easiest way to create a pick list is to define a text attribute named Picklist for a
dimension. You can then specify the members of a pick list for each element within
the dimension, using any of the pick list types described above. When an element
has a pick list defined, any cube cell identified by that element displays a
drop-down list containing the pick list values.

Procedure
1. In the Server Explorer, right-click the dimension for which you want to define
   pick lists, then click Edit Element Attributes.
2. In the Attributes Editor, click Edit, Add New Attribute.
3. In the New Attribute dialog box, enter Picklist as the attribute name.
4. Select Text as the attribute type.
5. Click OK.
   The Attributes Editor now contains a new column titled Picklist.
6. For each element for which you want to create a pick list, enter a valid pick list
definition at the intersection of the element name and the Picklist column.
   a. To enter a static pick list, enter a comma-delimited list of values using the
   b. To enter a subset pick list, enter the pick list definition using the syntax
      `subset:dimension_name:subset_name`.
   c. To enter a a dimension pick list, enter the pick list definition using the
      syntax `dimension:dimension_name`.
7. Click OK to close the Attributes Editor and save the pick list definitions.

Creating Pick Lists with Control Cubes
You can also create pick lists with control cubes. This gives you greater control
over which cube cells should contain pick lists and allows greater flexibility in
defining pick lists for individual cells. You can also create rules for the pick list
control cube, which allows you to define pick lists for any section of a cube, from a
single cell to the entire cube.

A pick list control cube is composed of the same dimensions as the regular cube it
is associated with, along with an additional dimension named \{Picklist. The
\{Picklist dimension contains a single string element, named Value.

Creating a Pick List Control Cube
Use this procedure to create a pick list control cube.
**Procedure**

In the Server Explorer, right-click the regular cube for which you want to create a pick list control cube, then **Create Pick List Cube**.

A new control cube is created, using the naming convention `|Picklist_cubename`. For example, when you create a pick list control cube for the Orders cube, the control cube is named `|Picklist_Orders`.

**Note:** If you cannot view control cubes in the Server Explorer, click **View, Display Control Objects** to enable the display of control cubes and other control objects.

**Defining Pick Lists for Individual Cells in a Control Cube**

Follow these steps to define pick lists for individual cells in a control cube. The pick lists defined in the control cube are used to display pick list values in the associated regular cube.

**Procedure**

1. Double-click the control cube in the Server Explorer.
   
   The control cube opens in the Cube Viewer.

2. Configure the view of the control cube as necessary to view the cells for which you want to define pick lists. For details on configuring cube views, see "Working with Cube Views" in the IBM Cognos TM1 User Guide.

3. In each cell for which you want to create a pick list, enter a pick list definition. You can enter any of the pick list types in the control cube: static, subset, or dimension.

4. Click **File, Recalculate** to recalculate the cube view.

**Using Rules to Define Pick Lists In a Control Cube**

Rules that define pick lists follow the same conventions as all other TM1 rules. As with other TM1 rules, a rule statement that defines a pick list must include an area definition (the portion of the cube to which the rule applies), a string qualifier, and a formula. In the case of pick list rules, the formula is simply the pick list definition that you want to apply.

Also as in all other TM1 rules, when multiple rule statements apply to overlapping areas, the statements should be ordered from most restrictive area to least restrictive area.

For more information on creating rules, including details on specifying an area definition, please see the IBM Cognos TM1 Rules Guide.

**Procedure**

1. In the Server Explorer, right-click the |Picklist control cube for which you want to create a rule, then click **Create Rule**.
   
   The Rules Editor opens.

2. Using a standard rules area definition, specify the cells you want the pick list to appear in.

3. Immediately after the area definition, type `|=S:`. This is the string qualifier, indicating that the rule applies to string cells.

4. Immediately after the string qualifier, enter a pick list definition, enclosed in single quotes, then enclosed in parentheses. For example, `('static:summer:winter')`. 
5. Immediately after the pick list definition, type a semi-colon (;) to terminate the rule statement.

Using the procedure described in these steps should result in a rule statement similar to the following example, which indicates that an cell identified by the fabric element will display a static pick list containing the values wool, cotton, silk, and nylon.


Other Pick List Rules Examples:

The following examples illustrate rules statements that define pick lists.

<table>
<thead>
<tr>
<th>Rule statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>['size','shirts']=S:('static:16:17:18');</td>
<td>This rule statement indicates that any cell identified by the elements size and shirts will display a static pick list composed of the values 16, 17, an 18.</td>
</tr>
<tr>
<td>['size',{'sweaters','vests','jackets'}]=S:('static:XS:S:M:L:XL');</td>
<td>This rule statement indicates that any cell identified by the element size and any of the elements sweaters, vests, or jackets will display a static pick list composed of the values XS, S, M, L, and XL.</td>
</tr>
<tr>
<td>['fabric']=S:('dimension:materials');</td>
<td>This rule statement indicates that any cell identified by the element fabric will display a pick list composed of all elements in the materials dimension.</td>
</tr>
</tbody>
</table>

Excluding Cells from Pick Lists

There may be some circumstances when you do not want an individual cell or a specific area of a cube to use pick lists. To prevent a cell from displaying a pick list, enter none in the appropriate pick list control cube cell or use ('none') as the formula in a rules statement. For example, ['season']=S:('none');

Null Values in Pick Lists

All pick lists in TM1 Web (both Websheets and the Cube Viewer) always contain a null value that a user can select. The null value is automatically inserted into all pick lists in TM1 Web; it does not have to be explicitly defined.

Pick lists in TM1 Architect and TM1 Perspectives only contain a null value if the pick list is a static type and a null value has been explicitly defined for the pick list. Dimension pick lists and subset pick lists can never contain a null value when used in TM1 Architect and TM1 Perspectives.

Selecting Null Values in Pick Lists

There are two ways to select null values from pick lists.

**Procedure**

1. In any of the TM1 clients, you can click the pick list and then click the null value.

   **Important:** In TM1 Perspectives slices and Active Forms, do not click the null value in a string cell. This will delete the DBRW formula from the cell and you will no longer be able to retrieve data for that cell from the IBM Cognos TM1 server. You can safely click the null value in numeric cells.
2. In TM1 Web and Architect, you can press the Delete key in a cell containing a pick list to select the null value.

   **Important:** Do not press Delete to select a null value in TM1 Perspectives slices or Active Forms. This will delete the DBRW formula from the cell and you will no longer be able to retrieve data for that cell from the TM1 server.

### Pick List Order of Precedence

When multiple pick lists apply to an individual cube cell, the following order of precedence is used to determine which pick list is used in the cell:

- If a pick list control cube exists and contains a pick list definition for the current cube cell, the definition in the pick list control cube is used.
- If a pick list control cube does not exist, the elements that identify the current cell are examined in reverse order in a search for Picklist element attributes. The first Picklist element attribute that is encountered in this search is used in the cell.

### Replicating Cubes between Servers

Using the TM1 Replication feature, you can copy cubes and other associated objects from a remote server to your local server, or between two remote servers. You can also synchronize the data updates among the copied cubes either at specified time intervals or on demand.

Replication offers the following advantages:

- Enhances response time because you can update a cube locally without communicating across a network.
- Lets you copy the latest shared data to a laptop for presentations outside of your organization.

TM1 provides bi-directional synchronization for replicated data. During the synchronization process, TM1 checks the servers involved in a replication for the latest updates to the data, and then copies the latest updates to the other servers.

Replication creates a relationship between two cubes and between two servers. These relationships are described in "Cube Relationships."

### Cube Relationships

Replication creates a relationship between two cubes:

- **Source cube** - The original cube in a replication
- **Mirror cube** - A copy of the source cube

Depending on your access privileges, you can replicate a single cube on many different servers, and you can replicate a replicated cube.

### Server Relationships

Before replicating a cube, you need to log on to a remote server and create a replication connection. Replication creates a relationship between two servers:

- **Source server** - The remote server you log in to
- **Target server** - The server you logged in from
The Server Explorer window lists the current replication connections beneath the Replications icon. In this example, regions 1 is the target server and sales is a source server.

**Required Access Privileges**

The following access privileges are required to replicate a cube:

- Your security group must have Read access or higher to the cube you want to replicate.
- You must be the TM1 administrator on the target server. On your local server, you are always the TM1 administrator.
Chapter 3. Advanced Calculations for Business Data

This section describes how to create "drill-through" processes and rules, which enable you to link the cells with related data to provide the details or context for cube values.

For a thorough examination of rules in a business scenario, see the IBM Cognos TM1 Rules Guide. The guide contains a tutorial that steps you through developing rules in a business environment.

Note: The images of the Rules Editor in this section show the old Rules Editor. To see images of the new Rules Editor, see the IBM Cognos TM1 Rules Guide.

Depending on your platform, some of the supplied example data may not be available to all users.

Overview of Cube Rules

The most common calculations in OLAP applications involves aggregating data along a dimension. In TM1, you create these calculations by using consolidation hierarchies. For example, in a Month dimension, you can define a quarterly total that sums the January, February and March values.

In many applications, you need to perform calculations that do not involve aggregating, such as cost allocations and exchange translations. With cube rules, you can create formulas to perform these calculations.

With cube rules, you can perform the following tasks:
- Multiply prices by units to yield the sales amounts.
- Override consolidations when necessary. For example, you can prevent a quarterly price from displaying a tally of individual monthly prices.
- Use data in one cube to perform calculations in another cube, or share data between cubes. For example, you can pull sales data into a cube that contains Profit and Loss information.
- Assign the same values to multiple cells.

Note: You must be a member of the ADMIN group to create or edit TM1 rules.

You associate a cube rule with an individual cube. When you create a rule, TM1 stores the rule information in two files:

- cube_name .rux - Stores the compiled rules. When you load a cube for which you have defined rules into memory, TM1 searches for the cube's .rux file in the data directory containing the cube.
- cube_name .blb - Stores the format information for the Rules Editor.

Note: If you want to edit a .rux file in a text editor other than the Rules Editor, be sure to delete the corresponding .blb file. If you do not delete the file, there will be a discrepancy between the contents of the .rux file and the display in the Rules Editor, as the .blb file determines the display in the Rules Editor.
Guidelines for Writing TM1 Rules Statements

The general format of a rules statement is:

```
[Area]=Formula;
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>Specifies the portion of a cube affected by the rule.</td>
</tr>
<tr>
<td>Formula</td>
<td>Describes how TM1 calculates the cells in the cube area.</td>
</tr>
</tbody>
</table>

To restrict a rule to simple values in the Area:

```
[Area]=N:Formula;
```

To restrict a rule to consolidated values in the Area:

```
[Area]=C:Formula;
```

**General Considerations**

- The syntax is not case-sensitive. You can use both uppercase and lowercase letters.
- You can use spaces within rules to improve clarity.
- A rules statement can occupy one or more lines in the Rules Editor. A statement can also contain one or more formulas.
- You must end each statement with a semicolon (;).
- To add comments and to exclude statements from processing, insert a number sign (#) at the beginning of a line or statement. For example:

  ```
  # The following rule is not active
  # ['Gross Margin']=['Sales']*0.53;
  ```

**Syntax for Describing the Area**

The Area identifies one or more cells in a cube.

Consider the following guidelines when you create an Area definition.

- Specify no dimension elements, or one or more dimension elements.
- Each element must be from a different dimension of the cube.
- Enclose each element in single quotes.
- Use commas to separate each element.
- Enclose the entire Area definition in brackets.

The following table shows four Area examples. Each successive example narrows the scope.

<table>
<thead>
<tr>
<th>Sample Area</th>
<th>Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ ]</td>
<td>All cells in the cube.</td>
</tr>
<tr>
<td>['January']</td>
<td>All cells identified by a January element.</td>
</tr>
<tr>
<td>['Sales','January']</td>
<td>All cells identified by the Sales and January elements.</td>
</tr>
<tr>
<td>['Germany','Sales','January']</td>
<td>All cells identified by the Germany, Sales, and January elements.</td>
</tr>
</tbody>
</table>
Using Subsets in an Area Definition
You can use a subset in place of a single element in an Area definition by enclosing all subset members in curly braces.

For example, the following Area definition applies a rule to all cube cells identified by the element Sales and the element January, February, or March:

`['Sales', {'January', 'February', 'March'}]`

Using Special Characters and Non-unique Element Names in an Area Definition
You can use the syntax 'dimensionname':elementname' in a rules Area definition to specify elements that are not unique to a single dimension, or for dimension names that contain special characters.

For example,

`['Units', 'Mar', 'Groups':'ADMIN']`

allows you to write a rule for the |Groups dimension, which contains the curly brace () special character.

Similarly,

`['Units', 'Mar', 'Region':'North America']`

lets you write a rule when the element North America is not unique to the Region dimension.

Syntax for Formulas
A rules formula is an expression composed of:

- Numeric constants
- Arithmetic operators and parentheses
- Numeric and string functions -- see the IBM Cognos TM1 Reference Guide.
- Conditional logic
- Cube references

Numeric Constants
The simplest components of rules formulas are numeric constants.

- Consists of numerals, an optional leading minus sign (-), and an optional decimal point. For example, 5.0, 6, -5. Some examples of invalid numeric constants are: 1-, 1A, 3..4.
- Contains a maximum length of 20 characters.
- You can use scientific notation to enter a numeric constant.

For example, the following rules statement assigns the value 200 to all cells in the cube.

`[ ] = 200;`

Arithmetic Operators
You can combine numeric constants with the following arithmetic operators.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ (Plus sign)</td>
<td>Addition</td>
</tr>
</tbody>
</table>
### Operator Meaning

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>- (Minus sign)</td>
<td>Subtraction</td>
</tr>
<tr>
<td>* (Asterisk)</td>
<td>Multiplication</td>
</tr>
<tr>
<td>/ (Forward slash)</td>
<td>Division - returns an undefined value and displays N/A in the view when you divide by zero</td>
</tr>
<tr>
<td>\ (Backslash)</td>
<td>Zero Display Division - same as Division operator, but returns zero when you divide by zero.</td>
</tr>
<tr>
<td>^ (Caret)</td>
<td>Exponentiation</td>
</tr>
</tbody>
</table>

### Using Conditional Logic

Use the IF function to include conditional logic in rules. The general format is:

\[ \text{IF}(\text{test}, \text{value1}, \text{value2}) \]

- The IF function returns one of two values depending on the result of a logical test.
- When the expression Test is true, the IF function returns Value1.
- When the expression Test is false, the IF function returns Value2.
- The data type returned by an IF function is determined by the data types of Value1 and Value2.
- Value1 and Value2 must be the same data type, either string or numeric.
- An IF function where Value1 is a string and Value2 is a number yields an error statement.

You can also nest IF statements:

\[ \text{IF}(\text{test1}, \text{value1}, \text{IF} (\text{test2}, \text{value2}, \text{value3})) \]

The following table shows two IF examples.

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF (7&gt;6,1,0)</td>
<td>yields 1</td>
</tr>
<tr>
<td>IF (7&gt;6, 'True', 'False')</td>
<td>yields 'True'</td>
</tr>
</tbody>
</table>

### Using Comparison Operators

You can compare numbers with the following operators.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;</td>
<td>Greater than</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equal to</td>
</tr>
<tr>
<td>Operator</td>
<td>Meaning</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal to</td>
</tr>
<tr>
<td>=</td>
<td>Equal to</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>Not equal to</td>
</tr>
</tbody>
</table>

To compare two string values, insert the @ symbol before the comparison operator, as shown in the following example:

If ('A'@='B',0,1) yields the number 1.

You can combine logical expressions with logical operators.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp; (Ampersand)</td>
<td>AND</td>
<td>(Value1 &gt; 5) &amp; (Value1 &lt; 10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Returns TRUE if the value is greater than 5 and less than 10.</td>
</tr>
<tr>
<td>% (Percent sign)</td>
<td>OR</td>
<td>(Value1 &gt; 10) % (Value1 &lt; 5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Returns TRUE if the value is greater than 10 or less than 5.</td>
</tr>
<tr>
<td>~ (Tilde)</td>
<td>NOT</td>
<td>~(Value1 &gt; 5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Equivalent to (Value1 &lt;= 5)</td>
</tr>
</tbody>
</table>

You can concatenate strings using the pipe (|) character.

For example, the following expressions returns Rheingold.

(Rhein | gold)

If the string resulting from a concatenation is longer than 254 bytes, TM1 returns an error.

**Using Cube References**

All rules formulas contain cube references, which point to areas within a cube for data. The cube references can point to the cube for which you are writing a rule (internal cube references) or to areas within other cubes (external cube references).

**Internal Cube References**

Internal cube references use the same syntax as the area for which you write the rule. Examples include:

['January']

['Sales', 'January']

['Germany', 'Sales', 'January']

In the following example, TM1 calculates the Gross Margin for Germany by multiplying the Sales for Germany in the same cube by 0.53:
['Gross Margin','Germany']=['Sales']×0.53;

**External Cube References**

Use the `DB` function to point to external cubes.

```
DB('cube', dimension1, dimension2,...dimensionn)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cube</td>
<td>Name of the external cube.</td>
</tr>
</tbody>
</table>
| dimension| One of the following arguments:  
  - The name of an element in a dimension of the external cube, enclosed in single quotes.  
  - The name of a dimension preceded by an exclamation mark (!), which is called *variable notation*. An argument using variable notation returns the current dimension element in the cell to which a rule statement applies. For example, in a rules-calculated cell that is identified by the Germany element of the Region dimension, `!Region` returns Germany.  
  - An expression that resolves to an element name. |

Specify a dimension argument for each dimension of the external cube. You must order the dimension arguments to correspond to the order of the dimensions in the external cube.

In the following rules statement, all Sales values in the internal cube are computed by multiplying Units in the internal cube by the values in the external PriceTab cube:

```
['Sales']=['Units']×DB('PriceTab',!Region,!Product,!Month)
```

- The PriceTab cube contains only prices. Each of its cells is identified by an element in three dimensions: Region, Product, Month. The internal cube contains these dimensions and at least one other dimension that has both the Sales and Units elements.
- Every Sales cell in the internal cube is identified by Sales and elements in the three dimensions the internal cube shares with the PriceTab cube. To populate any Sales cell, TM1 pulls a PriceTab value located at the intersection of the corresponding elements in the shared dimensions.
- The external cube can differ from the internal cube in terms of the number of dimensions and the number of elements along each dimension. However, a dimension you reference as a variable (as in `!Region` or `!Product`) must at least contain all the elements found in the internal cube’s corresponding dimension.

**Arranging Rules Statements**

When more than one statement in a rule applies to the *same* Area, the first statement takes precedence.

Consider this example. A cube named Priority has two dimensions, Region and Year. The rule has four statements:

```
['Germany', 'Year1'] = 10;['Year1'] = 5;['United States'] = 6;[
] = 2;
```

Here are sample values for the Priority cube, all of which are derived by the preceding rule.
<table>
<thead>
<tr>
<th>Region</th>
<th>Year 1</th>
<th>Year 2</th>
<th>Year 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>France</td>
<td>5</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Germany</td>
<td>10</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>United States</td>
<td>5</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

TM1 processes the rule statements as follows:
- The first statement assigns the value 10 to the Germany, Year1 cell. The first statement takes precedence over the second statement, which specifies that all Year1 cells contain 5.
- The second statement takes precedence over the third statement. Therefore, the cell for United States, Year 1 contains 5, even though the third statement specifies that all values for United States should be 6.
- The last statement \( [ ] = 2 \) specifies that all values in the cube contain the value 2. This rule applies to all cells that are not affected by preceding statements, such as the cell France, Year2.

**Specifying Different Rules at the N: and C: Levels**

It is often necessary to differentiate the way C: level and N: level cells within an area are treated.
- **N: Level Cells** - Identified only by simple elements.
- **C: Level Cells** - Identified by at least one consolidated element.

Use the following syntax to write a rules statement that applies only to N: level cells in an area:

\[ [\text{Area}] = \text{N:}[\text{Formula}]; \]

For example:
\[ ['Sales'] = \text{N:}'Price'\times['Units']\times1000; \]

Use the following syntax to write a rules statement that applies only to C: level cells in an area.

\[ [\text{Area}] = \text{C:}[\text{Formula}]; \]

For example:
\[ ['Price'] = \text{C:}'Sales'\div['Units']\times1000; \]

When a specific area of a cube is calculated differently at the C: and N: levels, you can use the following syntax:

\[ [\text{Area}] = \text{N:}[\text{Formula A}]; \text{C:}[\text{Formula B}]; \]

For example:
\[ ['Price'] = \]
\[ \text{N:DB('PriceCube', !Actvsbud, !Region, !Model, !Month)}; \]
\[ \text{C:}'Sales'\div['Units']\times1000; \]
Bypassing Rules

By using the STET function, you can bypass the effect of a rules statement for specific areas of a cube.

For example, you might want to write a rules statement for Gross Margin that applies to all regions except France. You can write the general rule and the exception in two ways.

- Write the STET statement first followed by the general statement:

  ```
  ['Gross Margin', 'France'] = STET;
  ['Gross Margin'] = ['Sales'] * 0.53;
  ```

- Write one rules statement that includes an IF function:

  ```
  ['Gross Margin'] = IF(!Region @= 'France', STET, ['Sales'] * 0.53);
  ```

Qualifying Element Names

When you want to limit a rules statement to values identified by an element that appears in multiple dimensions, qualify the element name with its dimension name using the following syntax:

```
['dimname': 'element']
```

For example, if Total occurs in both the Region and Product dimensions and you want the rule limited to cells identified by Total in the Region dimension, specify:

```
['Region': 'Total'] =
```
Server Data Directory box in the TM1 Options dialog box. The directory is the location where your TM1 client saves the rules worksheets (.xru files).

**Note:** If you do not set a directory, or if the directory is not valid, TM1 cannot save your rules worksheets.

**Procedure**

1. Click **TM1, Rule Worksheets, New** in Excel.
   
   The Select Cube for Rules dialog box opens. The list contains the cubes on your local server and any servers you have logged in to during the current TM1 session.

2. Select a cube and click **OK**.
   
   TM1 displays a blank rules worksheet that looks like other Excel worksheets, except the first column has a width of 100.

3. Place each rule statement on a separate line in column A, as in the following example.
   
   ```plaintext
   ['Gross Margin%']=['Gross Margin']/['Sales']*100;
   ['Price']=N:DB('PriceCube',!actvsbud,!region,!model,!month);
   C:['Sales']/['Units']*1000;
   ['Sales']=N:['Price']*['Units']\1000;
   ```
   
   You can click **TM1, Edit Formula** to access the **TM1 Edit Formula** dialog box. This dialog box, which helps you construct accurate rules statements, provides all of the functionality found in the Rules Editor. The Rules Editor is described later in this section.

**Saving Rules Worksheets**

To save the rule in a format that TM1 can use, click **TM1, Rule WorkSheets, Save** in Excel.

The **Save** option updates the rule worksheet file (cube.xru) and creates a compiled rules file (cube.rux). TM1 saves the .xru file in your local server data directory, and saves the .rux file in the data directory for the IBM Cognos TM1 server. TM1 immediately applies the new rules to the cube.

**Note:** If you click **File, Save** in Excel, only the cube.xru file is updated. To use the new rule in a cube, you must create the compiled rules file.

**Rules and Dimension Consolidations**

Rules work in concert with consolidations you define in dimensions. Although you can define consolidations using rules, this is not recommended for performance reasons. Consolidations defined in dimensions are calculated much more quickly than rules-derived values, especially in very large, sparse cubes.

**Order of Calculation**

Rules take precedence over consolidations within dimensions.

When TM1 calculates a cube cell and a consolidation by a rule, TM1 examines the rules statement first. However, if the rules statement refers to the cells that are the result of consolidations, TM1 first performs the consolidation and then calculates the rules statement using the results.
Conversely, if you define a cell by consolidation only, TM1 looks at the values needed to perform the consolidation. When some values are the result of calculation rules, TM1 then performs the rules calculation before performing the consolidation.

**Overriding C: Level Elements with Rules**

You should avoid writing a rule that overrides a consolidated value that is a component of another consolidation.

A simple example illustrates this issue. Suppose you have a two-dimensional cube named Sales that is composed of the dimensions Product and Month, with product (Total) and quarterly (1 Quarter) consolidations defined.

To calculate the grand total (Total, 1 Quarter), TM1 can consolidate the product totals for each month or consolidate quarterly totals for each product.

Suppose further that you write a rule that calculates a value for Total product sales in Jan, and that the rules-calculated value does not sum the individual product values for Jan. A rule that defines the value of Total products in Jan as 999 serves as an illustration.

```
['Jan','Total'] = 999;
```

If the grand total is calculated by consolidating the product totals for each month, the value will differ from the consolidation of the quarterly totals for each product. This is because the rules-calculated value for total product sales in Jan overrides the natural consolidation defined in the Product dimension.
You have no control over the order in which TM1 performs dimension consolidations. Furthermore, depending on which consolidation path is optimal at any given moment, TM1 may alternate between paths. Consequently, you may request the Total, 1 Quarter value twice in the same session and get different results.

You can remedy this situation by writing a rules statement that calculates the value of the Total, 1 Quarter consolidation as the sum of its immediate children along the Month dimension, thereby overriding the Product dimension consolidation. The statement \([''Total'']=ConsolidateChildren('Month')\) performs this calculation.

However, there remains an implicit inconsistency when viewing the cube: the sum of the quarterly totals for each product is different from the sum of product total for each month. Thus, overriding C: level values that are components of other consolidations is not recommended.

### Stacking Rules

A rules statement can refer to a cell in a cube that is defined by other rules statements. TM1 stacks these rules statements until it can obtain a final value, and then works back to return a result. The number of levels of stacking that TM1 can accommodate is limited only by the available memory.

If a circular reference occurs within a rules stack, or the maximum level of stacking is exceeded, TM1 displays the error message:
Error Evaluating Rule: Possible Circular Reference

Here is an example of a circular reference:

\[
['Sales'] = ['Units'] * ['Price'];
['Price'] = ['Sales'] / ['Units'];
\]

Sample Applications

This section contains examples of commonly used rules applications. Examine these examples to develop an understanding of the syntax and scope of rules.

Calculating Ratios

In the following example, a rule calculates the Gross Margin as a percentage of the Sales in the SalesCube cube. You associate this ratio with the Gross Margin%, a new numeric element in the Account1 dimension.

First, you need to create the Gross Margin% element.

Creating the Margin% Element

If your SalesCube dimension already contains the GrossMargin element, skip to the next section. Follow these steps if you need to add the element GrossMargin% to the Account1 dimension.

Procedure

1. Open the Server Explorer.
2. Select the Account1 dimension.
3. Click Dimension, Edit Dimension Structure.
   The Dimension Editor opens.
4. Click Edit, Insert Element.
   The Dimension Element Insert dialog box opens.
5. Type Gross Margin% and click Add.
6. Click OK.
7. Save the dimension.

Creating the Margin% Rule

Follow these steps to create the Gross Margin% formula.

Procedure

1. In the Server Explorer, right-click Sales Cube. If the rule already exists, click Edit Rule. If you have not yet created the rule, click Create Rule.
   The Rules Editor opens.
2. Click Area .
   The Reference to Cube dialog box displays the dimensions of the SalesCube cube.
3. Click Account1.
   The Subset Editor window opens.
4. In the left pane, select Gross Margin% and click OK.
   The Reference to Cube dialog box reopens.
5. Click OK.
   The Rules Editor displays ['Gross Margin%'] in the entry field.
6. Click Equal ▼.
7. Click Area ▼ again, and click Account1.
8. In the Subset Editor window, select Gross Margin and click OK.
9. Click OK in the Reference to Cube dialog box.
10. Click Zero Display Division ▼.
   TM1 places a division sign after [‘Gross Margin’].
   Note that there are two division buttons in the Rules Editor.
   Division Operator ▼ - If you use this division operator in a rule that results in division by zero, TM1 returns an undefined value, and displays N/A in the view.
   Zero Display Division Operator ▼ - If you use this division operator in a rule that results in division by zero, TM1 returns the value 0.
11. Click Area ▼ again, and click Account1
12. In the Subset Editor window, select Sales and click OK.
13. Click OK in the Reference to Cube dialog box.
14. Click Multiplication ▼.
15. Type 100 at the end of the formula.
16. Click Semicolon ▼.
   Note: You must end all rules statements with a semicolon.
   The complete rule should now appear as follows.
   [‘Gross Margin%’]*[‘Gross Margin’]/[‘Sales’]*100;
17. Click Save.

Results

TM1 saves the rule and applies it to the cube. A new Rule object opens beneath the SalesCube cube. A rule always bears the same name as the cube with which it is associated.

Understanding the Generated Rule

Let’s examine the components of the new rules statement.

- **Area** - Specifies the cube area that TM1 calculates. In this example, the Gross Margin% element identifies all cell values derived through a rule.
- **Formula** - Defines the calculation.
- **Terminator** - Terminates all rules statements with a semicolon (;).

   ![Diagram of rule components]

   [‘Gross Margin%’]*[‘Gross Margin’]/[‘Sales’]*100;

For more information about rules syntax, see the IBM Cognos TM1 Rules Guide.

Browsing the Gross Margin% Values

The sample view salesmargin% now displays the calculated values for GrossMargin%.
Procedure
1. In the Server Explorer window, click the Views icon for the SalesCube cube.
2. Double-click the view salesmargin%.
   The view opens in the Cube Viewer.
   The values for Gross Margin%, which are derived through the rule you just created, appear in the view.

Sharing Data Between Cubes
The SalesCube cube does not contain price data. The price information for this cube is stored in a separate four-dimensional cube called PriceCube.

You can share values between cubes by using the DB rules function. Values are stored in one cube and referenced in other cubes. The following example shows how rules for one cube can reference values in a separate cube.

Procedure
1. In the Server Explorer, double-click the SalesCube rule.
   The Rules Editor opens.
2. Position the cursor on the second entry line.
3. Click Area.
   The Reference to Cube dialog box displays the dimensions of the SalesCube cube.
4. Click Account1.
   The Subset Editor window opens.
5. In the left pane, select Price and click OK.
   The Reference to Cube dialog box reopens.
6. Click OK.
   The Rules Editor displays [’Price’] in the entry field.
7. Click Equal.
8. Click Database Reference.
   The Select Cube dialog box opens.
9. Select PriceCube and click OK.
   The Reference to Cube dialog box displays the dimensions of the PriceCube cube.
10. Click OK.
    The following formula opens in the Rules Editor.
    [’Price’]=DB(’PriceCube’,!Actvsbud,!Region,!Model,!Month)
    Read the formula as follows: Any cell in the SalesCube cube that is identified by the Price element takes its value from a cell in the PriceCube cube. The location of the PriceCube cell is found at the intersection of corresponding elements in the four dimensions that PriceCube shares with SalesCube.
    For example, the SalesCube cell identified by the elements Actual, Germany, S Series 1.8 L Sedan, Price, Jan takes its value from the PriceCube cell identified by the elements Actual, Germany, S Series 1.8 L Sedan, Jan.
11. Click Semicolon to place a semicolon at the end of the formula.
12. Click Save to save the rule.
You can now change a price in the PriceCube cube and see the change reflected in the SalesCube cube. However, you cannot edit the prices in SalesCube because they are derived through the rule you just created.

The sample view SalesPrice includes the price values.

To open the sample view SalesPrice:
13. In the Server Explorer window, click the Views icon for the SalesCube cube.
14. Double-click the view SalesMargin\%.

The view opens in the Cube Viewer, complete with the Price values pulled from the PriceCube cube.

Calculating Sales
In the previous exercise, you brought prices into the SalesCube cube. If you change a price in PriceCube, TM1 does not change the corresponding sales value in SalesCube. This is because the sales values in SalesCube exist as data values in the cube. You need to create a rule to derive the sales values in SalesCube from the prices and units.

Procedure
1. Reopen the Rules Editor for the SalesCube cube.
2. Beneath the ['Price'] formula, enter the following rule:
   ['Sales']=['Price']*['Units']/1000;

   Note: All values in the SalesCube cube are in thousands except the Price values, which are actual numbers. Because Sales numbers should remain in thousands, you divide by 1000 in the rules statement.
3. Click Save to save the SalesCube rule.
4. Reopen the sample SalesPrice view.

   TM1 shades all cells identified by the Sales element, which indicates that the values in these cells are derived through rules.
5. Change the January units value to 10,000 by typing 10000 in the cell at the intersection of Units and Jan.
6. Press F9 to recalculate the cell values.
7. Observe the new sales value for January.
Note that the Gross Margin% value for January updates, as this value is derived through a rule, which references the Sales element.

8. Restore the value 313 to the cell at the intersection of Jan and Sales.

Restricting Rules to Simple Values

In the previous exercise, you created a rule that applies to all cells containing sales numbers. This type of rule supersedes consolidations within dimensions, producing incorrect results. In the following exercise, you inspect a quarterly total through a slice worksheet, and restrict the SalesCube rule to simple values, thereby allowing consolidations to function properly.

Procedure

1. Open the sample view Sales1qtr of the SalesCube cube.

2. Click Slice to slice the view to an Excel worksheet.
   The slice should appear as follows.
3. Save the slice worksheet with the name Test.

4. Examine the Sales value for 1Quarter in cell B10.

   According to the SalesCube rule, the value is the product of multiplying the
   first quarter price by a fraction (1/1000) of first quarter units. Instead, the value
   should be the consolidation of sales for the first three months. However, the
   values derived by the SalesCube rule (as it currently exists) take precedence
   over the values derived through consolidation. To correct this, you need to
   modify the rule so that it does not calculate the values for consolidated
   elements.

5. Open the SalesCube rule in the Rules Editor.

6. On the third line, insert N: in front of ['Price'] so that the formula reads:

   ['Sales']=N: ['Price']*['Units']\1000;

   The restrictor N: limits the rule to the cells identified only by simple elements.
   The rule no longer applies to consolidated elements, leaving consolidations to
   function properly.

7. Click Save to save the edited rule.

8. Press F9 to recalculate the test worksheet.

   The correct value now opens at the intersection of Sales and 1Quarter.

**Calculating an Average Price**

Examine the Price, 1Quarter value in the Test worksheet. This number is the sum
of the prices for January, February, and March. However, the number should reflect
the average price for the three months. The following rules statement yields the
desired value:

['Price']=C:[‘Sales’]/[‘Units’]*1000;
The restrictor C: limits this rules statement to consolidations; that is, only when one or more of the elements that identify a Price cell are consolidated elements.

**Procedure**

1. Reopen the **Rules Editor** for the SalesCube cube.
2. Enter the following statement without the Area definition, as the third line in the entry field.
   
   `['Price']=DB('PriceCube',!actvsbud,!region,!model,!month);C:['Sales']
   ['Units']*1000;
   
   You omit the Area definition because you already defined the Price area in the second line. When you want to apply different formulas to the same Area, you define the area and then specify the formulas sequentially.
3. Click **Save** to save the edited rule.
4. In the test worksheet, press F9 to recalculate and update the values.
   
   Observe the Price, 1Quarter value, which still displays the value 76,132.59. This is because TM1 uses the first rules formula it encounters that applies to the Price, 1 Quarter cell:
   
   `['Price']=DB('PriceCube',!actvsbud,!region,!model,!month);
   
   This statement is appropriate only for N: level cells, such as Price, Jan. Consolidated Price values should be calculated through the second Price statement. By restricting the first Price formula to N: level cells only, you enable TM1 to apply the second Price formula to consolidations.

   **Note:** TM1 evaluates the rules statements in the order they appear within a rule, but the first formula for a given Area takes precedence over later formulas for the same area. If you have multiple rules statements that address the same Area, you should order them least-restrictive to most-restrictive. For details, see "Arranging Rules Statements" on page 46.
5. To restrict the first Price formula to N: level cells, open the **SalesCube** rule in the **Rules Editor**.
6. Insert N: in front of the first formula portion of the first Price statement:
   
   `['Price']=N:DB('PriceCube',!actvsbud,!region,!model,!month);
   
   The entire rule for the Price area should now appear as follows:
   
   `['Price']=N:DB('PriceCube',!actvsbud,!region,!model,!month);
   C:['Sales']\['Units']*1000;
   
   7. Click **Save** to save the rule.
8. In the Test worksheet, press F9 to recalculate and display the updated values.
   
   All Price values should now reflect the correct calculations, with the 1 Quarter, Price value as an average of the first three months of the year.

**Linking Two Asymmetrical Cubes**

Most companies do not break out overhead costs by product below the Gross Margin line. These numbers, such as rent and utilities, are available only on a regional or divisional basis. In other words, the structure of the cost numbers are not symmetrical with the sales numbers. Therefore you would normally store the data in separate cubes.

The cost data that corresponds to the sales data in the SalesCube cube is available in the PnLCube cube. The two cubes compare, as follows:

- PnLCube has four dimensions. The cost data is dimensioned by version (actual versus budget), region, account, and month.
- SalesCube has five dimensions. The sales data is dimensioned by version (actual versus budget), region, product (model), account, and month.
- The two cubes share three dimensions: Region, Actvsbud, and Month.
- Measures tracked in these cubes are identified by elements in different dimensions. SalesCube uses the Account1 dimension. PnLCube uses the Account2 dimension.
- PnLCube has no data for Sales or Variable Costs. These values are already calculated in detail in SalesCube.

In the following exercise, you write rules for the PnLCube that pulls the sales and variable costs data from the SalesCube.

**Procedure**

1. In the **Server Explorer** window, right-click the **PnLCube**, and click **Create Rule**. The **Rules Editor** opens.
2. Enter two statements to specify that the Sales and Variable Costs values in the PnLCube pull the corresponding values from the SalesCube.
   - On the first line of the entry field, create this Sales formula:
     ```
     ['Sales']=DB('SalesCube', !Actvsbud, !Region, 'Total', 'Sales', !Month);
     ```
   - On the second line, create this Variable Costs formula:
     ```
     ['Variable Costs']=DB('SalesCube', !Actvsbud, !Region, 'Total', 'Variable Costs', !Month);
     ```
3. Click **Save** to save the rules.

   **Note:** Note that the Sales and Variable Costs values now appear in the cube. The Gross Margin values are also available because you defined the Gross Margin in the account2 dimension as the difference between the Sales and VariableCosts.

4. In Microsoft Excel, open the sample worksheet **TwoCubes**.
   - The TwoCubes worksheet is set up to simultaneously pull numbers from SalesCube and PnLCube. This worksheet demonstrates how numbers can flow between the two cubes.
   - Let’s change the Units number in cell B6 to a very large number so that you can see the effect of the change ripple through the worksheet.
5. Click cell B6 and type **100000**.
6. Press F9 to recalculate the worksheet.

   **Note:** Note that the Units value for the S Series 2.5 L Sedan changes the Sales value for that model and for the model total. These changes affect the Sales values of the PnLCube, which ripples all the way down to the Earnings Before Taxes value in the PnLCube.

   **Note:** All the cell values in this worksheet are the result of references to two cubes, SalesCube or PnLCube. The worksheet does not calculate any of the values.
Creating Drill-Through Processes and Rules

You can create a drill process and drill rules to associate a cell with more detailed data. This data can provide underlying detail for the cell, or other information relevant to the cell.

TM1 drill-through consists of two components.
- **Drill Process** - Defines the detailed data you want to associate with a cell
- **Drill Assignment Rule** - Defines the relationship between the cell and the detailed data

After you create a drill process and a drill assignment rule for a cube, you can execute the process and open the detailed data in a new window, thereby "drilling through" to a new level of detail.

Creating a Drill Process

A drill process is a TurboIntegrator process that defines the detailed data, which opens in a new window. The cube from which a drill-through originates is called the **origination cube**.

Before you create a drill process, you should be familiar with the data you want to open when drilling from the origination cube.

**Procedure**

1. Right-click the origination cube in the **Server Explorer**.
2. Click **Drill**, **Create Drill Process**.
   - The first window of the Drill Process Setup Wizard opens.
   - The wizard displays a table with parameter values for the origination cube and all its dimensions. TM1 uses these parameter values to set up the drill process.
   - When you execute the drill process to drill from an origination cube to the detailed data, TM1 updates the parameter values to reflect the cube location from which the drill-through originates.
3. Click **Next**.
   - The second window of the Drill Process Setup Wizard opens.
4. Select the **Datasource Type** for the detailed data you want to drill through to from the origination cube.
   - There are three **Datasource Type** options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ODBC</td>
<td>Drills from the origination cube to an ODBC source. The ODBC source must be accessible from the computer on which the IBM Cognos TM1 server is running. <strong>Note:</strong> TM1 requires DataDirect drivers to access an Oracle ODBC source on Solaris or AIX. These drivers are not supplied with TM1 and must be acquired separately.</td>
</tr>
</tbody>
</table>
Option | Description
--- | ---
Cube View | Drills from the origination cube to a different cube view. You can drill to any cube that resides on the same server as the origination cube. You can define a cube view data source that exceeds the maximum amount of memory that TM1 can allocate when you access a view. By default, the memory threshold for the MaximumViewSize parameter in the Tm1s.cfg file is 100MB on a 32-bit system, and 500 MB on a 64-bit system. **Important:** If you do not specify the maximum view size in the configuration file, TM1 displays an error message when you attempt to drill to the cube view.

Other | Drills from the origination cube to any data source TurboIntegrator supports.

TM1 displays the data source options for the data source type you selected in the wizard.

5. Define the data source.
   - For an ODBC data source, you must supply the following information.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datasource Name</td>
<td>Name of the ODBC data source (DSN) you want to access when drilling from the origination cube.</td>
</tr>
<tr>
<td>User Name</td>
<td>Valid user name to log on to the ODBC source.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the user name.</td>
</tr>
<tr>
<td>Query</td>
<td>Query that defines the data to return from the ODBC source. Query results data displays in a separate window when you drill from the origination cube.</td>
</tr>
</tbody>
</table>

   - For a Cube View data source, you must supply the following information.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datasource Name</td>
<td>Name of the view you want to open when drilling from the origination cube. Click Browse to select a view name or create a view.</td>
</tr>
</tbody>
</table>

   - For the Other data source, click **Launch TurboIntegrator** to define the data source.

6. Click **Finish**.
   - The Save Process As dialog box opens.

7. Enter a name for the drill process in the Name box.

**Note:** For best practice, we recommend that you use a drill process name that identifies the origination cube associated with the drill process. For instance, if you create a process to drill from a cube named PriceCube to an ODBC source, you would name the drill process PriceCubeToODBCSource. This type of
naming convention makes it easier to identify a drill process name when you edit a drill process, or select from several drill processes associated with a cube.

8. Click **Save**.
   TM1 saves the drill process as a TurboIntegrator process, but prefixes the name you assigned in step 7 with the string }Drill_. For example, if you save a drill process with the name PriceCubeToODBCSource, TM1 saves the process as }Drill_PriceCubeToODBCSource.

**Editing Drill Processes**

When you create a drill process with a Cube View data source, TurboIntegrator inserts the function ReturnViewHandle('Cube','View') above or below the Generated Statements area, which is located on the **Epilog** subtab of the **Advanced** tab in the TurboIntegrator window.

If you change the data source for a drill process, TurboIntegrator does not update the function with the new data source because the function is outside the Generated Statements area. You must edit the Cube View data source in the ReturnViewHandle function for the drill process.

**Note:** For a drill process with an ODBC data source, TurboIntegrator does not insert the ReturnViewHandle function. Therefore, you do not need to edit the function when you change an ODBC data source for a drill process.

**Procedure**

1. In the **Server Explorer**, right-click the origination cube with which the drill process is associated.
2. Click **Drill**, **Edit Drill Process**.
   The Select dialog box opens.
3. Select a drill process and click **OK**.
   The TurboIntegrator window opens.
4. Click the **Advanced** tab.
5. Click the **Epilog** tab.
6. Edit the **ReturnViewHandle** function to reflect the new view.
   For example, to use the Europe_1Q view of the Sales cube as a data source, the ReturnViewHandle function would look like this:
   ReturnViewHandle('Sales','Europe_1Q')
7. Click **Save**.
8. Close the TurboIntegrator window.

**Deleting Drill Processes**

Follow these steps to delete a drill process.

**Procedure**

1. In the Server Explorer, right-click the origination cube with which the drill process is associated.
2. Click **Drill**, **Delete Drill Processes**.
   The Delete Drill Processes dialog box opens.
3. Select the process(es) you want to delete.
   - To select multiple adjacent drill processes, click and drag across the processes.
To select multiple non-adjacent processes, hold down CTRL, and click each drill process.

4. Click OK.

Creating a Drill Assignment Rule

A drill assignment rule is the TM1 rule that links cube cells with related detailed data. As indicated above, the related data can be a cube view, ODBC source, or any other data source accessible through TurboIntegrator.

Procedure

1. In the Server Explorer, select the origination cube for which you want to create a drill assignment rule.
2. Click Cube, Drill, Create Drill Assignment Rule.
   The Rules Editor opens.
   For each cube area you want to associate with detailed data, continue with steps 3 through 8.
3. Click Area to define the cube cells (area) you want to associate with detailed data.
   When you click the Area button, the Reference to Cube dialog box opens.
4. To define the area, do one of the following:
   • To define the area as the entire cube, click OK.
   • To narrow the area definition, click the dimension buttons and select the elements that define the cells you want to associate with the detailed data, and then click OK.
5. Click Equal.
6. Click String.
7. Enter the name of the drill process enclosed in single quotation marks to define the detailed data you want to associate with the area. For example, enter 'PriceCubeToSource'.
   Caution: Do not include the }Drill_ prefix in a drill process name. For example, enter 'PriceCubeToSource' for a drill process named }Drill_PriceCubeToSource.
   You can associate more than one drill process with an area. Enclose all drill processes, separated by commas, within a set of single quotation marks.
   Note: You can also use conditional logic or other functions to return the name of a drill process.
8. Click Semicolon.
   The semi-colon indicates the end of a rules statement.
9. Click Save.

Results

You can now drill through to detailed data for which you have created a drill process and a drill assignment rule.

Drill-Through Example

This section guides you through the creation of a drill process and drill assignment rule, which allows you to drill from the SalesByQuarter cube to a relational table that is the original source for the cube data. The table contains data at the monthly
level, while the SalesByQuarter cube contains data at the quarterly level. By drilling through to the relational source, you can view the underlying detail for the cube data.

Setting Up the ODBC Data Source
The example in this section drills through to an ODBC source (Access database). Before looking at the example, you must set up the ODBC data source.

Procedure
1. Open the Microsoft Windows ODBC Data Source Administrator dialog box.
   The procedure for opening this dialog box varies, depending on the version of Microsoft Windows you are running. For details, see the Microsoft Windows Help.
2. On the System DSN tab, click Add.
   The Create New Data Source dialog box opens.
3. Select Microsoft Access Driver and click Finish.
   The ODBC Access Setup dialog box opens.
4. Type TM1_sample_data in the Data Source Name box.
5. Click Select.
   The Select Database dialog box opens.
6. Navigate to your \install_dir\Custom\TM1Data\PData\RelationalData directory and select Sales.mdb.
7. Click OK to exit the Select Database dialog box.
8. Click OK to exit the ODBC Administrator dialog box.
   The Access database named Sales is now available as an ODBC source. The example drill processes use this ODBC data source.

Creating a Drill Process
You can now create a drill process for the SalesByQuarterCube cube. The drill process defines an ODBC data source as the detailed data that you can view when you drill from the SalesByQuarterCube cube.

Procedure
1. In the Server Explorer, right-click SalesByQuarterCube.
2. Click Drill, Create Drill Process.
   The Drill Process Setup Wizard opens. The table contains the default parameters for the drill process.
3. Select the Year parameter value.
4. Click Select Element.
   The Subset Editor opens.
5. Select 1 Quarter and click OK.
6. Repeat steps 3 through 5 to change the Gross Margin% parameter value to Units.
7. Click Next.
   The second screen of the wizard opens.
8. Select ODBC as the Datasource Type.
9. Click Browse next to the Data Source Name box and select TM1_sample_data.
10. Enter admin as the User Name.
11. Enter the following query in the **Query** box.

   This query is specific to the sample Access database. It uses IIF functions that are unique to Access and cannot be used against any other database.

   ```sql
   SELECT ActvsBud, Region, Model, Account1, Month, Data
   FROM Sales WHERE ( ActvsBud = '?actvsbud?' AND Region = '?region?' AND Model = '?model?' AND Account1 = '?account1?' AND
   ( IIF( '?quarter?'= '1 Quarter', Month = 'Jan' OR Month = 'Feb' OR Month = 'Mar',
     IIF( '?quarter?'= '2 Quarter', Month = 'Apr' OR Month = 'May' OR Month = 'Jun',
     IIF( '?quarter?'= '3 Quarter', Month = 'Jul' OR Month = 'Aug' OR Month = 'Sep',
     IIF( '?quarter?'= '4 Quarter', Month = 'Oct' OR Month = 'Nov' OR Month = 'Dec',
     IIF( '?quarter?'= 'Year',TRUE,TRUE))))
   )
   )
   
   **Note:** Be sure to enclose the references to the TM1 parameter names in question marks (?).

12. Click **Finish**.

   The Save Process As dialog box opens.

13. Save the process as RELATIONALTABLE_SalesByMonth.

**Results**

The new drill process displays as an available process on the server. Note that the process name includes the **Drill** prefix, indicating that it is a drill process.

You can now create a drill assignment rule to make the ODBC data source available from the SalesByQuarterCube cube.

**Creating a Drill Assignment Rule for SalesByMonth**

Follow these steps to create a drill assignment rule that makes the RELATIONALTABLE_SalesByMonth process available from the SalesByQuarterCube cube.

**Procedure**

1. In the Server Explorer, right-click **SalesByQuarterCube**.

2. Click **Drill, Create Drill Assignment Rule**.

   The Rules Editor opens.

3. Enter the following rule in the large box.

   ```sql
   ['Year'] = S:IF( ( ELLEV( 'actvsbud', !actvsbud) = 0 ) &
   ( ELLEV( 'region', !region) = 0 ) & ( ELLEV( 'model', !model) = 0 ) &
   ( ELLEV( 'account1', !account1) = 0),
   'RELATIONALTABLE_SalesByMonth', '');
   [] = S:IF( ISLEAF = 1, 'RELATIONALTABLE_SalesByMonth', '');
   
   This rule indicates that the RELATIONALTABLE_SalesByMonth drill process will be executed when you select the Drill option from a cell that is either:
   - Identified by the Year element and 0-level elements from all other dimensions
   - Identified by all 0-level elements
4. Save the rule.
   You can now test the drill-through functionality for SalesByQuarterCube cube.

**Viewing the Drill-Through Example**

Follow these steps to view the result of the drill process and drill assignment rule.

**Procedure**

1. Open the **Drill_relational** view of the SalesByQuarterCube.
2. Right-click the cell at the intersection of Units and Year.
3. Click **Drill**.
   An extract from the TM1 _sample_data data source opens, displaying the monthly detail for the cell.
4. Click **OK** to close the window.
5. Drill through other cells in the Drill_relational view to view the monthly detail.
   You should be able to drill through any cell in the view.
6. Change the Region title element from Germany to **Europe**.
7. Right-click any cell in the view.

**Results**

Note that the Drill option is not available. This is because the drill assignment rule for SalesByQuarterCube indicates that the Drill option is only available for cells identified by the 0-level elements or for the cells identified by the Year element and 0-level elements. Europe is a consolidation, so the **Drill** option is not available.

---

**Debugging Rules**

TM1 provides a tool called the Rules Tracer to assist in the development and debugging of rules. The Rules Tracer functionality is available only in the Cube Viewer.

With the Rules Tracer, you can do the following:

- **Trace calculations** - Ensure that rules are being assigned to selected cells and calculated properly, or trace the path of consolidated elements
- **Trace feeders** - Ensure that selected leaf cells are feeding other cells properly
- **Check feeders** - Ensure that the children of a selected consolidated cell are fed properly

**Tracing Calculations**

To trace a calculation, whether a consolidation or a rules calculation, right-click the cell containing the calculation and click **Trace Calculation**.

The **Rules Tracer** window opens. From here, you can trace a calculation to its leaf level components. The Rules Tracer window contains two panes.

- **Tracing Calculation (top pane)** - Displays the definition of the current cell location, with an icon indicating whether the value in the cell is derived by **Consolidation [z]** or by **Rules [x]**. Also displays the current value of the cell. If the value is derived by rules, the rule displays in the status bar of the **Tracing Calculation** pane.
- **Trace (bottom pane)** - Displays the components of the first consolidated element or the first rule in the cell definition. You can double-click any item in the bottom pane to trace a path to the leaf level elements that define the cell.
Tracing a Rule Calculation Example (Simple)
Follow these steps to view a simple example of tracing a rule calculation.

Procedure
1. Open the Trace_simple view of the SalesCube cube.
   This view contains the Price values that are derived through rules you created earlier in this section.
2. Right-click the cell at the intersection of Price and Jan.
3. Select Trace Calculation.
   The Rules Tracer window opens.
   In the Tracing Calculation pane at the top, TM1 displays the current cell definition, along with Rules icon indicating that the cell value is derived through rules. The rule that applies to the cell is shaded, and the Calculated value of the cell displays in the lower right corner of the pane.
   In the Trace pane at the bottom, TM1 shows the evaluation of the rule that applies to the current cell. In this example, the rule
   \([''Price''] = N:DB(''PriceCube'', !actvsbud, !region, !model, !month);
   C: [''Sales''][.''Units''] * 1000;
   \) evaluates to
   \([''Price''] = PriceCube(Actual, Germany, L Series 1.8L Sedan, Jan)
   because the current cell is an N: location (not defined by any consolidations).
   You now know that the cell SalesCube(Actual, Germany, L Series 1.8L Sedan, Price, Jan) takes its value from PriceCube(Actual, Germany, L Series 1.8L Sedan, Jan).

Tracing a Consolidation Example (Complex)
The previous simple example shows how the Rules Tracer works for cells at the N: level. The following example shows how you can use the Rules Tracer to trace values that include consolidations.

Procedure
1. Open the Trace_complex view of the SalesCube cube.
2. Right-click the cell at the intersection of Sales and 1 Quarter.
3. Click Trace Calculation.
   The Rules Tracer opens.
   In the Tracing Calculation pane at the top, TM1 displays the current cell definition, along with the Consolidation icon indicating that the cell value is derived through consolidation. The value of the Consolidated cell displays in the lower right corner of the pane.
   In the Trace pane at the bottom, TM1 displays the components of the first consolidated element in the cell definition. In this example, T Series is the first consolidated element in SalesCube(Actual, Germany, T Series, Sales, 1Quarter). TM1 displays the T Series components with their values.
4. Double-click T Series 2.8L Coupe in the Trace pane.
   You are now tracing the value of SalesCube(Actual, Germany, T Series, Sales, 1Quarter) through T Series 2.8L Coupe.
   The Tracing Calculation pane now displays the cell definition for the element you double-clicked.
   SalesCube(Actual, Germany, T Series 2.8L Coupe, Sales, 1 Quarter)
The Trace pane now displays the components of the first consolidated element in this cell definition. The first consolidated element in SalesCube(Actual, Germany, T Series 2.8L Coupe, Sales, 1 Quarter) is 1 Quarter. TM1 shows the three components Jan, Feb, and Mar, with their values.

In the Trace pane, note that a Rules icon precedes each component name, indicating that the components derive their values through rules.

5. Double-click Jan in the Trace pane.

You are now tracing the value of SalesCube(Actual, Germany, T Series, Sales, 1Quarter) through the T Series 2.8L Coupe and Jan.

The Tracing Calculation now displays the cell definition for the element you double-clicked. In this case, SalesCube(Actual, Germany, T Series 2.8L Coupe, Sales, Jan). This cell is calculated by a rule, ['Sales']=N: ['Price']*['Units']*1000, which is shaded. The Consolidated value of the cell, 18730.0772, displays in the lower right corner of the pane.

The Trace pane now displays the components of the rule formula and their values. The component Price has a value of 43156.86, and is derived by another rule. The component Units has a value of 434, and is an Input Value, as denoted by the gray bullet.

By plugging these values into the rule formula, you can see that (43156.86 X 434)/1000 = 18730.0772, confirming the value of SalesCube(Actual, Germany, T Series 2.8L Coupe, Sales, Jan).

6. Double-click Price to view the rule that is used to derive the Price value.

The following rule calculates the Price:

\[
\text{['Price']=N:DB('PriceCube',{!actvsbud, !region, !model, !month});}
\text{C: ['Sales']*['Units']*1000;}
\]

This rule evaluates to:

PriceCube(Actual, Germany, T Series 2.8L Coupe, Jan)

which displays in the Trace pane as an Input Value, denoted by a gray bullet, indicating that you cannot trace the calculation any further.

**Tracing a New Calculation Path**

You can click any cell definition in the Tracing Calculation pane of the Rules Tracer to begin tracing a new calculation path.

For instance, using the previous complex example, you could click the first cell definition in the Tracing Calculation pane to begin tracing a new path through the T Series consolidation.

When you click a cell definition in the Tracing Calculation pane, the Trace pane displays the components of the first consolidated element in the cell definition. You can then double-click any component to begin tracing a new calculation path.

**Tracing Feeders**

The Rules Tracer lets you trace the way a selected cell feeds other cells.

Because you can only feed other cells from a leaf element, the Trace Feeders command is available for the leaf cells you define by rules, but is not available for consolidated cells.

**Procedure**

1. In the Cube Viewer, right-click the cell you want to trace.
2. Click Trace Feeders.
The Rules Tracer window opens. This window contains two panes.

- **Tracing Feeders (top pane)** - Displays the definition of the current cell location, and the feeder rules associated with the current cell
- **Trace (bottom pane)** - Displays the locations fed by the current cell

3. Double-click a location in the **Trace** pane.
   This location becomes the current cell location in the Tracing Feeders pane, and the Trace pane displays any locations fed by the current cell.

4. Continue double-clicking the locations in the **Trace** pane until you have traced the feeders to the level you require.

**Checking Feeders**

If a cube contains a rule with SKIPCHECK and FEEDERS statements, you can use the Rules Tracer to check that TM1 properly feeds the components of the consolidation.

**Procedure**

1. In the **Cube Viewer**, right-click the consolidated cell you want to check.
2. Click **Check Feeders**.
   - The Rules Tracer opens. This window contains two panes.
     - **Checking Feeders (top pane)** - Displays the definition of the current cell (consolidation)
     - **Trace (bottom pane)** - Displays all components of the consolidation that are not properly fed
   - The Trace pane is empty, which means the consolidation is fed properly and the cubes values are accurate.
   - If the Trace pane displays the components of the consolidation, you must edit the rule associated with the current cube to add the FEEDERS statements that feed all the listed components.

**Note:** You can only check the FEEDERS for a cell once per TM1 session. The action of checking the FEEDERS actually feeds the components of the consolidation. Any subsequent checking of the FEEDERS does not yield accurate results. If you want to check the FEEDERS for a cell more than once, you must recycle the IBM Cognos TM1 server before every check.
Chapter 4. Organizing Objects in TM1 Applications

IBM Cognos TM1 lets you organize objects logically by application, and by type. This section describes how to create TM1 applications.

TM1 Application Overview

TM1 applications are objects that function as virtual folders to organize shortcuts to other TM1 objects, files, and URLs in a logical, job-specific grouping.

For example, you might create an application that organizes all TM1 objects and related files for a North American sales organization.

You insert objects, files, and URLs into an application by creating a shortcut or reference. Applications and references provide a quick and organized way to open the target objects to which the references point.

Types of References

IBM Cognos TM1 applications can contain references to any of the following items:

- **TM1 objects** - An application can contain references to any type of TM1 object except for rules and replications.
  
  You can reference TM1 objects on the same IBM Cognos TM1 server containing the application or any other TM1 server to which you have access.

- **Files** - An application can reference any type of file, such as an Excel spreadsheet, a Word document, or any other file. Any Excel file can be referenced, regardless of whether the file contains TM1 slices, functions, or other TM1 features.
  
  You can reference both external and uploaded files:
  
  A reference to an external file provides a shortcut to a file that can exist in any shared directory on your network.
  
  A reference to an uploaded file saves a copy of the original file on the TM1 server. However, changes to the original source file, outside of TM1, are not automatically reflected in the uploaded copy of the file on the TM1 server. You must update the file on the server to make the changes available.

- **URLs** - An application can contain links to any web pages or resources that use the http or https protocol.
Behavior of References

It is important to understand that when you add object and file references to an application, they exist by reference only - as a shortcut that opens the original IBM Cognos TM1 object or file. Deleting a reference in an application is different from deleting the source TM1 object or file:

- Deleting a reference from an application has no impact on the corresponding source TM1 object or file. Only the reference is deleted.
- Deleting a source object in TM1, or a file on disk, breaks any corresponding reference in an application, but does not delete the reference. The reference remains but is not functional if the source object or file is deleted.

One exception to the above is for uploaded file references. In this case, the file is actually copied to the IBM Cognos TM1 server. If you delete a reference to an uploaded file, TM1 deletes the uploaded copy of the file from the TM1 server.

For details on file references, see “Adding File References to an Application” on page 79.

Display Order for References in Applications

TM1 displays references in applications in the following order. Within each reference group, TM1 sorts the references alphabetically in descending order.

- Cubes
- Views
- Dimensions
- Subsets
- Processes
- Chores
- Files and URLs

Note: You cannot set the order in which reference groups display, nor the sort order within reference groups.

Using Applications and References in TM1 Web

When you create TM1 applications and references in the Server Explorer, they are automatically available in TM1 Web. This applies for references to cubes, views, files, and URLs.

For details, see “Publishing TM1 Applications to TM1 Web” on page 89.

Application Folders and Files on the TM1 server

The folders and files that support TM1 applications are stored in the IBM Cognos TM1 server data directory.

Folders

The structure for TM1 application folders are organized in the following location:

\TM1 Data Directory\applications

Information about both referenced and uploaded files are stored in their related sub-folders here.
**Referenced Files**

Information about referenced files are stored in placeholder files named with the .extr extension.

For example, if you create an application reference to a file named sheet1.xls, TM1 creates a placeholder file named sheet1.xls.extr to keep track of that entry.

**Uploaded Files**

Information about uploaded files are stored in placeholder files named with the .blob extension.

The actual files that are uploaded to the TM1 server are stored in the following location:

`TM1 Data Directory \ }Externals`

For example, if you upload a file to the Planning Sample database, it would be saved here:

C:\Program Files\Cognos\TM1\Custom\TM1Data\PlanSamp\}Externals\`

TM1 automatically appends a date and time suffix to the end of the file name for any file you upload. The suffix uses the format `year month day time.file extension`.

For example, if you upload the file Sample_Budget.xls to the Planning Sample database, TM1 renames it to `Sample_Budget.xls_20090617155650.xls` and stores the file in the following location:

C:\Program Files\Cognos\TM1\Custom\TM1Data\PlanSamp\}Externals\Sample_Budget.xls_20090617155650.xls

---

**Security Considerations for Creating and Viewing Applications**

The following table describes the security privileges required to perform actions related to TM1 applications.

For details, see "Administering Security for TM1 Applications" on page 86.

<table>
<thead>
<tr>
<th>Action</th>
<th>Required security privilege</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a top-level application</td>
<td>Must be member of ADMIN or DataAdmin group</td>
</tr>
<tr>
<td>Create a secondary-level application</td>
<td>Admin access to parent application</td>
</tr>
<tr>
<td>View and use applications and references</td>
<td>Read</td>
</tr>
<tr>
<td>Add a private reference to an application</td>
<td>Read</td>
</tr>
<tr>
<td>Publish a private application</td>
<td>Must be member of ADMIN or DataAdmin group</td>
</tr>
<tr>
<td>Publish a private reference to a public object</td>
<td>Admin</td>
</tr>
</tbody>
</table>
Creating and Managing Applications

To begin adding applications to a IBM Cognos TM1 server, a member of the ADMIN or DataAdmin group must first create one or more top-level applications. Top-level applications are applications that appear directly below the Applications group in the Server Explorer. Secondary-level applications appear within a parent application, as explained in "Creating a Secondary-Level Application within an Existing Application."

In the following example, you see an applications group (Applications), three top-level applications (European Sales, North American Sales, and PacRim Sales), and two secondary-level applications (China and Indonesia) beneath the PacRim Sales top-level application.

![Applications Tree](image)

Creating a Top-Level Application

Follow these steps to create a top-level application.

**Procedure**

1. Ensure that the Applications group is visible on your IBM Cognos TM1 server. If the group is not visible, click **View, Applications** in the Server Explorer.
2. In the Server Explorer, right-click the Applications group on the server where you want to create the application and click **Create New Application.** TM1 inserts a new top-level application temporarily named New Folder in the selected Applications group.
3. Assign a name to the new application.
   
   Note that Microsoft Windows has some reserved device names that cannot be used as folder or file names. See the MS Windows website for an updated list of reserved device names.
   
   All applications are private objects that only the user who creates the application can access. You must publish an application to make it available to

---

<table>
<thead>
<tr>
<th>Action</th>
<th>Required security privilege</th>
</tr>
</thead>
<tbody>
<tr>
<td>Publish a private reference to a private object</td>
<td>Cannot be done</td>
</tr>
<tr>
<td>Privatize a public application or reference</td>
<td>Admin</td>
</tr>
<tr>
<td>Delete a public application or reference</td>
<td>Admin</td>
</tr>
<tr>
<td>Delete a private application</td>
<td>Admin</td>
</tr>
<tr>
<td>Delete a private reference</td>
<td>Read</td>
</tr>
</tbody>
</table>

---
other TM1 users. Only TM1 Architect and TM1 Perspectives users can create public applications. TM1 Client users can create only private applications.

4. Right-click the application and click **Security, Make Public** to make the application available to other users.

**Note:** The type of icon that is used for an application depends on the private or public status of the application.

- A *private* application displays with a **Private Applications** icon 🗝 which includes a key in the upper right corner.
- A *public* application displays with a **Public Applications** icon 🗝.

You can now begin adding references and/or secondary-level applications to the application.

**Renaming Applications**

Follow these steps to rename an application.

**Procedure**

1. Right-click the application in the Server Explorer.
2. Click **Rename**.
   
   The current application name is selected and ready to be edited.
3. Type a new name for the application.
4. Press **ENTER**.

**Deleting Applications**

You can delete any application, public or private, to which you have ADMIN privilege.

**Note:** When you delete an application, IBM Cognos TM1 deletes all sub-applications and references contained in the application.

When you have ADMIN privilege to an application, you must also have ADMIN privilege to all references and sub-applications within the application, regardless of the security privileges that TM1 applies through the TM1 Security Assignments window. Therefore, when you haveADMIN access to an application, you can delete sub-applications and references to which you were assigned only the READ or NONE privilege.

To illustrate the ADMIN privilege for deleting an application, consider the following example. There are three applications on a server (App1, App2, and App3), all created by a member of the ADMIN group.

The administrator has defined the following security privileges to the applications for the North America user group:

<table>
<thead>
<tr>
<th>Application Name</th>
<th>Security Privilege for North America User Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>App1</td>
<td>Admin</td>
</tr>
<tr>
<td>App2</td>
<td>Read</td>
</tr>
</tbody>
</table>
When a member of the North America user group logs on to the IBM Cognos TM1 server, he will see App1, to which he has ADMIN privilege, and App2, to which he has READ privilege. He will not see App3, as he has NONE privilege for that application.

Now, if a member of the North America user group attempts to delete App1, the deletion will succeed. This is expected because the North America user group has ADMIN privilege to App1, which allows deletion of applications. However, both App2 and App3 (along with any references contained therein) will also be deleted without warning, despite the fact that the North America group has been assigned READ and NONE privileges for the applications, respectively.

**Procedure**

1. Right-click the application in the Server Explorer.
2. Click Delete.
3. Click Yes when prompted for confirmation.

**Creating a Secondary-Level Application within an Existing Application**

Any IBM Cognos TM1 user with Admin privilege to an existing application can create secondary-level applications.

**Procedure**

1. Right-click the existing application.
2. Click New, Application.
   TM1 inserts a new secondary-level application temporarily named New Folder in the existing application.
3. Assign a name to the new application.
   TM1 creates the application as a private object which only you can access.
4. If you are a member of the ADMIN or DataAdmin group and you want to publish the application so that other TM1 users can access it, right-click the application and click Security, Make Public.

When you publish a secondary-level application, security privileges for the secondary-level application are inherited from the security privileges defined for the parent application. You can change security privileges by following the steps in "Assigning Security Privileges for TM1 Applications and References to User Groups" on page 86.

**Adding TM1 Object References to an Application**

You can add references to objects from the IBM Cognos TM1 server on which the application resides as well as objects from other servers to which you are connected.
Procedure

1. Select the object in the Server Explorer.
2. Drag and drop the object onto the application.
   A reference to the object displays in the application.

By default, any reference you add to an application is a private reference, as indicated by a key superimposed over the object icon. Only you can access the reference.

<table>
<thead>
<tr>
<th>Icon Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Image of North American Sales cube]</td>
<td>A reference to the SalesCube cube that was added to the North American Sales application.</td>
</tr>
<tr>
<td>[Image of key]</td>
<td>TM1 adds an image of a shortcut arrow to an object's icon to represent a reference.</td>
</tr>
<tr>
<td>[Image of key]</td>
<td>An image of a key is added to the lower corner of the reference icon to represent private references and to both lower and upper right-hand corners to represent private objects.</td>
</tr>
</tbody>
</table>

Making a Reference Public

If you want to make the reference available to other IBM Cognos TM1 users, you must publish the reference by right-clicking the reference and clicking **Application Item, Security, Make Public**.

TM1 combines the object's icon with a shortcut arrow to identify a public reference to a public object.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Image of public subset]</td>
<td>Public reference to a public subset.</td>
</tr>
<tr>
<td>[Image of public view]</td>
<td>Public reference to a public view.</td>
</tr>
</tbody>
</table>

To simplify the process of creating a public reference, you can right-click an object in the Server Explorer, hold down the right mouse button, and then drag and drop the object onto an application. When you drop the object, TM1 displays a shortcut menu. Click **Create Public Reference**.

You can publish references in public applications only. When you publish a reference in an application, security privileges for the references are inherited from the security privileges defined for the parent application. You can change security privileges by following the steps outlined in “Assigning Security Privileges for TM1 Applications and References to User Groups” on page 86.
Adding References to Private Views and Subsets to an Application

When you add a reference to a private view or subset to an application, the reference is created as a private reference, which is the default behavior when creating any reference in an application.

IBM Cognos TM1 adds an image of two keys to a reference icon to identify a private reference to a private object.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Icon]</td>
<td>Private reference to a private subset.</td>
</tr>
<tr>
<td>![Icon]</td>
<td>Private reference to a private view.</td>
</tr>
</tbody>
</table>

These icons help you differentiate:
- private references to private objects, from private references to public objects.

TM1 adds a single key to a reference icon to identify a private reference to a public object.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Icon]</td>
<td>Private reference to a public subset.</td>
</tr>
<tr>
<td>![Icon]</td>
<td>Private reference to a public view.</td>
</tr>
</tbody>
</table>

If you publish a private view or subset for which a reference exists, the reference is no longer valid, and displays an error message when accessed.

For example, if create a reference to a private view called View1, and then make View1 public, the reference breaks and can not open the view. The following error message displays: Cannot find private view 'View1' of cube 'plan_BudgetPlan' on server 'planning sample'.

In such a circumstance, you should delete the old reference and insert a new reference to the (now) public view or subset.

Simultaneously Creating a New Object and Adding a Reference to an Application

You can create a new IBM Cognos TM1 object from within an application. TM1 creates the new object on the server where the application resides, and inserts a
private reference to the object in the application. You can create dimensions, cubes, processes, and chores from within an application.

**Procedure**

1. In the Server Explorer, right-click the Application from which you want to create the object, and click **New <Object type>** on the shortcut menu. For example, to create a new cube, click **New, Cube**.

   When you select an object type, a dialog box or window opens. For example, if you click **New, Cube**, the Creating Cube dialog box opens.

2. Complete the procedure required to create the type of object you chose.

   When you are done, TM1 creates the object on the server and inserts a private reference to the object in the application.

   The example shows the result of creating the cube New Cube from within the North American Sales application.

---

**Adding File References to an Application**

You can add file references to IBM Cognos TM1 applications for any type of file on your computer or network, such as Excel, Word, PowerPoint, PDF, or any other file.

**Notes:**

- You can add a reference to any Excel file to an application. You are not restricted to working only with Excel files containing TM1 slices or other TM1 features.
- Files are opened with the program with which they are associated, as configured in the Microsoft Windows file type settings. TM1 displays an icon for each file based on this association.

**Procedure**

1. In the Server Explorer, right-click the Application to which you want to add the file reference, and click **Add File** on the shortcut menu.

   The Add File dialog box opens.

2. Browse to the directory containing the file and select the file you want to add to the application.

3. Select an option that determines how TM1 will add the file to the application.

   **Attach the file as a reference** - Keeps the file in its current location and inserts a reference to the file in the TM1 application.

   When a file is added as a reference, it displays as an icon with an arrow in TM1. For example, the icon for an Excel file that is added using the Attach the file as a reference option looks like this ![Excel Icon](image) in the TM1 application object tree.

   The primary advantage of this option is that any edits or modifications to the Excel file are immediately available in the application. If you select this option,
the Excel file should reside in a shared folder and you should navigate to the file through your network to create a UNC path to the file.

To maintain access to an attached Excel file whenever the IBM Cognos TM1 server is running, the computer where the file resides must be available on the network.

**Copy the file to the TM1 server** - Copies the file to the TM1 server on which the application resides.

This type of reference displays as an icon without an arrow in TM1. For example, an Excel file that is added using the Copy the file to the TM1 server option has the following icon in TM1 applications:

1. TM1 copies the uploaded files to the |Externals subdirectory of the TM1 server data directory. For example, if you add an Excel file to an application on the sdata server that is part of the sample TM1 database, TM1 saves the file to C:\Program Files\Cognos\TM1\Custom\TM1Data\sdata\|Externals.

   The primary advantage to this option is that the file is available whenever the TM1 server is running. However, changes to the original source file are not automatically reflected in the file on the TM1 server. You must update the file on the server to make the changes available.

   - For details on updating Excel files that have been copied to TM1, see “Updating Excel Files on the TM1 server” on page 81.
   - For details on updating non-Excel files that have been copied to TM1, see “Updating Non-Excel Files on the TM1 server” on page 82.

**Note:** If you select the Copy the file to the TM1 server option, the file you want to upload cannot be currently in use by another program. If you attempt to upload an open file, TM1 displays an error message. Make sure the file is closed before uploading it to TM1.

4. Select either the **Public** or **Private** option.

   - **Public** - Makes the file reference available to other users who have access to the TM1 application.
   - **Private** - Only you can access the reference to the file.

5. Click **Open**.

   The file is now available from within the TM1 application.

   You can also access files in TM1 applications from TM1 Web. For details, see the following sections later in this section:

   - “Publishing TM1 Applications and References” on page 87.
   - “Publishing TM1 Applications to TM1 Web” on page 89.

---

### Understanding Public and Private File References

IBM Cognos TM1 uses the following icon formats to visually identify public and private file references in TM1 applications.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image.png" alt="Icon" /></td>
<td>Public reference to a file that was added to TM1 as a reference.</td>
</tr>
<tr>
<td><img src="image.png" alt="Icon" /></td>
<td>Private reference to a file that was added to TM1 as a reference.</td>
</tr>
</tbody>
</table>
Behavior of Files Uploaded to the TM1 server

An uploaded file is any file that is added to an IBM Cognos TM1 application using the Copy the file to the TM1 server option.

Uploaded files behave differently than referenced files because uploaded files are actually copied to and stored within the IBM Cognos TM1 server.

- TM1 copies and saves uploaded files to the TM1 server in the following directory: `<server_data_dir>\Externals directory.`
- When a file is uploaded to the TM1 server, the file name is appended with a time/date stamp. For example, if you upload the file US Budget.xls to the TM1 server, the file is saved as US Budget.xls_20040702193054.xls.
- When you delete an uploaded file from a TM1 application, TM1 deletes the copy of the uploaded file from the }Externals directory. The original file, outside of TM1, that the uploaded file was copied from, is not deleted.
- If you want to create a hyperlink that opens an uploaded file from a Websheet or other TM1 application file in TM1 Web, the hyperlink must include the TM1 assigned name for the uploaded file. For details, see “Creating Hyperlinks to Uploaded Files” on page 90.

Updating Excel Files on the TM1 server

IBM Cognos TM1 provides a dedicated process to just update Excel files that have been uploaded to the IBM Cognos TM1 server. For details on updating non-Excel files (Word, PowerPoint, or other file types) see “Updating Non-Excel Files on the TM1 server” on page 82.

Procedure

1. Double-click the file in the Server Explorer.
   The file opens in Excel with a temporary file name such as TM12C5D.xls displayed in the title bar.

   Note: It is important to remember the original name of the file you are updating (as opposed to the temporary file name). You will need to select the original file later in step 4.

2. Apply your edits to the Excel document.

3. Click TM1, Save Workbook on TM1 Server, Update Existing Application File on TM1 Server.
   The Select TM1 External File to Update dialog box opens.

4. Select the original Excel file you want to update.
Be sure to select the original Excel file you opened in Step 1. If you select any other file, TM1 will overwrite that selected file without warning.

5. Click OK.

TM1 updates the file on the TM1 server. The Excel file is available from within its parent application.

**Updating Non-Excel Files on the TM1 server**

You can update a non-Excel file that has been uploaded to the IBM Cognos TM1 server by saving the file as an external file and then manually re-adding the file to the server.

**Procedure**

1. Double-click the file in the Server Explorer.

   The file opens in the associated program with a temporary file name such as TM163.doc displayed in the title bar.

   For example, an uploaded Word file opens in Microsoft Word.

2. Using the program associated with the file, make your changes to the file and then use the program's **Save As** feature to save the file to a new location and filename.

   **Note:** Remember the name and location of the new updated file so you can add it back into TM1 in the next step.

3. In Server Explorer, right-click on the old version of the file and click **Delete**.

   TM1 displays the **Confirm Delete** dialog so you can delete the old file.

4. Re-add the updated file using the steps described in "**Adding File References to an Application**" on page 79.

**Adding URL References to an Application**

You can add a URL address to an application for any of the following URL protocols:

- http://
- https://

When you open a URL reference in an IBM Cognos TM1 application from within TM1 Server Explorer, or TM1 Web, the target source of the URL displays in your system's default web browser.

**Procedure**

1. In the Server Explorer, right-click the Application to which you want to add the URL reference, and click **Add URL** on the shortcut menu.

   The Add URL dialog opens.

2. Enter a complete URL, including the http:// or https:// protocol.

   For example: http://www.Company.com

3. Enter a descriptive name for the URL.

   For example: Company Web Site

4. Click **OK** to add the URL.

   The URL is added to the TM1 application, using your system's default icon for a URL link.
You can double-click a reference to an object, file, or URL in an IBM Cognos TM1 application to perform the default action on the object.

You can right-click an object reference, and select any supported action for the object from the shortcut menu.

The following table describes the default action for all objects, files, and URLs that you can access from TM1 applications.

<table>
<thead>
<tr>
<th>Reference Type</th>
<th>Default Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cube</td>
<td>Opens the default view of the cube in the Cube Viewer.</td>
</tr>
<tr>
<td>Cube view</td>
<td>Opens the view in the Cube Viewer.</td>
</tr>
<tr>
<td>Dimension</td>
<td>Opens the default subset of the dimension in the Subset Editor. If a default subset is not defined, the All subset is opened.</td>
</tr>
<tr>
<td>Subset</td>
<td>Opens the subset in the Subset Editor.</td>
</tr>
<tr>
<td>Process</td>
<td>Opens the process for editing in the TurboIntegrator window.</td>
</tr>
<tr>
<td>Chore</td>
<td>Opens the chore in the Chore Setup Wizard. (You cannot open an active chore.)</td>
</tr>
<tr>
<td>File</td>
<td>Opens the file in the program with which it is associated, as configured in the Microsoft Windows file type settings. For example, an .xls file opens in Excel.</td>
</tr>
<tr>
<td>URL</td>
<td>Opens the URL in your system’s default web browser.</td>
</tr>
</tbody>
</table>

Note: When accessing TM1 objects that are located on another IBM Cognos TM1 server, the server must be running and you must be logged in to it.

- If you try to access a reference to an object that resides on a server that is running, but to which you are not currently connected, TM1 prompts you to log in to the server.
- If you try to access a reference to an object that resides on a server that is not running, TM1 issues the following warning: The server on which this object resides is not responding. Refresh Server Explorer display?

To restore access to the reference, start the server on which the source object resides.

**Viewing TM1 Application Properties**

Follow these steps to view the properties of references and sub-applications in an IBM Cognos TM1 application.

**Procedure**

1. Select the application in the Server Explorer.
2. If the Properties pane is not visible in the Server Explorer, click Display, Properties Window.

The Properties pane displays information about all references and sub-applications to which you have at least Read access. References and sub-applications to which you have None access do not appear in the Server Explorer, so you cannot view the properties of these items.

Only immediate sub-applications of the selected TM1 application appear in the Properties pane.

3. To sort items in the Properties pane alphabetically by property value, click the column label to which you want to apply the sort. For example, to sort items alphabetically by current status, click the Status column label.

For each reference and sub-application, the following properties display.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the reference or sub-application as it appears in the selected application. You can edit reference names, which do not have to directly correspond to source object names. For example, in the figure above, the Price reference refers to the source cube named PriceCube.</td>
</tr>
<tr>
<td>System Name</td>
<td>• For most references, System Name is the actual name of the source object to which a reference points.</td>
</tr>
<tr>
<td></td>
<td>• For files that have been uploaded to the IBM Cognos TM1 server, System Name is the name assigned to the file on the TM1 server.</td>
</tr>
<tr>
<td></td>
<td>• Naming conventions for files uploaded to the TM1 server are described in “Behavior of Files Uploaded to the TM1 server” on page 81.</td>
</tr>
<tr>
<td></td>
<td>• For files that are attached as a reference, System Name is the UNC path to the file.</td>
</tr>
<tr>
<td></td>
<td>• The System Name property does not apply to sub-applications.</td>
</tr>
<tr>
<td>Server</td>
<td>The TM1 server on which the source object for a reference resides. For example, in the above image the source cube for the Currency reference resides on the inventory server.</td>
</tr>
<tr>
<td>Private</td>
<td>This property applies only to subsets and views. The Private property indicates whether the source for a subset or view reference is a private object. A property value of Yes indicates that the source is a private object. No indicates a public object. For example, in the image above, the sources for the North American Currencies subset reference, as well as the North American Price Plan and North American Models views, are all private objects.</td>
</tr>
</tbody>
</table>
### Property Description

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| Status   | This property indicates the current availability of references and sub-applications. There are three possible Status values:  
   - **Available** - The reference or sub-application is available for use.  
   - **Not Connected** - You are not connected to the TM1 server on which the source object for the reference resides. Log in to the server to restore your access to the reference.  
   - **Not Available** - The TM1 server on which the source object for the reference resides is not running, and therefore you cannot access the reference. |

| Security | This property indicates your security privilege for a reference or sub-application. |

### Deleting Object, File, and URL References from TM1 Applications

Follow these steps to delete an object reference from an IBM Cognos TM1 application.

**Procedure**

1. Right-click the object reference in the application.

   **Note:** You must select the object reference in the application. If you select the source object elsewhere in the server hierarchy, you cannot delete the object from the application.

2. Click **Application Item, Delete**.

   To delete a file or URL reference from an application:
   - Right-click the file in the application.
   - Click **Delete**.

### Renaming Object, File, and URL References in TM1 Applications

A reference to an object or file in a IBM Cognos TM1 application does not have to use the name of the source file with which it is associated. You can rename an object or file reference in an application and maintain the connection to the source file.

**Procedure**

1. Right-click the reference in the application.

2. Select the **Rename** option as follows:
   - Click **Application Item, Rename** to rename an object reference.
   - Click **Rename** to rename a file or URL reference.

   The reference name is selected and ready to be edited.

3. Type a new name for the reference.

4. Press Enter.
Deleting Source Objects that are Referenced by TM1 Applications

When you delete a source object that is referenced by an IBM Cognos TM1 application, TM1 does not delete the corresponding object reference from the application.

For example, if you delete the Canada Sales view from the data server, the reference to the Canada Sales view remains in the North American Sales application.

If you attempt to open an object or file reference in an application, and the source for the object or file has been deleted from the IBM Cognos TM1 server, TM1 displays an error message, stating that the object ‘US Sales’ on server ‘sdata’ cannot be found and prompts you to delete the reference.

When the source for an object or file in an application has been deleted from the server, you should delete the corresponding reference from the application. For details, see “Deleting Object, File, and URL References from TM1 Applications” on page 85.

Administering Security for TM1 Applications

The following sections describe how to assign security privileges for IBM Cognos TM1 applications and references to user groups on the IBM Cognos TM1 server, as well as how to publish and privatize TM1 applications and references.

Assigning Security Privileges for TM1 Applications and References to User Groups

You can assign security privileges for public items (either references or sub-applications) within public IBM Cognos TM1 applications to user groups on the IBM Cognos TM1 server. You must have Admin privilege to an application to assign security to items within the application.

You cannot assign security for private applications and references; only the user who creates a private item can access the item.

Procedure

1. In the Server Explorer, right-click the TM1 application that contains the items to which you want to assign security.
2. Click Security, Security Assignments.
   The Security Assignments window opens. The window lists all public items (TM1 objects, files, URLs, and sub-applications) that reside in the current application.
3. Select the cell at the intersection of the item for which you want to define security and the user group to which you want to assign security.
4. Click one of the available security privileges.

<table>
<thead>
<tr>
<th>Security Privilege</th>
<th>As applied to TM1 applications</th>
<th>As applied to references</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Members of the user group cannot see the application or its contents.</td>
<td>Members of the user group cannot see the reference.</td>
</tr>
<tr>
<td>Security Privilege</td>
<td>As applied to TM1 applications</td>
<td>As applied to references</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Read</td>
<td>Members of the user group can see the application and use any references within the application to which the group has at least Read privilege. Members can also create private references in the application.</td>
<td>Members of the user group can use the reference.</td>
</tr>
<tr>
<td>Admin</td>
<td>Members of the user group can see the application, use references within the application, and create both public and private references in the application. They can also create private sub-applications. Members with Admin privilege to an application can set security privileges for all references and sub-applications within the application.</td>
<td>Members of the user group can use the reference. They can also update or delete the reference. They can publish private references, and privatize public references.</td>
</tr>
</tbody>
</table>

5. Repeat steps 3 and 4 for any other items for which you want to define security.
6. Click OK.

**Results**

For any given user group, it is possible to assign the READ or ADMIN privilege to a reference when the privilege assigned to the object associated with the reference is NONE. In this scenario, members of the user group will not be able to see the reference in an application.

For example, if a user group is assigned NONE privilege for a cube, but READ privilege for a reference to the same cube, members of the user group will not see the reference to the cube in an application.

**Publishing TM1 Applications and References**

The following sections describe how to publish IBM Cognos TM1 applications and references. To determine the security privileges required to perform these procedures, see “Security Considerations for Creating and Viewing Applications”.

**Publishing Private TM1 applications**
Follow these steps to publish a private application.

**Procedure**
1. Right-click the application in the Server Explorer.
2. Click Security, Make Public.

   **Note:** When you publish a private application, IBM Cognos TM1 also publishes all private references to public objects within the application.

**Publishing Private References to Public Objects**
You can publish private references that reside in public applications
Privatizing TM1 Applications and References

You can privatize public IBM Cognos TM1 applications and references. When you make an application or reference private, only you can access the item.

Privatizing a Public TM1 Application

Follow these steps to privatize a public application.

Procedure
1. Right-click the application in the Server Explorer.

When you privatize a public application, all public references within the application are automatically privatized as well.

If a public application contains identically named references to a single object type, one public and one private, the string _Public is appended to the public reference when the application is privatized.

For example, the following image shows the European Sales application, which contains two references to views named Northern Europe Sales, one public and one private.

When you privatize the European Sales application, the public reference is converted to a private reference and its name is changed to Northern Europe Sales_Public, indicating that this is a private reference to a public object.

Privatizing a Public Reference

Follow these steps to privatize a public reference.

Procedure
1. Right-click the reference in the Server Explorer.
2. Select Application Item, Security, Make Private.
Viewing Logical Groupings in TM1 Applications

One of the primary advantages of IBM Cognos TM1 applications is that they let you view and manage objects and files in logical groupings. This simplifies the task of using TM1, because you can easily identify and locate files and objects in job-specific applications without having to scan through large lists of objects arranged by type.

To further simplify the use of TM1, you can suppress the display of objects by type. This yields a cleaner, more easily navigated display in the Server Explorer. (By default, all object types are displayed in the Server Explorer.)

To suppress the display of a particular type of object, click View, <Object type> from the Server Explorer. This clears the check mark next to the object type in the View menu and suppresses the display of the object type in the Server Explorer. In the following sample, all objects are suppressed except for Applications.

Note: When you suppress the display of a given object type in the Server Explorer, references to objects of that type still appear within TM1 applications. However, control objects are suppressed within applications when the display of control objects is suppressed in the Server Explorer.

Publishing TM1 Applications to TM1 Web

All references to cubes, views, files, and URLs in IBM Cognos TM1 applications are automatically available in TM1 Web.

As of IBM Cognos TM1 version 10.2, to view a Microsoft Excel worksheet as a Websheet in Cognos TM1 Web, the file must be in the .xlsx or .xlsm format for Microsoft Excel 2007 or later.

Assume North American Sales is a public application that contains references to a collection of TM1 objects and Excel files. When you access the sdata IBM Cognos TM1 server through TM1 Web, these references are displayed under Applications.

When working in TM1 Web, you can click on a reference to open and display it as follows:
• Cubes and views display directly in TM1 Web.
• Excel files display as TM1 Websheets directly in TM1 Web.

Note: Excel files that have been protected through the Excel command, Tools, Protection, cannot be accessed through TM1 Web.
• Non-Excel files open and display in their associated program.

Note: Some file types and programs may not be viewable from TM1 Web.
• URL references open and display in a separate web browser.

For example, click on an Excel file to open it in TM1 Websheet format.

Note: TM1 determines the column widths of the Websheet based on the Excel file from which the Websheet is generated. If the columns in the Excel file do not accommodate the full display of row and column labels, the corresponding labels in the Websheet are truncated.
You can access both public and private applications and references through TM1 Web. Only the user who creates a private application or reference can access the item through TM1 Web. Access to public applications and references in TM1 Web is determined by the security privileges defined for the source applications and references on the TM1 server. For details on setting security privileges, see “Assigning Security Privileges for TM1 Applications and References to User Groups” on page 86.

Setting TM1 Websheet Properties

Websheet properties are configured in Server Explorer to control how an Excel file appears and behaves when viewed as a Websheet in IBM Cognos TM1 Web.

Procedure

1. In the Server Explorer, open the TM1 application containing the Excel file from which the TM1 Websheet is generated.
2. Right-click the Excel file.
3. Click Properties.
4. Use the options on the General and Display Properties tabs to set properties for the TM1 Websheet.
   For details on the TM1 Web Properties options, see the IBM Cognos TM1 User Guide.
5. Click OK.

Creating Hyperlinks to Uploaded Files

If you want a Websheet to contain a hyperlink to an uploaded file, the hyperlink must include the location and name that IBM Cognos TM1 assigns to the uploaded file.

When you add an uploaded file to a TM1 application, a copy of the file is saved on the IBM Cognos TM1 server and the file name is appended with a data and time stamp. For example:

Report_2006.xls_20070123212746.xls

If you do not include the TM1 assigned file name in the hyperlink, the link does not work in TM1 Web and an error displays, for example,

"File does not exist: TM1://planning sample/blob/PUBLIC/\}Externals\upload_test_2.xls".

Procedure

1. In Server Explorer, use the Properties pane to find the System Name, which is TM1 assigned name for the uploaded Excel file that will be the target of the hyperlink.
2. Create the hyperlink to the uploaded Excel file using the following format:

   TM1://ServerName/blob/PUBLIC/\}Externals\Filename

   where:
   - ServerName is the TM1 sever name where the Excel file is located.
   - Filename is the name that TM1 assigned to the uploaded Excel file.

For example:
3. In Excel, add the hyperlink to the worksheet where you want the link to exist.
4. Add the worksheet to a TM1 application and then view the file as a Websheet in TM1 Web.

**Viewing Websheets that Contain the 0x1A Hexadecimal Character**

IBM Cognos TM1 Web cannot open a Websheet that contains the 0x1A hexadecimal character. If you attempt to open a Websheet containing the 0x1A hexadecimal character, TM1 Web issues the following error:

Error occurred while converting the MS Excel workbook into XML format : ", hexadecimal value 0x1A, is an invalid character. Line 54, position 34.

If you remove the 0x1A hexadecimal character from the Websheet, the file will open in TM1 Web.

**Note:** The ASCIIOutput TurboIntegrator function places the 0x1A hexadecimal character at the end of all generated files. If you use ASCIIOutput to export TM1 data to an ASCII file and then attempt to open the file in the TM1 Websheet, you will encounter this error.
Chapter 5. Importing Data with Processing Worksheets

This section describes how to import data into an IBM Cognos TM1 cube using a processing worksheet. A processing worksheet is a modified Excel worksheet in which you use TM1 functions to send values to a location in an existing cube.

Note: Processing worksheets, while a valid means of importing data, are deprecated functionality in the current TM1 release. We strongly recommend using TurboIntegrator to import data into TM1 cubes.

The following topics are described in this section.
- Processing Worksheets Overview
- Importing Data Using Processing Worksheets

Processing Worksheets Overview

A processing worksheet is a modified Excel worksheet in which you use functions to send input values to a location in an existing TM1 cube.

You can use processing worksheets to convert input values that do not map directly to existing elements. For example, in the following table, the first two columns contain codes that do not directly correspond to elements in the cube to which values are being sent.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Region</th>
<th>Model</th>
<th>Measure</th>
<th>Jan</th>
<th>Feb</th>
<th>Mar</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>R54</td>
<td>S Series 1.8 L ...</td>
<td>Price</td>
<td>25259.93</td>
<td>25830.76</td>
<td>25041.90</td>
</tr>
<tr>
<td>002</td>
<td>R54</td>
<td>S Series 1.8 L ...</td>
<td>Price</td>
<td>25259.93</td>
<td>25830.76</td>
<td>25041.90</td>
</tr>
<tr>
<td>001</td>
<td>R32</td>
<td>S Series 1.8 L ...</td>
<td>Price</td>
<td>25259.93</td>
<td>25830.76</td>
<td>25041.90</td>
</tr>
<tr>
<td>002</td>
<td>R32</td>
<td>S Series 1.8 L ...</td>
<td>Price</td>
<td>25259.93</td>
<td>25830.76</td>
<td>25041.90</td>
</tr>
<tr>
<td>001</td>
<td>R1A</td>
<td>S Series 1.8 L ...</td>
<td>Price</td>
<td>25259.93</td>
<td>25830.76</td>
<td>25041.90</td>
</tr>
<tr>
<td>002</td>
<td>R1A</td>
<td>S Series 1.8 L ...</td>
<td>Price</td>
<td>25259.93</td>
<td>25830.76</td>
<td>25041.90</td>
</tr>
<tr>
<td>001</td>
<td>R30</td>
<td>S Series 1.8 L ...</td>
<td>Price</td>
<td>25259.93</td>
<td>25830.76</td>
<td>25041.90</td>
</tr>
<tr>
<td>002</td>
<td>R30</td>
<td>S Series 1.8 L ...</td>
<td>Price</td>
<td>25259.93</td>
<td>25830.76</td>
<td>25041.90</td>
</tr>
</tbody>
</table>

In this example, the codes in the first column represent elements in the Actvsbud dimension. By using a simple IF function, you can convert 001 to Actual and 002 to Budget.

The second column contains four region codes, which map to regions such as Argentina, United States, and Greece. If these were the only values, you could write a nested IF formula to convert the values to elements. As the number of possible values increases, writing a nested IF formula can be cumbersome and
error-prone. As an alternative, you can create a two-dimensional cube that serves as a lookup table for retrieving element names.

**Importing Data Using Processing Worksheets**

You can create a processing worksheet to perform the following tasks:
- Import data from input rows that require transformations.
- Update cubes but not create cubes, nor create consolidations.
- Use as a staging area for importing data.

TM1 reads the input records, one at a time, into the first row of the processing worksheet, and then sends the data values associated with the record to a TM1 cube.

Beneath the first worksheet row, a processing worksheet includes:
- Conversion instructions for values that map to element names but do not match element name spellings.
- Any data transformation calculations that modify data values before importing.
- Database Send (DBS) formulas that map the input data to cells in the cube. Each formula sends a value from the first row to a location in the cube identified by one element in each dimension of the cube.

**Note:** You must use DBS formulas, not DBSW formulas, in processing worksheets. You must use DBR formulas rather than DBRW formulas in processing worksheets.
- Other values in each input row supply the element names directly or through conversion instructions.

The following process summarizes the steps required to import data:
- Read in the first input record into the processing worksheet as an example.
- Compare the input with the cube structure.
- Map the input values to element names, if necessary.
- Build a DBS formula for each input value that populates a cube cell.
- Process all input records.

**Reading in the First Input Row**

You can use processing worksheets to process data from the following data sources:
- ASCII files
- ODBC data sources
- TM1 cubes

The next three sections provide the steps for reading an initial record from each data source.

**Procedure**

1. Create a new Excel worksheet, and close any other ones.
2. Click **TM1, Process Data, Example**.
   - The Select Cube, ODBC or Flat File dialog box opens.
3. To choose a data source, click **Cube, ODBC**, or **Flat File**.
The selection dialog box opens.

4. Select the source cube, data source or input file and click **OK**.
   
   For example, for an input file, select price.cma file in your \install_dir\PData directory. For an ODBC data source, change the Client and Password, if necessary.
   
   For an input file, the first record of the input file displays in the first row of the processing worksheet.

5. For an input file, click **Edit**, **Save** and save the processing worksheet as PriceProcessing.xls.

6. For an ODBC data source, select a table and click **OK**.
   
   The first record of the ODBC source displays in the first row of the worksheet.

7. For a source cube, click **Export**.
   
   The first record of the source cube displays in the first row of the worksheet.

**Comparing Input Records with a Cube's Structures**

The examples in this manual process data into the sample SalesCube cube, which has the following structure.

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Sample Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actvsbud</td>
<td>Actual, Budget</td>
</tr>
<tr>
<td>Region</td>
<td>Argentina, Belgium, United States</td>
</tr>
<tr>
<td>Model</td>
<td>S Series 1.8 L Sedan, S Series 2.0 L Sedan</td>
</tr>
<tr>
<td>Account1</td>
<td>Units, Sales, Price</td>
</tr>
<tr>
<td>Month</td>
<td>Jan, Feb, Mar, Apr</td>
</tr>
</tbody>
</table>

To populate the SalesCube cube, each record in the data source must contain the following detail:

- One or more cell values.
- Names of elements from different dimensions that identify the cell location for each imported value, or input values that you can map to element names.

The element information can be incomplete. For example, when the source records contain multiple values for a single measure, the measure is optional. You can supply the missing measure using a DBS formula.

Consider the following record, as it appears in a processing worksheet. This record contains monthly price data for a single car model. Mapping this data to the SalesCube cube first requires converting two input values to element names and supplying missing element names.
Let's examine each input value:

- Cell A1 contains a code that identifies the price data as actual or budget amounts. Code 001 represents Actual, and Code 002 represents Budget, which are two elements in the Actvsbud dimension. By using the Excel IF function, you can convert these values to element names.

- Cell B1 supplies a region code that corresponds to an element name in the Region dimension. Suppose that the input has 21 region codes that require conversion. To convert these values, you can create a two-dimensional cube that serves as a lookup table.

- Cell C1 supplies car model names, exactly as found in the Model dimension. No conversion is required.

- Cells D1 through I1 supply the monthly data. You need to map this data to elements in the Month dimension.

Missing from each source record is a cell containing Price, which is an element in the Account1 dimension. You specify this value directly in the DBS formulas that send the data values to the cube.

**Converting Using IF Formulas**

The first column in price.cma contains a scenario code, 001 for actual and 002 for budget. You can use the Excel IF function to convert the code to the name of the corresponding element from the Actvsbud dimension.

**Procedure**

1. Click cell A3 of the processing worksheet.
2. Enter the following formula:
   
   `=IF(A1="001","Actual","Budget")`

**Mapping Using Fixed Labels**

Cells D1 through O1 contain values that map to the 12 elements in the Month dimension (Jan - Dec). Because these input columns always map to the same months, you can enter the element names directly in the processing worksheet.

Element names must exactly match the spellings in the dimension. You can avoid misspelling names by copying them from the Subset Editor window.

The following steps illustrate how to copy element names from the Subset Editor window.

**Procedure**

1. Open the Server Explorer.
2. Double-click the Month dimension.
   
   The Subset Editor window opens.
3. Select the twelve months Jan - Dec in the Tree pane.
4. Click Edit, Pick Elements, Horizontal.
5. Return to the processing worksheet.
6. Right-click cell D3 and click Paste.

   TM1 pastes the element names horizontally starting in cell D3.
Converting Using a Lookup Cube

Column B, the second input column, supplies the codes that identify the 21 regions in which the car models are sold. For example, R54 represents Argentina. To convert these codes to element names, you have two choices:

- Create a nested IF formula. As the list of codes increases, this becomes a cumbersome choice.
- Create a two-dimensional cube that serves as a lookup cube for the region names, and then retrieve the names using a DBR formula.

We'll create a lookup cube called Translate that contains two dimensions, RegCodes and RegName.

Importing Unique Names

Using TurboIntegrator, you can create a dimension whose elements are unique values from an input column. In this example, the second column is price.cma.

Procedure

1. Open the Server Explorer.
2. In the Tree pane, right-click Processes and click Create New Process.
   The TurboIntegrator dialog box opens.
3. Specify an ASCII data source type.
4. Click the Data Source Name Browse button and browse to the price.cma file in your \install_dir\Pdata directory.
5. Click the Variables tab.
6. Specify a Content type of Ignore for all columns except the one that supplies the codes you want to import. In this example, column 2 (identified by a sample value of R54) supplies the codes you want to import.
7. Click the Maps tab.
8. Specify No Action in both the Cube Action and Data Action sections of the Cubes subtab.
9. Click the Dimensions subtab, and do the following:
   - Type Translate in the Dimension field.
   - Select Create from the Action list.
   - Select Numeric from the Element Type list.
10. Click File, Save and save the process as create_RegCodes_dimension.
11. Click File, Execute to create the RegCodes dimension.

Results

RegCodes is now available as a dimension in the Server Explorer.

Creating a RegName Dimension

Follow these steps to create the RegName dimension with a single string element.

Procedure

1. Open the Server Explorer.
2. In the Tree pane, right-click Dimensions and click Create New Dimension.
   The Dimension Editor opens.
3. Click Edit, Insert Element.
   The Dimension Element Insert dialog box opens.
4. Type **Name** in the **Element Name** field.
5. Select **String** from the **Element Type** list.
6. Click **Add**.
   The **Name element** now opens as a string element.
7. Click **OK**.
8. Click **Edit**, **Save** and save the dimension as RegName.

### Creating the Translate Cube
Follow these steps to create the Translate cube.

#### Procedure
1. Right-click **Cubes** in the Server Explorer, and click **Create new cube**.
   The Creating Cube window opens.
2. Type **Translate** in the **Cube Name** field.
3. In the **Available Dimensions** box, double-click **RegCodes**.
   RegCodes moves to the Dimensions in new cube box.
4. In the **Available Dimensions** box, double-click **RegName**.
   RegName moves to the Dimensions in new cube box.
5. Click **OK** to save the two-dimensional Translate cube.

### Populating the Translate Cube
Using the Cube Viewer, you can now enter the corresponding region names for the region codes.

The following steps illustrate how to populate the Translate cube with region names.

#### Procedure
1. In the **Tree** pane of the Server Explorer window, double-click **Translate**.
   The Cube Viewer opens.
2. Press F9 to see the elements in each dimension of the Translate cube.
3. Enter the region names that correspond to the region codes, using the table as your guide.

<table>
<thead>
<tr>
<th>RegCode</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>R54</td>
<td>Argentina</td>
</tr>
<tr>
<td>R32</td>
<td>Belgium</td>
</tr>
<tr>
<td>R55</td>
<td>Brazil</td>
</tr>
<tr>
<td>R1B</td>
<td>Canada</td>
</tr>
<tr>
<td>R56</td>
<td>Chile</td>
</tr>
<tr>
<td>R45</td>
<td>Denmark</td>
</tr>
<tr>
<td>R33</td>
<td>France</td>
</tr>
<tr>
<td>R49</td>
<td>Germany</td>
</tr>
<tr>
<td>R44</td>
<td>Great Britain</td>
</tr>
<tr>
<td>R30</td>
<td>Greece</td>
</tr>
<tr>
<td>R353</td>
<td>Ireland</td>
</tr>
<tr>
<td>R39</td>
<td>Italy</td>
</tr>
</tbody>
</table>
Creating the DBR Formula
You can now create a DBR formula that retrieves region names for each region code read into the processing worksheet.

Note: You must use DBR formulas, not DBRW formulas, in processing worksheets.

Procedure
1. In the processing worksheet, click cell B3.
2. Click TM1, Edit Formula.
   The Edit Formula bar opens.
3. Click DB Ref.
   The Select Cube dialog box opens.
4. Click Pick.
   A different Select Cube dialog box opens.
5. Select local:Translate and click OK.
   TM1 correctly assumes that the element from the RegCodes dimension is in cell B1, but cannot find an element for RegName and shows it as undefined.
6. Click RegName.
   The Subset Editor opens.
7. Select the element Name and click OK.
   The regname field now displays Picked.
8. Click OK in the Edit Reference to Translate dialog box.
   The Edit Formula bar now displays the complete formula:
   =DBR("local:Translate", $B$1, "Name")
   This formula returns the value from the Translate cube found at the intersection of the Regcodes element in cell B1 and the Regname element Name.
9. Click OK to insert the formula in cell B3.

Results

Cell B3 now displays Argentina, which is the correct region for the code R54.
Creating Database Send (DBS) Formulas

You can create DBS formulas that send numeric data values to the cube because:

- You converted codes that map to elements in the Actvsbud dimension.
- You converted codes that map to elements in the Region dimension.
- You mapped multiple data values to their appropriate months.

Insert the DBS formulas in a row beneath the rows containing the data and mapping instructions. Do not insert them in the first row because they will be overwritten as TM1 reads in records into the processing worksheet.

Procedure

1. Click cell D4, an empty cell that will store the first DBS formula.
2. Click TM1, Edit Formula.
   The Edit Formula bar opens.
3. Click DB Send.
   TM1 prompts you to select the value to be sent to the cube.
4. Double-click cell D1, which contains the Jan cell value.
   TM1 prompts you to select the type of cell reference.
5. Click Column Rel.
   The DBS formula always references row 1, but the column reference will be relative to the location of the formula.
   TM1 prompts you to indicate the type of data in the cell.
6. Click Numeric.
   The Select Cube dialog box prompts you to select the cube to be populated.
7. Click Pick.
   A different Select Cube dialog box opens.
8. Select the local:SalesCube cube and click OK.
   The Edit Reference to Cube dialog box displays with most of the mapping instructions for the value (cell D1) to be sent to the cube.
   Cell A3 supplies an element in the Actvsbud dimension.
   - Cell B1 supplies an element in the Region dimension.
   - Cell C1 supplies an element in the Model dimension.
   - Cell D3 supplies an element in the Month dimension.
   To complete the mapping, you must identify an element for the Account1 dimension. Price.cma contains price values, so all DBS formulas should map to the price element.
9. Click account1.
   The Subset Editor opens.
10. Select Price and click OK.
    The account1 field of the Edit Reference to Cube dialog box now displays Picked.
11. Click OK.
    The Edit Formula bar displays the generated DBS function:
    DBS(D$1,"local:SalesCube",$A$3,$B$3,$C$1,"Price",D$3)
    For a full explanation of the formula, see "DBS Syntax" on page 101.
12. Click OK to place this formula in the processing worksheet.
13. Copy the formula in D3 to the range E3:O3.
14. Save the PriceProcessing worksheet.

**DBS Syntax**
The DBS function uses the following syntax:

```
DBS (value, server:cube, e1, e2[...en]):
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Numeric value that is sent to the cube.</td>
</tr>
<tr>
<td>server:cube</td>
<td>The name of the cube that receives the sent value. The cube name must be</td>
</tr>
<tr>
<td></td>
<td>prefixed with the name of the server on which the cube resides, for example</td>
</tr>
<tr>
<td></td>
<td>sdata:SalesCube.</td>
</tr>
<tr>
<td>e1,...en</td>
<td>Elements that identify the cell location in the cube that receives the</td>
</tr>
<tr>
<td></td>
<td>value. Specify the element arguments in dimension order. For example, e1</td>
</tr>
<tr>
<td></td>
<td>must be an element from the first dimension of the cube, e2 must be an</td>
</tr>
<tr>
<td></td>
<td>element from the second dimension of the cube.</td>
</tr>
</tbody>
</table>

**Processing a Data Source into a Cube**

After you create a processing worksheet, you can process data into a cube.

If you have been completing the exercises in this section, you know that the PriceProcessing worksheet processes price values into the SalesCube cube.

You cannot use a processing worksheet to write values to cube cells that are calculated by rules, as you cannot edit rules-derived cell values.

- You must first verify that Price values in SalesCube are not derived by rules.
- Then you can process a data source into a cube.

**Procedure**

1. Open the Server Explorer.
2. Double-click the **SalesCube** cube.
3. Check if there is a rule attached to the cube. If not, skip to step 8.
4. If a rule is attached, open the rule in the **Rules Editor**.
5. Examine the rule to see if Price is calculated by rules.

   **Note:** Note that the rule includes the following statement which calculates the value for Price at both the numeric and consolidated levels:
   
   ```
   ['Price'] = N:DB('PriceCube', !actvsbud, !region, !model, !month); C: ['Sales'] \ ['Units'] * 1000;
   ```
   This statement calculates the value for Price at both the numeric and consolidated levels.
6. Insert a pound sign (#) at the beginning of both statements to disable the calculation of Price.
   ```
   #['Price'] = N:DB('PriceCube', !actvsbud, !region, !model, !month); #C: ['Sales'] \ ['Units'] * 1000;
   ```
7. Save the rule.
   You can now use the processing worksheet to process the Price.cma source file into SalesCube.
8. If no rule is attached, open the processing worksheet that contains the DBS formulas and any mapping instructions.
If you followed the earlier examples in this section, open the PriceProcessing processing worksheet.

9. Close any other worksheets.

10. Click **TM1, Process Data, Process**.
    The Select Cube, ODBC, or Flat File dialog box opens.

11. Click **Flat File**.
    The Select Input File dialog box opens.

12. Select the Price.cma sourcefile and click **OK**.
    TM1 processes the source file. During the process, a progress bar displays.
    TM1 sequentially reads each record of the source file into the first row of the processing worksheet. The processing worksheet recalculates after each record is read, and the DBS formulas send the values in the first row to the appropriate cell of the cube.

13. Browse SaleCube, and note that the Price values have been updated by the values in Price.cma.
Chapter 6. Controlling Access to TM1 Objects

This section describes how you can limit access to objects on an IBM Cognos TM1 server for all IBM Cognos TM1 installations, regardless of the authentication method.

Assigning Security Rights to Groups

You can assign object-level security for any non-administrative user group in TM1. By assigning security rights to groups, you can control a user's access to TM1 objects.

Note: You can not assign security rights for the ADMIN, DataAdmin or SecurityAdmin groups. The rights for these groups are predefined and appear disabled in the TM1 Security Assignments dialog box.

Note also that only the English versions of the security-level entries are accepted. The following keywords must be used as listed here when manually entering security levels in the TM1 cell security control cube.

The object-level security rights for TM1 groups are:

- **Admin** - Group has complete access to a cube, element, dimension, or other object.
- **Lock** - Group can view and edit a cube, element, dimension, or other object and can permanently lock objects to prevent other users from updating them.
- **Read** - Group can view a cube, element, dimension, process, or chore, but cannot perform operations on the object.
- **Reserve** - Group can view and edit a cube, element, dimension, or other object, and can temporarily reserve objects to prevent other users from updating them.
- **Write** - Group can view and update a cube, element, dimension, process, or chore.
- **None** - Group cannot see a cube, element, dimension, process, or chore, and cannot perform operations on the object.

The following table describes the security rights that you can assign to groups.
<table>
<thead>
<tr>
<th>Privilege</th>
<th>Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin</td>
<td>Cube</td>
<td>Members of the group can read, write, reserve, lock, and delete the cube. They can save public cube views. They can also grant security rights to other users for this object.</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Members of the group can access, update, reserve, lock, and delete the element. They can also grant security rights to other users for this object.</td>
</tr>
<tr>
<td></td>
<td>Dimension</td>
<td>Members of the group can add, remove, and reorder elements in the dimension, and can reserve or lock the dimension. They can save public dimension subsets. They can also grant security rights to other users for this object.</td>
</tr>
<tr>
<td></td>
<td>Application</td>
<td>Members of the group can see the application, use references within the application, and create both public and private references in the application. When a group has Admin privilege to an application, members of the group can set security privileges for all references and sub-applications within the application for other groups but not their own group.</td>
</tr>
<tr>
<td></td>
<td>Reference</td>
<td>Members of the group can use the reference, as well as update or delete the reference. They can publish private references, and privatize public references.</td>
</tr>
<tr>
<td>Lock</td>
<td>Cube</td>
<td>Members of the group have all privileges implied by Write permission, and can also lock the cube. When a cube is locked, nobody can update its data. The lock can be removed only by users who have Admin rights for the cube. Locks stay in place after the remote server shuts down.</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Members of the group have all privileges implied by Write permission, and can also lock the element. When an element is locked, nobody can update cube cells identified by the element. The lock can be removed only by users who have Admin rights for the element. Locks stay in place after the remote server shuts down.</td>
</tr>
<tr>
<td></td>
<td>Dimension</td>
<td>Members of the group have all privileges implied by Write permission, and can also lock the dimension. When a dimension is locked, nobody can edit the dimension structure. The lock can be removed only by users who have Admin rights for the dimension. Locks stay in place after the remote server shuts down.</td>
</tr>
<tr>
<td>Privilege</td>
<td>Object</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Read</td>
<td>Cube</td>
<td>Members of the group can see the cells in the cube, but cannot change their data.</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Members of the group can see the cells identified by the element, but cannot change their data.</td>
</tr>
<tr>
<td></td>
<td>Dimension</td>
<td>Members of the group can see the elements in a dimension, but cannot add, remove, or reorder the elements.</td>
</tr>
<tr>
<td></td>
<td>Process</td>
<td>Members of the group can see the process in the Server Explorer, and can manually execute the process, but cannot edit the process. <strong>Note:</strong> Privileges assigned to processes are ignored when a process is executed from within a chore.</td>
</tr>
<tr>
<td></td>
<td>Chore</td>
<td>Members of the group can see the chore in the Server Explorer, and can manually execute the chore, but cannot edit the chore.</td>
</tr>
<tr>
<td></td>
<td>Application</td>
<td>Members of the group can see the application and use any public references within the application to which they have at least <strong>Read</strong> privilege. They can create private references in the application, and also create private sub-applications.</td>
</tr>
<tr>
<td></td>
<td>Reference</td>
<td>Members of the group can open the reference, but cannot update the reference in the application. Members of the group can, however, perform a &quot;save-as&quot; operation to save a new private version of the reference.</td>
</tr>
<tr>
<td>Privilege</td>
<td>Object</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>Reserve</td>
<td>Cube</td>
<td>Members of the group have all privileges implied by Write permission, and can also reserve the cube to prevent other users from applying edits. The reservation can be removed either by the user who reserved the cube or by users who have Admin rights for the cube. A reservation expires automatically when the reserving user disconnects from the remote server or when the server shuts down.</td>
</tr>
<tr>
<td>Element</td>
<td>Element</td>
<td>Members of the group have all privileges implied by Write permission, and can also reserve the element to prevent other users from updating cube cells identified by the element. The reservation can be removed either by the user who reserved the element or by users who have Admin rights for the element. A reservation expires automatically, when the reserving user disconnects from the remote server or when the server shuts down.</td>
</tr>
<tr>
<td>Dimension</td>
<td>Dimension</td>
<td>Members of the group have all privileges implied by Write permission, and can also reserve the dimension to prevent other users from redefining the dimension. The reservation can be removed either by the user who reserved the dimension or by users who have Admin rights for the dimension. A reservation expires automatically when the reserving user disconnects from the remote server or when the server shuts down.</td>
</tr>
<tr>
<td>Write</td>
<td>Cube</td>
<td>Members of the group can read and update cells. They can save private cube views. The Write access privilege does not apply to cells identified by consolidated elements or to cells derived from rules.</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Members of the group can read and update the cells identified by the element and edit attributes of the element.</td>
</tr>
<tr>
<td></td>
<td>Dimension</td>
<td>Members of the group can edit element attributes, edit element formats, and create private subsets for the dimension. Members can also edit attributes for the dimension itself.</td>
</tr>
</tbody>
</table>
Privilege | Object | Description
---|---|---
None | Cube | Members of the group cannot see the cube in the Server Explorer, and thus cannot browse the cube.

Element | Members of the group cannot see the element in the Subset Editor or Dimension Editor, and cannot see the cells identified by the element when browsing a cube.

Dimension | Members of the group cannot see the dimension in the Server Explorer, and cannot browse a cube that contains the dimension.

Process | Members of the group cannot see the process in the Server Explorer, and thus cannot execute the process. **Note:** Privileges assigned to processes are ignored when a process is executed from within a chore.

Chore | Members of the group cannot see the chore in the Server Explorer, and thus cannot execute the chore.

Application | Members of the group cannot see the application or its contents in the Server Explorer.

Reference | Members of the group cannot see the reference in the Server Explorer.

### Interaction of Different Object Security Rights

If you apply different security rights to the objects that identify a cell of data, TM1 applies the most restrictive security right to the cell.

**Scenario 1**

Suppose you assign a user Read access to the SalesCube cube, and Write access to the elements in this cube. In this scenario, the Read access of the cube overrides the Write access of the elements, and the user can view cube data but cannot update the cube data.

**Scenario 2**

The SalesPriorCube cube contains the following dimensions:
- Actvsbud
- Region
- Model
- Account1
- Month

Suppose a user has Write access to the SalesPriorCube cube, Read access to all of the elements in the Actvsbud dimension, and Write access to all of the elements in the other dimensions. The elements in the Actvsbud dimension identify every cell in the cube, and therefore the user cannot update any cube data.
Scenario 3

You can change the security rights for both cubes and dimensions. When groups have security rights for a cube, those rights apply to all dimensions in the cube, unless you further restrict access for specific dimensions or elements.

Suppose you want several regional groups of users to read all data in the SalesPriorCube cube. You also want each group to update data in its own region. For example, you want salespeople in the North America group to update North America data.

To implement this security scheme, you could:
- Create groups that reflect sales regions.
- Add users to the appropriate groups.
- Grant each regional group write access to the SalesPriorCube cube.
- Grant the North America group read access to those elements that do not reflect data for the North America region.

The TM1 sample data reflects this security scheme. Usr1 is in the North America group, which has write access to the data associated with areas in the North America region, and read access to the data associated with areas in other regions.

Securing Cubes

You can enhance or restrict a group’s access to individual cubes. When you create a new cube, other groups initially have None access to the new cube. You must assign security rights to the new cube for other groups.

Assigning Security Rights for Cubes

Follow these steps to assign security rights for a cube.

Procedure

1. Open the Server Explorer.
2. Select the Cubes icon for the server you are working with.
3. Click Cubes, Security Assignments.
   The TM1 Security Assignments dialog box opens.
4. Click the cell at the intersection of the cube name and the group name for which you want to assign rights.
   You can assign rights for multiple cubes or to multiple groups by selecting a range of cells. To select a range of cells, click a cell to establish the top of the range, hold down Shift, and click further down the column or row to establish the bottom of the range.
5. Select the access level you want to assign.
   The name of the assigned access privilege displays in the cell.
6. Click OK.

Reserving and Releasing Cubes

When a user reserves a cube, that user gains exclusive rights to update the data in the cube. Other users cannot update the cube data until the cube is released. A cube can be released by either the user who reserved it or by a user who has Admin rights for that cube.
Consider reserving a cube as a way to temporarily freeze its data. A reservation expires automatically when the reserving user disconnects from the remote server or when the server shuts down.

**Procedure**
1. Open the Server Explorer.
2. Select the cube you want to reserve.
3. Click **Cube, Security, Reserve**.
   
   To release a cube:
4. Follow Steps 1 and 2 for reserving a cube.
5. Click **Cube, Security, Release**.

**Locking and Unlocking a Cube**
When a user locks a cube, only those users who have Admin rights for that cube can update its data or unlock the cube. Even the user who locks the cube cannot update the cube data or unlock the cube, unless he/she has Admin rights for that cube.

Consider locking a cube as a way to permanently archive its data. Locks stays in place after a server shuts down.

**Procedure**
1. Open the Server Explorer.
2. Select the cube you want to lock.
3. Click **Cube, Security, Lock**.
   
   To unlock a cube:
4. Follow Steps 1 and 2 for locking a cube.
5. Click **Cube, Security, Unlock**.

**Securing Elements**
You can enhance or restrict a group's access to individual elements using the **Element Security Assignments** dialog box.

**Assigning Security Rights for Elements**
Follow these steps to assign security rights for elements.

**Procedure**
1. Open the Server Explorer.
2. Select the dimension you want to work with.
3. Click **Dimension, Security, Elements Security Assignments**.
   
   The TM1 Security Assignments dialog box displays.
4. Click the cell at the intersection of the element name and the group name.
   
   You can assign rights for multiple elements or to multiple groups by selecting a range of cells. To select a range of cells, click a cell to establish the top of the range, hold down Shift, and click further down the column or row to establish the bottom of the range.
5. Select the access level you want to assign.
   
   The name of the assigned access privilege displays in the cell.
6. Click **Save** or **OK**.
**Note:** If you click **Save**, you can continue to assign security rights to different elements. You can access elements in other dimensions by selecting a dimension in the **Select Dimension** field.

**Interaction of Security Rights for Leaf and Consolidated Elements**

You can set different levels of security for a consolidated element and the leaf elements that belong to the consolidation.

For example, the Region dimension in the sample data has the following element hierarchy:

```
North America
  - Canada
  - Mexico
  - United States
```

Suppose Usr4 has **Read** access to the Canada leaf element and **None** access to the North America consolidated element. Usr4 can see the data identified by the Canada element, but cannot see the consolidated data identified by the North America element.

**Reserving and Releasing Elements**

When a user reserves an element, that user gains exclusive rights to update the data identified by that element. Other users cannot update the element's data, until the element is released. An element can be released by either the user who reserved it or by a user who has **Admin** rights for that element.

Consider reserving an element as a way to temporarily freeze the data that it identifies. A reservation expires automatically when the reserving user disconnects from the remote server or when the server shuts down.

**Procedure**

1. Open the Server Explorer.
2. Double-click the dimension you want to work with.
   - The Subset Editor displays.
3. Select the element you want to reserve.
4. Click **Edit, Security, Reserve**.
   - To release an element:
5. Follow Steps 1 through 3 for reserving an element.
6. Click **Edit, Security, Release**.

**Locking and Unlocking an Element**

When a user locks an element, only those users who have **Admin** rights for that element can update the data that it identifies. Even the user who locks the element cannot update its data, unless they have **Admin** rights for that element.

Consider locking an element as a way to permanently archive the data that it identifies. Locks stays in place after the remote server shuts down.
**Procedure**
1. Open the Server Explorer.
2. Double-click the dimension you want to work with.
   - The Subset Editor displays.
3. Select the element you want to lock.
4. Click **Edit, Security, Lock**.
   - To unlock an element:
5. Follow Steps 1 through 3 for locking an element.
6. Click **Edit, Security, Unlock**.

### Securing Dimensions

You can enhance or restrict a group’s access to individual dimensions.

By default, TM1 security controls dimensions on the IBM Cognos TM1 server, as follows:

- Only members of the ADMIN and DataAdmin groups can create and delete dimensions on the TM1 server.
- Groups with Read access to a dimension can view dimension and element attributes through the Server Explorer, but cannot edit attribute values.
- Other groups initially have None access to new dimensions.
- When no security has been assigned to an element in a dimension, groups have Write access to new elements in that dimension.
- When you assign security rights to at least one element in a dimension, groups have None access to new elements in that dimension. Existing elements keep their original access (Write), unless you change that access.

**Note:** If you change the security in a dimension, and you want to reset that security to the default setting (groups have Write access to new elements added to the dimension), shut down your TM1 server and manually delete the jElementSecurity<dimname>.cub file.

### Assigning Security Rights for Dimensions

Follow these steps to assign security rights for a dimension.

**Procedure**
1. Open the Server Explorer.
2. Select the **Dimensions** icon for the server you are working with.
3. Click **Dimensions, Security Assignments**.
   - The TM1 Security Assignments dialog box displays.
4. Click the cell at the intersection of the dimension name and the group name.
   - You can assign rights for multiple dimensions or to multiple groups by selecting a range of cells. To select a range of cells, click a cell to establish the top of the range, hold down Shift, and click further down the column or row to establish the bottom of the range.
5. Select the access level you want to assign.
   - The name of the assigned access privilege displays in the cell.
6. Click **OK**.
Reserving and Releasing Dimensions

When a user reserves a dimension, that user gains exclusive rights to add, remove, and reorder elements in that dimension. Other users cannot modify the dimension, until it is released. A dimension can be released by either the user who reserved it or by a user who has Admin rights for that dimension.

Consider reserving a dimension before you re-define it. A reservation expires automatically when the reserving user disconnects from the remote server or when the server shuts down.

Procedure
1. Open the Server Explorer.
2. Select the dimension you want to work with.
3. Click Dimension, Security, Reserve.
   To release a dimension:
4. Follow Steps 1 and 2 for reserving a dimension.

Locking and Unlocking a Dimension

When a user locks a dimension, only those users who have Admin rights for that dimension can add, remove, or reorder elements in that dimension. Even the user who locks the dimension cannot modify it, unless they have Admin rights for that dimension.

Consider locking a dimension if you want exclusive control of its definition.

Procedure
1. Open the Server Explorer.
2. Select the dimension you want to work with.
3. Click Dimension, Security, Lock from the pop-up menu.
   To release a dimension:
4. Follow Steps 1 and 2 for locking a dimension.
5. Right-click the mouse, and click Dimension, Security, Unlock.

Securing Processes

You can enhance or restrict a group's access to individual TurboIntegrator processes.

Note: TM1 ignores security rights assigned to TurboIntegrator processes when you execute a process from a chore. The security rights you assign to the chore determine the ability of a group to execute a process from a chore. For example, if a group has None access to Process1, but Read access to a chore that includes Process1, the group can execute Process1 from the chore.

Assigning Security Rights for Processes

Follow these steps to assign security rights for a process.

Procedure
1. Open the Server Explorer.
2. Select the Processes icon for the server you are working with.
3. Click **Processes, Security Assignments**.
   The TM1 Security Assignments dialog box opens.

4. Click the cell at the intersection of the process name and the group name.
   You can assign rights for multiple processes or to multiple groups by selecting multiple cells.
   To select a range of adjacent cells, click a cell to establish the top of the range, hold down Shift, and click further down the column or row to establish the bottom of the range.
   To select multiple non-adjacent cells, hold down CTRL, and click each cell.

5. Select the access level you want to assign.
   The name of the assigned access privilege displays in the cell(s).

6. Click OK.

**Allowing Processes to Modify Security Data**

The TM1 **Security Access** option controls whether a process is allowed to modify security data in the script of the process. Only members of the ADMIN and SecurityAdmin groups are allowed to set this option. This option is set on a process-by-process basis from the **Process** menu in Server Explorer.

When the **Security Access** option is enabled for a process:

- Members of the DataAdmin group are not allowed to edit the process since it may contain scripts that could modify TM1 security.
- Only members in the full ADMIN group can edit a process after the **Security Access** option has been enabled.
- Members of the SecurityAdmin group can view processes and turn this option on and off, but are never allowed to edit the contents of a process.
- The **View** option on the **Process** menu becomes available to allow users in the DataAdmin and SecurityAdmin groups to view processes in read-only mode.

For more details about the ADMIN, SecurityAdmin and DataAdmin groups, see the IBM Cognos TM1 Operation Guide.

**Enabling the Security Access Option for a Process**

By default, the **Security Access** option is disabled for any new process. If you want to allow a new process to modify TM1 security, you need to manually enable the **Security Access** option for that process.

**Procedure**

1. In Server Explorer, select a process.
2. Click **Process, Security Access**.
   A check mark next to the Security Access option indicates that TM1 will allow the selected process to modify security data.

   **Note:** If you upgraded to TM1 9.4 or later from an earlier TM1 database, the Security Access option is automatically enabled for all of your existing TM1 processes. This allows your existing processes to continue running without requiring you to individually set the Security Access option for each process.

**Viewing a Process in Read-only Mode**

When the **Security Access** option is enabled for a process, users in the DataAdmin and SecurityAdmin groups can only view the process in read-only mode.
Procedure
1. In Server Explorer, select a process.
2. Right-click the process and click **View**.
   The process displays in read-only mode.

Securing Chores
You can enhance or restrict a group's access to individual chores.

**Assigning Security Rights for Chores**
Follow these steps to assign security rights for a chore.

**Procedure**
1. Open the Server Explorer.
2. Select the Chores icon for the server you are working with.
3. Click **Chores, Security Assignments**.
   The TM1 Security Assignments dialog box opens.
4. Click the cell at the intersection of the chore name and the group name.
   You can assign rights for multiple chores or to multiple groups by selecting multiple cells.
   To select a range of adjacent cells, click a cell to establish the top of the range, hold down Shift and click further down the column or row to establish the bottom of the range.
   To select multiple non-adjacent cells, hold down CTRL, and click each cell.
5. Select the access level you want to assign.
   The name of the assigned access privilege displays in the cell(s).
6. Click **OK**.

Securing Applications and References
You can assign security privileges for public items (references or sub-applications) within public TM1 applications to user groups on the IBM Cognos TM1 server. You must have **Admin** privilege to an application to assign security to items within the application.

You cannot assign security for private applications and references; only the user who creates a private item can access the item.

**Procedure**
1. In the Server Explorer, right-click the application that contains the items to which you want to assign security.
2. Click **Security, Security Assignments**.
   The TM1 Security Assignments window opens. The Name list contains all public items (TM1 objects, Excel files, and sub-applications) that reside in the current application.
3. Select the cell at the intersection of the item for which you want to define security and the user group to which you want to assign security.
4. Click one of the available security privileges.
   For details, see "Assigning Security Rights to Groups” on page 103.
5. Repeat steps 3 and 4 for any other items for which you want to define security.
6. Click OK.

For any given user group, you can assign the Read or Admin privilege to a reference when you assign the None privilege to the source object associated with the reference. In this scenario, members of the user group cannot see the reference in an application.

For example, if you assign the None privilege to a user group for a cube, but assign the Read privilege for a reference to the same cube, members of the user group cannot see the reference to the cube in an application.

Securing Cells

Cell-level security applies to a specified cell and overrides all other TM1 security.

Cell-level security requires:

- Creating a cell security control cube that contains a subset of the dimensions of a cube whose cell-level security you configure.
- Setting security for the appropriate cells in the security control cube by assigning security rights for TM1 security groups.

Note: For element-level security to apply to a cell, no security rights can be assigned to any TM1 security group for the cell. Cell-level security overrides element-level security, so cell-level security for the cell must be undefined.

Cell-level security applies to leaf elements and generally does not apply to consolidations, although you can use the None and Read security rights to control the display or editing of consolidations.

Creating a Cell Security Control Cube

Follow these steps to create a cell security control cube.

Procedure

1. In TM1 Architect or TM1 Perspectives, right-click on the cube for which you want to define cell-level security and then select Security, Create Cell Security Cube.

   TM1 automatically creates a security control cube using the naming format }CellSecurity_CubeName where CubeName is the name of the cube that you selected. For example, if you selected the cube SalesCube, then TM1 creates the security control cube }CellSecurity_SalesCube.

   TM1 adds the dimensions of the original cube required to set security to the newly created security control cube, plus the Groups dimension is added as the last dimension in the new cube.

2. Click View, Display Control Objects if the control cubes are not already visible.

   TM1 displays the new security control cube along with the original cube.
To apply security to cells in the security control cube, by TM1 security group:

3. Open the security control cube you just created by double-clicking the security control cube, for example,

4. Click Recalculate to display the security control groups, or click Options, Automatic Recalculate.
5. Expand the rows to display the cells to which you want to assign security rights.

Note: Remember, cell-level security applies to leaf elements.
6. Enter the security level in the cube cells to assign security rights by user group. For details, see "Assigning Security Rights to Groups" on page 103. For example, the months in the second quarter for the Inspectors group have security assigned as None.
7. Close the security control cube.
8. Save the view.
9. Test the security levels by logging in as a user who is a member of the affected security group and viewing the cube for which you have set security.

Using Rules to Define Cell-Level Security
In the security control cube, you can use TM1 rules to apply cell-level security instead of entering security rights into security control cube cells. For details, see the IBM Cognos TM1 Operations Guide.
Suppose you want to create a rule to apply cell-level security for the CellSecurity_SalesCube cube. The following rule prevents any users in the Inspectors group from viewing any cells identified by the element Greece.

\[
[\text{Greece}', \text{Inspectors'}] = S: \text{NONE};
\]

**Note:** Make sure that security group names are unique and other dimensions or elements do not use the same name.

The benefits of using rules to implement security are:

- You do not enter security rights into cells in the security control cube, saving data entry time.
- Because TM1 does not store string rule values in memory nor saves them to disk, you save on memory consumption and disk storage.
Chapter 7. Using TM1 Action Buttons to Build Worksheet Applications

This section describes IBM Cognos TM1 Action button functionality which you can use to run processes and navigate between worksheets and Websheets.

Overview

You can insert an Action button into a worksheet so users can run a TurboIntegrator process and/or navigate to another worksheet. Users can access these buttons when working with worksheets in Microsoft Excel with TM1, or with Websheets in TM1 Web.

An Action button can perform any of the following tasks:
- Run a TurboIntegrator process.
- Navigate to another worksheet.
- Run a TurboIntegrator process and then navigate to another worksheet.
- Recalculate a worksheet or rebuild the TM1 Active Form in a worksheet.

The following figure shows an example of an Action button in a worksheet.

Adding an Action Button to a Worksheet

You can insert an Action button into any empty cell in a worksheet using the following steps.

Procedure

1. In Excel, select an empty cell in your worksheet where you want to insert the Action button.

   Note: An Action button can not be inserted into a cell that contains data.

2. Select Insert Action Button from the TM1 menu or click the Insert Action button on the TM1 toolbar.
The Action button is inserted into the currently selected cell and the Action Button Properties dialog box appears.

3. In the Action Button Properties dialog box, click the TM1 server list to select the server where your data is located.
   If you want to dynamically retrieve the IBM Cognos TM1 server name from a cell or named-range whenever the Action button is run, click the Use Reference checkbox and then enter a cell or named-range reference.

   - To select a cell reference, click the Excel Reference button and then click the cell in the current worksheet where the server name is located.
   - To retrieve the process name by referencing a named range in Excel, use the following format:
     =NameOfRange
     The named range must point to only a single cell that contains text for the server name.
   If you are not currently connected to the server that you want to use, click Connect to log in.

4. Click the Action option that you want the Action button to perform.
   You then need to configure the Action button depending on the type of action that you selected. For detailed steps, see the following sections:

<table>
<thead>
<tr>
<th>Action</th>
<th>See</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run a TurboIntegrator Process</td>
<td>&quot;Configuring an Action Button to Run a Process&quot;</td>
</tr>
<tr>
<td>Go to another Worksheet</td>
<td>&quot;Configuring an Action Button to Navigate to Another Worksheet&quot; on page 124</td>
</tr>
<tr>
<td>Run a Process, then go to another Worksheet</td>
<td>&quot;Configuring an Action Button to Run a Process and Navigate to a Worksheet&quot; on page 126</td>
</tr>
<tr>
<td>Recalculate / Rebuild</td>
<td>&quot;Configuring an Action Button to Recalculate or Rebuild a Worksheet&quot; on page 127</td>
</tr>
</tbody>
</table>

5. Set the appearance properties of the Action button. See "Setting the Appearance of an Action Button" on page 128.

6. To finish the Action button and return to your worksheet, click OK in the Action Button Properties dialog box.

Results

The Action button is updated in your worksheet and can now be used.

**Configuring an Action Button to Run a Process**

The following steps summarize how to configure an Action button to run a TurbolIntegrator process.

**Procedure**

1. In the Action Button Properties dialog box, select the Run a TurboIntegrator Process option.
   The Process tab displays.
2. Select the process that you want to run. See “Selecting the Process to Run.”
4. Select the calculation option that you want TM1 to apply before the Action button runs the process. See “Configuring an Action Button to Recalculate or Rebuild a Worksheet” on page 127.
5. Select the calculation option that you want TM1 to apply after the process has completed. See “Setting Process Options for Calculation” on page 123.
6. Configure the messages that you want TM1 to show before and after the process is run. See “Setting Process Options to Show Messages” on page 123.

Selecting the Process to Run

You can select the process you want to run in one of either two ways:
• Select the process name from the Process list.
• Dynamically retrieve the process name using an Excel reference.

Select the Process Name from the Process List

If you want to select the process name from a list of available processes on the current IBM Cognos TM1 server, perform the following steps.

Procedure

In the Process tab of the Action Button Properties dialog, click the Process list to select an available process from the TM1 server to which you are currently connected.

The Parameters grid opens and displays the parameters for the selected process.

Results

You then need to enter the parameter values for the selected process in the Parameter grid. See “Enter Parameter Values into the Parameter Grid” on page 122.

Use an Excel Reference to Retrieve the Process Name

If you want to dynamically retrieve the process name by referencing a cell or named range in the current worksheet, perform the following steps. The process name will be retrieved when the Action button is run.

Procedure

2. Click the Excel Reference button next to the Process Name box to select a cell from the current worksheet.
   The Select a Cell dialog box opens.
3. Click the cell in the current worksheet where the process name is located.
   The location is automatically entered into the Select a Cell dialog box.
4. Click OK to close the Select a Cell dialog box.
   The cell reference appears in the Process Name box.
5. To retrieve the process name by referencing a named range in Excel, use the following format:
   =NameOfRange
The named range must point to only a single cell that contains text for the process name.

**Setting Process Parameters**
Depending on how selected the name of the process to run, enter the parameter values in one of the following two ways.
- Enter parameter values into the **Parameter** grid.
- Create an Excel reference to retrieve the parameter values.

**Enter Parameter Values into the Parameter Grid**
If you selected the process name directly from the **Process** list, you can then enter the parameter values into the **Parameter** grid using the following steps. You can either type the values into the grid for each parameter, or use an Excel reference to dynamically retrieve a parameter value from the current worksheet when the Action button is clicked.

**Procedure**
1. In the **Process** tab of the **Action Button Properties** dialog, enter the parameter values into the **Parameters** grid.
   - To directly enter the parameter values, type the values into the grid for each parameter.
2. To create a reference that dynamically retrieves a parameter value from the current worksheet, select the **Value** cell and then click ![Select a Cell](image).
   - The Select a Cell dialog box opens.
3. Use the Select a Cell dialog box to select the cell in your worksheet where the parameter value can be found.
**Use an Excel Reference to Retrieve the Parameter Values**

If you selected *Get Process info from Worksheet* in the *Process* list, you must create an Excel reference that dynamically retrieves the process parameters from a worksheet.

**Procedure**

1. Click the Excel Reference button next to the *Parameters* box to select a reference from the current worksheet.
   The *Select a Range* dialog box opens.
2. Select the range of cells in your worksheet where the parameter values are located. Each cell must contain the value for only one parameter.

   **Note:** Parameters must be entered in the same order and type (string, numeric) as in the process.
3. Click **OK** to close the *Select a Range* dialog.
   The selected cell reference appears in the *Parameters* box.
   If you want to reference a named range in Excel, use the following format:
   
   **=NameOfRange**
   The named range must point to a single cell or a range of cells, depending on the parameters that the process is expecting.
   If the parameters for the process change, you must also update the process name and parameter settings for the Action button so the button can correctly run the process.

   **Note:** Due to a problem in Microsoft Excel if an Action button that runs a TurboIntegrator process with parameter inputs is created and saved in an Excel 2007 worksheet, the Action button stops working after the file is saved and then re-opened. As a workaround, when using an Action button in Excel 2007 to run a TI process that requires parameters, save the worksheet file in the .xls format.

**Setting Process Options for Calculation**

Use the *Process Options* dialog box to select the calculation operation that will be performed after the process is run.

**Procedure**

1. Click **Options** to show the *Process Options* dialog box.
2. Select the calculation operation that you want performed after the process has run. The available calculation options include:
   - **Automatically Recalculate Sheet** - Recalculates the values in the current worksheet.
   - **Rebuild Sheet** - Reloads the TM1 Active Form to its original report definition configuration.
   - **None** - The Action button will not perform any calculation or rebuild operation on the worksheet.

**Setting Process Options to Show Messages**

Use the *Process Options* dialog box to control the different message boxes that TM1 can show before and after the process is run.
**Procedure**

1. On the Process tab, click Options to open the Process Options dialog box.
2. Select the confirmation and status messages that you want TM1 to show.
   - **Show Success Message** - Displays a message after the process has successfully run.
   - **Show Failure Message** - Displays a message if the process does not run successfully.
   - **Show Confirmation Dialog** - Displays a confirmation message before the process is run. The user can click Yes or No.
3. Enter or edit the text for the messages you selected. You can also reference a cell or a named range to dynamically retrieve the message text. For example:
   - To retrieve the message text from the contents of cell A1 in the current worksheet, enter =A1 into the message text box.
   - To reference a named range in Excel, use the format =NameOfRange.
     The named range must point to only a single cell that contains the text for the message.

**Using the Action Button Server Name Property**

The Action button Properties dialog box includes a field where you can enter a cell or named-range reference to dynamically retrieve the IBM Cognos TM1 server name for an Action button. To use this feature, click the Use Reference checkbox in the TM1 server section on the Action Button Properties dialog box and then enter a cell or named-range reference.

**Configuring an Action Button to Navigate to Another Worksheet**

You can use an Action button to navigate to another worksheet in the same workbook, or to a worksheet in another workbook.

The following steps summarize how to configure an Action button to navigate to another worksheet.

**Procedure**

1. In the Action Button Properties dialog box, click the Go to another Worksheet option.
   The Worksheet tab displays.
2. Select the target worksheet. See “Select a Target Worksheet.”
3. Set the Match Title Elements option. See “Enable the Match Title Elements Option” on page 125.
4. Set the Replace Current Workbook option. See “Setting the Replace Current Workbook Option” on page 126.
5. Select the calculation option that you want TM1 to apply before the Action button navigates to another worksheet. See “Configuring an Action Button to Recalculate or Rebuild a Worksheet” on page 127.
6. Select the calculation option that you want TM1 to apply after the navigation has completed. See “Setting Calculation Options for after Navigating to a Worksheet” on page 126.

**Select a Target Worksheet**

You can select a worksheet by choosing it from a list or by directly typing its name.
Procedure

1. Click the **TM1 Applications** option and then click **Browse**.
   The Select a Worksheet dialog opens.

   **Note:** To select a target workbook, you can also click the **Files** option and then click **Browse**. Then select a worksheet file from the **Open** dialog and then click **Open**. Another method to select a target workbook, is to click the **Excel Reference** button.

2. Select a worksheet and then click **OK** to return to the **Action Button Properties** dialog.
   The Application folder path and name of the worksheet you selected displays in the Workbook box.

3. To manually enter the Application folder path to the workbook, type the path into the **Workbook** box:
   - Start with the first folder name under Applications and use a back-slash \\ character to separate folders. Do not include the Applications folder in the path. For example:
     Planning
     Sample\Bottom Up Input\Budget Input
   - To specify a worksheet and cell location to which you want to navigate, type the worksheet name and location in the **Sheet** box using the following format:
     =SheetName!ColumnNameRowName

   **Note:** If you enter a worksheet name that includes spaces, you must enclose the name in single quotes as shown in the following example.
   =’My First Sheet’!$A$2

Enable the Match Title Elements Option

The **Match Title Elements** option automatically matches and sets the text of the title dimensions in the target worksheet when a user clicks the Action button to navigate to the target worksheet.

When the **Match Title Elements** option is enabled, the dimensions in the source and target worksheets are automatically matched by the TM1 SUBNM and DBRW functions as follows:

- TM1 automatically matches title dimensions in the source and target worksheets based on the SUBNM formula in a cell.
  For example, when the same dimension exists in both the source and target worksheets, the element selected in the source worksheet is set for the same dimension in the target worksheet. When a column is selected in the source worksheet, it matches to the column with the same title dimensions in the target worksheet.

- TM1 automatically matches the row and column dimensions of the currently selected DBRW cell in the source worksheet to the matching title dimensions in the target worksheet, if they exist.
  For example, if Operating Expense and Feb-2004 are the row and column dimension elements for the currently selected element in the source worksheet, when navigating, these dimension elements are then matched to the title dimensions in the target worksheet. If the row Operating Expense and the column Feb-2004 are both selected in the source worksheet, the DBRW cell is the...
cell that exists in both the Operating Expense row and the Feb-2004 column. The row and column dimensions of the DBRW cell are then matched to title dimensions in the target worksheet.

**Setting the Replace Current Workbook Option**

The Replace Current Workbook option determines if the target worksheet is opened in a new window or in the same window, replacing the source worksheet.

For example:
- If Replace Current Workbook is not selected (default) and you are working in TM1 Web, then the source worksheet remains open and the target worksheet opens in a new tab.
- If Replace Current Workbook is selected and you are working in TM1 Web, then the source worksheet will be replaced by the target worksheet on the same tab and a new tab will not open.

**Important:** If you enable the Replace Current Workbook option, remember to save your workbook before testing the new button. You could lose your changes if you click the button and cause the current workbook to close.

**Setting Calculation Options for after Navigating to a Worksheet**

Set the calculation options to be applied to the target worksheet after navigating.

**Procedure**

On the Worksheet tab, select the Calculation options that you want to use.
- **Automatically Recalculate Sheet** - Recalculates the values in the current worksheet.
- **Rebuild Sheet** - Reloads the TM1 Active Form to its original report definition configuration.
- **None** - The Action button will not perform any calculation or rebuild operation on the worksheet.

**Configuring an Action Button to Run a Process and Navigate to a Worksheet**

To configure an Action that runs a TurboIntegrator process and then navigates to another worksheet, perform the following steps.

**Procedure**

1. In the Action Button Properties dialog box, click the Run a Process, then go to a Worksheet option. The Process tab displays.
2. Select the process that the Action Button will run. See "Configuring an Action Button to Run a Process” on page 120.
3. Select the worksheet to which the Action button will navigate. See "Configuring an Action Button to Navigate to Another Worksheet” on page 124.
Configuring an Action Button to Recalculate or Rebuild a Worksheet

You can use an Action button to perform only a recalculation or rebuild operation without running a TI process or navigating to a new worksheet. This can be useful if you only want to update the current sheet or reload the original version of an Active Form.

You can also use the Calculate tab to select the calculation operation that you want TM1 to perform before running a TI process or navigating to another worksheet.

Procedure

1. Insert an Action button into your worksheet as described in “Adding an Action Button to a Worksheet” on page 119.
2. On the Action Button Properties dialog box, click the Calculate tab.
3. In the Calculate tab, select the calculation option that you want to use.
   - Automatically Recalculate Sheet - Recalculates the values in the current worksheet.
   - Rebuild Sheet - Reloads the TM1 Active Form to its original report definition configuration.
   - None - The Action button will not perform any calculation or rebuild operation on the worksheet.
4. Set the appearance of the button as described in “Setting the Appearance Properties of an Action Button” on page 128.
5. On the Action Button Properties dialog box, click OK to close the dialog and insert the Action button into your worksheet.

Understanding Action Button Behavior with TM1 Active Forms

This section summarizes the behavior and order of operations when using an Action button with Active Forms.

Basic Action Button Behavior

Each time you click an Action button, the following steps are performed in this order:

1. The DBRW formula is captured for the currently selected cell in the source worksheet.
   - The DBRW cell formula is captured before the following Recalc (F9) operation because the recalculation could change the number of active form rows due to zero suppression and/or MDX-based row subsets.
2. A Recalculation (F9) or rebuild operation is performed on the current worksheet, depending on the calculation option that was selected on the Calculate tab of the Action Button Properties dialog box.
3. The updated values are captured for any other cells that are referenced by advanced mapping options.

Additional Action Button Behavior

After the above basic steps have been completed, the following additional steps are performed, depending on whether the Action button is running a TI process, navigating to another worksheet or both.

When using an Action button to run a TI process only:

1. The TI process is run.
2. The calculation operation that was selected on the Action button’s **Process Options** dialog box is performed.

When using an Action button to navigate only:
1. The navigation action begins.
2. Target values are set in the target worksheet.
3. The calculation operation that was selected in the **Worksheet** tab of the **Action Button Properties** dialog box is performed on the target worksheet.

When using an Action button to run a TI process and navigate to a worksheet:
1. The TI process is run.
2. The calculation operation that was selected on the Action button’s **Process Options** dialog box is performed.
3. The navigation action begins.
4. Target values are set in the target worksheet.
5. The calculation operation that was selected in the **Worksheet** tab of the **Action Button Properties** dialog box is performed on the target worksheet.

**Setting the Appearance Properties of an Action Button**

Use the **Appearance** tab to set the caption, background picture, and other visual features for the Action button.

Click the **Appearance** tab to adjust the appearance properties of the button.

The Appearance tab has the following options:
- **Caption** - Sets the caption text that displays on the button.
- **Font** - Displays a standard font dialog where you can change the font type and size of the button text.
- **Show Background Image** - Allows you to select an image file (bmp, gif, or jpg format) that will be stretched to fit the button.
- **Display as Hyperlink** - When enabled, this option displays the button as a hyperlink with blue, underlined text instead of a standard button.
- **Preview** - This area shows an example of the button.
- **Colors** - Allows you to set text and background colors of the button. Click the color sample to display a Color dialog where you can select a standard color or define a custom color.

**Using Advanced Navigation and Mapping Options**

Use the **Advanced Options** dialog to manually map fields between the source worksheet and the target worksheet when inserting an Action button that navigates from one worksheet to another. This tool helps you map dimensions, cells, and values from the source worksheet to the target worksheet.

**Note:** Advanced mapping is applied after any automatic mapping has been performed by the **Match Title Elements** option.

To open the **Advanced Options** dialog, click **Advanced Options** on the **Worksheet** tab.
The Advanced Options dialog includes a grid where you define the mapping of fields between the source and target worksheets. Use the Add and Delete buttons to manage the rows in the grid.

**Configuring Source to Target Mapping**

You map the source worksheet to the target worksheet by setting values for the Source Type, Source Object, Target Type, and Target Object fields in the Advanced Mapping grid.

You can use the grid to specify how elements in the source and target worksheets get matched up when the target sheet opens. Each row in the grid defines one mapping configuration.

Use the following steps to configure advanced mapping:

<table>
<thead>
<tr>
<th>Step</th>
<th>See</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indicate the type of object to map</td>
<td>“Set the Source Type”</td>
</tr>
<tr>
<td>Determine the value for the type of object you are using</td>
<td>“Set the Source Object”</td>
</tr>
<tr>
<td>Indicate the type of cell to map</td>
<td>“Set the Target Type” on page 130</td>
</tr>
<tr>
<td>Indicate where the value from the Source Object will be inserted</td>
<td>“Set the Target Object” on page 130</td>
</tr>
</tbody>
</table>

Repeat these steps to create more mapping configurations.

**Set the Source Type**

The Source Type field represents the type of object for the value you want to map.

Select the Source Type as follows:

- **SUBNM** - Indicates that you are mapping from a cell that contains a title dimension in the source worksheet.
- **Selected DBRW** - Indicates that you are mapping from a cell that contains a DBRW formula in the source worksheet.
- **Value** - Indicates that you will enter a string or numeric value that will be sent to the target.

**Set the Source Object**

The Source Object field takes a value, or Excel expression that evaluates to a value, depending on what is selected in the Source Type field.

Enter the Source Object as follows:

- If Source Type is set to SUBNM, then you need to specify the name of the title dimension that exists in the source worksheet.
- If Source Type is set to Selected DBRW, then you need to specify the name of a row or column title dimension that exists in the source worksheet.
- If Source Type is set to Value, then you need to enter a string or numeric value that will be sent to the target worksheet.

**Note:** You can also retrieve these values from the source worksheet by using the = symbol to create an Excel reference.
**Set the Target Type**

The **Target Type** is the type of cell in the target worksheet where the value from the **Source Object** field will be inserted.

Select the **Target Type** as follows:

- **SUBNM** - Indicates the target is a title dimension in the target worksheet.
- **Named Range** - Indicates the target is a named range in the target worksheet.
- **Range** - Indicates the target location is a cell in the target worksheet.

**Note:** If you set **Target Type** to either a **Named Range** or **Range**, any pre-existing data or formula in the target cell will be overwritten when you use the Action button to navigate. If the target cell contains a DBRW function, then the function will be lost and the cell will not be able to connect to, read from, or write to the IBM Cognos TM1 server.

**Set the Target Object**

The **Target Object** is the location in the target worksheet where the value from the **Source Object** will be inserted. This location can be either the name of a title dimension, a specific cell location, or a named range in the target worksheet, depending on what you selected for **Target Type**.

Enter the **Target Object** as follows:

- If **Target Type** is set to **SUBNM**, specify the name of the title dimension in the target worksheet. When **Target Type** is set to **SUBNM**, you must also enter a value for the **Subset** and **Alias** fields.
- If **Target Type** is set to **Named Range**, specify the name of the range in the target worksheet.
- If **Target Type** is set to **Range**, specify the cell location in the target worksheet.

You can enter a value for the **Target Object** by directly entering the location value or you can use an Excel reference to retrieve the location.

- **Directly enter value**
  
  To identify a location in the target worksheet, enter the value directly into the **Target Object** field without the = symbol. For example, enter C3 to identify the location of the **Target Object** as the cell C3 of the target worksheet.

- **Reference Excel cell**
  
  To reference a cell in the source worksheet that contains a location for the **Target Object**, include the = symbol. For example, the cell A1 in the source worksheet might contain the value C3 to represent the cell location for the **Target Object** in the target worksheet.

Repeat all of the above steps to create more mapping configurations.

**Example of Mapping a Source Value to a Target Cell**

This example shows the mapping of a hard-coded value to one cell in the target worksheet.
Example of Mapping a Source SUBNM to a Target Cell
You can specify a SUBNM title dimension in the source worksheet to map to a corresponding SUBNM, named range, or range (cell) in the target worksheet.

For example, the following figure shows the S Series 2.0 L Wagon title element being inserted in the cell B2 in the target worksheet as a formatted caption.

Example of Mapping the Selected DBRW to the Target Worksheet
In this example, the row and column title dimensions for the selected DBRW cell are displayed in the target worksheet.
Modifying an Existing Action Button

After an Action button is inserted, you can edit its properties, size, and location as described in the following sections.

Editing the Caption, Background Image, and Properties of an Existing Action Button

To edit the properties of an Action button, right-click on the button and select an option from the menu.

The right-click menu provides the following main options:
- **Caption** - Opens the Button Caption dialog to edit the button's caption.
- **Background Image** - Opens a file selection dialog so you can browse and select an image to use as the background of the button.
- **Properties** - Opens the Action Button Properties dialog where you configure the actions to take place when a user clicks the button.

Moving and Resizing an Existing Action Button

After you click away from a new Action button, the button becomes enabled and is no longer in design mode. If you want to resize or move the button, you must turn on design mode in Excel as follows:

**Procedure**

1. In Excel, click **View, Toolbars, Control Toolbox**.
   The Excel Control Toolbox opens.
2. Click the **Design Mode** button.
   The text labels disappear on the Action buttons in the current worksheet.
3. Click on the **Action** button that you want to move or resize.
Handles appear on the button indicating it can be resized and moved.
• To resize the button, click and drag any of the handles.
• To move the button, click and drag the button to a new location.

4. Exit design mode by clicking on the Design Mode button.
The handles on the Action button disappear and the button returns to active mode.
Chapter 8. Using Data Reservations

This section describes all of the administrator, modeler, developer and programmer tasks related to enabling and implementing the Data Reservation feature available in IBM Cognos TM1.

Note: By default, Data Reservation is not enabled. An administrator must enable and configure the feature before you can use the related TurboIntegrator (TI) and API functions to manage Data Reservations.

Data Reservation Overview

Data Reservation (DR) is a server-related feature in TM1 that allows you to configure exclusive write-access to regions of a cube for individual users. Once reserved, the data in that region can only be modified by that specific user until the reservation is released.

You can use DR to support your specific business processes or to control data entry by acquiring and releasing Data Reservations on an as-needed or dynamic basis. For example, DR provides an alternative way to dynamically control write-access to cube data instead of trying to dynamically adjust TM1 security to achieve similar write-access restrictions.

Note: All DRs must be managed via custom applications that you develop using the provided TurboIntegrator (TI) and API functions. You design the application to obtain, release and manage DRs based on the required business process.

When to use Data Reservations

You would use Data Reservations if you want to manually or dynamically manage user write-access to TM1 data in a way that relates to your business process. Depending on your exact needs, you can apply the feature either as-needed or dynamically using TurboIntegrator and API functions.

For example, you might use DR to do the following:

- Manually run a one-time TurboIntegrator process that uses Data Reservations to lock out all users from making further edits to certain data after completing a specific project or closing a recent budget.
- Apply Data Reservations using TM1 Action buttons and TurboIntegrator processes in a Microsoft Excel spreadsheet or TM1 Websheet to dynamically control write-access to data as users interact with the data based on their roles and your company’s business process.
- Use TM1 API functions within a custom external application to dynamically acquire and release Data Reservations.

How to use Data Reservations

To use Data Reservation, you must first enable and configure the feature for individual cubes and user groups and then use TurboIntegrator or API functions to programmatically apply and manage DRs.

To enable DR, use the following TM1 tools:
**Data Reservation monitoring tools**

You can use Server Explorer, the TM1Top utility, the TM1 Audit log, and certain TurboIntegrator and API functions to monitor Data Reservation activity and/or assignments. For details, see “Monitoring Data Reservations” on page 143.

**Data Reservation modes**

Data Reservation modes allow you to configure the Data Reservation feature depending on how you want to control write-access to your data. When you enable the Data Reservation feature for a cube, you choose one of the available Data Reservation modes.

The available Data Reservation modes include:

- Off (OFF)
- Required (REQUIRED)
- Required Shared (REQUEREDSHARED)
- Allowed (ALLOWED)

By default, Data Reservation is set to OFF for all cubes.

You configure the Data Reservation mode individually for each cube by entering the keyword for the mode in the cube’s `DataReservationMode` property in the `CubeProperties` control cube. For more details, see “DataReservationMode property” on page 141.

**Required mode**

The REQUIRED mode disables write access for all users for the entire cube and requires you to explicitly assign Data Reservations for any user that needs to write to this cube.

For example, a user must have a Data Reservation on a cell if they want to write to that cell.
You set this mode by entering a value of `REQUIRED` in the `CubeProperties` control cube for a specific cube.

**Required Shared mode**

The `REQUIREDSHARED` mode is a variation of the `REQUIRED` mode that allows Data Reservations for different users to overlap. All other aspects of this mode behave the same as `REQUIRED` mode.

You set this mode by entering a value of `REQUIREDSHARED` in the `CubeProperties` control cube for a specific cube.

`REQUIREDSHARED` mode was implemented to accommodate overlapping requests leveraging multi-node edit capability in IBM Cognos TM1 Applications. This mode is the default assigned DR mode on all cubes represented by Cube Views or Manual Dependencies in TM1 Applications.

In `REQUIRED` mode, the TM1 server restricts write access to a slice by only allowing a single user to have a reservation for a node at any one time. In `REQUIREDSHARED` mode the application must enforce this restriction if necessary.

For example, `REQUIREDSHARED` mode can be used to allow access for multiple users to the same consolidated node. However, the application would then need to restrict access to the leaf nodes of the consolidation by assigning TM1 security rights to the related elements. An example of this is shown in the following hierarchy.

**Table 2. Example hierarchy using `REQUIREDSHARED` mode for a Data Reservation**

<table>
<thead>
<tr>
<th>Consolidation: New England</th>
<th>User A</th>
<th>User B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leaf Nodes:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MA</td>
<td>WRITE access</td>
<td>READ access</td>
</tr>
<tr>
<td>CT</td>
<td>WRITE access</td>
<td>READ access</td>
</tr>
<tr>
<td>VT</td>
<td>WRITE access</td>
<td>READ access</td>
</tr>
<tr>
<td>NH</td>
<td>READ access</td>
<td>WRITE access</td>
</tr>
<tr>
<td>RI</td>
<td>READ access</td>
<td>WRITE access</td>
</tr>
<tr>
<td>ME</td>
<td>READ access</td>
<td>WRITE access</td>
</tr>
</tbody>
</table>

An application can restrict write access to the leaf nodes MA, CT, and VT for User A, and to NH, RI, and ME for User B. This restriction can be enforced using element level security. Both User A and User B can then acquire a shared data reservation on the consolidation named New England. The reservation allows both users write access to the slice, while the underlying security restricts access to each user’s set of leaf nodes.

**Allowed mode**

The `ALLOWED` mode maintains write access, based on security, for all users across the entire cube, but allows you to selectively restrict write access to an area of the cube by assigning Data Reservations to individual users as needed.

For example, `ALLOWED` mode lets you use Data Reservation to set aside a section of a cube for a specific user while keeping write access available for all other users to the rest of the cube.
You set this mode by entering a value of ALLOWED in the CubeProperties control cube for a specific cube.

---

**Understanding Data Reservation behavior with other TM1 features**

This section describes how Data Reservations (DR) interact with other TM1 features such as security, data spreading, TurboIntegrator (TI) processes and sandboxes.

**Data Reservations and Security**

Data Reservation is different from TM1 Security in the following ways

- DR does not override TM1 security assignments, but only adds another layer of write restriction on top of standard security for cube objects.
- DR applies to *individual* users while TM1 security applies to *groups* of users.
- DR applies restrictions only to cube data and individual users. It does not control any other TM1 objects.

This behavior is the same for both the REQUIRED and ALLOWED Data Reservation modes.

**Data Reservations and the TM1 user interface**

Data Reservation affects the appearance of cells in a cube view in all of the different TM1 user interfaces, such as Server Explorer, TM1 Perspectives and TM1 Web.

This behavior is different depending on the Data Reservation mode.

**REQUIRED mode**

When a cube is configured to use Data Reservation in REQUIRED mode, all of the cells appear with a gray background and are not writable except for the cells contained in your DR region. Only the cells in the DR for the current user will be writable and appear with a white background.

**ALLOWED mode**

When a cube is configured to use Data Reservation in ALLOWED mode, all of the cells are writable, depending on security, and appear with a white background except for the cells contained in the Data Reservations of other users. Cells reserved by other users appear with a gray background and are not writable.

For details on checking whether a cell is part of a DR, see “Displaying Data Reservation cell status in Server Explorer” on page 143.

**Data Reservations and Data spreading**

Data spreading behaves the same as before - cells that are not writable, such as calculated cells or cells with a hold, are skipped during a spread process. When a user has a Data Reservation and performs a data spreading operation, only the cells in that user’s DR region will be considered writable and all other cells will be skipped.

This behavior is the same for both the REQUIRED and ALLOWED Data Reservation modes.
Data Reservations and TurboIntegrator processes and chores

You should understand the following considerations when using Data Reservation and also running interactive (non-scheduled) and scheduled TurboIntegrator (TI) chores/processes:

Some of this behavior is different depending on which Data Reservation mode is being used and whether the chore is run interactively or scheduled.

Interactive Processes and Chores

When a user interactively runs a process or a chore, for example from the TM1 user interface, then that process/chore runs as that user.

- For REQUIRED mode, this means that the process/chore can write only to data defined in the DRs held by that user.
- For ALLOWED mode, the process/chore can write to any cell that is either contained in a DR for that user or has the appropriate security rights for that user, but the process/chore cannot write to cells contained in another user’s DR.

The following behavior is the same for both the REQUIRED and ALLOWED Data Reservation modes.

- If a write operation in the Interactive process/chore conflicts with the Data Reservation of another user, then the process/chore fails and an error message is displayed to the user.
- To run a process that acquires and releases DRs, the user running the process must belong to a user group that has the ManageDataReservation capability set to GRANT.

Scheduled Chores

When a scheduled chore automatically runs, it runs as the Admin user. Because of this, the chore may not be able to write to some cells if those cells are in the Data Reservation of another user.

When a scheduled chore encounters a Data Reservation conflict, the behavior is different depending on which Data Reservation mode is being used by the cube.

- In REQUIRED mode, if a write operation in a scheduled chore conflicts with a Data Reservation, the chore fails.
  To allow a scheduled chore to write to reserved cells when using DR in REQUIRED mode, you must handle this programatically in your TurboIntegrator process by allowing the Admin user to write to the reserved cells. You can modify the TurboIntegrator process to acquire and release the necessary DRs for the Admin or possibly have the process temporarily turn off/on the DR feature for the applicable cubes.
- In ALLOWED mode, if a write operation in a scheduled chore conflicts with a Data Reservation, the chore will perform a data rollback and wait for the reservation to be released. When the reservation is released, the chore will retry the operation.
  You cannot cancel a waiting chore by changing the Data Reservation mode to REQUIRED or OFF. The chore will continue to wait until one of the following events occurs.
  - Release the Data Reservation that is blocking the chore so the chore can continue.
  - Use the TM1 Top utility to cancel the chore.
- Shut down the server to cancel the chore.
- Deactivate the scheduled chore.

**Data Reservations and Sandboxes**

You should understand the following considerations about obtaining and releasing Data Reservations on a specific cube while also entering data into sandboxes of that same cube.

DRs apply to the base version of a cube and any sandbox created from that cube.

The following sandbox behavior applies only to the REQUIRED Data Reservation mode.

- If the DR feature is enabled for a cube, then you must have a DR on that cube to write to the base data or any sandbox of that cube.
  
  For example, if a user has a DR to the Sales cube, then that user can only write to that same set of cells in any sandbox created from the Sales cube.

- If your DR for a cube has been released, then you can no longer write to any sandbox of that cube.

- You can commit a sandbox to base data after a DR is released, however, the edited values for any cells that are no longer reserved for you will be dropped with errors and your changes will be lost. To successfully commit the edited values to base data, you must first re-acquire the necessary DRs on that cube before attempting to commit the sandbox.

**Data Reservations, Sandboxes, and Job Queue**

If you are using Data Reservations with sandboxes and the Job Queue feature, the following behavior applies when committing sandbox data to base data, depending on which Data Reservation mode is being used by the cube.

- In REQUIRED mode, all necessary Data Reservations must be held until the queued operation is complete for the sandbox data to be successfully saved to base data. If the necessary Data Reservations are released before the Job Queue saves the data or if a Data Reservation conflict is encountered, the values in the conflicting cells will not be saved.

- In ALLOWED mode, if a sandbox commit in the Job Queue conflicts with a Data Reservation, the Job Queue will perform a data rollback for that operation and wait for the conflicting reservation to be released. All jobs in the Job Queue will be blocked until the waiting job is allowed to continue. The only way to unblock the Job Queue is to release the conflicting Data Reservations.

**Enabling Data Reservation**

By default, Data Reservation is not enabled. An administrator must enable and configure the feature before you can use the related TurboIntegrator (TI) and API functions to manage Data Reservations.

Before using Data Reservations, you must:

- Enable the feature for individual cubes (configured in the |CubeProperties control cube).

  For details, see "Enabling Data Reservation for cubes" on page 141

- Allow user groups the ability to manage, acquire and release DRs for themselves and other users (configured in the Capability Assignments window).
For details, see “Enabling user groups to manage Data Reservations” on page 142.

After DR has been enabled for a cube, the feature applies to all users of that cube, but can only be managed by members of the user groups granted permission in the Capability Assignments window.

**Enabling Data Reservation for cubes**

Use the DataReservationMode property in the }CubeProperties control cube to enable or disable Data Reservation for a specific cube.

For more details about the }CubeProperties control cube, see the "Control Cubes" section in the *IBM Cognos TM1 Operation Guide*.

**Procedure**

1. In Server Explorer, click the View menu and then click to select Display Control Objects.
   
   All of the TM1 Control cubes appear in Server Explorer with a prefix of a right curly brace }. For example, }CubeProperties.
2. In the Tree pane of Server Explorer, click to expand the Cubes node and then double-click the }CubeProperties control cube.
3. Enter a value for one of the available Data Reservation modes at the intersection of the DataReservationMode element (property) and the cube name to enable the Data Reservation feature for that cube.

**DataReservationMode property**

The DataReservationMode property uses the following values in the }CubeProperties control cube to configure the Data Reservation feature for individual cubes.

Enter these values into the }CubeProperties control cube using English only. These keyword values are not translated.

For more details about the differences between the different Data Reservation modes, see “Understanding Data Reservation behavior with other TM1 features” on page 138.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>Turns off the Data Reservation feature for the specific cube.</td>
</tr>
<tr>
<td></td>
<td>The default value is OFF.</td>
</tr>
<tr>
<td>REQUIRED</td>
<td>Sets the Data Reservation feature to REQUIRED mode for a specific cube.</td>
</tr>
<tr>
<td></td>
<td>This mode disables write access for all users for the entire cube and requires you to explicitly assign Data Reservations for any user that needs to write to this cube.</td>
</tr>
<tr>
<td></td>
<td>For example, a user must have a Data Reservation on a cell if they want to write to that cell.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>REQUIREDSHARED</td>
<td>Sets the Data Reservation feature to REQUIREDSHARED mode for a specific cube. This mode is a variation of the REQUIRED mode that allows Data Reservations for different users to overlap. All other aspects of this mode behave the same as REQUIRED mode. REQUIREDSHARED mode was implemented to accommodate overlapping requests leveraging multi-node edit capability in IBM Cognos TM1 Applications. This mode is the default assigned DR mode on all cubes represented by Cube Views or Manual Dependencies in TM1 Applications. In REQUIRED mode, the TM1 server restricts write access to a slice by only allowing a single user to have a reservation for a node at any one time. In REQUIREDSHARED mode the application must enforce this restriction if necessary.</td>
</tr>
<tr>
<td>ALLOWED</td>
<td>Sets the Data Reservation feature to ALLOWED mode for a specific cube. This mode maintains write access, based on security, for all users across the entire cube, but allows you to selectively restrict write access to an area of the cube by assigning Data Reservations to individual users as needed. For example, ALLOWED mode lets you use DataReservation to set aside a section of a cube for a specific user while keeping write access available for all other users to the rest of the cube.</td>
</tr>
</tbody>
</table>

**Enabling user groups to manage Data Reservations**

Use the following capabilities in the TM1 Capability Assignments window to allow members of a user group to manage (acquire and release) Data Reservations for themselves and other users.

These capabilities are configured for an entire user group and not for individual users. A user must be a member of the group for the capability to apply.

**Note:** Only TM1 administrators have access to the Capability Assignments window.

For more details about Capability Assignments, see the IBM Cognos TM1 *Operation Guide*.
# Capability Description

<table>
<thead>
<tr>
<th>Capability</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManageDataReservation</td>
<td>Allows the members of the group to acquire and release Data Reservations. This capability is server-wide. Capabilities can not be applied to specific cubes or users. This capability is always set to GRANT for the standard ADMIN, DataAdmin, and SecurityAdmin groups and cannot be modified. Values for non-administrator groups: DENY - Default value for all non-administrator groups. GRANT - Enables this capability for a user group.</td>
</tr>
<tr>
<td>DataReservationOverride</td>
<td>Allows the members of the group to release reservations held by other users. This capability is always set to GRANT for the standard ADMIN, DataAdmin, and SecurityAdmin groups and cannot be modified. Values for non-administrator groups: DENY - Default value for all non-administrator groups. GRANT - Enables this capability for a user group.</td>
</tr>
</tbody>
</table>

## Procedure

1. In Server Explorer, right-click on a server and select **Capability Assignments**.
2. In the Capability Assignments window, enter values for the **ManageDataReservation** and **DataReservationOverride** capabilities at the intersection with the user group you want to configure.

## Monitoring Data Reservations

You can use the following tools to monitor Data Reservation activity and status.
- Server Explorer
- TM1 Top utility
- TM1 Audit Log
- TurboIntegrator and API functions

### Displaying Data Reservation cell status in Server Explorer

You check the status of any cell in a cube view in Server Explorer as follows:

**Procedure**

In Server Explorer, open a cube view, right-click a cell and then click **Edit Status**. A message box appears indicating whether the cell’s value can be edited or that the cell is not editable because of a Data Reservation or other data access restriction.
Using TM1 Top to monitor threads waiting for Data Reservations

You can use the TM1 Top utility to monitor threads that are waiting for a data reservation to be released. Any thread in this state is shown in TM1 Top with a Data Reservation Release (DDR) value in the State field:

Wait:DRR

For more details about TM1 Top wait states, see the section "Understanding Thread Processing States" in the "System and Performance Monitoring" chapter in the IBM Cognos TM1 Operation Guide.

Using the Audit Log to monitor Data Reservation events

If audit logging is enabled for the TM1 server, Cognos TM1 will record Data Reservation events. You can then use the TM1 Audit log to query and view the history of Data Reservation assignments.

For more details about using the TM1 Audit Log, see the "System and Performance Monitoring" chapter in the IBM Cognos TM1 Operation Guide.

The following table lists the five Audit log events that are specific to Data Reservations.

<table>
<thead>
<tr>
<th>Audit Log Event</th>
<th>Event Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Property Set: CubeDataReservationEnable</td>
<td>Recorded when the value of the CubeDataReservationEnable property for a cube has changed.</td>
</tr>
<tr>
<td>Data Reservation: Acquired</td>
<td>Recorded when acquiring a Data Reservation.</td>
</tr>
<tr>
<td>Data Reservation: Released</td>
<td>Recorded when releasing a Data Reservation.</td>
</tr>
<tr>
<td>Data Reservation: Rollback Acquire</td>
<td>Recorded when rollback of a reservation acquire occurs.</td>
</tr>
<tr>
<td>Data Reservation: Rollback Release</td>
<td>Recorded when rollback of a reservation release occurs.</td>
</tr>
</tbody>
</table>

Notes about Rollback Events

- The rollback of a Data Reservation acquire operation is to release the reservation.
- The rollback of a Data Reservation release operation is to re-acquire the reservation.

Querying for Data Reservation events

All Data Reservation events are associated with the Cognos TM1 cube object and are therefore grouped as cube events in the Audit log. You can query for these events in the Audit Log window as follows.
Procedure
1. In Server Explorer, open the Audit Log window.
2. In the Event Type section, select the Object option.
3. Set the Object Type field to Cube.
4. Set the Event drop-down list to one of the Data Reservation event types.
5. Run the query.

Using TurboIntegrator and API functions to monitor Data Reservations
As a custom solution, you could use the following Cognos TM1 TurboIntegrator and API functions to programmatically iterate cube objects and retrieve information about the related Data Reservations for each cube.

- TurboIntegrator - CubeDataReservationGet on page 148
- TM1 API - TM1DataReservationGetAll on page 153

Managing Data Reservations with TurboIntegrator functions
You can use the following TurboIntegrator (TI) functions to programmatically obtain, release and manage Data Reservations.

- CubeDataReservationAcquire
- CubeDataReservationRelease
- CubeDataReservationReleaseAll
- CubeDataReservationGet
- CubeDataReservationGetConflicts

Data Reservations are defined by a specific cube, user, and tuple (an ordered list of elements).

**CubeDataReservationAcquire**
CubeDataReservationAcquire acquires a Data Reservation for the specified cube, user and tuple.

This is a TurboIntegrator function, valid only in TurboIntegrator processes.

**Syntax**

CubeDataReservationAcquire(Cube, User, bForce, Address, [AddressDelimiter])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cube</td>
<td>Name of the cube.</td>
</tr>
<tr>
<td>User</td>
<td>Name of the owner for the new reservation.</td>
</tr>
<tr>
<td></td>
<td>The user name supplied will be validated to make sure it is an existing user.</td>
</tr>
</tbody>
</table>
Argument Description

bForce Boolean value that determines the behavior if the requested reservation conflicts with an existing reservation.

If set to 0 (false), then the request is rejected if it conflicts with an existing reservation.

If set to 1 (true) and the user running the TurboIntegrator process has the DataReservationOverride capability, then the conflicting reservations are released, and the requested one is granted.

Address Tokenized string sequence of element names that define the tuple. The order must match the original dimension order of the cube. All the cells in the cube contained by the tuple make up the region being reserved. You can choose one element from each dimension or use an empty string between the delimiters to select an entire dimension. Depending on where the element is located in the hierarchy, the request reserves a single cell, a slice, or the entire cube.

AddressDelimiter Optional character string that is used to separate element names in the Address parameter. Default value is '|'.

Return Value

Boolean - returns true if the acquisition succeeded.

Example

CubeDataReservationAcquire('DRTestCube','User1',0,'ElemX|ElemY|ElemZ');

The following example sets the bForce parameter to 1 to force the DR request if a conflict exists and uses a different delimiter character for the AddressDelimiter parameter.

CubeDataReservationAcquire('DRTestCube','User2',1,'ElemX*ElemY*ElemZ','*');

CubeDataReservationRelease

CubeDataReservationRelease releases the specified Data Reservation.

This is a TurboIntegrator function, valid only in TurboIntegrator processes.

If the user specified is not the same as the owner of the reservation, then the release will only succeed if the user specified has the DataReservationOverride capability enabled.

Syntax

CubeDataReservationRelease(Cube, User, Address,[AddressDelimiter])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cube</td>
<td>Name of the cube.</td>
</tr>
</tbody>
</table>
Cube Data Reservation Release

CubeDataReservationRelease('DRTestCube','User1','ElemX|ElemY|ElemZ');

The following example uses a different character for the AddressDelimiter parameter.
CubeDataReservationRelease('DRTestCube','User2','ElemX*ElemY*ElemZ','*');

CubeDataReservationReleaseAll

CubeDataReservationReleaseAll releases multiple existing Data Reservations.

This is a TurboIntegrator function, valid only in TurboIntegrator processes.

All reservations fully contained by the specified address that match the user filter will be released. A blank user filter means all users.

If the user filter specified is not the same as the user running the TurboIntegrator process, then the DataReservationOverride capability must be enabled.

Using a blank user filter and all wildcards in the address field releases all reservations.

Syntax

CubeDataReservationReleaseAll(Cube, UserFilter, Address, [AddressDelimiter])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cube</td>
<td>Name of the cube.</td>
</tr>
<tr>
<td>UserFilter</td>
<td>User name filter to match against existing reservations.</td>
</tr>
<tr>
<td>Address</td>
<td>Tokenized string sequence of element names that define the tuple.</td>
</tr>
<tr>
<td>AddressDelimiter</td>
<td>Optional character string that is used to separate element names in the Address parameter. Default value is 'l'.</td>
</tr>
</tbody>
</table>
**Argument Description**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddressDelimiter</td>
<td>Optional character string that is used to separate element names in the Address parameter. Default value is '</td>
</tr>
</tbody>
</table>

**Return Value**

Boolean - returns true if no errors.

**Example**

CubeDataReservationReleaseAll('DRTestCube','User1','ElemX|ElemY|ElemZ');

The following example releases all reservations in the specified cube for all users.
CubeDataReservationReleaseAll('DRTestCube','','|');

**CubeDataReservationGet**

CubeDataReservationGet finds existing reservations on a specific cube for all or one user.

This is a TurboIntegrator function, valid only in TurboIntegrator processes.

**Syntax**

CubeDataReservationGet(Index, Cube, User, [AddressDelimiter]) returns Address;

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index</td>
<td>A one-based loop index to use for iterating through reservations on the specified cube.</td>
</tr>
<tr>
<td>Cube</td>
<td>Name of the cube to search.</td>
</tr>
<tr>
<td>User</td>
<td>Reservation owner name to use as a filter.</td>
</tr>
<tr>
<td></td>
<td>If left blank, the function returns reservations for any owner.</td>
</tr>
<tr>
<td></td>
<td>If a name is provided, the function filters the results for just the specified owner.</td>
</tr>
<tr>
<td>AddressDelimiter</td>
<td>Optional character string that is used to separate element names in the returned Address parameter. Default value is '</td>
</tr>
</tbody>
</table>

**Return Value**

Address - Reservation creation time, name of the reservation owner and Element address of the reservation. Creation time comes first, followed by delimiter, followed by UserID, followed by delimiter, followed by Elements IDs separated by the delimiter in order of dimensions in the cube (original order).
An empty string is returned if there is no entry for the specified index.

The format of the return value is:

```
[creation time][delimiter][owner name][delimiter][element1][delimiter][element2][delimiter]...[elementN]
```

For example:

```
"20100622211601 | Fred Bloggs | Element1 | Element2 | Element3"
```

**Note:** The reservations can change while iterating the list of reservations so the use of index is not guaranteed to give a complete list of reservations. Reservations can be added or removed at any position in the list, so reservations can be skipped or repeated when looping through index values.

If the owner filter is specified, then the index applies only to the members of the filtered list. If the list of reservations has owners as follows: User1, User1, User2 and the request specifies an owner of User2 then an index of 1 will retrieve the third member of the list.

**Example**

```
CubeDataReservationGet(1,'DRTestCube','User1','*');
CubeDataReservationGet(1,'DRTestCube','');
```

The following sample would find all the reservations owned by user Fred Bloggs in the Expense Input cube and do "something useful" with them:

```
vIndex = 1;
vCube = 'Expense Input';
vUserFilter = 'Fred Bloggs';
vDelim = '|';
vAddress = CubeDataReservationGet( vIndex, vCube, vUserFilter,vDelim);
WHILE (vAddress @<> '');
    vSep1 = SCAN( vDelim, vAddress);
    vDRUser = SUBST( vAddress, 1, vSep1 - 1);
    vDRAddress = SUBST( vAddress, vSep1 + 1, LONG(vDRAddress) - vSep1);
    # do something meaningful with the user and reservation address here
    vIndex = vIndex + 1;
vAddress = CubeDataReservationGet( vIndex, vCube, vUserFilter,vDelim);
END;
```

**CubeDataReservationGetConflicts**

`CubeDataReservationGetConflicts` finds existing reservations on a specific cube that would conflict with the specified user, address and tuple.

This is a TurboIntegrator function, valid only in TurboIntegrator processes.

**Syntax**

```
CubeDataReservationGetConflicts(Index, Cube, User, Address, [AddressDelimiter]) returns ConflictAddress;
```
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index</td>
<td>A one-based loop index to use for iterating through conflicts that satisfy this query.</td>
</tr>
<tr>
<td>Cube</td>
<td>Name of the cube to search</td>
</tr>
<tr>
<td>User</td>
<td>The query will search for reservations that will conflict with this user.</td>
</tr>
<tr>
<td>Address</td>
<td>Tokenized string sequence of element names that define the tuple. The order must match the original dimension order of the cube.</td>
</tr>
<tr>
<td>AddressDelimiter</td>
<td>Optional character string that is used to separate element names in the Address parameter. Default value '</td>
</tr>
</tbody>
</table>

**Return Value**

ConflictAddress - Reservation creation time, name of the reservation owner and Element address of the reservation. The creation time comes first, followed by delimiter, followed by UserID, followed by delimiter, followed by Elements IDs separated by the delimiter in order of dimensions in the cube (original order).

An empty string is returned if there is no entry for the specified index.

The format of the return value is:

```
[creation time][delimiter][owner name][delimiter][element1][delimiter][element2][delimiter]...[elementN]
```

For example:

"20100622211601|Fred Bloggs|Element1|Element2|Element3"

**Note:** The reservations can change while iterating the list of conflict reservations so the use of index is not guaranteed to give a complete list of reservations. Reservations can be added or removed at any position in the list, so reservations can be skipped or repeated when looping through index values.

**Managing Data Reservations with TM1 API functions**

You can use the following TM1 C API functions to programmatically obtain, release and manage Data Reservations.

- TM1DataReservationAcquire
- TM1DataReservationRelease
- TM1DataReservationReleaseAll
- TM1DataReservationGetAll
- TM1DataReservationGetConflicts
- TM1DataReservationValidate
**TM1DataReservationAcquire**

Requests a DR for a specific IBM Cognos TM1 cube, user and tuple.

If there is an existing reservation owned by a different user whose region overlaps the requested reservation, then the reservation request will be rejected unless the bForce flag is used. If the bForce flag is true and the user running the API has the DataReservationOverride capability, then any conflicting reservations will be released and the new reservation will be granted.

**Syntax**

```
TM1DataReservationAcquire(TM1P hPool, TM1V hCube, TM1V hClient, TM1V bForce, TM1V elementArray);
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hPool</td>
<td>Standard memory pool used by all API commands.</td>
</tr>
<tr>
<td>hCube</td>
<td>Handle to the cube you want to access.</td>
</tr>
<tr>
<td>hClient</td>
<td>The owner to use for the reservation</td>
</tr>
<tr>
<td>bForce</td>
<td>Boolean value that determines the behavior if the requested reservation conflicts with an existing reservation. If set to 0 (false), then the request is rejected if it conflicts with an existing reservation. If set to 1 (true), then the function replaces any conflicting reservations.</td>
</tr>
<tr>
<td>elementArray</td>
<td>Array of element handles that define the tuple, the order must match the dimension order.</td>
</tr>
</tbody>
</table>

**Return Value**

Boolean value of true if the request was granted or false otherwise.

**Possible Errors**

- TM1ErrorCubeNumberOfKeysInvalid
- TM1ErrorObjectHandleInvalid
- TM1ErrorCubeKeyInvalid
- TM1ErrorObjectSecurityNoReserveRights

**TM1DataReservationRelease**

Releases an existing DR for a specific IBM Cognos TM1 cube, user and tuple.

The owner used for hClient must match the holder of the DR for the command to succeed unless the user invoking the API has the DataReservationOverride capability enabled.

The addresses supplied must be an exact match.
Syntax

TM1DataReservationRelease(TM1P hPool, TM1V hCube, TM1V hClient,TM1V elementArray);

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hPool</td>
<td>Standard memory pool used by all API commands.</td>
</tr>
<tr>
<td>hCube</td>
<td>Handle to the cube you want to access.</td>
</tr>
<tr>
<td>hClient</td>
<td>The owner of the reservation.</td>
</tr>
<tr>
<td>elementArray</td>
<td>Array of element handles that define the tuple. The order must match the dimension order.</td>
</tr>
</tbody>
</table>

Return Value

Boolean value of true if the request succeeded or false otherwise. Not finding the reservation is a failure and returns false. Insufficient privilege is handled as an error.

Possible Errors

- TM1ErrorCubeNumberOfKeysInvalid
- TM1ErrorObjectHandleInvalid
- TM1ErrorCubeKeyInvalid
- TM1ErrorObjectSecurityNoAdminRights
- TM1ErrorObjectSecurityNoReserveRights

**TM1DataReservationReleaseAll**

Releases multiple Data Reservations for the specified IBM Cognos TM1 user.

The specified address tuple specifies the starting point for the search. All reservations owned by the specified user fully contained within the region defined by the address are released. Any reservation that overlaps the address but is not fully contained is not released.

Specifying a NULL client will remove reservations for all users. If the owner is not the same as the user executing the command, then the user must have the DataReservationOverride capability. Attempts to execute this command for a different user or all users without the override capability will be rejected without searching for existing reservations.

An administrator can release all reservations on a cube by specifying a NULL client and wildcards for every element in the address.

Syntax

TM1DataReservationReleaseAll(TM1P hPool, TM1V hCube, TM1V hClient,TM1V elementArray);
Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hPool</td>
<td>Standard memory pool used by all API commands.</td>
</tr>
<tr>
<td>hCube</td>
<td>Handle to the cube you want to access.</td>
</tr>
<tr>
<td>hClient</td>
<td>The owner of the reservation.</td>
</tr>
<tr>
<td>elementArray</td>
<td>Array of element handles that define the starting point for the release operation. The order must match the dimension order.</td>
</tr>
</tbody>
</table>

**Return Value**

Boolean value of true if there were no errors.

**Possible Errors**

- TM1ErrorCubeNumberOfKeysInvalid
- TM1ErrorObjectHandleInvalid
- TM1ErrorCubeKeyInvalid
- TM1ErrorObjectSecurityNoReserveRights
- TM1ErrorObjectSecurityNoAdminRights

**TM1DataReservationGetAll**

Determines which Data Reservations are currently held on a IBM Cognos TM1 cube.

The client parameter is optional. If it is not supplied (the parameter is set to TM1ObjectNull), then all the DRs on the cube are returned.

If the client parameter is supplied, then only the DRs held by that particular user are returned.

**Syntax**

TM1DataReservationGetAll(TM1P hPool, TM1V hCube, TM1VhClient);

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hPool</td>
<td>Standard memory pool used by all API commands.</td>
</tr>
<tr>
<td>hCube</td>
<td>Handle to the cube you want to access.</td>
</tr>
<tr>
<td>hClient</td>
<td>Optional handle for the user you want to query for.</td>
</tr>
</tbody>
</table>

**Return Value**

Array of DR data with the following format:

- [1] Cube name (TM1ValTypeString)
- [2-n] Array of DR information (TM1ValTypeArray)
  - [1] Creation Time
Possible Errors

TM1ErrorObjectHandleInvalid

**TM1DataReservationGetConflicts**

Determines which reservations currently held on a IBM Cognos TM1 cube will conflict with the specified client (user) and address.

This command can be used to gather the information needed to determine why an attempt to acquire a reservation failed, assuming the reservation that caused the denial is still there.

**Syntax**

```
TM1DataReservationGetConflicts(TM1P hPool, TM1V hCube, TM1V hClient, TM1V elementArray)
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hPool</td>
<td>Standard memory pool used by all API commands</td>
</tr>
<tr>
<td>hCube</td>
<td>Handle to the cube you want to access</td>
</tr>
<tr>
<td>hClient</td>
<td>The client (user) to compare against current reservation owners.</td>
</tr>
<tr>
<td>elementArray</td>
<td>Array of element handles that define the tuple to compare against.</td>
</tr>
<tr>
<td></td>
<td>The order must match the dimension order</td>
</tr>
</tbody>
</table>

**Return Value**

Returns an array of DR data with the following format:

- [1] Cube name (TM1ValTypeString)
- [2-n] Array of DR information (TM1ValTypeArray)
  - [1] Creation Time
  - [2] User name (TM1ValTypeString)
  - [3-n] Array of element names defining the tuple (TM1ValTypeArray)
    - [1-n] Element name (TM1ValTypeString)

**TM1DataReservationValidate**

Validates all the Data Reservations on a IBM Cognos TM1 cube.

Any reservation owned by a client (user) that no longer exists will be removed.
Syntax

TM1DataReservationValidate(TM1PHPool, TM1VhCube);

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hPool</td>
<td>Standard memory pool used by all API commands</td>
</tr>
<tr>
<td>hCube</td>
<td>Handle to the cube we want to access</td>
</tr>
</tbody>
</table>

Return Value

Boolean value of true.

Possible Errors

TM1ErrorObjectNotFound (invalid cube)

API error codes for data reservations

The following table describes the possible error codes that can be returned by the IBM Cognos TM1 C API functions for data reservations.

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TM1ErrorCubeNumberOfKeysInvalid</td>
<td>Number of elements doesn't match the number of cube dimensions.</td>
</tr>
<tr>
<td>TM1ErrorObjectHandleInvalid</td>
<td>Cube, Client, or Element handle does not map to an existing object.</td>
</tr>
<tr>
<td>TM1ErrorCubeKeyInvalid</td>
<td>The element supplied doesn't match an element in the dimension at that position. The element supplied is a UDC.</td>
</tr>
<tr>
<td>TM1ErrorObjectSecurityNoReserveRights</td>
<td>Capability to use reservation is not granted.</td>
</tr>
<tr>
<td>TM1ErrorObjectSecurityNoAdminRights</td>
<td>Attempt to release a reservation when not the owner and without the override capability being granted.</td>
</tr>
</tbody>
</table>
Appendix A. Supported Microsoft Excel Functions - TM1 Web

IBM Cognos TM1 Web supports many Excel worksheet functions.

Supported Functions

This appendix lists the supported Excel functions by category and in alphabetical order, and describes any differences in performance between the Excel functions and TM1 Web functions.

Date and Time Functions

The following table lists date and time functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>Returns the serial number of a particular date.</td>
</tr>
<tr>
<td>DATEVALUE</td>
<td>Converts a date in the form of text to a serial number.</td>
</tr>
<tr>
<td>DAY</td>
<td>Converts a serial number to a day of the month.</td>
</tr>
<tr>
<td>DAYS360</td>
<td>Calculates the number of days between two dates based on a 360-day year.</td>
</tr>
<tr>
<td>HOUR</td>
<td>Converts a serial number to an hour.</td>
</tr>
<tr>
<td>MINUTE</td>
<td>Converts a serial number to a minute.</td>
</tr>
<tr>
<td>MONTH</td>
<td>Converts a serial number to a month.</td>
</tr>
<tr>
<td>NOW</td>
<td>Returns the serial number of the current date and time.</td>
</tr>
<tr>
<td>SECOND</td>
<td>Converts a serial number to a second.</td>
</tr>
<tr>
<td>TIME</td>
<td>Returns the serial number of a particular time.</td>
</tr>
<tr>
<td>TIMEVALUE</td>
<td>Converts a time in the form of text to a serial number.</td>
</tr>
<tr>
<td>TODAY</td>
<td>Returns the serial number of today's date.</td>
</tr>
<tr>
<td>WEEKDAY</td>
<td>Converts a serial number to a day of the week.</td>
</tr>
<tr>
<td>YEAR</td>
<td>Converts a serial number to a year.</td>
</tr>
</tbody>
</table>
## Financial Functions

The following table lists financial functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB</td>
<td>Returns the depreciation of an asset for a specified period using the fixed-declining balance method.</td>
</tr>
<tr>
<td>DDB</td>
<td>Returns the depreciation of an asset for a specified period using the double-declining balance method or some other method you specify.</td>
</tr>
<tr>
<td>FV</td>
<td>Returns the future value of an investment.</td>
</tr>
<tr>
<td>IPMT</td>
<td>Returns the interest payment for an investment for a given period.</td>
</tr>
<tr>
<td>IRR</td>
<td>Returns the internal rate of return for a series of cash flows.</td>
</tr>
<tr>
<td>ISPMT</td>
<td>Calculates the interest paid during a specific period of an investment.</td>
</tr>
<tr>
<td>MIRR</td>
<td>Returns the internal rate of return where positive and negative cash flows are financed at different rates.</td>
</tr>
<tr>
<td>NPER</td>
<td>Returns the number of periods for an investment.</td>
</tr>
<tr>
<td>NPV</td>
<td>Returns the net present value of an investment based on a series of periodic cash flows and a discount rate.</td>
</tr>
<tr>
<td>PMT</td>
<td>Returns the periodic payment for an annuity.</td>
</tr>
<tr>
<td>PPMT</td>
<td>Returns the payment on the principal for an investment for a given period.</td>
</tr>
<tr>
<td>PV</td>
<td>Returns the present value of an investment.</td>
</tr>
<tr>
<td>RATE</td>
<td>Returns the interest rate per period of an annuity.</td>
</tr>
<tr>
<td>SLN</td>
<td>Returns the straight-line depreciation of an asset for one period.</td>
</tr>
<tr>
<td>SYD</td>
<td>Returns the sum-of-years' digits depreciation of an asset for a specified period.</td>
</tr>
</tbody>
</table>

## Information Functions

The following table lists information functions that are supported in TM1 Web.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
</table>
| CELL     | Returns information about the formatting, location, or contents of a cell.  
Support for the Cell function is limited to the following info_types: address, col, row, protect, contents, type. |
## Logical Functions

The following table lists logical functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>Returns TRUE if all its arguments are TRUE.</td>
</tr>
<tr>
<td>FALSE</td>
<td>Returns the logical value FALSE.</td>
</tr>
<tr>
<td>IF</td>
<td>Specifies a logical test to perform.</td>
</tr>
<tr>
<td>NOT</td>
<td>Reverses the logic of its argument.</td>
</tr>
<tr>
<td>OR</td>
<td>Returns TRUE if any argument is TRUE.</td>
</tr>
<tr>
<td>TRUE</td>
<td>Returns the logical value TRUE.</td>
</tr>
</tbody>
</table>

## Lookup and Reference Functions

The following table lists lookup and reference functions.

**Note:** Certain functions, such as LOOKUP and ROWS, may accept two dimensional arrays as arguments. TM1 Web does not support two dimensional arrays. Depending on the data organization and requirements, these functions can still obtain correct values, for example, when the data being retrieved falls in the initial portions of the array. To ensure correct values when working with these functions on TM1 Web you may need to reorganize the input data into repeated functions using one dimensional arrays or you may need to use direct cell references.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDRESS</td>
<td>Returns a reference as text to a single cell in a worksheet.</td>
</tr>
<tr>
<td>CHOOSE</td>
<td>Chooses a value from a list of values.</td>
</tr>
<tr>
<td>COLUMN</td>
<td>Returns the column number of a reference.</td>
</tr>
<tr>
<td>COLUMNS</td>
<td>Returns the number of columns in a reference.</td>
</tr>
</tbody>
</table>
Function Description

HLOOKUP Looks in the top row of an array and returns the value of the indicated cell.

HYPERLINK Creates a shortcut or jump that opens a document stored on a network server, an intranet, or the Internet.

INDEX Uses an index to choose a value from a reference or array.

LOOKUP Looks up values in a vector or array.

OFFSET Returns a reference offset from a given reference.

ROW Returns the row number of a reference.

ROWS Returns the number of rows in a reference.

VLOOKUP Looks in the first column of an array and moves across the row to return the value of a cell.

### Math and Trigonometric Functions

The following table lists math and trigonometric functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABS</td>
<td>Returns the absolute value of a number.</td>
</tr>
<tr>
<td>ACOS</td>
<td>Returns the arccosine of a number.</td>
</tr>
<tr>
<td>ACOSH</td>
<td>Returns the inverse hyperbolic cosine of a number.</td>
</tr>
<tr>
<td>ASIN</td>
<td>Returns the arcsine of a number.</td>
</tr>
<tr>
<td>ASINH</td>
<td>Returns the inverse hyperbolic sine of a number.</td>
</tr>
<tr>
<td>ATAN</td>
<td>Returns the arctangent of a number.</td>
</tr>
<tr>
<td>ATAN2</td>
<td>Returns the arctangent from x- and y-coordinates.</td>
</tr>
<tr>
<td>ATANH</td>
<td>Returns the inverse hyperbolic tangent of a number.</td>
</tr>
<tr>
<td>CEILING</td>
<td>Rounds a number to the nearest integer or to the nearest multiple of significance.</td>
</tr>
<tr>
<td>COMBIN</td>
<td>Returns the number of combinations for a given number of objects.</td>
</tr>
<tr>
<td>COS</td>
<td>Returns the cosine of a number.</td>
</tr>
<tr>
<td>COSH</td>
<td>Returns the hyperbolic cosine of a number.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>DEGREES</td>
<td>Converts radians to degrees.</td>
</tr>
<tr>
<td>EVEN</td>
<td>Rounds a number up to the nearest even integer.</td>
</tr>
<tr>
<td>EXP</td>
<td>Returns e raised to the power of a given number.</td>
</tr>
<tr>
<td>FACT</td>
<td>Returns the factorial of a number.</td>
</tr>
<tr>
<td>FLOOR</td>
<td>Rounds a number down, toward zero.</td>
</tr>
<tr>
<td>INT</td>
<td>Rounds a number down to the nearest integer.</td>
</tr>
<tr>
<td>LN</td>
<td>Returns the natural logarithm of a number.</td>
</tr>
<tr>
<td>LOG</td>
<td>Returns the logarithm of a number to a specified base.</td>
</tr>
<tr>
<td>LOG10</td>
<td>Returns the base-10 logarithm of a number.</td>
</tr>
<tr>
<td>MOD</td>
<td>Returns the remainder from division.</td>
</tr>
<tr>
<td>ODD</td>
<td>Rounds a number up to the nearest odd integer.</td>
</tr>
<tr>
<td>PI</td>
<td>Returns the value of pi.</td>
</tr>
<tr>
<td>POWER</td>
<td>Returns the result of a number raised to a power.</td>
</tr>
<tr>
<td>PRODUCT</td>
<td>Multiplies its arguments.</td>
</tr>
<tr>
<td>RADIANS</td>
<td>Converts degrees to radians.</td>
</tr>
<tr>
<td>RAND</td>
<td>Returns a random number between 0 and 1.</td>
</tr>
<tr>
<td>ROMAN</td>
<td>Converts an arabic numeral to roman, as text.</td>
</tr>
<tr>
<td>ROUND</td>
<td>Rounds a number to a specified number of digits.</td>
</tr>
<tr>
<td>ROUNDDOWN</td>
<td>Rounds a number down, toward zero.</td>
</tr>
<tr>
<td>ROUNDUP</td>
<td>Rounds a number up, away from zero.</td>
</tr>
<tr>
<td>SIGN</td>
<td>Returns the sign of a number.</td>
</tr>
<tr>
<td>SIN</td>
<td>Returns the sine of the given angle.</td>
</tr>
<tr>
<td>SINH</td>
<td>Returns the hyperbolic sine of a number.</td>
</tr>
<tr>
<td>SQRT</td>
<td>Returns a positive square root.</td>
</tr>
<tr>
<td>SUM</td>
<td>Adds its arguments.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>SUMIF</td>
<td>Adds the cells specified by a given criteria.</td>
</tr>
<tr>
<td>TAN</td>
<td>Returns the tangent of a number.</td>
</tr>
<tr>
<td>TANH</td>
<td>Returns the hyperbolic tangent of a number.</td>
</tr>
</tbody>
</table>

## Text and Data Functions

The following table lists text and data functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>Returns the character specified by the code number.</td>
</tr>
<tr>
<td>CLEAN</td>
<td>Removes all nonprintable characters from text.</td>
</tr>
<tr>
<td>CODE</td>
<td>Returns a numeric code for the first character in a text string.</td>
</tr>
<tr>
<td>CONCATENATE</td>
<td>Joins several text items into one text item.</td>
</tr>
<tr>
<td>DOLLAR</td>
<td>Converts a number to text, using the $ (dollar) currency format.</td>
</tr>
<tr>
<td>EXACT</td>
<td>Checks to see if two text values are identical.</td>
</tr>
<tr>
<td>FIND</td>
<td>Finds one text value within another (case-sensitive).</td>
</tr>
<tr>
<td>FIXED</td>
<td>Formats a number as text with a fixed number of decimals.</td>
</tr>
<tr>
<td>LEFT</td>
<td>Returns the leftmost characters from a text value.</td>
</tr>
<tr>
<td>LEN</td>
<td>Returns the number of characters in a text string.</td>
</tr>
<tr>
<td>LOWER</td>
<td>Converts text to lowercase.</td>
</tr>
<tr>
<td>MID</td>
<td>Returns a specific number of characters from a text string starting at the position you specify.</td>
</tr>
<tr>
<td>PROPER</td>
<td>Capitalizes the first letter in each word of a text value.</td>
</tr>
<tr>
<td>REPLACE</td>
<td>Replaces characters within text.</td>
</tr>
<tr>
<td>REPT</td>
<td>Repeats text a given number of times.</td>
</tr>
<tr>
<td>RIGHT</td>
<td>Returns the rightmost characters from a text value.</td>
</tr>
<tr>
<td>SEARCH</td>
<td>Finds one text value within another (not case-sensitive).</td>
</tr>
<tr>
<td>SUBSTITUTE</td>
<td>Substitutes new text for old text in a text string.</td>
</tr>
<tr>
<td>T</td>
<td>Converts its arguments to text.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>TEXT</td>
<td>Formats a number and converts it to text.</td>
</tr>
<tr>
<td>TRIM</td>
<td>Removes spaces from text.</td>
</tr>
<tr>
<td>UPPER</td>
<td>Converts text to uppercase.</td>
</tr>
<tr>
<td>VALUE</td>
<td>Converts a text argument to a number.</td>
</tr>
</tbody>
</table>

### Statistical Functions

The following table lists statistical functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVEDEV</td>
<td>Returns the average of the absolute deviations of data points from their mean.</td>
</tr>
<tr>
<td>AVERAGE</td>
<td>Returns the average of its arguments.</td>
</tr>
<tr>
<td>AVERAGEA</td>
<td>Returns the average of its arguments, including numbers, text, and logical values.</td>
</tr>
<tr>
<td>BINOMDIST</td>
<td>Returns the individual term binomial distribution probability.</td>
</tr>
<tr>
<td>CONFIDENCE</td>
<td>Returns the confidence interval for a population mean.</td>
</tr>
<tr>
<td>CORREL</td>
<td>Returns the correlation coefficient between two data sets.</td>
</tr>
<tr>
<td>COUNT</td>
<td>Counts how many numbers are in the list of arguments.</td>
</tr>
<tr>
<td>COUNTA</td>
<td>Counts how many values are in the list of arguments.</td>
</tr>
<tr>
<td>COUNTIF</td>
<td>Counts the number of nonblank cells within a range that meet the given criteria.</td>
</tr>
<tr>
<td>COVAR</td>
<td>Returns covariance, the average of the products of paired deviations.</td>
</tr>
<tr>
<td>DEVSQ</td>
<td>Returns the sum of squares of deviations.</td>
</tr>
<tr>
<td>EXPONDIST</td>
<td>Returns the exponential distribution.</td>
</tr>
<tr>
<td>FISHER</td>
<td>Returns the Fisher transformation.</td>
</tr>
<tr>
<td>FISHERINV</td>
<td>Returns the inverse of the Fisher transformation.</td>
</tr>
<tr>
<td>FORECAST</td>
<td>Returns a value along a linear trend.</td>
</tr>
<tr>
<td>GEOMEAN</td>
<td>Returns the geometric mean.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GROWTH</td>
<td>Returns values along an exponential trend.</td>
</tr>
<tr>
<td>HARMEAN</td>
<td>Returns the harmonic mean.</td>
</tr>
<tr>
<td>INTERCEPT</td>
<td>Returns the intercept of the linear regression line.</td>
</tr>
<tr>
<td>KURT</td>
<td>Returns the kurtosis of a data set.</td>
</tr>
<tr>
<td>LARGE</td>
<td>Returns the k-th largest value in a data set.</td>
</tr>
<tr>
<td>LINEST</td>
<td>Returns the parameters of a linear trend.</td>
</tr>
<tr>
<td>LOGEST</td>
<td>Returns the parameters of an exponential trend.</td>
</tr>
<tr>
<td>MAX</td>
<td>Returns the maximum value in a list of arguments.</td>
</tr>
<tr>
<td>MATCH</td>
<td>Returns the relative position of an item in an array that matches a specified value in a specified order.</td>
</tr>
<tr>
<td>MAXA</td>
<td>Returns the maximum value in a list of arguments, including numbers, text, and logical values.</td>
</tr>
<tr>
<td>MEDIAN</td>
<td>Returns the median of the given numbers.</td>
</tr>
<tr>
<td>MIN</td>
<td>Returns the minimum value in a list of arguments.</td>
</tr>
<tr>
<td>MINA</td>
<td>Returns the smallest value in a list of arguments, including numbers, text, and logical values.</td>
</tr>
<tr>
<td>NEGBINOMDIST</td>
<td>Returns the negative binomial distribution, the probability that there will be Number_f failures before the Number_s-th success, with Probability_f probability of a success.</td>
</tr>
<tr>
<td>MODE</td>
<td>Returns the most common value in a data set.</td>
</tr>
<tr>
<td>NORMDIST</td>
<td>Returns the normal cumulative distribution.</td>
</tr>
<tr>
<td>NORMINV</td>
<td>Returns the inverse of the normal cumulative distribution.</td>
</tr>
<tr>
<td>NORMSDIST</td>
<td>Returns the standard normal cumulative distribution.</td>
</tr>
<tr>
<td>NORMSINV</td>
<td>Returns the inverse of the standard normal cumulative distribution.</td>
</tr>
<tr>
<td>PEARSON</td>
<td>Returns the Pearson product moment correlation coefficient.</td>
</tr>
<tr>
<td>PERMUT</td>
<td>Returns the number of permutations for a given number of objects.</td>
</tr>
<tr>
<td>RSQ</td>
<td>Returns the square of the Pearson product moment correlation coefficient.</td>
</tr>
<tr>
<td>SKEW</td>
<td>Returns the skewness of a distribution.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SLOPE</td>
<td>Returns the slope of the linear regression line.</td>
</tr>
<tr>
<td>SMALL</td>
<td>Returns the k-th smallest value in a data set.</td>
</tr>
<tr>
<td>STANDARDIZE</td>
<td>Returns a normalized value.</td>
</tr>
<tr>
<td>STDEV</td>
<td>Estimates standard deviation based on a sample.</td>
</tr>
<tr>
<td>STDEVA</td>
<td>Estimates standard deviation based on a sample, including numbers, text, and logical values.</td>
</tr>
<tr>
<td>STDEVP</td>
<td>Calculates standard deviation based on the entire population.</td>
</tr>
<tr>
<td>STDEVPA</td>
<td>Calculates standard deviation based on the entire population, including numbers, text, and logical values.</td>
</tr>
<tr>
<td>STEYX</td>
<td>Returns the standard error of the predicted y-value for each x in the regression.</td>
</tr>
<tr>
<td>TREND</td>
<td>Returns values along a linear trend.</td>
</tr>
<tr>
<td>VAR</td>
<td>Estimates variance based on a sample.</td>
</tr>
<tr>
<td>VARA</td>
<td>Estimates variance based on a sample, including numbers, text, and logical values.</td>
</tr>
<tr>
<td>VARP</td>
<td>Calculates variance based on the entire population.</td>
</tr>
<tr>
<td>VARPA</td>
<td>Calculates variance based on the entire population, including numbers, text, and logical values.</td>
</tr>
<tr>
<td>WEIBULL</td>
<td>Returns the Weibull distribution.</td>
</tr>
</tbody>
</table>
Appendix B. Unsupported Microsoft Excel Functions - TM1 Web

IBM Cognos TM1 Web supports many Excel worksheet functions. This appendix lists the Excel functions, by category and in alphabetical order, that are not supported in TM1 Web.

Database and List Management Functions

This table lists the management functions that are not supported in TM1 Web.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAVERAGE</td>
<td>Returns the average of selected database entries.</td>
</tr>
<tr>
<td>DCOUNT</td>
<td>Counts the cells that contain numbers in a database.</td>
</tr>
<tr>
<td>DCOUNTA</td>
<td>Counts nonblank cells in a database.</td>
</tr>
<tr>
<td>DGET</td>
<td>Extracts from a database a single record that matches the specified criteria.</td>
</tr>
<tr>
<td>DMAX</td>
<td>Returns the maximum value from selected database entries.</td>
</tr>
<tr>
<td>DMIN</td>
<td>Returns the minimum value from selected database entries.</td>
</tr>
<tr>
<td>DPPRODUCT</td>
<td>Multiplies the values in a particular field of records that match the criteria in a database.</td>
</tr>
<tr>
<td>DSTDEV</td>
<td>Estimates the standard deviation based on a sample of selected database entries.</td>
</tr>
<tr>
<td>DSTDEVP</td>
<td>Calculates the standard deviation based on the entire population of selected database entries.</td>
</tr>
<tr>
<td>DSUM</td>
<td>Adds the numbers in the field column of records in the database that match the criteria.</td>
</tr>
<tr>
<td>DVAR</td>
<td>Estimates variance based on a sample from selected database entries.</td>
</tr>
<tr>
<td>DVARP</td>
<td>Calculates variance based on the entire population of selected database entries.</td>
</tr>
</tbody>
</table>

Date and Time Functions

This table lists the date and time functions that are not supported in TM1 Web.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDATE</td>
<td>Returns the serial number of the date that is the indicated number of months before or after the start date.</td>
</tr>
<tr>
<td>EOMONTH</td>
<td>Returns the serial number of the last day of the month before or after a specified number of months.</td>
</tr>
<tr>
<td>NETWORKDAYS</td>
<td>Returns the number of whole workdays between two dates.</td>
</tr>
<tr>
<td>WEEKNUM</td>
<td>Converts a serial number to a number representing where the week falls numerically with a year.</td>
</tr>
<tr>
<td>WORKDAY</td>
<td>Returns the serial number of the date before or after a specified number of workdays.</td>
</tr>
</tbody>
</table>
**Function Description**

YEARFRAC Returns the year fraction representing the number of whole days between start_date and end_date.

---

### Financial Functions

This table lists the financial functions that are not supported in TM1 Web.

<table>
<thead>
<tr>
<th>Functions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCINT</td>
<td>Returns the accrued interest for a security that pays periodic interest.</td>
</tr>
<tr>
<td>ACCRINTM</td>
<td>Returns the accrued interest for a security that pays interest at maturity.</td>
</tr>
<tr>
<td>AMORDEGRC</td>
<td>Returns the depreciation for each accounting period by using a depreciation coefficient.</td>
</tr>
<tr>
<td>AMORLINC</td>
<td>Returns the depreciation for each accounting period.</td>
</tr>
<tr>
<td>COUPDAYBS</td>
<td>Returns the number of days from the beginning of the coupon period to the settlement date.</td>
</tr>
<tr>
<td>COUPDAYS</td>
<td>Returns the number of days in the coupon period that contains the settlement date.</td>
</tr>
<tr>
<td>COUPDAYSNC</td>
<td>Returns the number of days from the settlement date to the next coupon date.</td>
</tr>
<tr>
<td>COUPNCD</td>
<td>Returns the next coupon date after the settlement date.</td>
</tr>
<tr>
<td>COUPNUM</td>
<td>Returns the number of coupons payable between the settlement date and maturity date.</td>
</tr>
<tr>
<td>COUPPCD</td>
<td>Returns the previous coupon date before the settlement date.</td>
</tr>
<tr>
<td>CUMIPMT</td>
<td>Returns the cumulative interest paid between two periods.</td>
</tr>
<tr>
<td>CUMPRINC</td>
<td>Returns the cumulative principal paid on a loan between two periods.</td>
</tr>
<tr>
<td>DISC</td>
<td>Returns the discount rate for a security.</td>
</tr>
<tr>
<td>DOLLARDE</td>
<td>Converts a dollar price, expressed as a fraction, into a dollar price, expressed as a decimal number.</td>
</tr>
<tr>
<td>DOLLARFR</td>
<td>Converts a dollar price, expressed as a decimal number, into a dollar price, expressed as a fraction.</td>
</tr>
<tr>
<td>DURATION</td>
<td>Returns the annual duration of a security with periodic interest payments.</td>
</tr>
<tr>
<td>EFFECT</td>
<td>Returns the effective annual interest rate.</td>
</tr>
<tr>
<td>FVSCHEDULE</td>
<td>Returns the future value of an initial principal after applying a series of compound interest rates.</td>
</tr>
<tr>
<td>INTRATE</td>
<td>Returns the interest rate for a fully invested security.</td>
</tr>
<tr>
<td>MDURATION</td>
<td>Returns the Macauley modified duration for a security with an assumed par value of $100.</td>
</tr>
<tr>
<td>NOMINAL</td>
<td>Returns the annual nominal interest rate.</td>
</tr>
<tr>
<td>ODDFPRICE</td>
<td>Returns the price per $100 face value of a security with an odd first period.</td>
</tr>
<tr>
<td>ODDFYIELD</td>
<td>Returns the yield of a security with an odd first period.</td>
</tr>
<tr>
<td>ODDLPRICE</td>
<td>Returns the price per $100 face value of a security with an odd last period.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ODDLYIELD</td>
<td>Returns the yield of a security with an odd last period.</td>
</tr>
<tr>
<td>PRICE</td>
<td>Returns the price per $100 face value of a security that pays periodic interest.</td>
</tr>
<tr>
<td>PRICEDISC</td>
<td>Returns the price per $100 face value of a discounted security.</td>
</tr>
<tr>
<td>PRICEMAT</td>
<td>Returns the price per $100 face value of a security that pays interest at maturity.</td>
</tr>
<tr>
<td>RECEIVED</td>
<td>Returns the amount received at maturity for a fully invested security.</td>
</tr>
<tr>
<td>TBILLEQ</td>
<td>Returns the bond-equivalent yield for a Treasury bill.</td>
</tr>
<tr>
<td>TBILLPRICE</td>
<td>Returns the price per $100 face value for a Treasury bill.</td>
</tr>
<tr>
<td>TBILLYIELD</td>
<td>Returns the yield for a Treasury bill.</td>
</tr>
<tr>
<td>VDB</td>
<td>Returns the depreciation of an asset for a specified or partial period using a declining balance method.</td>
</tr>
<tr>
<td>XIRR</td>
<td>Returns the internal rate of return for a schedule of cash flows that is not necessarily periodic.</td>
</tr>
<tr>
<td>XNPV</td>
<td>Returns the net present value for a schedule of cash flows that is not necessarily periodic.</td>
</tr>
<tr>
<td>YIELD</td>
<td>Returns the yield on a security that pays periodic interest.</td>
</tr>
<tr>
<td>YIELDDISC</td>
<td>Returns the annual yield for a discounted security; for example, a Treasury bill.</td>
</tr>
<tr>
<td>YIELDMAT</td>
<td>Returns the annual yield of a security that pays interest at maturity.</td>
</tr>
</tbody>
</table>

**Information Functions**

This table lists the information functions that are not supported in TM1 Web.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERROR.TYPE</td>
<td>Returns a number corresponding to an error type.</td>
</tr>
<tr>
<td>INFO</td>
<td>Returns information about the current operating environment.</td>
</tr>
<tr>
<td>ISBLANK</td>
<td>Returns TRUE if the value is blank.</td>
</tr>
<tr>
<td>ISEVEN</td>
<td>Returns TRUE if the number is even.</td>
</tr>
<tr>
<td>ISLOGICAL</td>
<td>Returns TRUE if the value is a logical value.</td>
</tr>
<tr>
<td>ISNONTEXT</td>
<td>Returns TRUE if the value is not text.</td>
</tr>
<tr>
<td>ISNUMBER</td>
<td>Returns TRUE if the value is a number.</td>
</tr>
<tr>
<td>ISODD</td>
<td>Returns TRUE if the number is odd.</td>
</tr>
<tr>
<td>ISREF</td>
<td>Returns TRUE if the value is a reference.</td>
</tr>
<tr>
<td>ITEXT</td>
<td>Returns TRUE if the value is text.</td>
</tr>
</tbody>
</table>
### Function Description

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Returns a value converted to a number.</td>
</tr>
<tr>
<td>TYPE</td>
<td>Returns a number indicating the data type of a value.</td>
</tr>
</tbody>
</table>

### Lookup and Reference Functions

This table lists the lookup and reference functions that are not supported in TM1 Web.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AREAS</td>
<td>Returns the number of areas in a reference.</td>
</tr>
<tr>
<td>INDIRECT</td>
<td>Returns a reference indicated by a text value.</td>
</tr>
<tr>
<td>MATCH</td>
<td>Looks up values in a reference or array.</td>
</tr>
<tr>
<td>RTD</td>
<td>Retrieves real-time data from a program that supports COM automation.</td>
</tr>
<tr>
<td>TRANSPOSE</td>
<td>Returns the transpose of an array.</td>
</tr>
</tbody>
</table>

### Math and Trigonometric Functions

This table lists the math and trigonometric functions that are not supported in TM1 Web.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FACTDOUBLE</td>
<td>Returns the double factorial of a number.</td>
</tr>
<tr>
<td>GCD</td>
<td>Returns the greatest common divisor.</td>
</tr>
<tr>
<td>LCM</td>
<td>Returns the least common multiple.</td>
</tr>
<tr>
<td>MDETERM</td>
<td>Returns the matrix determinant of an array.</td>
</tr>
<tr>
<td>MINVERSE</td>
<td>Returns the matrix inverse of an array.</td>
</tr>
<tr>
<td>MMULT</td>
<td>Returns the matrix product of two arrays.</td>
</tr>
<tr>
<td>MROUND</td>
<td>Returns a number rounded to the desired multiple.</td>
</tr>
<tr>
<td>MULTINOMIAL</td>
<td>Returns the multinomial of a set of numbers.</td>
</tr>
<tr>
<td>QUOTIENT</td>
<td>Returns the integer portion of a division.</td>
</tr>
<tr>
<td>RANDBETWEEN</td>
<td>Returns a random number between the numbers you specify.</td>
</tr>
<tr>
<td>SERIESSUM</td>
<td>Returns the sum of a power series based on the formula.</td>
</tr>
<tr>
<td>SQRTPI</td>
<td>Returns the square root of (number * pi).</td>
</tr>
<tr>
<td>SUBTOTAL</td>
<td>Returns a subtotal in a list or database.</td>
</tr>
<tr>
<td>SUMPRODUCT</td>
<td>Returns the sum of the products of corresponding array components.</td>
</tr>
<tr>
<td>SUMSQ</td>
<td>Returns the sum of the squares of the arguments.</td>
</tr>
<tr>
<td>SUMX2MY2</td>
<td>Returns the sum of the difference of squares of corresponding values in two arrays.</td>
</tr>
<tr>
<td>SUMX2PY2</td>
<td>Returns the sum of the sum of squares of corresponding values in two arrays.</td>
</tr>
</tbody>
</table>
### Function Description

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUMXMY2</td>
<td>Returns the sum of squares of differences of corresponding values in two arrays.</td>
</tr>
<tr>
<td>TRUNC</td>
<td>Truncates a number to an integer.</td>
</tr>
</tbody>
</table>

## Statistical Functions

This table lists the statistical functions that are not supported in TM1 Web.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BETADIST</td>
<td>Returns the beta cumulative distribution function.</td>
</tr>
<tr>
<td>BETAINV</td>
<td>Returns the inverse of the cumulative distribution function for a specified beta distribution.</td>
</tr>
<tr>
<td>CHIDIST</td>
<td>Returns the one-tailed probability of the chi-squared distribution.</td>
</tr>
<tr>
<td>CHIINV</td>
<td>Returns the inverse of the one-tailed probability of the chi-squared distribution.</td>
</tr>
<tr>
<td>CHITEST</td>
<td>Returns the test for independence.</td>
</tr>
<tr>
<td>COUNTBLANK</td>
<td>Counts the number of blank cells within a range.</td>
</tr>
<tr>
<td>CRITBINOM</td>
<td>Returns the smallest value for which the cumulative binomial distribution is less than or equal to a criterion value.</td>
</tr>
<tr>
<td>FDIST</td>
<td>Returns the F probability distribution.</td>
</tr>
<tr>
<td>FINV</td>
<td>Returns the inverse of the F probability distribution.</td>
</tr>
<tr>
<td>FREQUENCY</td>
<td>Returns a frequency distribution as a vertical array.</td>
</tr>
<tr>
<td>FTEST</td>
<td>Returns the result of an F-test.</td>
</tr>
<tr>
<td>GAMMADIST</td>
<td>Returns the gamma distribution.</td>
</tr>
<tr>
<td>GAMMAINV</td>
<td>Returns the inverse of the gamma cumulative distribution.</td>
</tr>
<tr>
<td>GAMMALN</td>
<td>Returns the natural logarithm of the gamma function, G(x).</td>
</tr>
<tr>
<td>HYPGEOMDIST</td>
<td>Returns the hyper geometric distribution.</td>
</tr>
<tr>
<td>LOGINV</td>
<td>Returns the inverse of the lognormal distribution.</td>
</tr>
<tr>
<td>LOGNORMDIST</td>
<td>Returns the cumulative lognormal distribution.</td>
</tr>
<tr>
<td>NEGBINOMDIST</td>
<td>Returns the negative binomial distribution.</td>
</tr>
<tr>
<td>PERCENTILE</td>
<td>Returns the k-th percentile of values in a range.</td>
</tr>
<tr>
<td>PERCENTRANK</td>
<td>Returns the percentage rank of a value in a data set.</td>
</tr>
<tr>
<td>POISSON</td>
<td>Returns the Poisson distribution.</td>
</tr>
<tr>
<td>PROB</td>
<td>Returns the probability that values in a range are between two limits.</td>
</tr>
<tr>
<td>QUARTILE</td>
<td>Returns the quartile of a data set.</td>
</tr>
<tr>
<td>RANK</td>
<td>Returns the rank of a number in a list of numbers.</td>
</tr>
<tr>
<td>TDIST</td>
<td>Returns the Student's t-distribution.</td>
</tr>
<tr>
<td>TINV</td>
<td>Returns the inverse of the Student's t-distribution.</td>
</tr>
<tr>
<td>TRIMMEAN</td>
<td>Returns the mean of the interior of a data set.</td>
</tr>
<tr>
<td>TTEST</td>
<td>Returns the probability associated with a Student's t-test.</td>
</tr>
<tr>
<td>ZTEST</td>
<td>Returns the one-tailed probability-value of a z-test.</td>
</tr>
</tbody>
</table>
# Text and Data Functions

This table lists the text and data functions that are not supported in TM1 Web.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASC</td>
<td>Changes full-width (double-byte) English letters or katakana within a character string to half-width (single-byte) characters.</td>
</tr>
<tr>
<td>BAHTTEXT</td>
<td>Converts a number to text, using the ฿ (baht) currency format.</td>
</tr>
<tr>
<td>JIS</td>
<td>Changes half-width (single-byte) English letters or katakana within a character string to full-width (double-byte) characters.</td>
</tr>
<tr>
<td>PHONETIC</td>
<td>Extracts the phonetic (furigana) characters from a text string.</td>
</tr>
<tr>
<td>AutoShapes</td>
<td>TM1 Web does not support Microsoft Office Autoshapes.</td>
</tr>
</tbody>
</table>
Notices

This information was developed for products and services offered worldwide.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. This document may describe products, services, or features that are not included in the Program or license entitlement that you have purchased.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.
IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Software Group
Attention: Licensing
3755 Riverside Dr
Ottawa, ON K1V 1B7
Canada

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.
Trademarks

IBM, the IBM logo and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

The following terms are trademarks or registered trademarks of other companies:

• Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

• Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Microsoft product screen shot(s) used with permission from Microsoft.
Index

Special characters
/ (forward slash) in rules 44
\ (backslash) in rules 44
% (percent sign) in rules 44
\ (pipe) in rules 44
& (ampersand) in rules 44
}Externals subdirectory
   Excel files 79
- (tilde) in rules 44

A
access privileges
   admin 103
   assigning to chores 114
   assigning to cubes 108
   assigning to dimensions 111
   assigning to elements 109
   assigning to processes 112
   consolidated elements 110
   interactions 107
   lock 104, 106
   none 107
   read 105, 107
   replication rights 40
   reserve 106
   write 106
Action button
   adding to a worksheet 119
   advanced navigation and mapping options 128
   appearance properties 128
   modifying 132
   moving and resizing 132
   navigating to another worksheet 124
   overview 119
   running a process and navigating to a worksheet 126
aggregate
   rule-aware 12
aliases
   Dimension Editor 24
   element attributes 13
   samples 14
application
   admin access privilege 104
   none access privilege 107
   read access privilege 105
   security privileges 114
applications
   adding file references 79
   adding object references 77
   converting from private to public 87
   converting from public to private 88
   creating and managing 74
   creating secondary-level 76
   deleting references 85
   location on server 72
   overview 71
   private 87, 88
   privatizing public 88
   privatizing public applications 88
applications (continued)
   public 87, 88
   publishing 74, 87
   publishing private references 88
   publishing to Web client 90
   references to private subsets 78
   references to private views 78
   renaming references 85
security privileges 73, 86
top-level 74
viewing logical groupings 89
working with references 83
Area (rule target) 42
assigning security rights 103
asymmetrical cubes
   linking 59
attributes
   examples 13
   format 14
   overview 13
   versus elements 16

B
bypassing rules statements 48

C
calculating averages
   rules statements 58
cell-level security
   consolidations 115
   leaf elements 115
   security control cube 115
child
   adding to consolidated elements 18
   chore
      none access privilege 107
      read access privilege 105
      security 114
circular references (rules) 51
comparison operators
   rules formulas 44
conditional logic
   rules formulas 44
consolidated elements 110
   adding children 18
   defined 13
   defining with dimension worksheets 28
   deleting elements 19
consolidations
   creating with weight factors 10
   multiple paths within a single dimension 11
   versus rules 12
constants
   rules formulas 43
control cubes
   }PickList 36
cube
   admin access privilege 103
cube (continued)
  as lookup tables 97
  assigning access privileges 108
  creating 32
  creating cube dialog box 32
  creating with external data sources 31
  creating with no data 31
  DB rules function for sharing data 54
  DBS formula values 100
  display value formats 14
  examples 7, 9
  linking asymmetrical 59
  loading on demand 34
  lock access privilege 104
  locking 109
  mirror 39
  none access privilege 107
  optimizing 33
  ordering dimensions 31
  properties 32
  read access privilege 105
  releasing 109
  reserve access privilege 106
 -reserving 109
  retrieving values using DBR formulas 99
  rules formulas references 45
  security 111
  selecting dimensions 8
  source 39
  unlocking 109
  write access privilege 106
CubeDataReservationAcquire 145
CubeDataReservationGet 148
CubeDataReservationGetConflicts 149
CubeDataReservationRelease 146
CubeDataReservationReleaseAll 147

Data Reservation TurboIntegrator functions
  CubeDataReservationAcquire 145
  CubeDataReservationGet 148
  CubeDataReservationGetConflicts 149
  CubeDataReservationRelease 146
  CubeDataReservationReleaseAll 147
DataReservationMode property 141
DB rules function 46
DBS formulas
  creating 100
  overview 94
  debugging rules
    checking FEEDERS 69
    overview 66
    tracing calculations 66
    tracing FEEDERS 68
  deleting
    consolidated elements 19
    elements from dimensions 19
    references from applications 85
  dense dimensions 31
  dialog boxes
    Creating Cube 32
  Dimension Editor
    selecting elements 22
    sorting elements 23, 24
    using aliases 24
  dimensions
    adding elements 18
    adding siblings 18
    admin access privilege 104
    admin group 111
    assigning access privileges 111, 112, 114
    consolidations 12, 50
    creating 16
    creating with Dimension Editor 17
    deleting elements 19
    dense 31
    editing 18
    hierarchies 9, 28
    lock access privilege 104
    locking 112
    methods for creating 16
    named levels 29
    none access privilege 107
    ordering in cubes 31
    read access privilege 105
    releasing 112
    reserve access privilege 106
   -reserving 112
    selecting in a cube 8
    setting order of elements 20, 22
    sparse 31
    unlocking 112
    worksheets 26, 29
    write access privilege 106
  display formats
    cube values 14
    setting 15
  drill
    -through processes 60, 62, 63, 64
    down 10

E
  element attributes
  Picklist 36
elements
  adding siblings 18
  adding to dimension 17, 18
  admin access privilege 103
  assigning access privileges 109
  changing position within hierarchy 19
  consolidated 13, 110
  deleting consolidated 19
  deleting from dimension 19
  lock access privilege 104
  locking 111
  none access privilege 107
  numeric 13
  order in dimensions 20
  read access privilege 105
  releasing 110
  reserve access privilege 106
  reserving 110
  security 111
  select all visible elements 19
  selecting in Dimension Editor 22
  setting order from Server Explorer 22
  setting order in dimensions 20, 22
  sorting in Dimension Editor 22
  string 13
  types 13
  unlocking 111
  versus attributes 16
  weight factors 10
  write access privilege 106
Excel
  jExterals subdirectory 79
  file references 83, 85
  worksheet functions 157
  worksheet functions unsupported 167

F
  functions
    date and time 157
    DB 46, 54
    financial 158
    IF 44
    information 158
    logical 159
    lookup and reference 159
    math and trigonometric 160
    statistical 163
    STET 48
    supported Excel worksheet 157
    text and data 162
    unsupported 167, 168, 169, 170, 171, 172

G
  groups
    assigning security rights 103
    dimension 115

H
  hierarchy levels overview 9
  hyperlinks
    opening uploaded application files 90

I
  IF function
    rules formulas 44
  importing data
    processing worksheets 94

L
  level 0 (dimension hierarchies) 9
  linking asymmetrical cubes 59
  loading cubes on demand 34
  lock access privilege 104, 106
  logical operators
    rules formulas 44

M
  mirror cubes 39
  multidimensionality
    diagram 2
    example 1
    overview 1
  multiple consolidation paths 11

N
  named hierarchy levels 29
  naming conventions
    TM1 objects 3
  none access privilege 107
  numeric
    constant in rules formulas 43
    define elements 13

O
  object references
    default actions 83
    deleting from applications 85
    renaming 85
  objects
    display in Server Explorer 89
    OLAP 34
    OLE DB 34
  operators (rules)
    arithmetic 44
    comparison 44
    logical 44
    optimizing cubes 33

P
  pick lists
    creating 34
    creating with control cubes 36
    creating with element attributes 36
    defining with rules 37
    dimension 36
    order of precedence 39
    static 35
    subset 35
    types 35
  populating cubes
    DBS formulas 100
private
  applications 87
  references in applications 88
process
  modifying security data 113
  none access privilege 107
  read access privilege 105
  security 112
public
  applications 87
  references in applications 88
  publishing applications to Web client 90

R
  ratios
    calculating using rules 52
    RDCLS 115
  read access privilege 105, 107
  references
    admin access privilege 104
    converting from private to public 88
    converting from public to private 88
    none access privilege 107
    private 88
    public 88
    read access privilege 105
  renaming application references 85
replcations
  access rights required 40
  advantages 39
  cube relationships 39
  mirror cube 39
  server relationships 39
  source cube 39
  source server 39
  target server 39
reserves access privilege 106
reserved characters
  object names 3
  rolling up 10
  rule-aware
rules
  aggregate 12
  debugging 66
  deriving values 41
  file names 41
  overview 41
  pick lists 37
  Rules Tracer 66
  versus consolidations 12
rules formulas
  arithmetic operators 44
  comparing expressions 44
  comparison operators 44
  conditional logic 44
  cube references 45
  external cube references 46
  internal cube references 45
  logical operators 44
  numeric constants 43
  syntax 43
rules statements (continued)
  calculating ratios 52
  circular references 51
  consolidated and numeric elements 47
  DB function 54
  DB rules function for sharing cube data 54
  Formula 42
  guidelines 42
  linking asymmetrical cubes 59
  order of calculation 49
  overriding dimension consolidations 50
  precedence over dimension consolidations 49
  restricting scope 48
  restricting to simple values 56
  sample 53
  sample applications 52
  specifying target 42
  STET function 48
  syntax 42, 53
Rules Tracer
  checking FEEDERS 69
  complex example 67
  overview 66
  simple example 67
  tracing calculations 66
  tracing FEEDERS 68
rules worksheets
  creating 49
  file names 48
  overview 48
  saving 49
  using 49

S
security
  access privileges 103
  applications 114
  applying privileges to applications 86, 114
  cell-level 115
  chores 114
  cube-level 108, 111
  dimension-level 111
  element-level 109, 111
  group 111
  group rights 103
  processes 112
  references 114
  rights interaction 107
Security Access option 113
security control cube 115
server
  source 39
  target 39
Server Explorer
  setting element order 22
  suppressing display of selected objects 89
sharing data between cubes 54
siblings
  adding to dimensions 18
sorting elements
  alphabetically in Dimension Editor 23
  by hierarchy in Dimension Editor 24
  by index in Dimension 23
  Dimension Editor 22
source cubes 39
source server 39
sparse dimensions 31
STET rules function 48
string elements
defined 13
supported Excel functions
ABS 160
ACOS 160
ACOSH 160
ADDRESS 159
AND 159
ASIN 160
ASINH 160
ATAN 160
ATAN2 160
ATANH 160
AVEDEV 163
AVERAGE 163
AVERAGEA 163
BINOMDIST 163
CEILING 160
CELL 158
CHAR 162
CHOOSE 159
CLEAN 162
CODE 162
COLUMN 159
COLUMNS 159
COMBIN 160
CONCATENATE 162
CONFIDENCE 163
CORREL 163
COS 160
COSH 160
COUNT 163
COUNTA 163
COUNTIF 163
COVAR 163
DATE 157
DATEVALUE 157
DAY 157
DAYS360 157
DB 158
DDB 158
DEGREES 161
DEVSQ 163
DOLLAR 162
EVEN 161
EXACT 162
EXP 161
EXPONDIST 163
FACT 161
FALSE 159
FIND 162
FISHER 163
FISHERINV 163
FIXED 162
FLOOR 161
FORECAST 163
FV 158
GEOMEAN 163
GROWTH 164
HARMEAN 164
HLOOKUP 160
HOUR 157
HYPERLINK 160
IF 159
INDEX 160
supported Excel functions (continued)
INT 161
INTERCEPT 164
IPMT 158
IRR 158
ISERR 159
ISERROR 159
ISNA 159
ISPMT 158
KURT 164
LARGE 164
LEFT 162
LEN 162
LINEST 164
LN 161
LOG 161
LOG10 161
LOGEST 164
LOOKUP 160
LOWER 162
MATCH 164
MAX 164
MAXA 164
MEDIAN 164
MID 162
MIN 164
MINA 164
MINUTE 157
MIRR 158
MOD 161
MODE 164
MONTH 157
MORMINV 164
NA 159
NEGBINOMDIST 164
NORMDIST 164
NORMSDIST 164
NORMSINV 164
NOT 159
NOW 157
NPV 158
NPV 158
ODD 161
OFFSET 160
OR 159
PEARSON 164
PERMUT 164
PI 161
PMT 158
POWER 161
PPMT 158
PRODUCT 161
PROPER 162
PV 158
RADIANS 161
RAND 161
RANDBETWEEN 164
RANK 158
RANK 158
RANKA 164
RANDBETWEEN 164
RANGD 161
RANDBETWEEN 161
ROWS 160
ROWS 160
RSQ 164
Index 181
supported Excel functions (continued)
SEARCH 162
SECOND 157
SIGN 161
SIN 161
SINH 161
SKEW 164
SLN 158
SLOPE 165
SMALL 165
SQRT 161
STANDARDDIZE 165
STDEV 165
STDEVA 165
STDEVPA 165
STEYX 165
SUBSTITUTE 162
SUM 161
SUMIF 162
SYD 158
T 162
TAN 162
TANH 162
TEXT 163
TIME 157
TIMEVALUE 157
TODAY 157
TREND 165
TRIM 163
TRUE 159
UPPER 163
VALUE 163
VAR 165
VARA 165
VARP 165
VARPA 165
VLOOKUP 160
WEEKDAY 157
WEBULL 165
YEAR 157

syntax
rules formulas 43

U
unsupported Excel functions
ACCRINT 168
ACCRINTM 168
AMORDEGRC 168
AMORLINC 168
AREAS 170
ASC 172
BAHTTEXT 172

unsupported Excel functions (continued)
BETADIST 171
BETAINV 171
CHIDIST 171
CHIINV 171
CHITEST 171
COUNTBLANK 171
COUPDAYBS 168
COUPDAYS 168
COUPDAYSNC 168
COUPNCD 168
COUPNUM 168
COUPPCD 168
CRITBINOM 171
CUMIPMT 168
CUMPRINC 168
DAVERAGE 167
dCOUNT 167
dCOUNTA 167
dGET 167
disc 168
dMAX 167
dMIN 167
dOLLARDE 168
dOLLARFR 168
dPRODUCT 167
dSTDEV 167
dSTDEVP 167
dSUM 167
dURATION 168
dVAR 167
dVARP 167
dATE 167
effect 168
eOMONTH 167
error.type 169
FACTDOUBLE 170
FDIST 171
FINV 171
FREQUENCY 171
FTEST 171
FVSCHEDULE 168
GAMMADIST 171
GAMMAINV 171
GAMMALN 171
GCD 170
HYPGEOMDIST 171
INDIRECT 170
INFO 169
INTRATE 168
ISBLANK 169
ISEVEN 169
ISLOGICAL 169
ISNONTEXT 169
ISNUMBER 169
ISODD 169
ISREF 169
ISTEXT 169
JIS 172
LCM 170
LOGINV 171
LOGNORMDIST 171
MATCH 170
MEDIAN 170
MDURATION 168
MINVERSE 170
MINVER 170
MMULT 170
unsupported Excel functions (continued)
MROUND 170
MULTINOMIAL 170
N 170
NEGBINOMDIST 171
NETWORKDAYS 167
NOMINAL 168
ODDFPRICE 168
ODDFYIELD 168
ODDLPRICE 168
ODDLYIELD 169
PERCENTILE 171
PERCENTRANK 171
PHONETIC 172
POISSON 171
PRICE 169
PRICEDISC 169
PRICEMAT 169
PROB 171
QUARTILE 171
QUOTIENT 170
RANDBETWEEN 170
RANK 171
RECEIVED 169
RTD 170
SERIESSUM 170
SQRTPI 170
SUBTOTAL 170
SUMPRODUCT 170
SUMSQ 170
SUMX2MY2 170
SUMX2PY2 170
SUMXMY2 171
TBILLEQ 169
TBILLPRICE 169
TBILLYIELD 169
TDIST 171

unsupported Excel functions (continued)
TINV 171
TRANSPOSE 170
TRIMMEAN 171
TRUNC 171
TTEST 171
TYPE 170
VDB 169
WEEKNUM 167
WORKDAY 167
XIRR 169
XNPV 169
YEARFRAC 167
YIELD 169
YIELDDISC 169
YIELDMAT 169
ZTEST 171

V
values
deriving with rules 41

W
Websheet objects
hyperlinks to uploaded application files 90
Websheets
setting properties 90
weight factors
consolidations 10
windows
Creating Cube 32
Dimension Editor 17
write access privilege 106