Tabula Manager’s Guide

Document revision 1.1, 1999/01/20 13:13:17

Tabula Managers Guide

Tabula provides a way of publishing tabular based data within Zope. Reports, search interfaces, hierarchical views, and computed fields can be used to implement data-backed Web based applications quickly and flexibly.

Tabula combines the simplicity and speed of tabular data with Zope’s rich object model. Tabular content and presentation are kept separate, allowing a database administrator and a Web developer to keep their jobs separate and focused. A Web developer can use Zope’s powerful Document Template Markup Language to place database fields freely on a Web page, do batch processing (displaying only 20 records at a time for instance), and interface with other Zope objects and values. The person maintaining the database can upload data from their desktop computer in a single click, set up indexes to speed up searching, and even manage computed fields using Zope’s powerful expression language.

Getting Started - A Guided Tour

Scenario Overview

Plutonia has been selling a new line of Widgets over the past year, and the results are in. The widgets are a small enough line of products that they have been tracked in an advanced desktop database capable of doing small relational and other jobs. One of the database files, UnitsSold, is a small file with information about how strongly a particular widget sold over the past year in the four regions Plutonia distributes their products to. Plutonia management has asked Joe, who is responsible for maintaining the database, to publish the year-end information on the company intranet. "Nothing fancy," they said, "just enough to show how strongly each product has done and in which regions." Fortunately, Joe has been tracking this information already in the file, UnitsSold. He just needs to get it on the Web. More importantly, he needs to set up some options to enable different views of the data. Fortunately, Joe has *Tabula*.

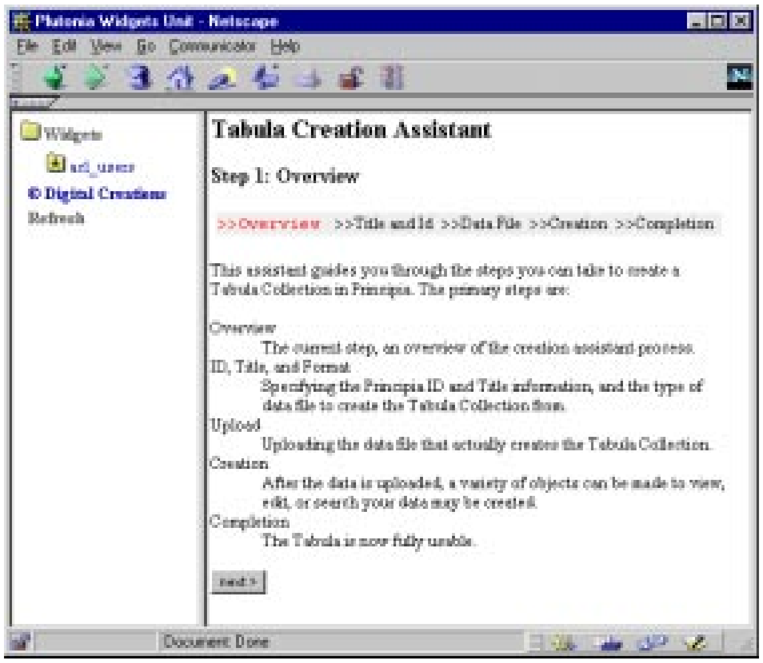
A major component of the software suite that runs on top of Zope, Tabula adds tabular data publishing capabilities to Zope's object model.

Tabula Creation

The first thing Joe does is export his UnitsSold file as a DBF file, a common database format, named *UnitsSold.dbf*. Now he needs to create a new Tabula. There is already a Zope folder for the Widgets division and Joe has security clearance to add new items to the folder, so all he needs to do is go to the Widgets folder's management screen. (To learn more about Zope folders, management screens and other terminology, consult the **Zope Manage's Guide**). He goes to the Add List and selects "Tabula Collection" and clicks "Add." Step 1 of the *Tabula Creation Assistant* (TCA) comes up, as shown in Figure 1.

Since there may be multiple steps involved in setting up a Tabula such as defining indexes, generating a report, and generating a search form, the TCA groups some of these steps together at creation time (at the users choice) to streamline the initial setting-up process. A fully functioning Tabula with a search form and a report to show the data can all be done with the TCA, which is what Joe is going to do now. After reading over Step 1, which is an overview of the following steps, he clicks the "next>" button to go on to Step 2

In Step 2 (Figure 2), he needs to set the id, title, and the format of the data file to be published. An id is required by every Zope object and can consist of any characters legal in a URL id. Letters, numbers, and most punctuation, including periods are acceptable. Since the Tabula Joe's creating concerns the number of Widgets sold per region, he sets the id to `UnitsSold`. The title is an attribute that all Zope objects have. It is always an optional field to fill in, but is recommended since it offers a more detailed description of the object. Joe gives the Tabula the title `Plutonia Widget units sold per region`. Lastly, he has to set the format of the data file to be uploaded. The choices are "DBase III, IV, V" (any DBF file), "DBase III, IV, V with Memo" (for DBF files with external memo files usually, .DBT files), and "Rand RDB" (a tab-delimited format with extra header information). Since Joe's data is in DBF, he chooses "DBase III, IV, V" and clicks the "next>" button to go on to Step 3.


Plutonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Widgets
acl_users
© Digital Creations
Refresh
next >
Document: Done

Tabula Creation Assistant

Step 1: Overview

>>Overview >>Title and Id >>Data File >>Creation >>Completion

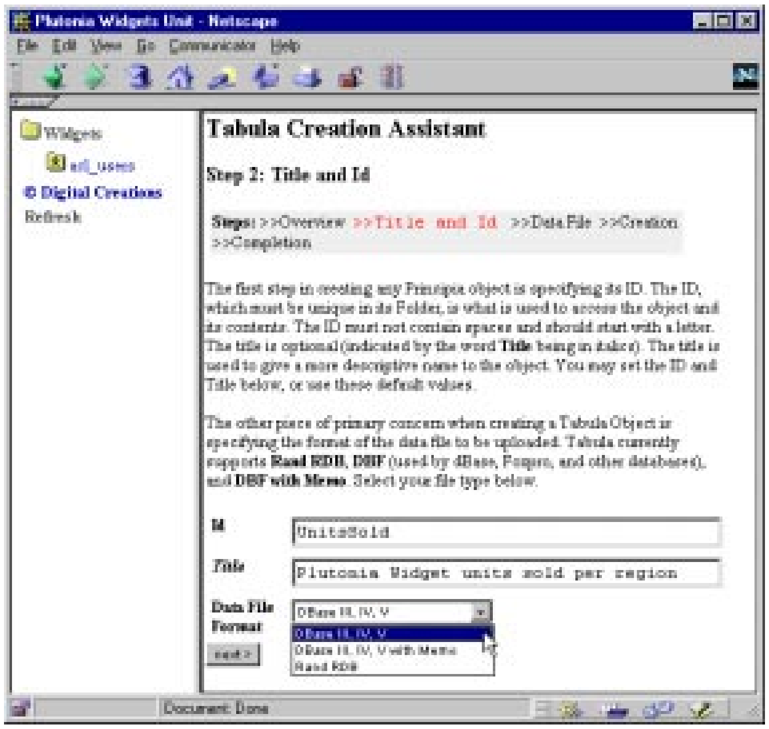
This assistant guides you through the steps you can take to create a Tabula Collection in Principia. The primary steps are:

Overview
The current step, an overview of the creation assistant process.
ID, Title, and Format
Specifying the Principia ID and Title information, and the type of data file to create the Tabula Collection from.
Upload
Uploading the data file that actually creates the Tabula Collection.
Creation
After the data is uploaded, a variety of objects can be made to view, edit, or search your data may be created.
Completion
The Tabula is now fully usable.

Figure 1. Step 1 of the TCA: Overview

In Step 3 (Figure 3), Joe needs to select the data file from his hard disk for uploading. There is a single field and a "Browse..." button that allows Joe to choose the file for uploading. If he was publishing a DBF file with an external Memo file, there would be a second field for the Memo file. After choosing the *UnitsSold.dbf* file off of his hard drive, he clicks the "next>" button to go on to Step 4. *Note: Depending on the size of the file being uploaded and the speeds of both the Internet connection and the computer to which the data is being uploaded, there may be a sizable pause between steps 3 and 4 as the data uploads.*

When Joe gets to Step 4 (Figure 4), he notices the Zope Navigation Frame changes to include his `UnitsSold` Tabula. The Tabula is now actually created but only partially ready for, since it is empty (no reports or search forms defined yet or indexes set on certain fields). Step 4 in the TCA gives the user three choices of sub-operations to perform to fully prepare the Tabula for use. Joe is presented with the following choices: "Set up indexes...", "Create a report...", and "Create a search form...". The "Create a report..." option is already checked (but can be unchecked) since it is the recommended step, and required in order to successfully generate a search form. Joe's orders were "...to show how strongly each product has done and in which regions." Deciding he needs all three options, he checks all three checkboxes and clicks "next>" to go on to the next operation.


Plutonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Widgets
acl_users
© Digital Creations
Refresh
next >
Document: Done

Tabula Creation Assistant

Step 2: Title and Id

Steps: >>Overview >>Title and Id >>Data File >>Creation >>Completion

The first step in creating any Principia object is specifying its ID. The ID, which must be unique in its Folder, is what is used to access the object and its contents. The ID must not contain spaces and should start with a letter. The title is optional (indicated by the word **Title** being in italics). The title is used to give a more descriptive name to the object. You may set the ID and Title below, or use these default values.

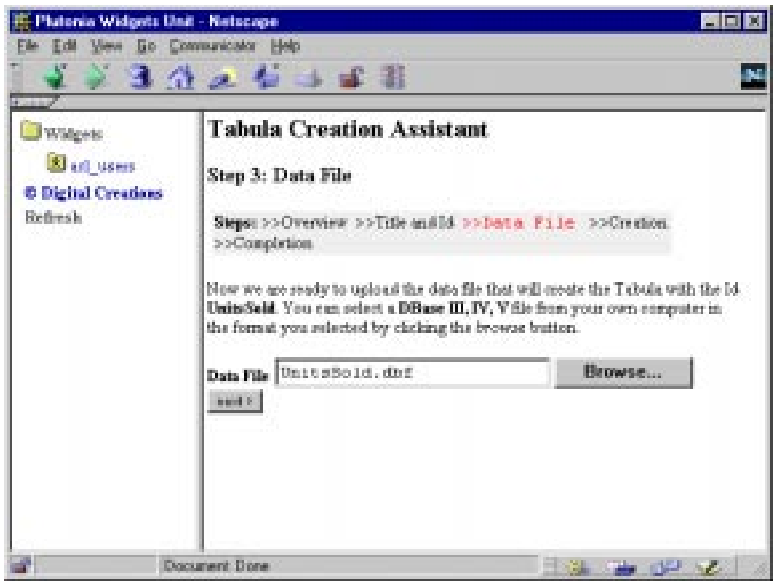
The other piece of primary concern when creating a Tabula Object is specifying the format of the data file to be uploaded. Tabula currently supports **Rand RDB**, **DBF** (used by dBase, Foxpro, and other databases), and **DBF with Memo**. Select your file type below.

Id UnitsSold

Title Plutonia Widget units sold per region

Data File Format DBase III, IV, V
- DBase III, IV, V
- DBase III, IV, V with Memo
- Rand RDB

Figure 2. Step 2 of the TCA: Title and Id



Tabula Creation Assistant

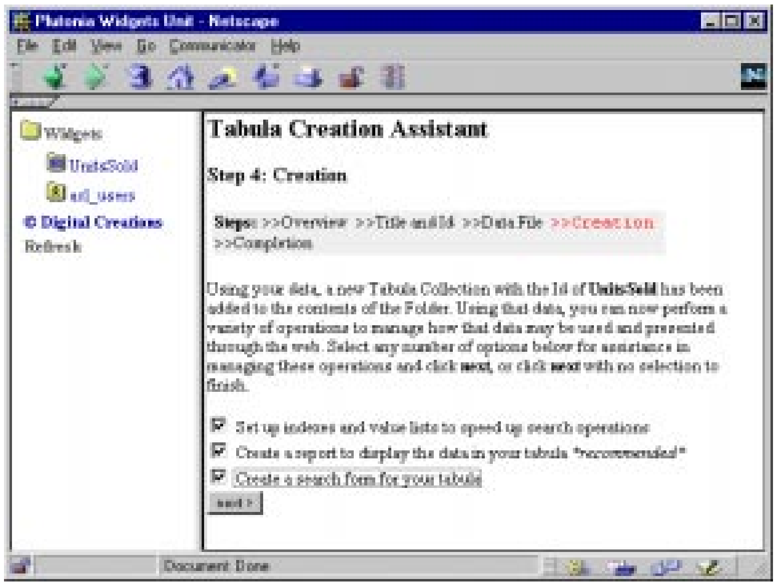
Step 3: Data File

Steps: >>Overview >>Title and Id >>Data File >>Creation >>Completion

Now we are ready to upload the data file that will create the Tabula with the Id **UnitsSold**. You can select a **DBase III, IV, V** file from your own computer in the format you selected by clicking the browse button.

Data File UnitsSold.dbf

Figure 3. Step 4 of the TCA: Data File


Platonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Widgets
UnitsSold
acl_users
© Digital Creations
Refresh
Document Done

Tabula Creation Assistant

Step 4: Creation

Steps: >>Overview >>Title and Id >>Data File >>Creation >>Completion

Using your data, a new Tabula Collection with the Id of **UnitsSold** has been added to the contents of the Folder. Using that data, you can now perform a variety of operations to manage how that data may be used and presented through the web. Select any number of options below for assistance in managing these operations and click **next**, or click **next** with no selection to finish.

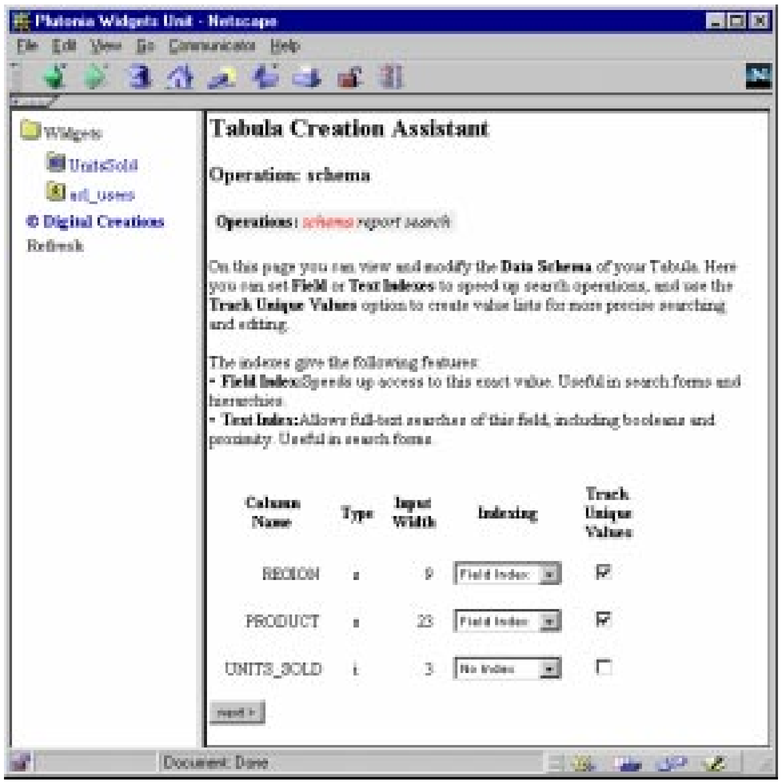
[✓] Set up indexes and value lists to speed up search operations
[✓] Create a report to display the data in your tabula *recommended*
[✓] Create a search form for your tabula

next >

Figure 4. Step 4 of the TCA: Creation

He is next brought to the *Schema* screen which shows all the columns (or *fields*) that were in the uploaded data. His schema is *REGION*, *PRODUCT*, and *UNITS_SOLD* (remember, he uploaded from a DBF file, which always has upper-case column names). People will most likely be searching on *REGION* and *PRODUCT*, so he sets the index options (represented by the drop-down lists) on those columns to `FieldIndex`. He also checks the `Track Unique Values` checkbox for those two fields. *Track Unique Values* keeps a list of the unique values in a certain column. Search forms can use these to present selection lists based on the data in the Tabula to offer easier and more efficient searching. In Joe's case, he has four unique values for *REGION* spread across all of the records in his database, with the four values being `Northwest`, `Northeast`, `Southwest`, and `Southeast`. So when he generates his search form (a couple steps later), the generated form will have a list with those four values enabling a user to select any combination of those values using just their mouse. After setting these options, Joe clicks the "next>" button to go on to the *Report* screen.

On the Report screen, Joe can select which columns he wishes to display and what style of report to display them as. Reports are just Zope documents tailored to query and display batches of results from a Tabula. Once generated, they can be edited and customized like any other Zope document object. The report wizard just takes care of the more advanced DTML involved with reports (such as batch processing, where only a certain amount of records are displayed at a time with links to get the next or previous batch). A Tabula can have any number of reports. Like all Zope objects, reports must have an id and optionally a title. Joe sets the id to `report` and the title to `Widgets Sold`. He chooses all three columns for inclusion in the report and chooses `Tabular` for the report type, which displays the results in an HTML table. He could have chosen Record-oriented which displays results in a single paragraph separated by commas, but Tabular is more fitting to the type of data he's publishing. Having set his options for the report, Joe clicks `Next` to move on to the *Search Form* screen.


Phalonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Widgets
UnitsSold
sql_users
© Digital Creations
Refresh
Document: Done

Tabula Creation Assistant

Operation: schema

Operations: schema report search

On this page you can view and modify the **Data Schema** of your Tabula. Here you can set **Field** or **Text Indexes** to speed up search operations, and use the **Track Unique Values** option to create value lists for more precise searching and editing.

The indexes give the following features:
- **Field Index:** Speeds up access to this exact value. Useful in search forms and hierarchies.
- **Text Index:** Allows full-text searches of this field, including booleans and proximity. Useful in search forms.

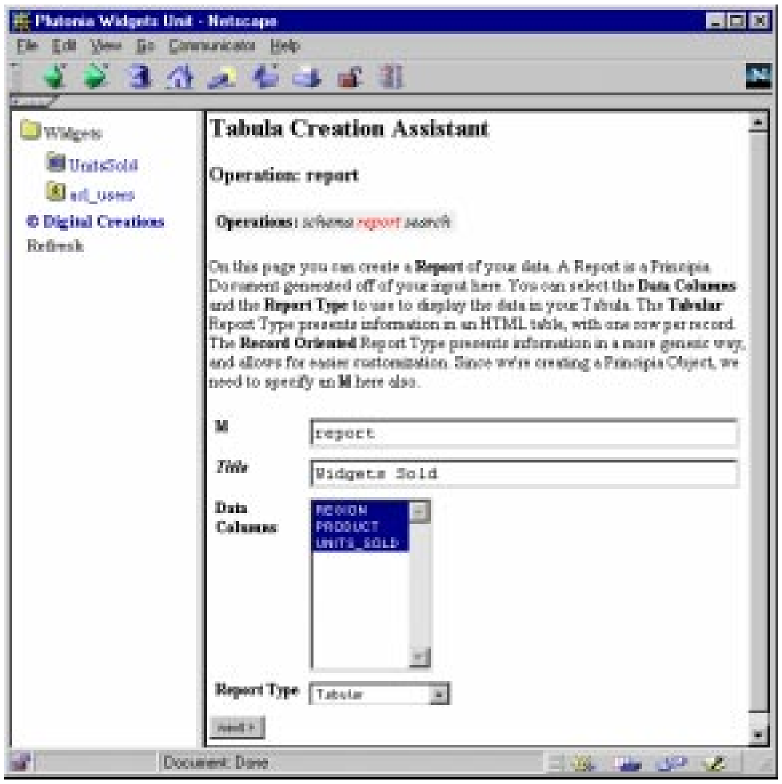
Column Name	Type	Input Width	Indexing	Track Unique Values
REGION	s	9	Field Index	✓
PRODUCT	s	23	Field Index	✓
UNITS_SOLD	i	3	No Index	☐

next >

Figure 5.

The Search Form screen is very similar to the Report screen. There's options here also for id and title. Since Tabula's act like Zope folders, a document with the id `index_html` is also known as the *public interface* to a folder. Just like a `default.html` or `index.html` file on normal Web servers. Joe decides that when users come to the `UnitsSold` Tabula, they'll be doing a search. So he gives the search form the id `index_html` and the title `Search Units Sold`. A search form in Tabula is also just a generated document. Here also is the list of columns again, but this time only two of the columns are selected (REGION and PRODUCT). This is because there are indexes set on those two columns. The search wizard assumes that searches are going to be done on these columns because they are indexed or have `Track Unique Values` setting and selects them automatically. The user is free to choose any column to be included in the search forms. Joe leaves the selections as they are since those are the fields he wants people to search on, knowing that since `track unique values` is set on both of those fields DTML for producing an HTML "select" list will be generated on each one. The last option on the screen is the report to use to display the results. Since there's only one (the one made in the previous step), Joe chooses `report` and clicks "Next" to go on to Step 5.

Step 5 is completion. The screen simply signifies that the Tabula Creation Assistant processes executed successfully, and provides some "Did you know?" information about Tabula's more advanced features. Since Joe is now finished with the creation process, He clicks "Next" and is directed back to the folder where he created the Tabula, which now shows up in the contents list.



Tabula Creation Assistant

Operation: report

Operations: *schema report search*

On this page you can create a **Report** of your data. A Report is a Principia Document generated off of your input here. You can select the **Data Columns** and the **Report Type** to use to display the data in your Tabula. The **Tabular** Report Type presents information in an HTML table, with one row per record. The **Record Oriented** Report Type presents information in a more generic way, and allows for easier customization. Since we're creating a Principia Object, we need to specify an **Id** here also.

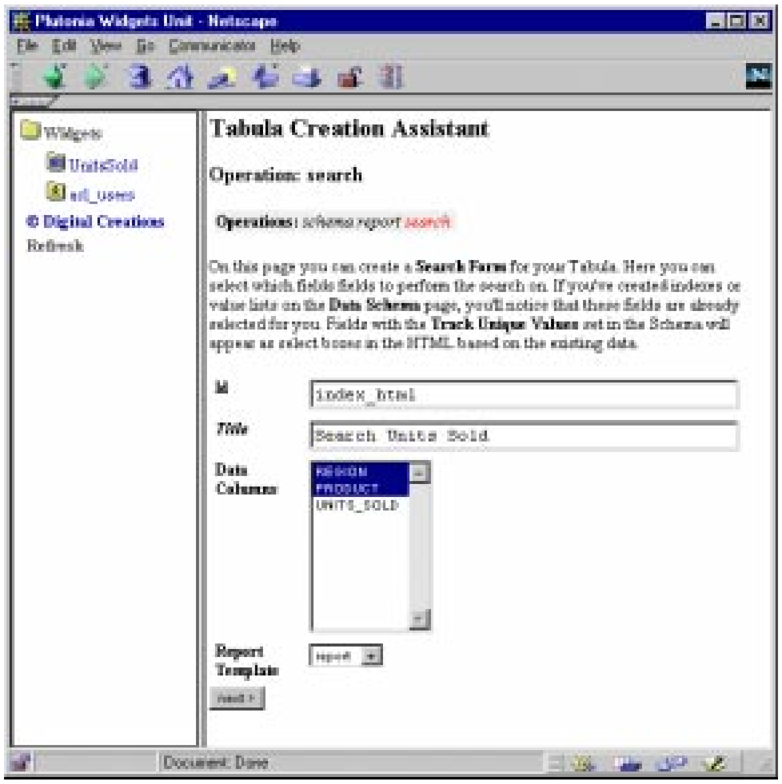
Id	report
Title	Widgets Sold
Data Columns	REGION PRODUCT UNITS_SOLD
Report Type	Tabular

Figure 6.

Using the Tabula

Since the TCA combined much of the initial work involved in setting up a working Tabula, the `UnitsSold` Tabula is already available for use from the end-users perspective. Joe gives it a trial run by going to the URL *.../Widgets/UnitsSold/* (where `...` is the URL preceding the Widgets folder). The search form made in the TCA process comes up.

Right away Joe notices the select lists are displaying all the right values for their respective fields. They're both a bit too big, but since it's a Zope document it can be edited easily. Joe selects the `northeast` and `southeast` regions, and the `Widget Bank Notes` and `Widget Construction Set` products and clicks the `search` button. A report comes up showing 4 results in an HTML table. Joe sees that they match the search criteria that he submitted.


Phatonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Widgets
UnitsSold
acl_users
© Digital Creations
Refresh
Document: Done

Tabula Creation Assistant

Operation: search

Operations: schema report search

On this page you can create a **Search Form** for your Tabula. Here you can select which fields fields to perform the search on. If you've created indexes or value lists on the **Data Schema** page, you'll notice that these fields are already selected for you. Fields with the **Track Unique Values** set in the Schema will appear as select boxes in the HTML based on the existing data.

Id	index_html
Title	Search Units Sold
Data Columns	REGION, PRODUCT (selected); UNITS_SOLD
Report Template	report

Next >

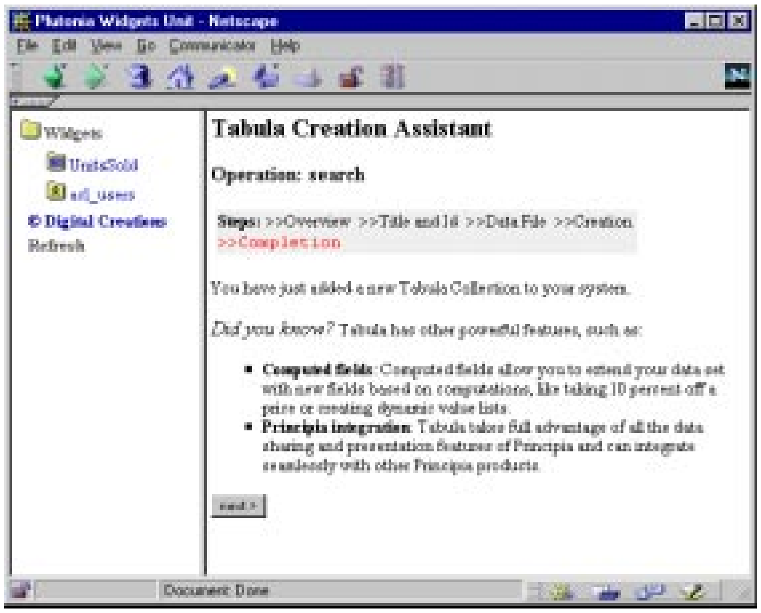
Figure 7.

It's not the prettiest of HTML, but his initial assignment was not to create anything fancy but to create something that works. The Tabula is fully usable at this point, and Joe didn't need to write *any* HTML code, *any* CGI code, or even spend extra time creating the initial needed objects (report and search form) outside of the creation process. In fact, the whole thing was done in a matter of minutes. Joe's bosses are happy and are ready to assign him his next task, which we'll get to in just a moment. First, let's review some tips that make not only the creation process of a Tabula flow smoothly, but make working with and extending the Tabula much easier.

Tabula Creation Tips

These are some points that help make the creation process of a Tabula flow smoothly and quickly. The TCA helps guide the creator through the process as easily as possible, but knowing what you want to do before hand helps this process even more.

- **Make sure the data is in the correct format** -- Almost all desktop database programs have an option to export to DBF, so this shouldn't be a problem.
- **Know what the main use for the Tabula will be** -- While there are many ways a single Tabula may be used and many combinations of searching and reporting, there is often a "main case." Knowing what the main case's requirements for searching and reporting are allow using the TCA to create a functioning Tabula right off the bat, as seen in the scenario above.



Tabula Creation Assistant

Operation: search

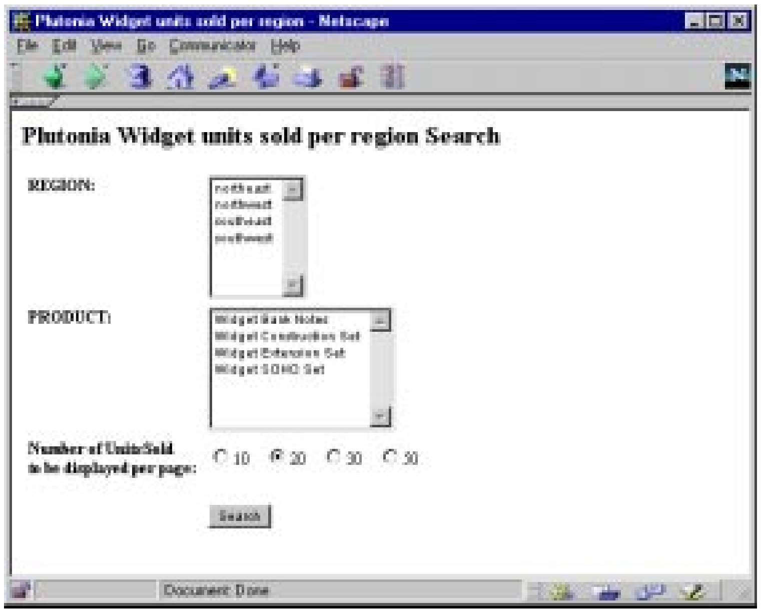
Steps: >>Overview >>Title and Id >>Data File >>Creation >>Completion

You have just added a new Tabula Collection to your system.

Did you know? Tabula has other powerful features, such as:

- **Computed fields**: Computed fields allow you to extend your data set with new fields based on computations, like taking 10 percent off a price or creating dynamic value lists.
- **Principia integration**: Tabula takes full advantage of all the data sharing and presentation features of Principia and can integrate seamlessly with other Principia products.

Figure 8.


Plutonia Widget units sold per region - Netscape
File Edit View Go Communicator Help
Search
Document: Done

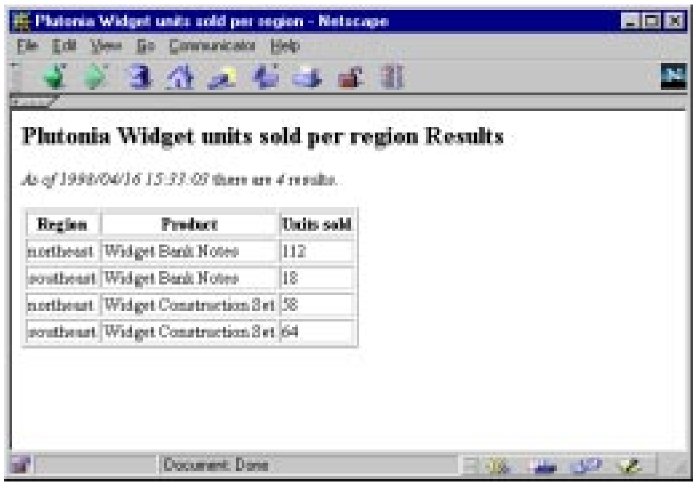
Plutonia Widget units sold per region Search

REGION: northeast, northwest, southeast, southwest

PRODUCT: Widget Bank Notes, Widget Construction Set, Widget Extension Set, Widget SOHO Set

Number of UnitsSold to be displayed per page: 10 [✓ 20] 30 30

Figure 9.


Plutonia Widget units sold per region - Netscape
File Edit View Go Communicator Help
Document: Done

Plutonia Widget units sold per region Results

As of 1998/04/16 15:33:03 there are 4 results.

Region	Product	Units sold
northeast	Widget Bank Notes	112
southeast	Widget Bank Notes	18
northeast	Widget Construction Set	58
southeast	Widget Construction Set	64

Figure 10.

More Advanced - Computed Fields

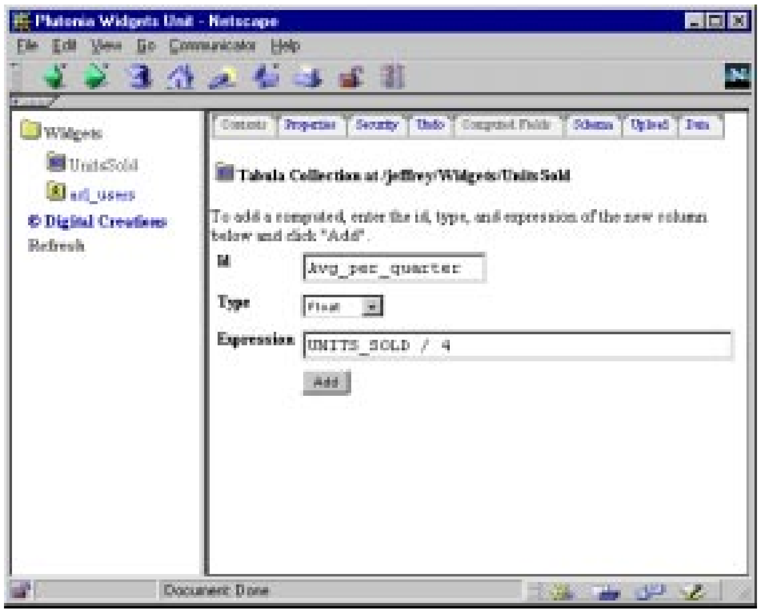
Let's return to Joe's assignment. Now that he has fulfilled his initial requirements, he is given some new ones. The bosses want the schema to be extended in the following manner:

- **Show the average units sold per quarter** -- Since the Widgets have only been on the market for a year, the number of units sold divided by four will suffice.
- **Break widgets into categories based on units sold** -- Products that have had more than 100 units sold in a particular region are considered `Hot Reception`, products that have had between 50 and 100 units sold are considered `Good Reception`, and products that have sold less than 50 are considered `Cold Reception`.

Joe also knows that there are some retail stores that haven't reported their final fourth quarter sales yet, so the new fields must be able to adapt to changes. Fortunately for Joe, Tabula has *Computed Fields*.

Since the `UnitsSold` Tabula is now actually being used, Joe needs to make his changes without disrupting the end users. So he needs to add a Zope Session object to the `UnitsSold` Tabula. A session allows Joe to work on the Tabula (or any Zope) in "private", so only he will be affected by the changes he makes to the system until he saves the changes. The end users of the `UnitsSold` Tabula won't see the new Computed Fields until Joe has implemented them and tested them to be satisfied that they work properly. To add a session, Joe goes to the contents screen of the `UnitsSold` Tabula, selects `Session` from the add list, and clicks "Add." He gives it the id `JoesSession` and clicks "Add." He then opens the session from the `UnitsSold` contents screen by clicking on it, and clicks the `Start working in session` button.

Adding Computed Fields can take a few steps. The main ones are creating and testing the new fields. Once they're added, they behave similarly to normal fields. They can be indexed, track unique values, and added to reports and search forms. The first thing Joe does is go to the *Computed Fields* tab of the `UnitsSold` Tabula. If there were any Computed Fields defined, they would be listed here. But since there are none, there is only a button to add new ones. Joe clicks "Add" and the *Add Computed Field* screen comes up.



Tabula Collection at /jeffrey/Widgets/UnitsSold

To add a computed, enter the id, type, and expression of the new column below and click "Add".

Id	Avg_per_quarter
Type	Float
Expression	UNITS_SOLD / 4

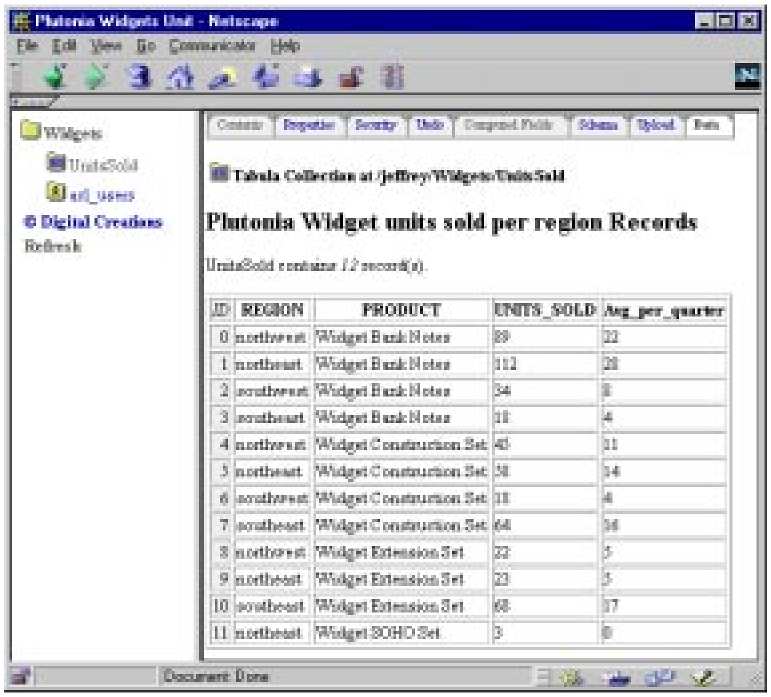
Add

Figure 11.

The first one Joe plans to add is for the average units sold per quarter, which he expects to be a floating point number. So he sets the field id to `Avg_per_quarter`, the type to `float`, and the expression to `UNITS_SOLD / 4`, and clicks "Add." The Computed Fields screen now comes back up with the new field listed. The Computed Fields list is similar to the Contents listing of any folderish object (but without icons). They can be edited by clicking on their id, and deleted by marking their checkbox and clicking the "Delete" button.

Now Joe wants to see the results of the new field. Tabula's *Data tab* is for such purposes. It gives a clean view of all the data in a Tabula, including computed fields. Joe clicks on the Data tab. He sees a table with the new *Avg_per_quarter* column.

But, all the results are integers! This is because the *UNITS_SOLD* column is an integer column, so Python computes the expression as an integer expression. So Joe needs to convert the *UNITS_SOLD* in the expression to a floating point number. He clicks on the Computed Fields tab again and clicks on the *Avg_per_quarter* id in the Computed Fields list. The edit screen for computed fields is similar to the Add screen.



Tabula Collection at /jeffrey/Widgets/UnitsSold

Plutonia Widget units sold per region Records

UnitsSold contains *12* record(s).

ID	REGION	PRODUCT	UNITS_SOLD	Avg_per_quarter
0	northwest	Widget Bank Notes	89	22
1	northeast	Widget Bank Notes	112	28
2	southwest	Widget Bank Notes	34	8
3	southeast	Widget Bank Notes	18	4
4	northwest	Widget Construction Set	45	11
5	northeast	Widget Construction Set	58	14
6	southwest	Widget Construction Set	18	4
7	southeast	Widget Construction Set	64	16
8	northwest	Widget Extension Set	22	5
9	northeast	Widget Extension Set	23	5
10	southeast	Widget Extension Set	68	17
11	northeast	Widget SOHO Set	3	0

Figure 12.

The computed fields expression language is a secured subset of the Python programming language. This enables modular namespaces to be opened to expressions to avoid name collisions between available functions and local variable names. Computed fields are detailed in the computed fields section of this document. For the purposes of this expression, Joe needs *UNITS_SOLD* to be a floating point number instead of an integer. Python has a *float()* function that does this. It's stored in the _ *namespace*, meaning it's accessed with *_.float()*. Joe changes the expression to read `_.float(UNITS_SOLD) / 4` and clicks "Change." He clicks on the Data tab again. The *Avg_per_quarter* column now shows his expected results.

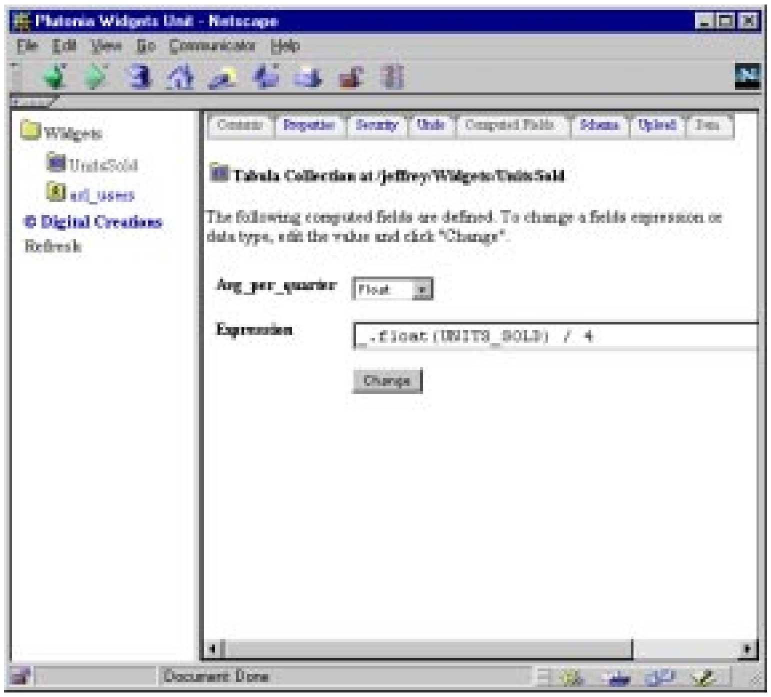
The next computed field he needs to add is a little bit more complex. He needs to show how the publics reception to each product per region has been, based on the number of units sold. The *_.test()* function will do this, offering *`if...else if...else`* functionality in a single function. The syntax of *_.test()* is:

```
_.test(test1, result1, test2, result2,..., default_result)
```

where *default_result* is returned if all tests fail. Joe goes back to the Computed Fields tab and clicks the "Add" button to add a new computed field. On the add screen he gives it the id `Reception`, sets the type to `String`, and sets the expression to:

```
_.test(UNITS_SOLD = 100, 'Hot', UNITS_SOLD = 50, 'Good', 'Cold')
```

The *test* function goes through the tests in order, first checking to see if the units sold are greater than or equal to 100 and returning the value `Hot` if true, then checking if the units sold is greater than or equal to 50 and returning the value `Good` if true, and finally returning `Cold` if the first tests fail. After Joe clicks "Add", he goes to the Data tab again to check his results. They all appear accurate.


Phalonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Contents Properties Security Undo Computed Fields Schema Upload Test
Widgets
UnitsSold
acl_users
© Digital Creations
Refresh
Change
Document: Done

Tabula Collection at /jeffrey/Widgets/UnitsSold

The following computed fields are defined. To change a fields expression or data type, edit the value and click "Change".

Avg_per_quarter Float

Expression `_.float(UNITS_SOLD) / 4`

Figure 13.

Now for the tricky part: merging these new fields into the report and search form he made in the creation process. Joe has a couple of options here: he can edit the DTML himself, or he can regenerate the documents. Since he hasn't done any editing of the documents yet, he can regenerate them without losing anything. And since he's in a session, it won't affect anyone currently using the Tabula.

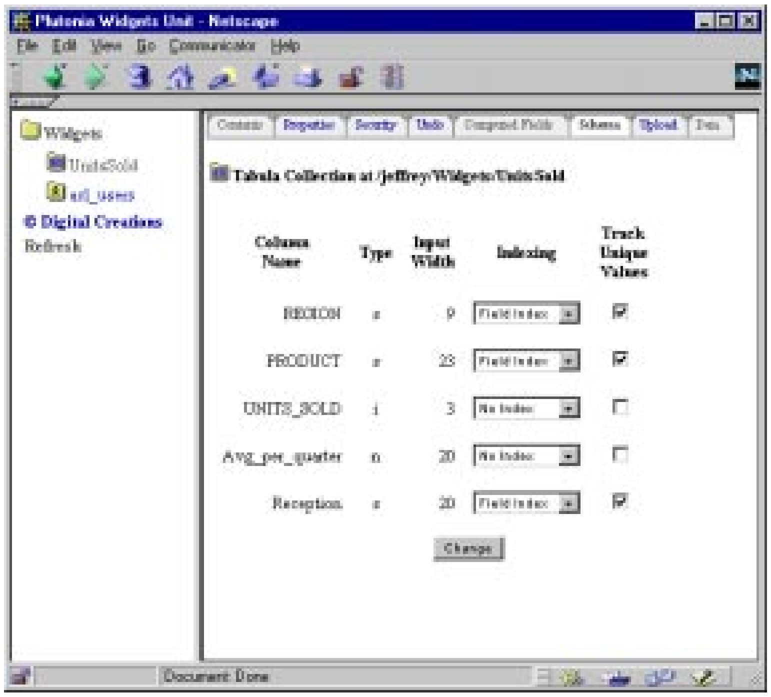
Thinking that users will want to search on the Reception value of the widgets, he decides first to create an index and track unique values on the `Reception` field. He clicks on the Schema tab.

The Schema screen is very similar to the one in the Tabula Creation Assistant, with the same setting options for modifying the settings for each column. The newly added computed fields are now listed as well as the original columns. He sets the index type of *Reception* to `FieldIndex` and enables `Track Unique Values` for that field and clicks "Change."

Then on the contents list of the UnitsSold Tabula, Joe selects the `index_html` and `report` documents by clicking their respective checkboxes and clicking "Delete." Now he adds a new report by choosing `Report` from the add list and clicking "Add."

The Ad*d Report screen* here is again very similar to the one in the TCA. Joe gives it the id `report` and the title `Units Sold Report`. He then makes sure all columns are selected for the report, chooses Tabular for the report type and clicks "Add"

Returning to the contents screen, he chooses `Search Form` from the Add List and clicks "Add."


Plutonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Widgets
UnitsSold
acl_users
© Digital Creations
Refresh
Contents Properties Security Undo Computed Fields Schema Upload Data
Change
Document: Done

Tabula Collection at /jeffrey/Widgets/UnitsSold

Column Name	Type	Input Width	Indexing	Track Unique Values
REGION	s	9	Field Index	✓
PRODUCT	s	23	Field Index	✓
UNITS_SOLD	i	3	No Index	☐
Avg_per_quarter	n	20	No Index	☐
Reception	s	20	Field Index	✓

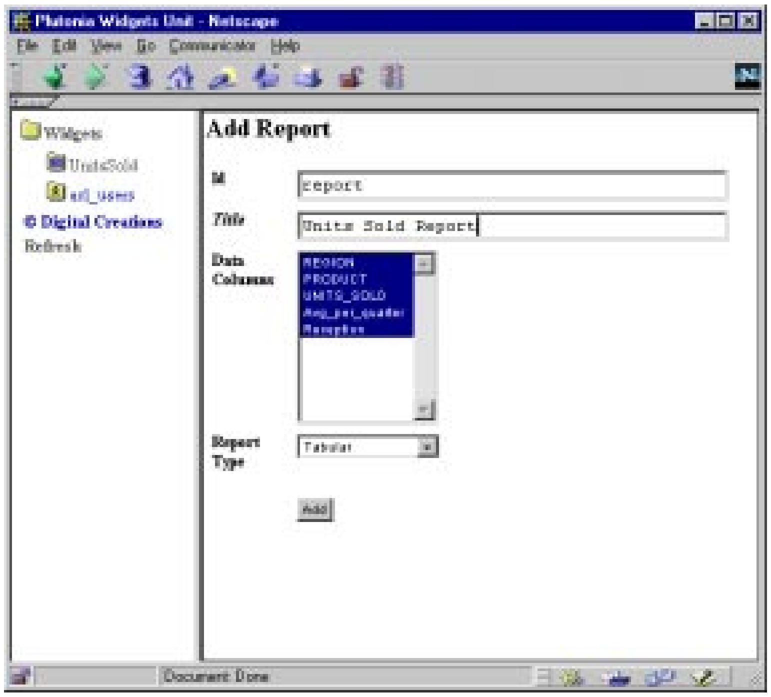
Figure 14.

The Add Search Form screen here is very similar to the one in the TCA. Hoe gives it the id `index_html` and the title `Units Sold Search Form`. He makes sure that the `REGION`, `PRODUCT`, and `Reception` fields are selected and that the report to use is set to `report`. He then clicks "Add."

In a new browser window, he goes back to the URL *.../Widgets/UnitsSold/* and the new search form comes up. He does a search for `Good` reception and the new report with the added computed fields comes up. Joe sees that these results are correct. Feeling pleased with himself, he saves the session by opening it and clicking on the "Save" button after entering some text explaining that the new fields were added, and quits the session. Task 2 accomplished!

New Data

Some stores have reported in new totals and Joe needs to update the on-line data. With Tabula, this is basically a one step operation. After exporting the updated data from his database to a DBF file again, Joe clicks on the `Upload` tab of the `UnitsSold` Tabula.



Add Report

Id: report

Title: Units Sold Report

Data Columns: REGION, PRODUCT, UNITS_SOLD, Avg_per_quarter, Reception

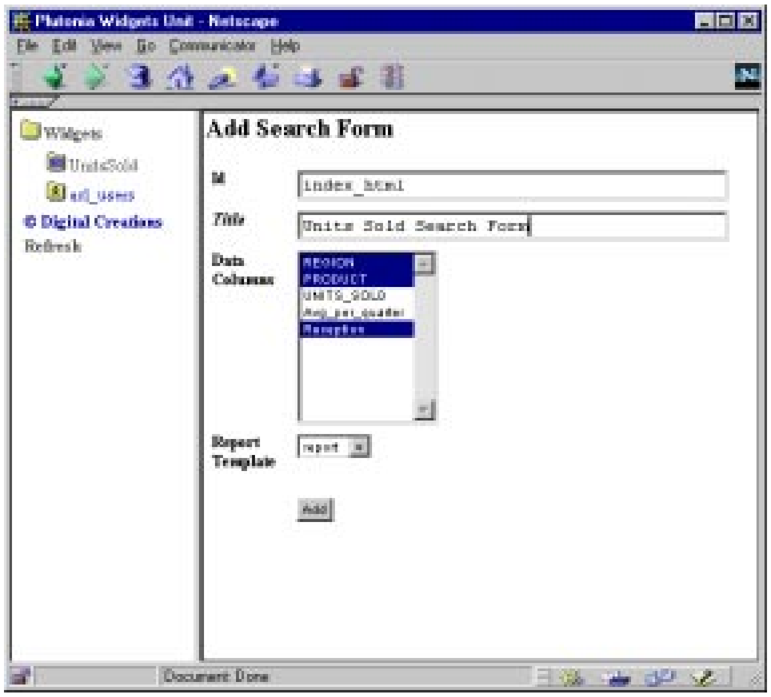
Report Type: Tabular

Figure 15.

The first two fields, **Data File Format** and **Data File** are like their TCA counterparts, except, in this case, they are both in the same step. Tabula remembers the format of the most recently uploaded data file and automatically selects it, but it can be changed. *(Note, if you change data file types, the schema* ***must*** remain the same!). Joe uses the "Browse..." button to select the *UnitsSold.dbf* file from his hard disk for the Data File. The three remaining options and Joe's actions on them are:

- **Remove existing data** -- If checked, the existing data is removed before the new data is added. Joe checks the box for this option since he is overwriting the old records with new ones. If he were to leave it unchecked, the new data would be added to the old, which is undesirable in this situation.
- **Allow columns in the upload file that are not in the existing data** -- If the database Joe had exported contained extra columns that weren't in the original schema used for the UnitsSold Tabula, an error would be raised if this box wasn't checked. The exported schema is the same however, so Joe leaves the box unchecked.
- **Allow columns to be omitted from the upload file that are in the existing data** -- If the database Joe had exported had a column deleted from the data set, an error would be raised if this box wasn't checked. Again, the exported schema is the same, so Joe leaves this box unchecked.

Joe clicks the "upload" button. When the data has been uploaded and placed in the Tabula, a message indicating the upload was successful comes up. Joe clicks "OK" and returns to the contents list. Without having to modify any documents or any other objects, he just updated the data in the Tabula and it's immediately seen by anyone using the system. And it isn't even his lunch break yet.



Add Search Form

Id	index_html
Title	Units Sold Search Form
Data Columns	REGION, PRODUCT (selected); UNITS_SOLD; Avg_per_quadar; Reception (selected)
Report Template	report

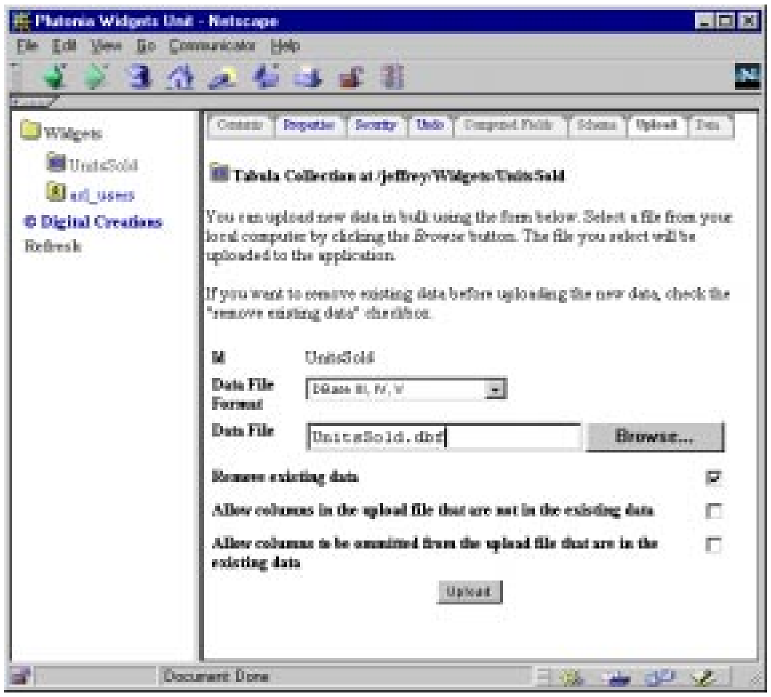
Figure 16.

Even More Advanced - Hierarchies and Acquisition

Now that the basic interface has been designed, Joe would like to do some things to make the Tabula even more usable and present some different views and ways of accessing the data. Chiefly, he wants some documents that give expanding / contracting views of the data, and maybe the ability to differentiate certain expansions of those sub views without conflicting with others. And it would be nice not to have to do a lot of copying and pasting of documents around. And it would be really nice for these structures to stay up to date with new data as it comes in. Enter Hierarchies.

Hierarchies are a way of turning flat tabular data into more interesting hierarchical data, based on values of columns. They can be accessed directly in a URL or they can use the DTML `Tree` tag to make documents that can expand and collapse different levels of a hierarchy. Adding a hierarchy is a one step process. It creates and manages its own sub-hierarchies. Hierarchies and their contained sub-hierarchies all act like folders. Documents and other Zope objects can be added at any sub level in a hierarchy. Suddenly, that flat tabular data set comes alive!

The first Hierarchy Joe wants to make is one based solely on public reception, as computed by the `Reception` field. Since he's going to be making some new changes to the site, he enters back into the session `JoesSession` before starting. Then, in the `UnitsSold` Tabula he selects `Hierarchy` from the add list and clicks "Add."



Tabula Collection at /jeffrey/Widgets/UnitsSold

You can upload new data in bulk using the form below. Select a file from your local computer by clicking the *Browse* button. The file you select will be uploaded to the application.

If you want to remove existing data before uploading the new data, check the "remove existing data" checkbox.

Id	UnitsSold
Data File Format	DBase III, IV, V
Data File	UnitsSold.dbf
Remove existing data	✓
Allow columns in the upload file that are not in the existing data	☐
Allow columns to be ommitted from the upload file that are in the existing data	☐

Figure 17.

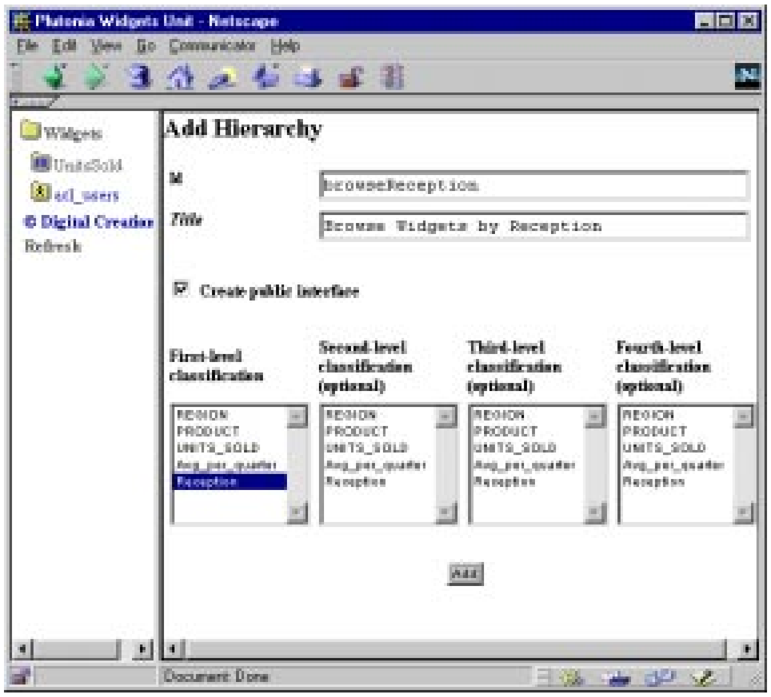
The *Add Hierarchy screen* has the following fields.

- **Id** -- Zope id for the hierarchy. Joe sets "Id" to `browseReception`.
- **Title** -- *optional* Title attribute for the hierarchy. Joe sets "Title" to `Browse Widgets by Reception`.
- **Create Public Interface** -- If checked, an `index_html` document will be created in the top level of the hierarchy containing a simple Tree Tag. Joe checks this to help cut down on some of the initial work.
- **XXXX Level Classification** -- A hierarchy can have up to four levels of classification, that is, four levels of sub-hierarchies. Every hierarchy needs at least one, the rest are optional. Since Joe's only making this one on the `Reception` field, he selects `Reception` for the Level 1 classification and clicks "Add."

After adding the Hierarchy, the Navigation Frame gets updated (since the Navigation Frame shows a tree of folderish objects, and Hierarchies are folderish). Joe expands the `UnitsSold` Tabula by clicking its expand icon (represented by a + symbol) and sees the `browseReception` Hierarchy. He expands that one too and sees three more sub-hierarchies relating to the values of the Reception field.

Since he selected "Create public interface" when he created the Hierarchy, he decides to look at it. In a new window he goes to the URL

.../Widgets/UnitsSold/browseReception/



Add Hierarchy

Id: browseReception

Title: Browse Widgets by Reception

[✓] Create public interface

First-level classification	Second-level classification (optional)	Third-level classification (optional)	Fourth-level classification (optional)
REGION PRODUCT UNITS_SOLD Avg_per_quarter [✓ Reception]	REGION PRODUCT UNITS_SOLD Avg_per_quarter Reception	REGION PRODUCT UNITS_SOLD Avg_per_quarter Reception	REGION PRODUCT UNITS_SOLD Avg_per_quarter Reception

Figure 18.

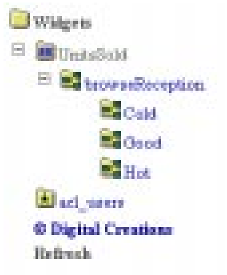


Figure 19.

Bummer. All he sees are the values Cold, Good, and Hot. Where was all of this exciting new presentation of information? Since it’s only a single level tree (only on the field ‘Reception’), It only has one “branch”. The data in each of these sub-hierarchies are called “leaves”, since it’s the last level of classification of the hierarchy. Joe needs to show the data that’s in the leaves. Thanks to options of the tree tag, this is actually rather easy.

Joe has a couple of options here. Trees can expand leaves with the addition of a single parameter to the tree tag, so he could have the tree open a report at each level using the same "+" and "-" icons in the Zope Navigation Frame. Another option would be to jump to that sub-hierarchy directly and display a report. Or he could even do both. This is a small dataset, so he decides to go with the “leaves” option.

All he has to do is edit the `index_html` document in the `browseReception` hierarchy. He goes to the `contents` tab of the `browseReception` hierarchy and clicks on the `index_html` document to edit it. The document's source is:

```
<!--#var standard_html_header-->

<!--#tree-->
 <!--#var title-->
<!--#/tree-->

<!--#var standard_html_footer-->
```

(Note: for full documentation on the tree tag, see the DTML Manual). Joe wants the leaves to expand and show a report. He's already defined a report in the Tabula and would like to reuse that here. The problem is, it's in the `UnitsSold` Tabula folder, not the `browseReception` folder. This is where *acquisition* steps in. Acquisition allows Zope objects to acquire attributes and objects from their parents, who acquire from their parents, and so on. So Joe can use the report defined in the `UnitsSold` Tabula folder as if it were in the `browseReception` folder. Using acquisition this way opens up some powerful possibilities that he's going to get to in a moment. Right now, changes the `#tree` tag opener to read:

```
<!--#tree leaves=report-->
```

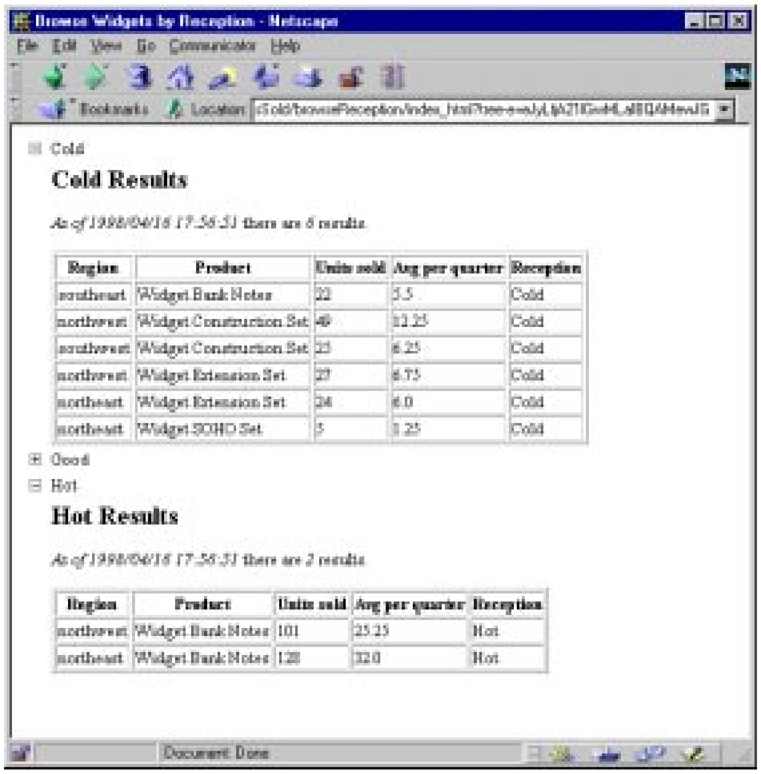
Joe clicks change, goes to the window where he was looking at the tree itself and reloads the page. Now each of the lines has a tree expand (+) icon in front of it. Joe clicks on the expand icon for `Cold` and sees a report come up on the page under the `Cold` entry showing only the widgets designated as Cold. He clicks the expand icon for `Hot` and underneath the Hot icon is a report showing only the hot sellers.

In essence, this part of what he wanted to do is accomplished. This use of hierarchies and the tree tag gives an interesting alternate view of the Data. He can make many different hierarchies to form different views (grouped by Region or Product for example). But what is really going on here? What are some of the real possibilities of Hierarchies?

First, to address what is really going on here. When a hierarchy is created, its sublevels become folderish objects. Each of these sublevels is like a mini-Tabula in that reports and search forms can be added and executed in each sublevel. Because of acquisition, however, reports and search forms don't have to be created for each sublevel. But the cool trick is that each sublevel *can* be made unique! If Joe wants, he can make a new report in the `Hot` sub-hierarchy display its text in red. The `Good` and `Cold` levels would be unaffected by this. Without going into document editing and HTML specifics, Joe decides to make the Reception column in the `Hot` sub-hierarchy appear in red. One way of doing this is to do a Zope copy and paste operation. Joe goes to the contents screen of the `UnitsSold` Tabula, selects the `report` document by marking its checkbox in the contents list, and clicking the `Copy` button. He then goes to the contents screen of the Hot sub-hierarchy and clicks the "Paste" button, and giving the pasted object the id of `report`. He then edits the reports source code to color the `Reception` text red, clicks the `Change` button, and reloads the page where he was looking at the tree. It now looks like:

Only the `Hot` leaf displays the Reception column in red. The `Cold` leaf remains unaffected. Here is what's happening:

1. The URL *.../Widgets/UnitsSold/browseReception/* is called. Zope calls the index_html document which contains the tree tag, and sends information to the tree tag to expand the `Cold` and `Hot` branches into leaves.
2. When the tree tag visits any object, it calls that object directly. The main source inside the tree tag simply has a `<!--#var title-->` statement, but that statement actually gets the `title` attribute or method of the object. When expanding a leaf, it uses the parameter passed to the tree tags 'leaves' option to call that method (usually a Document) on the object and display its results. So when Cold is expanded, it looks for the `report` document in Cold. Cold


Browse Widgets by Reception - Netscape
File Edit View Go Communicator Help
Bookmarks Location
Document: Done

Cold

Cold Results

As of 1998/04/16 17:58:51 there are 6 results.

Region	Product	Units sold	Avg per quarter	Reception
southeast	Widget Bank Notes	22	5.5	Cold
northwest	Widget Construction Set	49	12.25	Cold
southwest	Widget Construction Set	25	6.25	Cold
northwest	Widget Extension Set	27	6.75	Cold
northeast	Widget Extension Set	24	6.0	Cold
northeast	Widget SOHO Set	5	1.25	Cold

Good

Hot

Hot Results

As of 1998/04/16 17:58:51 there are 2 results.

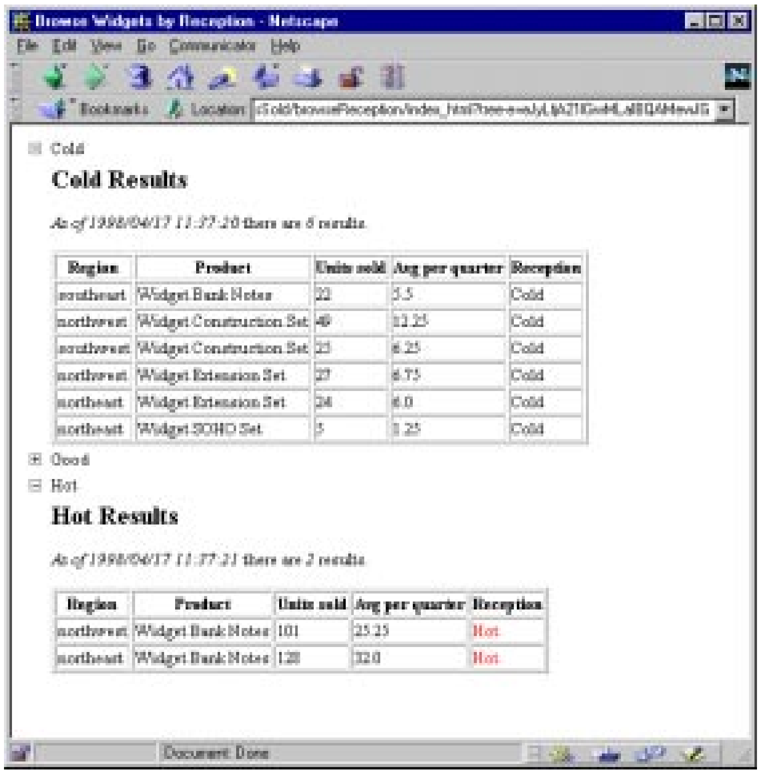
Region	Product	Units sold	Avg per quarter	Reception
northwest	Widget Bank Notes	101	25.25	Hot
northeast	Widget Bank Notes	128	32.0	Hot

Figure 20.

doesn't have a `report` document, so acquisition kicks in. The nearest appearance of report is in the `UnitsSold` Tabula. Cold acquires `report` from `UnitsSold` and it gets rendered as if a normal member of the Cold object.

3. When the tree tag expands the leaf on Hot, it does encounter a `report` document as a member of Hot. So it renders Hot's report document instead of acquiring it from above. Since the `report` document in Hot has some extra HTML to make the Reception column appear red, it displays as red in Joe's Web browser.

But what if Joe were to upload new data that didn't have any Hot sellers. That level of the Hierarchy would disappear. Would his customizations disappear too? No. Tabula keeps track of sub-hierarchies even when there might not be data supporting that level. When new data supporting an empty level comes back into the system, that sub-hierarchy and all of its customizations will re-appear. Hierarchies really make Tabula and Zope stand out by enabling traditionally "flat" data to become hierarchical, with all levels of a hierarchy able to take on their own appearance and behavior.


Browse Widgets by Reception - Netscape
File Edit View Go Communicator Help
Bookmarks
Location
Document: Done

Cold

Cold Results

As of 1998/04/17 11:37:20 there are 6 results.

Region	Product	Units sold	Avg per quarter	Reception
southeast	Widget Bank Notes	22	5.5	Cold
northwest	Widget Construction Set	49	12.25	Cold
southwest	Widget Construction Set	25	6.25	Cold
northwest	Widget Extension Set	27	6.75	Cold
northeast	Widget Extension Set	24	6.0	Cold
northeast	Widget SOHO Set	5	1.25	Cold

Good

Hot

Hot Results

As of 1998/04/17 11:37:21 there are 2 results.

Region	Product	Units sold	Avg per quarter	Reception
northwest	Widget Bank Notes	101	25.25	Hot
northeast	Widget Bank Notes	128	32.0	Hot

Figure 21.

The Tabula Object

The basis of all Tabula operations comes from the Tabula Object, which in many ways acts like a Zope Folder object. Tabula extends the Folder object by containing tabular-based data that can be searched and viewed in many ways. Tabulas can contain any Zope object that a folder can, along with some Tabula specific objects such as Hierarchies.

Creating using the Creation Assistant

Creating a working Tabula can require more than just creating the new object and uploading some data. Setting indexing options, creating a report, and generating a search interface are all common steps. The Tabula Creation Assistant allows any combination of these steps to be done at creation time, streamlining some common tasks. There are five main steps and three optional substeps. To get started, choose *Tabula Collection* from the Add List on the contents screen of the folder to add the Tabula to.

Step 1 - Overview

This step is a quick introduction to the Tabula Creation Assistant and the steps involved in creation. Click on the “Next” button to move onto step 2.

Step 2 - Title and Id

The first step in creating any Zope object is specifying its ID. The ID, which must be unique in its Folder, is what is used to access the object and its contents. The ID must not contain spaces and should start with a letter. The title is optional (indicated by the word Title being in italics). The title is used to give a more descriptive name to the object. These are the same rules that govern all Zope objects.

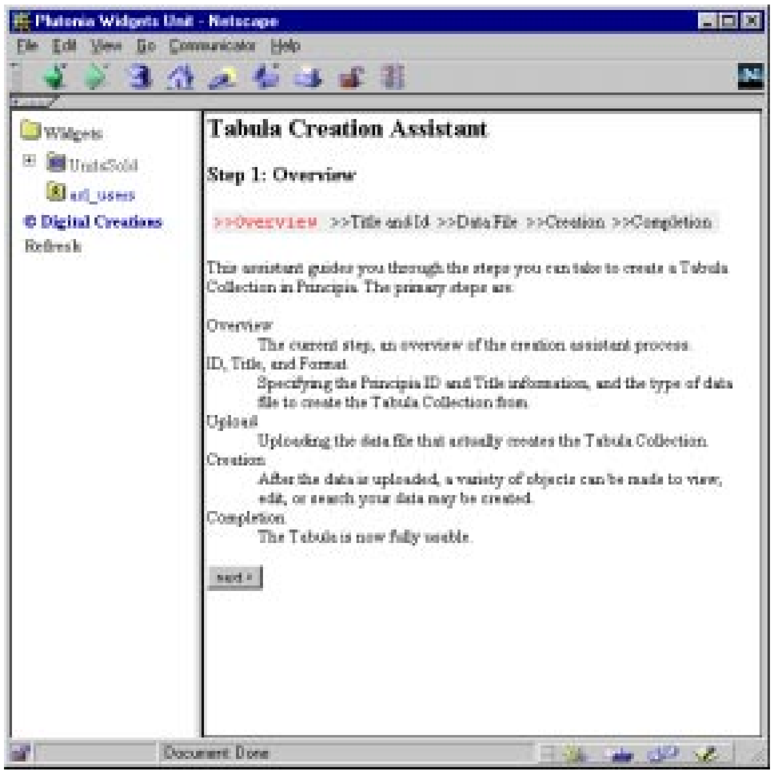
The other piece of primary concern when creating a Tabula Object is specifying the format of the data file to be uploaded. Tabula currently supports Rand RDB, DBF (used by dBase, Foxpro, and other databases), and DBF with Memo. Use the `Data File Format` drop-down list to select the type of file the data is being supplied from. Click the `Next` button to move onto step 3.

Step 3 - Data File

The next step in creating a Tabula is supplying it with data. In step 3, select a file that matches the type specified in Step 2 (Step 3 reprints the specified file type) in the field specified. If the `DBF With Memo` type was chosen, a second field for uploading the Memo file is given. Click the `Next` button to move onto step 4.

Step 4 - Creation

Once the data is uploaded, the Tabula is created. The Zope Navigation frame is updated to include the new Tabula (since Tabula’s are "folderish"). Note that it is possible to end the creation process and start working directly in the new Tabula at this point without disrupting anything, but the Step 4 screen presents three operations that may be performed on the new Tabula. Selecting none of the operations and clicking `Next` will go directly to Step 5.



Tabula Creation Assistant

Step 1: Overview

>>Overview >>Title and Id >>Data File >>Creation >>Completion

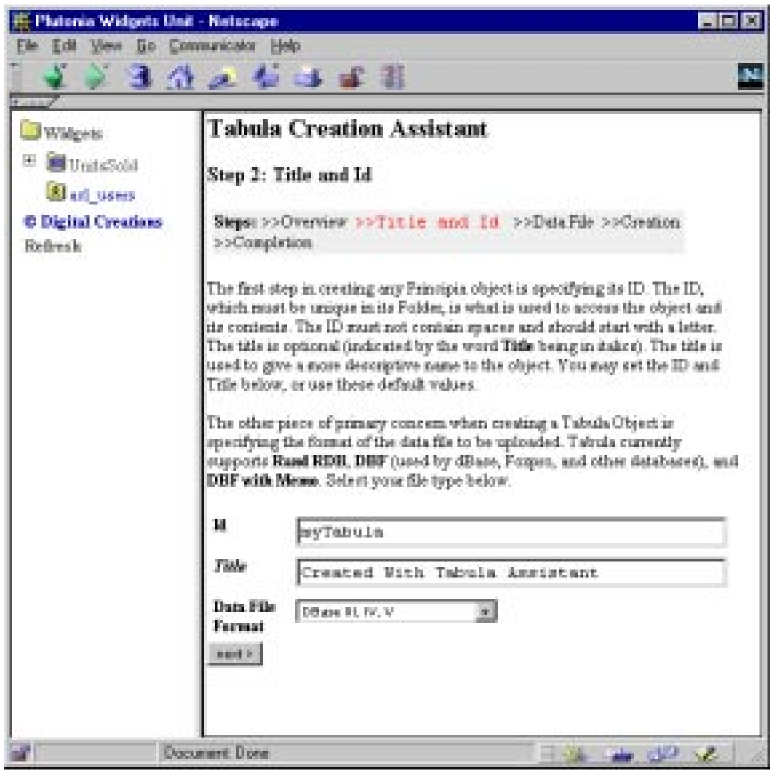
This assistant guides you through the steps you can take to create a Tabula Collection in Principia. The primary steps are:

Overview
The current step, an overview of the creation assistant process.
ID, Title, and Format
Specifying the Principia ID and Title information, and the type of data file to create the Tabula Collection from.
Upload
Uploading the data file that actually creates the Tabula Collection.
Creation
After the data is uploaded, a variety of objects can be made to view, edit, or search your data may be created.
Completion
The Tabula is now fully usable.

Figure 22. Step 1 - Overview

Selecting any combination of the operations presented by checking their appropriate checkboxes and clicking `Next` will go to the first option selected. The operations are covered in more detail in their respective sections. Briefly, they are:

- **Set up indexes and value lists to speed up search operations** -- Indexes speed up search operations and value lists offer a way of tracking unique values in a specific field for more precise searching. Field indexes speed up accesses to exact values, Text indexes allow full text searches on their fields including boolean and proximity searches.
- **Create a report to display the data in your tabula** -- This option is highly recommended and automatically marked. A report is required if a search interface is to be generated. A Tabula report is a Zope Document object generated from the input specified at report creation time. Like all Zope objects, there is the required Id field and the optional Title field. There is also a select list with all the field names in the Tabula. All of the selected fields will be included in the report. Lastly, there is a drop-down list for the report type to use. Tabular generates a report using an HTML table; Record-oriented generates a report with commas separating all of the selected fields. Since reports are Zope documents, they can be edited and customized in any fashion.
- **Create a search form for your tabula** -- In a similar fashion to reports, creating a search form means generating a Zope document that gets input from a user and uses a report to display the results of the search. There are fields to specify the Id and Title, and a list of fields to search on. If value lists or indexes have been made on any fields, those fields are automatically highlighted. When a search form is generated with fields marked with `Track Unique Values`, instead of generating a normal text input field it will generate a select list with all the unique values of that field. Lastly, there's a drop-down list for specifying which report to use to display the results. Since this is a brand new Tabula and assuming a report has been generated, there should only be one option at this time.


Platonia Widgets Unit - Netscape

Tabula Creation Assistant

Step 2: Title and Id

Steps: >>Overview >>Title and Id >>Data File >>Creation >>Completion

The first step in creating any Principia object is specifying its ID. The ID, which must be unique in its Folder, is what is used to access the object and its contents. The ID must not contain spaces and should start with a letter. The title is optional (indicated by the word **Title** being in italics). The title is used to give a more descriptive name to the object. You may set the ID and Title below, or use these default values.

The other piece of primary concern when creating a Tabula Object is specifying the format of the data file to be uploaded. Tabula currently supports **Rand RDB**, **DBF** (used by dBase, Foxpro, and other databases), and **DBF with Memo**. Select your file type below.

Id	myTabula
Title	Created With Tabula Assistant
Data File Format	DBase III, IV, V

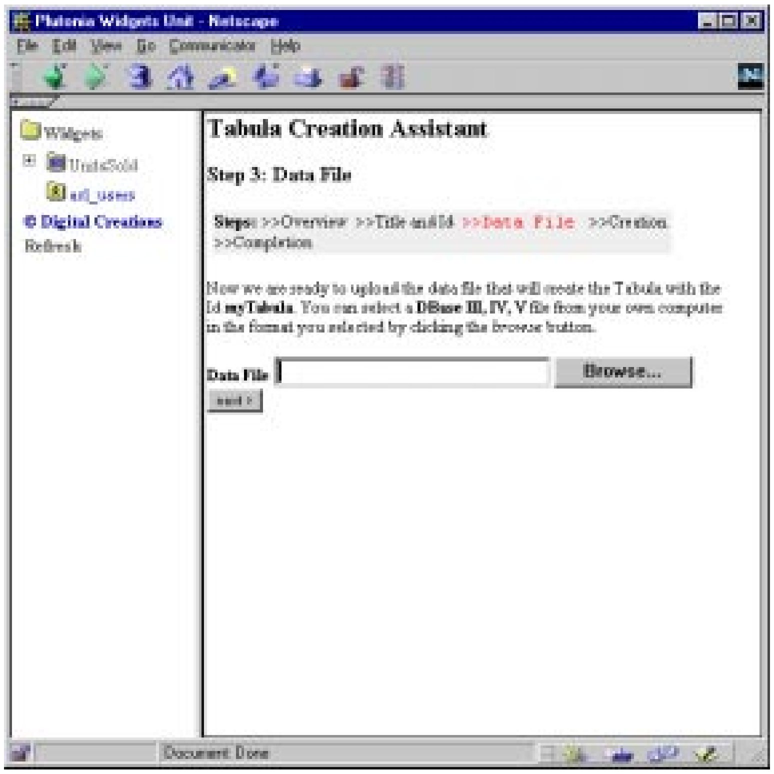
Figure 23. Step 2 - Title and Id

Step 5 - Completion

This screen signifies completion of the creation process and that all went well. It also has some *Did you know?* information about the more advanced features of Tabula. Clicking `Next` returns to the contents screen of the folder where the Tabula was created.

Working With Tabula

Working with Tabula is like working with any other Zope object. In the Workspace frame there is a set of “tabs” across the top for working with different views or aspects of an object. Since Tabula’s extend Zope Folder objects, the first four tabs (`Contents`, `Properties`, `Security`, `Undo`) behave exactly like their Folder counterparts. Tabulas have four more tabs specific to working with Tabula objects. Those tabs are `Computed Fields`, `Schema`, `Upload`, and `Data`.



Tabula Creation Assistant

Step 3: Data File

Steps: >>Overview >>Title and Id >>Data File >>Creation >>Completion

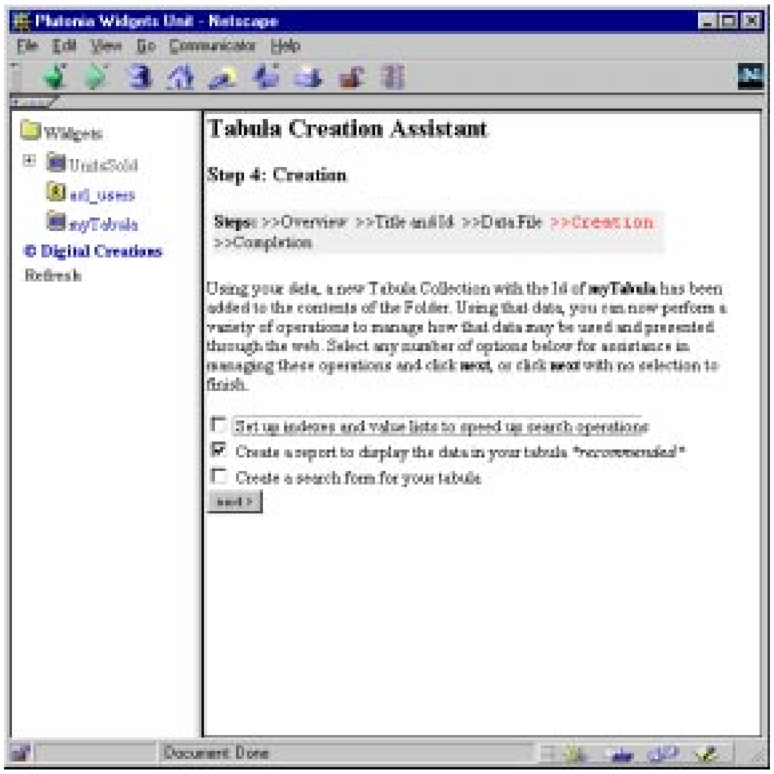
Now we are ready to upload the data file that will create the Tabula with the Id **myTabula**. You can select a **DBase III, IV, V** file from your own computer in the format you selected by clicking the browse button.

Data File

Figure 24. Step 3 - Data File

Tabula also has some special properties that can effect searching and reporting. These can be set on the Properties tab. (For information on adding and editing Properties, see the Folder section of the Zope Managers Guide). These properties can also be parameters passed to reports. Setting these as properties sets a "default" behavior. These properties are:

- **batch_size** -- An integer specifying how many records to display in a report. The Tabula Report Generator (see below) generates DTML code that uses this property to determine the default batch size for a report. If not set, the default number shown is 20.
- **sort-on** -- Currently, Tabula lets you sort on only one column at a time. The value of this property should be the id of a column with a **FieldIndex** (see "Modifying the Schema").
- sort-order -- If either 'reverse' or 'descending', reverses the sort order performed on the 'sort-on' field.



Tabula Creation Assistant

Step 4: Creation

Steps: >>Overview >>Title and Id >>Data File >>Creation >>Completion

Using your data, a new Tabula Collection with the Id of **myTabula** has been added to the contents of the Folder. Using that data, you can now perform a variety of operations to manage how that data may be used and presented through the web. Select any number of options below for assistance in managing these operations and click **next**, or click **next** with no selection to finish.

- [ ] Set up indexes and value lists to speed up search operations
- [✓] Create a report to display the data in your tabula *recommended*
- [ ] Create a search form for your tabula

next >

Figure 25. Step 4 - Creation

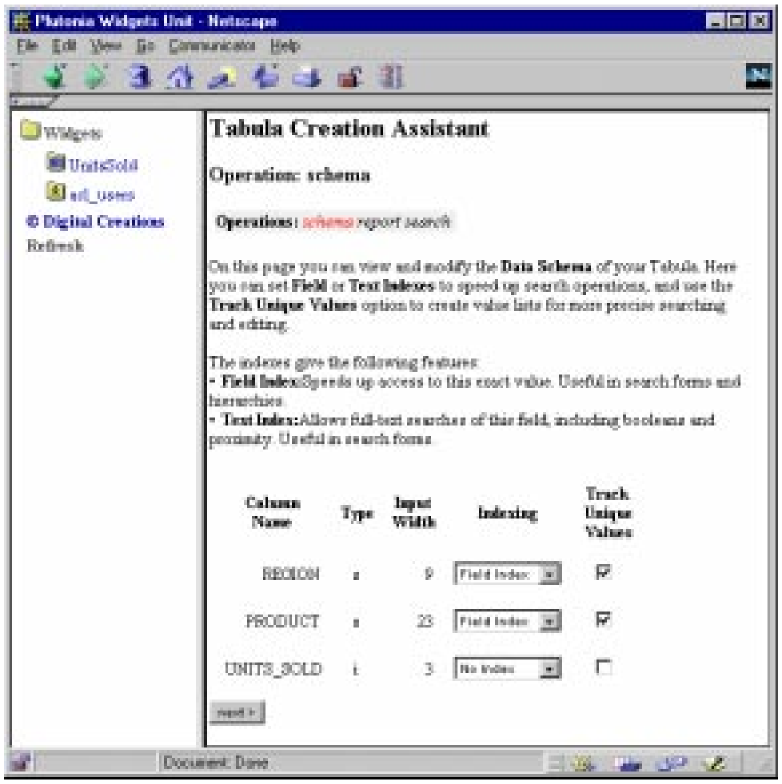
Working with Computed Fields

Overview

Computed Fields are one of the more advanced features of Tabula. They allow the schema to be extended with new dynamically calculated fields which can be indexed and treated like normal. The use the same machinery as the DTML *expr* machinery, with the most local namespace being each record. For example, using a dataset with the fields `Price` and `Discount` (both being floating-point fields in this example), a computed field `FinalPrice` might look like `Price-Discount`. Furthermore, a computed field `FinalPriceWithTax` could use the FinalPrice field to add the tax. `FinalPrice+(0.065*FinalPrice)` would work.

Adding Computed Fields

To add a new computed field, go to the `Computed Fields` tab. If there are any computed fields already defined, they will be listed followed by buttons for performing operations. To add a new computed field, click on the `Add...` button.



Platonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Widgets
UnitsSold
acl_users
© Digital Creations
Refresh
Document: Done

Tabula Creation Assistant

Operation: schema

Operations: schema report search

On this page you can view and modify the **Data Schema** of your Tabula. Here you can set **Field** or **Text Indexes** to speed up search operations, and use the **Track Unique Values** option to create value lists for more precise searching and editing.

The indexes give the following features:
- **Field Index:**Speeds up access to this exact value. Useful in search forms and hierarchies.
- **Text Index:**Allows full-text searches of this field, including booleans and proximity. Useful in search forms.

Column Name	Type	Input Width	Indexing	Track Unique Values
REGION	s	9	Field Index	✓
PRODUCT	s	23	Field Index	✓
UNITS_SOLD	i	3	No Index	☐

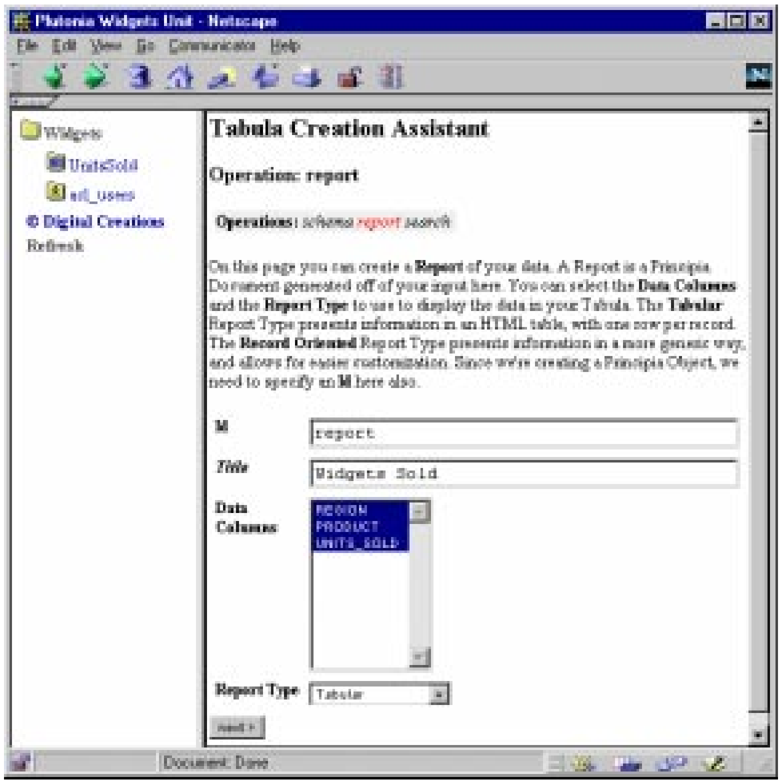
next >

Figure 26.

On the “Add Computed Field” screen there are three fields. The Id is used to address the field. It should be a legal Zope id. The type is similar to the types of regular fields. It is loosely enforced and exists primarily to enable computed fields to fit into the Schema. It is recommended to enforce a desired data type using functions available in the Expr machinery. The last field, "Expression", is for the expression to be used for the computed field. Due to how some Web browsers work with field contents it recommended to use single quotes instead of double quotes in expressions.

Deleting Computed Fields

To delete a computed field, go to the `Computed Fields` tab. Select the computed field(s) to be deleted by checking their respective checkboxes and click the `Delete` button. If any of the selected computed fields are being used by objects who may depend on them (primarily Hierarchies), a message will be displayed showing what objects are using the fields. The computed fields in question cannot be deleted if there is a conflict. Using a computed field in a Document is *not* considered a conflict and will not be flagged.



Tabula Creation Assistant

Operation: report

Operations: *schema report search*

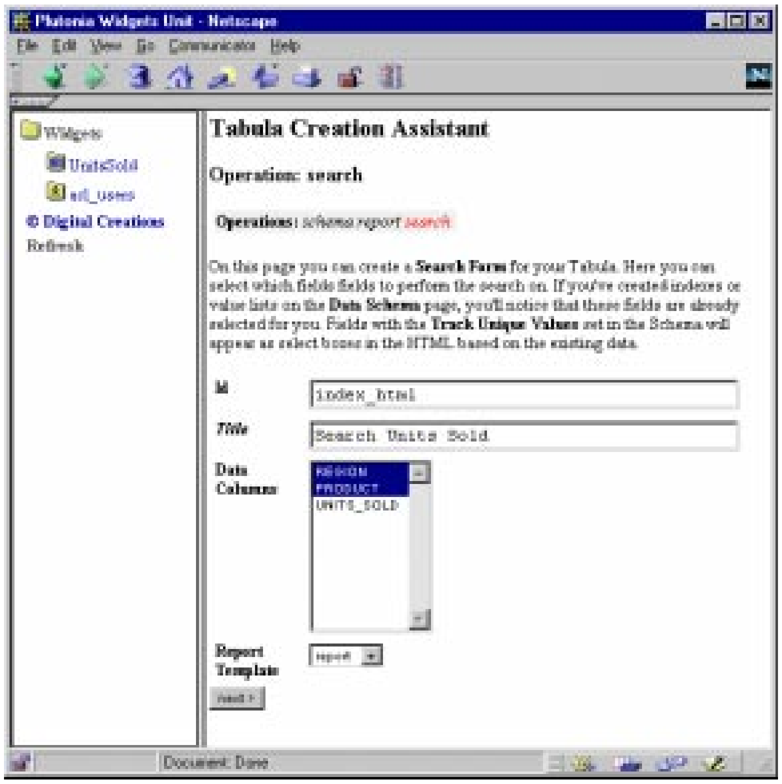
On this page you can create a **Report** of your data. A Report is a Principia Document generated off of your input here. You can select the **Data Columns** and the **Report Type** to use to display the data in your Tabula. The **Tabular** Report Type presents information in an HTML table, with one row per record. The **Record Oriented** Report Type presents information in a more generic way, and allows for easier customization. Since we're creating a Principia Object, we need to specify an **Id** here also.

Id	report
Title	Widgets Sold
Data Columns	REGION PRODUCT UNITS_SOLD
Report Type	Tabular

Figure 27.

Editing Computed Fields

To edit a computed field, go to the `Computed Fields` tab. Clicking the id of a computed field opens up its edit screen. It's similar to adding a new computed field, except the ID cannot be changed. The drop-down list next to the id sets the type, and the `Expression` field is for editing the expression to be used for the field.


Phatonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Widgets
UnitsSold
sql_users
© Digital Creations
Refresh
Document: Done

Tabula Creation Assistant

Operation: search

Operations: *schema report search*

On this page you can create a **Search Form** for your Tabula. Here you can select which fields fields to perform the search on. If you've created indexes or value lists on the **Data Schema** page, you'll notice that these fields are already selected for you. Fields with the **Track Unique Values** set in the Schema will appear as select boxes in the HTML based on the existing data.

Id	index_html
Title	Search Units Sold
Data Columns	REGION, PRODUCT (selected); UNITS_SOLD
Report Template	report

Next >

Figure 28.

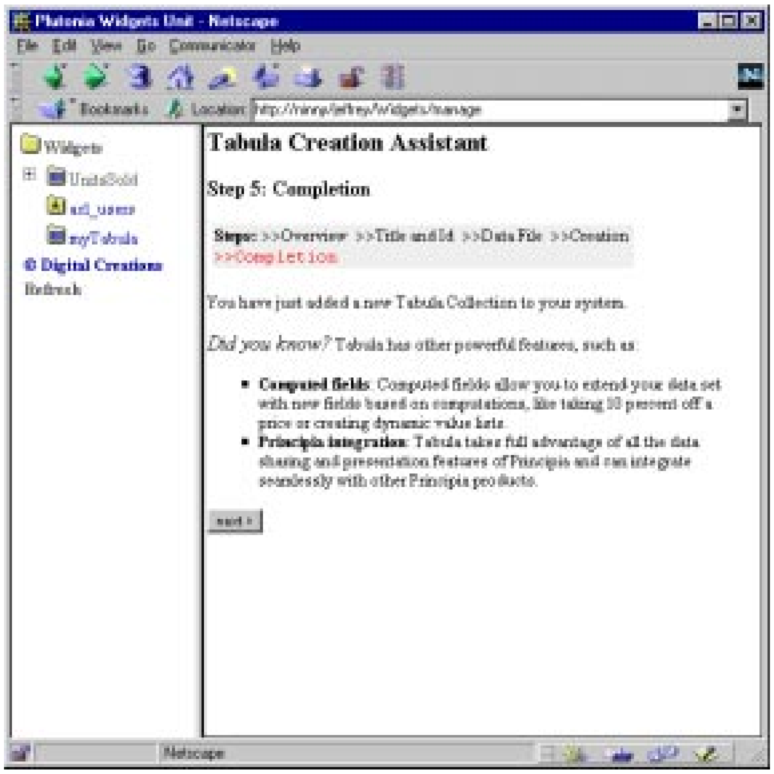
Modifying the Schema

The schema is the "meta-data" for the records in a Tabula. It contains information about each field, including whether a field is indexed or not. Indexes can be set on a field by field basis. Tabula has three index options: "No Index", "Field Index", and "Text Index". "No Index" is exactly what its name is: there is no index on that field. Text Indexes allow full text searching on a particular field, including boolean and proximity searches. Field Indexes index exact data in a certain field in each record. Field indexes also allow sorting of results.

- **Field Indexes** -- Best used on fields with many records of the same value. For example, a car maker or company department. When used in conjuction with `Track Unique Values` (see below), Field Indexes can produce very fast and accurate searches. Field Indexes can also be the target of the `sort-on` property/parameter for sorted results.
- **Text Indexes** -- Best used on large text fields or even small text fields where boolean or proximity searches would be appropriate. This is the only index that searches inside of a field for content, so a search for `jim` would return results with `Jim Parks` 'jim' `Jim P Stevens` and even "Jim's All-American Steakhouse".

Modifying the Schema

To modify the schema, go to the `Schema` tab.


Platonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://ninny/jeffrey/widgets/manage
Widgets
UnitsSold
acl_users
myTabula
© Digital Creations
Refresh
next >
Netscape

Tabula Creation Assistant

Step 5: Completion

Steps: >>Overview >>Title and Id >>Data File >>Creation >>Completion

You have just added a new Tabula Collection to your system.

Did you know? Tabula has other powerful features, such as:

- **Computed fields**: Computed fields allow you to extend your data set with new fields based on computations, like taking 10 percent off a price or creating dynamic value lists.
- **Principia integration**: Tabula takes full advantage of all the data sharing and presentation features of Principia and can integrate seamlessly with other Principia products.

Figure 29. Step 5

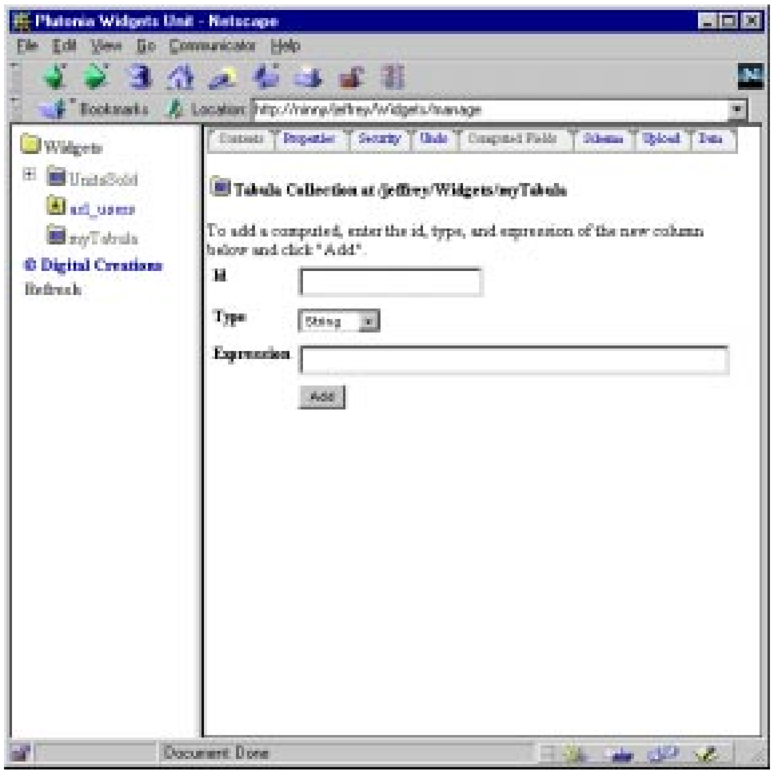
The schema tab holds a list of all fields in a table. It holds information for:

- **Column Name** -- The column (field) name
- **Type** -- The data type of the field. Should be one of **S** for string, **I** for integer, **N** for number (floating point), **D** for date, and **T** for text (a special Rand RDB format).
- **Input Width** -- The width of a field.
- **Indexing** -- The indexing options for a field. See above.
- **Track Unique Values** -- If this checkbox is marked, a list of all unique values in a specific field are kept track of. If a field with this option enabled is selected when creating a search form (see below), an HTML select list with all of the unique values will be generated.

These settings can all be modified at once and then made effective by clicking the "Change" button.

Uploading New Data

Tabula’s wouldn’t be very interesting if their data had to stay the same all the time. As information changes, Tabula makes it easy to make those changes available on the Web. To upload new data, go to the `Upload` tab.



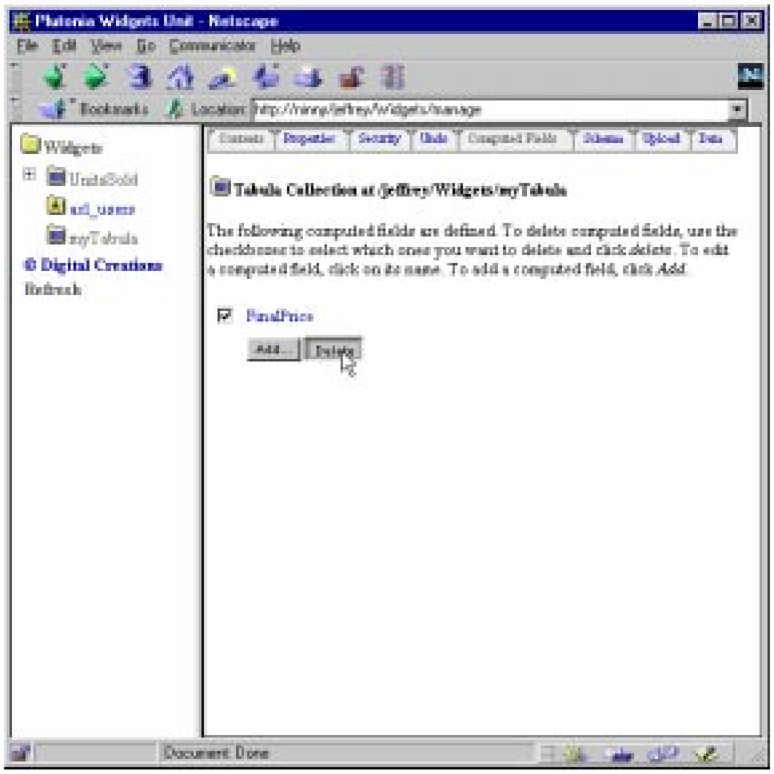
Tabula Collection at /jeffrey/Widgets/myTabula

To add a computed, enter the id, type, and expression of the new column below and click "Add".

Figure 30. Add Computed Field

The fields on the screen and their uses are:

- **Data File Format** -- The file format (DBase, Rand RDB) the data is in.
- **Data File** -- The file itself. Click on the `Browse...` button to select the data file off a local system, or type the full path to the file manually in the text field.
- **Memo File** -- *Only* if the Tabula was created from a DBF and Memo file will this field show up. If there's a memo file involved with the Tabula, follow the same instructions for the Data File field.
- **Remove Existing Data** -- Checking this option removes the records currently in the Tabula before adding the new records being uploaded. If unchecked, the new records are appended to the existing data.
- **Allow columns in the upload file that are not in the existing data** -- If checked, no errors will be reported if the file being uploaded contains columns that aren't part of the Tabula's schema. *This option does not add any new columns to the Tabula!*.
- **Allow columns to be omitted from the upload file that are in the existing data** -- If checked, no errors will be reported if the file being uploaded is missing columns that are part of the Tabula's schema. Empty values are placed in the missing fields. *This option does not remove columns from the Tabula!*.


Platonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://ninny/jeffrey/widgets/manage
Widgets
UnitsSold
acl_users
myTabula
© Digital Creations
Refresh
Contents Properties Security Undo Computed Fields Schema Upload Data
FinalPrice
Add...
Delete
Document: Done

Tabula Collection at /jeffrey/Widgets/myTabula

The following computed fields are defined. To delete computed fields, use the checkboxes to select which ones you want to delete and click *delete*. To edit a computed field, click on its name. To add a computed field, click *Add*.

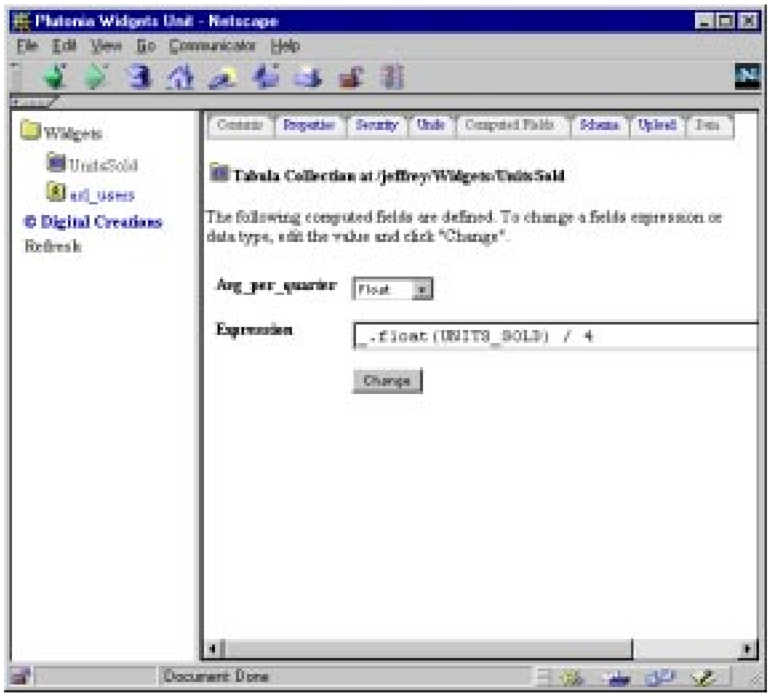
Figure 31. Deleting a Computed Field

Checking on data using the Data tab

As data changes and computed fields get added, it's nice to be able to check on an clean view of the data to make sure all is well. The `Data` tab presents such a view, unfettered by sorting options or REQUEST variables that can affect normal Tabula reports.

To view the data, click on the `Data` tab. The `Data` screen is basically one large table showing every column in the Tabula, including Computed Fields. Since the Data tab adjusts to the fields, it is a good testing ground for Computed Field results.

The Data screen is very similar to the default `Tabular Report` generated by the Tabula Report wizard (see below). The records are shown in batches of 20 with links at the top and bottom of the table for navigating forward and backward through the data.


Phalonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Contents Properties Security Undo Computed Fields Schema Upload Data
Widgets
UnitsSold
acl_users
© Digital Creations
Refresh
Document: Done

Tabula Collection at /jeffrey/Widgets/UnitsSold

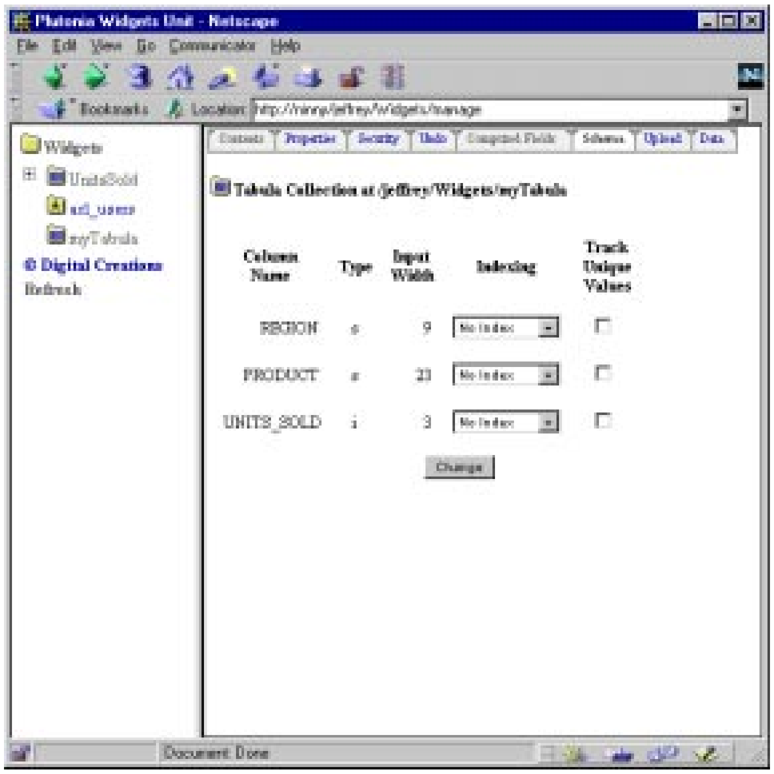
The following computed fields are defined. To change a fields expression or data type, edit the value and click "Change".

Avg_per_quarter: Float

Expression: `_.float(UNITS_SOLD) / 4`

Change

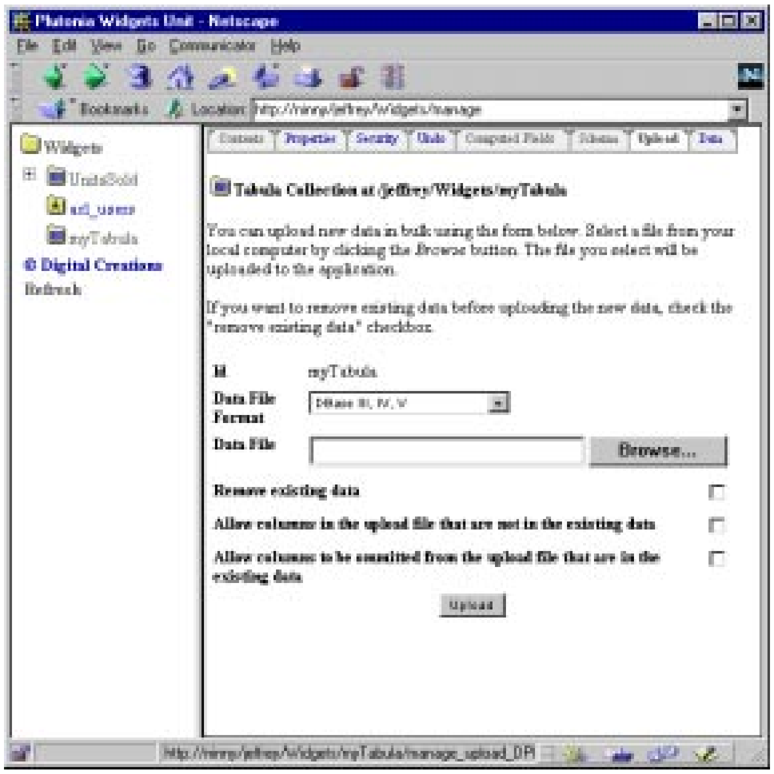
Figure 32.


Plutonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://ninny/jeffrey/widgets/manage
Widgets
UnitsSold
acl_users
myTabula
© Digital Creations
Refresh
Contents Properties Security Undo Computed Fields Schema Upload Data
Change
Document: Done

Tabula Collection at /jeffrey/Widgets/myTabula

Column Name	Type	Input Width	Indexing	Track Unique Values
REGION	s	9	No Index	☐
PRODUCT	s	23	No Index	☐
UNITS_SOLD	i	3	No Index	☐

Figure 33. The Schema Tab


Platonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://ninny/jeffrey/Widgets/manage
Contents Properties Security Undo Computed Fields Schema Upload Data
Widgets
UnitsSold
acl_users
myTabula
© Digital Creations
Refresh
Browse...
Upload
http://ninny/jeffrey/Widgets/myTabula/manage_upload_DPI

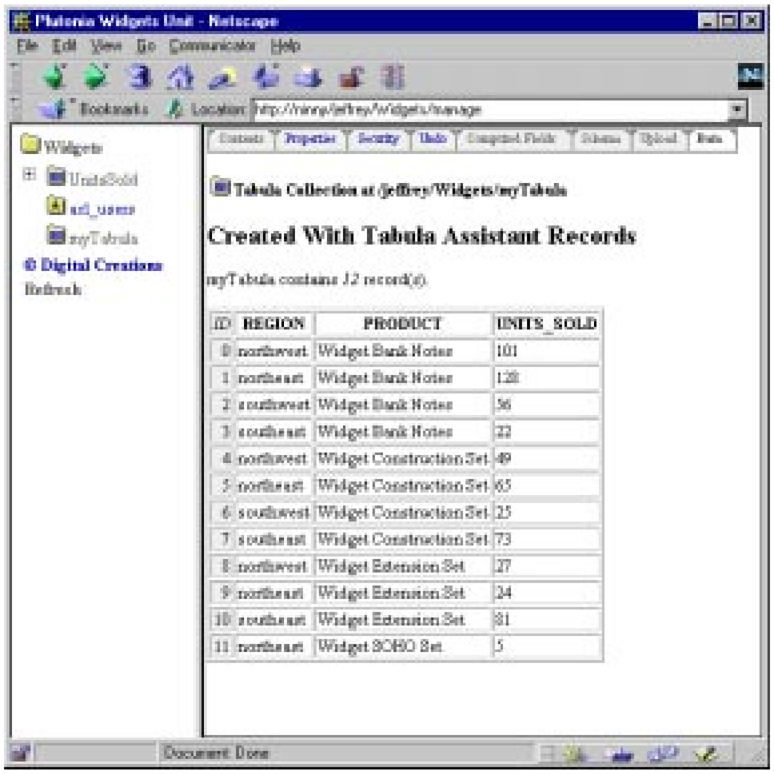
Tabula Collection at /jeffrey/Widgets/myTabula

You can upload new data in bulk using the form below. Select a file from your local computer by clicking the *Browse* button. The file you select will be uploaded to the application.

If you want to remove existing data before uploading the new data, check the "remove existing data" checkbox.

Id	myTabula
Data File Format	DBase III, IV, V
Data File	
Remove existing data	☐
Allow columns in the upload file that are not in the existing data	☐
Allow columns to be ommitted from the upload file that are in the existing data	☐

Figure 34. Upload


Platonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://ninny/jeffrey/widgets/manage
Contents Properties Security Undo Computed Fields Schema Upload Data
Widgets
UnitsSold
acl_users
myTabula
© Digital Creations
Refresh
Document: Done

Tabula Collection at /jeffrey/Widgets/myTabula

Created With Tabula Assistant Records

myTabula contains *12* record(s).

ID	REGION	PRODUCT	UNITS_SOLD
0	northwest	Widget Bank Notes	101
1	northeast	Widget Bank Notes	128
2	southwest	Widget Bank Notes	36
3	southeast	Widget Bank Notes	22
4	northwest	Widget Construction Set	49
5	northeast	Widget Construction Set	65
6	southwest	Widget Construction Set	25
7	southeast	Widget Construction Set	73
8	northwest	Widget Extension Set	27
9	northeast	Widget Extension Set	24
10	southeast	Widget Extension Set	81
11	northeast	Widget SOHO Set	5

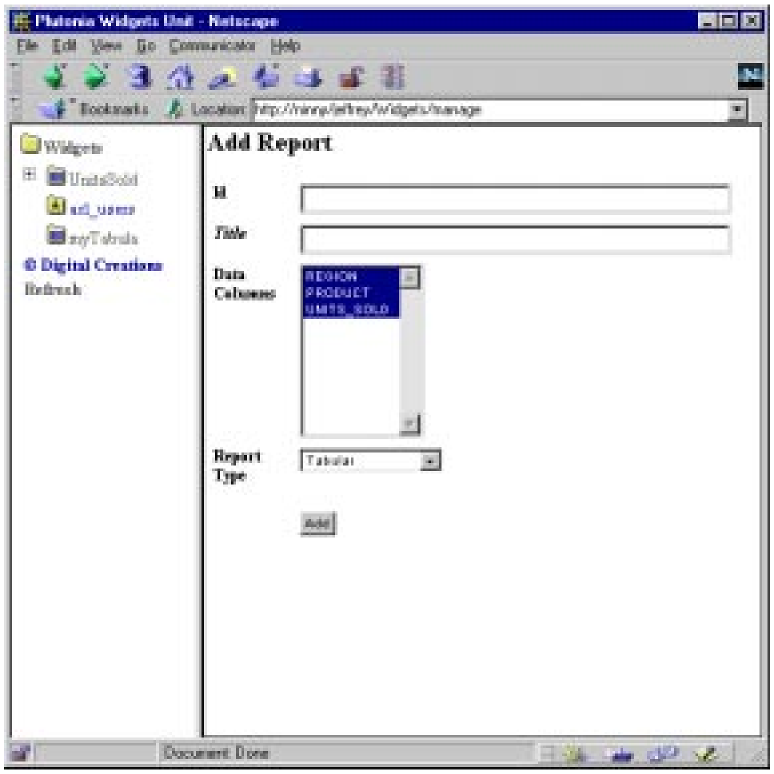
Figure 35. The Data Tab

Generating and Working with Tabula Reports

Data is useless if no one can see it. But writing a report by hand can be difficult work. Tabula offers a *Report Wizard* to assist in their creation. There are no report *objects* in Tabula. Instead, the report wizard generates Zope Document objects with DTML code based on the wizards input. Once generated, the report can be used as is, or edited like any other Zope document. The main benefit of using the report wizard is that it takes care of some powerful features such as batch processing (viewing only X amount of records at a time and being able to navigate forward and backward to the next X amount of records). For detailed coverage of batch processing using the DTML `In` tag, see the DTML documentation. Since reports are used to perform actual searches, some care must be taken to avoid conflicts and unreported data. This is discussed in the subsection "Reports as Search Targets".

Adding a Report

To add a Report, go to the `contents` tab of the Tabula to add the report to, choose "report" from the add list and click "Add."


Plutonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://ninny/jeffrey/Widgets/manage
Widgets
UnitsSold
acl_users
myTabula
© Digital Creations
Refresh
Document: Done

Add Report

Id

Title

Data Columns: REGION, PRODUCT, UNITS_SOLD

Report Type: Tabular

Add

Figure 36. Add Report

The Add Report screen comes up with the following fields:

- **ID** -- Zope Id for the report.
- **Title** -- Title attribute for the report (*optional*).
- **Data Columns** -- This selection list is of all the current columns (fields) in the Tabula. Select the columns you would like to include in the report.
- **Report Type** -- Tabular generates a report using an HTML table with the column names at the top of the table; Record-oriented generates a report with commas separating all of the selected fields. More detail is shown below.

Working with Reports

Reports as Search Targets

Tabula reports are also used for searching. A search form (see next section) performs a search by calling a report and passing the form data as parameters. Tabula performs searches *by example*. Name and value pairs are matched against the data to yield search results. For example:

```
.../cars/report?Dealer=Family+Auto
```

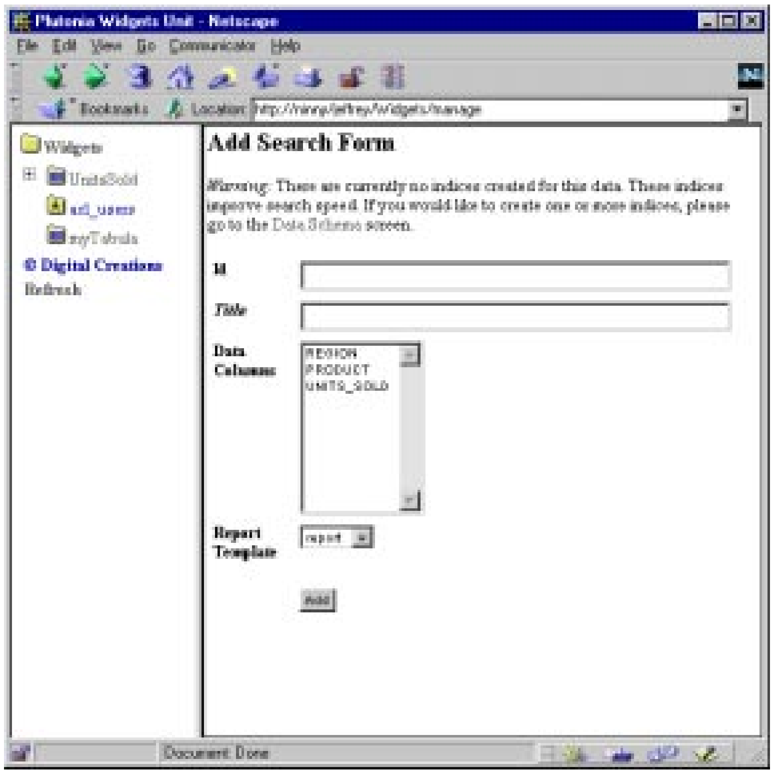
would only display records where the field `Dealer` contained the value `Family Auto`. Because reports take data out of passed parameters like above as well as the *REQUEST*, care must be taken to not accidentally set cookies or attributes with the same Id as fields in the Tabula since every time the report will be called, the cookie or attribute values will be passed along as search parameters. Since field Id's are case sensitive, an attribute with the id of `Dealer` would not conflict with a field with id `DEALER`. Tabula's created from DBF files will usually have all upper-case field ids.

Generating and Working with Tabula Search Forms

Tabula Search Forms are similar to Tabula Reports in that search forms are also Documents generated from a wizards input form.

Adding a Search Form

To add a Search Form, go to the Contents tab of the Tabula to add the search form to, choose "Search Form" from the add list and click "Add."


Platonia Widgets Unit - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://ninny/jeffrey/Widgets/manage
Widgets
UnitsSold
sql_users
myTabula
© Digital Creations
Refresh
Document: Done

Add Search Form

Warning: There are currently no indices created for this data. These indices improve search speed. If you would like to create one or more indices, please go to the Data Schema screen.

Id	
Title	
Data Columns	REGION PRODUCT UNITS_SOLD
Report Template	report

Add

Figure 37. Add Search Form

The Add Search Form screen comes up with the following fields:

- **ID** -- Zope Id for the search form.
- **Title** -- Title attribute for the search form (*optional*).
- **Data Columns** -- This selection list is of all the current columns (fields) in the Tabula. Select the columns you would like to perform the search on. The search wizard automatically selects columns which have indexing or "Track Unique Values" enabled. These don't have to be what's chosen as the search columns. Note that if a column is chosen that has the "Track Unique Values" option set (on the schema tab), when the search form is generated the column will have a selection list with all of the unique values of the field. All other fields get rendered as normal text input fields. Remember, once a document is generated it can be edited and modified at will.
- **Report Template** -- This is a list of all Document objects in the Tabula. Choose the document (which should be a report) that should display the results of the search.

Hierarchies

A very powerful feature of Tabula is its ability to create `Hierarchies`. Nested folder-like structures based on data. This gives many of the same benefits of normal Zope folders, allowing customizations to be made on any level. Hierarchies also are good users of the Zope `Tree` tag. Using the tree tag with a hierarchy would, for example, enable a document that could expand and contract records based on a Financial Quarter and Region, just by the user clicking on `+` and `-` icons. Furthermore, Second Quarter 1998 could have a unique look independent of the other quarters. When the data changes, the hierarchies get updated with it.

Creating a new Hierarchy

To create a new Hierarchy, go to the `contents` tab of the Tabula to add the hierarchy to, choose `Hierarchy` from the add list and click `Add...`.

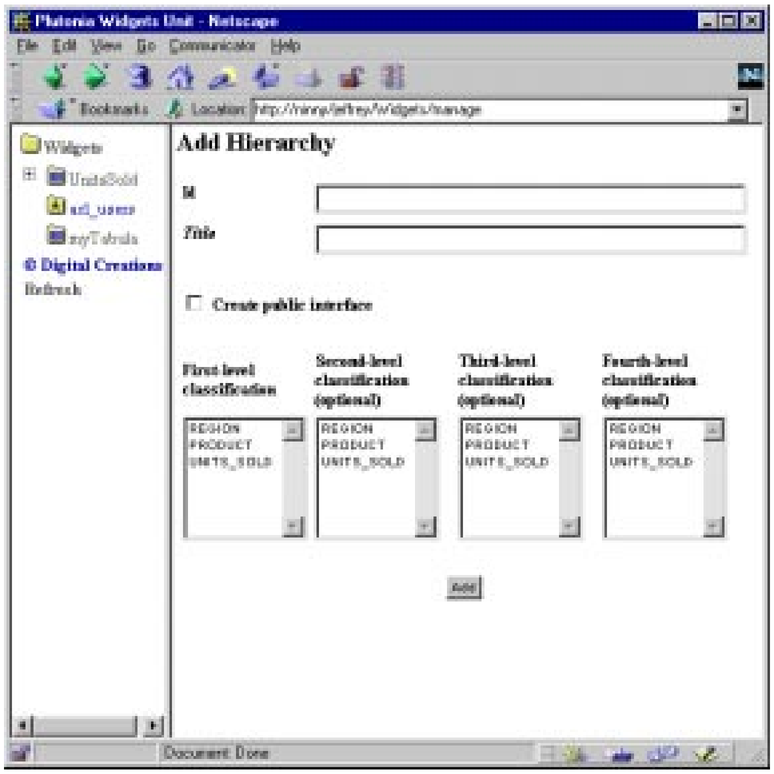

Add Hierarchy
Id
Title
Create public interface
First-level classification
Second-level classification (optional)
Third-level classification (optional)
Fourth-level classification (optional)
REGION
PRODUCT
UNITS_SOLD
Add

Figure 38. Add Hierarchy

The "Add Hierarchy" screen comes up with the following fields:

- **Id** -- Zope id for the hierarchy.
- **Title** -- *optional* Title attribute for the hierarchy.
- **Create Public Interface** -- If checked, an `index_html` document will be created in the top level of the hierarchy containing a simple Tree Tag.
- **XXXX Level Classification** -- These are the columns that the Hierarchies levels get built on. For example, the column `Financial_Quarter` might have 14 records with the value `1998Q1` and 8 with the value `1997Q4`. If `Financial_Quarter` were used for the first level classification, the first sub-hierarchies would be `1998Q1` and `1997Q4`. If the second level classification was the column `Region` with values `NorthEast`, `NorthWest`, and `South`, then there would be sub-hierarchies for those values that would be contained by the `Financial_Quarter` sub-hierarchies. The sub-hierarchies for the `Regions` would only contain references to records that also matched their parents Financial_Quarter value. This is explained in more detail below.

Working with Hierarchies as Folders

When a Hierarchy is made, it creates its own sub-hierarchies based on the contents of the column classifications it is created on. The base Hierarchy and the sub-hierarchies all act like Zope Folder objects. New Zope objects can be added to their contents list. But Hierarchies are more like mini-Tabulas. They have a data tab like a Tabula, but which only has the data relating to that particular level in the hierarchy. Reports and Search Forms can be created in a Hierarchy just like in a Tabula. New Hierarchies cannot be made inside of an existing Hierarchy or sub-hierarchy.

Working with Hierarchies and Acquisition

A very key element to working with Hierarchies is *Acquisition*. Acquisition is a powerful feature used in Zope that allows the sharing of information between objects hierarchically. So if an index_html document is defined at the root level of the hierarchy, all of the sub-hierarchies "acquire" it, unless they overwrite it. If there is a report document in the Tabula, any level of the Hierarchy can use it. The interesting thing is that a report viewed from deep down in the hierarchy will only show the records it references. This is another very powerful feature of Zope and Tabula that makes it easy to create different views of published data. For more information, read the Tabula Quickstart section on Hierarchies.