mitsdiscover

MitsMaker User Guide

Version 8.1
August 2012

Management Information Tools, Inc. DBA MITS
701 Fifth Avenue, Suite 2610
Seattle, WA 98104

Table of Contents

mitsdiscover

MitsMaker User Guide

Welcome to MitsMaker

Introduction

This document is presented to assist in developing MITS Discover applications using MitsMaker.

MitsMaker is a Windows-based component of MITS Discover, used to develop MITS Discover applications. The instructions in this document are based on developing MITS Discover applications using MitsMaker connected to a MITS.SYSTEM account on a MultiValue platform in a client/server environment. Once you have set up these functions and created a MITS Discover Hypercube, you can use the browser-based MITS Discover query tool to access the information in the Hypercube.

See the *MITS Discover User Guide* for more information about the operation of the MITS Discover browser client.

mitsdiscover

MitsMaker User Guide

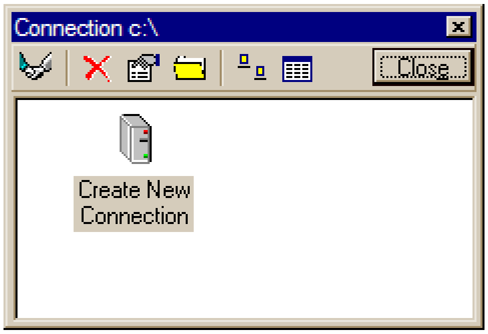
Configuration and Setup

Initial Setup

Before using any MITS Discover Client Software Application, you must first create a MITS Discover Software Connection Script. This script will need to be configured to initiate a TCP/IP telnet or RS-232 serial connection with the MultiValue source database and also to navigate the login process.

Connection Interface

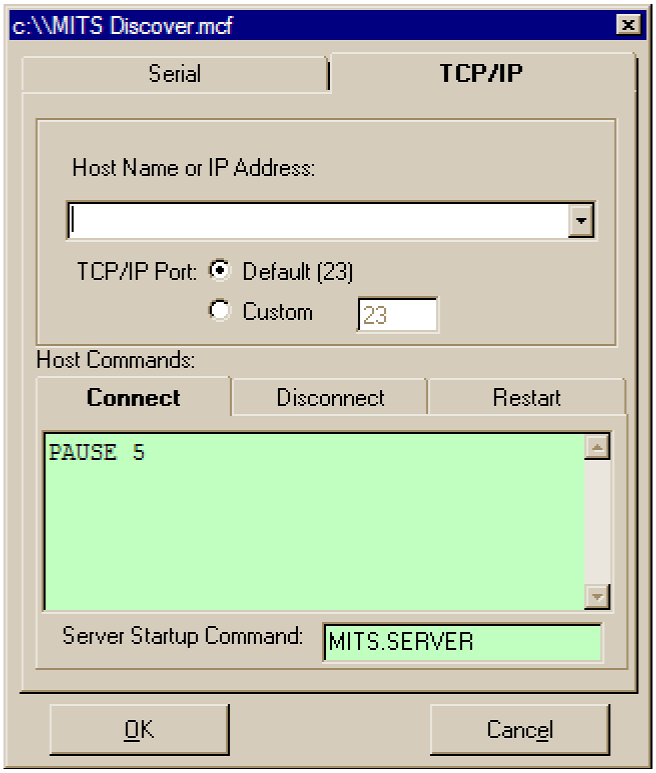
This is the initial Connection Interface window that appears when the MitsView software application is started. MITS Discover Connection Files (which have an extension of **.mcf**) are displayed here. These files contain the connection definitions and login scripts for MitsMaker.



Configuring Connections

To create a new connection for MITS, double-click the "Create New Connection" icon. You will be asked to enter a name for the new connection. Once this has been completed you will be taken to the Connection Configuration window.

NOTE: These instructions outline the setup of a TCP/IP connection to MitsServer. For information on setting up a serial or dialup connection, see "Serial Connection Setup" on page 6.



```
PAUSE 5
```

Server Startup Command: MITS.SERVER

In the **Host Name or IP Address** box, enter the host name or IP address for the computer that hosts the database where MitsServer is installed. Unless your MITS Discover administrator has specified another port for connecting to MitsServer, use the default telnet port of 23. See your network administrator if you need assistance with these settings.

Creating the Login/Logoff Scripts

The first tab in the lower "Host Commands" section will contain the green **Connect** script. The information in this box will be passed to the MitsServer during the login process. Keep in mind that information in this box is *case-sensitive*. The final goal of this login script is a TCL prompt that is logged to the MITS Discover operational account. (This is usually MITS.SYSTEM.) There are a few available keywords that can be used to make the connection script creation process easier:

Connection Script Syntax

`WAITFOR <string>`	The login script process will wait for <string> before proceeding to the next line.
`<< >>`	Any text label surrounded by double angle brackets specifies something that the connection process will prompt for.
`PAUSE`	Inserts a one second pause during the login process. PAUSE 2 will pause for two seconds, PAUSE 3 for three seconds, and so on.

Here's an example of the components of a common MITS Discover login script. Note that the actual commands for your script may vary based on the database and operating system you are using.

Connection Script Example

`WAITFOR Login:`	Waits for the string `Login:` before moving on to the next line of the script.
`<<Username>>`	Prompts for **Username**. You could also simply type the username here without the double angle brackets.
`WAITFOR Password:`	Waits for the string `Password:` before moving on to the next line of the script.
`<<Password>>`	Prompts for **Password**. You could also simply type the password here without the double angle brackets. Note: When prompting for a password, the letters will be obscured as they are typed into the prompt box.
`WAITFOR :`	Waits for a : prompt (UniData TCL prompt) before proceeding with the script.
`LOGTO MITS.SYSTEM`	Passes the literal string `LOGTO MITS.SYSTEM` to the host.

The contents of the **Server Startup Command** box will be processed as the last step of the login process. This will nearly always be the command **MITS.SERVER**.

The second tab in the lower Host Commands section contains the red **Disconnect** script. This box will be used to create your logoff script. The information in this box will be passed to the MitsServer when the client program is closed. Keep in mind that information in this box is *case-sensitive*.

NOTE: All of the same conventions that were used in the login script can also be used here with the exception of `WAITFOR`, which should not be used in a logoff script.

Here's an example of what a MITS Discover logoff script might look like:

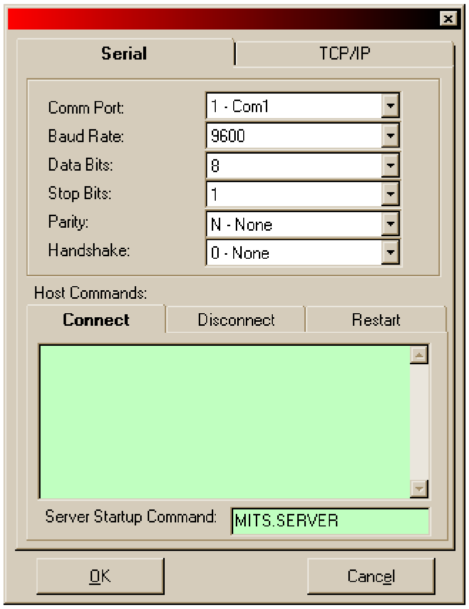
Disconnection Script Example

`END`	Log out of MITS Discover (this is necessary in every logoff script).
`QUIT`	Log out of the database (this command may vary).
`exit`	Log out of the operating system (this command may vary).
`PAUSE`	Wait for 1 second to allow the logoff process to complete before closing the connection.

The third tab in the Host Commands section contains the yellow **Restart** script. This box should almost always contain *MITS.SERVER* and nothing more.

Serial Connection Setup

If you are connecting to the host from a standard serial terminal cable or modem, select the **Serial** tab. The **Serial Host Connection** box appears:



Comm Port: 1 - Com1
Baud Rate: 9600
Data Bits: 8
Stop Bits: 1
Parity: N - None
Handshake: 0 - None
Server Startup Command: MITS.SERVER

The settings you enter in the upper part of this box (Comm Port, Baud Rate, Data Bits, Stop Bits, Parity, Handshake) should correspond to the settings of the communications port on the server where MitsServer is installed.

NOTE: For information about the configuration of the lower "Host Commands" section, see "Creating the Login/Logoff Scripts" on page 5.

Troubleshooting a MITS Discover Connection Attempt

The following table outlines some common troubleshooting tips for failed MITS Discover connection attempts.

MITS Discover Connection Attempt Troubleshooting

Problem	Solution
The MITS Discover Client states that the selected serial port is already in use	Be sure you have specified the correct serial port in the connection script. Close any processes, such as terminal applications, that are using the port.
Telnet session cannot be established	Check to see that the IP address or host name can be reached using the Windows PING command.
The connection script commands move faster than the server prompts	Extend the pause time between commands by adding PAUSE or WAITFOR commands to the script.

Closing MitsMaker

You can close MitsMaker in the following ways:

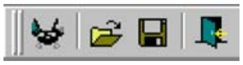
- Click the 🚪 button on the **Standard** toolbar.
- Select **Exit** from the **Connection** menu.
- Click the "X" in the top right corner of the MitsView program display.

When any MITS Discover client software is closed, the disconnect script will be executed, passing the commands on the Disconnect tab of the login script to MitsServer. Once the script completes, the MITS Discover client session ends.

MitsMaker User Guide

MitsMaker Toolbars & Program Options

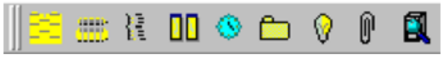
Main Toolbar



The main toolbar contains the following four buttons:

- The **Connect** button, used to initiate the MitsServer connection script selection box.
- The **Open** button. When you click **Open**, an open dialog box appears, enabling you to browse to the location of the MitsMaker specification file you wish to open.
- The **Save** button. If you previously defined a location to save the file, MitsMaker automatically saves it to that file. Otherwise, MitsMaker prompts you to enter the path and name of the file in which to store your existing selections.
- The **Exit** button. You can use this button to exit MitsMaker.

The Shortcut Toolbar

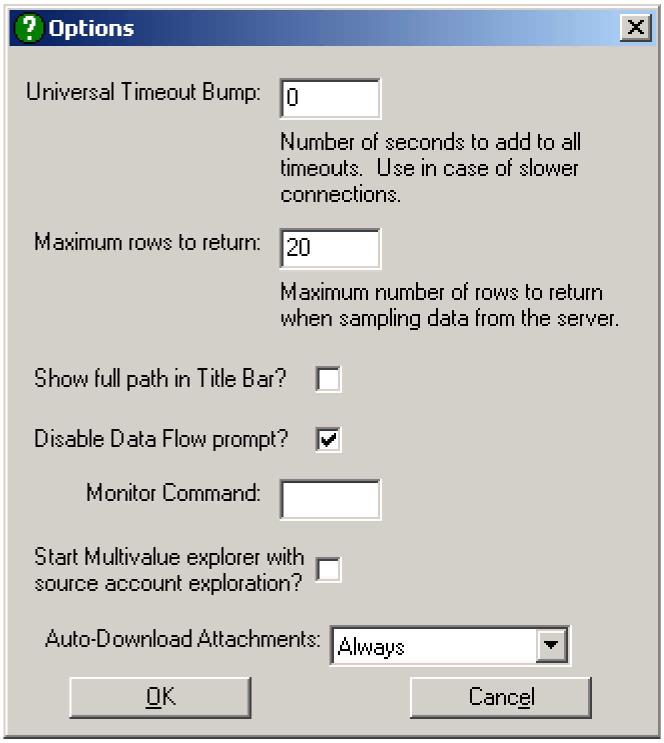


The **Shortcut** toolbar enables you to navigate MitsMaker. From left to right, the buttons are:

- Data Pool
- Data Element Assignment
- Paths
- Column Templates
- Eons
- File Sizes
- Create Application
- Attachments
- MultiValue Explorer

Options

There are several options that you can set to tailor MitsMaker to your needs. The following example illustrates the **Options** dialog box, which can be accessed from the **Tools** menu of MitsMaker:



Options

Universal Timeout Bump: 0

Number of seconds to add to all timeouts. Use in case of slower connections.

Maximum rows to return: 20

Maximum number of rows to return when sampling data from the server.

Show full path in Title Bar? □

Disable Data Flow prompt? ☑

Monitor Command:

Start Multivalue explorer with source account exploration? □

Auto-Download Attachments: Always

Universal Timeout Bump

The **Universal Timeout Bump** is a way to increase the amount of time MitsMaker waits for responses from the server. If your server or connection is slow, you may need to increase this value.

Rows to Return

Rows to Return defines how many rows the server will return when sampling data. This value can be any number between 1 and 99.

Show Full Path in Title Bar

Normally, MitsMaker only shows the name of the MitsMaker file in the title bar. If you want to view the full path, select the **Show full path in Title Bar** check box.

Disable Data Flow Prompt

Select the **Disable Data Flow Prompt** check box to change how the data flow window in the main window behaves. Normally, when you click the data flow window, an input box appears prompting for the command you wish to send to the server. The disadvantage of this method is you cannot send a simple carriage return and line feed. By disabling this prompt, you can enter simple commands. However, the window will not show the characters you are typing. Because of these limitations, MitsMaker allows you to select in which mode to run the data flow window.

Monitor Command

Monitor Command is used for diagnostics and support. You do not need to enter any values during normal MitsMaker operation. To view the communication between MitsServer and MitsMaker, enter `IO` into this box.

Start with Source Account Exploration

If you select the **Start MultiValue explorer with source account exploration** check box, the MultiValue Explorer window appears with the source account or path you specify, rather than the root of the system. Use this option to deal with remote accounts that are not normally visible in the MultiValue Explorer window.

Auto-Download Attachments

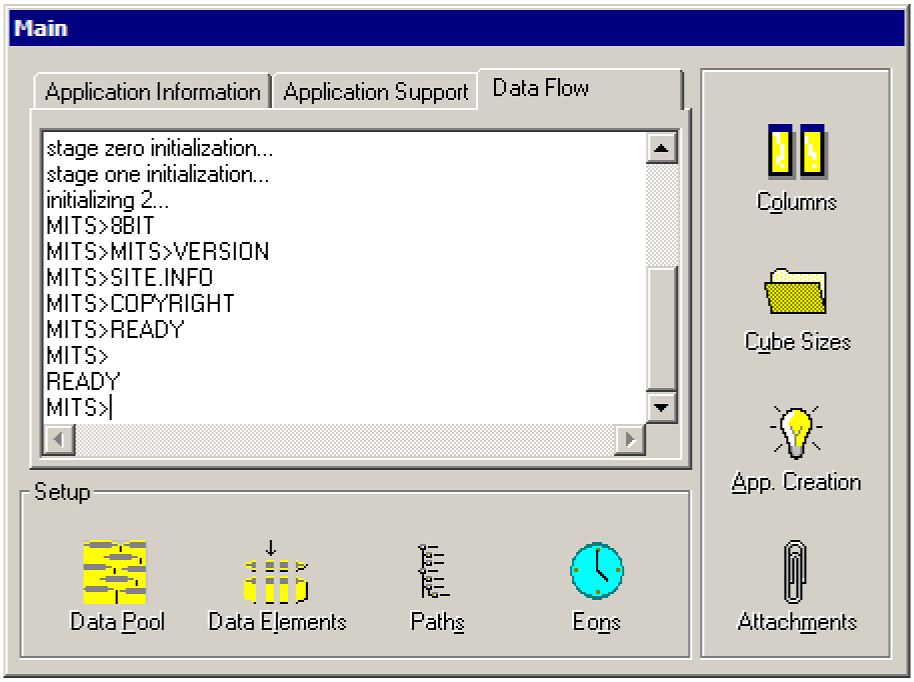
Attachments can be automatically downloaded when downloading a Hypercube Definition from MitsServer. The three settings for this box are:

- Always - MitsMaker will always attempt to download attachments
- Ask Me - MitsMaker will prompt you to download attachments with each definition download
- Never - MitsMaker will never download attachments

For more information about Attachments, see the section titled *Attachments* on page 62.

Data Flow

The Data Flow window enables you to view information the server is sending. When you click **Connect**, MitsMaker automatically displays the Data Flow window for you, as shown in the following example:



```
stage zero initialization...
stage one initialization...
initializing 2...
MITS>8BIT
MITS>MITS>VERSION
MITS>SITE.INFO
MITS>COPYRIGHT
MITS>READY
MITS>
READY
MITS>|
```

NOTE: The data flow window is not updated unless you select it's tab.

The **Application Creation** window also has a data flow window that is slightly different from the window illustrated in the previous example. You cannot enter information into that data flow window, but you can copy information from the window. This is useful for recording error conditions. When you click **Copy data flow** from the tools menu, MitsMaker copies the data flow window to the clipboard.

mitsdiscover

MitsMaker User Guide

Configuring Hypercubes

Initial Hypercube Configuration

Complete the steps described in this section to complete the initial configuration of a MITS Discover application using MitsMaker.

MitsMaker Hypercube Specifications

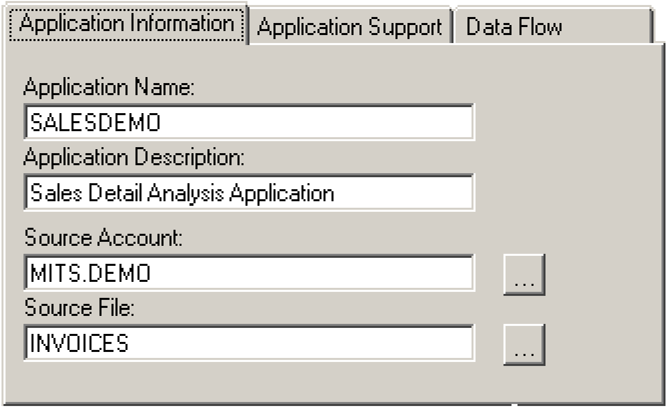
Download vs. Open

If the Hypercube you will be working on already exists on the MitsServer you are connecting to, go to the **File** menu and select **Download Application from Server**.

If you will be resuming work on a Hypercube that was previously started but never written to the server, go to the **File** menu and select **Open**. MitsMaker will prompt you for the location of the **.mkr** file for the previously started Hypercube definition.

The Application Information Tab

Start on the **Application Information** tab in the main MitsMaker window. Enter the appropriate identifying information for the application as shown in the following example:



Application Name:
SALESDEMO
Application Description:
Sales Detail Analysis Application
Source Account:
MITS.DEMO
Source File:
INVOICES

Application Name

This name is used as a keyword within MITS Discover to identify all the elements MitsMaker creates that relate to the Hypercube. Keep this name simple, yet descriptive–capital letters and digits only–since it becomes the suffix to numerous generated files.

Application Description

Enter a brief description of the application in the **Application Description** box. This is displayed when a user begins the **Create a New Flash Screen** process in MITS Discover.

Source Account

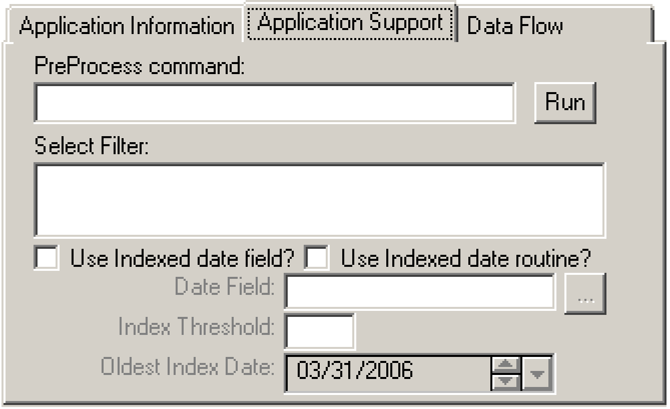
Specify the database account where the Hypercube's source transaction data is located.

Source File

Enter the name of the file in the source database account that will be used as the source data file for building the Hypercube.

The Application Support Tab

The **Application Support Tab** holds optional control information for the Hypercube as described below.



PreProcess Command

In the **PreProcess Command** box, you can enter an optional preprocess command. MitsMaker will execute this command one time before each build process. You can use a preprocess command in cases where an action needs to be performed before the Hypercube transformation process begins, such as data reformatting or consolidation. Click **Run** to execute the preprocess command immediately.

Select Filter

Enter an optional SELECT, SSELECT, or equivalent statement here. A Select Filter command produces an active select list of source IDs to be used by the transformation process.

NOTE: By default, MitsMaker selects all transactions that have not yet been transformed.

Use Indexed Date Field

Check this box if you are using a date field on your source file that has an active database index. Using an indexed date field dramatically reduces the time necessary to collect and filter the transactions that will be transformed during a build process.

Date Field

Enter the name of the date-referring dictionary item that has been indexed in the **Date Field** box.

Index Threshold

Enter the number of days you want MitsMaker to continue to check each date in the **Index Threshold** box. If no activity occurs for this threshold, MitsMaker considers the date “closed.”

Oldest Index Date

Enter a date in the **Oldest Indexed Date** box to limit how far back MitsMaker searches for transactions using the value entered in the **Indexed Date Field** box.

Use Index Date Routine

Instead of specifying a dictionary item to use as an index date, you may create a subroutine which returns a list of source file keys for a specific date.

NOTE: For an example of an index date routine, see the source code for the program MITS.SAMPLE.SELECT in the MITS.BP_SAMPLE file.

How "Index Dates" Work in MITS Discover

One of the following sets of potential components are used:

- INDEX.DATE.FIELD (with optional PRESELECT)
- INDEX.DATE.START (also known as *Oldest Index Date*)
- INDEX.DATE.THRESHOLD

-or-

- INDEX.DATE.ROUTINE
- INDEX.DATE.START (also known as *Oldest Index Date*)
- INDEX.DATE.THRESHOLD

For each date, MITS.MAKER.BUILD keeps track of the last time the date was inspected and the last time activity was found for that date. This is done with the application's MITS.ACTIVITY file.

Dates, from the INDEX.DATE.START to the current date, are reinspected only when one of the following is true:

- The date has never been inspected.
- A date has been inspected but never has had activity, and then only if the age of that date in days is less than or equal to the INDEX.DATE.THRESHOLD.
- The number of days between when the date was last inspected and when activity was last found is less than or equal to the INDEX.DATE.THRESHOLD.

Therefore, the INDEX.DATE.THRESHOLD is the measure of how long we'll keep inspecting a date since activity was last found for it.

For each date to be inspected, either a SELECT is performed against the INDEX.DATE.FIELD in the transaction file (followed by the PRESELECT, if provided) or, if an INDEX.DATE.ROUTINE is provided, that routine is called to get the transaction IDs using the arguments: (INDEX.DATE, TRANSACTION.IDS, ERROR). If an error is returned, it will be presented and the process will stop.

If an application's MITS.ACTIVITY file is ever cleared, all the dates from INDEX.DATE.START to the present will be reinspected, after which the index date processing will return to normal.

mitsdiscover

MitsMaker User Guide

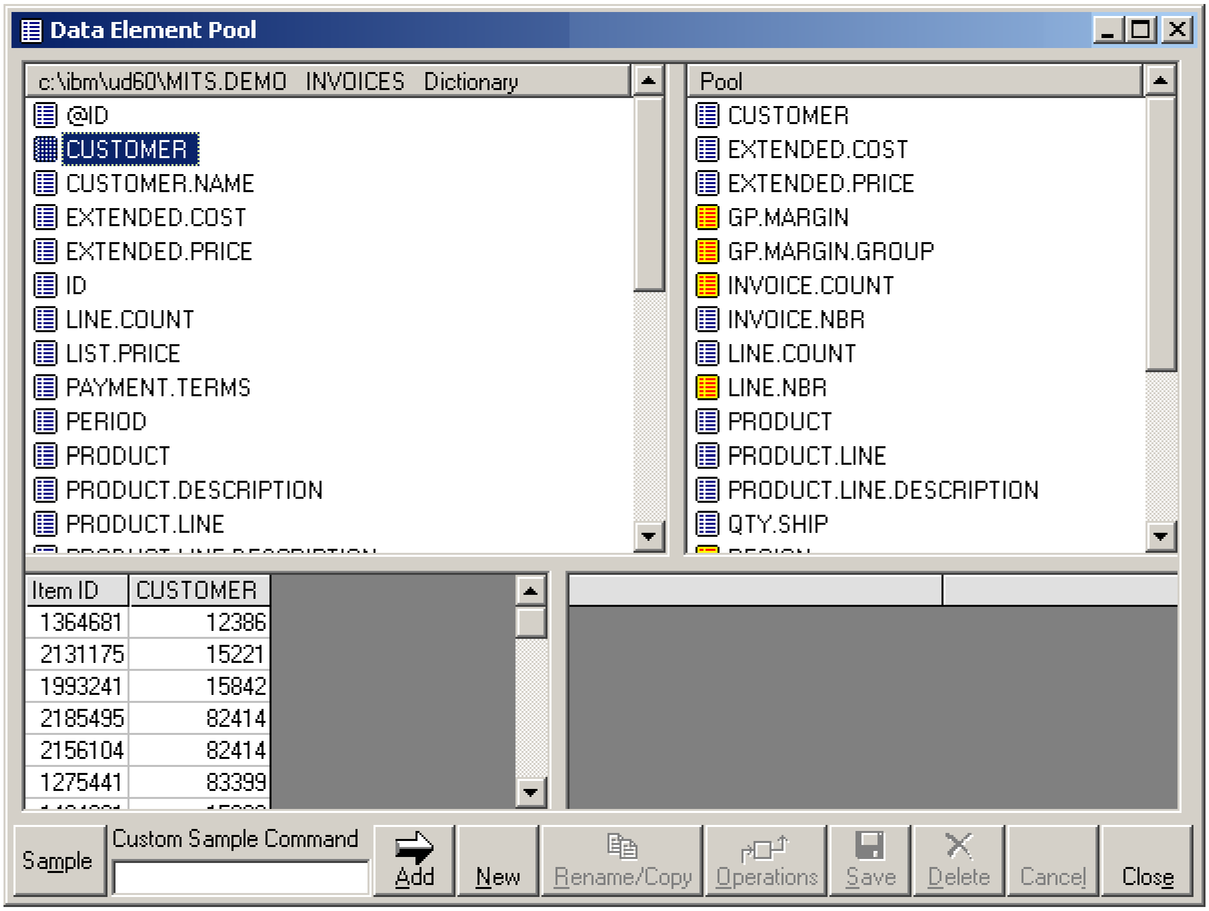
The MitsMaker Data Pool

Populating the MitsMaker Data Pool

MitsMaker uses an isolated dictionary, called the “data pool,” to extract data from the primary source file. The purpose of the data pool is to:

- Isolate changes that may need to be made to a dictionary item to support the MITS Discover Hypercube.
- Add information (including logic) to the dictionary items without modifying the operational system files or dictionary entries.

From the Main window, click the **Data Pool** button. A window similar to the following example appears:



c:\ibm\ud60\MITS.DEMO INVOICES Dictionary

- @ID
- CUSTOMER
- CUSTOMER.NAME
- EXTENDED.COST
- EXTENDED.PRICE
- ID
- LINE.COUNT
- LIST.PRICE
- PAYMENT.TERMS
- PERIOD
- PRODUCT
- PRODUCT.DESCRIPTION
- PRODUCT.LINE

Pool

- CUSTOMER
- EXTENDED.COST
- EXTENDED.PRICE
- GP.MARGIN
- GP.MARGIN.GROUP
- INVOICE.COUNT
- INVOICE.NBR
- LINE.COUNT
- LINE.NBR
- PRODUCT
- PRODUCT.LINE
- PRODUCT.LINE.DESCRIPTION
- QTY.SHIP

Item ID	CUSTOMER
1364681	12386
2131175	15221
1993241	15842
2185495	82414
2156104	82414
1275441	83399

Source Dictionary Entries - Upper Left Pane

The **Source Dictionary Entries** pane contains the dictionary items from the source file you specified in the application information. To see a sample of the data produced by any of these dictionary items, select the item and click **Sample**. To view the contents of the dictionary item, double-click the item.

Pool Dictionary Entries - Upper Right Pane

The **Pool** pane contains the dictionary items in the data pool. You can sample and view the contents of each element in the pool. You can also modify or delete these dictionary items.

WARNING: Any other Hypercube that uses the same source file could be sharing the same data pool. Use care when making changes to data pool entries.

You can capture a dictionary item in one of the following ways:

- By selecting one or more items from the left pane and clicking **Add**, or;
- By dragging the items from the left pane to the right pane. You can select multiple items by either holding down SHIFT or CTRL while clicking the items in the left pane.

When a pool dictionary entry has an operation attached to it, the dictionary entry is displayed with a yellow icon instead of the standard white icon. See “Operations” on page 21.

Sampled Data - Lower Left Pane

The lower left pane contains sample data using the dictionary item you select. You can sample either the source file dictionary or the pool dictionary by selecting the desired dictionary item and then clicking **Sample**. By default, MitsMaker presents 20 rows of data. You can change the default using the **Options** menu. Additionally, you can use the Custom Sample Command to run a custom Select Statement against the database before the sample is taken, allowing you to filter or sort the information to be sampled.

Dictionary Item - Lower Right Pane

The list in the lower right pane contains the dictionary item itself. To modify a dictionary item, select the item you want to modify and enter the new text. No changes will be made on the server until you click **Save**. If you want to cancel your changes, click **Cancel**.

Operations

Operations are optional field-specific subroutines that modify the data output of a dictionary item in the MitsMaker Data Pool. Selecting a data pool item and clicking **Operations** will bring up an Operation generation window where your operation can be entered. You can enter BASIC subroutine logic in the operation. MitsMaker then compiles the subroutine and runs it each time it uses that data pool element in an extraction.

Rename/Copy

The **Rename/Copy** function enables you to rename or copy a dictionary item in the data pool. When you select a pool item, click this button, type in a new item name, and hit enter, you will be asked if you wish to rename this item or create a duplicate item with a new name. Click the

Copy button to create a copy of the original item, and click the **Rename** button to create a duplicate with a new name.

WARNING: Multiple applications may be sharing the same data pool. Use care when you change data pool entries if you have any other Hypercubes that use the same source file.

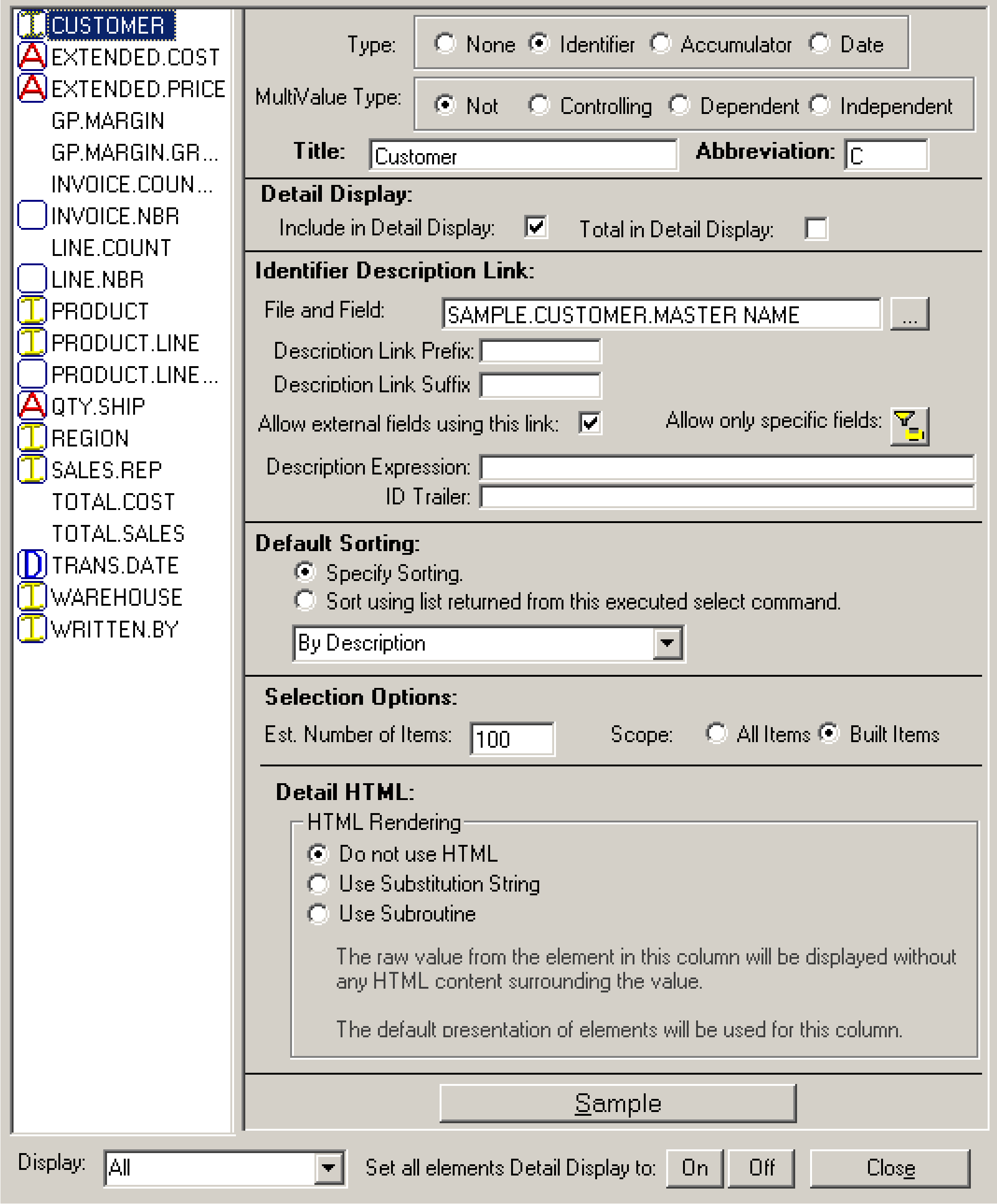
mitsdiscover

MitsMaker User Guide

Data Element Assignment

Assigning Data Element Roles

When you open the Data Elements window by clicking the **Data Elements** button on the Main window, the **Data Element Assignment** window appears. This is where the dictionary pool entries are assigned a purpose within the Hypercube. MitsMaker displays each of the items from the data pool in the left pane of the window. As these data pool items are assigned Data Element roles, an icon will appear to the left of the item name and the **Properties** form will change to display the fields pertinent to that data element type. Here's an example of a Data Element Assignment form showing a **Properties** form for an *identifier* data element:



I CUSTOMER
A EXTENDED.COST
A EXTENDED.PRICE
GP.MARGIN
GP.MARGIN.GR...
INVOICE.COUN...
INVOICE.NBR
LINE.COUNT
LINE.NBR
I PRODUCT
I PRODUCT.LINE
PRODUCT.LINE...
A QTY.SHIP
I REGION
I SALES.REP
TOTAL.COST
TOTAL.SALES
D TRANS.DATE
I WAREHOUSE
I WRITTEN.BY

Type: ○ None ● Identifier ○ Accumulator ○ Date

MultiValue Type: ● Not ○ Controlling ○ Dependent ○ Independent

Title: Customer **Abbreviation:** C

Detail Display:

Include in Detail Display: ☑ Total in Detail Display: ☐

Identifier Description Link:

File and Field: SAMPLE.CUSTOMER.MASTER NAME ...

Description Link Prefix:

Description Link Suffix:

Allow external fields using this link: ☑ Allow only specific fields:

Description Expression:

ID Trailer:

Default Sorting:

● Specify Sorting.

○ Sort using list returned from this executed select command.

By Description

Selection Options:

Est. Number of Items: 100 Scope: ○ All Items ● Built Items

Detail HTML:

HTML Rendering

● Do not use HTML

○ Use Substitution String

○ Use Subroutine

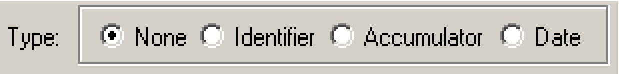
The raw value from the element in this column will be displayed without any HTML content surrounding the value.

The default presentation of elements will be used for this column.

Display: All

Set all elements Detail Display to:

Data Element Type



Type: ● None ○ Identifier ○ Accumulator ○ Date

Each data element that will be added to the Hypercube must be assigned a *Type*. To assign a type for a data element, select the data element from the **Data Elements** list and select the radio button to the left of the desired data element type. When you select an item, the Properties form will change based on the selected type.

None

A type of *None* applies to data elements that are not being used or that are being used only for detail display.

Identifier

MitsMaker uses the *identifier* data element type for those elements that form the drill-down dimensions of the Hypercube and the horizontal rows of flash screens. Some examples of common identifiers are customer, warehouse, general ledger account, and product. For more information on setting up an identifier, see “Configuring Identifier Elements” on page 26.

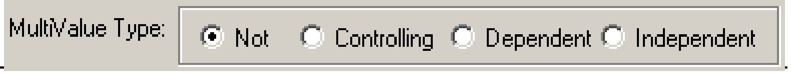
Accumulator

Unlike an identifier, an accumulator is a measurement, such as sales amount, cost of sales, transaction count, and so forth. Accumulators are always numeric values, and are the foundation for the development of most vertical flash screen columns. MITS Discover keeps track of totals for each application's accumulators by date, based on all the configured identifier combinations. For more information on setting up an accumulator, see “Configuring Accumulator Elements” on page 29.

Date

Each Hypercube can have only one date element. This date is used to create the virtual accumulation “buckets” which are maintained to hold the accumulated data displayed in the MITS Discover flash screen columns.

MultiValue Type

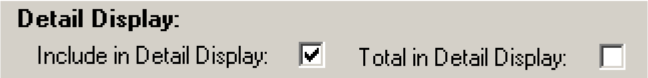


MultiValue Type: ● Not ○ Controlling ○ Dependent ○ Independent

If any of the fields you are using for your Hypercube contain multivalued data, you must declare the multivalue relationship. The following general rules apply:

- If any of your data elements are multivalued, all of the Hypercube’s accumulators must also be multivalued.
- There can never be more than one controlling multivalue.
- If there are any dependent multivalues, there must also be a controlling multivalue.
- The Independent type can be used for multivalued fields such as comments that have no relationship to other multivalued fields.

Detail Display



Detail Display:

Include in Detail Display: ☑ Total in Detail Display: □

Each data element (assigned or unassigned) can be added as a column in the MITS Discover **detail display** by selecting the check box labeled **Include in Detail Display**. At configured points of a Hypercube drill-down path, you can specify that detail display will be available to the user. (This is set up when you are configuring the drill-down paths in Chapter .) The Detail Display feature allows users to double-click a column cell and see information specific to the individual source file transactions that contributed to the number that was clicked.

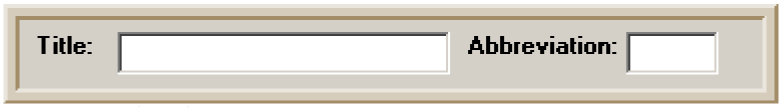
If the detail display element is a numeric value, you can choose to have the column totalled in the detail display screen by selecting the **Total in Detail Display** check box.

NOTE: All elements in the Data Elements pane can be included in Detail Display by clicking the On button at the bottom of the page, next to the heading "Set all elements Detail Display to:". Likewise, Detail Display can be removed from all elements by clicking the Off button.

Configuring Identifier Elements

Identifier elements (see "Identifier" on page 25) include a number of settings that must be defined in order for the identifier values in your Hypercube flash screens to be displayed in a meaningful way.

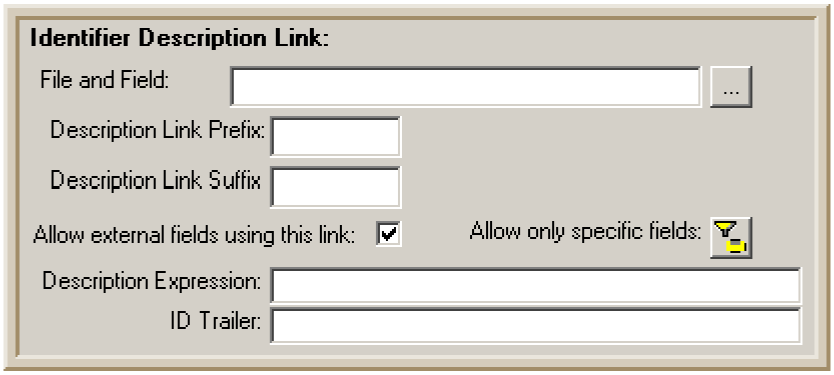
Title and Abbreviation



The identifier title will be used by MITS Discover when referring to this identifier for presentation purposes where a more verbose description of the identifier is required. The identifier abbreviation will be used when referring to this identifier in MQL statements.

Identifier Description Link

The fields in this section of the Identifier Element Assignment screen are used to control what the user sees in the identifier column of a flash screen. This includes the identifier descriptions for this identifier as well as the formatting of the identifier IDs.



Allow external fields using this link: ☑

Using an Existing Description on a Supporting File

Your data pool dictionaries and operations determine what specific data is built into your Hypercube. In most situations, your data pool dictionaries will be returning numeric data that can be used as the key to a supporting file. For example, if you are working with a CUSTOMER identifier, the data pool dictionary should return the customer number. This customer number can then be used at flash-time to retrieve the customer description from the customer master file.

If the values being written to the Hypercube for an identifier represent the database key to a supporting file where the description for that identifier is stored, simply enter the file name and the name of the dictionary item in the **File and Field** box.

Description Link Prefix/Description Link Suffix

If the value stored in the Hypercube requires the addition of a prefix or suffix before it can be used as a key for reading from a supporting file (assuming this prefix or suffix is the same for every value in the Hypercube), enter a prefix and/or suffix into the **Description Link Prefix** and/or **Description Link Suffix** box(es).

"Allow External Fields Using This Link"

When using the File and Field option for retrieving an identifier description, you can grant your users access to some or all of the other fields in the dictionary of the supporting file by selecting the **Allow external fields using this link** check box. The users will then be able to populate flash screen columns with this data.

Adding an External Fields Filter

You can specify which dictionary records the users will be able to use by clicking the **Allow only specific fields** button. You will be asked if you want to create an external filter for this identifier. Click the **Yes** button, and you will be taken to a screen that looks very similar to the Data Pool screen showing the dictionary from the supporting file specified in the **File and Field** box. Drag the dictionary items you would like to make available to your users from the box in the upper left to the box in the upper right, and click the **Close** button when you have finished adding records.

Using a Description Expression to Create an Identifier Description

A Description Expression allows you to create a basic expression that specifies to MITS Discover exactly how to generate the description for that identifier. Here are some reasons you may choose to use Description Expression instead of File and Field:

- The identifier value in the Hypercube is the value you want also displayed as the description. In this case, simply place the identifier abbreviation in the **Description Expression** box.
- Multiple reads will be required to acquire the identifier description. In this case, use a nested READV statement. Notice that the inner READV statement becomes the "identAbbrev" segment for the outer READV statement:

```
READV(READV(identAbbrev, "FileName1", attribNum1), "FileName2", attribNum2)
```

- More complex logic will be required to derive the identifier description. In this case, use the IDENT.CALLSUB function to call a subroutine that will produce and return the proper description. The syntax is as follows, where **subName** should be replaced with the name of a subroutine that has been compiled and cataloged locally to the MITS Discover operational account, and **identAbbrev** should be replaced with the identifier abbreviation:

```
IDENT.CALLSUB('subName',identAbbrev)
```

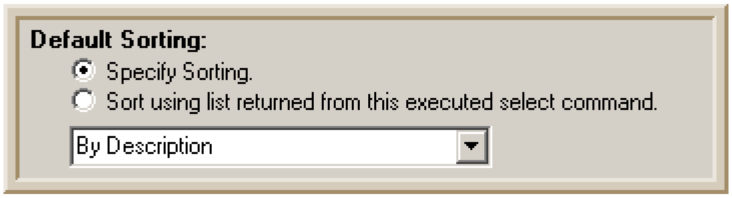
NOTE: The created subroutine must handle exactly two arguments. The first argument represents the data being passed into the subroutine, and the second argument represents the data being passed back out of the subroutine.

ID Trailer Routine

A Trailer Routine is a subroutine that is applied to the values stored in the Hypercube prior to displaying them when the **Show IDs** feature is active in MITS Discover. If this box is left blank, the value that is stored in the Hypercube for that identifier will be displayed in parentheses after the identifier description when the **Show IDs** feature (which is a user-specified feature) is turned on. To display something other than that when this feature is turned on, create a subroutine that handles only one argument. The identifier value will be passed into the subroutine, and the value you would like displayed as the identifier ID should be passed back out.

Default Sorting

This is where you can specify how the values in the identifier lists within MITS Discover flash screens will be sorted.



Default Sorting:

● Specify Sorting.

○ Sort using list returned from this executed select command.

By Description ▾

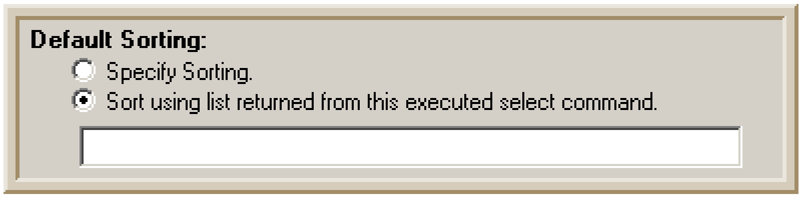
Specify Sorting

You can specify the following sort options for identifier values:

- By description
- Ascending order, left justified by ID
- Ascending order, right justified by ID
- Descending order, left justified by ID
- Descending order, right justified by ID

Sort using list returned from this executed select command

The other option for sorting involves the creation of an SSELECT statement that will be used to sort the values. Simply select the **Sort using list returned from this executed select command** radio button and type a valid SSELECT statement in the box that appears. The SSELECT command must be valid when run against the file where the identifier description resides.

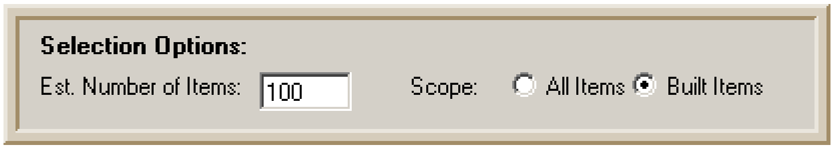


Default Sorting:

○ Specify Sorting.

● Sort using list returned from this executed select command.

Selection Options



Selection Options:

Est. Number of Items: 100 Scope: ○ All Items ● Built Items

MITS Discover provides a mechanism that allows users to select only specific identifier elements for exploration. This is called *Select Exploration*. Users select the elements for their exploration from a sorted list. MitsServer indexes these sorted lists.

The number entered into the **Estimated Number of Items** box should represent an estimate of how many unique values this identifier's index will contain. The **Scope** field allows you to specify that these indexes should include only those values found in the Hypercube (**Built Items**) or all of the values found on the source file (**All Items**).

Configuring Accumulator Elements

For more information about Accumulators, see the section titled "Accumulator" on page 25.

1. Select the data element from the **Data Elements** list.
2. Select the **Accumulator** radio button in the **Type** box. (See "Data Element Type" on page 25.)
3. Configure the multivalue type by selecting the proper radio button in the **MultiValue Type** box. (See "MultiValue Type" on page 25.)
4. Select whether the accumulator should be included as a column in the detail display, and whether the column should be totaled in the detail display. (See "Detail Display" on page 26.)
5. Configure the **Detail HTML** feature if required. (See "Detail Display HTML" on page 30.)

Configuring a Date Field Element

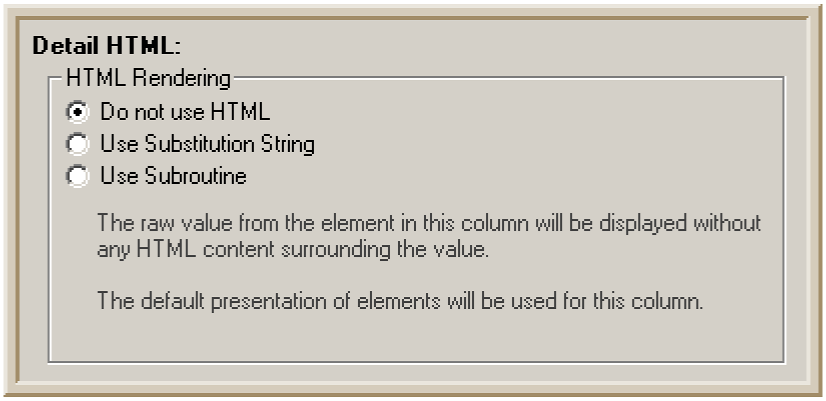
1. Select the data element from the **Data Elements** list.
2. Select the **Date** radio button in the **Type** box. (See "Data Element Type" on page 25.)
3. Configure the multivalue type by selecting the proper radio button in the **MultiValue Type** box. (See "MultiValue Type" on page 25.)
4. Select whether the date field should be included as a column in the detail display. (See "Detail Display" on page 26.)
5. Configure the **Detail HTML** feature if required. (See "Detail Display HTML" on page 30.)

Detail Display HTML

You can add hypertext markup language (HTML) rendering to the detail display cell values. (For more information about the detail display feature, see “Detail Display” on page 26.) This new feature allows you to modify the display of the information in the detail display for a given element using HTML. For example, you could modify the colors or font style, or even place hyperlinks into the detail display columns. There are three options that can be set separately for each element:

Do not use HTML

No HTML rendering will occur for this detail display column. Any HTML found in this column will be ignored and will be displayed as HTML code.



Detail HTML:

HTML Rendering

● Do not use HTML
○ Use Substitution String
○ Use Subroutine

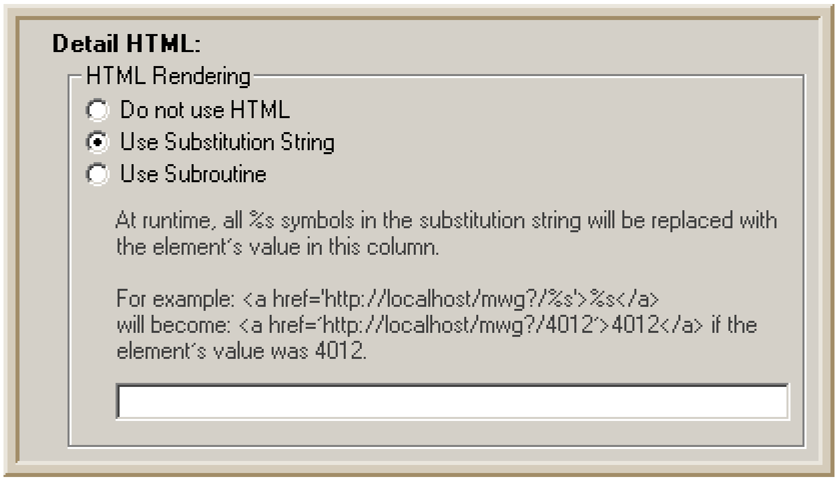
The raw value from the element in this column will be displayed without any HTML content surrounding the value.

The default presentation of elements will be used for this column.

Use Substitution String

Enter an HTML string into the box that utilizes the variable %s. The %s variable will be swapped out at run-time for the original cell value. For example, to create a hyperlink that includes the cell value, type the following:

```
<a href='http://
localhost/mwg?/
%s'>%s</a>
```



Detail HTML:

HTML Rendering

○ Do not use HTML
● Use Substitution String
○ Use Subroutine

At runtime, all %s symbols in the substitution string will be replaced with the element's value in this column.

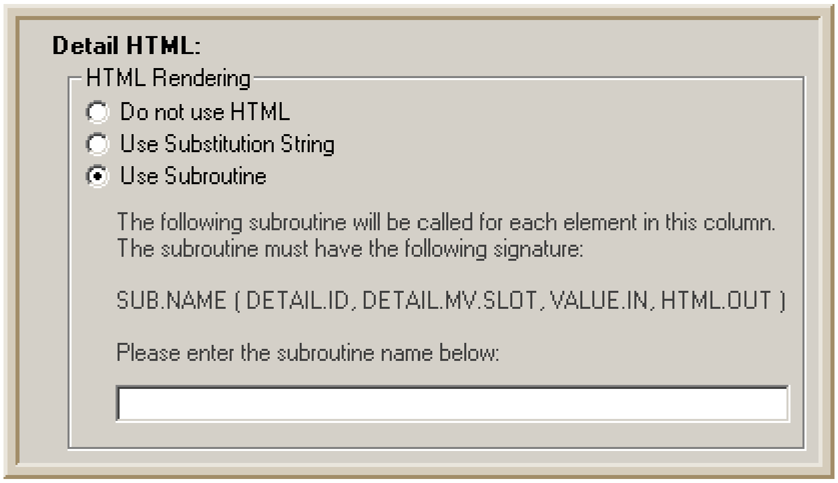
For example: <a href='http://localhost/mwg?/%s'>%s</a>
will become: <a href='http://localhost/mwg?/4012'>4012</a> if the element's value was 4012.

If the cell value is *4012*, this will be swapped out at run-time with the following HTML:

```
<a href='http://localhost/mwg?/4012'>4012</a>
```

It will then be rendered to display only the value *4012* as a hyperlink that will take the user to the address *http://localhost/mwg?/4012*.

Use Subroutine



Detail HTML:

HTML Rendering

○ Do not use HTML
○ Use Substitution String
● Use Subroutine

The following subroutine will be called for each element in this column.
The subroutine must have the following signature:

SUB.NAME (DETAIL.ID, DETAIL.MV.SLOT, VALUE.IN, HTML.OUT)

Please enter the subroutine name below:

If your specific HTML generation needs dictate more complexity, you can create a subroutine that will complete the logic required to create the necessary HTML. The subroutine should be compiled and cataloged in the MITS Discover operational account, and must have the following signature:

```
SUB.NAME(DETAIL.ID, DETAIL.MV.SLOT, VALUE.IN, HTML.OUT)
```

Here are the definitions of the elements in this signature:

- **DETAIL.ID**: The ID of the data element, as configured in the data pool, is passed into the subroutine in this variable. (This is listed on the left side of the **Data Elements** window.) This allows you to use the same subroutine for multiple variables.
- **DETAIL.MV.SLOT**: If the element value is being read from a multivalued field, the multivalue "slot" will be passed into the subroutine in this variable. For example, the first multivalue will have a DETAIL.MV.SLOT of "1," the second multivalue will have a DETAIL.MV.SLOT of "2," and so on.
- **VALUE.IN**: The cell value is passed into the subroutine in this variable. This is the same value that is passed in the %s variable when the **Use Substitution String** method is selected.
- **HTML.OUT**: The HTML string that will be rendered in the cell is passed out of the subroutine in this variable.

The Sample Button

The **Sample** button enables you to look at a small sampling of data from the selected element.

Data Element Filtering

The **Display** drop-down menu allows you to filter which elements are visible within the **Data Elements** pane. The options are:

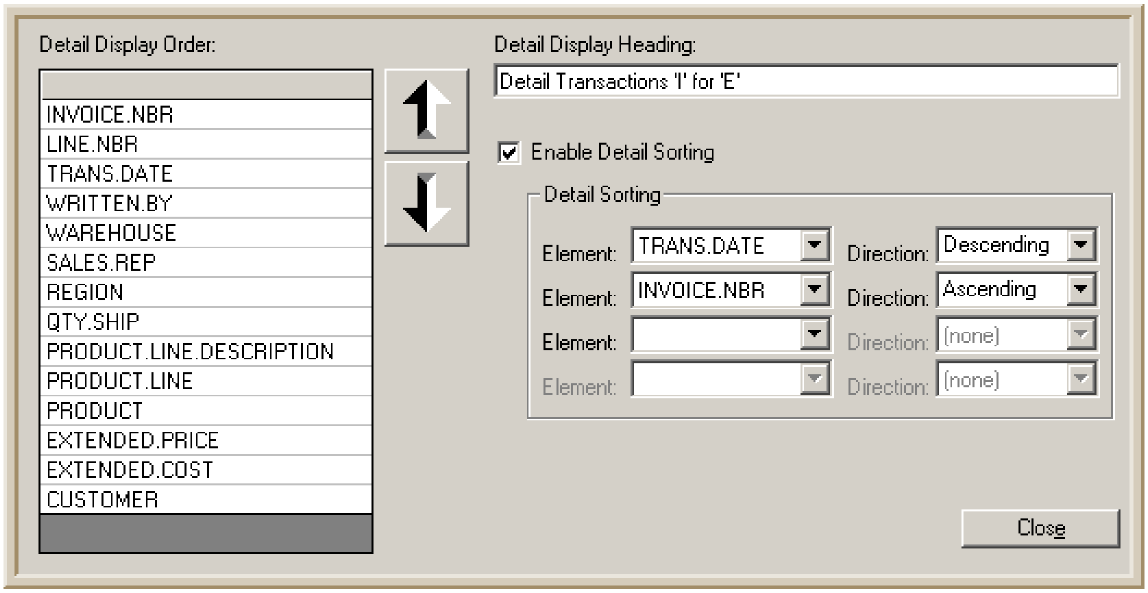
- All
- Identifiers Only
- Accumulators Only
- Date Only
- Detail Display
- All Assigned
- All Unassigned

Detail Display Order and Sorting

The **Detail Display** tab is used for formatting the column order, heading, and default sorting in the detail display.

Detail Display Order

Change the order of these fields by selecting a field and clicking the up and down arrows to the right of the element list. The top element in this list will be displayed as the left most column in the detail display, and the bottom element in this list will be displayed as the right most column in the detail display.



Detail Display Order:

INVOICE.NBR
LINE.NBR
TRANS.DATE
WRITTEN.BY
WAREHOUSE
SALES.REP
REGION
QTY.SHIP
PRODUCT.LINE.DESCRIPTION
PRODUCT.LINE
PRODUCT
EXTENDED.PRICE
EXTENDED.COST
CUSTOMER

Detail Display Heading: Detail Transactions 'I' for 'E'

☑ Enable Detail Sorting

Detail Sorting

Element: TRANS.DATE Direction: Descending
Element: INVOICE.NBR Direction: Ascending
Element: Direction: (none)
Element: Direction: (none)

Detail Display Heading

Each detail display can have a heading that is customized based on a number of variables. You can use the following variables when creating the heading:

- 'E' – is replaced with a description of the eon used in the column[1]
- 'I' – is replaced with the identifier descriptions
- 'D' – is replaced with the report date
- 'T' – is replaced with the report time
- ' ' – is replaced with just a space

NOTE: You can use more than one code in the heading, but be sure to wrap the variables in single quotes.

Here are a few examples:

```
Detail Display for 'I' for the time span 'E'
Identifiers: 'I' Span: 'E' Date & Time: 'D T'
```

Detail Display Sorting

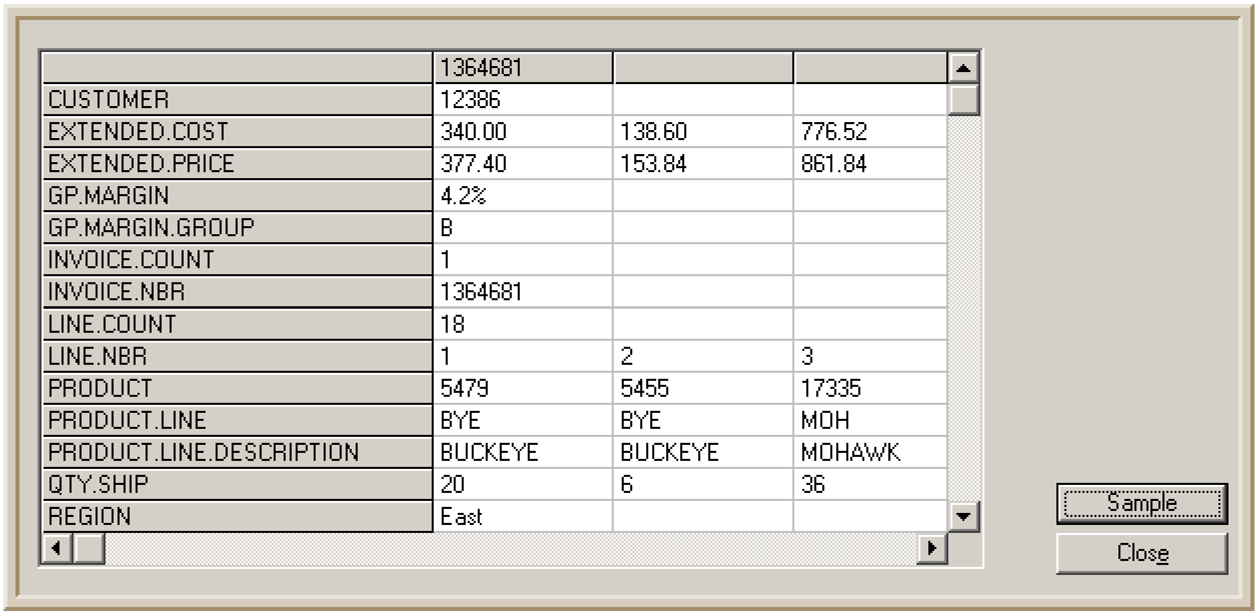
Select the **Enable Detail Sorting** box to turn on the detail display sorting feature. Select a detail display element from the first **Element** drop-down list, and select a sorting direction from the first **Direction** drop-down list. If a secondary sort is also desired, select another element from the next **Element** drop-down list, and the appropriate direction from the second **Direction** drop-down list. Repeat this for the third and fourth sort elements if desired.

In the example above, the detail display will be sorted first in descending order by TRANS.DATE, then in ascending order by INVOICE.NBR by default. The user can then modify the sorting if desired.

[1] For more information about eons, see "What is an Eon?" on page 42.

Data Extraction Sample

To produce a sample of each of the presently visible data elements on the **Data Element Assignment** tab, select the **Data Extraction Sample** tab and click **Sample**.



	1364681		
CUSTOMER	12386		
EXTENDED.COST	340.00	138.60	776.52
EXTENDED.PRICE	377.40	153.84	861.84
GP.MARGIN	4.2%		
GP.MARGIN.GROUP	B		
INVOICE.COUNT	1		
INVOICE.NBR	1364681		
LINE.COUNT	18		
LINE.NBR	1	2	3
PRODUCT	5479	5455	17335
PRODUCT.LINE	BYE	BYE	MOH
PRODUCT.LINE.DESCRIPTION	BUCKEYE	BUCKEYE	MOHAWK
QTY.SHIP	20	6	36
REGION	East		

mitsdiscover

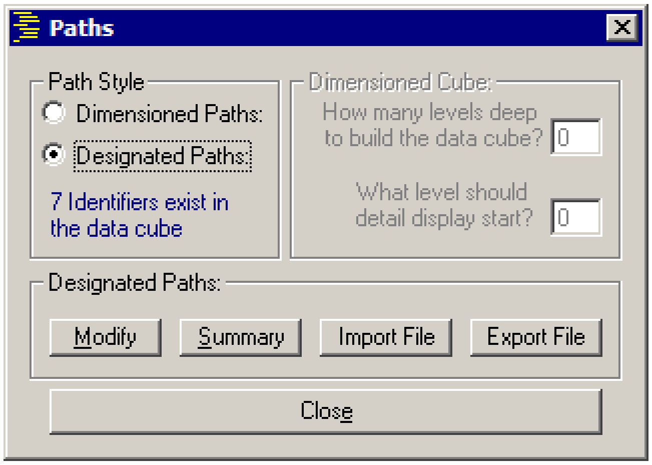
MitsMaker User Guide

Designating Drill-Down Paths

Paths

The structure of a MITS Discover Hypercube is based on how users need to drill into their data, and at which point they require the underlying transaction detail.

The easiest way to define the "shape" of a Hypercube is by specifying a "dimensioned" Hypercube, where you define only the maximum drill-down depth and the depth where the underlying transaction detail becomes available. However, *the dimensioned approach may cause unacceptably long transformation times*. If your Hypercube has more than a few identifiers and/or the length of transformation times is an important factor for your organization, you should use a **Designated Path** approach.



Paths

Path Style

○ Dimensioned Paths:

● Designated Paths:

7 Identifiers exist in the data cube

Dimensioned Cube:

How many levels deep to build the data cube? 0

What level should detail display start? 0

Designated Paths:

In the case of designated paths, an integrated version of the MitsTree utility is used to specify every drill-down path that will be available to the users. You will also be able to specify the location in each path where transaction detail becomes available.

Dimensioned Paths

If you select the **Dimensioned Path** option, there are two sub-options: **How many levels deep to build the data cube** and **What level should detail display start**.

Designated Paths

If you select the Designated Paths option, the **Designated Paths** buttons are enabled.

Modify

Click **Modify** if you want to modify the path tree.

NOTE: Prior to MITS Discover 7.0, changing the drill-down paths necessitated performing a "Major Remake". As of MITS Discover 7.0, that is no longer required.

Import File

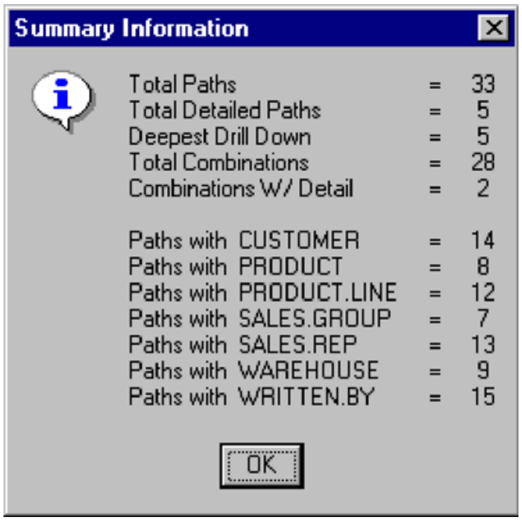
Click **Import File** if you want to import external MitsTree files.

Export File

Click **Export File** if you want to change the MitsTree configuration stored in MitsMaker into a .tre file for the external MitsTree program.

Summary

Click **Summary** to display an overview of the paths you have indicated with MitsTree, as shown in the following example:



```
Summary Information

Total Paths                 =  33
Total Detailed Paths        =   5
Deepest Drill Down          =   5
Total Combinations          =  28
Combinations W/ Detail      =   2

Paths with  CUSTOMER        =  14
Paths with  PRODUCT         =   8
Paths with  PRODUCT.LINE    =  12
Paths with  SALES.GROUP     =   7
Paths with  SALES.REP       =  13
Paths with  WAREHOUSE       =   9
Paths with  WRITTEN.BY      =  15
```

- **Total Paths** reports the number of paths you selected.
- **Total Detailed Paths** is the number of paths where detail has been made available.
- **Deepest Drill-Down** indicates the longest path into the Hypercube.
- **Total Combinations** is the number of updates involved when transforming each transaction.
- **Combinations W/ Detail** is the same as Total Combinations, except these combinations involve access to the underlying transaction detail.
- **Paths with…** shows which identifiers are used and in how many paths they are involved.

NOTE: The *Total Combinations* value is more critical in its effect on build speeds than the *Total Paths* value. If you are trimming your designated paths due to excessive build times, keep track of and compare the difference in the *Total Combinations* value rather than in the *Total Paths* value.

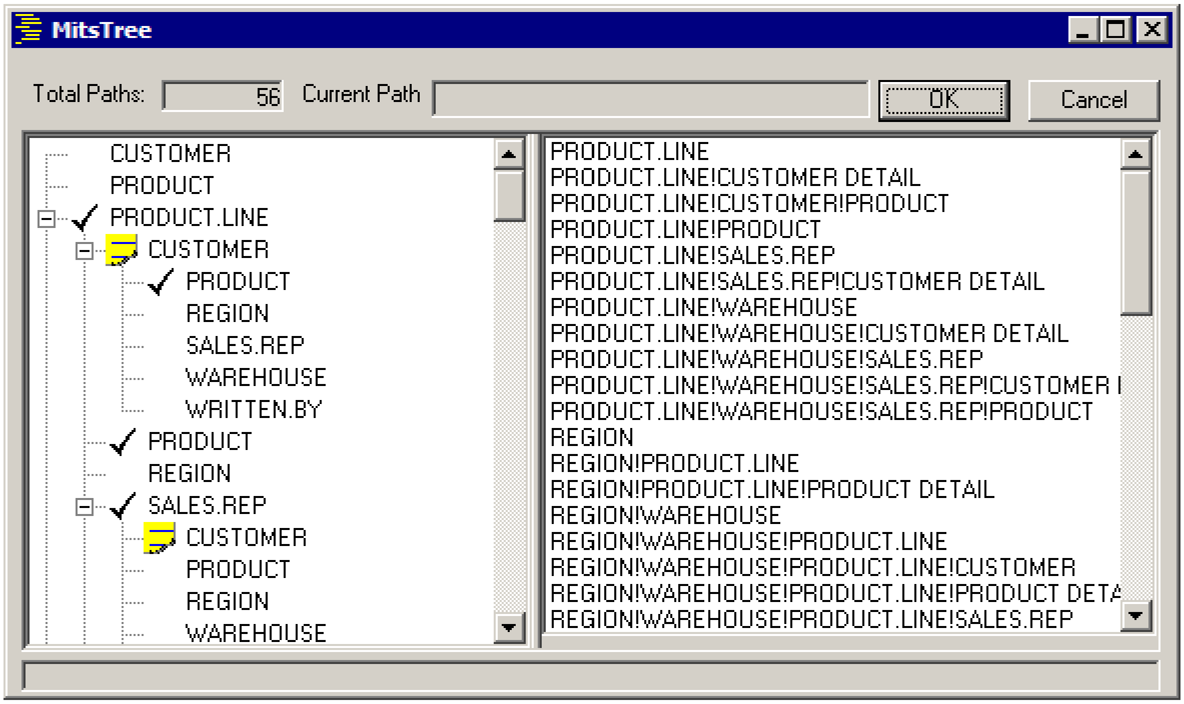
MitsTree

MitsTree is the facility used both internally in MitsMaker and externally, as the MitsTree application, to modify designated paths. Click **Modify** when defining designated paths to access this facility.

MitsTree uses the following basic commands:

- ✓ or SPACE selects or clears a path.
- ▤ or "D" toggles detail display on or off.
- ⊞ or ENTER expands the current path.
- ◁ ▷ You can use these two buttons, contained on the tool bar, to expand or collapse all the paths in the tree. The right button collapses the tree, while the left button expands the tree. If you are using the standalone MitsTree utility, you can also collapse and expand the tree with the Ctrl-C and Ctrl-E keyboard shortcuts, respectively.

The following example displays two types of paths. A normal path is represented by a check mark, and a path that provides underlying transaction detail is represented by the yellow icon.



Total Paths: 56 Current Path

```
CUSTOMER
PRODUCT
⊟ ✓ PRODUCT.LINE
    ⊟ ▨ CUSTOMER
        ✓ PRODUCT
          REGION
          SALES.REP
          WAREHOUSE
          WRITTEN.BY
    ✓ PRODUCT
      REGION
    ⊟ ✓ SALES.REP
        ▨ CUSTOMER
          PRODUCT
          REGION
          WAREHOUSE
```

```
PRODUCT.LINE
PRODUCT.LINE!CUSTOMER DETAIL
PRODUCT.LINE!CUSTOMER!PRODUCT
PRODUCT.LINE!PRODUCT
PRODUCT.LINE!SALES.REP
PRODUCT.LINE!SALES.REP!CUSTOMER DETAIL
PRODUCT.LINE!WAREHOUSE
PRODUCT.LINE!WAREHOUSE!CUSTOMER DETAIL
PRODUCT.LINE!WAREHOUSE!SALES.REP
PRODUCT.LINE!WAREHOUSE!SALES.REP!CUSTOMER I
PRODUCT.LINE!WAREHOUSE!SALES.REP!PRODUCT
REGION
REGION!PRODUCT.LINE
REGION!PRODUCT.LINE!PRODUCT DETAIL
REGION!WAREHOUSE
REGION!WAREHOUSE!PRODUCT.LINE
REGION!WAREHOUSE!PRODUCT.LINE!CUSTOMER
REGION!WAREHOUSE!PRODUCT.LINE!PRODUCT DETA
REGION!WAREHOUSE!PRODUCT.LINE!SALES.REP
```

Drill-Down Path Usage Logging

MITS Discover has the ability to keep a log of the drill-down paths that are used most often in a Hypercube so that drill-down paths which are not used regularly can be removed. Removing seldom-used drill-down paths from a Hypercube can greatly improve build speeds and lessen disk space requirements for that Hypercube.

Important Files and Commands:

- **MITS.DRILLDOWN.USAGE** - This program is run from a database prompt that has been logged to the MITS Discover operational account. When run, the drill-down path usage statistics are displayed. This utility is described in more detail below.

- **FLASH.LOGGING.ENABLED** - This item must exist in the site configuration file (the MITS.CONFIG file) for this feature to be active.

- **DRILLDOWN.HISTORY.MAXIMUM** - By default, only 1000 drill-down actions are maintained by the system. This number can be changed by modifying the DRILLDOWN.HISTORY.MAXIMUM item in the site configuration file (the MITS.CONFIG file). All client sessions (including MITS Discover gateway host connections) need to be restarted in order for a change to this setting to take effect.

Specifications:

To set up this feature, create an empty item in the MITS.CONFIG file within the MITS Discover operational account named **FLASH.LOGGING.ENABLED**. Alternately, create another item in the MITS.CONFIG file named **DRILLDOWN.HISTORY.MAXIMUM**. On attribute one of this item, specify the number of drill-down actions that the system should maintain before rolling over and overwriting the oldest records.

To generate the Drill-Down Path Usage Report, run the following command from a database prompt:

```
MITS.DRILLDOWN.USAGE [ASCENDING|DESCENDING|UNSORTED] [SHOWALL|SHOWTREE]
[NODASHBOARD] [APPLICATION] [HELP]
```

Command Options for MITS.DRILLDOWN.USAGE

Option	Description
ASCENDING	Sorts the drill-down history in ascending order by usage.
DESCENDING	Sorts the drill-down history in descending order by usage.
UNSORTED	Displays the drill-down history unsorted.
SHOWALL	Displays all of the drill-down paths, regardless of whether or not they are being used.
SHOWTREE	Similar to SHOWALL, but sorts the view by drill-down path instead of usage.
NODASHBOARD	Specifies that the program should not consider drill-down usage that occurred as a result of using MITS Dashboard.
APPLICATION	Specifies which Hypercube Application to reference.
HELP	Displays information on the use of this command.

Resetting Usage Data

To reset the usage data, clear the MITS.DRILLDOWN.HISTORY file and set the value in attribute one of the LAST.ID item within the DICT of MITS.DRILLDOWN.HISTORY to zero (0).

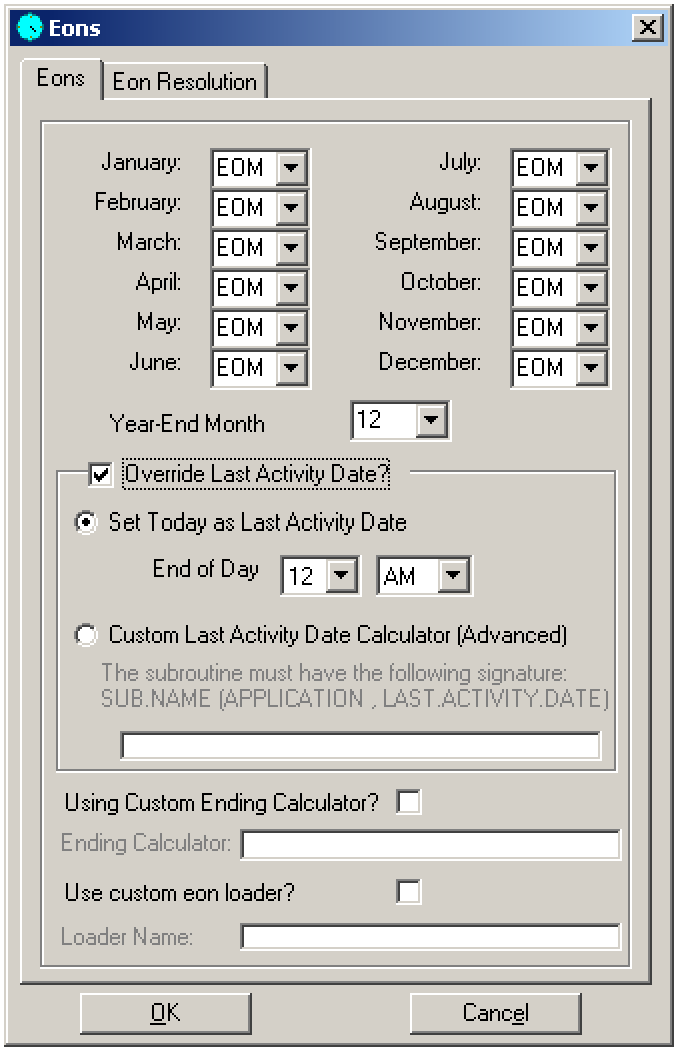
mitsdiscover

MitsMaker User Guide

Configuring Hypercube Eons

What is an Eon?

Central to MITS Discover is the concept of an "eon," which is a robust mechanism for describing time periods (current month, quarter before last, last 10 weeks, and so forth). MITS Discover regenerates a Hypercube's eons automatically as a part of each build wrap-up process, and stores them in the Hypercube's MITS.EONS file. Clicking the **eons** button displays a window similar to the following example, which allows you to provide MITS Discover with the criteria necessary for eon generation.



January:	EOM	July:	EOM
February:	EOM	August:	EOM
March:	EOM	September:	EOM
April:	EOM	October:	EOM
May:	EOM	November:	EOM
June:	EOM	December:	EOM

Year-End Month 12

☑ Override Last Activity Date?

● Set Today as Last Activity Date

End of Day 12 AM

○ Custom Last Activity Date Calculator (Advanced)

The subroutine must have the following signature:
SUB.NAME (APPLICATION , LAST.ACTIVITY.DATE)

Using Custom Ending Calculator? □

Ending Calculator:

Use custom eon loader? □

Loader Name:

Each month's fiscal end is specified, using either a constant day or "EOM" for "end of month."

Year-End Month

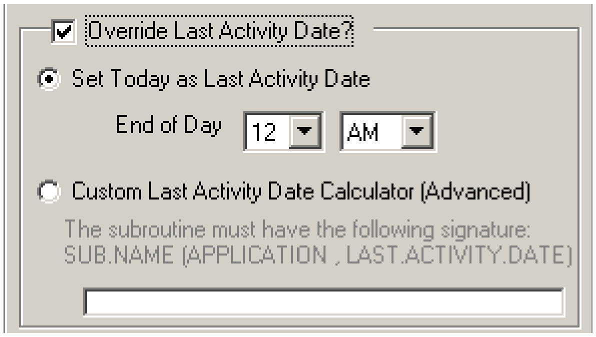
The year-end month corresponds to the last month in your organization's fiscal year.

Last Activity Date Override

By default, MITS Discover eons are created based on a *Last Activity Date*, which is equal to the date of the transaction that is furthest forward in time. Sometimes it is desirable to have a Hypercube's eons generated based on a different date. This can be especially useful as one month is ending and another is beginning. For example, if a Hypercube build is run on the last day of April and no transactions have yet been added to the Hypercube for May, all of the Hypercube's eons will be generated based on a Last Activity Date of April 30th. (A month-to-date eon will still refer to April rather than May, etc.) This feature can be used to force the Hypercube to generate these eons based on May 1st instead, even though no transactions have yet been added to the Hypercube for May.

The Last Activity Date Override feature is implemented by either selecting an end of day time or creating and specifying a custom Last Activity Date Calculator.

The default Last Activity Date Calculator subroutine compares the current time to the END.OF.DAY parameter. If the current time is prior to the time specified in the END.OF.DAY parameter, the Last Activity Date is not changed. If the current time is after the time specified in the END.OF.DAY parameter, the Last Activity Date is modified by adding one day to the Last Activity Date that was passed into the subroutine.



☑ Override Last Activity Date?

● Set Today as Last Activity Date

End of Day 12 AM

○ Custom Last Activity Date Calculator (Advanced)

The subroutine must have the following signature:
SUB.NAME (APPLICATION , LAST.ACTIVITY.DATE)

Use Custom Eon Loader

An application's **Eon Loader** program works in partnership with the application's **Ending Calculator** subroutine to generate the various eons for a Hypercube. These are stored in the MITS.EONS_<appName> file. If you leave this box unchecked, the default Eon Loader will be used. Checking this box and entering the name of a custom Eon Loader program will force MitsServer to use an alternate Eon Loader.

NOTE: The source code for the default Eon Loader program and some examples of custom Eon Loader programs are located in the MITS.BP_SAMPLE folder in the MITS Discover operational database account. It is recommended that you copy the source code from MITS.BP_SAMPLE to MITS.BP_SITE before attempting to edit any of these programs.

Ending Calculator

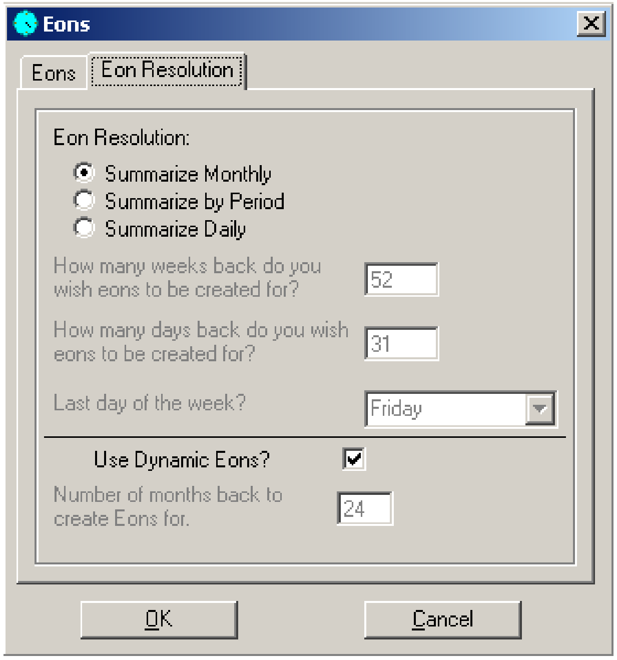
An application's **Ending Calculator** subroutine is called from a number of places throughout the MITS Discover system. The calling program sends a *span name* and a *date* to the Ending Calculator subroutine, and the Ending Calculator subroutine returns the *ending date for the span of time that includes the given date.*

If you leave this box unchecked, the default Ending Calculator will be used. Checking this box and entering the name of a custom Ending Calculator subroutine will force MitsServer to use an alternate Ending Calculator.

Eon Resolution

The specified *Eon Resolution* of a Hypercube tells MitsServer what the smallest eon in the Hypercube will be. This will commonly be set for MONTH (calendar) or PERIOD (fiscal), but will also often be set for DAY. The following example illustrates the **Eon Resolution** dialog box.

NOTE: If your reporting needs require reports based on a daily or weekly span of time, daily resolution will be necessary, but if your reporting needs do not involve time frames less than a month (or period), selecting monthly or period resolution will save significant disk space and build time.



Eon Resolution:

● Summarize Monthly
○ Summarize by Period
○ Summarize Daily

How many weeks back do you wish eons to be created for? 52

How many days back do you wish eons to be created for? 31

Last day of the week? Friday

Use Dynamic Eons? ☑

Number of months back to create Eons for. 24

Days and Weeks to Create Eons

If you choose daily resolution, you should indicate how many daily and weekly eons MitsServer should maintain over time. While specifying a large number is acceptable, it will often result in an unnecessarily large number of available columns, causing an increase in the effort required for the user to locate the column they are looking for.

Last Day of the Week

This is needed for daily resolution only if your last day of the week is something other than Friday.

Dynamic Eons

If Dynamic Eons are specified, monthly columns will be generated for each month that has been transformed into MITS. If you would like MITS Discover to maintain a smaller or larger number of monthly eons, clear this check box and specify a **Number of months back to create Eons for**.

mitsdiscover

MitsMaker User Guide

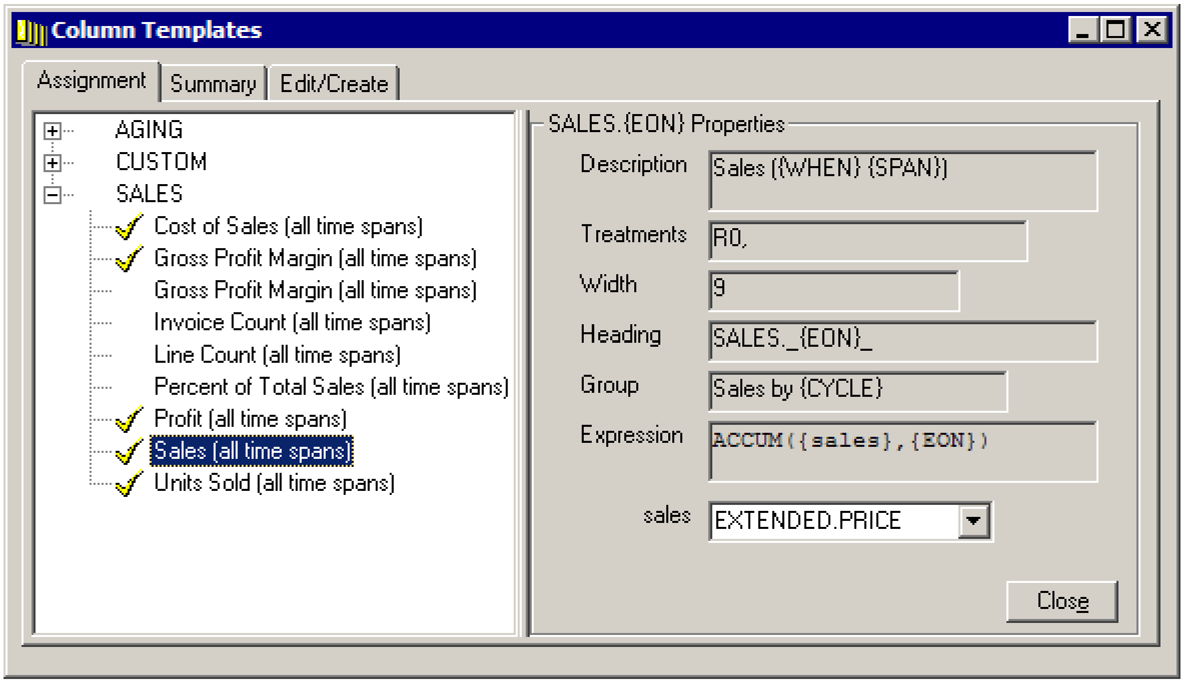
Column Templates

Column Templates

Click the **Columns** button to access the form used for associating accumulators with column templates. MITS Discover uses column templates to automatically generate the wide variety of columns available for presentation. At the conclusion of each transformation process (during the wrap-up phase) MITS Discover uses the date range of the transformed transactions to generate the Hypercube's eons.[1] These eons are then applied to the designated column templates to generate the columns.

MitsServer includes a few column templates as examples. Each template is assigned to an application group. These groups are displayed in an outline in the left pane of the Column Templates window.

Much of the data specified in column template definitions is represented by MITS Discover system variables like {WHEN}, {EON}, and {CYCLE}. The following example illustrates the "Sales (all time spans)" column template. (Note that the "{sales}" user variable in the template's expression has been assigned to the EXTENDED.PRICE accumulator.)



- ⊞ AGING
- ⊞ CUSTOM
- ⊟ SALES
  - ✓ Cost of Sales (all time spans)
  - ✓ Gross Profit Margin (all time spans)
  - Gross Profit Margin (all time spans)
  - Invoice Count (all time spans)
  - Line Count (all time spans)
  - Percent of Total Sales (all time spans)
  - ✓ Profit (all time spans)
  - ✓ Sales (all time spans)
  - ✓ Units Sold (all time spans)

SALES.{EON} Properties

Description	Sales ({WHEN} {SPAN})
Treatments	R0,
Width	9
Heading	SALES._{EON}_
Group	Sales by {CYCLE}
Expression	ACCUM({sales},{EON})
sales	EXTENDED.PRICE

If a column template only requires other columns to generate itself, such as profit which will be computed from sales and cost, MitsMaker displays a check box that is labeled *Calculate this template from other columns?* Selecting this check box will assign this column template.

[1] For more information about eons, see "What is an Eon?" on page 42.

Summary

The **Summary** tab displays a list of which accumulators have been assigned to which templates, as shown in the following example:

Editing Column Templates

Every MITS Discover system includes a few column templates, organized by general Hypercube class.

Column Template Notation

At build time, each column template will be used to combine accumulated Hypercube data with system eons to create multiple columns. A typical Hypercube might employ six or seven templates, potentially generating a few hundred available columns.

Internally, a column template definition has the same structure as a column definition with the exception of an additional TEMPLATE.GROUP field. The major difference between a column definition and a column template definition is that Column template definitions use a series of system variables, called *placeholders*, that are resolved at the time the columns are generated.

Placeholders

Placeholders in column expressions are surrounded by braces (for example: {**EON**}). They are either *internal* or *external*. The internal placeholders–{EON}, {CYCLE}, {WHEN}, {SPAN} and {N}–are provided automatically by MITS.LOAD.COLUMNS. You can also use external placeholder variables which will be assigned to accumulators on the Assignment tab.

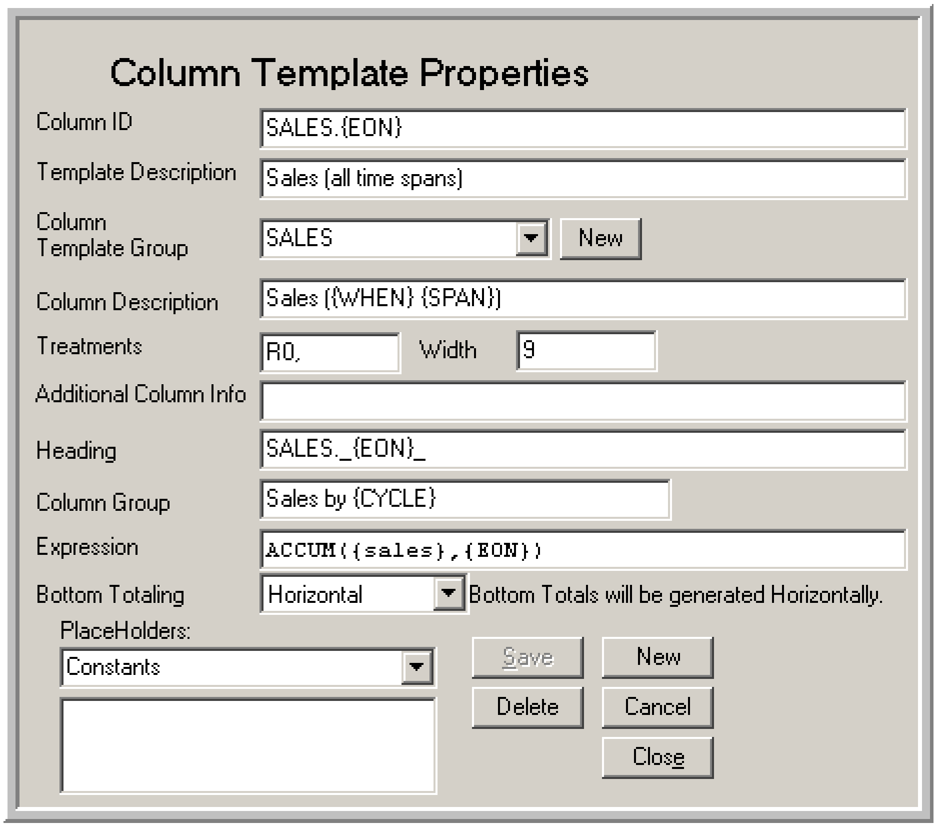
MitsMaker replaces the {EON} placeholder with each eon designator found in the application's EONS file in turn. The {CYCLE} and {WHEN} holders are descriptive variations of the eon designators. For example, an EON of “M-7” has a WHEN of “back 7 months” and a CYCLE of “month.”

The {SPAN} placeholder can be used in a column description to indicate what period of time the column refers to. For example, when placed in a column using the “Y” or Year-To-Date eon, a column description of `Sales ({WHEN}, {SPAN})` will be converted to *Sales (year-to-date, YEAR 2002 thru July 1)*. The {N} variable is replaced with the numeric portion of an Eon ID. For example, a column that uses an eon of “M-7” (representing “back seven months”) has an “{N}” of 7.

NOTE: The only placeholders that are allowed within column template IDs are {EON} and {N}.

Creating New Column Templates

You can create and/or modify a template using the **Edit/Create** tab from the **Column Templates** window. The **Column Template Properties** dialog box is shown in the following example:



Column Template Properties

Column ID: SALES.{EON}

Template Description: Sales (all time spans)

Column Template Group: SALES

Column Description: Sales ({WHEN} {SPAN})

Treatments: R0, Width: 9

Additional Column Info:

Heading: SALES._{EON}_

Column Group: Sales by {CYCLE}

Expression: `ACCUM({sales},{EON})`

Bottom Totaling: Horizontal Bottom Totals will be generated Horizontally.

PlaceHolders: Constants

WARNING: It is not safe to delete a column template that has assignments. Before deleting a column template, you must un-assign any assignments, then run a create-app to clear the assignments from the column template.

Template-Specific Fields

- **Column ID:** The Column ID establishes the column name, and is used as a suffix in the generated column subroutine. As such, keep this name meaningful but simple: capitals, digits, dots, and placeholders only. The only valid placeholders in a column ID are {EON} and {N}.
- **Template Description:** The template description appears when selecting templates.
- **Column Template Group:** The column template group is used to organize column templates into various application classes (sales, inventory, and so forth).

Column-Specific Fields

- **Column Description:** The column description should provide an easy-to-understand description of the column and is displayed when columns are being selected for inclusion in a flash screen in MITS Discover.
- **Treatments:** Treatments are similar to Pick's numeric conversion codes. Some experimentation may be needed. MITS Discover generally stores accumulators with their decimals intact. As such, when specifying a template to produce decimals, indicate a "scaling" of zero, such as 'R02' (rather than 'R2').
- **Width:** Width is only critical in terminal-based MITS, since MITS Discover automatically corrects column widths. The width should be a best estimate of the maximum number of characters, including punctuation, required by the grand total of the largest generated column.

NOTE: In terminal-based MITS, if a column filled with asterisks is displayed, use the RESIZE command to automatically adjust all column widths.

- **Additional Column Info:** Verbose descriptions can be attached to a column template that will provide end-users with more information about how the values displayed in the columns created from the column template were derived.
- **Heading:** MitsMaker removes dots in the column heading and the heading is folded at that point. MitsMaker replaces _{EON}_ with each eon's "column blip" (a terse description of the time span represented by the eon).
- **Column Group:** Used to group like columns in the column selection display of MITS Discover. For example, a column group of "Sales by {CYCLE}" will create individual eon-specific column groups to hold "Sales by Year" columns, "Sales by Quarter" columns, and so on.
- **Expression:** Determines what will be displayed in the column cells. A complete discussion of Column Expressions can be found in a MITS Technical Document named *MITS Discover Expressions* which can be downloaded from http://www.mits.com/techfocus.htm.
- **Bottom Totaling**: If the columns generated by this column template will display the result of an arithmetic function performed against one or more other columns, specify whether you want the bottom totals for the column to be generated horizontally or vertically.

  Vertical bottom totaling adds the values in the column and displays the result in the **Grand total** cell.

  Horizontal bottom totaling performs the same arithmetic function on the bottom totals from the base component columns.

  For example, if you have created a column template that displays the result of dividing the values in one column by the values in another column, you would not want the bottom totals to be the sum of the individual results for each row. Instead, you would need to add up the results from the numerator column and add up the results from the denominator column then perform the division on these two totals to achieve the correct grand total.

mitsdiscover

MitsMaker User Guide

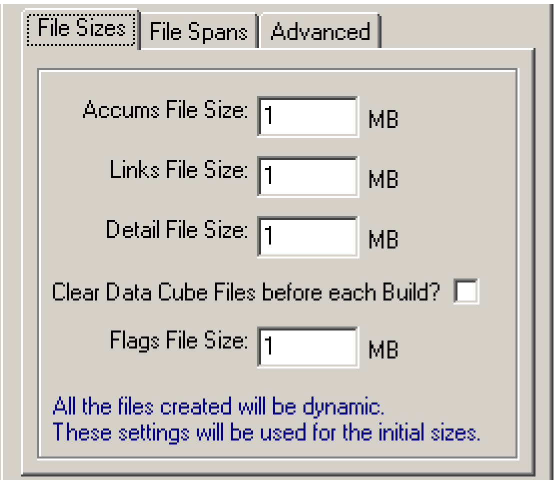
Specifying File Sizes

The “File Sizes” Screen

Every MITS Discover Hypercube is actually a collection of special files. MitsMaker requires you to enter the size you wish these files to be when it creates them. In the case that your environment supports dynamic (auto-sizing) files, the size you enter will be the base size used to create the files. MitsMaker will inform you if the files will be created dynamically.

The following example illustrates the **Files** dialog box:

File Sizes



Accums File Size: 1 MB

Links File Size: 1 MB

Detail File Size: 1 MB

Clear Data Cube Files before each Build? □

Flags File Size: 1 MB

All the files created will be dynamic.
These settings will be used for the initial sizes.

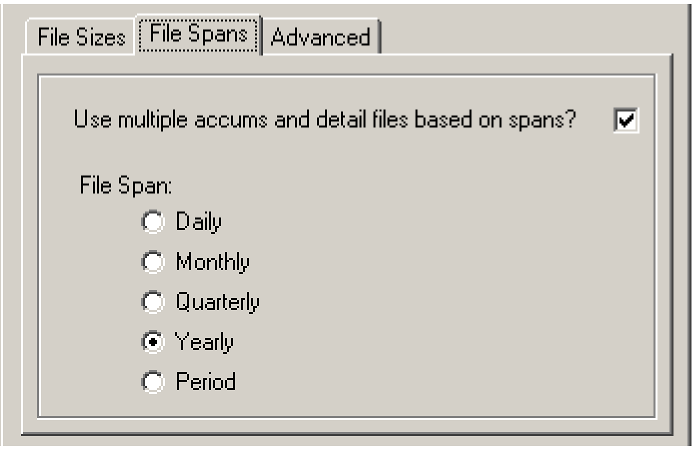
MitsMaker stores the data from the accumulators in the ACCUMS file.

MitsMaker stores the combinations made by identifiers in the LINKS file.

The DETAIL file contains the list of transactions for each combination and time frame.

The items in the FLAGS file represent all of the transactions that have been built into the Hypercube.

File Spans



Use multiple accums and detail files based on spans? ☑

File Span:

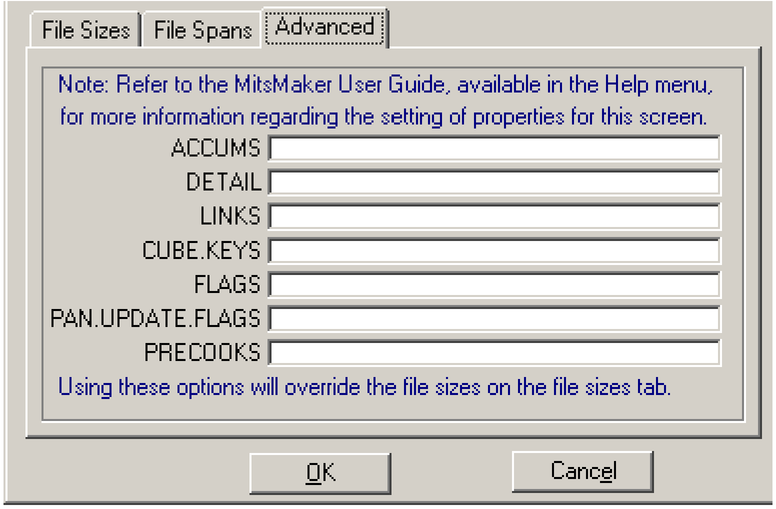
○ Daily
○ Monthly
○ Quarterly
● Yearly
○ Period

The ACCUMS and DETAIL files can become very large, and continue to grow in a linear fashion as transactions are added to the Hypercube. To simplify file management and performance, you may decide to break these files up by day, month, quarter, or year spans. If you select file span, the corresponding file size applies to each span created.

NOTE: If you are using a custom eon loader to generate periods (for example, 13 periods in one year rather than 12 months), you may want to use a *period-based* resolution for your file spans.

Advanced File Settings

WARNING: This feature is intended for advanced users that have an intimate familiarity with the various options available when running the CREATE-FILE command on their database. Entering incorrect values on this tab could seriously impact the build speed and flash screen response times in your Hypercube!



Note: Refer to the MitsMaker User Guide, available in the Help menu, for more information regarding the setting of properties for this screen.

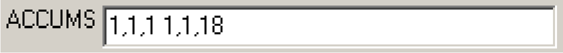
Using these options will override the file sizes on the file sizes tab.

This tab allows you to specify the parameters that will be used to create each of the MITS Discover Hypercube files represented, overriding the defaults. The only information you need to place in the box is what would come after the filename in a CREATE-FILE command.

For example, imagine that the default command to create a MITS.ACCUMS_SALESDEMO file on UniVerse is:

```
CREATE-FILE MITS.ACCUMS_SALESDEMO 1,1,1 1,1,30
```

If you wanted to override the “type” specifier for the DATA portion of the file so that it will be created as a Type 18 file, it would look like this:

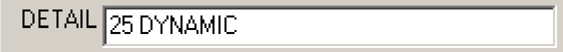


ACCUMS 1,1,1 1,1,18

Here’s another example. Let’s say that the command to create a MITS.DETAIL_SALESDEMO file on UniData is:

```
CREATE-FILE MITS.DETAIL_SALESDEMO 50 DYNAMIC
```

If you wanted to override the “size” specifier and have future files created with a modulo of 25 instead, it would look like this:



DETAIL 25 DYNAMIC

For more information about the various file creation options available to you, refer to your database documentation.

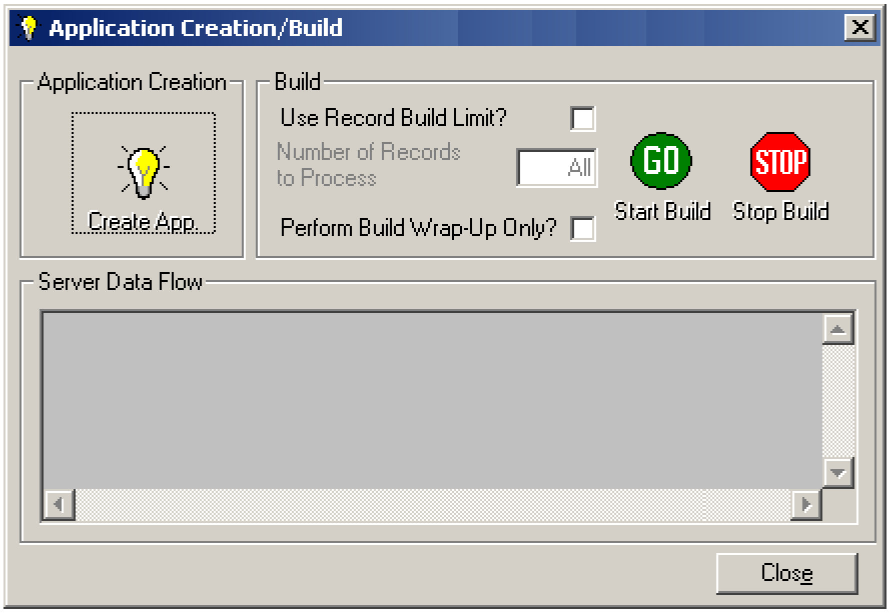
mitsdiscover

MitsMaker User Guide

Application Creation and Build

Application Creation

At this point, you have established all the configuration criteria for your Hypercube. You are now ready to create the MITS Discover application. This is done with the **Application Creation/ Build** window, as shown in this example:



Use Record Build Limit? □

Number of Records to Process: All

Perform Build Wrap-Up Only? □

Major vs. Minor Remakes

The **Server Data Flow** window displays the interaction between MitsMaker and MitsServer. This information is helpful when diagnosing problems. When you click the light bulb icon, MitsMaker submits the configuration to MitsServer, and the server executes an extensive process to create the application. If errors are presented during the application creation, close the Application Creation/Build window and correct whatever caused the error before attempting to create the application again.

Clicking **Create App** causes one of three types of application creation to occur:

- **Major Make:** A “major make” occurs the first time an application is created.
- **Minor Remake:** A “minor remake” signifies that the application has already been created, but a minor, or non-structural, change has been made. This minor remake does not clear the Hypercube data therefore no rebuild of the Hypercube is necessary.
- **Major Remake:** A “major remake” signifies that there have been changes made to the Hypercube configuration that have altered the structure of the cube in some way. In this instance, a warning will be displayed and if accepted, the application creation process will clear any existing information from the cube requiring a complete rebuild of the Hypercube.

Modification	Remake Type			Build		
	Major	Minor	No Create-App Required	Full Rebuild	Build Wrap-up Only	None
Data Pool Changes			X	R*		X
Add/Remove Ident or Accum	X			X		
Identifier Description Logic		X			R*	
Detail Display Assignment/Order		X				X
Drill Down Path Changes		X				S*
Eon Changes	X			X		
Column Template Definitions			X		X	
Column Template Assignment		X			X	
File Sizes		R*		X		

R* = Recommended. You will not be prompted, but this action is recommended
S* = Special. The next time a build is run, the changes will be applied to the Hypercube

Building the Hypercube

Once the MITS Discover application has been successfully made, you can run a *Hypercube build process*. The Hypercube build process includes two phases: the *transformation phase* and the *wrap-up phase*.

The Transformation Phase

The transformation phase of a build process consists of the following steps:

1. Selecting transactions from the Hypercube's source file

2. Transforming transactions into the Hypercube

The Wrap-up Phase

The wrap-up phase of a build process consists of the following steps:

1. Loading lookup b-trees for identifier abbreviations

2. Generating eons

3. Generating column definitions

4. Sorting drill-down path lists (also known as "links")

WARNING: The build process can be highly resource-intensive. Therefore, you may want to limit the number of transactions involved or limit the execution of the build process to off-hours using the MITS Discover build scheduler. This is covered in more detail in the *MITS Discover User Guide*.

Stopping a Build

You may interrupt a build process that is running within MitsMaker by clicking **Stop Build**. This will force the transformation phase to terminate and drop the build process into the wrap-up phase.

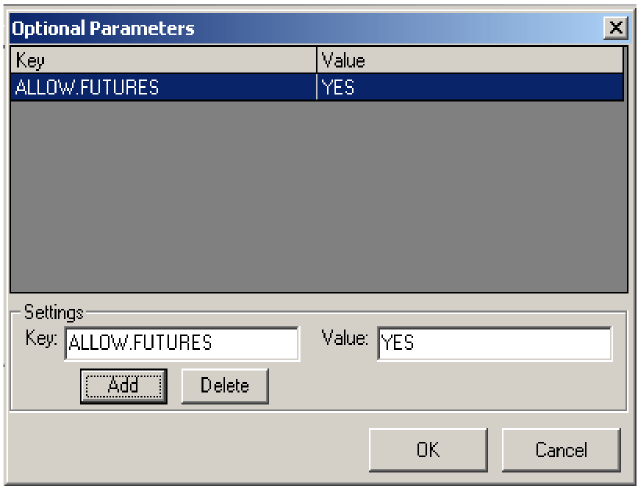
mitsdiscover

MitsMaker User Guide

Optional Parameters and Attachments

Optional Parameters

Optional Parameters is a way to support extra features that may be supported on the server but do not have an equivalent in MitsMaker. Access the Optional Parameters list by selecting **Parameters List** from the **Tools** menu of MitsMaker. Entries placed in this list will be added to the configuration file for the application you are creating.



Key	Value
ALLOW.FUTURES	YES

Key: ALLOW.FUTURES

Value: YES

Add a Parameter

To add a parameter, enter the parameter key and the desired value in the boxes provided, then click the **Add** button. To remove a key, select the key by clicking it in the grid, and click the button labeled **Delete**.

Optional Parameters

Parameter	Description
Options	
ALLOW.FUTURES	Allows for the selection and processing of future transaction dates. If this parameter is not listed, the default is NO. NOTE: This parameter will have no effect if an Index Date Routine is in use.
Yes or No	
CLEANUP.LINKS	This is used in conjunction with the optional MITS.CLEANUP.LINKS feature. If this feature has been configured, a value of YES will activate a MITS.CLEANUP.LINKS phase during the build. If this parameter is set for NO, this phase will be skipped.
Yes or No	
COLUMN.LOADER	The name of a database program that will be used in place of the default MITS.LOAD.COLUMNS column loader program during the build wrap-up.
Program Name	
EON.ACTIVITY	By default, eon IDs and descriptions are generated in reference to the first and last activity dates. You can use this parameter to force MitsServer to generate eon IDs and descriptions that refer to a specific span of time. An example of a *relative* eon is "back one year." An example of an *absolute* eon is "Year 2004." NOTE: If absolute eons are specified, the default Eon Loader of MITS.LOAD.RELATIVE.EONS will be replaced with the alternate MITS.LOAD.ABSOLUTE.EONS.
RELATIVE or ABSOLUTE	

Optional Parameters (continued)

Parameter	Description
Options	
EXPLORE.MODE	Setting this parameter to STRICT will limit a user's drill-down options to *exactly* what is specified in the drill-down paths defined in MitsMaker at the time of the application creation. Leaving this parameter at the default of OPEN will allow users to utilize what are called "bonus" drill-down paths. These drill-down paths may not have been specified at the exact location where they become available, but were specified elsewhere in the Hypercube. This means they can be made available to the user with no further build speed or disk space overhead.
OPEN or STRICT	
INTER.TRANS.SLEEP	The number of seconds to pause between processing source transactions during the build. This can be used to release CPU time and disk access in an attempt to minimize the impact a build process has on system response.
# of seconds	
HIDDEN	Hypercubes that include this optional parameter will not appear in the list of available applications on the "Create New Flash Screen" page. This is intended for use on Hypercubes that are only viewable using the "external accums" feature and therefore should not be flashed directly.
1	
POST.CLIP.EONS.MESSAGE	This message appears after MITS.CLIP.EONS is run.
Text Message to Display	
POST.DELETE.MESSAGE	This message appears after MITS.DELETE.APP is run.
Text Message to Display	
POST.MAKE.MESSAGE	This message appears after an Application Creation (MITS.MAKE.APP) is run.
Text Message to Display	
POST.PURGE.MESSAGE	This message appears after MITS.PURGE.APP is run.
Text Message to Display	
POSTPROCESS	The name of a database command or program that will be executed at the end of each build process, after the wrap-up phase is complete.
Program name	
PRE.CLIP.EONS.MESSAGE	This message appears when MITS.CLIP.EONS is run.
Text Message to Display	
PRE.DELETE.MESSAGE	This message appears when MITS.DELETE.APP is run.
Text Message to Display	

Optional Parameters (continued)

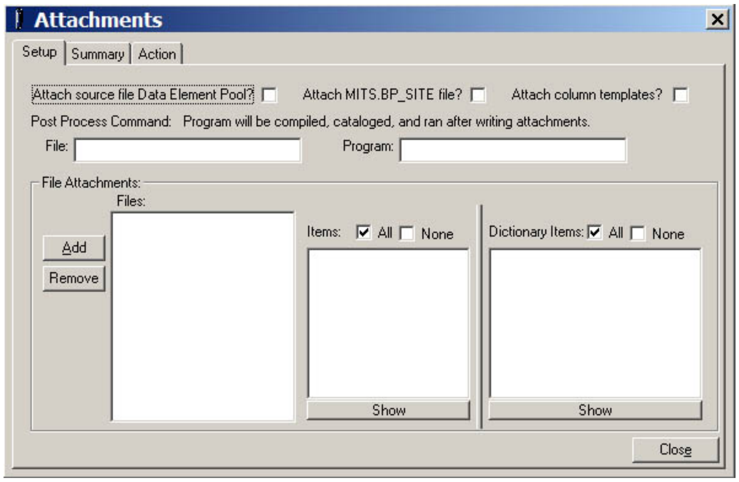
Parameter	Description
Options	
PRE.MAJOR.REMAKE.MESSAGE	This message appears during an Application Creation (MITS.MAKE.APP) if the changes would require a major remake (meaning all existing historical data will be purged from the Hypercube).
Text Message to Display	
PRE.PURGE.MESSAGE	This message appears when MITS.PURGE.APP is run.
Text Message to Display	
TRANSFORMATION.ORDER	MitsServer normally processes source transactions in reverse date order. To force MitsServer to process source transactions in forward date order, add this optional parameter and set its value to FORWARD.
FORWARD or REVERSE	

NOTE: Once a parameter has been added, it cannot be modified. If you need to modify a parameter for any reason, you will need to DELETE the parameter and then re-add it.

Attachments

Attachments are a way to take files, items, and programs from the MITS Discover operational account of a database server and include them in the MitsMaker specification (.mkr) file for transport to other systems. Any item or dictionary item out of a file in the MITS Discover operational database account can be attached to a ".mkr" item.

The following example illustrates the **Attachments** dialog box:



Attachments

Setup | Summary | Action

Attach source file Data Element Pool? □ Attach MITS.BP_SITE file? □ Attach column templates? □

Post Process Command: Program will be compiled, cataloged, and ran after writing attachments.

File: Program:

File Attachments:

Files:

Items: ☑ All □ None

Dictionary Items: ☑ All □ None

Selecting Data to Attach

Attach Column Templates

Select this check box to attach the application's custom column templates.

Attach MITS.BP_SITE

Select this check box to attach any custom programs you have placed in the MITS.BP_SITE file.

Attach Source File Data Element Pool

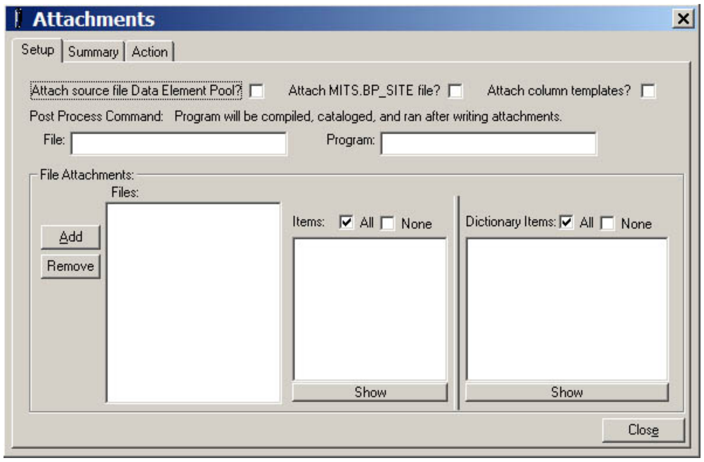
Select this check box to attach the entire data pool for the source file. Operation logic is included when attaching the data pool.

Post Process Command

Specify the location and name of a program that will be compiled and executed after the WRITE process is completed on the destination server.

Add/Remove

If you click **Add**, the MultiValue Explorer appears, where you can select the file you wish to add to the list of available files. Click **Remove** to remove the selected file from the list of available files. Click **Show** to display a list of the available items or dictionary items. You can then select the specific items you want to attach by selecting the check box next to the item name. Alternately, you can select all the items by selecting the **All** check box. Select **None** to disable all the selections you have already made.



Attachments

Setup | Summary | Action

Attach source file Data Element Pool? □ Attach MITS.BP_SITE file? □ Attach column templates? □

Post Process Command: Program will be compiled, cataloged, and ran after writing attachments.

File: Program:

File Attachments:

Files:

Items: ☑ All □ None

Dictionary Items: ☑ All □ None

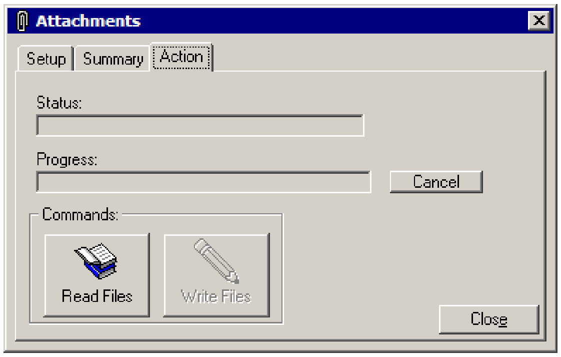
NOTE: After adding the file, you must select the newly added file from the file list *before* you select which items you wish to attach.

Summary

Click the **Summary** tab to view a list of all the items you have selected.

Reading/Writing Data

After you review the summary, click the **Action** tab to retrieve the data from the server. The following example illustrates the **Action** window. Click the **Read Files** button to retrieve the data from the server. The status line displays what MitsMaker is doing, and the progress bar shows you how much is left to finish. MitsMaker stores the data read from the server in the .mkr file. To write the data back to the server, connect to the server, open the .mkr file, and click **Write Files**.



Attachments

Setup | Summary | Action

Status:

Progress:

Commands:

mitsdiscover

MitsMaker User Guide

Appendix A: Operations and Column Treatments

MITS Discover Operations

MITS Discover draws data from the operational system using native dictionaries. In the cases where logic is required over and above the processing available from the dictionary items, you can use a MITS Discover operation. An operation is a BASIC subroutine attached to a source file dictionary item that is called each time data is retrieved. Click **Operation** to access the **Data Pool** form. The first data pool form tab starts off with a field's subroutine template, as shown in the following example:

```
SUBROUTINE (MAT OPERATION.IO, MAT OPERATION.STATIC)
***
*   OPERATION.STATIC is already initialized by the optional
*   INITIAL.OPERATIONS routine.
*
*   OPERATION.IO contains the following (equated) values:
*   OPERATION.VALUE    contains the current field's value (input/output)
*   OPERATION.ID       id of the current item (input)
*   OPERATION.ITEM     all attributes of the current item (input)
*   OPERATION.MV.SLOT  multi-value slot of the current value (input)
*  OPERATION.ACTION   return "SKIP" to skip a lie or "STOP" to stop (output)
*   OPERATION.ERR      optional error message (output)
***
$INCLUDE MITS.BP_CORE MITS.INCLUDE.OPERATION.DIMS
** place operation logic here
RETURN
```

Data is communicated with MITS Discover through the OPERATION.IO vector, using the equated variable names described in the template. In the event that one or more operations require opened files or other initialization, a Global Initialization Operation should be used. This routine can be accessed on the second tab of any of the **Operation** forms, as shown in the following example:

```
SUBROUTINE (MAT OPERATION.STATIC, ERR)
$INCLUDE MITS.BP_CORE MITS.INCLUDE.OPERATION.DIMS
MAT OPERATION.STATIC = ''
ERR = ''
** place operation initialization logic here
RETURN
```

There is only one Global Initialization Operation despite the possible presence of multiple regular operations. MITS Discover processes the Global Initialization Operation prior to processing any of the application's individual operation routines. Anything assigned to the 100-element OPERATION.STATIC variable array (such as an opened file) can be referenced in any of the application's other operations. If anything is assigned to the variable named **ERR**, processing will terminate after displaying the contents of ERR.

Column Treatments

Treatments are made up of a concatenated string of the following components:

- Justification (“L” or “R”)
- Implied decimals or *masking*
- Scaling
- Punctuation
- Trailing character

Justification is the only required component, and is the only component for nonnumeric data.

Accumulator data is usually stored in a normalized form (including the decimal). This is in contrast to storing data in a scaled form (for example, 1000 for $10.00).

The implied decimals component of treatments (also known as *masking*) is used to insert a decimal at the position you specify.

MitsMaker uses scaling to shift a resulting value to the right, dropping the least significant digits.

MitsMaker includes a comma as punctuation to separate thousands, millions, and so forth.

You can add any character to the end of the treatment as a *trailing character*. That character will be displayed at the end of the result. Here are some situations where a trailing character could be useful:

- % (for percentages)
- K (for “kilo” or “thousands” when a value has been scaled to thousands)
- M (for “mega” or “millions” when a value has been scaled to millions)

MitsMaker User Guide

Appendix B: Hypercube Design Considerations

MITS Discover Hypercube Design

This appendix is presented to help MITS Discover Hypercube designers weigh the competing factors in the design and implementation of a MITS Discover Business Intelligence solution and subsequently create a design that maximizes the benefits of the system, while making the best use of available resources. It is assumed the user is familiar with the MitsMaker design tool and the terminology detailed in the glossary of this document.

The Goal

A MITS Discover Business Intelligence system has the potential of creating a very satisfying end-user experience. The speed and efficiency with which a user can navigate the data brings a whole new level of both productivity and usability. Those companies that have effectively implemented a MITS Discover solution often make comments such as:

- “We would be lost without it.”
- “MITS (Discover) has become a large part of the daily management of our business.”
- “We have become addicted to MITS (Discover).”
- “There is a night-and-day difference between running a report and running MITS (Discover).”
- “With MITS (Discover), it’s all right there.”

The design of a Hypercube, or MITS Discover application, can have a significant effect on the performance of the Hypercube both in terms of the end-user’s experience and the time and resources necessary to transform the source transactions and store the resulting data files. Our goal here is to discuss the ways that a designer can optimize the MITS Discover system to provide this type of end-user experience.

Assessing the Available Resources

Every information system is a collection of resources such as CPU power, disk space and access bandwidth, memory and so on. All these resources are placed within a fixed framework of available time; there are only so many processing hours in a day. The MITS Discover application designer must be careful to consider the available computing resources early in the design phase.

The MITS Discover Business Intelligence system, and MitsMaker in particular, are powerful, flexible tools that may be used to define very useful and robust data reporting solutions. One pitfall to be aware of is that these tools can also be used to create data retrieval solutions that require more than the available resources. Attempting to design an overly ambitious solution will result in a frustrating experience both for the designer and the users who are looking forward to the benefits of the MITS Discover system.

The available computer resources typically consist of:

- Processing Power
- Disk space
- Disk access I/O
- System Ram Available
- System availability and load

We will present more detail about the way MITS Discover uses these resources later.

Certainly, resources in each of these categories can be added if necessary, and often, additions are justified based on the value of the MITS Discover solution.

Benefits

In each of the sections below, we will discuss the balance between these resources and the desired benefits of:

- End-user response time
- Features available to the user
- Appropriate “scope” to the analysis
- Data presented in an understandable, intuitive context
- Source file transformation (Hypercube build) speed
- Historical records
- On-going records (additions)
- Changed records

Design Considerations

The following section details some of the design considerations. Each element of the design is discussed in turn with observations and recommendations.

Application name should be concise

The application name is limited to letters, numbers, and the "." character. The application name will be used throughout the application to create file names and other data structures. Because of this, long or complex application names should be avoided.

Extracted identifiers should be concise

The data that is extracted and used to create drill-down identifiers should be as concise as possible. Extract the customer ID, not the customer name. The name can always be read from the file and displayed at run-time. The data that is extracted and used as identifiers will be combined to make compound keys in several files. These data elements should typically not contain spaces. Although spaces in record keys will work on all supported platforms, it is not recommended.

If all the keys in the identifier description file include a similar prefix or suffix, it is best to use the "Description Link Prefix" and "Description Link Suffix" feature of MitsMaker to add these to the key prior to reading the record. This is more efficient than repeatedly storing the prefix or suffix in the Hypercube.

If a concise key is available, but the user would benefit by seeing a more verbose or descriptive key during data exploration, it is possible to define a "Trailer Routine" in MitsMaker to convert from one to the other. In this way, the actual key stored in the Hypercube is as concise as possible but information is still presented to the end user in a familiar context.

Avoid storing plain text identifier values in the Hypercube

Storing file key information in the MITS Discover Hypercube and then using that information to link to a file containing the description at run-time has several advantages:

- The key is more concise and will take less room in the Hypercube. Since it becomes part of the key in many files, the files will hash more efficiently and take up less space.
- By "linking" the identifier to the source file in this way, the end user can insert other fields from the source file and use them for analysis without any additional design work. This is an enormous benefit to both the end user and the Hypercube designer.
- Providing a description helps present the information in a usable, easy-to-interpret context. The user is always more productive when working in a familiar context requiring a minimum amount of interpretation of the data.

The number of identifiers should be limited

Each time an identifier is added to the application it must be combined in some way with all the other identifiers. The number of these combinations (see next topic below) has a direct impact on the transformation speed. If a Hypercube has identifiers that are never used in combination with others, then consider providing two different Hypercubes to provide similar analysis. These two smaller Hypercubes will often be transformed more quickly than one large one. A small Hypercube will have four to six identifiers; a large one may have up to nine or ten.

Limit identifier combinations and drill-down paths

The transformation process must update the MITS Discover data files for each drill-down path and combination of identifiers available in a MITS Discover Hypercube. Defining these paths carefully will assure that the system provides the necessary results without excessive transformation time.

MITS Discover includes a feature where the drill-down paths available to the end user can be designated explicitly. When using this feature, known as "designated paths," the designer should always start with a blank list and add drill-down paths as needed. This method will always result in a more concise list of drill-down paths and more closely meet the needs of the organization.

The list of identifiers selected for a MITS Discover application will typically represent several hierarchies. These relationships should be considered when creating the drill-down paths. For example, one identifier might be "Product Line," and another "Product." In such a case, it would be wasteful to provide a drill-down path from Product to Product Line, since theirs is a one-to-one relationship in that direction. Certainly, a drill-down path from Product Line to Product makes perfect sense.

With the use of the MitsTree stand-alone utility, defining these drill-down paths can be a collaborative exercise. Many people can participate, and the end result is a Hypercube that exactly meets the end user's requirements. This step alone, although it is possibly the most difficult, will yield the greatest improvement in all aspects of the MITS Discover application's performance.

Designs with 20 to 30 combinations are considered manageable. More than 100 combinations will tend to be burdensome to transform and maintain.

Limit cell combinations where Detail Display is available

The transformation process must update the MITS Discover data files for each "cell" or combination of identifiers where Detail Display has been made available in a MITS Discover Hypercube design. As a result, it is important to carefully consider where the availability of this feature is most beneficial and not be indiscriminate when adding this feature to the drill-down paths.

A single application should be of manageable "scope"

It is tempting to build a MITS Discover application that will serve the needs of an overly diverse group. This is usually a mistake. One symptom that the design is heading in this direction is when the data elements have ambiguous definitions. Usually this means that different groups are accustomed to looking at the data from different perspectives. A classic example would be a Sales department that considers the data element "order date" to be the date the order was placed, a Shipping department that concentrates on the shipping date, and an Accounts Receivable department that is interested in the invoice date. While there is some overlap in interest, this scenario is probably best addressed with three small Hypercubes rather than one large one that would be difficult to manage. As the end user explores data in MITS Discover, switching between Hypercubes is nearly transparent. Flash screens from different Hypercubes can appear next to each other and when the user chooses them, the MitsServer automatically switches applications without any other user involvement.

Consider the impact of more accumulators vs. more identifiers

While increasing the number of identifiers in a MITS Discover application has significant impact on the resources required for the application, the same is not true for accumulators. Adding an accumulator to the design requires only that the MITS Discover system update one additional value for each identifier combination. It does not increase the number of combinations, nor does it impact the resources necessary to provide Detail Display capabilities.

One thing to consider about adding accumulators is that they are typically used in conjunction with column templates to create column definitions. If a MITS Discover application includes a large number of column definitions, end users can experience some delay as they go to insert new columns into a flash screen. This can have a negative impact on the end user experience.

Avoid using "dimensioned paths" for a production application

The ability to quickly develop an application configuration by specifying dimensioned paths should be used for demonstration or prototype purposes only. This method of specifying the drill-down paths is very wasteful because many paths are defined that have no bearing on an organization's data analysis. Even though these paths are not useful, the transformation process must still update, store, and maintain them. This consumes processing time and disk space but provides no benefit to the user.

Avoid drill-down paths that yield a large number of rows

Drill-down paths that result in listings longer than a few hundred rows should be used only after careful consideration of the impact on the performance of the transformation process and the end-user experience. An illustration of this situation would be a mail-order house that has 100,000 customers. Providing the ability to drill down by customer for the entire company would yield a report with 100,000 rows. Not only is it difficult for the end user to wade through this list, it is difficult for the server to maintain and generate this output. A much better solution would be to add another level (or perhaps two) in this hierarchy. Starting a drill-down path with "customer state," then "customer type," then "customer" would present a reasonable list of customers against which to do analysis. Since MITS Discover is designed to do the aggregation of information, it is not difficult to get totals and sub-totals of this information. With the ability to do "explore all" or "explore all with selection" drill-down types, all the information is available but in a more manageable form.

Avoid Detail Display with many source transactions

For the same reason that reports with many rows should be avoided, enabling Detail Display on combinations derived from a large number of source transactions should also be avoided. Again, using the example of a Sales Analysis Hypercube for a mail-order house, it would not be advisable to enable the Detail Display feature that would show all source invoices for an entire division. Realize that the division could have hundreds of thousands of invoices in the course of a year. Giving the user the ability to see all these invoices in one display is not a practical use of the available resources. A better solution is to "push" the ability to view detail transactions to a lower level so that the resulting list is more manageable.

Avoid unnecessary transformations

MITS Discover is flexible about how source records are selected and processed. This flexibility can help assure that only new or changed records are included in the transformation step. This transformation step can be done in one of the following ways:

- Continuous Transformation - records are selected and transformed in near real-time.
- Periodic Transformation (nightly, weekly, monthly) - transactions are selected and transformed on a routine schedule. MITS Discover includes a scheduling component to accomplish this. More information about the MITS Discover build scheduler can be found in the *MITS Discover User Guide*.

In addition to the two transformation methods described above, it is also possible to create MITS Discover Hypercubes in one of the four following application modes:

- **Clear and Rebuild**. In this mode, the Hypercube is completely cleared of data each time it is to be rebuilt. The source records are selected and transformed, replacing any data previously transformed. This is the simplest but usually the least efficient of the application modes.
- **Incremental Addition.** This type of Hypercube is useful in situations where the records are added to the source file and are then static. A common example is an invoice file, where once an invoice is produced, it does not change. In this case, the transformation process would select new records from the source file and add them to the existing MITS Discover Hypercube. This is the default transformation behavior of a MITS Discover application, and MITS Discover includes provisions for automatically selecting any unprocessed records from a source file. This is of little consequence if the source file is relatively small and the transformation time is a few minutes or hours. This method becomes more useful with a source file that contains a significant number (possibly millions) of historical records. In this case the transformation could happen over a number of weeks, or perhaps even months. Since, by default, the transactions are added to the Hypercube in reverse-chronological order, the Hypercube becomes usable early in the transformation process and as more data is added, more analysis is possible.
- **Dynamic Change Update.** In this method it is assumed that the source records are subject to change, and that the MITS Discover Hypercube should reflect these changes. This is accomplished by reprocessing changed transactions to remove the initial values from the Hypercube and then re-processing back in the new values. This method requires some mechanism outside of MITS Discover to identify these changed transactions. In practice, this can be accomplished by the use of a database trigger or a simple modification to the production system to generate a flag when a record is changed or updated. This type of Hypercube design is often combined with the "continuous transformation" mentioned above to create a near-real-time view of data in the source system.
- **Multiple Snapshots.** This method is useful when the state of a database changes over time and there is a need to capture that state periodically for comparison or trend analysis. A typical example would be an analysis of inventory levels or accounts-receivable values. The values in the source file (inventory for example) change as activity is processed, so the data must be extracted and stored in a MITS Discover Hypercube in multiple "snapshots." These snapshots are then available to the end user as different columns of data that can be analyzed or compared. Since the source transactions are constantly changing, providing Detail Display analysis of this type of Hypercube requires another step. This step involves using the preprocess feature of MITS Discover to pull data from the source file and store it in an intermediate file. This file can then be used as the source for the MITS Discover Hypercube. With the creation of this intermediate file, Detail Display can be enabled in MITS Discover to provide a view of the state of the detailed source records at various points in time.

Select source transactions efficiently

If the source file is indexed on a date field, use the "Indexed Date" or the "Index Date Routine" features to quickly retrieve new transactions to be transformed. The less time spent selecting records, the more time that is available for actually doing the transformation.

Consider identifying changed records outside of MITS

By modifying the source application to write a simple flag record when a transaction has been added or changed, the MITS Discover transformation process can be enhanced to deal only with those records. This effectively eliminates the selection process.

Hardware Configuration Considerations

MITS Discover can be deployed a number of ways. The MitsServer software can be added to an existing operational computer system, or provided on a separate dedicated server, or even multiple servers.

Transformation Process vs. Interactive User Process

To provide the best end-user experience possible, most of the MITS Discover processing is done during the transformation process. This process is driven by the configuration created using the MitsMaker design tool. Several fundamental steps are involved in the transformation of source data into a MITS Discover Hypercube:

- Extract data from source file(s)
- Create and store drill-down results
- Aggregate and store numeric values
- Track and store source record IDs for Detail Display
- Store actual values extracted from the source records so future changes can be applied
- Track which records have been processed

During an interactive user session, the load on the host system is much lighter. Based on user requests, various stored data is retrieved and sent to the client browser via the MITS Discover gateway. At times during this process, the host computer may be asked to sort the results, or to read additional data from associated data files–such as a field from the Customer Master file that will be displayed in a column. The host system maintains a work file for each interactive process and this is where temporary results will be written and sorted.

A MITS Discover interactive user session typically requires fewer host resources than a traditional report that may have to scan a source file with a SORT or SELECT statement.

Disk Drives

The MITS Discover transformation process places the greatest demands on the host system and this demand is primarily for disk access. Because of this fact we have found that the most important factor in the host hardware design is the optimization of the disk sub-system.

If the MitsServer is added to an existing computer system, it can be located on dedicated disk drives (on most database platforms) and these drives can be optimized for the MITS Discover application. We have found that isolating the disk drives used for MITS Discover from the rest of the operational system can allow for running MITS Discover transformations even during the production day.

Since the MITS Discover application data is derived from the source data files, it is not as important that the entire MITS Discover system be backed up quite as often as the operational data. The prospect of re-building one week of data into the MITS Discover Hypercube usually does not represent too much effort, so we often recommend backing up the MITS Discover data no more frequently than weekly. It is also typically unnecessary and undesirable to use mirrored drives to store MITS Discover Hypercubes, since a mirrored drive configuration typically slows write performance. It is usually better to use a striped RAID configuration to maximize drive performance.

Processor Speed

Often, the CPU consumption of the transformation process will be quite light. The process will spend most of the time waiting for disk I/O. Where CPU does make an appreciable difference is when the MITS Discover application design requires “stretching” some of the recommendations in this paper. For example, if there is a compelling business reason to provide for drilling down into an identifier combination that yields many rows, the building, maintaining, and serving of these combinations can place a greater load on the CPU. This is because the CPU is heavily involved with the underlying multivalue array manipulation, and as those arrays get larger, the CPU’s role in their creation and maintenance increases.

Memory

Since most database platforms that support MITS Discover use some form of virtual memory management, additional memory will be of benefit to the transformation process only to the degree that it prevents paging activity. If the database platform supports dynamic files it is possible that during the early stages of the transformation process virtually the entire MITS Discover file system will fit into the available system memory, greatly reducing paging activity. As the files expand beyond the available physical memory it is common for the transformation process to slow.

Summary

While pertinent hardware choices can greatly improve MITS Discover application build speeds, the greatest impact in shortening build times will come from a well-designed Hypercube with appropriately refined drill-down paths.

mitsdiscover

MitsMaker User Guide

Appendix C:MITS Discover Commands

MITS Discover Server Commands

This chapter describes some of the commands available in MITS Discover. All of these commands (with the exception of MITS.BOOT) should be run from a TCL database prompt logged to the MITS Discover operational account.

The following list of commands will be presented in this format:

Title

Syntax:

```
How to use this command
```

Description:

A description of what this command does.

Arguments:

Argument	Description
Argument passed to/by the command	What the argument does.

Example:

A practical example of this command's use.

```
Command use example
```

```
Example of possible output generated by command
```

Explanation of what is happening in the example.

MITS

Syntax:

```
MITS [flash_screen_id | mql_statement]
```

Description:

Starts MITS Discover in character mode.

Arguments:

Argument	Description
flash_screen_id	The ID of an existing flash screen.
mql_statement	MITS Query Language expression used to create a flash screen.

Example:

Any valid flash screen can be executed by typing the flash screen ID after the MITS command. (If MitsSecurity is enabled the user will need to first log in.)

```
:MITS
```

Starts MITS Discover in character mode.

Output:

```
                        M I T S

 Copyright (C) 2008 Management Information Tools, Inc.
                  All Rights Reserved

                   Product: MitsServer
                   Version: 0.0.0
                     Build: 2479
          System Signature: UD0032155000
     License Serial Number: MDS-P102790
         Activation Status: Permanent
           Product Edition: Premium
  MitsWeb Dashboard Status: Licensed
```

```
:MITS 2008
```

Output:

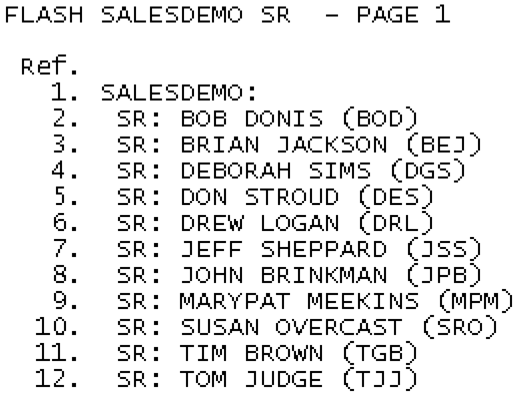
```
Comparison of Profit back 2 years

                        PROFIT    PROFIT DIFFERENCE    TREND
                      YTD thru  YTD thru     PROFIT   PROFIT
                      Dec 2004  Dec 2003   YTD thru YTD thru
                                           Dec 2003 Dec 2003
                                             PROFIT   PROFIT
                                           YTD thru YTD thru
                                           Dec 2004 Dec 2004

1. SALESDEMO:          470,400   363,039    107,361   +29.6%
2.  BOISE BRANCH        22,969    22,352        617    +2.8%
3.  PORTLAND BRANCH    165,071    79,988     85,083  +106.4%
4.  SEATTLE BRANCH     123,047   185,421    -62,374   -33.6%
5.  SPOKANE BRANCH     159,313    75,279     84,035  +111.6%
```

```
:MITS FLASH SALESDEMO SR
```

Output:



```
FLASH SALESDEMO SR  - PAGE 1

 Ref.
   1. SALESDEMO:
   2.  SR: BOB DONIS (BOD)
   3.  SR: BRIAN JACKSON (BEJ)
   4.  SR: DEBORAH SIMS (DGS)
   5.  SR: DON STROUD (DES)
   6.  SR: DREW LOGAN (DRL)
   7.  SR: JEFF SHEPPARD (JSS)
   8.  SR: JOHN BRINKMAN (JPB)
   9.  SR: MARYPAT MEEKINS (MPM)
  10.  SR: SUSAN OVERCAST (SRO)
  11.  SR: TIM BROWN (TGB)
  12.  SR: TOM JUDGE (TJJ)
```

A flash screen may be loaded by its ID or created with MQL on startup by passing optional arguments.

NOTE: Once you are at the `MITS>` prompt, you can run the `BOOTSTRAP.ADMIN` command to initialize MITS Discover security for the first time. More information can be found in the *MITS Discover Installation Guide*.

MITS.ACTIVATE

Syntax:

```
MITS.ACTIVATE
```

Description:

Allows you to access the MITS Discover activation screen. You can update your license information from this screen.

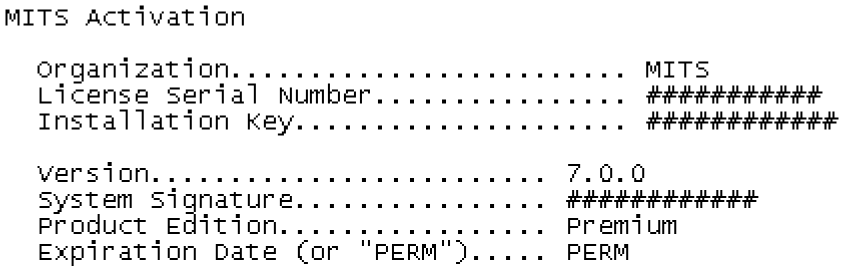
Arguments:

There are no arguments for this command.

Example:

```
:MITS.ACTIVATE
```

Output:



```
MITS Activation

  Organization......................... MITS
  License Serial Number................ ###########
  Installation Key..................... ############

  Version......................... 7.0.0
  System Signature................ ############
  Product Edition................. Premium
  Expiration Date (or "PERM")..... PERM
```

MITS.ACTIVATE can be used to view your current MitsServer license information; it can also be used to update your license information for other server-based MITS Discover products, such as MITS Dashboard.

MITS.BOOT

Syntax:

```
MITS.BOOT account
```

Description:

The MITS.BOOT program updates the named account with the necessary pointers to the MITS.CORE account.

Arguments:

Argument	Description
account	Path to a valid database account that is intended for use as a MITS Discover Operational account.

Example:

```
MITS.BOOT C:\IBM\UD60\MITS.DEMO
```

For more information, see the *MITS Discover Installation Guide*.

WARNING: MITS.BOOT is the only command that should be executed from the MITS.CORE account.

MITS.CALL

MITS.CALL is a subroutine used to execute MITS Discover flash screens and return the flash output to the calling program.

NOTE: MITS.CALL is only available in a booted MITS Discover Operational account.

Syntax

```
MITS.CALL (TOKENS.IN, DATA.IN, TOKENS.OUT, DATA.OUT)
```

Arguments

```
TOKENS.IN and DATA.IN
```

The TOKENS.IN and DATA.IN arguments are parallel arrays that pass values into the calling program. The supporting values are:

Supporting MITS.CALL Values for TOKENS.IN and DATA.IN

Token	Data	Comments
COMMAND	String	The flash (MQL) statement to execute. (See the MQL reference chapter in this document for more information on creating MQL statements.)
USER.ID	String	The user ID to use to execute the flash statement. This is only required if MitsSecurity is enabled.
PASSWORD	String	The password for the user ID specified in the USER.ID token. This is only required if MitsSecurity is enabled.
ROWS	Number	Optional. The number of rows of flash screen results to return.
CONTEXT	Number	Optional. A context number from MITS.CONTEXT. Use in concert with the CONTEXT.SAVE token to retrieve the "next set" of rows for a flash screen that is too large for MITS.CALL to retrieve all at once.
CONTEXT.SAVE	1	Optional. Include this token with a value of "1" to ensure the context for this request is saved. Use in concert with the CONTEXT token to retrieve the "next set" of rows for a flash screen that is too large for MITS.CALL to retrieve all at once. (Do not include this token if you are not using this feature.)

```
TOKENS.OUT and DATA.OUT
```

The TOKENS.OUT and DATA.OUT parameters are parallel arrays returned from the MITS Discover processing engine. The following is a partial list of tokens that may be returned from the MITS.CALL interface with a description of the data returned:

Supporting MITS.CALL Values for TOKENS.OUT and DATA.OUT

TOKENS.OUT Token	DATA.OUT Example
CONTEXT	`741`
MESSAGE	`context 741 saved`
COLUMN FILTERS	
CLIP	`0`
BOTTOM.TOTALS	`0`
NOZEROS	`0`
NOIDS	`0`
NOTAGS	`0`
NOGRAND	`0`
GRID FILL	`100`
ROWED OUT	
ROWS	`1.üSALESDEMO:ü-4`
APP	`SALESDEMO`
EXPANDABLE COLUMNS	
COLUMN EON EXPANSIONS	
COLUMN NAMES	`F3`
COLUMN HEADINGS	`Ref.ýýTRENDüPROFITüQUARTERü2004 Q2üPROFITüQUARTERü2004 Q3`
COLUMN MASKS	`R#5ýL#35ýR#8`
COLUMN HEADING DEPTH	`8`
SHOW OFFSET	`2`
INDENTION EXPLORESS	`CûCustomerüPLûProduct LineüRûRegionüSRûSales Representative…`
INDENTION ABBREVS	
FLASH INDENTIONS	`SALESDEMO`
FLASH HEADING	`FLASH SALESDEMO TREND_PROFIT.Q-2_PROFIT.Q-1`
FLASH	`FLASH SALESDEMO TREND_PROFIT.Q-2_PROFIT.Q-1`

Example (Simple)

This is a simple example that will only work on systems that do not have MITS Discover security enabled.

```
***
*   Program sample that calls into MITS Discover
***
*Build the input tokens
TOKENS.IN = 'COMMAND'
DATA.IN = 'FLASH SALESDEMO SALES.Y'
TOKENS.OUT = ''
DATA.OUT = ''
CALL MITS.CALL(TOKENS.IN, DATA.IN, TOKENS.OUT, DATA.OUT)
*Pull out the SALES.Y data
SALES.YTD = ''
LOCATE 'ROWS' IN TOKENS.OUT SETTING SLOT THEN
   *We only requested one row and one column so the data should be in the first
row, third slot
   SALES.YTD = DATA.OUT<SLOT, 1, 3>
END
PRINT SALES.YTD
```

Example (Advanced)

This example uses the security tokens as well as the ROWS/CONTEXT/CONTEXT.SAVE input tokens.

```
***
* Program example that calls into MITS Discover to retrieve all
* results for a flash screen (25 rows at a time) and prints the
* results to stdout
***
* Initialize the input and output tokens as well as a few other variables
***
TOKENS.IN  = ''
DATA.IN    = ''
TOKENS.OUT = ''
DATA.OUT   = ''
ROWS       = ''
RESULTS    = ''
UID        = 'ADMIN'
PWD        = 'mitspassword'
***
* Build the input token/data pair arrays
***
* The flash statement:
  TOKENS.IN<-1> = 'COMMAND'
  DATA.IN<-1> = 'FLASH SALESDEMO W SR C SALES.Y BY-DSND SALES.Y NOTAGS'
* The user ID:
  TOKENS.IN<-1> = 'USER.ID'
  DATA.IN<-1> = UID
* The password:
  TOKENS.IN<-1> = 'PASSWORD'
  DATA.IN<-1> = PWD
* Number of rows to grab:
  TOKENS.IN<-1> = 'ROWS'
  DATA.IN<-1> = '25'
```

```
* Save the context so we can refer to it again later:
  TOKENS.IN<-1> = 'CONTEXT.SAVE'
  DATA.IN<-1> = 1
***
* Make the initial call to get the first set of rows
***
  CALL MITS.CALL(TOKENS.IN, DATA.IN, TOKENS.OUT, DATA.OUT)
***
* Capture the flash screen results
***
  LOCATE 'ROWS' IN TOKENS.OUT SETTING SLOT THEN
    RESULTS = RESULTS : DATA.OUT<SLOT> : @VM
  END
***
* Loop through and get the rest of the rows, using the context we just started
***
LOOP
  * Get the context number from the previous request
  * and save it to the variable CONTEXT
    LOCATE 'CONTEXT' IN TOKENS.OUT SETTING SLOT ELSE
      PRINT 'Could not find CONTEXT in TOKENS.OUT'
      STOP
    END
    CONTEXT = DATA.OUT<SLOT>
  * Get the next set of rows
    TOKENS.IN = ''
    DATA.IN = ''
    TOKENS.OUT = ''
    DATA.OUT = ''
  * pass in the previous context so it knows where to start from
    TOKENS.IN<-1> = 'CONTEXT'
    DATA.IN<-1> = CONTEXT
  * The user ID:
    TOKENS.IN<-1> = 'USER.ID'
    DATA.IN<-1> = UID
  * The password:
    TOKENS.IN<-1> = 'PASSWORD'
    DATA.IN<-1> = PWD
  * Starting from the context given, go to the next set of rows:
    TOKENS.IN<-1> = 'COMMAND'
    DATA.IN<-1> = 'ROWS 25'
  * Save the context so we can refer to it again later:
    TOKENS.IN<-1> = 'CONTEXT.SAVE'
    DATA.IN<-1> = 1
  * Make the call
    CALL MITS.CALL(TOKENS.IN,DATA.IN,TOKENS.OUT,DATA.OUT)
  * Capture the flash screen results
    LOCATE 'ROWS' IN TOKENS.OUT SETTING SLOT THEN
      RESULTS = RESULTS : DATA.OUT<SLOT> : @VM
    END
  * Check to see if this is the bottom of the flash screen or not
    LOCATE 'ROWED OUT' IN TOKENS.OUT SETTING NOTHING THEN EXIT ELSE CONTINUE
REPEAT
PRINT RESULTS
```

MITS.COMPILE

Syntax:

```
MITS.COMPILE Hypercube ColumnID [ K | V ]
```

Description:

The MITS.COMPILE command is an internal command used to “compile” MITS Discover column definitions. It is run automatically as a part of the everyday use of MITS Discover. For example, when a column is run for the first time after a build wrap-up, MITS.COMPILE is normally run internally by MITS Discover. It can also be run manually to aid in the creation and/or debugging of custom columns. If you've created a column that does not compile properly and you would like to see more information about what may have gone wrong during the compile procedure, run the MITS.COMPILE command with the (**V** option. If you would also like to look at the generated source code, run MITS.COMPILE with the (**K** option, and then look in the MITS.SITE.LOGICS file for an item that has been named using the following convention:

Arguments:

Argument	Description
Hypercube	A Hypercube that currently exists in this MITS Discover operational account.
column	One or more column IDs to be compiled. An asterisk (*) can be used to compile all columns within the named Hypercube.
K	Keep. This option will tell the MITS Discover compiler to keep the generated source code (stored in the MITS.SITE.LOGICS file
V	Runs MITS.COMPILE in verbose mode, causing the program to report more information as it runs.

Example:

```
MITS.COMPILE SALESDEMO SALES.M (V
```

Output:

```
decataloging...
DECATALOG MITS_SALESDEMO_SALES.M
DELETE VOC MITS_SALESDEMO_SALES.M
MITS_SALESDEMO_SALES.M missing from VOC
DELETE MITS.SITE.LOGICS _MITS_SALESDEMO_SALES.M
_MITS_SALESDEMO_SALES.M deleted from MITS.SITE.LOGICS
compiling...
BASIC MITS.SITE.LOGICS MITS_SALESDEMO_SALES.M -D
  CATALOG MITS.SITE.LOGICS MITS_SALESDEMO_SALES.M DIRECT FORCE
DELETE MITS.SITE.LOGICS MITS_SALESDEMO_SALES.M
MITS_SALESDEMO_SALES.M deleted from MITS.SITE.LOGICS
DELETE MITS.SITE.LOGICS $MITS_SALESDEMO_SALES.M
$MITS_SALESDEMO_SALES.M missing from MITS.SITE.LOGICS
```

WARNING: *Do not modify the generated code found in this file.* It is generated for troubleshooting and debugging purposes only.

MITS.DELETE.APP

Syntax:

```
MITS.DELETE.APP Hypercube (Y
```

Description:

This removes all components of the specified Hypercube from the server.

WARNING: The original specification in the MITS.MAKER.SPECS file is also removed using this command.

Arguments:

Argument	Description
Hypercube	A Hypercube that currently exists in this MITS Discover operational account.
Y	Yes. This verifies that you do indeed wish to delete the entire application. If you do not use this option, the command will not be completed.

Example:

```
:MITS.DELETE.APP SALESDEMO (Y
```

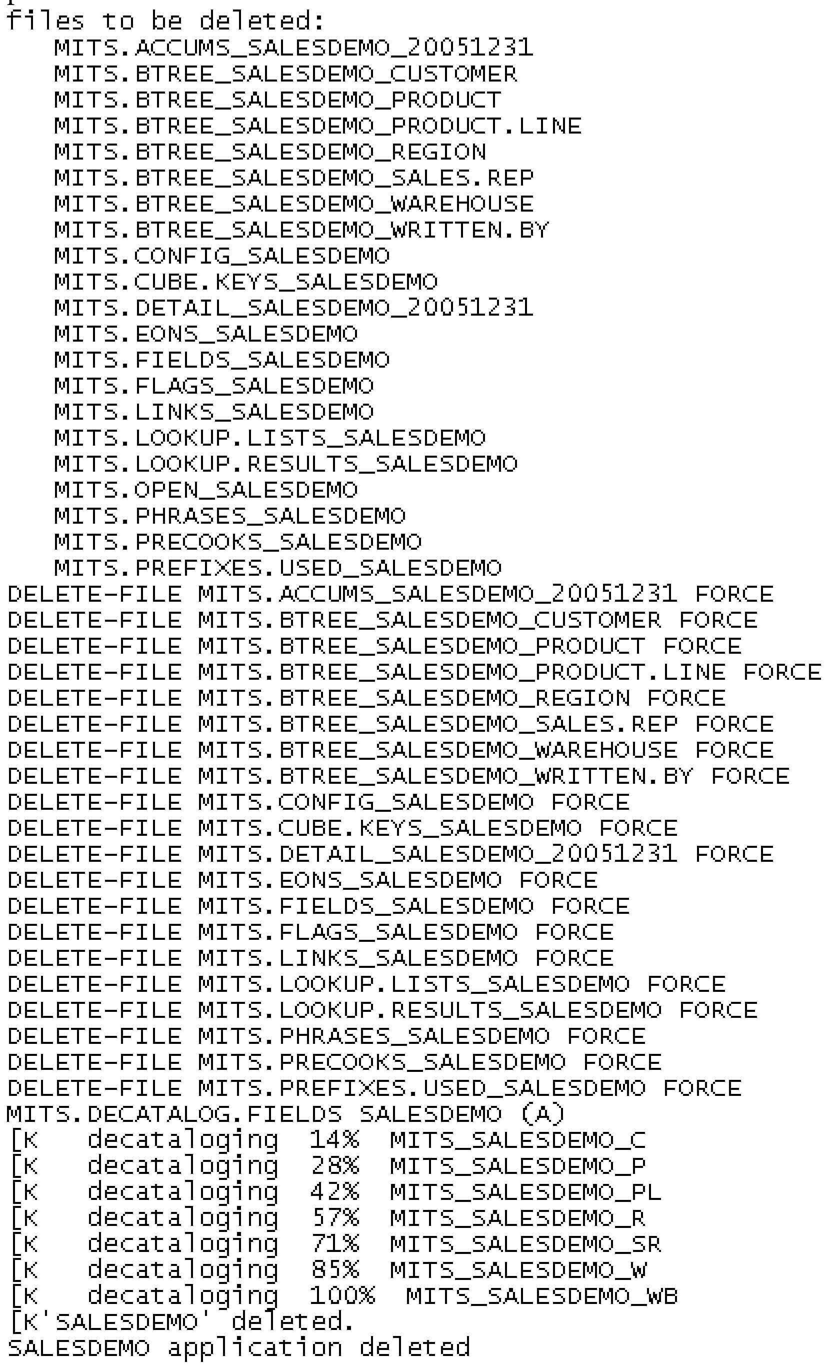
If you created an application and have no further use for it you may wish to delete the application from the server. MITS.DELETE.APP will remove all the pieces of application that are put in place when the application was created.

NOTE: Any stored flash screens (macros) that reference this application are left in place.

WARNING: An application deleted with this command *cannot be recovered.*

Sample output from this command can be found on the next page.

Output:



```
files to be deleted:
   MITS.ACCUMS_SALESDEMO_20051231
   MITS.BTREE_SALESDEMO_CUSTOMER
   MITS.BTREE_SALESDEMO_PRODUCT
   MITS.BTREE_SALESDEMO_PRODUCT.LINE
   MITS.BTREE_SALESDEMO_REGION
   MITS.BTREE_SALESDEMO_SALES.REP
   MITS.BTREE_SALESDEMO_WAREHOUSE
   MITS.BTREE_SALESDEMO_WRITTEN.BY
   MITS.CONFIG_SALESDEMO
   MITS.CUBE.KEYS_SALESDEMO
   MITS.DETAIL_SALESDEMO_20051231
   MITS.EONS_SALESDEMO
   MITS.FIELDS_SALESDEMO
   MITS.FLAGS_SALESDEMO
   MITS.LINKS_SALESDEMO
   MITS.LOOKUP.LISTS_SALESDEMO
   MITS.LOOKUP.RESULTS_SALESDEMO
   MITS.OPEN_SALESDEMO
   MITS.PHRASES_SALESDEMO
   MITS.PRECOOKS_SALESDEMO
   MITS.PREFIXES.USED_SALESDEMO
DELETE-FILE MITS.ACCUMS_SALESDEMO_20051231 FORCE
DELETE-FILE MITS.BTREE_SALESDEMO_CUSTOMER FORCE
DELETE-FILE MITS.BTREE_SALESDEMO_PRODUCT FORCE
DELETE-FILE MITS.BTREE_SALESDEMO_PRODUCT.LINE FORCE
DELETE-FILE MITS.BTREE_SALESDEMO_REGION FORCE
DELETE-FILE MITS.BTREE_SALESDEMO_SALES.REP FORCE
DELETE-FILE MITS.BTREE_SALESDEMO_WAREHOUSE FORCE
DELETE-FILE MITS.BTREE_SALESDEMO_WRITTEN.BY FORCE
DELETE-FILE MITS.CONFIG_SALESDEMO FORCE
DELETE-FILE MITS.CUBE.KEYS_SALESDEMO FORCE
DELETE-FILE MITS.DETAIL_SALESDEMO_20051231 FORCE
DELETE-FILE MITS.EONS_SALESDEMO FORCE
DELETE-FILE MITS.FIELDS_SALESDEMO FORCE
DELETE-FILE MITS.FLAGS_SALESDEMO FORCE
DELETE-FILE MITS.LINKS_SALESDEMO FORCE
DELETE-FILE MITS.LOOKUP.LISTS_SALESDEMO FORCE
DELETE-FILE MITS.LOOKUP.RESULTS_SALESDEMO FORCE
DELETE-FILE MITS.PHRASES_SALESDEMO FORCE
DELETE-FILE MITS.PRECOOKS_SALESDEMO FORCE
DELETE-FILE MITS.PREFIXES.USED_SALESDEMO FORCE
MITS.DECATALOG.FIELDS SALESDEMO (A)
[K   decataloging  14%  MITS_SALESDEMO_C
[K   decataloging  28%  MITS_SALESDEMO_P
[K   decataloging  42%  MITS_SALESDEMO_PL
[K   decataloging  57%  MITS_SALESDEMO_R
[K   decataloging  71%  MITS_SALESDEMO_SR
[K   decataloging  85%  MITS_SALESDEMO_W
[K   decataloging  100%  MITS_SALESDEMO_WB
[K'SALESDEMO' deleted.
SALESDEMO application deleted
```

MITS.LINK.SORT

Syntax:

```
MITS.LINK.SORT Hypercube ([ F | L | R | Z ]
```

Description:

The MITS.LINK.SORT process is run at the end of a build process's wrap-up phase. This program sorts the identifier value lists by their descriptions. The link sort process can potentially take a long time to complete. If the sort process is taking a long time you may wish to consider stopping this process in the wrap-up and running it manually at a later time. This may be especially desirable if you are building initial transactional history into an accumulated MITS Discover Hypercube.

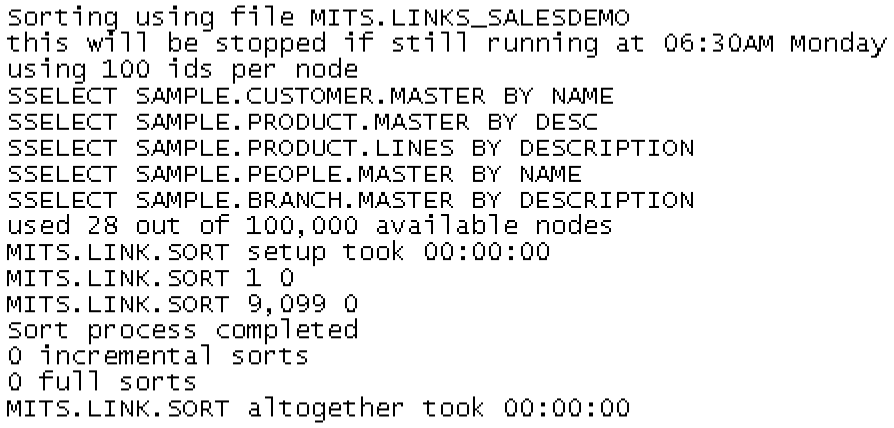
Arguments:

Argument	Description
Hypercube	A Hypercube that currently exists in this MITS Discover operational account.
F	Forces a re-sort of all the identifier links. Use this option if you have run the link sort process but are still experiencing incorrect default sorting on some identifiers. Often used with the (**L** option when the sort order for an identifier has been changed.
L	Used with the (**F** option when the sort order has been changed on an identifier.
R	Deltas only. This option will sort only the entries in the links file that have been flagged for update.
Z	No Interrupt. This should be used when the command is manually executed in a phantom process.

Example:

```
:MITS.LINK.SORT SALESDEMO (FL
```

Output:



```
Sorting using file MITS.LINKS_SALESDEMO
this will be stopped if still running at 06:30AM Monday
using 100 ids per node
SSELECT SAMPLE.CUSTOMER.MASTER BY NAME
SSELECT SAMPLE.PRODUCT.MASTER BY DESC
SSELECT SAMPLE.PRODUCT.LINES BY DESCRIPTION
SSELECT SAMPLE.PEOPLE.MASTER BY NAME
SSELECT SAMPLE.BRANCH.MASTER BY DESCRIPTION
used 28 out of 100,000 available nodes
MITS.LINK.SORT setup took 00:00:00
MITS.LINK.SORT 1 0
MITS.LINK.SORT 9,099 0
Sort process completed
0 incremental sorts
0 full sorts
MITS.LINK.SORT altogether took 00:00:00
```

MITS.UPDATE.BTREES

Syntax:

```
MITS.UPDATE.BTREES Hypercube ([ Z ]
```

Description:

This command repopulates the indexes used for the select exploration feature.

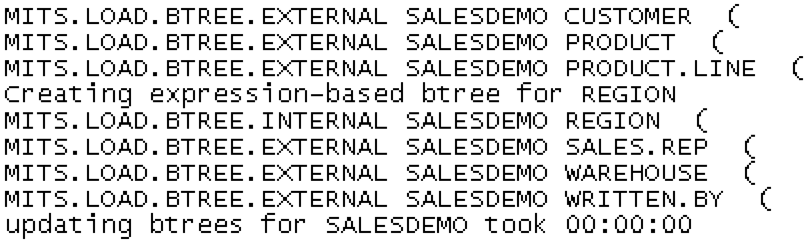
Arguments:

Argument	Description
Hypercube	A Hypercube that currently exists in this MITS Discover operational account.
Z	No Interrupt. This should be used when the process is executed in a phantom process, and the phantom is sensitive to the program checking for user input.

Example:

```
:MITS.UPDATE.BTREES SALESDEMO (R
```

Output:



```
MITS.LOAD.BTREE.EXTERNAL SALESDEMO CUSTOMER  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO PRODUCT  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO PRODUCT.LINE  (
Creating expression-based btree for REGION
MITS.LOAD.BTREE.INTERNAL SALESDEMO REGION  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO SALES.REP  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO WAREHOUSE  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO WRITTEN.BY  (
updating btrees for SALESDEMO took 00:00:00
```

MITS.LOAD.RELATIVE.EONS

Syntax:

```
MITS.LOAD.RELATIVE.EONS Hypercube ([ V ]
```

Description:

This command repopulates the MITS.EONS_<<application name>> file with eons that are relative in nature to the date of the last transaction in the Hypercube.

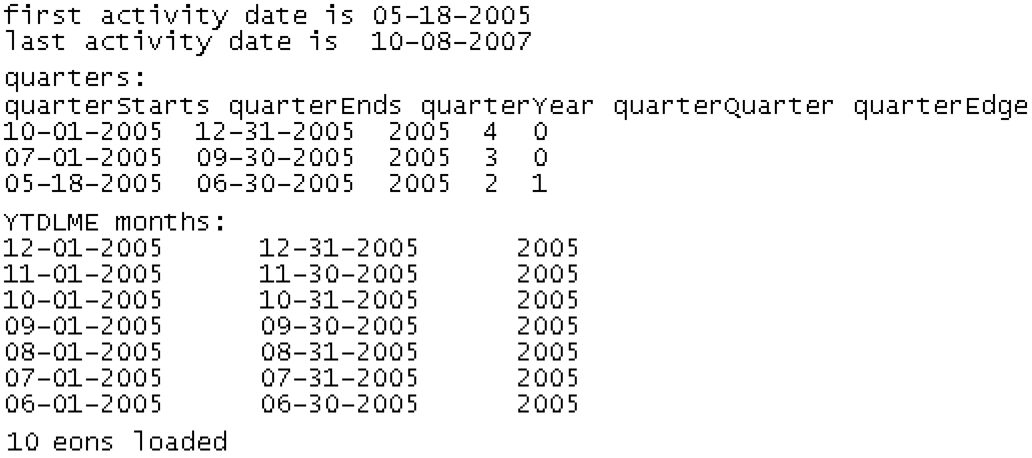
Arguments:

Argument	Description
Hypercube	A Hypercube that currently exists in this MITS Discover operational account.
V	Runs MITS.LOAD.RELATIVE.EONS in verbose mode, causing the program to report more information as it runs.

Example:

```
:MITS.LOAD.RELATIVE.EONS SALESDEMO (V
```

Output:



```
first activity date is 05-18-2005
last activity date is  10-08-2007
quarters:
quarterStarts quarterEnds quarterYear quarterQuarter quarterEdge
10-01-2005  12-31-2005  2005  4  0
07-01-2005  09-30-2005  2005  3  0
05-18-2005  06-30-2005  2005  2  1
YTDLME months:
12-01-2005      12-31-2005      2005
11-01-2005      11-30-2005      2005
10-01-2005      10-31-2005      2005
09-01-2005      09-30-2005      2005
08-01-2005      08-31-2005      2005
07-01-2005      07-31-2005      2005
06-01-2005      06-30-2005      2005
10 eons loaded
```

Eons are used in conjunction with column templates to create column definitions.

Relative Eons are the most common eon type used in MITS Discover. They are described in terms of their relation to the date of the *last transaction in the Hypercube*. For example, an eon of M-1 is defined as being “back one month,” or the month prior to the month that includes the last transaction in the Hypercube.

MITS.LOAD.ABSOLUTE.EONS

Syntax:

```
MITS.LOAD.ABSOLUTE.EONS Hypercube ([ V ]
```

Description:

This command repopulates the MITS.EONS_<<Hypercube name>> file with eons that are absolute in nature.

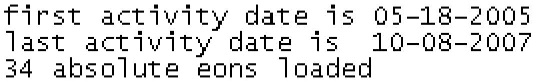
Arguments:

Argument	Description
Hypercube	A Hypercube that currently exists in this MITS Discover operational account.
V	Runs MITS.LOAD.ABSOLUTE.EONS in verbose mode, causing the program to report more information as it runs.

Example:

```
:MITS.LOAD.ABSOLUTE.EONS SALESDEMO
```

Output:



```
first activity date is 05-18-2005
last activity date is  10-08-2007
34 absolute eons loaded
```

NOTE: This type of eon is normally used in a snapshot-type Hypercube. See also MITS.LOAD.RELATIVE.EONS.

MITS.MAKER.BUILD

Syntax:

```
MITS.MAKER.BUILD Hypercube ([ ## | { B | W } | F | H | R | U | V | Z ]
```

Description:

Starts the MITS Discover Hypercube build process, which transforms source transactions into the MITS Discover Hypercube.

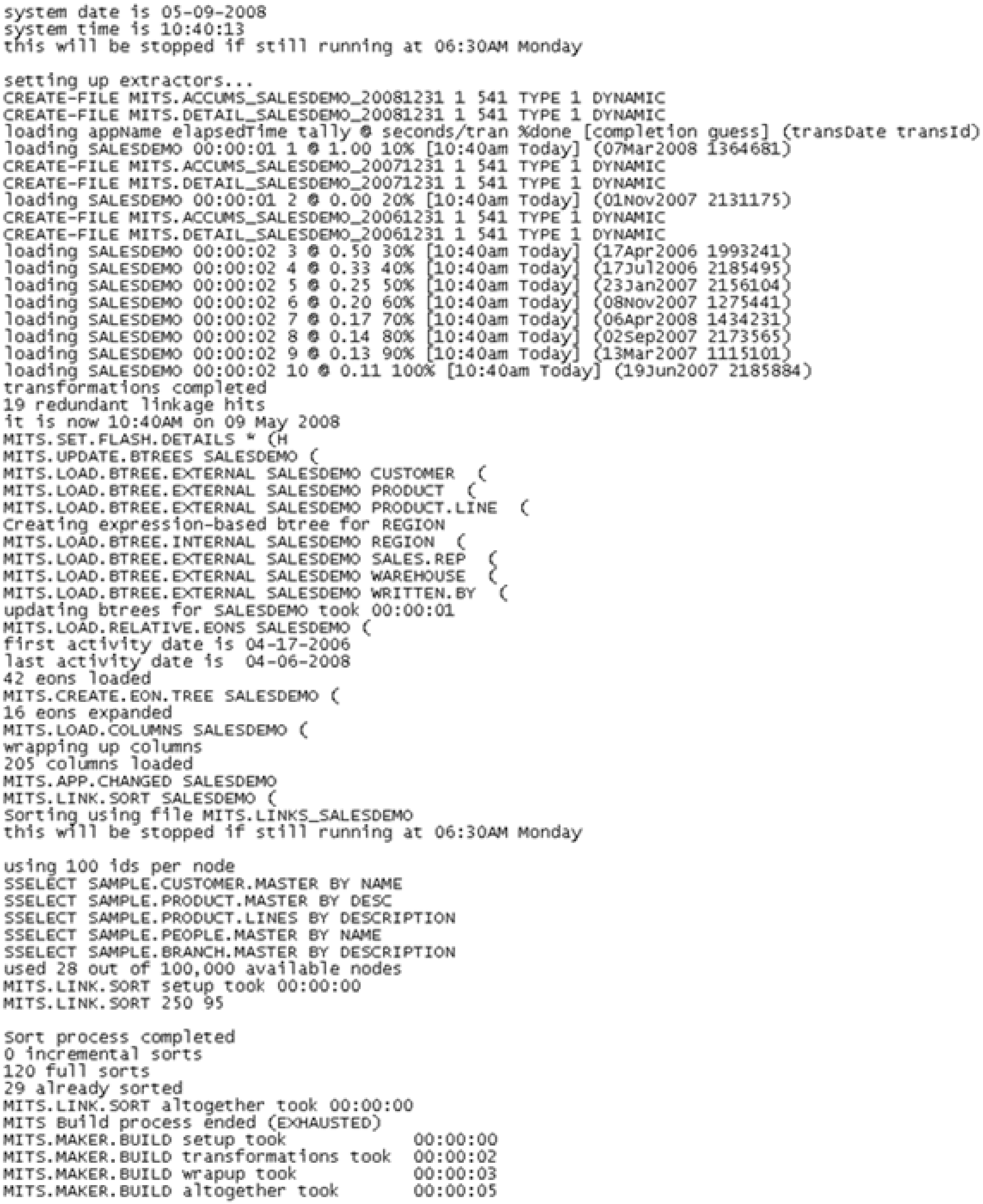
Arguments:

Argument	Description
Hypercube	A Hypercube that currently exists in this MITS Discover operational account.
##	Entering a number will limit the build to the specified number of transactions.
B	Runs the build transformation phase only. No wrap-up phase will be run.
F	Forces the Hypercube data to be purged prior to building.
H	Hushed. Causes the build to progress with an abbreviated amount of output.
I	Perform a standard select instead of a sort select when selecting transactions to add to the Hypercube.
K	Skip PREPROCESS
L	Skip POSTPROCESS
N	Skip Dynamic Change Update phase.
U	Dynamic Change Update only. Only items in the MITS.TODOS_<app>, MITS.UNDOS_<app>, and MITS.REDOS_<app> files will be processed, followed by a wrap-up. Note: Including the (**B** option will cause the wrap-up to be skipped.
V	Verbose. Causes the build to return more output. This is often useful for troubleshooting purposes.
W	Wrap-up only. Runs the build wrap-up phase only. No transformation phase will be run.
Z	Suppresses input prompts. This should be used when the MITS.MAKER.BUILD program is executed in a background or phantom process, especially on UniVerse systems.

Example:

```
:MITS.MAKER.BUILD SALESDEMO
```

Here is an example of the output generated by this command:



```
system date is 05-09-2008
system time is 10:40:13
this will be stopped if still running at 06:30AM Monday

setting up extractors...
CREATE-FILE MITS.ACCUMS_SALESDEMO_20081231 1 541 TYPE 1 DYNAMIC
CREATE-FILE MITS.DETAIL_SALESDEMO_20081231 1 541 TYPE 1 DYNAMIC
loading appName elapsedTime tally @ seconds/tran %done [completion guess] (transDate transId)
loading SALESDEMO 00:00:01 1 @ 1.00 10% [10:40am Today] (07Mar2008 1364681)
CREATE-FILE MITS.ACCUMS_SALESDEMO_20071231 1 541 TYPE 1 DYNAMIC
CREATE-FILE MITS.DETAIL_SALESDEMO_20071231 1 541 TYPE 1 DYNAMIC
loading SALESDEMO 00:00:01 2 @ 0.00 20% [10:40am Today] (01Nov2007 2131175)
CREATE-FILE MITS.ACCUMS_SALESDEMO_20061231 1 541 TYPE 1 DYNAMIC
CREATE-FILE MITS.DETAIL_SALESDEMO_20061231 1 541 TYPE 1 DYNAMIC
loading SALESDEMO 00:00:02 3 @ 0.50 30% [10:40am Today] (17Apr2006 1993241)
loading SALESDEMO 00:00:02 4 @ 0.33 40% [10:40am Today] (17Jul2006 2185495)
loading SALESDEMO 00:00:02 5 @ 0.25 50% [10:40am Today] (23Jan2007 2156104)
loading SALESDEMO 00:00:02 6 @ 0.20 60% [10:40am Today] (08Nov2007 1275441)
loading SALESDEMO 00:00:02 7 @ 0.17 70% [10:40am Today] (06Apr2008 1434231)
loading SALESDEMO 00:00:02 8 @ 0.14 80% [10:40am Today] (02Sep2007 2173565)
loading SALESDEMO 00:00:02 9 @ 0.13 90% [10:40am Today] (13Mar2007 1115101)
loading SALESDEMO 00:00:02 10 @ 0.11 100% [10:40am Today] (19Jun2007 2185884)
transformations completed
19 redundant linkage hits
it is now 10:40AM on 09 May 2008
MITS.SET.FLASH.DETAILS * (H
MITS.UPDATE.BTREES SALESDEMO (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO CUSTOMER  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO PRODUCT  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO PRODUCT.LINE  (
Creating expression-based btree for REGION
MITS.LOAD.BTREE.INTERNAL SALESDEMO REGION  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO SALES.REP  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO WAREHOUSE  (
MITS.LOAD.BTREE.EXTERNAL SALESDEMO WRITTEN.BY  (
updating btrees for SALESDEMO took 00:00:01
MITS.LOAD.RELATIVE.EONS SALESDEMO (
first activity date is 04-17-2006
last activity date is  04-06-2008
42 eons loaded
MITS.CREATE.EON.TREE SALESDEMO (
16 eons expanded
MITS.LOAD.COLUMNS SALESDEMO (
wrapping up columns
205 columns loaded
MITS.APP.CHANGED SALESDEMO
MITS.LINK.SORT SALESDEMO (
Sorting using file MITS.LINKS_SALESDEMO
this will be stopped if still running at 06:30AM Monday

using 100 ids per node
SSELECT SAMPLE.CUSTOMER.MASTER BY NAME
SSELECT SAMPLE.PRODUCT.MASTER BY DESC
SSELECT SAMPLE.PRODUCT.LINES BY DESCRIPTION
SSELECT SAMPLE.PEOPLE.MASTER BY NAME
SSELECT SAMPLE.BRANCH.MASTER BY DESCRIPTION
used 28 out of 100,000 available nodes
MITS.LINK.SORT setup took 00:00:00
MITS.LINK.SORT 250 95

Sort process completed
0 incremental sorts
120 full sorts
29 already sorted
MITS.LINK.SORT altogether took 00:00:00
MITS Build process ended (EXHAUSTED)
MITS.MAKER.BUILD setup took              00:00:00
MITS.MAKER.BUILD transformations took    00:00:02
MITS.MAKER.BUILD wrapup took             00:00:03
MITS.MAKER.BUILD altogether took         00:00:05
```

MITS.PURGE.APP

Syntax:

```
MITS.PURGE.APP Hypercube (F
```

Description:

This command removes all the data in a Hypercube.

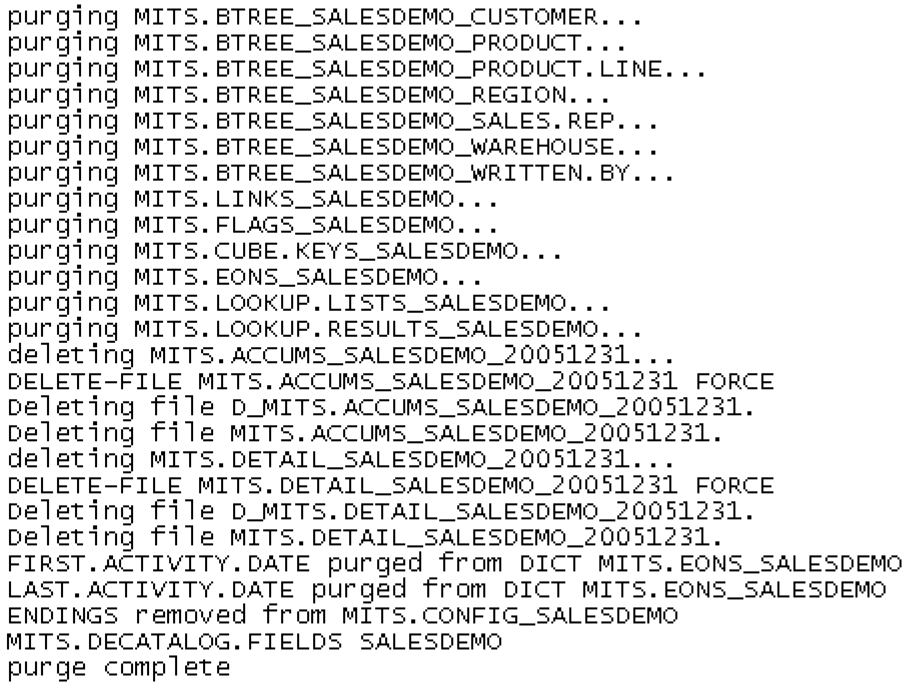
Arguments:

Argument	Description
Hypercube	A Hypercube that currently exists in this MITS Discover operational account.
F	Force. If a file cannot be deleted because of a file lock, it will be cleared instead.

Example:

```
:MITS.PURGE.APP SALESDEMO (F
```

Output:



```
purging MITS.BTREE_SALESDEMO_CUSTOMER...
purging MITS.BTREE_SALESDEMO_PRODUCT...
purging MITS.BTREE_SALESDEMO_PRODUCT.LINE...
purging MITS.BTREE_SALESDEMO_REGION...
purging MITS.BTREE_SALESDEMO_SALES.REP...
purging MITS.BTREE_SALESDEMO_WAREHOUSE...
purging MITS.BTREE_SALESDEMO_WRITTEN.BY...
purging MITS.LINKS_SALESDEMO...
purging MITS.FLAGS_SALESDEMO...
purging MITS.CUBE.KEYS_SALESDEMO...
purging MITS.EONS_SALESDEMO...
purging MITS.LOOKUP.LISTS_SALESDEMO...
purging MITS.LOOKUP.RESULTS_SALESDEMO...
deleting MITS.ACCUMS_SALESDEMO_20051231...
DELETE-FILE MITS.ACCUMS_SALESDEMO_20051231 FORCE
Deleting file D_MITS.ACCUMS_SALESDEMO_20051231.
Deleting file MITS.ACCUMS_SALESDEMO_20051231.
deleting MITS.DETAIL_SALESDEMO_20051231...
DELETE-FILE MITS.DETAIL_SALESDEMO_20051231 FORCE
Deleting file D_MITS.DETAIL_SALESDEMO_20051231.
Deleting file MITS.DETAIL_SALESDEMO_20051231.
FIRST.ACTIVITY.DATE purged from DICT MITS.EONS_SALESDEMO
LAST.ACTIVITY.DATE purged from DICT MITS.EONS_SALESDEMO
ENDINGS removed from MITS.CONFIG_SALESDEMO
MITS.DECATALOG.FIELDS SALESDEMO
purge complete
```

MITS.STOP

Syntax:

```
MITS.STOP SALESDEMO ([N]
```

Description:

This command can be used to send a stop request to a Hypercube build.

Arguments:

Argument	Description
Hypercube	A Hypercube that currently exists in this MITS Discover operational account.
N	Now. Causes the build process to proceed to the next logical phase of the build.

Example:

```
MITS.STOP SALESDEMO (N
```

This command does not generate any output. See the build status on the Hypercubes tab to verify the status of the build.

Drill-Down Path Usage Logging

MITS Discover now has the ability to keep a log of the drill-down paths that are used most often in a Hypercube so that drill-down paths which are not used regularly can be removed. Removing seldom-used drill-down paths from a Hypercube can greatly improve build speeds and lessen disk space requirements for that Hypercube.

Important Files and Commands:

- **MITS.DRILLDOWN.USAGE** - This program is run from a database prompt that has been logged to the MITS Discover operational account. When run, the drill-down path usage statistics are displayed. This utility is described in more detail below.
- **FLASH.LOGGING.ENABLED** - This item must exist in the site configuration file (the MITS.CONFIG file) for this feature to be active.
- **DRILLDOWN.HISTORY.MAXIMUM** - By default, only 1000 drill-down actions are maintained by the system. This number can be changed by modifying the DRILLDOWN.HISTORY.MAXIMUM item in the site configuration file (the MITS.CONFIG file). All client sessions (including MITS Discover gateway host connections) need to be restarted in order for a change to this setting to take effect.

Specifications:

To set up this feature, create an empty item in the MITS.CONFIG file within the MITS Discover operational account named **FLASH.LOGGING.ENABLED**. Alternately, create another item in the MITS.CONFIG file named **DRILLDOWN.HISTORY.MAXIMUM**. On attribute one of this item, specify the number of drill-down actions that the system should maintain before rolling over and overwriting the oldest records.

To generate the Drill-Down Path Usage Report, run the following command from a database prompt:

```
MITS.DRILLDOWN.USAGE [ASCENDING|DESCENDING|UNSORTED] [SHOWALL|SHOWTREE]
[NODASHBOARD] [APPLICATION] [HELP]
```

Command Options for MITS.DRILLDOWN.USAGE

Option	Description
ASCENDING	Sorts the drill-down history in ascending order by usage.
DESCENDING	Sorts the drill-down history in descending order by usage.
UNSORTED	Displays the drill-down history unsorted.
SHOWALL	Displays all of the drill-down paths, regardless of whether or not they are being used.
SHOWTREE	Similar to SHOWALL, but sorts the view by drill-down path instead of usage.
NODASHBOARD	Specifies that the program should not consider drill-down usage that occurred as a result of using MITS Dashboard.
APPLICATION	Specifies which Hypercube Application to reference.

Command Options for MITS.DRILLDOWN.USAGE

Option	Description
HELP	Displays information on the use of this command.

Resetting Usage Data

To reset the usage data, clear the MITS.DRILLDOWN.HISTORY file and set the value in attribute one of the LAST.ID item within the DICT of MITS.DRILLDOWN.HISTORY to zero (0).

Index

Year-End Month 42