Table Visualizer Application (TV)

version 2.0

Fredrik Gustafson

1997-04-30

Typeset in LaTeX from SGML source using the DOCBUILDER 3.0 Document System.

Contents

Chapter 1

TV User's Guide

The Table Visualizer Application (*TV*) enables the user to examine ETS and Mnesia tables. Once a certain table has been opened in the tool, the content may be viewed in various levels of detail.

1.1 The Table Visualizer

The *Table Visualizer* is a tool that enables the user to examine ETS and Mnesia tables on any (connected) node in the currently running Erlang system. Once a certain table has been opened in the tool, the content may be viewed in various levels of detail. The content may also be edited, as well as sorted, using any element as key. It is also possible to search for a specified object or element. The table may be polled anytime, either regularly, at specified intervals, or manually. New and deleted objects, as well as those altered, are marked with characteristic colours.

Information about the table itself (permissions, storage type, and so on) may also be obtained.

Terminology and Background

To avoid confusion, we have to distinguish between the *actual table*, i.e., the data stored in ETS or Mnesia, and the *image of the table*, i.e., the data shown in the Table Visualizer. The *image of the table* is simply a copy of the *actual table*, and can be manipulated in a number of ways, for example sorted. It follows that these manipulations in no way affects the *actual table*!

The expression *poll the table* is used for the operation of scanning through the content of the actual table (in order to keep the image of the table consistent with the actual table).

The ETS and Mnesia modules provides the user with the ability to store vast quantities of data in an Erlang system, the data organized as dynamic, unordered tables. The ETS facility stores *tuples*, while Mnesia stores *records*. Each tuple consists of one or more *elements*; each record consists of one or more *fields*. It should be noted that, since records are implemented as tuples, with the record name as the first element, the first field of a record becomes the second element in the corresponding tuple!
In the following, all table objects are mainly referred to as tuples, regardless of the table type.

For further information about ETS and Mnesia, please see the manual pages or the reference manual.

Starting the Table Visualizer

The Table Visualizer tool is started by giving the command

```
tv:start().
```

The window that appears, is hereafter referred to as *the Table Visualizer main window*. It consists of:

- a *menubar*.
- a *grid*, i.e., a multicolumnar array, where tables existing on the current node is shown. Each square in the grid is called a *cell*.

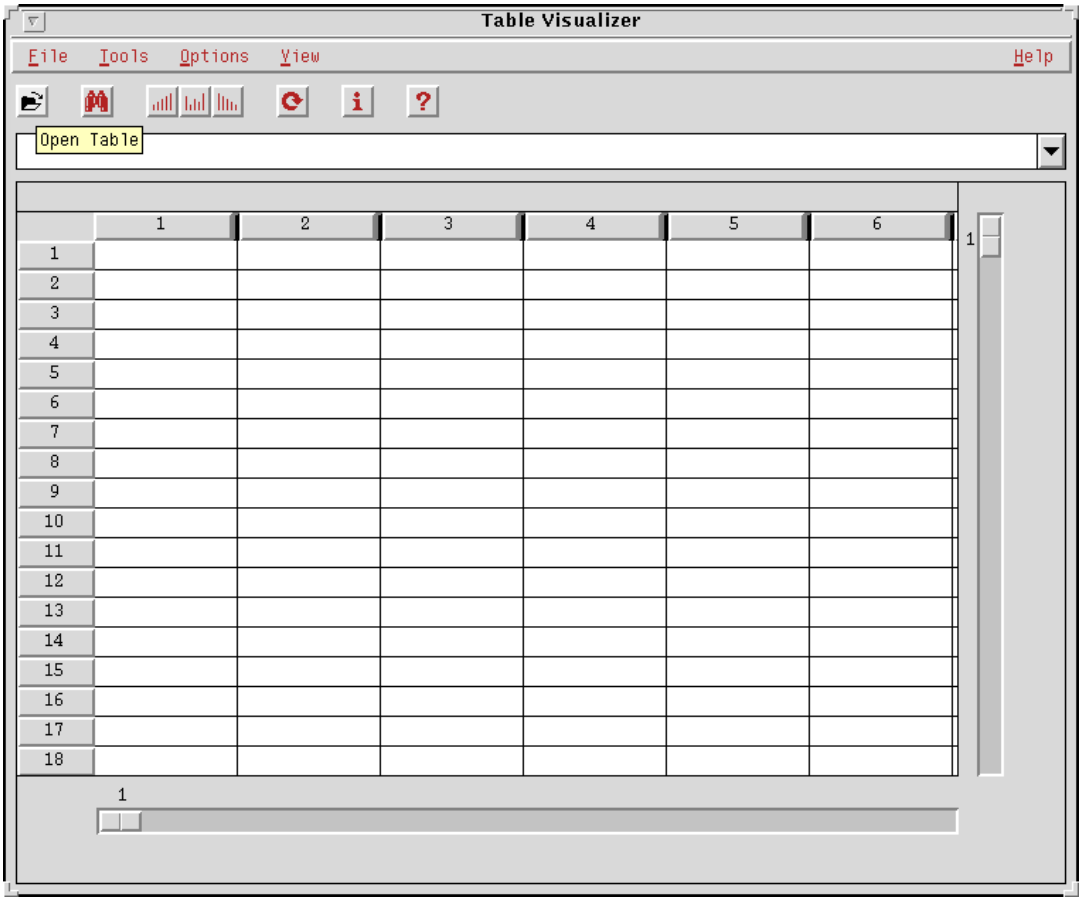

Figure 1.1: The Table Visualizer Main Window at startup.

For each table, the following information is shown, in order:

- the *table name*. If the table is accessible through this name, as is the case with Mnesia tables and named ETS tables, the table name is shown in black, otherwise in medium grey.
- the *table identifier*, if there is one; since Mnesia tables are accessed solely through the table name, this cell will in those cases be blank.
- the *process identifier (PID) of the process owning the table*.
- the *name of the process owning the table*, provided the process is registered.
- the *table size*, i.e., the number of objects currently stored in the table.

Changing the Table Visualizer View

The Table Visualizer will by default show currently existing ETS tables, but the user may easily switch to a Mnesia table view, by choosing the *Mnesia Tables* option in the *View* menu:

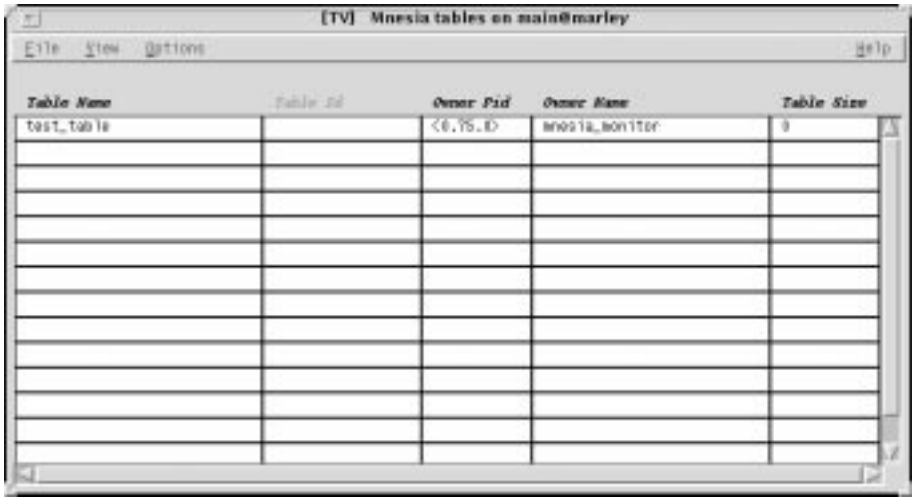

Figure 1.2: The Table Visualizer Main Window, showing Mnesia tables.

Normally, system tables (i.e., tables used by system applications) and unreadable tables are not shown. The menu option *System Tables*, in the *Options* menu, makes the system tables visible:

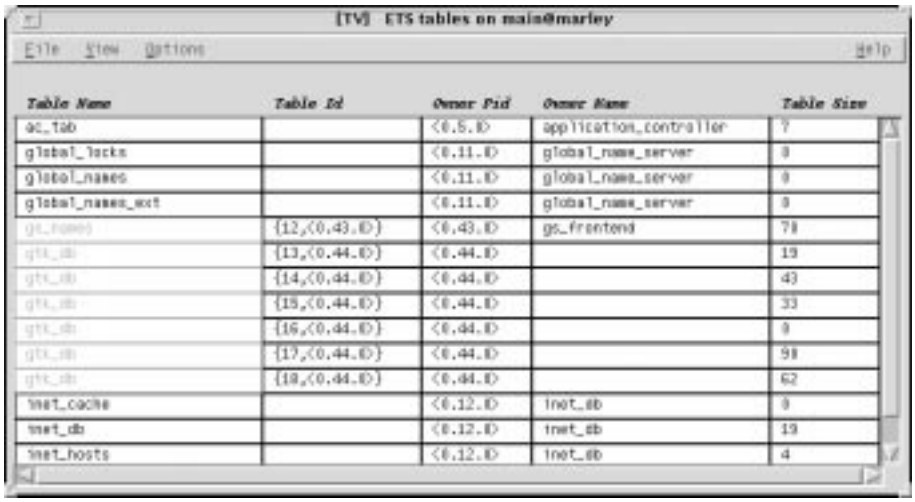

Figure 1.3: The Table Visualizer Main Window, showing readable user and system tables.

Still unreadable tables are hidden, but the menu option *Unreadable Tables*, also in the *Options* menu, makes even those tables visible. It shall be noted that rows containing unreadable tables are shaded, using a grey colour:

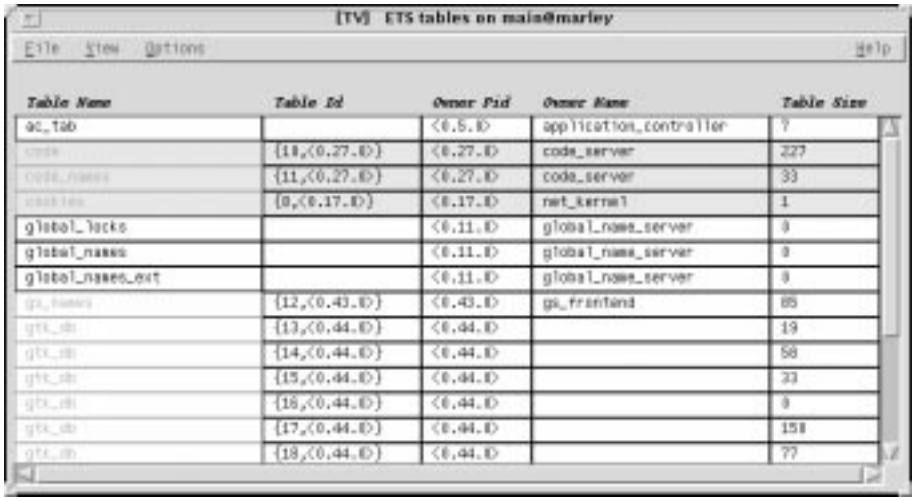

Figure 1.4: The Table Visualizer Main Window, showing both readable and unreadable user and system tables.

Once a table view has been opened, the user may choose how to view it: it may be sorted by the table names, by the table identifiers, by the process identifiers of the owner processes, or by the names of the owning processes. These sorting options are found in the *Options* menu.

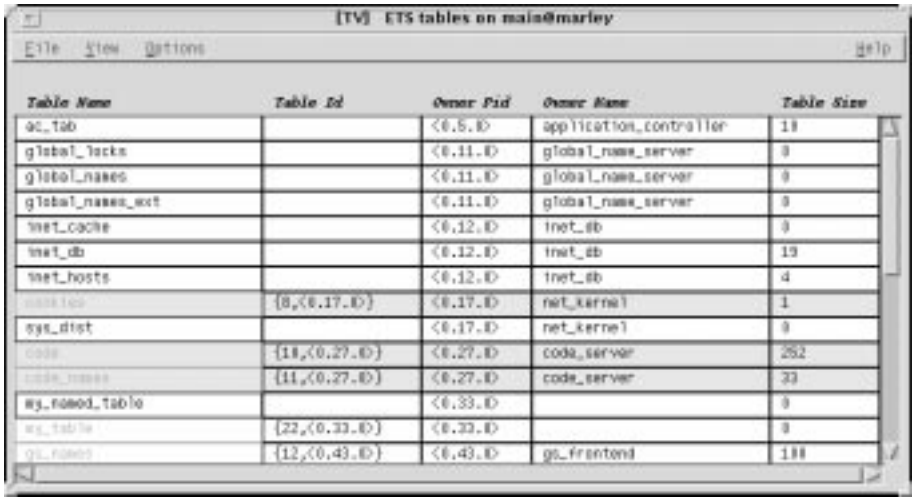

Figure 1.5: The Table Visualizer Main Window, tables sorted by owner PID.

Changing the Current Node

By default, the Table Vizualizer will show tables residing on the node it was started from. However, the user may easily view tables on other nodes. By choosing the *Nodes* option, in the *File* menu, a window

showing all connected nodes will appear. Clicking on any of the nodes in the list will cause the main window to immediately show the tables residing on the specified node:

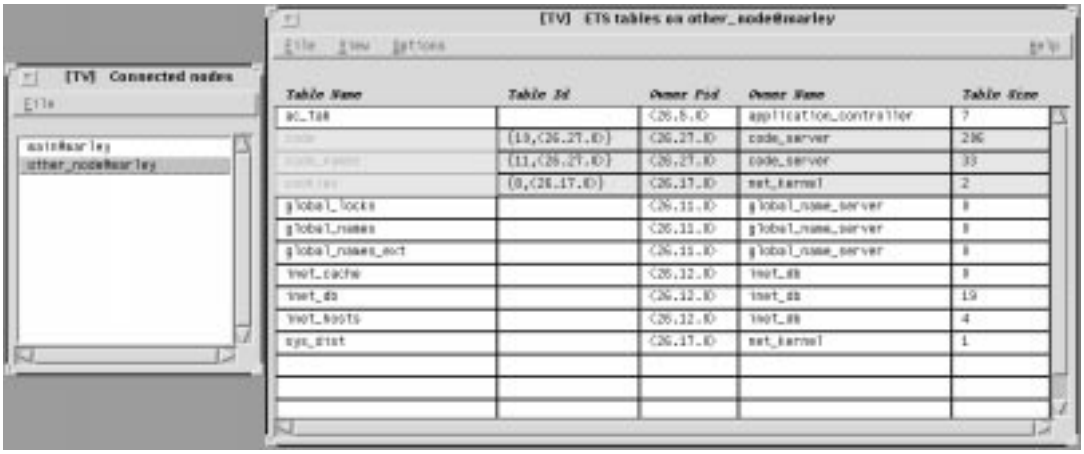

Figure 1.6: The Connected Nodes window, and the Table Visualizer Main Window, showing tables on the selected node.

Opening a Table in the Table Browser

Whenever a table shall be opened, the first step is to choose the corresponding *Table Name* or *Table ID* cell. Secondly, the *Open Table* menu item, in the *File* menu, has to be chosen. (Or, one may directly double-click on a *Table Name* or *Table Id* cell.)

If the table selected table is readable, a window will appear after a short delay. This new window is hereafter denoted the *Table Browser* window. Should the table be unreadable, the Table Information window will appear instead (see further description below).

The Table Browser Window

The Table Browser window consists of:

- a *menubar*.
- a *toolbar* with buttons providing shortcuts to the menubar options. If the cursor rests on any button, a so-called toolbar tip, explaining the button, will appear.
  (In the picture below, the cursor has lingered on the *Open Table* button for a while.)
- a *content and edit field*, showing the content of a specified row or cell. Through this field the row, or cell, may also be edited (see below for a detailed description).
- a *grid*, i.e., a multicolumnar array, where the content of the opened table will be shown. (As above, each square in the grid is called a cell.

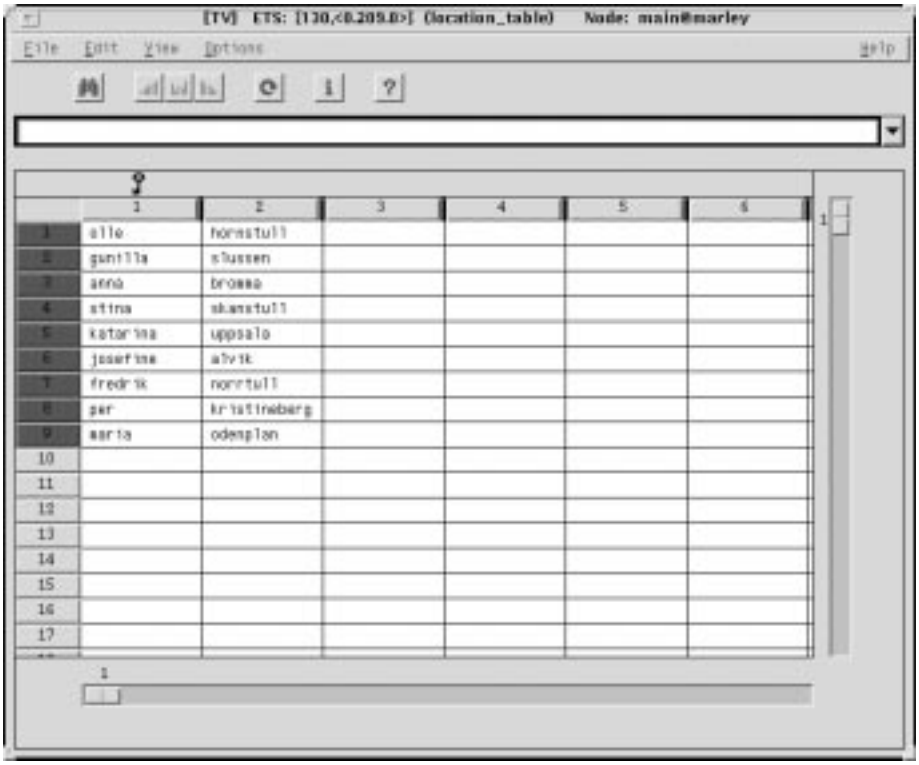

Figure 1.7: The Table Browser Window.

The successful appearance of the Table Browser window means that an image of the selected table has been created in the Table Visualizer. It is this image that is shown in the Table Browser.

How Table Data Is Presented Each object in the table is presented on a row of its own in the grid. Each element in the object is presented in a cell of its own.
The colours on the *vertical* buttons to the left of the grid show the status of the object on that very row: a bright red colour indicates that the object just has been inserted (when the table is opened, all objects are regarded as being just inserted), while a bright green colour indicates that the object has been changed. The colour fades away, shade by shade, every time the actual table is polled, until the normal background colour is encountered.
When an object has been deleted, the colour of the corresponding *vertical button* turns to black. The next time the table is polled, the object will be removed from the grid.

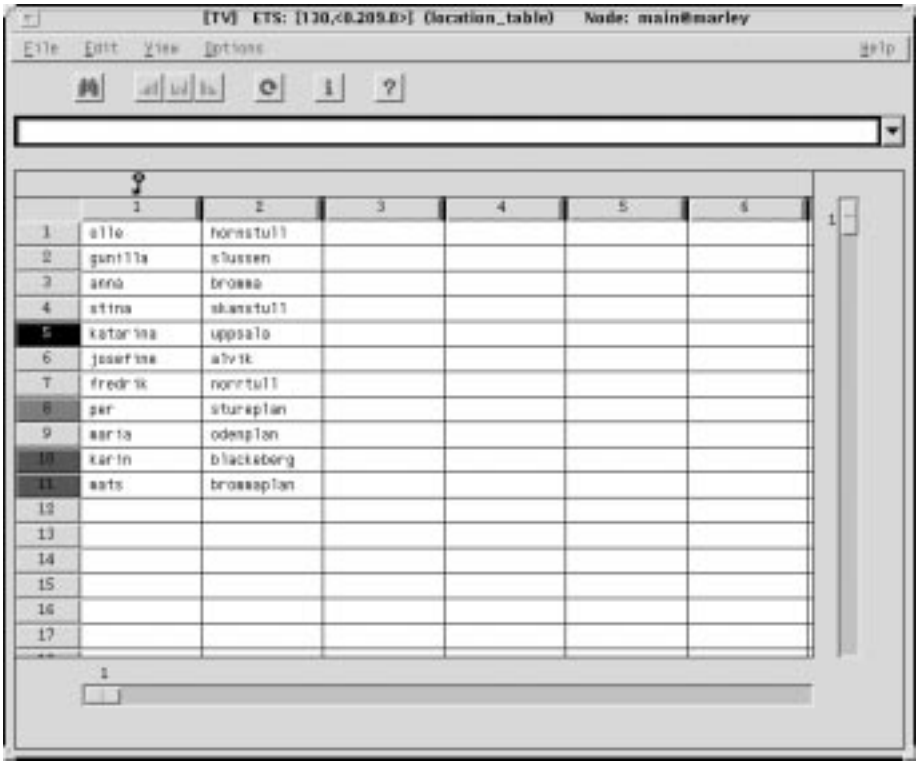

Figure 1.8: The Table Browser Window, with new, changed, and deleted objects.

Normally, new objects are placed at the end of the grid, while all other objects maintain their positions between successive polls. However, when sorting mode has been ordered, all objects, even new ones, are placed at the correct position according to the sorting ordered (see also below).

Immediately above the *horizontal buttons*, one or more *keys* may appear. These keys indicates which elements that are used as indices in the ETS/Mnesia table, i.e., which fields that are used by ETS/Mnesia as search keys when looking up data.

The grid columns may be resized, by clicking and dragging on the small black *resize areas* between any two horizontal buttons.

The rows are enumerated, as a help when navigating through the table. Note: it shall not be assumed that these numbers correspond to the placement of the objects in the *actual table*! The row numbers, as presented in the Table Visualizer, are only temporary, and only valid within the Table Visualizer! The number on the *vertical scrollbar* corresponds to the number the uppermost row has (or will have).

The number shown on the horizontal scrollbar relates to the leftmost column shown.

How to Poll the Table The table is polled whenever the *Poll Table* option in the *Options* menu is chosen (or the *Poll Table* toolbar button is pressed).
The user may also choose to let the Table Visualizer poll the table at regular intervals. This is done via the *Set Poll Interval...* option in the *Options* menu, which causes the *Set Poll Interval window* to appear.

In the Set Poll Interval window the user selects whether manual or automatic polling shall be used, and, in the automatic polling case, the poll interval.

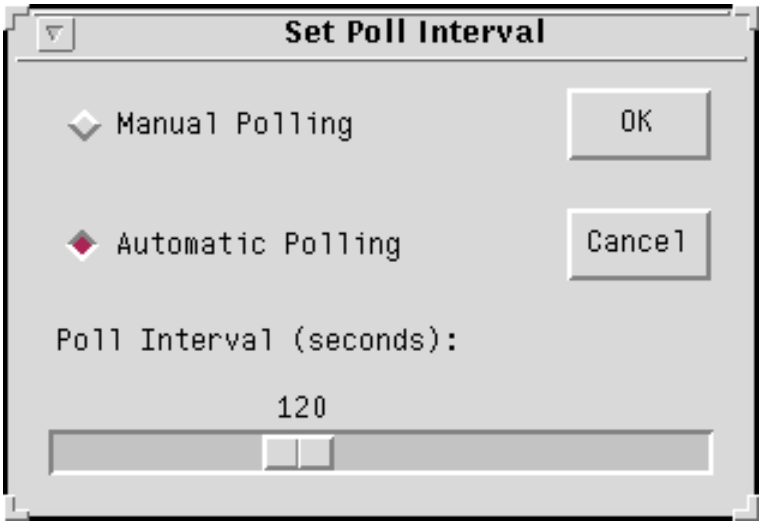

Figure 1.9: The Set Poll Interval Window.

It shall be noted that, in the case of a large table (or a slow computer/operating system), a short poll interval may cause the Table Visualizer to be flooded, i.e., the data resulting from one poll hasn't been fully treated and presented when the data from the next poll arrives. The user is therefore kindly requested to use the automatic polling facility with care!

How to Edit Objects in the Table Provided that the table is writable for other processes than the owning process, the user may insert, change and delete objects.

To *delete* an object, the corresponding row, or a single cell in the corresponding row, has to be chosen, by clicking either on the vertical button to the left of the row, or on a cell. Thereafter the *Delete Object* option in the *Edit Menu* is chosen. (Should the user regret the delete operation, the row may once again be selected, whereupon the *Return* button simply is pressed.)

To *insert* an object, the user may use the *Record Editor*, or simply enter the object in the content and edit field, and then press the *Return* button.
The *Record Editor* is started via the *Edit Object* option in the *Edit* menu, or via the *Edit Object* toolbar button. The editor that appears looks different depending on the kind of table: for Mnesia tables, a writable field is shown for each record entry, as well as the name of the entry. For ETS tables, only a writable field is shown; this is due to the fact that the size of the tuples inserted in ETS tables may vary, whereas the size of the records inserted in a Mnesia table (more or less) is fixed. One may select the next field in the record editor by pressing the 'Tab' (or 'Arrow Down') button, and select the previous field by pressing 'Shift'+'Tab' (or 'Arrow Up').
When the editing is finished, the *OK* button may be clicked, or 'Return' pressed. The Table Visualizer will then try to insert the new object.

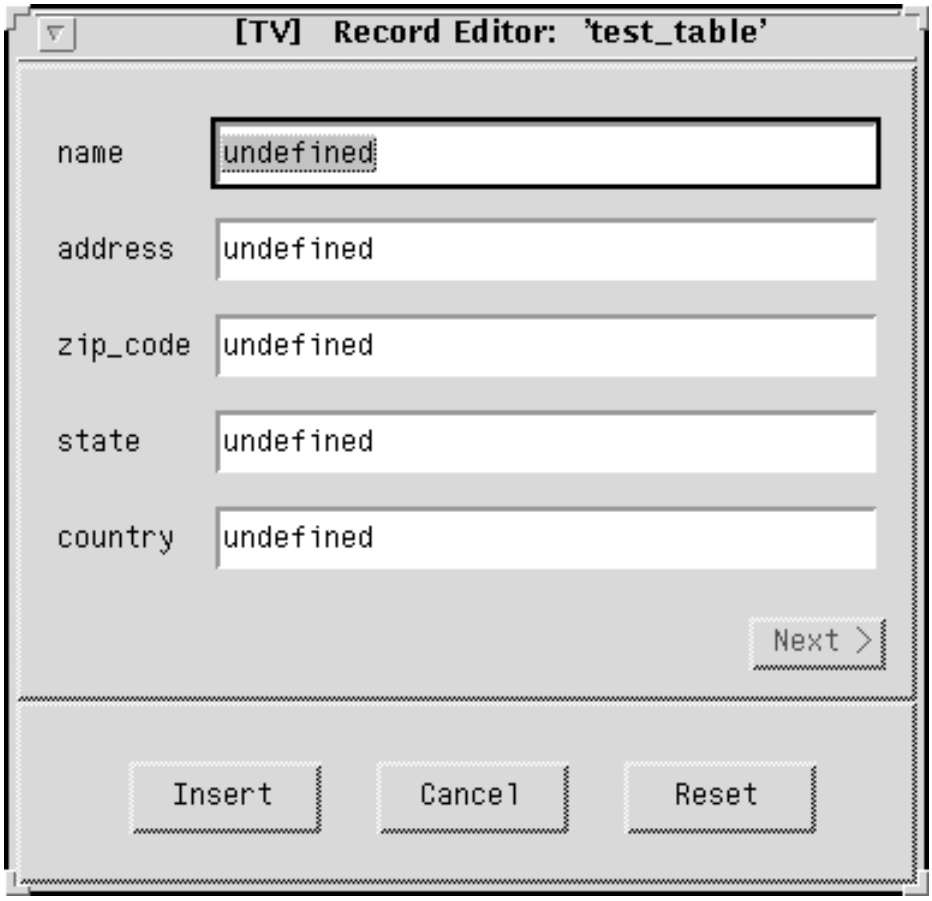

Figure 1.10: The Record Editor (shown for a Mnesia table).

To *change* an already existing object, the corresponding row, or a single cell in the corresponding row, has to be selected first (see below). Then one may edit the whole object (or the selected field in the object), either using the record editor or the content and edit field, whereupon 'Return' may be pressed (or the 'OK' button clicked).

It shall be noted that it is hard to edit objects containing *process identifiers*, *references*, *binaries* and *ports*, since it is only a textual representation of these terms that is shown on the screen. It is in the general case impossible for the Table Visualizer to correctly convert this textual representation back to the original term. As a courtesy to the user, an attempt to do this will nevertheless be done if the edited field consists of a single process identifier; however, this conversion will only be correct provided that the process identifier originates from the current Erlang session. (On the other hand, why on earth should any user want to store old process identifiers?)

It shall also be noted that it may be more or less confusing to edit the table, depending on whether the table type is *set*, *bag*, or *duplicate_bag*, i.e., depending on whether or not objects having the same key (or even duplicate objetcts) are allowed. Please study the ETS or Mnesia manual pages, should confusion arise!

How to Search For Objects One may search for an object, by choosing the *Search Object* option in the *Options* menu (or by pressing the *Search Object* toolbar button). In the *Search Object window* that

appears, any valid Erlang term or regular expression may be entered, whereupon all objects containing (or consisting of) this term, or matching the regular expression, will be shown.

Figure 1.11: The Search Object Window.

In the search result list, by clicking on any object, the Table Browser will immediately scroll to the corresponding row in the table shown. This enables the user to in a very powerful way quickly find the objects he's interested in.

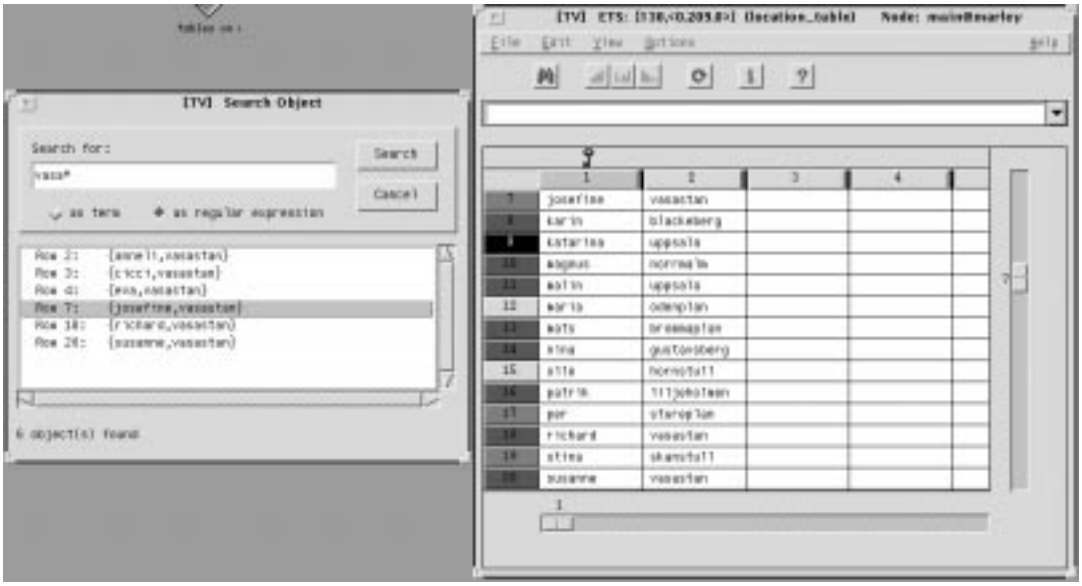

Figure 1.12: The Search Object Window interworking with the Table Browser.

How to Mark Table Data One may mark a row or a column by clicking on the buttons to the left and above the grid, respectively. A single cell is marked by clicking on it. Even empty rows and columns

may be marked; an empty cell cannot be marked - on the contrary, by clicking on an empty cell, all marks are removed.

Marks are indicated by a cyan blue colour.

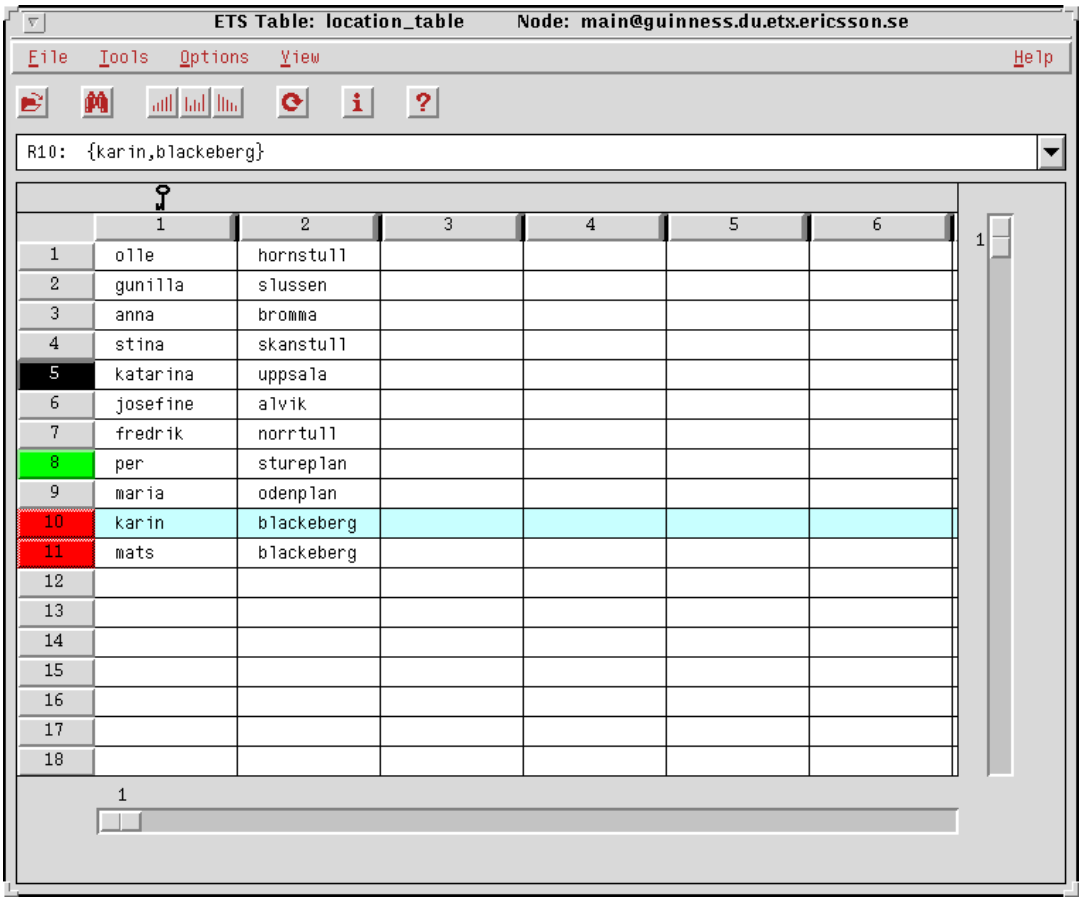

Figure 1.13: The Table Visualizer Main Window: a row has been marked.

When a row or a cell has been marked, the content will be shown in the content field, together with an indication of the row (and column when applicable) the marked area corresponds to. Should the object be very big, only a fraction of it may be shown in this field. By clicking on the down-arrow button to the right of the content and edit field, a pop-up content field will be shown, where the whole marked object may be viewed. The content of this pop-up field may be marked and copied to other windows; however, this field cannot be edited.

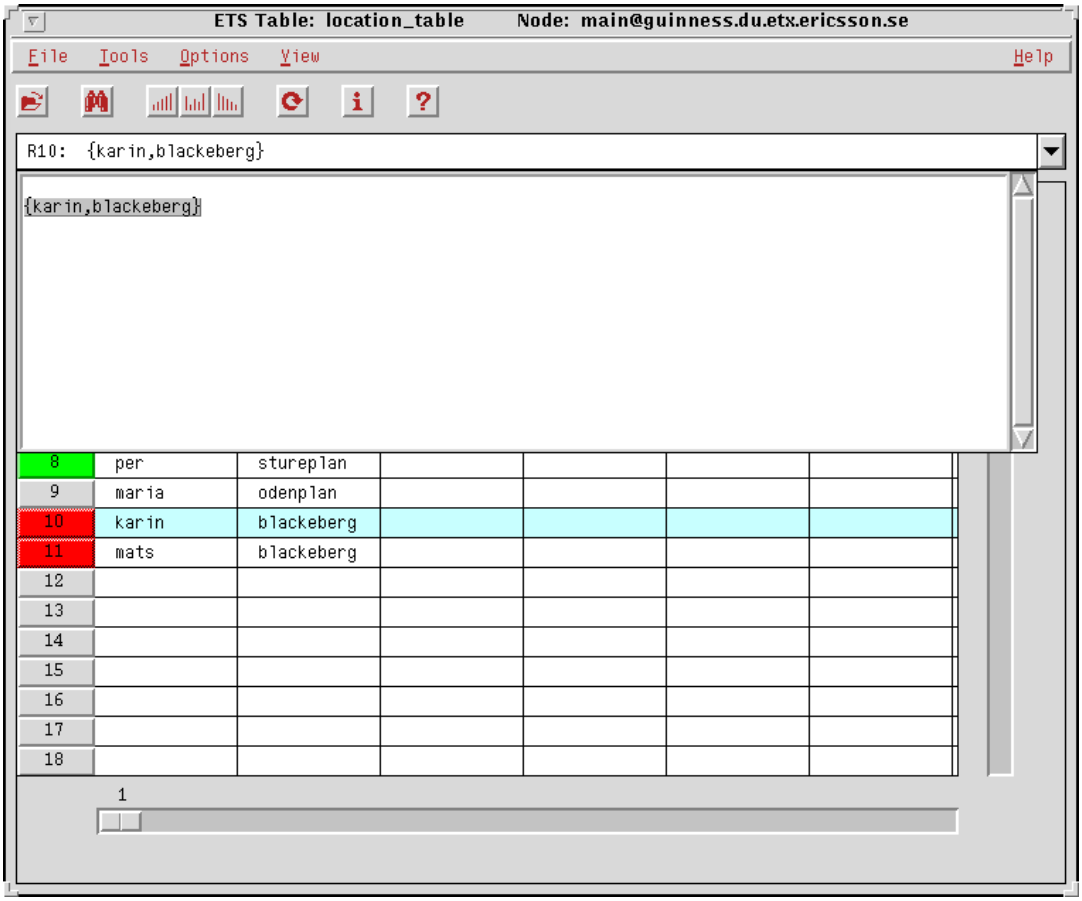

Figure 1.14: The Table Browser: the pop-up content field.

It shall be noted that the user may choose whether lists shall be shown as strings or lists; this is done via the *View* menu.

A marked column may be subject to sorting, see below. When sorting is ordered, marks are removed at each polling of the table (because of the difficulties to keep track of a certain object, or element, in this case).

How to Sort Table Data The image of the table may be sorted in rising or falling order, using any element as sorting key. The element desired is chosen by marking the corresponding column, and then choose (either via the *Options* menu, or via the toolbar buttons) any of the sorting options available, i.e., sorting in ascending or descending order. The colour of the column button will then change to gold, to indicate that this column is the basis for the sorting currently chosen.

Should no column have been marked, when sorting is ordered, the first element in each object (i.e, tuple) will be used as sorting key if the table is an ETS table; the second element (i.e., the first field in the record) will be used if the table is a Mnesia table.

Even columns with no elements in them may be subject to sorting. In this case the whole object is used as the sorting key.

When sorting is ordered, new elements will be inserted according to the current sorting mode. When the sorting is interrupted (via the *No Sorting* option), the current image of the table keep the current order, but new elements will from now on once again be inserted at the end of the image of the table.

How to Obtain Table Information Information about the actual table is obtained via the *File* menu (or via the *Table Info* toolbar button). The information is printed in a separate window, with similar pieces of information grouped together on “flap cards” of their own. By clicking on a flap, the information on the corresponding card is made visible.

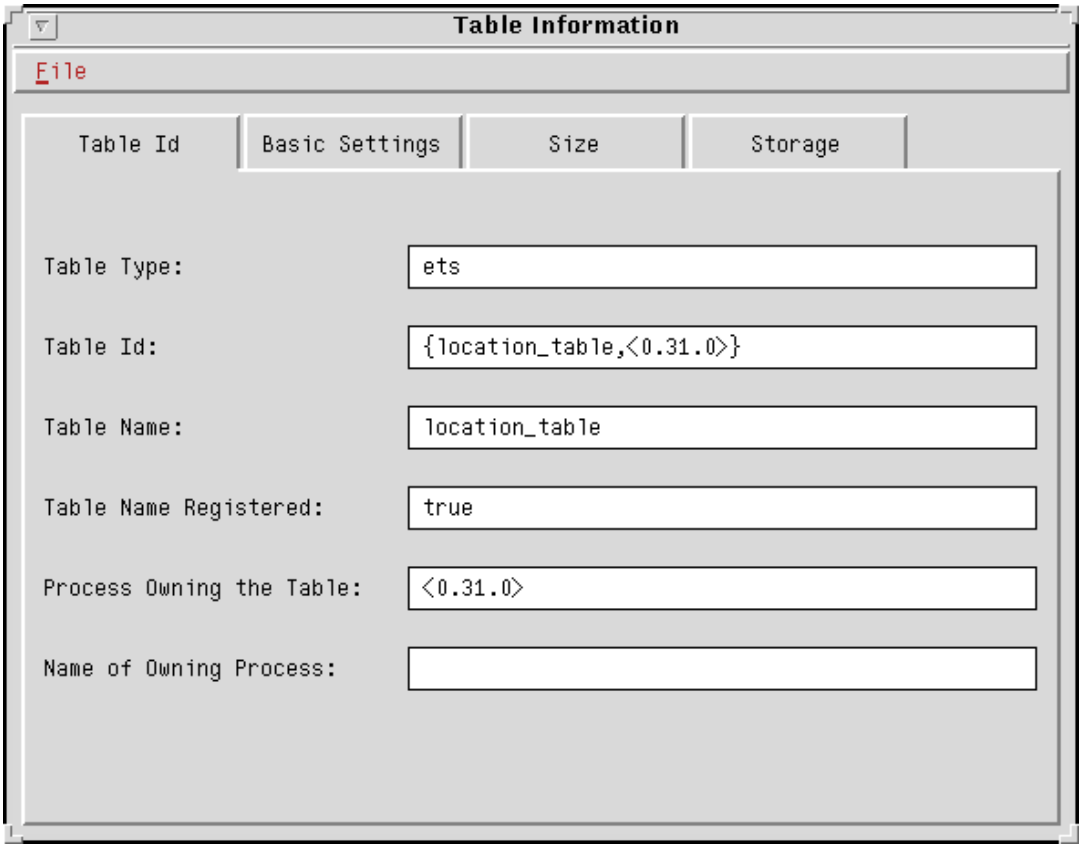

Figure 1.15: The Table Information Window, showing information about a Mnesia table.

The Table Information window may also be opened from the Table Visualizer Main Window, by selecting a table and then choose the *Table Info* option in the *File* menu (or by double-clicking on the *Table Size* field.

Note: The Table Information window will automatically be opened if the user tries to open an unreadable table, since this is the only information available in this case.

The Table Browser Menus

The Table Browser offers the following menus:

The File Menu

Table Info Opens the Table Information window, which shows the available information about the current table.

Close Closes the Table Browser window.

The Edit Menu

Edit Object... Opens the Record Editor. If an object is marked, it will be shown in the Record Editor.

Delete Object Deletes a marked object.

The View Menu

Lists As Lists Causes lists in the table to be shwon as lists.

Lists As Strings Causes lists in the table to be shown as strings.

The Options Menu

Poll Table An explicit order to poll the table, i.e., to scan the content.

Poll Interval... Choose between manual and automatic polling. In the case of automatic polling, the user gets the opportunity to choose the polling interval.

Search Object Enables search for objects containing (or consisting of) a specified Erlang term, or matching a regular pattern. The search result may be used for quick navigation in the table.

Sort Ascending Order Shows the table content sorted in ascending (i.e., rising) order. New objects will be shown with correct placement as long the as the sorting is going on.
Please note that it is only the image of the table that is affected, *not* the table itself!

Sort Descending Order Shows the table content sorted in descending (i.e., falling) order. New objects will be shown with correct placement as long the as the sorting is going on.

No Sorting Sorting mode is left. New objects will be shown last in the table. However, older objects will remain in the position they had when the sorting mode was left, i.e., their placement will not reflect their actual placement in the ETS/Mnesia table.

The Help Menu

Help Shows the help (about Table Visualizer usage) that is available. (The help will be shown in the Netscape Internet browser, if available.)

OTP Documentation Shows the Documentation about all OTP components that is available in the local installation of OTP.

Tracing the Owner Process

The process owning the table may easily be traced, by selecting either the *Owner Pid* or the *Owner Name* field, and then choosing the *Trace Process* option in the *File* menu.
(It is also possible to double-click on any of these fields.)

Creating a New Table

A new table may easily be created using the *New Table window*. Currently only ETS tables may be created. Since ETS tables dies together with the parent process, a special process, registered as *tv_table_owner*, will be the owner of tables created this way. This process will not be affected by any termination of the Table Visualizer, i.e., the ETS tables created will live on until they are explicitly killed.

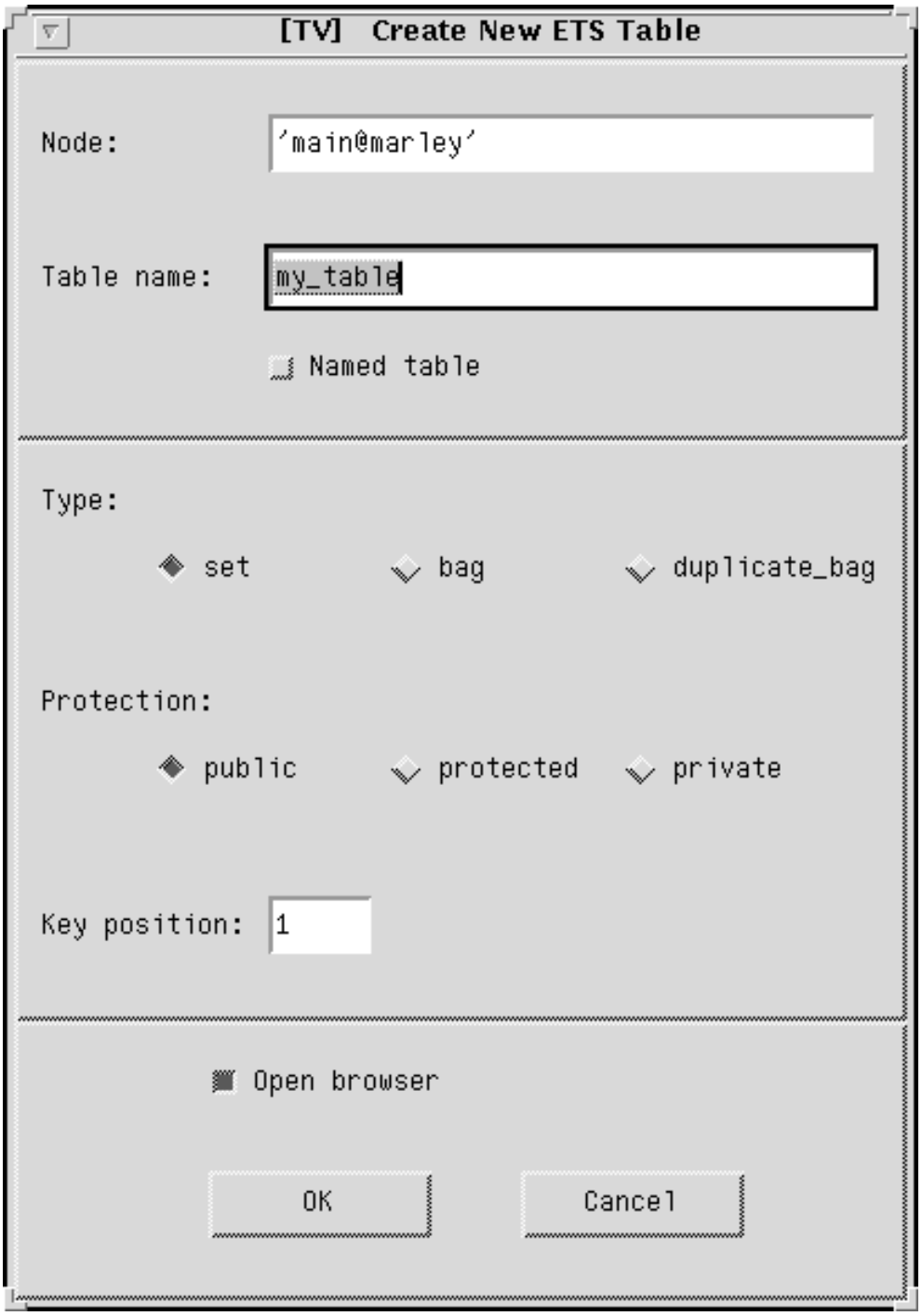

Figure 1.16: The New Table Window, enabling easy creation of ETS tables.

Note: the *tv_table_owner* is local to each node, meaning that the creation of a table on a new node also will start such a process on that node. This way only the tables on a specific node dies, should that specific node crash.

The Table Visualizer Main Window Menus

The Main Window offers the following menus:

The File Menu

Open Table Open a selected table in a new Table Browser.

New Table Open the New Table window, enabling easy creation of ETS tables.

Table Info Opens the Table Information window, showing the available information about a selected table.

Nodes... Open the Connected Nodes window, enabling the user to view tables residing on remote nodes.

Trace Process Opens a trace window, where the process owning a selected table can be traced.

Exit Terminates the Table Visualizer.

The View Menu

ETS Tables Shows ETS tables on the current node.

Mnesia tables Shows Mnesia tables on the current node.

The Options Menu

Refresh An explicit order to once again check the current node for existing tables, and list them.

Unreadable Tables Option to choose whether or not unreadable tables shall be shown.

System Tables Option to choose whether or not system tables shall be shown.

Sort by Name Shows the tables sorted by their names.

Sort by Id Shows the tables sorted by their table identifiers.

Sort by Owner PID Shows the tables sorted by the process identifiers of the owning processes.

Sort by Owner Name Shows the tables sorted by the registered names of the owning processes.

Error Messages in Haiku Option to choose whether or not error messages shall be shown in the Japanes poetry style called *Haiku*.

The Help Menu

Help Shows the help (about Table Visualizer usage) that is available. (The help will be shown in the Netscape Internet browser, if available.)

OTP Documentation Shows the Documentation about all OTP components that is available in the local installation of OTP.

1.2 Table Visualizer Release Notes

TV - Table Visualizer v2.0.0

Improvements and New Features

- It is now possible to create new ETS tables from within the Table Visualizer.
- It is now possible to start a trace of the process owning a table from within the Table Visualizer.
- Table information may now be viewed even for unreadable tables.
- The node handling is improved, i.e., both the node supervision and setting, and the handling of remote tables.
- Error messages is now (optionally) presented in Haiku style.
- It is now possible to open the OTP documentation available, using the 'Help' menu.
- Terms consisting of single process identifiers are now accepted when searching for objects and when editing objects.
- Menus in the previous main window are improved and redesigned.

Fixed Bugs and Malfunctions

- It is now possible to edit existing tables, provided they are writable. A record editor is also included, which makes it really easy to edit Mnesia tables.
  Own Id: OTP-2629
- The look and feel of the main window now resembles other tools.
  Own Id: OTP-2631
- Lists may now (optionally) be presented as strings.
  Own Id: OTP-2634
- Regular expressions may now be used when searching for objects.
  Own Id: OTP-2636
- All columns are automatically updated, even when previously not shown.
  Own Id: OTP-2637

Incompatibilities With Table Visualizer v1.2.4

- The main window is altered, meaning that the procedure to open a table is changed.
- The field showing the content of a row or single cell is now more loosely coupled to the row/cell. This is due to the fact that editing of the field may go on, making it undesirable to update the field every now and then.
- Menus in the previous main window are improved and redesigned.

Known Bugs and Problems

- DETS tables cannot be shown in TV.
  Own Id: OTP-2630
- Tables cannot be saved, neither can they be revived if they have crashed.
  Own Id: OTP-2630
- Test API is missing, as well as test suites.
  Own Id: OTP-2191

TV - Table Visualizer v1.2.4

Improvements and New Features

- It is now possible to start a new TV using the 'File' menu.
- In the "Open Table" window, Mnesia tables are now sorted alphabetically.

Fixed Bugs and Malfunctions

-

Incompatibilities With Table Visualizer v1.2.3

-

Known Bugs and Problems

- The on-line help isn't context sensitive, and could be extended.
  Own Id: OTP-1620, OTP-1667

TV - Table Visualizer v1.2.3

Improvements and New Features

-

Fixed Bugs and Malfunctions

- TV now handles ETS tables of type duplicate_bag.
  Own Id: OTP-2260

Incompatibilities With Table Visualizer v1.2.2

-

Known Bugs and Problems

- The on-line help isn't context sensitive, and could be extended.
  Own Id: OTP-1620, OTP-1667

TV - Table Visualizer v1.2.2

Improvements and New Features

- Tables residing on remote nodes may now be opened.
  Own Id: OTP-1940
- Objects containing a user specified Erlang term may now be searched for.
  Own Id: OTP-2065
- Rows are now enumerated, thereby improving navigation in the table.
  Own Id: OTP-2064
- The general graphical design has been revised and improved in some minor ways.

Fixed Bugs and Malfunctions

-

Incompatibilities With Table Visualizer v1.1.2

- New, changed and deleted objects are from now on only indicated with colours, not with letters.
- The design of the Open Table window has been altered, enabling remote nodes to be shown. As a result, unreadable tables are now only shown on request.

Known Bugs and Problems

- The on-line help isn't context sensitive, and could be extended.
  Own Id: OTP-1620, OTP-1667

TV - Table Visualizer v1.1.2

Improvements and New Features

- The Table Visualizer no longer crashes when the table polled has died. Instead a popup window informs the user about the no longer existing table.

Fixed Bugs and Malfunctions

- The Table Visualizer is now an application, i.e., a tv.app file has been created.
  Own Id: OTP-1689

Incompatibilities With Table Visualizer v1.0.2

-

Known Bugs and Problems

- The on-line help isn't context sensitive, and could be extended.
  Own Id: OTP-1620, OTP-1667
- Tables residing on remote nodes can currently not be loaded.
  Own Id: OTP-1940

Table Visualizer 1.0.2

New functionality:

Poll time supervision introduced, preventing frequent pollings to flood the system (in the case of very big tables).

Bug corrected: emptying a table and filling it once again doesn't make TV crash any more.

Table information now shown in a window of its own, with flaps grouping together related information.

The 'Open Table' dialog completely redesigned (new format of table identifiers, parent process shown, and so on).

Pid to main process returned when starting TV.

Letters introduced on the row buttons: 'N' for new elements, 'C' for changed elements, and 'D' for deleted elements.

Keyboard accelerators added for the menus.

Shortcuts added for the most frequently used menu options.

Table Visualizer 1.0.1

Fixed Bugs and malfunctions

- The usage/presentation of ets table identifiers has been changed due to changed (internal) representation in ets.
  Own Id: OTP-1393

Table Visualizer 1.0

The Table Visualizer is a new graphical application based on GS. It's purpose is to examine ETS and Mnesia tables in a passive way (i.e., elements in the tables viewed cannot be altered via the tool, and neither can new elements be inserted). For further information, see the *TV User's Guide*.

TV Reference Manual

Short Summaries

- Erlang Module tv [page 24] – The Table Visualizer graphically examines ETS and Mnesia tables.

tv

The following functions are exported:

- `start() -> Pid`
  [page 24] Starts the Table Visualizer.

tv (Module)

The Table Visualizer enables the user to examine ETS and Mnesia tables. Once a certain table has been opened in the tool, the content may be viewed at various levels of detail. The content viewed may also be sorted, using any element as key. The table may be polled anytime; either regularly at specified intervals, or manually. New and deleted objects, as well as those altered, are marked with characteristic colors.

Exports

```
start() -> Pid
```

Types:

- Pid = pid()

`start/0` starts the Table Visualizer.

See Also

For a complete description of the Table Visualizer, please see the TV User's Guide.

List of Figures

Chapter 1: TV User's Guide

Index

Modules are typed in *this way*.
Functions are typed in `this way`.