Getting Started with DeepSee

Version 2018.1
2020-11-13

InterSystems Corporation 1 Memorial Drive Cambridge MA 02142 www.intersystems.com

Getting Started with DeepSee
Caché Version 2018.1 2020-11-13
Copyright © 2020 InterSystems Corporation
All rights reserved.

InterSystems, InterSystems IRIS, InterSystems Caché, InterSystems Ensemble, and InterSystems HealthShare are registered trademarks of InterSystems Corporation.

All other brand or product names used herein are trademarks or registered trademarks of their respective companies or organizations.

This document contains trade secret and confidential information which is the property of InterSystems Corporation, One Memorial Drive, Cambridge, MA 02142, or its affiliates, and is furnished for the sole purpose of the operation and maintenance of the products of InterSystems Corporation. No part of this publication is to be used for any other purpose, and this publication is not to be reproduced, copied, disclosed, transmitted, stored in a retrieval system or translated into any human or computer language, in any form, by any means, in whole or in part, without the express prior written consent of InterSystems Corporation.

The copying, use and disposition of this document and the software programs described herein is prohibited except to the limited extent set forth in the standard software license agreement(s) of InterSystems Corporation covering such programs and related documentation. InterSystems Corporation makes no representations and warranties concerning such software programs other than those set forth in such standard software license agreement(s). In addition, the liability of InterSystems Corporation for any losses or damages relating to or arising out of the use of such software programs is limited in the manner set forth in such standard software license agreement(s).

THE FOREGOING IS A GENERAL SUMMARY OF THE RESTRICTIONS AND LIMITATIONS IMPOSED BY INTERSYSTEMS CORPORATION ON THE USE OF, AND LIABILITY ARISING FROM, ITS COMPUTER SOFTWARE. FOR COMPLETE INFORMATION REFERENCE SHOULD BE MADE TO THE STANDARD SOFTWARE LICENSE AGREEMENT(S) OF INTERSYSTEMS CORPORATION, COPIES OF WHICH WILL BE MADE AVAILABLE UPON REQUEST.

InterSystems Corporation disclaims responsibility for errors which may appear in this document, and it reserves the right, in its sole discretion and without notice, to make substitutions and modifications in the products and practices described in this document.

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Response Center (WRC)
Tel: +1-617-621-0700
Tel: +44 (0) 844 854 2917
Email: support@InterSystems.com

Table of Contents

About This Book **1**

1 Introduction to DeepSee **3**
- 1.1 Purpose 3
- 1.2 Dashboard Visual Details 4
  - 1.2.1 Pivot Table Widgets 5
  - 1.2.2 Scorecard Widgets 6
  - 1.2.3 Meter Widgets 6
  - 1.2.4 Map Widgets 7
  - 1.2.5 Calendar Widgets 7
  - 1.2.6 Custom Widgets 8
- 1.3 Data Sources for Widgets 8
- 1.4 DeepSee Models 8

2 Introduction to the DeepSee User Interfaces **11**
- 2.1 Logging On to DeepSee 11
- 2.2 Architect 12
- 2.3 Analyzer 13
- 2.4 User Portal 13
- 2.5 Query Tool 14
- 2.6 Term List Manager 15
- 2.7 Listing Group Manager 16
- 2.8 Quality Measure Manager 16
- 2.9 Model Browser 17
- 2.10 DeepSee Logs 18
- 2.11 Folder Manager 19
- 2.12 Settings 19
- 2.13 Cube Manager 20

3 Introduction to the Other DeepSee Tools **23**
- 3.1 MDX Shell 23
  - 3.1.1 Accessing the MDX Shell 23
  - 3.1.2 Viewing the Indices Used by a Query 24
- 3.2 Utility Methods 25
- 3.3 Data Connector 26
- 3.4 Result Set API 26
- 3.5 JavaScript and REST APIs 26

DeepSee Glossary **27**

About This Book

This book briefly introduces DeepSee. It includes the following sections:

- Introduction to DeepSee
- Introduction to the DeepSee User Interfaces
- Introduction to the Other DeepSee Tools
- DeepSee Glossary

For a detailed outline, see the table of contents.

The other developer books for DeepSee are as follows:

- *DeepSee Developer Tutorial* guides developers through the process of creating a sample that consists of a cube, subject areas, pivot tables, and dashboards.
- *DeepSee Implementation Guide* describes how to implement DeepSee, apart from creating the model.
- *Defining DeepSee Models* describes how to define the basic elements used in DeepSee queries: cubes and subject areas. It also describes how to define listing groups.
- *Advanced DeepSee Modeling Guide* describes how to use the more advanced and less common DeepSee modeling features: computed dimensions, unstructured data in cubes, compound cubes, cube relationships, term lists, quality measures, KPIs, plugins, and other special options.
- *Using MDX with DeepSee* introduces MDX and describes how to write MDX queries manually for use with DeepSee cubes.
- *DeepSee MDX Reference* provides reference information on MDX as supported by DeepSee.
- *Tools for Creating DeepSee Web Clients* provides information on the DeepSee JavaScript and REST APIs, which you can use to create web clients for your DeepSee applications.

The following books are for both developers and users:

- *DeepSee End User Guide* describes how to use the DeepSee User Portal and dashboards.
- *Creating DeepSee Dashboards* describes how to create and modify dashboards in DeepSee.
- *Using the DeepSee Analyzer* describes how to create and modify pivot tables, as well as perform ad hoc analysis.

Also see the article *Using PMML Models in Caché*.

For general information, see the *InterSystems Documentation Guide*.

1

Introduction to DeepSee

This chapter introduces DeepSee, which enables you to embed business intelligence (BI) into your applications. This chapter discusses the following topics:

- Purpose of DeepSee
- Introduction to dashboards
- Data sources for dashboards
- DeepSee models

Note: You can use DeepSee only in a specifically configured web application. See "Setting Up the Web Application" in the *DeepSee Implementation Guide*. Also, be sure to consult the online *InterSystems Supported Platforms* document for this release for information on system requirements.

1.1 Purpose

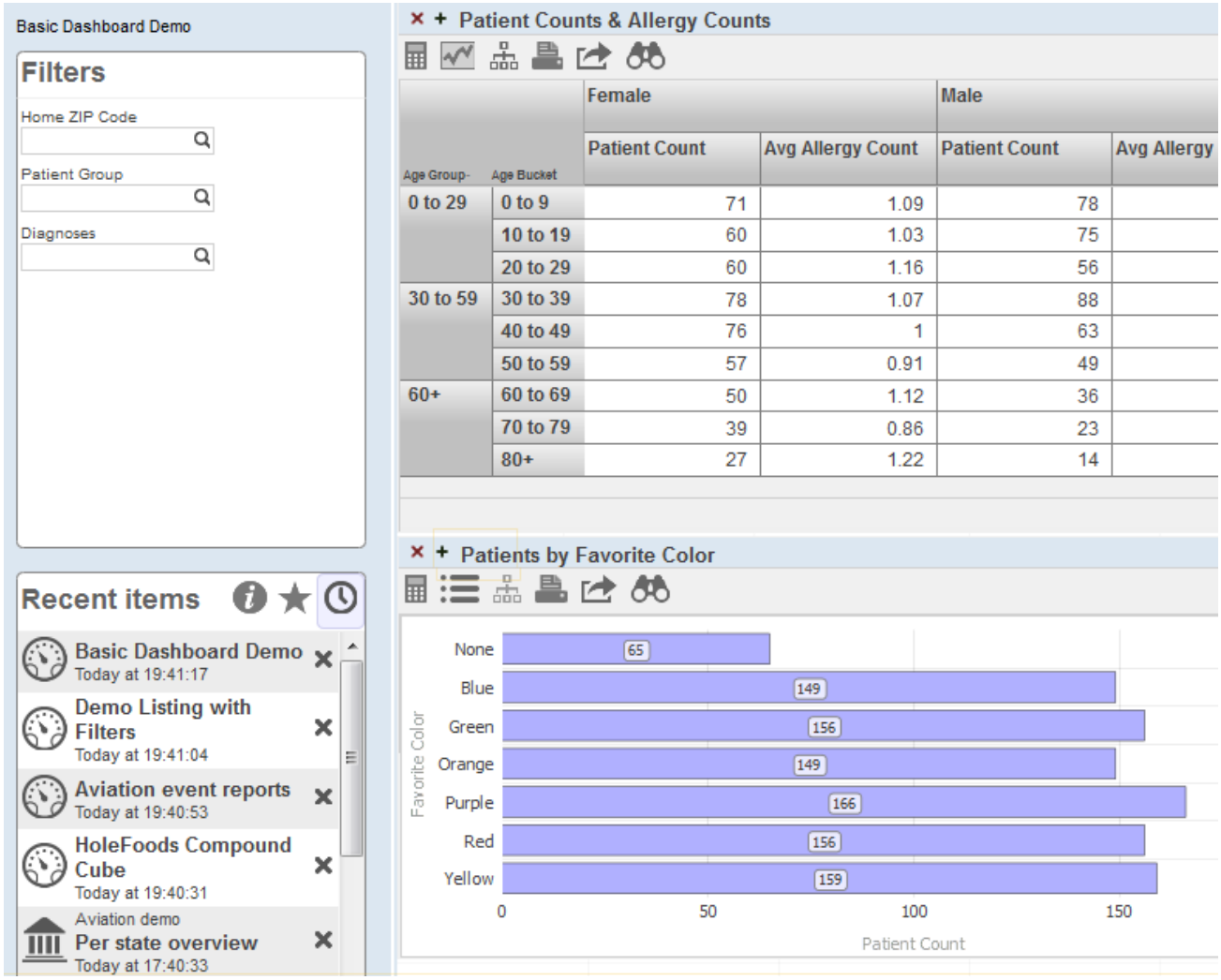
The purpose of InterSystems DeepSee is to enable you to embed business intelligence (BI) into your applications so that your users can ask and answer sophisticated questions of their data. Specifically, your application can include *dashboards*, which can include the following:

- Interactive widgets that execute queries designed for specific user roles or for specific areas of your user interface.
- Controls such as drop-down lists and data choosers that enable users to modify these queries.
- Interactive drill options that enable users to view the displayed data in different ways.
- Options to export, print, and send alerts to other users.
- An option to launch the Mini Analyzer, which supports free-form analysis.
- Execute custom actions that are provided as buttons or other controls.

In contrast to traditional BI systems that use static data warehouses, DeepSee is kept closely in synchronization with the live transactional data, as required for your business.

1.2 Dashboard Visual Details

The following example shows a sample dashboard:



A dashboard consists of the following areas:

- The upper left displays the name of the dashboard and (if defined) its title.
- Depending on the system configuration and on the individual layout of a dashboard, a dashboard can include zero, one, or two worklist areas on the left. For any worklist area, the upper right corner displays icons to indicate which worklists it can display. For example:

  The highlighted icon indicates which worklist is currently displayed. You can select a different icon to display the corresponding worklist in this area instead.

  The **Filters** worklist is specific to the dashboard. You use this to filter the widgets shown on this dashboard.
- The right area contains one or more widgets. Each *widget* is a rectangular panel that displays data in some form.

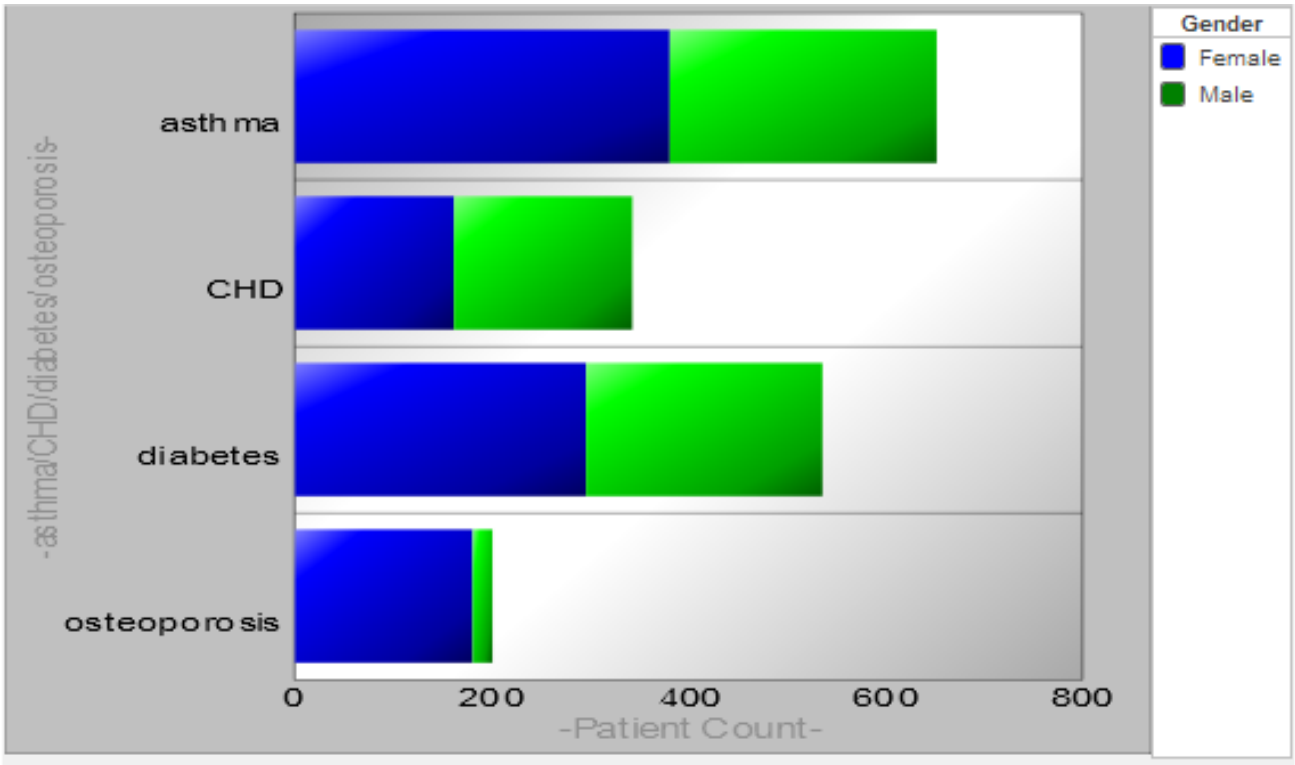
The following subsection describes the more common widgets.

1.2.1 Pivot Table Widgets

A pivot table widget displays data in one of three formats. First, it can display the data as a table:

Age Group		Female		Male	
		Patient Count	Avg Allergy Count	Patient Count	Avg Allergy Count
0 to 29	0 to 9	680	0.60	750	0.63
	10 to 19	756	0.66	769	0.69
	20 to 29	661	0.64	648	0.61
30 to 59	30 to 39	815	0.63	735	0.65
	40 to 49	728	0.68	741	0.61
	50 to 59	586	0.58	552	0.62
60+	60 to 69	397	0.64	319	0.68
	70 to 79	304	0.58	242	0.56
	80+	217	0.57	100	0.66

Second, it can display the data as a chart:



Third, it can display a detail listing, which is a table that shows selected fields from the lowest-level records:

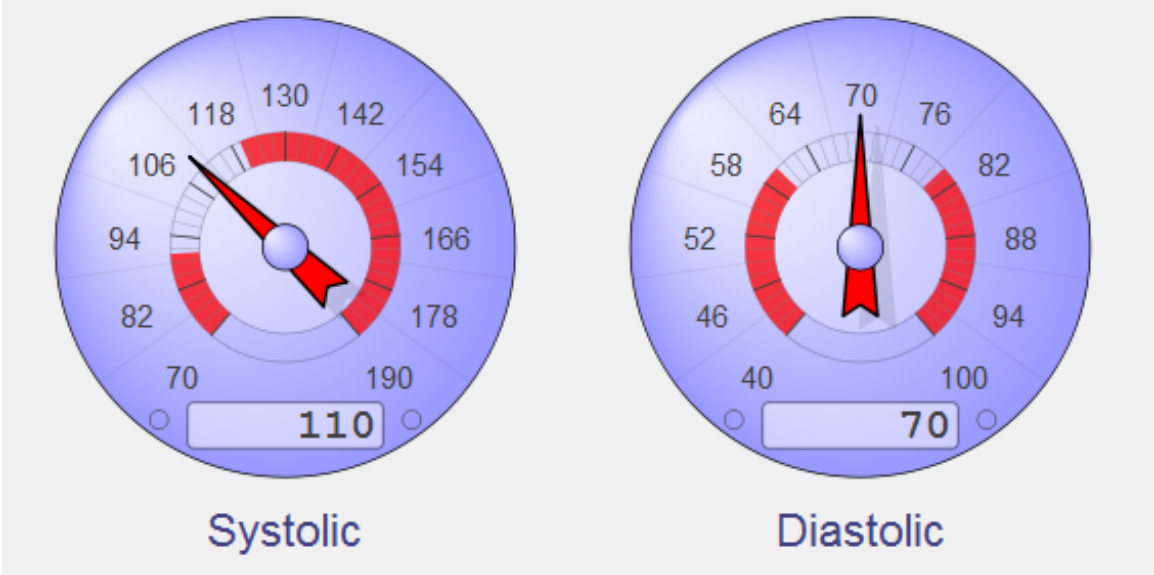
#	PatientID	Age	Gender	Home City	Test Score
1	SUBJ_100631	0	F	Elm Heights	50
2	SUBJ_100781	0	F	Redwood	78
3	SUBJ_100820	0	F	Magnolia	89
4	SUBJ_100966	0	F	Cypress	91
5	SUBJ_101274	0	F	Pine	
6	SUBJ_101340	0	F	Redwood	
7	SUBJ_101466	0	F	Magnolia	81
8	SUBJ_101532	0	F	Pine	67
9	SUBJ_101587	0	F	Elm Heights	77
10	SUBJ_102327	0	F	Redwood	79

1.2.2 Scorecard Widgets

A scorecard widget displays one or more rows of data in a tabular format that also includes features such as value-dependent lamps and arrows. For example:

1.2.3 Meter Widgets

A meter widget displays one or more values, each in a graphical object as follows:



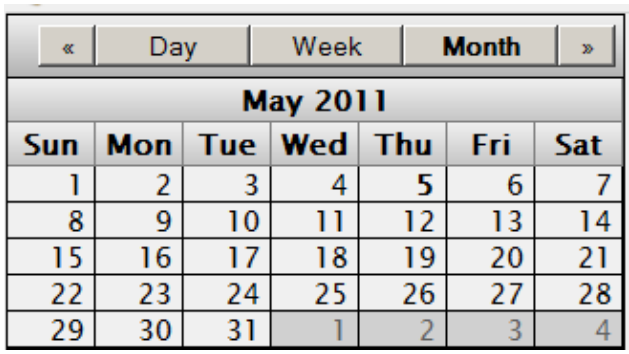
The preceding picture shows values in a speedometer. DeepSee supports several other forms of meters.

1.2.4 Map Widgets

A map widget shows a map with highlighted points that typically correspond to locations that are relevant to your business scenario:

1.2.5 Calendar Widgets

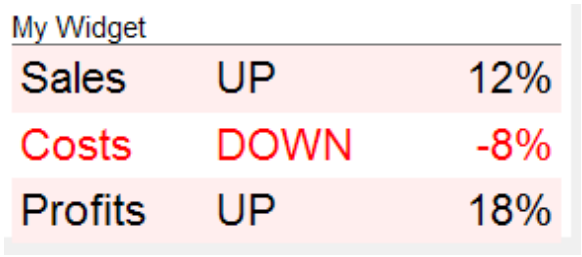
A dashboard can include an informational calendar widget like the following:



«	Day	Week	Month	»		
May 2011						
Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

1.2.6 Custom Widgets

A dashboard can also include custom widgets called *portlets*. The following shows an example:



My Widget

Sales	UP	12%
Costs	DOWN	-8%
Profits	UP	18%

1.3 Data Sources for Widgets

In a dashboard, most widgets use a data source, which is one of the following:

- A pivot table. Pivot tables are created in the Analyzer. A pivot table is a query based on a DeepSee cube, which is part of a DeepSee model. The following section discusses DeepSee models.
- A KPI (key performance indicator). A KPI is a more advanced query created by a programmer; it is also part of a DeepSee model.

1.4 DeepSee Models

A DeepSee model includes some or all of the following elements:

- At least one cube definition. A cube describes ways that you can query a set of specific base elements (such as patients or transactions). A cube includes *levels*, which enable you to group records from the base set, and *measures*, which show aggregate values of those records. It also defines listings and other items.

  You use a cube to create pivot tables. For example:

Patient Group	Avg Test Score
Group A	75.08
Group B	74.22
None	

In this pivot table, the rows correspond to the members of the `Patient Group` level; each member is shown as one row. The data column displays the aggregate value of the `Avg Test Score` measure for each of these members; for this measure, the system computes the average value. Notice that the `Avg Test Score` is null for the `None` patient group.

- Any number of subject areas. A *subject area* is a subcube that enables users to focus on smaller sets of data without the need for multiple cubes. A subject area also enables you to customize captions and defaults of the cube.
- Any number of KPIs (key performance indicators). In DeepSee, a KPI is an interactive dataset that can be displayed on a dashboard. It uses a custom query created by a programmer. The query can use SQL, MDX (MultiDimensional Expressions, which is also generated by the Analyzer), or custom code.

  The KPI can also define *actions*, which a user can launch and which execute your custom code.

2

Introduction to the DeepSee User Interfaces

This chapter provides a quick look at the DeepSee user interfaces. It discusses the following topics:

- How to log on to DeepSee
- Architect
- Analyzer
- User Portal
- Query Tool
- Term List Manager
- Listing Group Manager
- Quality Measure Manager
- Model Browser
- Logs
- Folder Manager
- Settings
- Cube Manager

Note: You can use DeepSee only in a specifically configured web application. See "Setting Up the Web Application" in the *DeepSee Implementation Guide*. Also, be sure to consult the online *InterSystems Supported Platforms* document for this release for information on system requirements.

The **DeepSee** menu contains a couple of items not described here. For information on DeepSee Visual Reporting, see *Using DeepSee Visual Reporting*. For information on the PMML Model Tester, see *Using PMML Models in Caché*.

2.1 Logging On to DeepSee

To log on to DeepSee, do the following in the Management Portal:

1. Switch to the appropriate namespace as follows:

a. Select **Switch**.

b. Select the namespace.

c. Select **OK**.

2. Select **DeepSee**. The system displays a list of the tools for DeepSee:

   - **Architect** — Enables you to define cubes.
   - **Analyzer** — Enables you to define pivot tables.
   - **User Portal** — Launches the User Portal, which includes the Analyzer and the Dashboard Designer.
   - **Tools** — Provides access to the Query Tool, the Term List Manager, Quality Measures, and the Model Browser.
   - **Admin** — Provides access to the DeepSee Logs, the Folder Manager, Settings, and the Cube Manager.

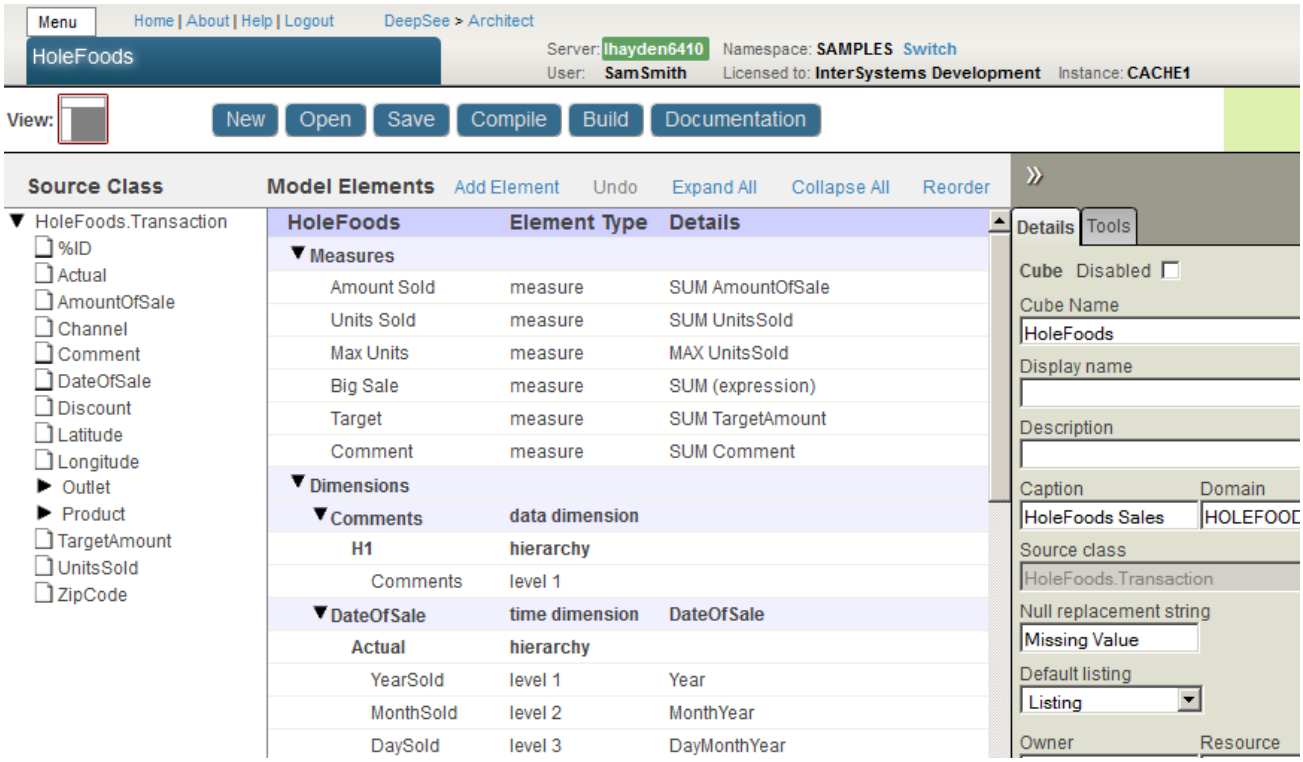
   For information on **Visual Reporting** options, see *Using DeepSee Visual Reporting*.

Note: The Management Portal provides access to all DeepSee tools, including the back-end tools such as the Architect as well as the User Portal. Because the User Portal is intended for end users, it does not enable most users to return to the Management Portal.

2.2 Architect

The DeepSee Architect enables you to define cubes and subject areas. You can use this tool, Studio, or both together.

When you first display the Architect, using the sample HoleFoods cube, you see the following:

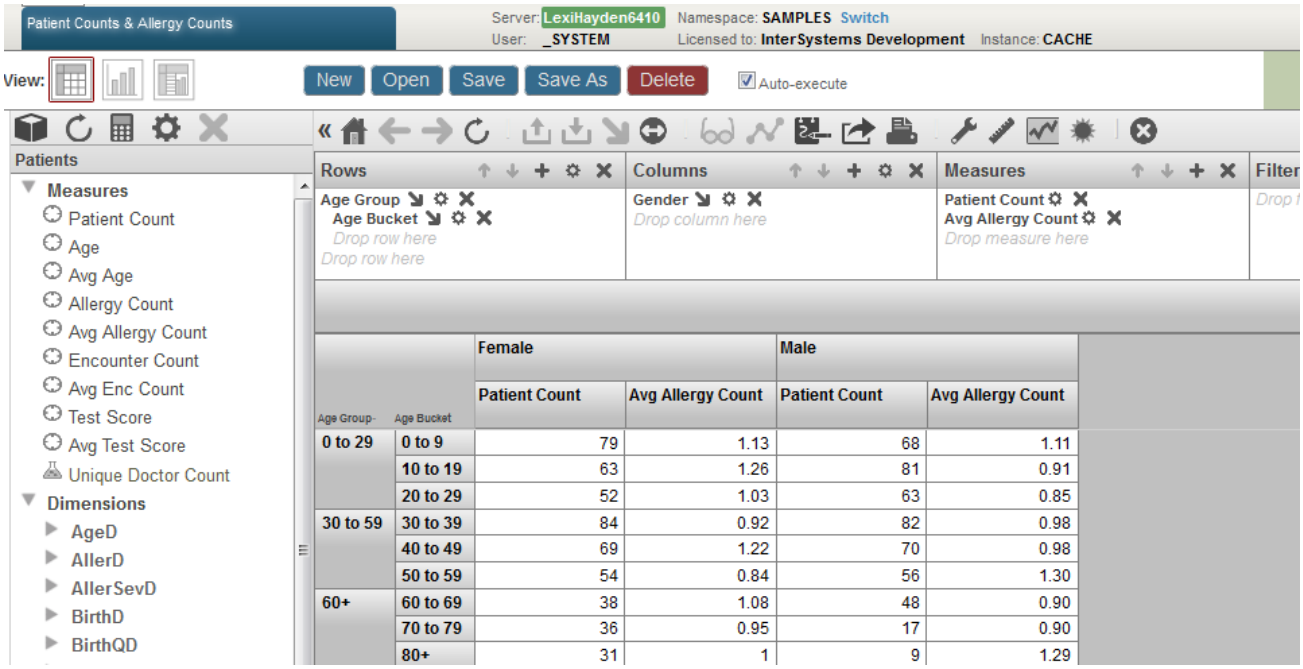


For details, see *Defining DeepSee Models*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.3 Analyzer

The DeepSee Analyzer enables you to define pivot tables. The Analyzer looks like the following.



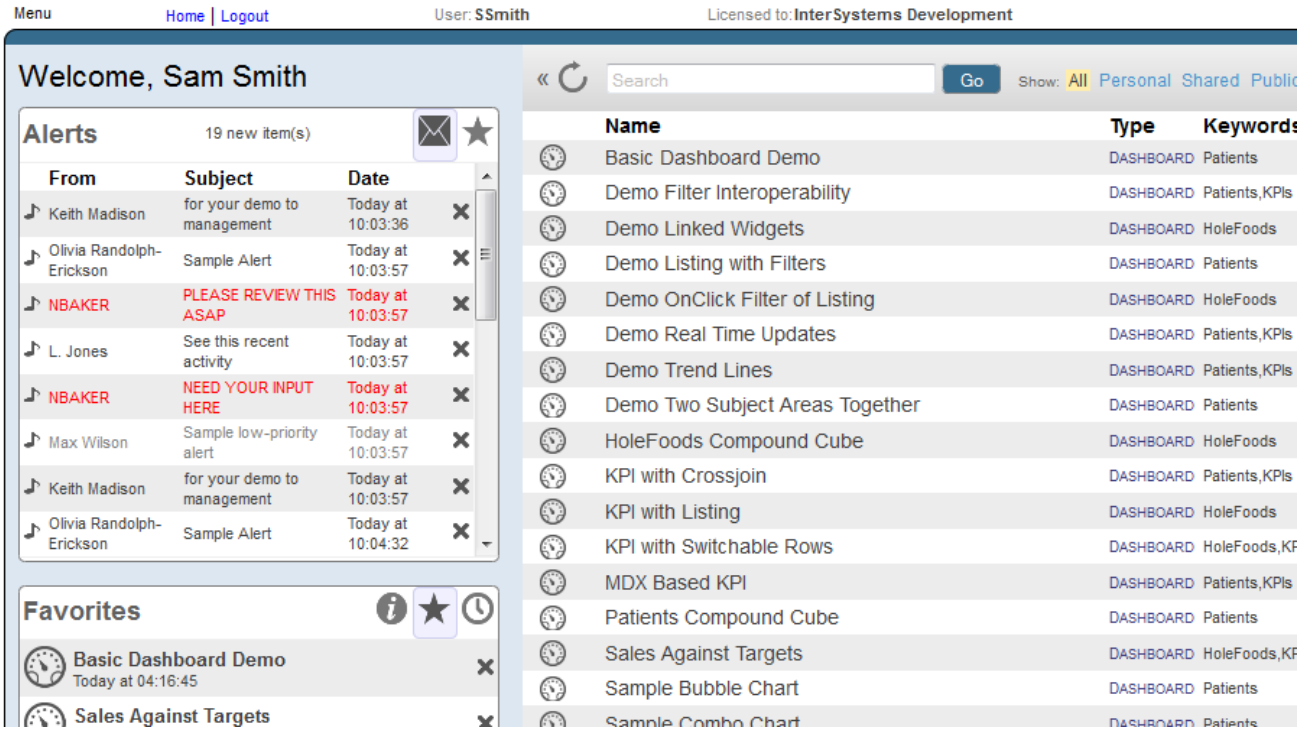
For information, see *Using the DeepSee Analyzer*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.4 User Portal

The User Portal is intended for direct use by end users (in contrast to such back end tools as Studio and the Management Portal). The User Portal includes the Analyzer and the Dashboard Designer.

The User Portal looks like the following:

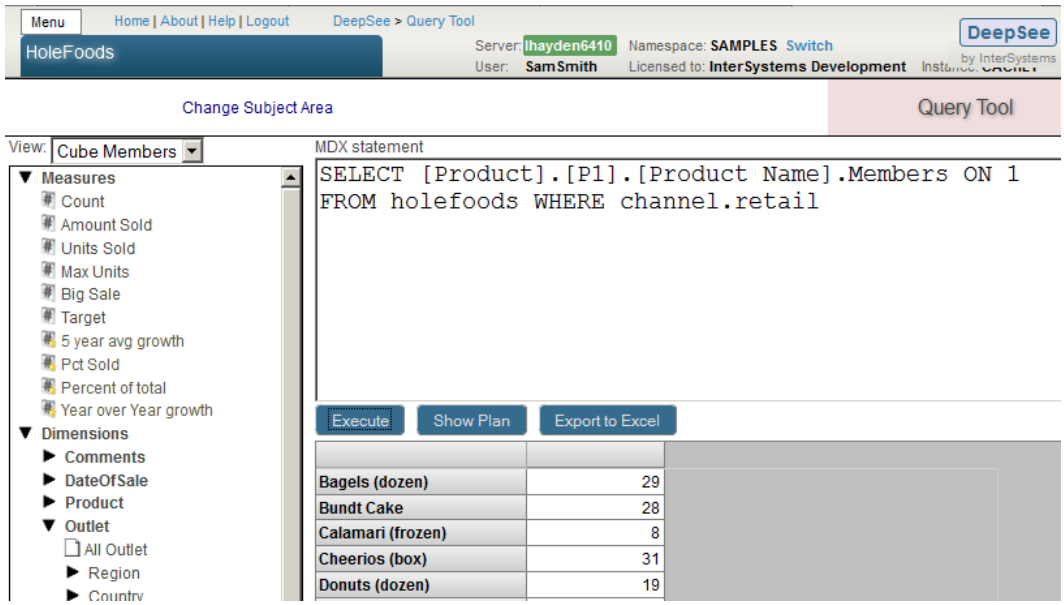


For information, see the *DeepSee End User Guide*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.5 Query Tool

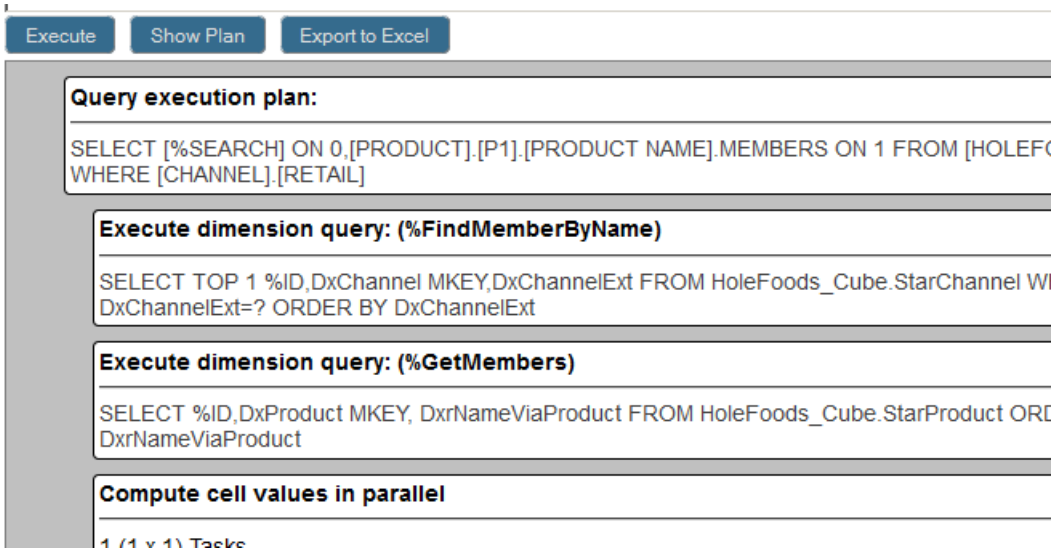
The DeepSee Query Tool enables you to run ad-hoc MDX queries. It looks like the following:



To execute an MDX query, type the query into the text box and then select **Execute**. You can also drag and drop items from the left area into the **MDX statement** area; if you do, the dropped items are added to the end of the query.

The bottom area on the right then displays the results.

To see the plan for the query, select **Show Plan**. For example:

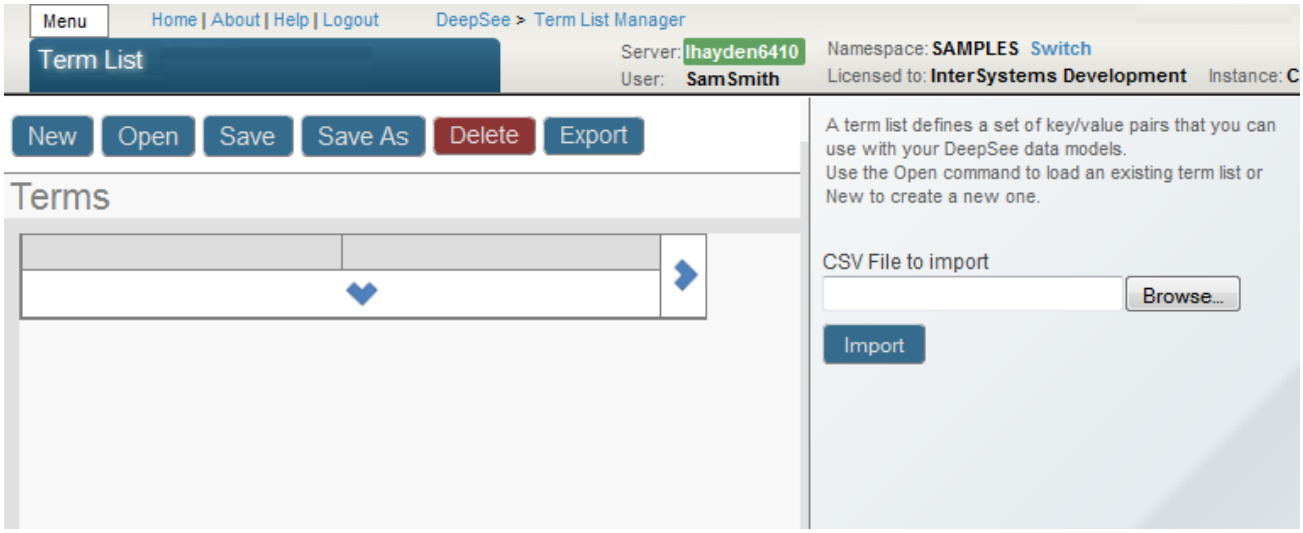


For an introduction to MDX, see *Using MDX with DeepSee*. For reference information on MDX, see the *DeepSee MDX Reference*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.6 Term List Manager

The Term List Manager enables you to build term lists, which provide a way to modify a DeepSee model without programming. It looks like the following:

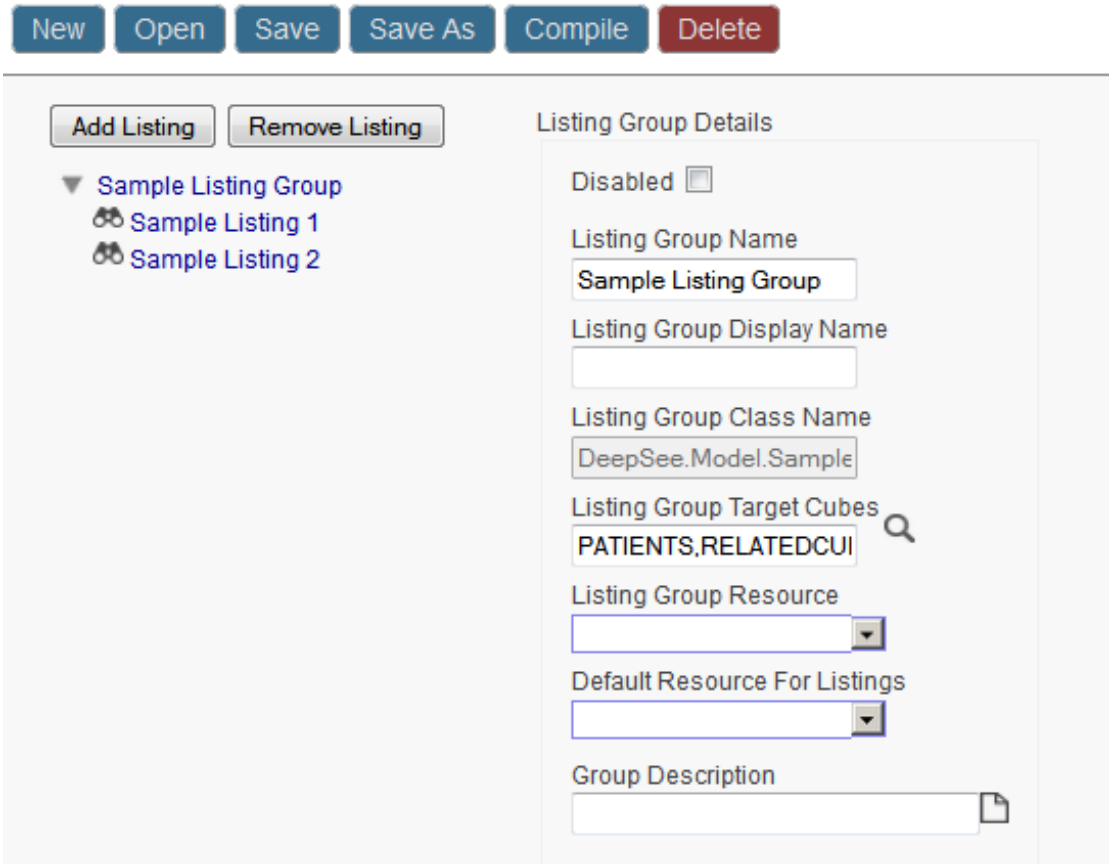


For information on creating term lists, see the *Advanced DeepSee Modeling Guide*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.7 Listing Group Manager

The Listing Group Manager enables you to define listings that are not contained in any cube definition. The purpose of this tool is to enable you (and your customers, if appropriate) to define listings outside of cube definitions and without needing access to the Architect. The Listing Group Manager looks like this:

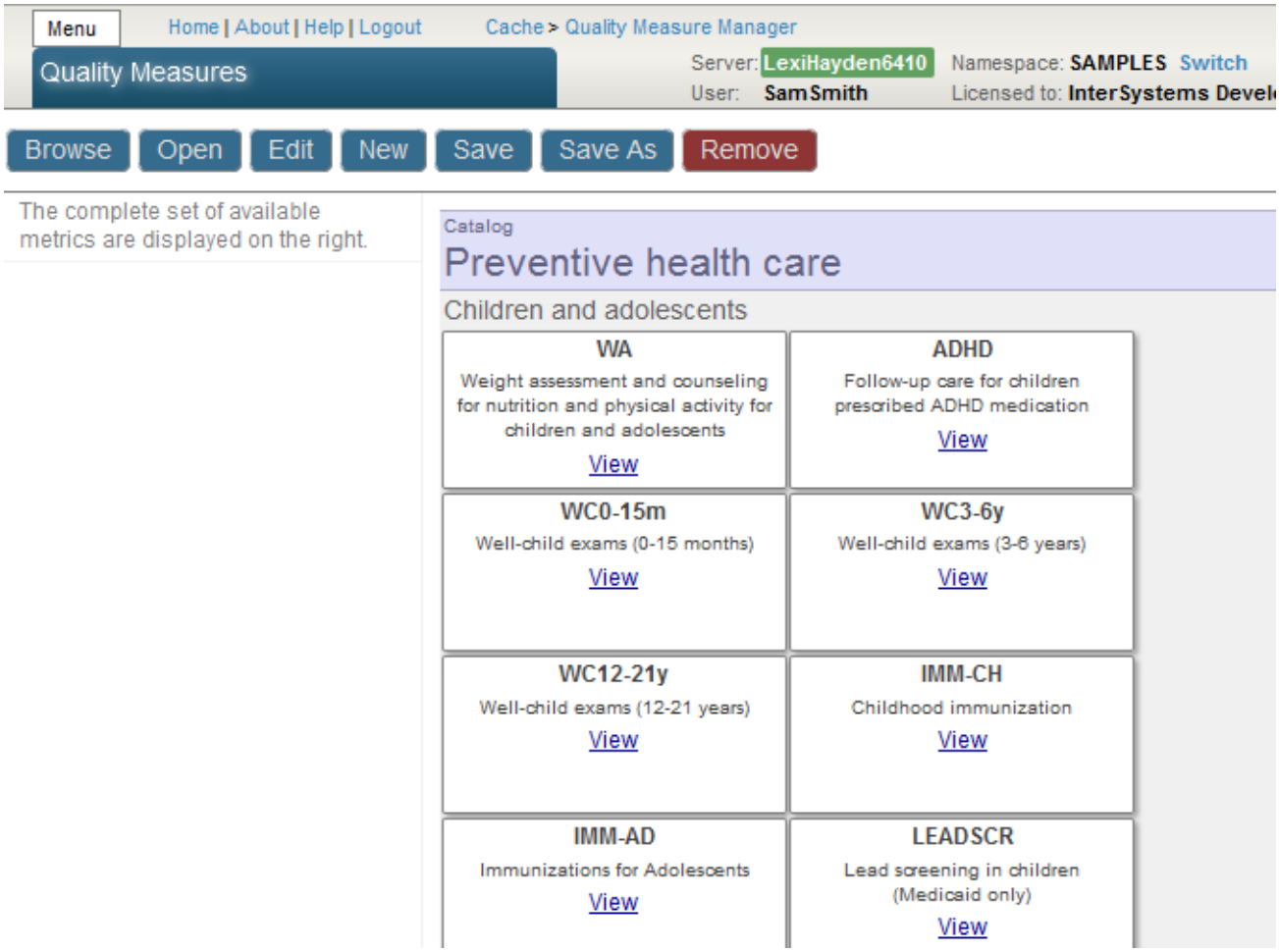


For information, see "Defining Listing Groups" in *Defining DeepSee Models*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.8 Quality Measure Manager

The Quality Measure Manager enables you to define quality measures, a kind of calculated measure that can be reused in multiple contexts. It looks like the following:

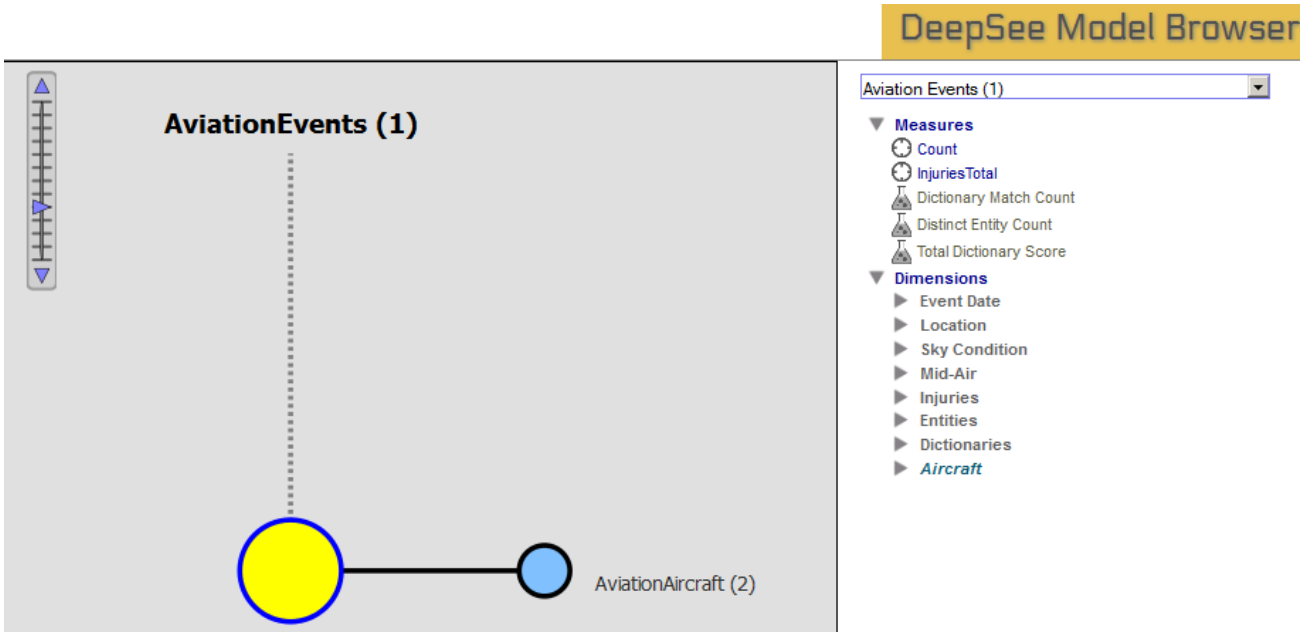


For information, see the *Advanced DeepSee Modeling Guide*

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.9 Model Browser

The Model Browser is a useful way of viewing relationships among cubes. It looks like the following:

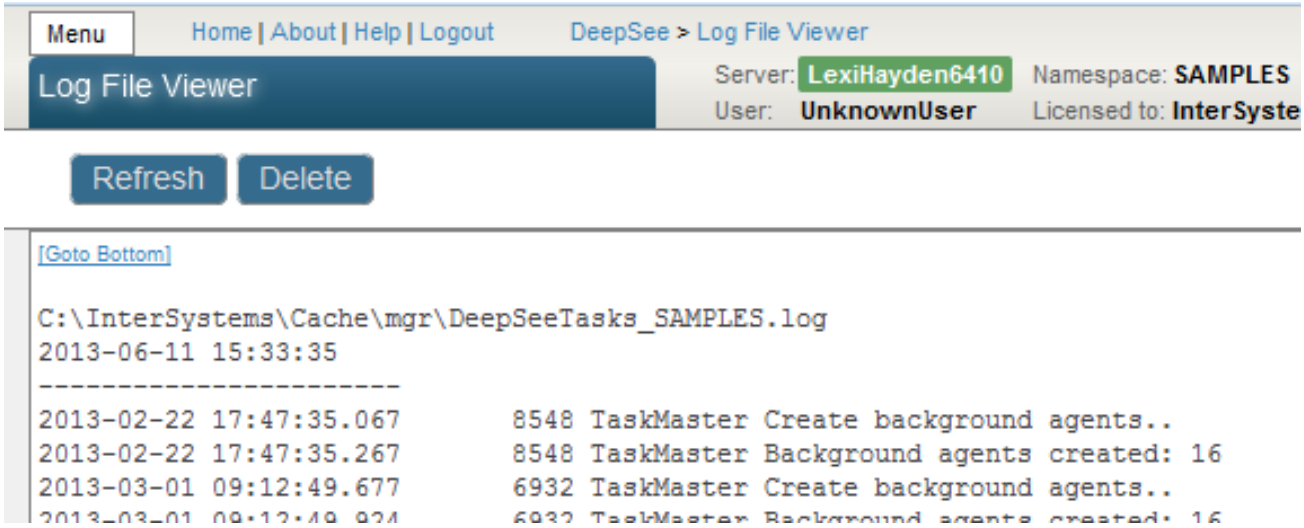


For information, see *Defining DeepSee Models*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.10 DeepSee Logs

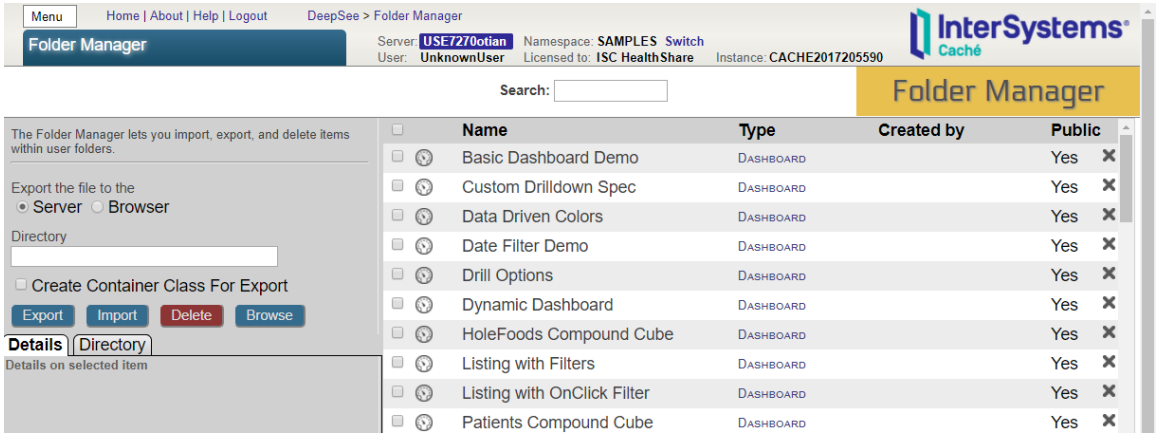
The **DeepSee Logs** option displays the DeepSee log file, which the system generates when it builds cubes. It looks like the following:



For information on the permissions needed to access this page, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.11 Folder Manager

The Folder Manager enables you to manage items within user folders. It looks like the following:

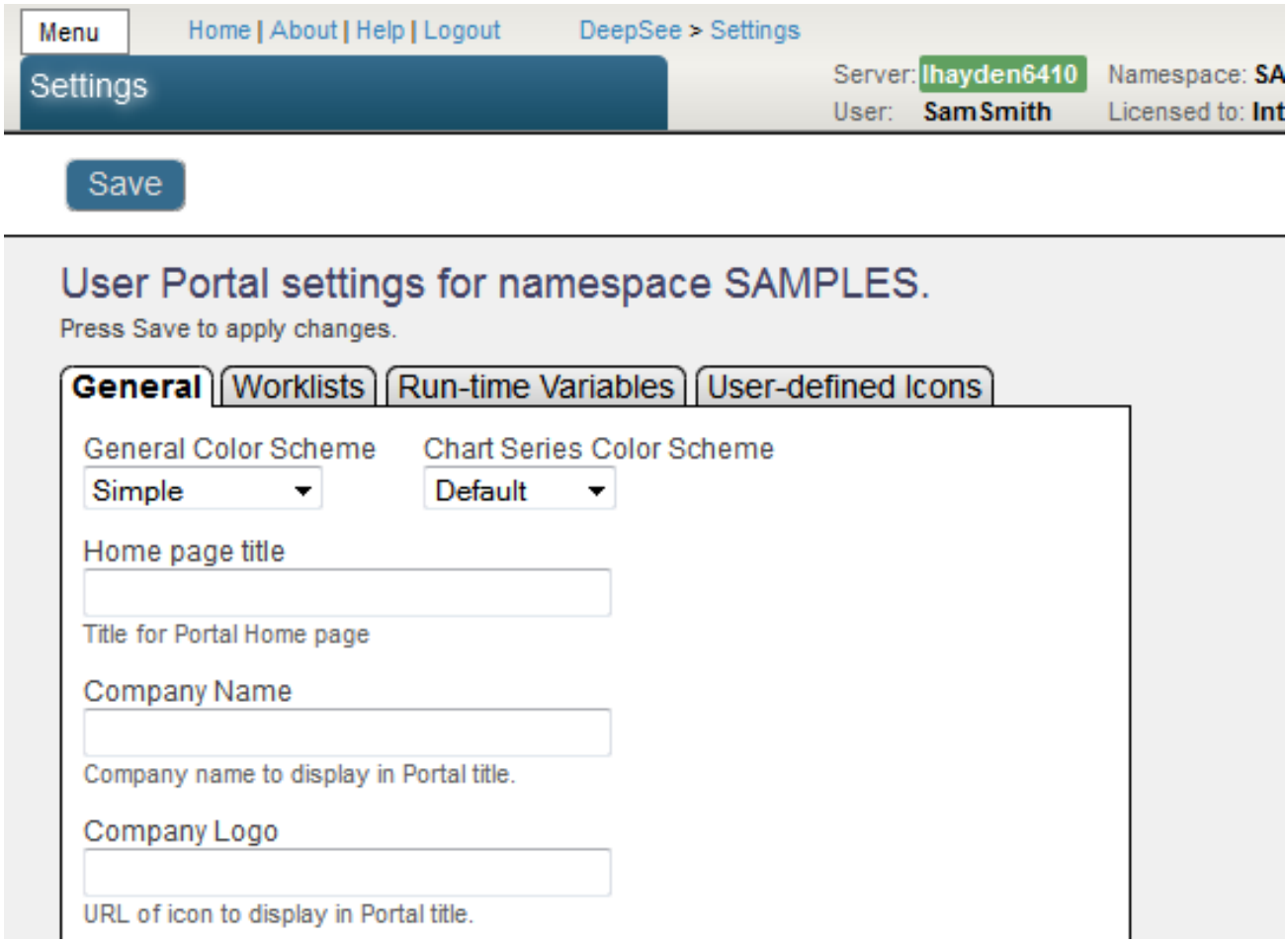


You can use this to export pivot tables and dashboards so that you can package their definitions into a class definition. See the *DeepSee Implementation Guide*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.12 Settings

The **Settings** option lets you specify settings that affect the appearance of DeepSee within this namespace. It looks like the following:

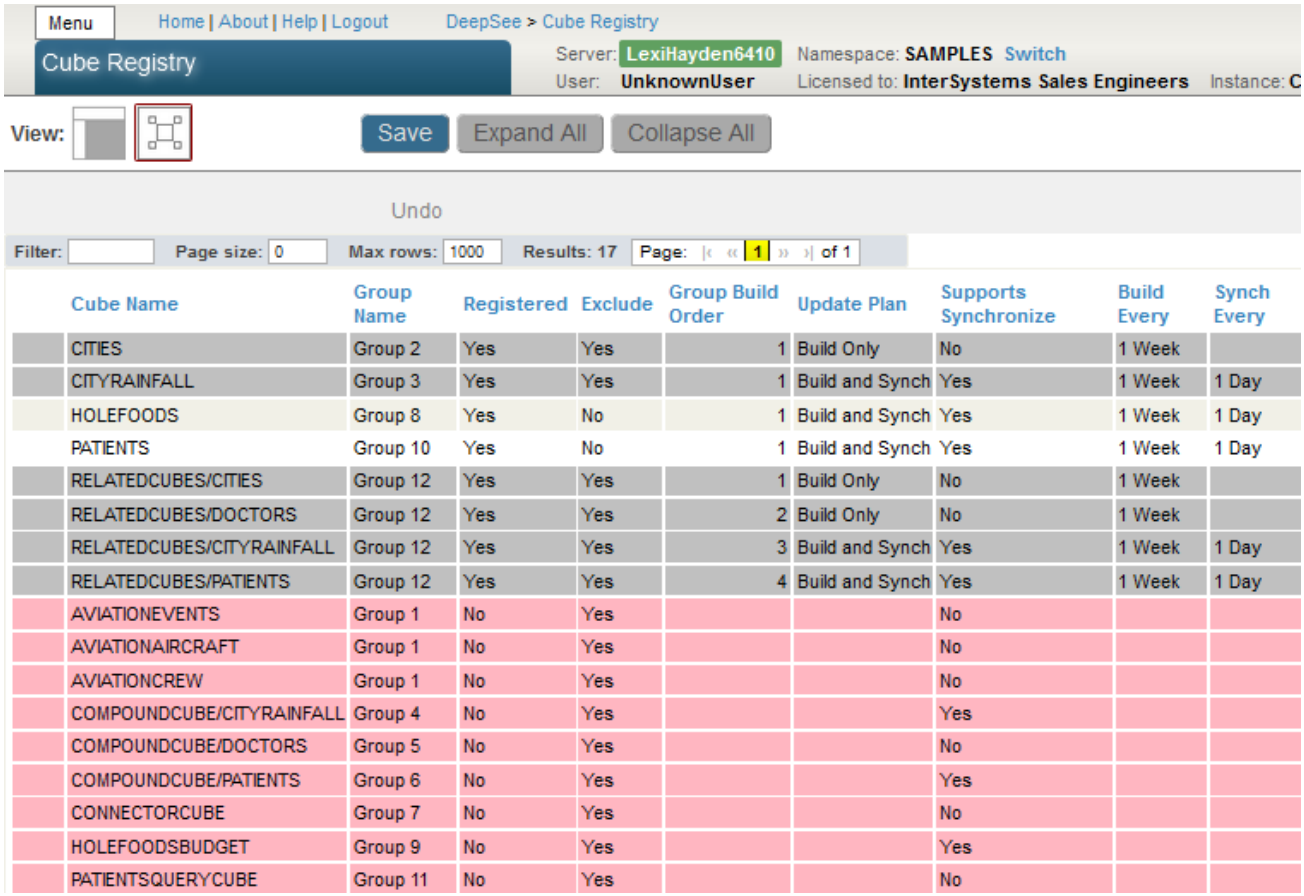


For information, see the *DeepSee Implementation Guide*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

2.13 Cube Manager

The Cube Manager enables you to easily update cubes. You use it to determine how and when to update cubes. It adds automated tasks that rebuild or synchronize cubes at the scheduled dates and times that you choose. It looks like the following:



Cube Name	Group Name	Registered	Exclude	Group Build Order	Update Plan	Supports Synchronize	Build Every	Synch Every
CITIES	Group 2	Yes	Yes	1	Build Only	No	1 Week	
CITYRAINFALL	Group 3	Yes	Yes	1	Build and Synch	Yes	1 Week	1 Day
HOLEFOODS	Group 8	Yes	No	1	Build and Synch	Yes	1 Week	1 Day
PATIENTS	Group 10	Yes	No	1	Build and Synch	Yes	1 Week	1 Day
RELATEDCUBES/CITIES	Group 12	Yes	Yes	1	Build Only	No	1 Week	
RELATEDCUBES/DOCTORS	Group 12	Yes	Yes	2	Build Only	No	1 Week	
RELATEDCUBES/CITYRAINFALL	Group 12	Yes	Yes	3	Build and Synch	Yes	1 Week	1 Day
RELATEDCUBES/PATIENTS	Group 12	Yes	Yes	4	Build and Synch	Yes	1 Week	1 Day
AVIATIONEVENTS	Group 1	No	Yes			No		
AVIATIONAIRCRAFT	Group 1	No	Yes			No		
AVIATIONCREW	Group 1	No	Yes			No		
COMPOUNDCUBE/CITYRAINFALL	Group 4	No	Yes			Yes		
COMPOUNDCUBE/DOCTORS	Group 5	No	Yes			No		
COMPOUNDCUBE/PATIENTS	Group 6	No	Yes			Yes		
CONNECTORCUBE	Group 7	No	Yes			No		
HOLEFOODSBUDGET	Group 9	No	Yes			Yes		
PATIENTSQUERYCUBE	Group 11	No	Yes			No		

For details, see "Keeping the Cubes Current" in the *DeepSee Implementation Guide*.

For information on the permissions needed to use this tool, see "Setting Up Security" in the *DeepSee Implementation Guide*.

3

Introduction to the Other DeepSee Tools

This chapter introduces the other tools for working with DeepSee.

3.1 MDX Shell

DeepSee provides a shell in which you can issue MDX queries to explore your DeepSee cubes and subject areas. This section introduces this shell and lists the supported MDX options and functions.

For an introduction to MDX queries, see *Using MDX with DeepSee*, which contains many examples.

Also see the *DeepSee MDX Reference*.

3.1.1 Accessing the MDX Shell

To access the MDX shell, start the Terminal and do the following:

1. Switch to the namespace in which you defined the cube or subject area.
2. Enter the following command:

```
Do ##class(%DeepSee.Utils).%Shell()
```

Now you can enter MDX queries like the following:

```
SELECT MEASURES.[%COUNT] ON 0, birthd.decade.MEMBERS ON 1 FROM patients
```

When you do so, the shell executes the query, displays its results to the console, and redisplays the shell prompt, as follows:

```
                            Patient Count
 1 1910s                               71
 2 1920s                              223
 3 1930s                              572
 4 1940s                              683
 5 1950s                            1,030
 6 1960s                            1,500
 7 1970s                            1,520
 8 1980s                            1,400
 9 1990s                            1,413
10 2000s                            1,433
11 2010s                              155
------------------------------------------------------------------------
Elapsed time:       .014128s
```

In the shell:

- To display a list of cubes and subject areas, enter `cube`

- To see the contents of a cube or subject area, enter `cube` *name_of_cube_or_subject_area*

  Note: This command does not display calculated members and named sets, although you can use these elements in the shell and elsewhere.

  For a subject area, this command lists all elements, even if those are specified as hidden in the subject area.

- To exit the shell, enter `q`
- To enable query caching, enter `cache on`. If you set `cache off` and then run an MDX query in the shell, the MDX shell purges all cached queries for that specific cube system-wide. This could lead to slower performance for other users.
- To enable the asynchronous mode, enter `async on`
- To build a cube, enter `build` *cubename*
- To reset the query cache, enter `reset`
- For a list of additional shell options, enter `?`

3.1.2 Viewing the Indices Used by a Query

The DeepSee shell provides a quick way to see the indices that a query uses:

1. Issue the following shell command:

   ```
   stats on
   ```

2. Enter the query, preceded by `%SHOWPLAN`. For example:

   ```
   %SHOWPLAN SELECT aged.[age group].members ON 0, allerd.H1.MEMBERS ON 1 FROM patients WHERE colord.red

                              0 to 29           30 to 59               60 +
    1 additive/colorin              27                 19                 14
    2 animal dander                 15                 25                  8
    3 ant bites                     15                 19                 11
    4 bee stings                    24                 27                  7
    5 dairy products                25                 25                  4
    6 dust mites                    28                 23                 10
    7 eggs                          19                 21                 13
    8 fish                          26                 17                 11
    9 mold                          23                 23                  6
   10 nil known allerg              80                 82                 21
   11 No Data Availabl             216                194                 92
   12 peanuts                       26                 15                  8
   13 pollen                        29                 22                 11
   14 shellfish                     29                 23                 14
   15 soy                           25                 25                  6
   16 tree nuts                     22                 18                  8
   17 wheat                         16                 17                  8
   -------------- Query Plan ---------------------
   **%SHOWPLAN SELECT [AGED].[AGE GROUP].MEMBERS ON 0,[ALLERD].[H1].MEMBERS ON 1 FROM [PATIENTS] WHERE
    [COLORD].[RED]**
   **DIMENSION QUERY (%FindMemberByName): SELECT TOP 1 %ID,Dx327553094 MKEY,Dx327553094 FROM
   Cubes_StudyPatients.Star327553094 WHERE Dx327553094=? ORDER BY Dx327553094**
   **EXECUTE PARALLEL: 1x1 task(s) **
   **CONSOLIDATE**
   -------------- End of Plan -----------------
   ```

   Line breaks were added here for readability.

DeepSee captures all the indices used by the query and reports them. Note that the query results are not necessarily correct because the query is only partially run; the purpose of `%SHOWPLAN` is to enable you to see the indices, not to get the query results.

3.2 Utility Methods

- The class %SYSTEM.DeepSee includes the most commonly used utility methods. These include:
  - **BuildCube()**
  - **KillCube()**
  - **ListCubes()**
  - **Reset()**
  - **Shell()**
  - **SynchronizeCube()**

  This class is available via the special variable **$SYSTEM**, as are all classes in the %SYSTEM package. For example, to build a cube, you can use the following:

```
Do $system.DeepSee.BuildCube("MyCube")
```

- The class %DeepSee.Utils includes a large set of utility methods, including:
  - **%ExportExcelToFile()** — exports a DeepSee query or KPI to a file in Microsoft Excel format
  - **%ExportPDFToFile()** — exports a DeepSee query or KPI to a file in PDF format
  - **%GetAgentCount()** — gets the current agent count
  - **%GetBaseCube()** — gets the name of cube on which a subject area is based
  - **%GetCubeFactClass()** — gets the name of fact table class associated with a cube
  - **%GetCubeLevels()** — gets the levels, measures, and relationships defined in a cube
  - **%GetDimensionMembers()** — gets the list of members of a dimension
  - **%GetMetricList()** — gets all Ensemble business metrics visible to current user
  - **%GetSQLTableName()** — gets SQL table name for a given class
  - **%ProcessFact()** — updates a single fact for a cube
  - **%GetMDXFromPivot()** — returns the MDX query defined by a pivot table
  - **%ExecutePivot()** — runs the MDX query defined by a pivot table and optionally returns an instance of %DeepSee.ResultSet
  - **%GetResultsetFromPivot()** — returns an instance of %DeepSee.ResultSet that holds the MDX query defined by a pivot table and optionally runs that query

- The class %DeepSee.UserLibrary.Utils includes methods that you can use to programmatically perform the tasks supported in the Folder Manager. These methods include:
  - **%AddFavorite()**
  - **%DeleteFolderContents()**
  - **%DeleteFolderItem()**
  - **%Export()**
  - **%GetFolderList()**
  - **%ImportContainer()**

3.3 Data Connector

The data connector class (%DeepSee.DataConnector) enables you to make arbitrary SQL queries available for use in DeepSee cubes and listings. See the *DeepSee Implementation Guide*.

3.4 Result Set API

The class %DeepSee.ResultSet enables you to execute MDX queries programmatically and access the results.

For information, see the *DeepSee Implementation Guide*.

3.5 JavaScript and REST APIs

The DeepSee JavaScript API is provided by the file DeepSee.js, which is in the *install-dir*/CSP/broker directory. This JavaScript library enables you to interact with DeepSee from a client that is based on JavaScript. The functions in this library are a wrapper for a REST-based API for DeepSee. You can also use the REST API directly.

For information, see *Tools for Creating DeepSee Web Clients*.

DeepSee Glossary

This glossary summarizes terms found in the DeepSee documentation. If you have not yet done so, InterSystems highly recommends that you read "Basic Concepts" in *Defining DeepSee Models*.

action

An operation that a user can start by using a control (such as a button) on a dashboard. DeepSee provides a set of standard actions (such as applying a filter, navigating to another dashboard, and others), and you can add custom actions. See "Defining Custom Actions" in the *DeepSee Implementation Guide*.

age dimension and age level

An age dimension is a dimension that contains age levels. An age level groups data by an age, relative to the cube build time, computed from a date or time value in the source data. Age dimensions and age levels are not generally recommended, because they require nightly rebuilds.

age measure

A measure that provides an aggregated age value in days. Age measures are not generally recommended, because they require nightly rebuilds.

All level and All member

The All level is a special, optional level, which appears in all the hierarchies of a dimension. If defined, this level contains one member, the All member, which corresponds to all records in the cube. You can use the All member to create a summary line in a pivot table.

BI

Business Intelligence, a set of tools and techniques that transform raw data into insights that can improve the operation of a business or other organization. BI is intended to support a measurement-based approach to making strategic and tactical decisions.

building a cube

The process of iterating through the source class for a cube and populating the fact table (and building the indices for that table). See also synchronizing a cube.

For details, see "Compiling and Building Cubes" in *Defining DeepSee Models* and "Keeping the Cubes Current" in the *DeepSee Implementation Guide*

business metric

A two-dimensional array of data generated by a running Ensemble production and generally providing data relevant to or about that production. Like pivot tables, business metrics can be displayed on a dashboard, within a widget. For information on creating Ensemble business metrics, see *Developing Ensemble Productions*.

business rule

A Ensemble concept that allows nontechnical users to change the behavior of Ensemble business processes. You can use them in source expressions in DeepSee cubes; see "Details for Source Expressions" in *Defining DeepSee Models*. For details on Ensemble business rules, see *Using Business Rules with Ensemble*.

calculated measure

A measure that is based on other measures via an MDX expression. The phrase *calculated measure* is not standard in MDX, but this documentation uses it for brevity. Formally, a calculated measure is a calculated member that belongs to the Measures dimension.

calculated member

A member that is based on other members via an MDX expression. You can define two kinds of calculated members:

- A calculated measure is a measure is based on other measures. (In MDX, each measure is a member of the Measures dimension.)

  For example, one measure might be defined as a second measure divided by a third measure.

  The phrase *calculated measure* is not standard in MDX, but this documentation uses it for brevity.

- A non-measure calculated member typically aggregates together other non-measure members. Like other non-measure members, this calculated member is a group of records in the fact table.

See "Defining Calculated Members" in *Defining DeepSee Models*.

compound cube

A special kind of subject area that combines multiple cube definitions (typically two) and that enables you to create pivot tables that contain elements from multiple cubes. See "Defining Shared Dimensions and Compound Cubes" in the *Advanced DeepSee Modeling Guide*.

computed dimension

A special kind of DeepSee dimension whose members are computed at runtime via an SQL or MDX expression. See "Defining Computed Dimensions" in the *Advanced DeepSee Modeling Guide*.

Computed dimensions do not have any association with calculated members. A computed dimension is specific to DeepSee. A calculated member is a standard concept in MDX.

container class

A class that extends %DeepSee.UserLibrary.Container. This class can contain the definitions of pivot tables, dashboards, and other DeepSee folder items. When you compile this class, Caché generates those folder items, replacing any current definitions that they might have. See the *DeepSee Implementation Guide*.

control

An interactive element on a dashboard. Controls include drop-down lists and buttons.

cube

An model of your data that defines elements that can be used in MDX queries. These elements determine how you can query your data, specifically, a set of specific records (such as patient records or transaction records). The set of records is determined by the source class for the cube. For an introduction, see "Basic Concepts" in *Defining DeepSee Models*.

cube inheritance

A mechanism in DeepSee that enables you to define multiple similar cubes. This mechanism has no relationship to class inheritance. See "Using Cube Inheritance" in the *Advanced DeepSee Modeling Guide*.

custom listing

A listing, specifically one of the following special kinds of listings:

- A listing that uses a custom SQL query that retrieves fields from some other table, not the source table used by the cube, and not a data connector. See "Defining Listings" in *Defining DeepSee Models*.
- A listing that consists of listing fields chosen by the user, in the Analyzer. See "Performing Ad Hoc Analysis" in *Using the DeepSee Analyzer*.

dashboard

An interactive display of data, particularly data that provides a high-level data of a business. See *Creating DeepSee Dashboards*.

data connector

A class that extends %DeepSee.DataConnector. A data connector maps the results of an arbitrary SQL query into an object that can be used as the source of a cube. Typically, a data connector accesses external non-Caché data, but you can also use it to specify an SQL query against Caché, including an SQL query on a view. See "Defining and Using Data Connectors" in the *DeepSee Implementation Guide*.

detail listing

See listing.

dimension

A container for levels. A dimension contains one or more hierarchies, which in turn contain levels. For example, a single dimension might contain multiple hierarchies related to allergies. There is no formal relationship between two different hierarchies or between the levels of one hierarchy and the levels of another hierarchy. The practical purpose of a dimension is to define the default behavior of the levels that it contains — specifically the All level.

See "Defining Dimensions, Hierarchies, and Levels" in *Defining DeepSee Models*.

dimension table

The table in which DeepSee stores the members of a level and any properties they have. See "Details for the Fact and Dimension Tables" in *Defining DeepSee Models*.

drill down

Examine a row of a pivot table and see the data for that row displayed in a more granular way. For example, a row might display data for a year, and you would drill down to see data for that year, broken out by month. DeepSee supports multiple forms of drill down. See "Performing Ad Hoc Analysis" in *Using the DeepSee Analyzer*.

Informally (although not in this documentation), the phrases drill down and drill through are sometimes used interchangeably, and it is wise to double-check which phrase is intended.

drill through

Formally, *to drill through* means to display a listing. Internally, DeepSee uses the MDX DRILLTHROUGH statement when it displays a listing. See "Performing Ad Hoc Analysis" in *Using the DeepSee Analyzer*.

Informally (although not in this documentation), the phrases drill through and drill down are sometimes used interchangeably, and it is wise to double-check which phrase is intended.

expression

An expression (`<expression>` element) whose value is available while DeepSee is building a row in the fact table. You can define an expression that uses complex or time-consuming logic, and then you can base multiple cube elements on the expression. Expressions are for use during cube build only and are provided for efficiency.

See "Other Options" in the *Advanced DeepSee Modeling Guide*.

fact

A row in the fact table.

fact table

A generated structure that DeepSee queries directly. When you compile a cube definition, DeepSee generates a fact table class. When you build a cube, DeepSee creates records for this table and indexes them. See "Basic Concepts" in *Defining DeepSee Models*.

filter

A restriction on the data. DeepSee provides two simple ways to filter data: member-based filters and measure-based filters. You can combine these, and more complex filters are also possible, especially if you write MDX queries directly. For an introduction, see "Filters" in "Basic Concepts" in *Defining DeepSee Models*.

folder item

Any of the following DeepSee items:

- Pivot tables
- Saved widgets
- Dashboards
- Themes

DeepSee folder items are visible in the Studio **Workspace** window, where they are shown in the **Other** folder.

geo listing

See map listing.

hierarchy

An organization of levels. Levels belong to hierarchies (which belong to dimensions). A hierarchy can contain only single level or can contain multiple levels. If it contains multiple levels, the "higher" levels of the hierarchy are less granular then the "lower" levels. That is, each member of a higher level contains a larger set of records than does a member of a lower level.

In casual usage, a higher level is called the parent of the lower level. However, it is useful to remember that the hierarchy is a actually a hierarchy among members. Thus it is more accurate to state that a member of the higher level is the parent of one or more members of the lower level. Conversely, any member of a lower level is the child of exactly one member of the higher level.

Hierarchies provide additional features beyond those provided by levels; see "Hierarchies and Dimensions" in *Defining DeepSee Models*. Also see "Defining Dimensions, Hierarchies, and Levels" in the same book.

iKnow dimension

A special kind of dimension that analyzes an iKnow measure, which in turn is a measure based on unstructured text. See "Using Unstructured Data in Cubes (iKnow)" in the *Advanced DeepSee Modeling Guide*.

iKnow measure

A special kind of measure that is based on unstructured text. You cannot display iKnow measures directly in pivot tables. Their purpose is to provide data for use by iKnow dimensions. See "Using Unstructured Data in Cubes (iKnow)" in the *Advanced DeepSee Modeling Guide*.

KPI

A class based on %DeepSee.KPI. In most cases, a KPI uses a query and displays a result set. Like pivot tables, KPIs can be displayed on a dashboard, within a widget. You can also use KPIs as building blocks for calculated members (including calculated measures). See "Defining Basic KPIs" and the chapters that follow it, in the *Advanced DeepSee Modeling Guide*.

level

A cube element that enables you to group records. A level consists of members, each of which is a set of records. See "Basic Concepts" in *Defining DeepSee Models*. Also see "Details of Defining Levels" in the same book.

list-based level

A level that is based upon a list value. For example, a patient can have multiple diagnoses. The `Diagnoses` level groups patients by diagnosis. With a list level, it is possible for a given record of the source class to have multiple values and thus for that record to belong to multiple members of the level.

listing

An SQL query that accesses the lowest-level records associated with one or more cells of a pivot table. See "Defining Listings" in *Defining DeepSee Models*.

listing group

A class that defines a group of listings. Listing groups are created in the Listing Group Manager. The purpose of this tool is to enable you (and your customers, if appropriate) to define listings outside of cube definitions and without needing access to the Architect. See "Defining Listing Groups" in *Defining DeepSee Models*.

listing field

A `<listingField>` element defined in a cube definition. Users can select the listing fields to include, when they create custom listings in the Analyzer. See "Defining Listing Fields" in *Defining DeepSee Models*.

This phrase can also refer more generally to any field in any listing.

map listing

A listing that contains location data and is displayed as a map. Each pin on the map corresponds to a source record.

Important: A map listing uses the Google Maps API. Be sure that your usage of this API is consistent with the Terms of Use, which you can access via a link displayed in this listing.

Note that in order to use the Google Maps API, you must obtain an API key. See Specifying Basic Settings in the *DeepSee Implementation Guide* for more information.

map widget

A dashboard widget that contains location data and is displayed as a map. Each pin on the map corresponds to a member of a level, particularly a level that refers to locations.

Important: A map widget uses the Google Maps API. Be sure that your usage of this API is consistent with the Terms of Use, which you can access via a link displayed in this widget.

Note that in order to use the Google Maps API, you must obtain an API key. See Specifying Basic Settings in the *DeepSee Implementation Guide* for more information.

MDX

MultiDimensional eXpressions, a standard query language for OLAP (online analytical processing) databases and used in many BI applications. See *Using MDX with DeepSee* and *DeepSee MDX Reference*.

measure

A cube element that (with rare exceptions) aggregates values across multiple records. Each measure is based on a source value, which is either a class property or an ObjectScript expression. The definition of a measure also includes an aggregation function, which specifies how to aggregate values for this measure. See "Basic Concepts" in *Defining DeepSee Models*. Also see "Defining Measures" in the same book.

member

A set of records. Every level has one or members. See "Basic Concepts" in *Defining DeepSee Models*. Also see "Details of Defining Levels" in the same book.

named filter

A reusable filter that is defined in the Analyzer. See "Filtering Pivot Tables" in *Using the DeepSee Analyzer*.

named set

A reusable MDX set that is defined within a cube. See "Defining Named Sets" in *Defining DeepSee Models*.

pivot table

An interactive, drillable display of data, generally with rows and columns, designed for specific user roles or for specific areas of your user interface. A pivot table is based on an MDX query that is executed at runtime can respond to input such as filter selections made by the user. Internally it obtains values from a cube. See *Using the DeepSee Analyzer*.

pivot variable

An element that is intended to be used in pivot tables, specifically, in selected parts of the query that defines the pivot table. When a dashboard displays the pivot table, that dashboard can include a control with which the user can change the value of the corresponding pivot variable. See "Defining and Using Pivot Variables" in *Using the DeepSee Analyzer*.

Pivot variables are entirely different from runtime variables.

portlet

A custom widget that can be displayed on dashboards. For information on creating portlets, see the *DeepSee Implementation Guide*.

plugin

A specialized form of KPI that defines one or more computations to use in the Analyzer and in queries. Plugins are especially appropriate for complex or time-consuming computations. For example, you might have a computation that uses several different parts of the source record, as well as external information; a plugin would be suitable in this case. See "Defining Plugins" in the *Advanced DeepSee Modeling Guide*.

property

A value that is specific to a member of a given level. If a level has a property, then each member of that level has a value for that property; other levels do not have values for the property. You can use properties in queries in much the same way that you use measures. In DeepSee, you can also use properties for other purposes such as controlling member names and controlling the order in which member are sorted. See "Defining Properties" in *Defining DeepSee Models*.

quality measure

A quality measure is similar to a calculated measure because it is defined by a formula that combines MDX expressions. You specify the subject area or subject areas in which it is available, and you can control whether the quality measure is *published* (and thus available in the Analyzer). Each quality measure is a subclass of %DeepSee.QualityMeasure.QualityMeasure.

For information, see "Defining Quality Measures" in the *Advanced DeepSee Modeling Guide*.

related cube

A cube whose dimensions, hierarchies, and levels are available within another cube, because there is a relationship between the two cubes.

relationship

A connection between two cubes that makes the dimensions of one cube available in the other cube (and possibly vice versa). If you define relationships, you can define a level once rather than multiple times, which minimizes the sizes of fact tables and their indices. See "Cube-Cube Relationships" in the *Advanced DeepSee Modeling Guide*.

runtime variable

A special element that is intended for use as the default value of a filter on a dashboard (currently this is their only use). The definition of a runtime variable is an ObjectScript expression that is evaluated at runtime. See "Configuring Settings" in the *DeepSee Implementation Guide*.

Runtime variables are entirely different from pivot variables.

searchable measure

A measure that enables you to apply a filter that considers the values in the source records. Searchable measures are an InterSystems extension to MDX. In standard MDX, a filter can be based only on members. See "Defining Measures" in *Defining DeepSee Models*.

set

A list of multiple MDX items, typically used for rows or columns of a pivot table. The items can be any combination of literal values, members, and tuples. For an introduction, see "Working with Sets" in *Using MDX with DeepSee*. For reference information, see "Set Expressions" in *DeepSee MDX Reference*.

shared dimension

A dimension that can be used in more than one cube. That is, more than one cube can use members of the dimension for rows or columns or for filtering. A dimension can be shared formally or informally. If the dimension is shared formally, you can define a compound cube that combines the cubes that use this dimension. See "Defining Shared Dimensions and Compound Cubes" in the *Advanced DeepSee Modeling Guide*.

source class, source records

The source class is the class that contains the data upon which a cube is based. Every cube has a source class, which is usually a persistent class. A source class has a set of source records. For an introduction, see "Basic Concepts" in *Defining DeepSee Models*.

star table

See dimension table.

subject area

A view of a cube with optional overrides. A subject area uses the fact table and related tables of the associated cube and does not require independent updates. You define subject areas to enable users to focus on smaller sets of data without the need for multiple cubes. See "Defining Subject Areas" in *Defining DeepSee Models*.

synchronizing a cube

The process of updating the fact table and indices for a cube, based on incremental changes to the source class. See "Compiling and Building Cubes" in *Defining DeepSee Models* and "Keeping the Cubes Current" in the *DeepSee Implementation Guide*

See also building a cube.

term list

A simple (but extendable) list of key and value pairs. Term lists provide a way to customize a DeepSee model without programming. See "Defining Term Lists" in the *Advanced DeepSee Modeling Guide*.

time dimension and time level

A time dimension is a dimension that contains time levels. A time level groups data by a date or time value in the source data.

tuple

A type of MDX value that consists of an intersection of members. If the tuple refers to each dimension in the cube, the tuple is *fully qualified*. Otherwise, it is *partially qualified*.

For an introduction, see "Tuples and Cubes" in *Using MDX with DeepSee*. For reference information, see "Tuple Expressions" in *DeepSee MDX Reference*.

unstructured data

Data that is written as text in a human language such as English or French. The iKnow semantic analysis engine analyzes unstructured data. This engine is built into Caché in the same way that DeepSee is. For a general introduction, see "Conceptual Overview," in *Using iKnow*.

You can use unstructured data within DeepSee cubes, if the source table for a cube includes a property that contains unstructured data. See "Using Unstructured Data in Cubes (iKnow)" in the *Advanced DeepSee Modeling Guide*.

widget

A rectangular area that lies within a dashboard and that (in most cases) displays data obtained from DeepSee. See *Creating DeepSee Dashboards*.