

Tutorial
Discussion Board



Table of contents

Introduction	3
Plan the Discussion Board	4
Post and List Messages	7
Display message threads	13
Building the recordset	13
Displaying messages	17
Controlling content with conditional regions	20
Trimming the page	22
Post message	26
Reply to users	35
Send reply notification	42
Other Resources	49
Copyright	50

Introduction

In this tutorial you will use **MX Kollection 3** to build a basic *Discussion Board* web application. The final application will allow users to:

- View a list of discussion topics.
- Read the messages associated with each topic.
- Post a new message to any topic, after registering an account.
- Reply to other messages.
- Subscribe to a discussion thread (a chain of messages that are replies to the original one) and receive email notifications when a new message was posted.

This tutorial contains three main sections

1. Planning the application, with database design and site structure.
2. Building the basic discussion board - the topic and message display and post / reply pages.
3. Improving the basic board by allowing subscriptions, reply notifications and filtering of bad words.

If you have the **MX Kollection 3** bundle installed, then you have all the necessary tools. Otherwise, the following separate products should be installed on your computer to complete the *Discussion Board* tutorial:

- MX Send E-mail
- MX Form Validation
- MX User Login
- MX Query Builder

The estimated completion time for this tutorial is about 40 - 60 minutes, depending on your authoring knowledge with **Macromedia Dreamweaver** (MX or MX 2004) and **MX Kollection 3**.

It's recommended that you follow the steps in their intended order as to avoid problems later on.

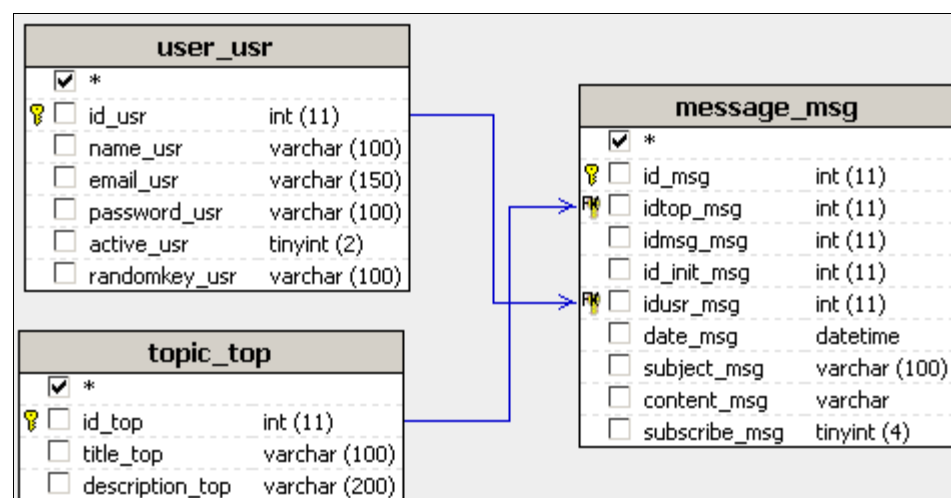
Plan the Discussion Board

Before you start building the discussion board, please make sure that:

1. You have a running database server and web server.
2. You have installed the required extensions for completing this tutorial:
The **MX Kollection 3** bundle, or, alternatively:
 - **MX Send E-mail**
 - **MX Form Validation**
 - **MX User Login**
 - **MX Query Builder**
3. You have a correctly defined **Dreamweaver** site.
4. You set up the database provided with this tutorial and create a connection to it.

For more instructions about setting up your site, consult the *Getting Started* help file that comes with **MX Kollection 3**. It can be accessed in **Dreamweaver** from **Help -> InterAKT -> Getting Started**.

The discussion board will store its information in a database having the following table and column structure:



Note: The database diagram in the image above was built with **MX Query Builder** (also referred as **QuB**) to better illustrate the database structure. You do not need to build it in order to complete this tutorial.

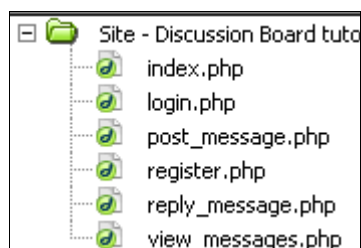
Here are the database tables and their columns:

1. *topic_top* - stores information regarding the main discussion topics:
 - **id_top** - the primary key for the topics table. No two topics can have the same ID.
 - **title_top** - the topic's title (e.g. Economy and Stock Markets).
 - **description_top** - a text description of the topic's subject. Will be displayed below the topic title
2. *message_msg* - stores all messages posted on the site, for all topics. The table's columns have the following meaning:
 - **id_msg** - the primary key for the messages table. Used to uniquely identify a message.

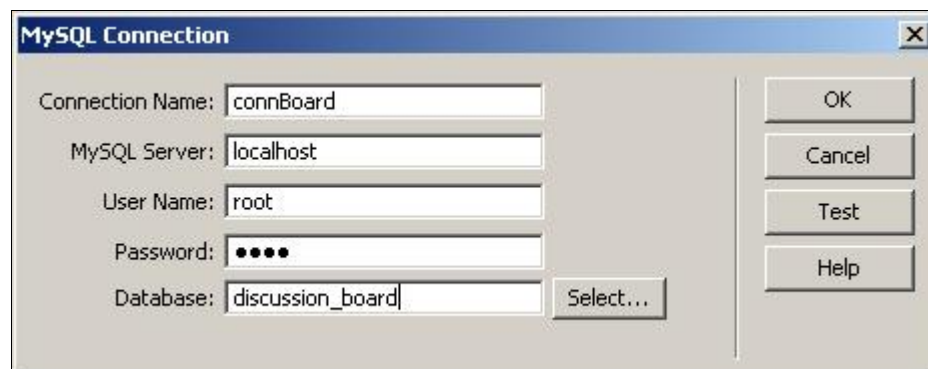
- ***idtop_msg*** - the foreign key that connects to the associated topic. Through this column, a message is marked as belonging to a topic.
 - ***idmsg_msg*** - self foreign key that stores the ID of the parent message. It is used when posting a reply to link the current message and the message that is being replied to.
 - ***id_init_msg*** - also a self foreign key used when the message is a reply. It is used to store the ID of the first message in the reply chain.
 - ***idusr_msg*** - foreign key to the user's table, stores the ID for the user that posted the message.
 - ***date_msg*** - the date when the message was posted.
 - ***subject_msg*** - the message subject. If a reply, it should be Re: and the old subject.
 - ***content_msg*** - the message's main text.
 - ***subscribe_msg*** - stores if the user is subscribed to this message thread or not (1/0).
3. *user_usr* - stores the user data: e-mail, password etc.
1. ***id_usr*** - the primary key for the user table. No two users can have the same ID.
 2. ***name_usr*** - the user account. Used to login on the site.
 3. ***email_usr*** - the email address of the user (john_smith@noplac.org).
 4. ***password_usr*** - stores the user's password.
 5. ***active_usr*** - a boolean variable (Y/N) based on whether the user account is active or not.
 6. ***randomkey_usr*** - stores a randomly generated code used for user activation.

To set up your application, follow these steps:

1. You can find the scripts needed to create an identical table structure inside the downloaded package, in the `\tutorials\Discussion Board\db\` folder, as an *sql* or *mdb* file, depending on the database server you intend to use. Execute the *sql* script in your database server management software (e.g. **PHPMyAdmin**, **Microsoft Access** etc.).
2. Open **Dreamweaver** and create the following folder and file structure for your site. You can create it easily by unpacking the *zip* file corresponding to your server model from `\tutorials\Discussion Board\` in your site root folder:



3. Open the *index* page and create a new connection named *connBoard* and configure it to connect to your newly created database.



Starting with the following topic, you will learn how to display and add messages and topics.

Post and List Messages

In this section you will build the basic discussion board functions - the ability to display, post, and reply to messages.

The first thing to do before creating any of the application's pages is to implement the user authentication system. In order to attract new members, the discussion board should be accessible and friendly, should present interest and should offer solutions. This way, viewing the messages will not require the visitor to have an account, but posting messages or replies will. The user authentication will use the *user_usr* table, as presented in the Planning section of this tutorial.

To build this section, follow the next steps:

1. Open the *register* page from the site root into **Dreamweaver**.
2. Open the InterAKT Control Panel and go to Login Settings. This is where you will have to define how user authentication will work on this particular application. The interfaces and settings are similar to the ones presented in the Job Site tutorial > User Authentication section, with the following differences:
 - Authentication is based on username and password only, without the use of user levels. Also, don't forget to check the encryption checkbox.
 - On the Database tab, configure all fields based on the existing columns, using the ***name_usr*** column as username, and no level column at all. Also, make sure you select the column *randomkey_usr* in the Random key menu. This is used in the account activation process, to prevent a user from activating the accounts of other users. The random key will be attached to the URL for account activation.
 - On the User Levels tab, you only need to fill in the main redirect pages (the ones on top of the user interface, without having to define pages for each level - levels don't exist).
 - The login page is *login*.
 - After successful authentication, users must be redirected to the *index* page.
 - If the login fails, users remain on the *login* page.
3. Once you've finished setting up the Control Panel options, press the OK buttons in each dialog box.
4. Back on the registration page, apply the User Registration Wizard, similar to what is shown on the User Authentication tutorial. Remember to remove the ***active_usr*** and ***randomkey_usr*** columns from the wizard's second step, and to check the ***Use account activation*** checkbox.
5. Once you're done with the wizard, close the page and upload it to the server. If you want, you can try it out and create an user account of your own.
6. When a user registers an account with the site, two e-mail messages will be sent to his e-mail address: an account activation message and a welcome message. To be able to send these messages, you need to configure InterAKT Control Panel > E-mail settings.
7. Next, you will create the login form, that allows registered users to authenticate to the forum. Open it in Dreamweaver and apply the Login Form Wizard to create the HTML form elements, as well as the activation and forgot password pages. An example of the configuration can be found here. Once you finish configuring and applying the wizard, save the login page and upload it on the remote server.
8. Finally, you should create a log out link, to allow users to safely exit the discussion board. You should follow the same steps as described here. Place the "Logout" link in the upper part of the *view_messages* page.

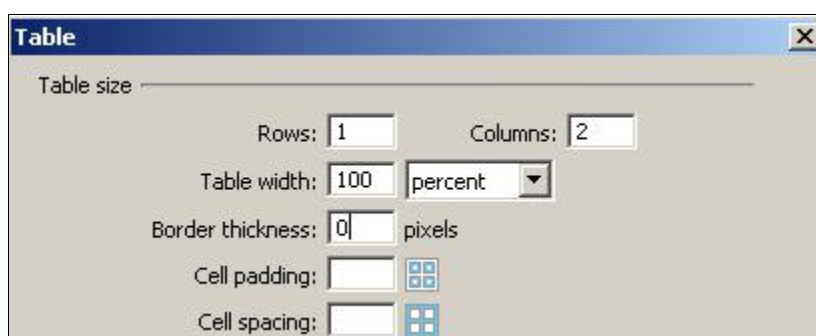
Once the user authentication section is complete, you will have to create the application's main page. This page will contain a list of all topics available on site, each with its description, and a link to show the associated messages. To keep things simple, the topic list will not be managed from the Discussion Board site. Adding new topics will have to be done in the database management software. Alternatively, you can create an administration section for the discussion board, as explained in another tutorial: Content Management System.

To create the application's main page, with the topic listing, follow the next steps:

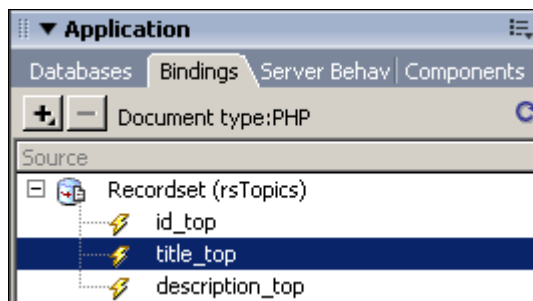
1. Open the *index* file from the site root in Dreamweaver.
2. The page design is already created, containing a table with three rows and one column. The topic listing will be placed in the middle row and will display the topic title, description and a link to view the messages.
3. Before placing the elements on the page, you have to create a recordset that will retrieve all topics from the *topic_top* table. To create a new recordset, click the **Plus (+)** button of the **Bindings tab** and select **Recordset (query)**.
4. In the dialog box that opens, select the *connBoard* database connection and the ***topic_top*** table. Since you do not need to filter out any elements, you can click the OK button to close the dialog box.



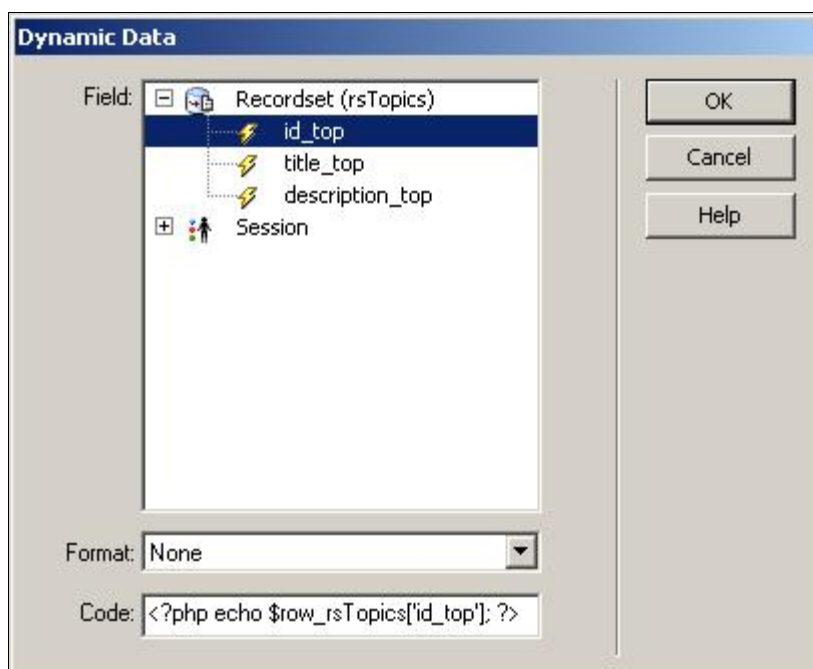
5. To display the topics in an orderly fashion, you will have to create a table containing a single row and two columns inside the main table's middle row.



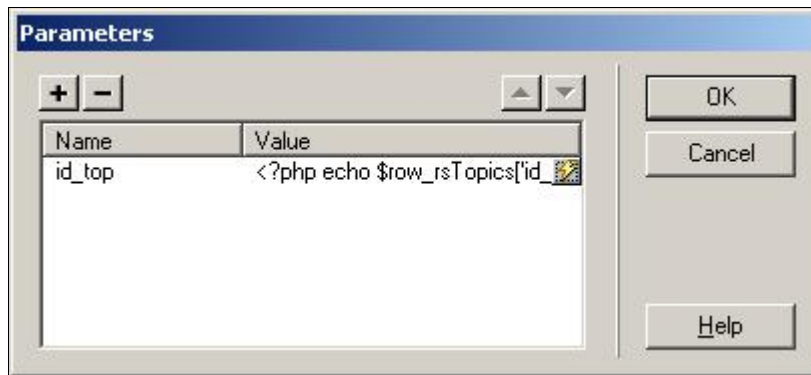
6. In the first column, the topic title and description will be displayed. Drag and drop their corresponding recordset fields from the Bindings tab to the page.



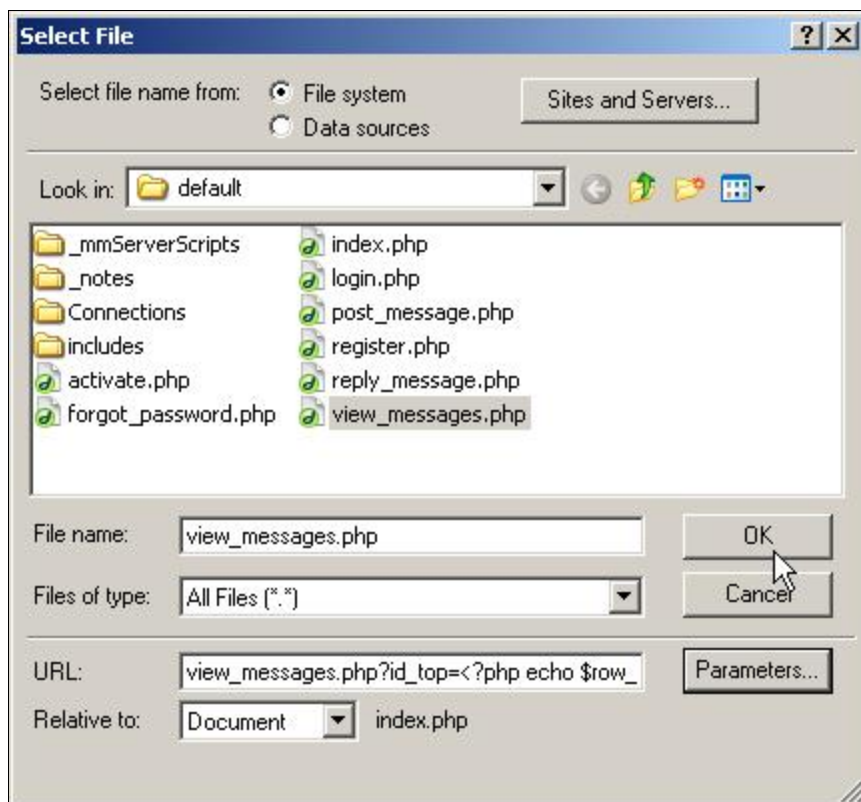
7. In the second cell, you have to create a link to the page that displays the messages: *view_messages*. Enter the link text **[View messages]** into the second cell. Select it and right click to display the menu. Choose **Make link**.
8. In the dialog box that opens, select the link's target page (e.g. *view_messages.php*) and then click on the **Parameters** button. This is needed, because the *view_messages* page will display all messages belonging to a single topic, and the topic's ID is passed as an URL parameter, to sort out only that topic's messages.
9. In the Parameters dialog box, enter the URL parameter's name: **id_top**. For the parameter's value, select the dynamic value *id_top* from the *rsTopics* recordset created earlier, by clicking the lightning bolt icon.



10. The Parameters dialog box should look like this. Click OK to confirm your settings.

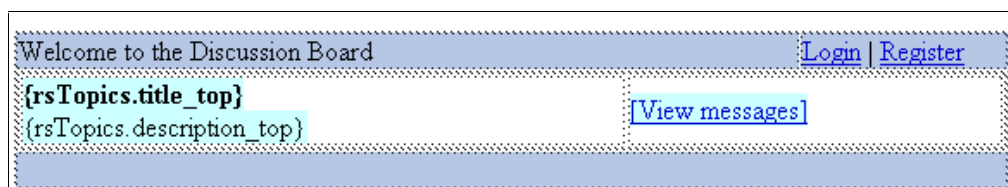


11. The dialog box for configuring the link to the view_messages page should look like this:



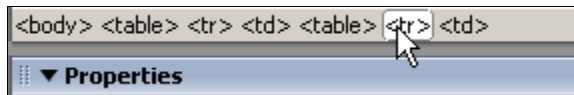
Click OK to apply your settings.

12. You might also want to apply a different formatting style to the topic title, to differentiate it from the rest of the text. At this point, your page should look like this in Dreamweaver Design view:

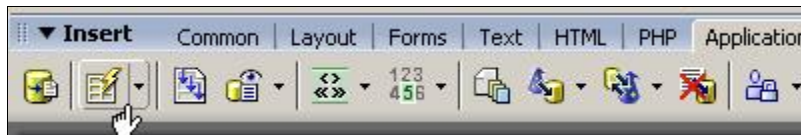


However, if you try to preview this page in the browser, you will notice that only the first topic is displayed. In order to display all topics, you must apply a Repeated region on the

table row containing the topic title and description. Click on the `<tr>` tag in the Tag selector to select that row:



Apply the **Repeated region** command from the **Application tab** of the **Insert** bar.



- Configure the dialog box to display all records of the `rsTopics` recordset and click the **Ok** button to close the dialog box.



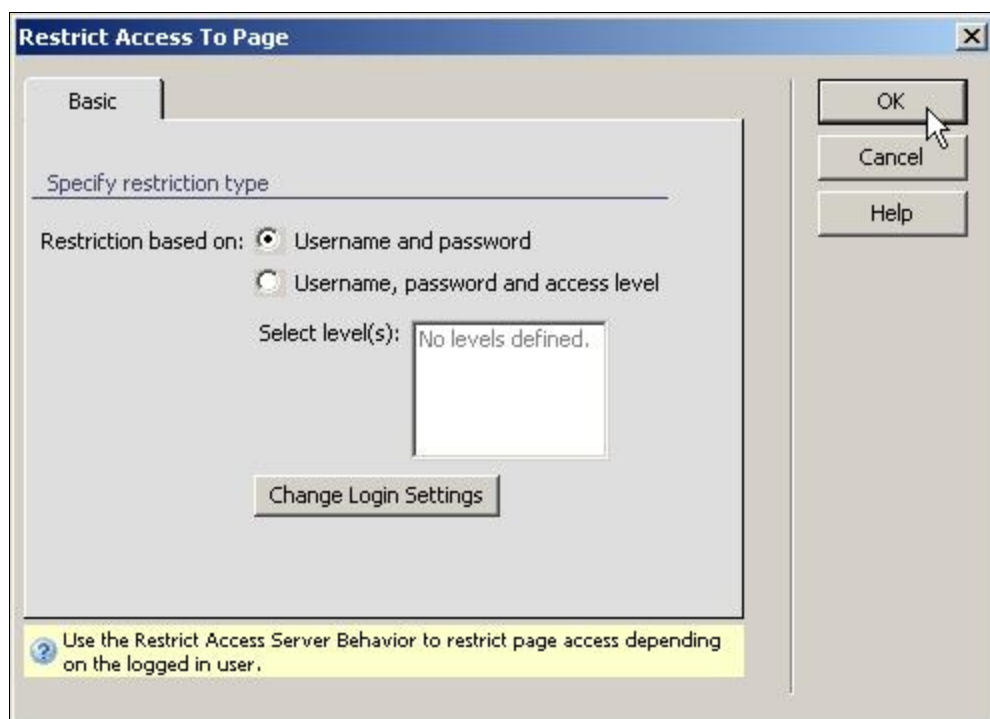
- Save the page and upload it to the remote server. When you preview it in the browser, it will display the list of topics from the database script delivered with this tutorial.



You have now completed the home page of the Discussion Board, which displays a list of topics. Next, you will create pages that allow viewing and posting messages in a topic.

Before creating the message viewing page, open the `post_message` and `reply_message` pages and apply the **Restrict Access to Page** server behavior on each of them. This will prevent unregistered users from adding messages to the Board. This server behavior can be accessed from the **Server**

Behaviors tab > + > **MX Kollection** > **User Login**. Configure the server behavior to restrict access based on username and password only, as shown in the image below:



After configuring the server behavior on both pages, you can move on and create the page that allows users to view messages associated to a certain topic.

Display message threads

In this topic, you will learn how to create the page that displays all messages associated to a specific topic, as well as options to Post a message and to Reply to a message. Since this page only displays messages, and does not allow users to post new ones, you do not need to apply a **Restrict Access to Page** server behavior. The page must remain public.

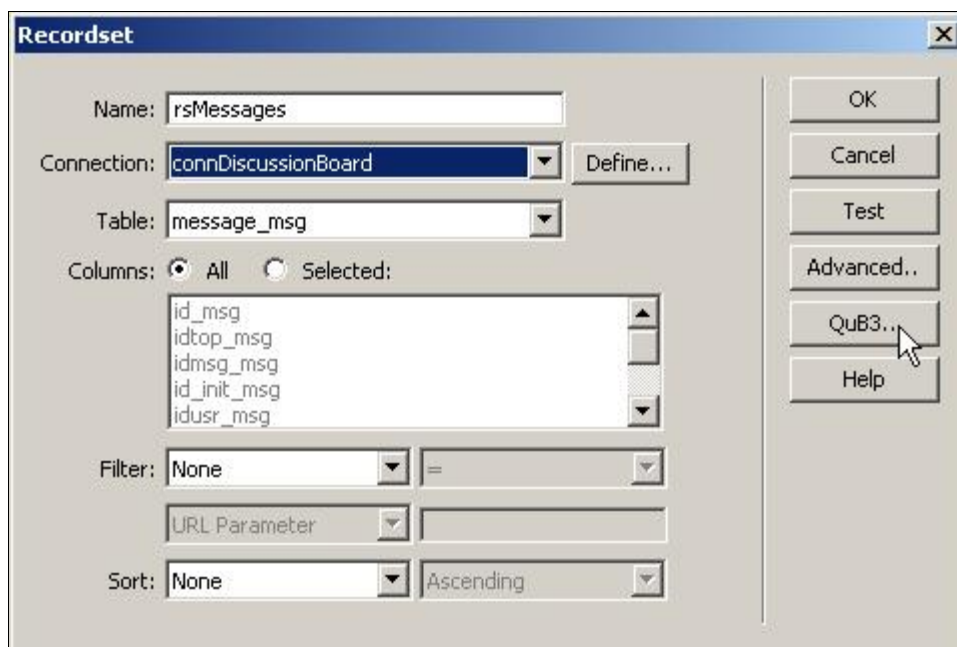
Building the recordset

To create this page, follow the next steps:

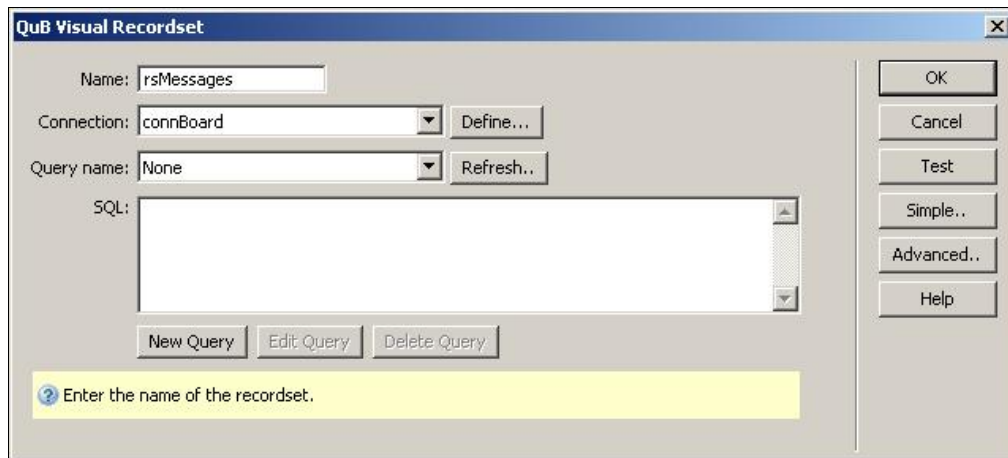
1. Open the *view_messages* page in Dreamweaver. The *view_messages* page receives from the main page an URL parameter containing the topic ID, which will be used to sort out the messages belonging to the selected topic.
2. Before creating any page elements, you need to decide what will be shown for each message:
 - the name of the user that posted it
 - the message subject
 - the date when the message was posted
 - the message text

All messages displayed on the page must belong to the same topic – the one whose ID is passed as URL parameter to the page.

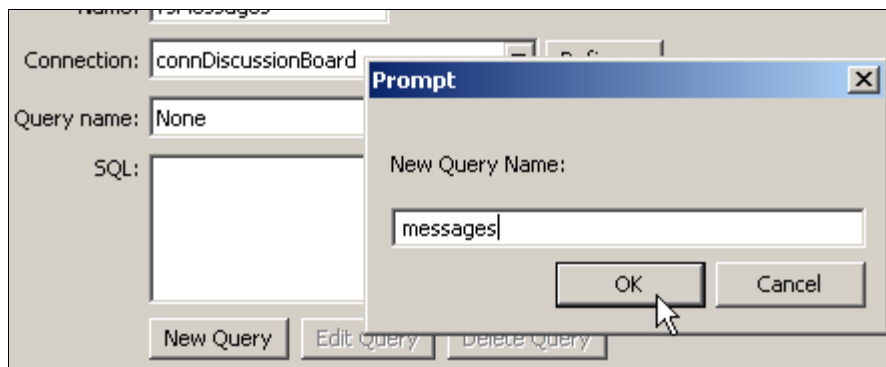
3. The recordset to retrieve the elements is filtered, and takes data from more than one table. The easiest way to create the complex query is to use **MX Query Builder** (or simply QuB).
4. To open the MX Query Builder interface, simply open the Recordset dialog box, by clicking the + button from the **Server Behaviors** tab, and selecting **Recordset**. Enter rsMessages for the recordset name, select the database connection, then click the button labeled **QuB3**, displayed on the right of the interface.



After clicking the QuB3 button, the **QuB Visual Recordset** dialog box opens:



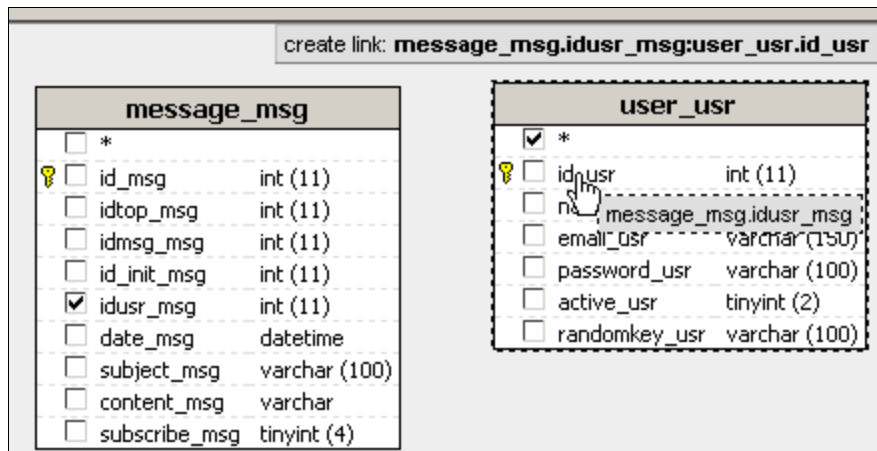
- You need to create a new query. Click on the **New Query** button to start the MX Query Builder web interface, where you will visually build the SQL query needed to extract the information for this page. Name the query **messages**.



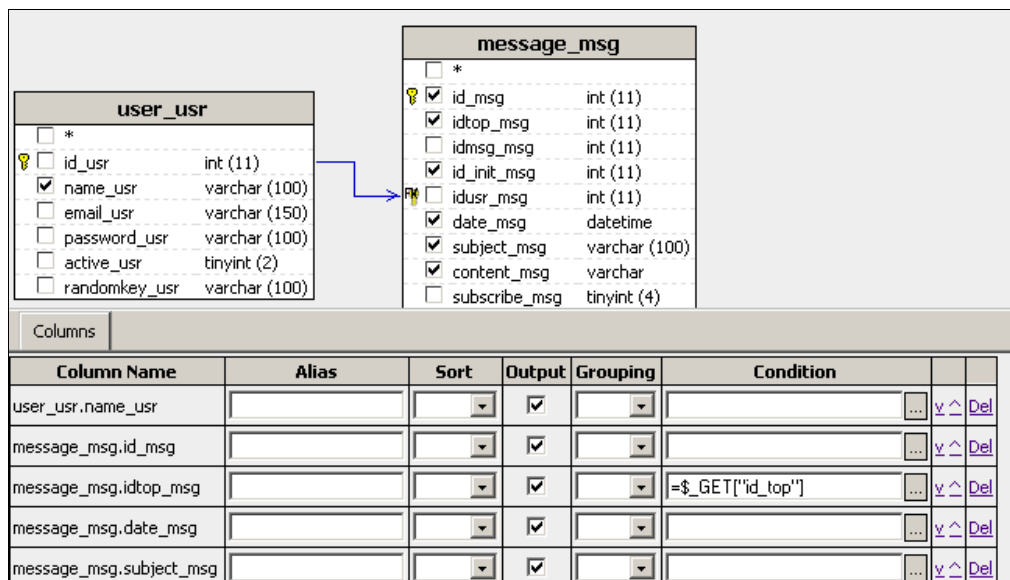
- Once the user interface opens, you have to select the tables to use in the query. Simply click on the user_usr and message_msg tables in the **Tables** panel and they will appear in the main area as well.



- In order to extract information from both tables, a relationship (also known as JOIN) must be defined between the two tables. To define the relationship, simply drag and drop the idusr_msg field from the messages_msg table onto the id_usr field of the user_usr table. A blue line will indicate the relationship.

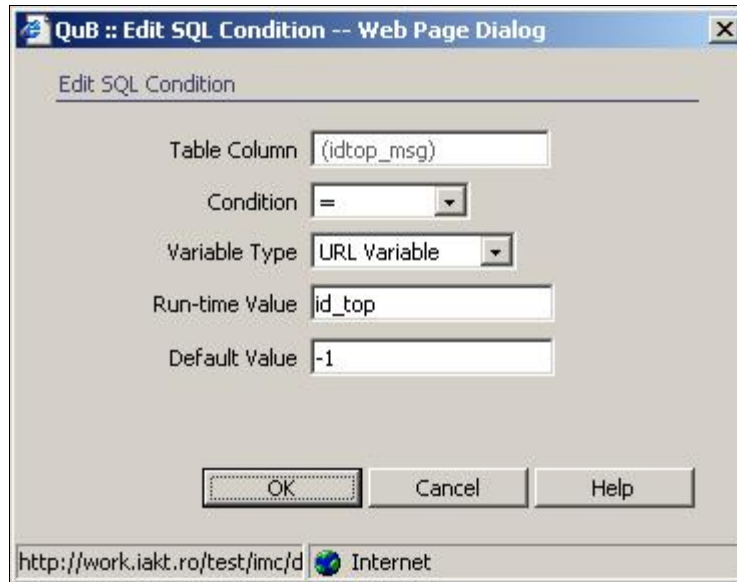


8. To decide which columns to use in the query, simply click on their names. Select the following columns:
 - from the user_usr table, you only need name_usr.
 - from the messages_msg: id_msg, idtop_msg, id_init_msg, date_msg, subject_msg and content_msg .
9. As you click on each column name, it is checked, and it also appears in the **Query Management** panel. This is where you can set aliases, grouping and sorting rules for each column, or define conditions.



10. All messages retrieved by the recordset must belong to a specific topic, referenced by its ID, which is received via an URL parameter from the index page. Therefore, you need to define a condition on the column **idtop_msg** (which stores the foreign key to the topic_top table). Click the button next to the Condition text field corresponding to the **idtop_msg** column.
11. A new dialog box opens where you can define the condition. Configure it as follows:
 - Set the Condition operator to equal (=).
 - Set the Variable Type to URL Variable.

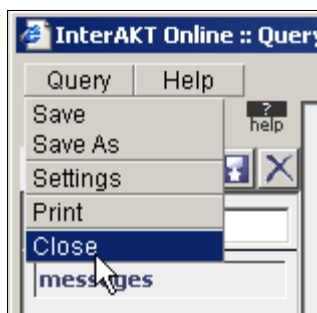
- In the Run-time Value, enter the URL parameter's plain name (id_top). You do not need to enter any code, as QuB will automatically generate the correct semantics.
- In the **Default Value**, enter a numeric value that does not match any existing record. Usually, a negative value is used.



12. After defining the condition, the query is complete. Click on the save icon in to top left corner:



13. Close the QuB web interface, by selecting Close from the **Query** menu.



This will take you back to the **Dreamweaver** interface.

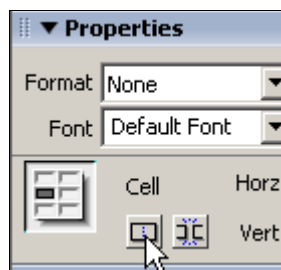
14. To display the newly created query, click the **Refresh** button next to the **Query name** drop-down. The SQL code generated by QuB should appear in the **SQL** text area:



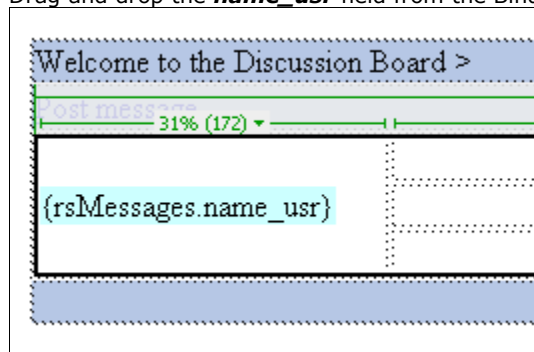
- To complete the recordset creation, click on the **OK** button. This will close the dialog box and place the new *rsMessage* recordset in the **Bindings tab** of the **Application panel**.

Displaying messages

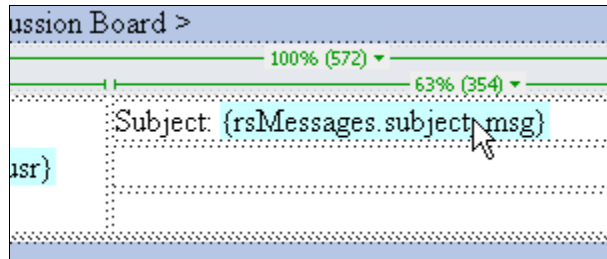
- Now that you have all required columns in the recordset, move on and add them to the page. Inside the middle row of the main table, create another table containing three rows and three columns, as you have done for the home page. Alternatively, you can use CSS styles to define a different layout for your page. The message details will be displayed as follows:
 - In the first column (which will span downward over the three rows) the author name will be displayed. Merge the first column's three cells by selecting them and clicking the Merge cells icon on the Property Inspector:



- Drag and drop the *name_usr* field from the Bindings tab in the merged cell.



- The first row's second cell will display the message subject. Type Subject:, then drag and drop the *subject_msg* recordset column after it.



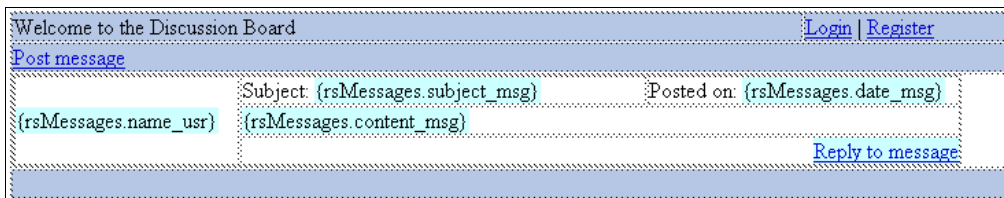
- On the third column of the first row, add the date when the message was posted. Type Posted on and drag and drop the **date_msg** field from the Bindings tab next to it.
- Merge the second row's last two columns, and drag and drop the **content_msg** recordset field inside the merged cell.
- Finally, merge the third row's last two columns, and type **Reply to message**. Select the text, right-click it, then select the **Make Link** option. As target, select the *reply_message* page. Click the **Parameters** button, to add three URL parameters, that will be passed to the *reply_message* page:
 - i. **id_top** - gets its value from the *idtop_msg* field of the **rsMessages** recordset, and is used to indicate the topic to which the reply belongs.
 - ii. **id_msg** - gets its value from the *id_msg* field of the **rsMessages** recordset, and is used to indicate the message to which the reply is posted.
 - iii. **id_init_msg** gets its value from the *id_init_msg* field of the **rsMessages** recordset, and is used to indicate the message that initiated the thread (the first message of the topic).



2. To improve the appearance of the page, align the link horizontally to the right, by selecting the text and then choosing Right from the Horz menu in the Property Inspector:



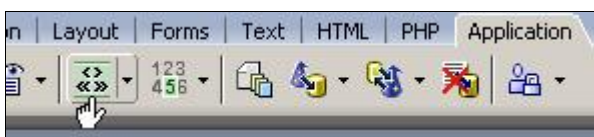
At this point, the *view_messages* page should look as follows in Dreamweaver Design View:



- As you can see, all elements that need to be displayed for a message are in place. Now, you only need to repeat the information for each message retrieved by the recordset. This is done as in the main page, by using a Repeated region. Select the table containing the message information (not the page table) and apply the **Repeated Region** command from the **Application** tab of the Insert panel. In the dialog box that opens, select the *rsMessages* recordset as source, and set a number of records to display on page (10 is just fine). This is needed as there can be many messages inside a single topic, and this allows a better view.



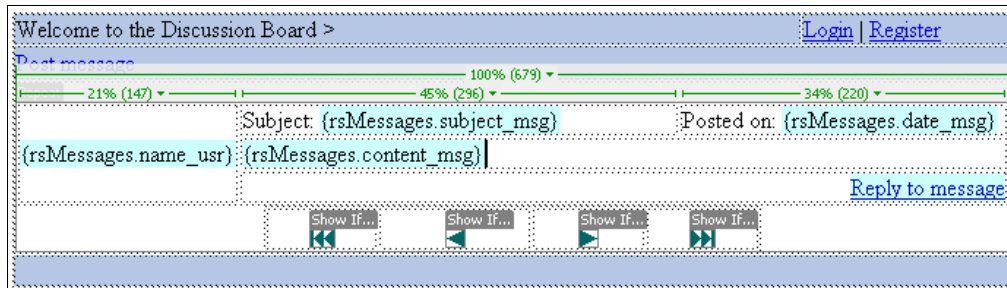
- The next step is to create the navigation elements that will allow users to browse through message pages. Without a navigation bar, visitors of your site will always see just the first 10 messages of each topic, without any possibility of seeing the rest. Place your mouse cursor immediately below the repeated region containing the message, then click the **Recordset Navigation Bar** command from the **Application** tab of the **Insert bar**:



Configure it to use the *rsMessages* recordset (like the Repeated Region). Also set it to use images as navigation buttons.



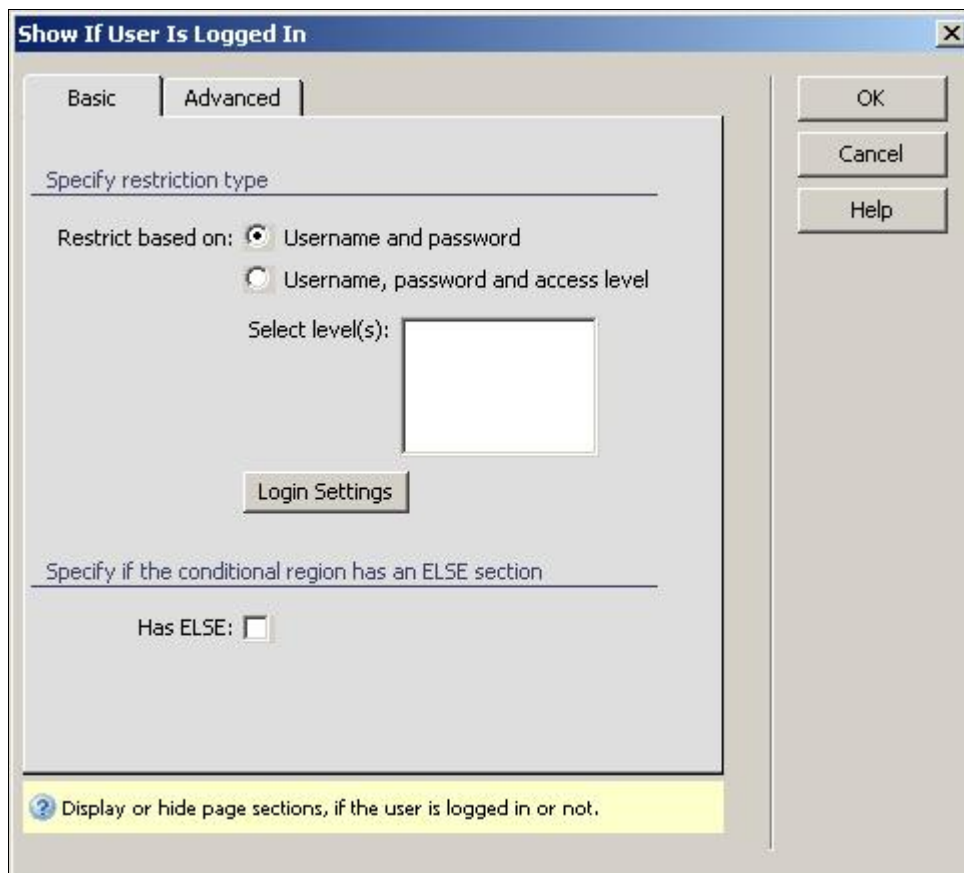
After you click OK, your page should look like this in Dreamweaver Design View:



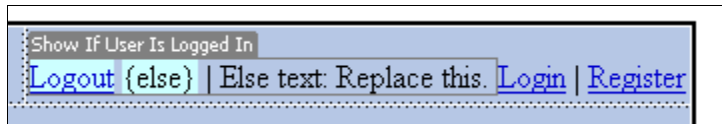
- Now all the elements necessary to display messages associated with a topic are in place, and the page is fully functional. It even contains links to post a new message, or to reply to an existing message. Posting and replying are however allowed only for registered and logged in users. Since the *post_message* and *reply_message* pages contain **Restrict Access to Page** server behaviors, visitors accessing these pages by clicking the links in the *view_message* page will be automatically redirected to the login page, where they will have to authenticate.

Controlling content with conditional regions

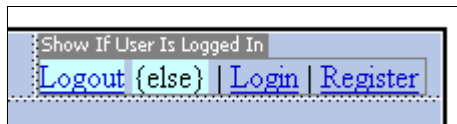
- Rather than redirecting the visitors, it would be better to show the links only when a user is logged in. **MX Kollektion 3** has a conditional region for this task called **Show If User Is Logged In**. Select each of these links and apply the server behavior from the **Server Behaviors Tab > + > MX Kollektion > Conditional Regions**. Configure the server behavior to use only the username and password for authentication:



2. Next, apply the same conditional region (Show If User Is Logged In) on the "Post message" link and on the "Reply to message" link.
3. The same thing must be done for the "Logout" link. This link should be visible only to those users who have already authenticated to the site, while the "Register" and "Login" links should be hidden. Obviously, a logged in user does not need to create an account or login again. In order to obtain this behavior, you'll have to:
Select the "Logout" link and apply the "Show If User Is Logged In" conditional region on it. This time, check the **Has ELSE** checkbox. You'll use the ELSE code block to display the remaining links (Register and Login) when the condition is not fulfilled, i.e. the user is not logged in. After you apply the conditional region on the "Logout" link, you'll notice a gray border surrounding it:



You need to replace the "Else text. Replace this." with the two links that remain outside the conditional region. Your page should look like this in the end:



When a user that is not logged in views messages inside a topic, each message will look as follows (it will not display the post and reply links):

Welcome to the Discussion Board		Login Register
John Doe	Subject: The stock market - an opportunity?	Posted on: 2005-05-17 00:00:00
<p>A friend recently suggested that the stock market might be a good opportunity. I tend to agree because the East European economy is expected to grow a lot in the next few years due to the EU accession and on the average stocks will grow more there than they will in the US. Right now, I don't know much about the stock market in Eastern Europe. There is an opportunity and there is lots of risk, but a question remains. What is the potential of the East European stock market?</p>		

For authenticated users, the "Log out", "Post message" and "Reply to message" links are available:

Welcome to the Discussion Board		Logout
Post message		
John Doe	Subject: The stock market - an opportunity?	Posted on: 2005-05-17 00:00:00
<p>A friend recently suggested that the stock market might be a good opportunity. I tend to agree because the East European economy is expected to grow a lot in the next few years due to the EU accession and on the average stocks will grow more there than they will in the US. Right now, I don't know much about the stock market in Eastern Europe. There is an opportunity and there is lots of risk, but a question remains. What is the potential of the East European stock market?</p>		
Reply to message		

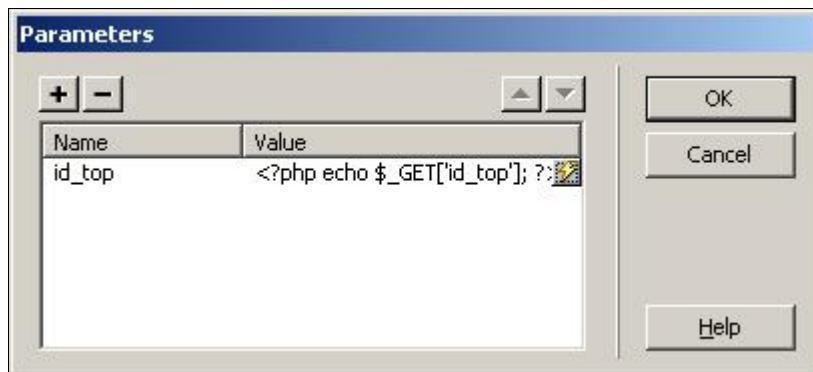
Trimming the page

You have to add an URL parameter to the **Post message** link, to make pass the topic ID to the `post_message` page. You will need the topic ID in that page in order to know under what topic the new message falls. Name the URL parameter `id_top` and set its value to the topic ID, which is already received as an URL parameter in the `view_messages` page. To add the parameter, select the "Post message" text and right-click on it. From the pop-up menu, select the Change Link option, then click the Parameter button in the dialog box that opens. Enter the new parameter's name (`id_top`), and in the value box, enter the code that will retrieve the `id_top` URL parameter. This code varies depending on the particular server model you are using:

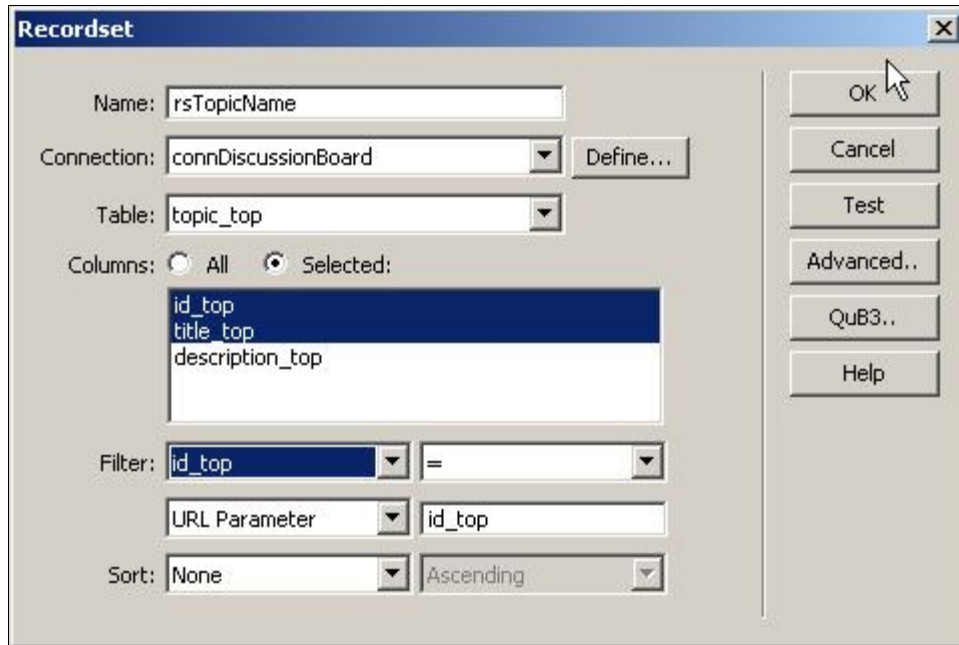
For **PHP_MySQL** and **PHP_ADODB**, write: `<?php echo $_GET['id_top']; ?>`

For **ColdFusion**, write: `<cfoutput>#URL.id_top#</cfoutput>`

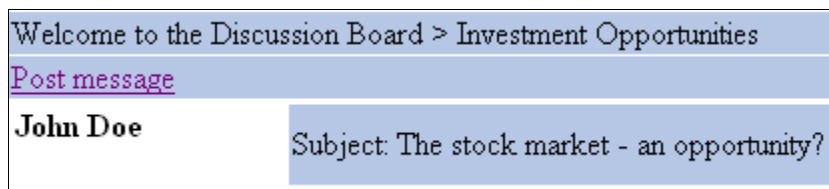
For **ASP_VBScript**, write: `<% = Request.QueryString("id_top") %>`



This page's functionality is now complete. There are some things left to be done, that only improve the overall look of the page, such as displaying the current topic name in the page. To accomplish this, you simply need to create a filtered recordset (`rsTopics`), that retrieves all records from the `topic_top` table with the same ID as the one passed as an URL parameter.



Once the recordset is created, simply drag the **title_top** recordset field from the Bindings tab and drop it onto the page. Here is how the page should in the browser:

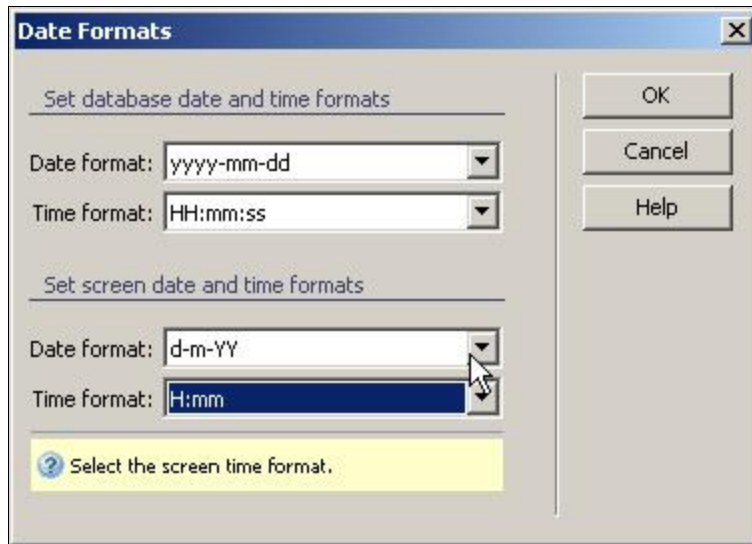


An improvement to the message viewing page is to transform the *Welcome to the Discussion Board* text into a link that will point to the site's main page (the home page displaying the list of topics).

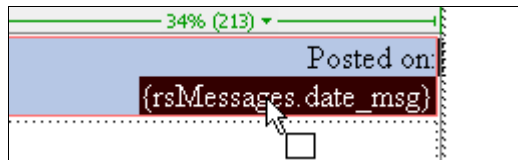
Finally, you might want to change the date format for displaying the date when messages were posted. It's not very user friendly to see dates such as 2005-05-17 00:00:00. You can change the way dates are displayed from the InterAKT Control Panel > Date formats.



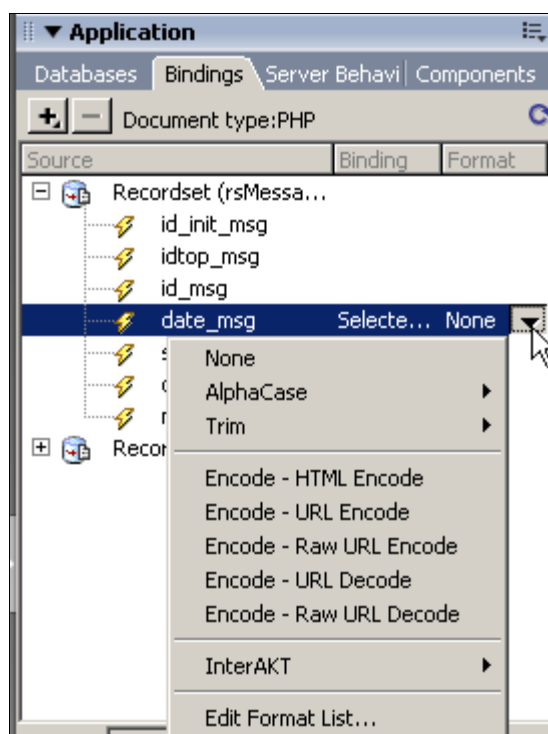
In the displayed dialog box, select the screen date and time format of your choice:



However, before you can see the changes, there's one extra step you need to do. The selected format applies only to code generated by MX Kollection. To apply it to the date you grabbed from the rsMessages recordset, select the {rsMessages.date_msg} dynamic text from your page:



Then go to the **Bindings** tab and click the arrow next to the **date_msg** field:



From the displayed menu, select **InterAKT > Format Date**:



Click OK and load the page in the browser to see how it looks.

Since the message viewing page is completed, save and close it. Then move on to the next section, where you will learn how to create the page that allows users post new messages.

Post message

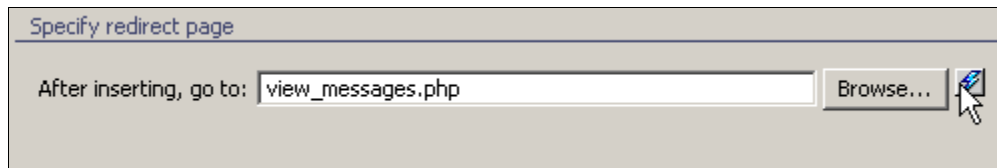
In this section of the tutorial, you will learn how to build the page that allows users post a new message in a certain topic. The page is accessed through the "Post message" link on the *view_messages* page, and passes to the *post_message* page the **id_top** URL parameter. This parameter contains the identifier of the topic to which the message will belong.

The value of the topic ID will be stored in the **idtop_msg** column of the **message_msg** table.

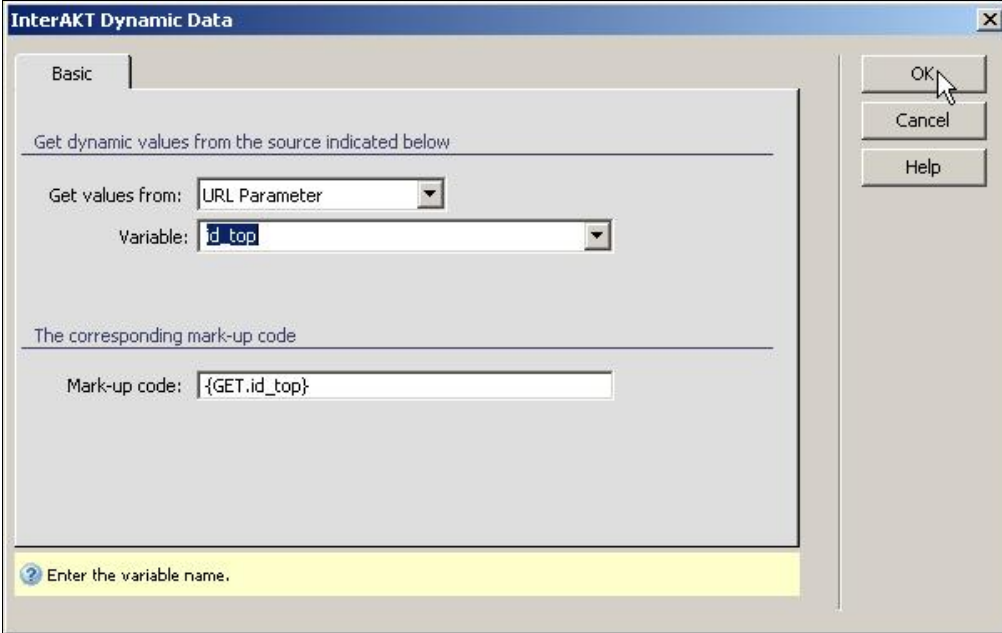
To allow users to post a message, you need to create an insert form, that takes user input and adds it to the *message_msg* database table. To build the page, follow the next steps:

1. Open the *post_message* page in Dreamweaver.
2. The layout is already created, containing the table with header and footer. You should first add the topic title to the table header, as shown in the previous topic in one of the last paragraphs.
3. Next, click in the table's middle row. This is where you will place the actual form that takes user input and adds the message to the database. Launch **Insert Record Form Wizard**, from the **MX Kollection tab** of the **Insert bar**. The wizard will generate the form and the insert transaction for you. Configure it as described next.
4. In the first step, select the database connection you've created at the beginning of this tutorial (*connBoard*), the table to insert into (**message_msg**). After posting the message, users should be redirected to the *view_messages* page, and the ID of the topic should be passed to it as an URL parameter. Otherwise, the page will be blank. To add the URL parameter to the page, without actually writing the code yourself, you will use the InterAKT Dynamic Data. After you select the *view_messages* page in the text field labeled "After inserting, go to". Next, type the following text after the page name:
`?id_top=`

Finally, click the blue lightning bolt icon:



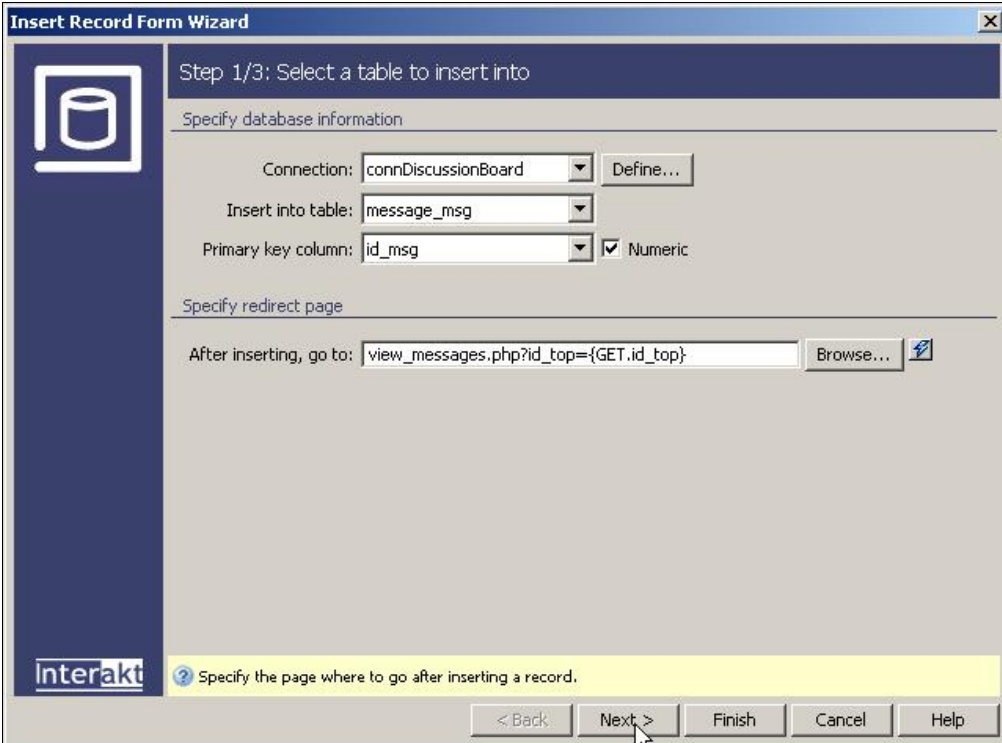
The InterAKT Dynamic Data window opens, where you need select the type of variable, and enter its name as shown below:



The dialog box is titled "InterAKT Dynamic Data" and has a "Basic" tab. It contains the following elements:

- A section titled "Get dynamic values from the source indicated below" with a dropdown menu set to "URL Parameter" and a text field containing "id_top".
- A section titled "The corresponding mark-up code" with a text field containing "{GET.id_top}".
- Buttons for "OK", "Cancel", and "Help" on the right side.
- A yellow status bar at the bottom with a question mark icon and the text "Enter the variable name."

Click **OK** to return to the first step of the **Insert Record Form Wizard**, which should look like in the following image:



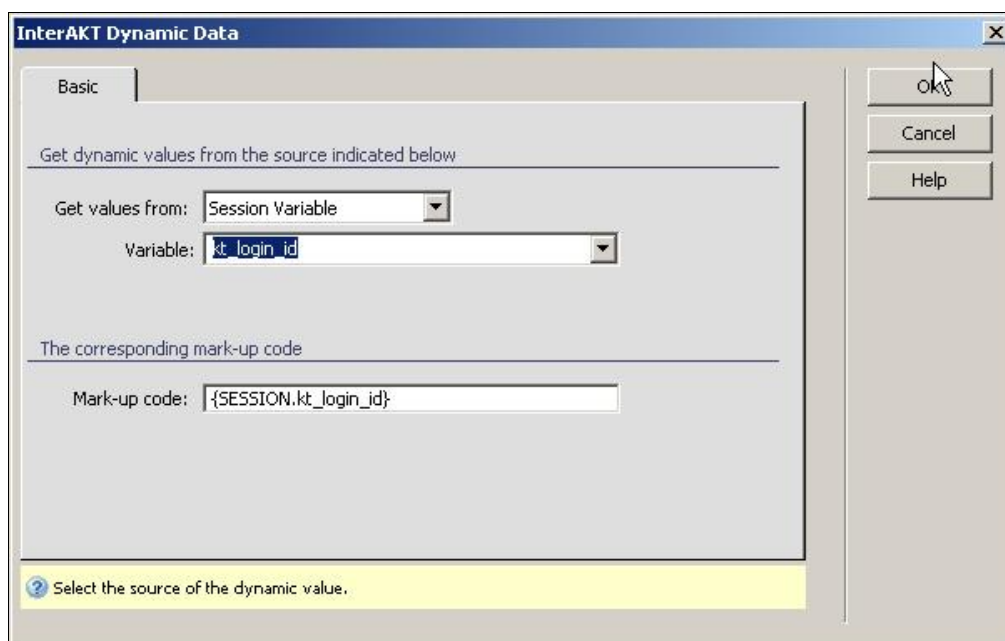
The wizard is titled "Insert Record Form Wizard" and is on "Step 1/3: Select a table to insert into". It contains the following elements:

- A database icon on the left side.
- A section titled "Specify database information" with a "Connection:" dropdown set to "connDiscussionBoard" and a "Define..." button.
- An "Insert into table:" dropdown set to "message_msg".
- A "Primary key column:" dropdown set to "id_msg" with a checked "Numeric" checkbox.
- A section titled "Specify redirect page" with a text field containing "view_messages.php?id_top={GET.id_top}" and a "Browse..." button.
- Buttons for "< Back", "Next >", "Finish", "Cancel", and "Help" at the bottom.
- A yellow status bar at the bottom with a question mark icon and the text "Specify the page where to go after inserting a record."

- In the second step, you have to configure what table columns to use, and what values to assign to each:
 - Remove the **idmsg_msg** and **id_init_msg** from the field list. These fields are not needed at this time. The **idmsg_msg** is only needed when replying (which is not currently the case), and the **id_init_msg** field will receive its value after the transaction.
 - Set the **idtop_msg** field to be displayed as Text, and set its Default Value to be equal to the URL parameter named **id_top**. To do that, click the InterAKT Dynamic Data icon, then in

the dialog box that opens, in the **Get values from** drop-down menu choose the **URL Parameter** option and enter `id_top` in the **Variable** field.

- The **idusr_msg** field must be set as Text, with the default value equal to the session variable **kt_login_id**. Click the blue lightning bolt icon to access the **InterAKT Dynamic Data** window and select the session variable `kt_login_id`, as seen in the image below:



- The date should also get a value automatically, in order to store the date and time when the message was posted. Set the field to display as text also, and for the default value, use the **{NOW_DT}** mark-up code. This InterAKT Mark-up will be replaced at runtime with the current date and time.

Note: If you are using a **Microsoft Access** database, in the **Submit as** drop-down menu, for the `date_msg` column, another option will be available: **Date MS Access**. Select this option when submitting the date.

- The message **Subject** will remain a text-field, but the **Content** should be set as a text area.
- The **subscribe** column is stored as a 1 or 0 value. Therefore, you should display it as a checkbox and submit it as a **Checkbox: 1,0**. Step 2 of the wizard should be configured as in the following image:

Step 2/3: Configure the form fields

Set form fields properties

Form fields: + -

Column	Label	Display as	Submit as
idtop_msg	Idtop:	Text	Numeric
idusr_msg	Idusr:	Text	Numeric
date_msg	Date:	Text	Date
subject_msg	Subject:	Text field	Text
content_msg	Content:	Text area	Text
subscribe_msg	Subscribe:	Check box	Checkbox: 1,0

Label:

Display as:

Submit as:

Initial state: Checked Unchecked

Interakt Build dynamic forms to insert records in your database tables.

< Back Next > Finish Cancel Help

Move to the next step, by pressing the **Next** button.

- The third step requires that you have **MX Form Validation** installed (if you do not have the **MX Kollection** bundle or the **ImpAKT** bundle). Otherwise, step 3 will not appear at all. This is where you can define validation rules for any of the fields involved in the transaction. It is a good idea to make the Content field required. Otherwise, users might be able to post empty messages:

Step 3/3: Enter the validation rules for the selected fields

Specify the validation rules for each form field

Form fields:

Column	Submit as	Required	Format
idtop_msg	Numeric	No	No Validation
idusr_msg	Numeric	No	No Validation
date_msg	Date	No	No Validation
subject_msg	Text	No	No Validation
content_msg	Text	Yes	No Validation
subscribe_msg	Checkbox: 1,0	No	N/A

Required:

Validation format:

Min char: Max char:

Check below if you want to overwrite the default error message

Custom message:

Error message:

Interakt The list of fields associated with the Insert Record transaction, and the associated validation options.

< Back Next > Finish Cancel Help

7. To apply the actions and dismiss the wizard window, click the **Finish** button.
8. In Dreamweaver, the HTML form will be displayed in the table's second row, and the necessary server behaviors will be added to the **Server Behaviors** tab. If you need to edit any of them (the Insert transaction, or the Validation options), simply double-click their names in the tab and the dialog box will open.
9. Since the *idtop_msg*, *idusr_msg*, and *date_msg* should not appear in the HTML form, select these three fields in the **Dreamweaver** page and press the **Delete** key. They are not removed from the **Insert Transaction**, but only from the HTML form. The idea is to have their values set automatically by the transaction, but not to make them visible to users.

At this point you can already add new messages to the board, in any topic that you want. However, the reply does not work properly at this point. This happens because the first message in the thread must pass its ID to all of the reply messages in the same thread. This ID is stored in the *id_init_msg* column of each message. For new messages (i.e., messages that are not replies to existing ones), the ID of the initial message must be identical to the value of the primary key.

The *id_init_msg* field was initially removed from the insert transaction, when you applied the Insert Record Form Wizard. It must have the same value as the ID of the newly inserted message. However, the primary key of the new message is not available until the message has been actually inserted. Therefore, you will have to create a custom trigger that retrieves the ID of the message that has just been inserted, then updates the message by adding the same value for the *id_init_msg* field.

To create the **Custom trigger** that will update the value of the *id_init_msg* with the value of the primary key, follow the next steps:

1. Access the **Custom Trigger** server behavior from the **Server Behaviors** tab > **++ MX Kollection > Forms**.
2. In the dialog box that opens, you have two tabs to configure:
 - **The Basic tab** - where you will write the trigger code.
 - **The Advanced tab**, where you will set the trigger properties (type, priority, etc)
3. In the **Basic tab**, enter the following code (select the one that matches your particular server model):

- For **PHP_MySQL**:

```
$pkval = $tNG->getColumnValue('id_msg');
$query_update_initmsgid = "UPDATE message_msg SET id_init_msg =
".$pkval." WHERE id_msg = ".$pkval;
$result_update = mysql_query($query_update_initmsgid);
```

- For **PHP_ADODB**:

```
$pkval = $tNG->getColumnValue('id_msg');
$query_update_initmsgid = "UPDATE message_msg SET id_init_msg =
".$pkval." WHERE id_msg = ".$pkval;
$result_update = $tNG->connection->execute($query_update_initmsgid);
```

- For **ASP VBScript**

```
pkval = tNG.getColumnValue("id_msg")
query_update_initmsgid = "UPDATE message_msg SET id_init_msg =
"&pkval&" WHERE id_msg = " & pkval
Set result_update = tNG.connection.Execute(query_update_initmsgid)
```

- For **ColdFusion**, the approach is a little different. Besides adding code in the **Custom Trigger** user interface, you will need to modify some of the code already generated so far, because of some technical limitations concerning ColdFusion.

The code to add in the **Custom Trigger** textarea is:

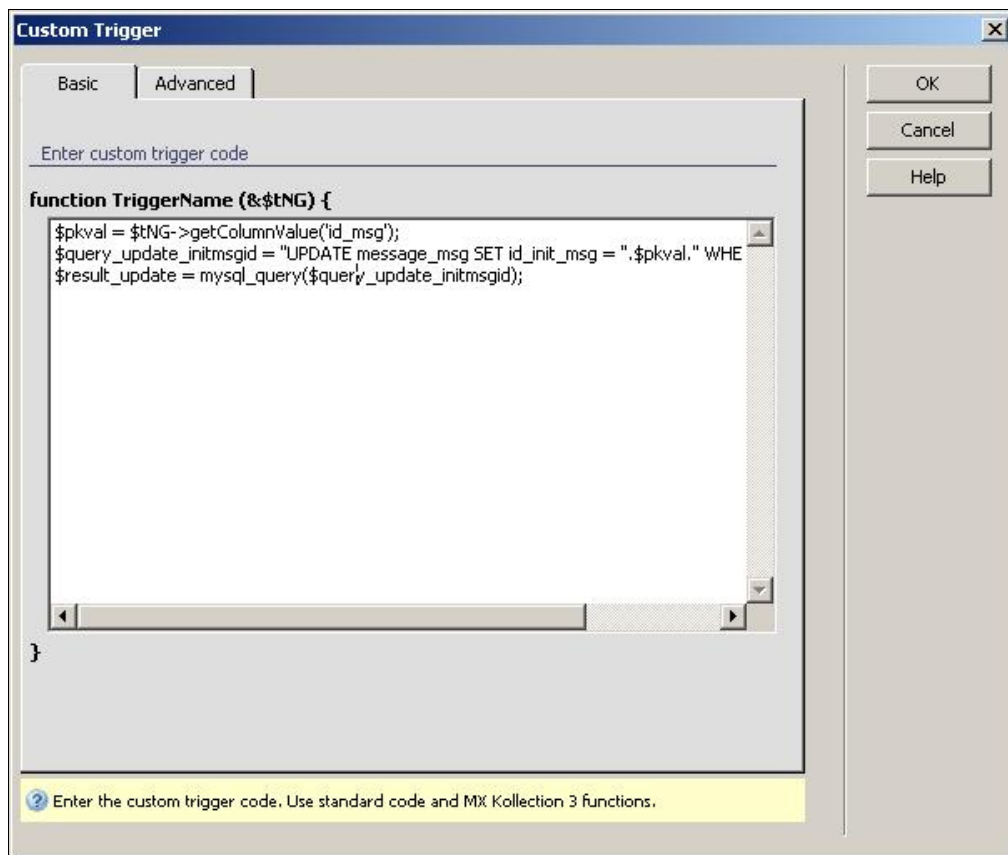
```
var pkval = "";
var query_update_initmsgid = "";
var result_update = "";
pkval = tNG.getColumnValue('id_msg');
query_update_initmsgid = "UPDATE message_msg SET id_init_msg = " &
pkval & " WHERE id_msg = " & pkval;
Request.UpdateMessage(query_update_initmsgid, tNG.connection);
return Request.KT_Null;
```

- This code declares two variables:

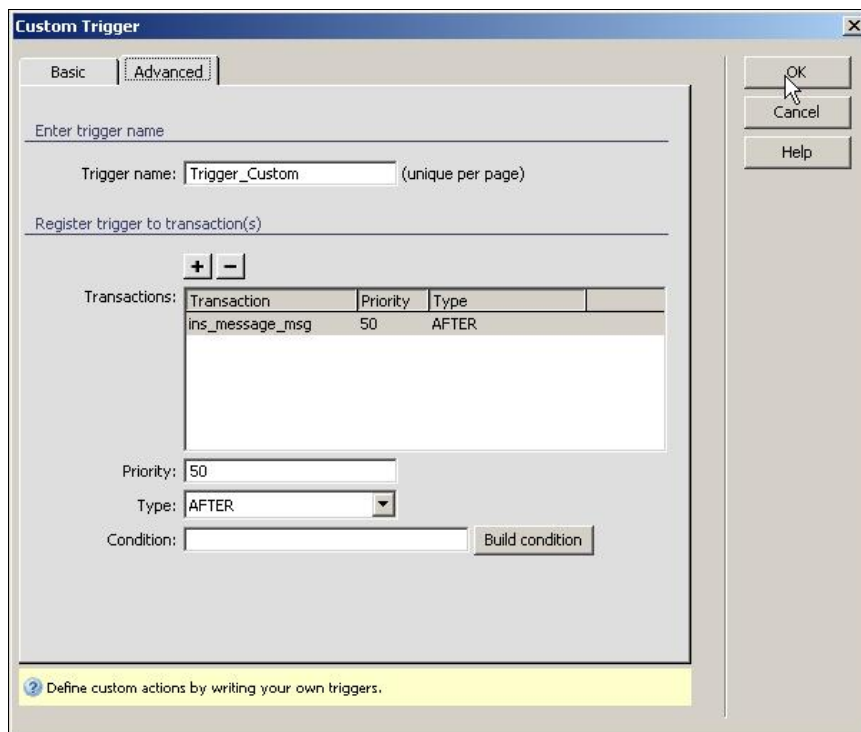
- `pkval`, which stores the intermediate ID of the last inserted record;
 - `query_update_initmsgid`, which stores the SQL query.

The query execution is performed by calling another function, which you will need to create later, at step 7.

4. The configured user interface should look similar to the image below (the screen shot was taken for the **PHP_MySQL** server model):



5. In the **Advanced tab**, you have to fill in the following options:
- The trigger name: leave it to its default value.
 - The trigger type: Must be an AFTER trigger, as it will get executed after the insert transaction. This way, the value of the primary key will be available.
 - The trigger priority: if you haven't added other triggers, you can leave it at its default value (50).
- The Advanced tab should look like in the image below:



6. When you're done configuring the **Custom Trigger**, click the **OK** button to close the dialog box and return to Dreamweaver. You can edit the code at any time, by double-clicking the **Custom Trigger** server behavior.
7. For **ColdFusion**, you need to perform some additional steps after applying the **Custom Trigger**:
 - Switch to code view in Dreamweaver.
 - In the **Server Behaviors** Tab of the **Application** panel, click the **Custom Trigger** you've just created. This will select the code corresponding to the trigger in Dreamweaver code view.
 - Before the opening `<cfscript>` tag that belongs to the trigger, you must add the following code:


```
<cffunction name="UpdateMessage">
<cfargument name="sql" type="string" required="true">
<cfargument name="connection" type="string" required="true">
<cfquery datasource="#connection#">
#PreserveSingleQuotes(sql)#
</cfquery>
</cffunction>
<cfset request.UpdateMessage = UpdateMessage>
```
 - This code block declares the UpdateMessage function that is used by the Custom Trigger for executing the update SQL query.

With this last action, the `post_message` page is complete, and you can start using it to post messages on the discussion board. All needed columns will be correctly filled in by the transaction. You can save and upload the page on the server. To view it however, you will need to go to the login page, and authenticate to the site. Then, browse to the topic of interest, and click the **Post message** link. As an additional improvement, you can change the label of the submit button to "Post", and replace "Content" with "Message", by going back to Dreamweaver and editing the page. Here is how your page should look:

Investment Opportunities	
Subject:	<input type="text" value="New car factory in Singapore"/>
Message: *	<input type="text" value="I've just heard they're opening a new car factory in Singapore. Do you happen to know more about this?"/>
Subscribe:	<input checked="" type="checkbox"/>
<input type="button" value="Post"/>	

Once you're done with the Post Message page, you can move on to the last section: building the reply to message page.

Reply to users

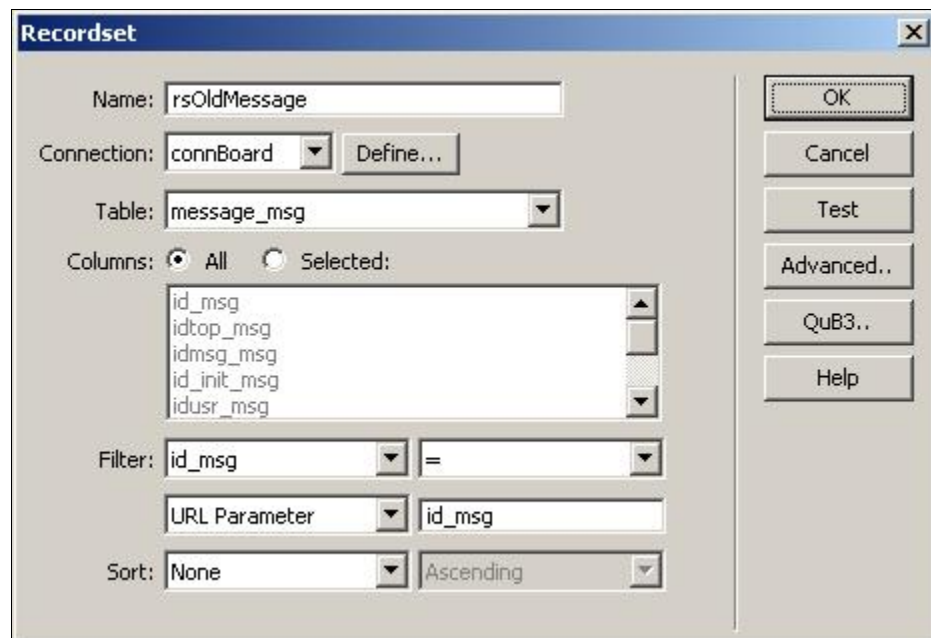
In this section of the tutorial you will build the page where users can reply to messages. The page will contain an insert form, but one which must also insert into the *message_msg* table details regarding about the parent message (i.e., the message being replied to) and also the original message (i.e., the message that started the thread).

The page used for replying to messages is called *reply_message*. It can be accessed by users from the *view_message* page, by clicking the **Reply to message** link. As you remember from a previous topic, the link also passes three URL parameters:

- ***id_top*** - the ID of the topic to which the message belongs.
- ***id_msg*** - the ID of the message that is being replied to. This will be stored as the parent ID.
- ***id_init_msg*** - the ID of the message that started the thread. All the other messages in the same thread have an foreign key pointing to the initial message.
- This page must also meet the following requirements (common for replies):
- The parent message ID must be the one passed through the URL parameter.
- Both messages must belong to the same topic (the ***idtop_msg*** for the new message will be the one retrieved from the URL parameter)
- The reply message subject entry field must have as default value "**Re: [old subject]**". This way, replies are easily identified.

With these goals and requirements in mind, you will build the page following these steps:

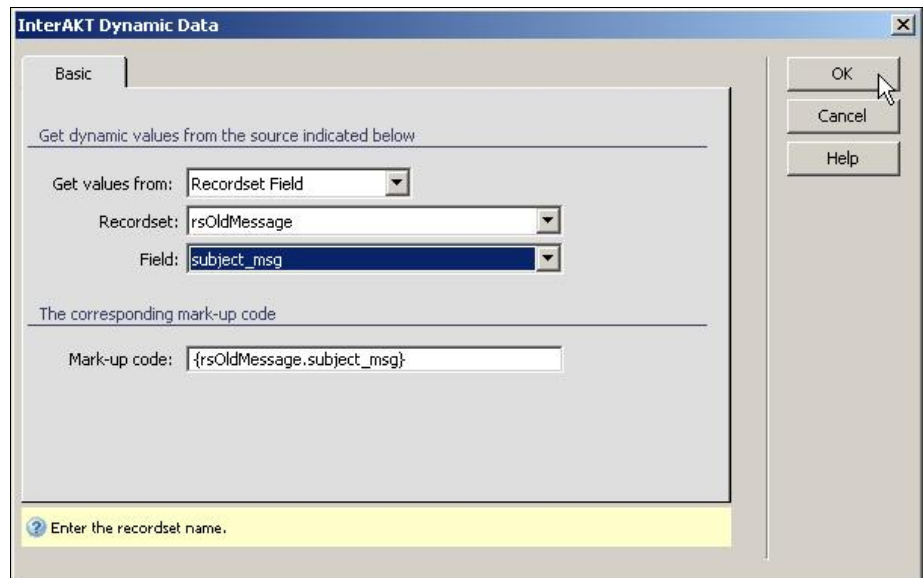
1. Open the *reply_message* page in Dreamweaver.
2. The same page layout as for the previous pages is already created (if you used the files in the ZIP package delivered with this tutorial). The insert form will be placed in the main table's middle row (the one without a background color).
3. Before creating the form, you must create a filtered recordset, that will retrieve the original message ID, based on the message ID passed as an URL parameter. To create the recordset, click the **Plus (+)** button of the **Bindings Tab > Recordset (Query)**. Set the database connection to *connBoard*, and the table to retrieve data from ***message_msg***. To filter the records, select in the filter drop-down menu the table's primary key (***id_msg***) and set as condition equality with the ***id_msg*** URL parameter, as seen below:



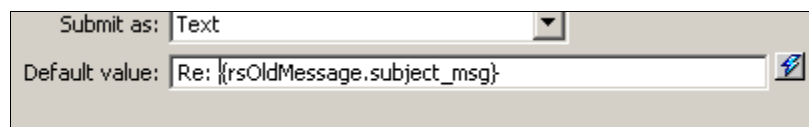
4. Click the **OK** button to create the recordset.
5. Drag the **subject_msg** field from the *rsOldMessage* recordset in the **Bindings** tab and drop it into the table's first row, after the **Reply to message:** text.



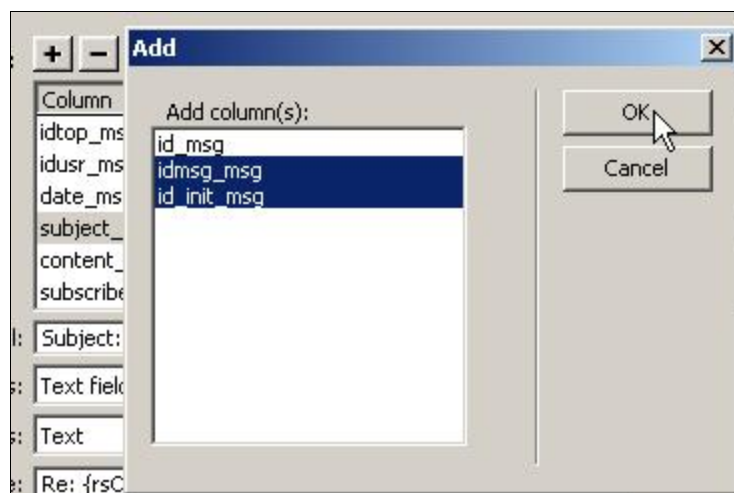
6. Now place the cursor in the middle row and start the **Insert Record Form Wizard**. You can access it from the **MX Kollection** tab of the **Insert bar**. If you used the same wizard in the previous topic, most of the interface fields should already be filled in, due to user interface persistence.
7. The first step is already filled in: the connection, table name, primary key and redirect page are preserved from the last time you applied the wizard in the *post_message* page. The first step should look like this.
8. On the second step, each field still has its default value and field type configured as in the Post Message page. The following fields are already set up: **idtop_msg**, **idusr_msg**, **date_msg**, **subject_msg**, **content_msg**, **subscribe_msg**.
 - Select the **subject_msg** field, and edit its default value. Click the **InterAKT Dynamic Data** icon (the blue lightning bolt) to select the **subject_msg** column from the *rsOldMessage* recordset.



Once the dynamic value is added to the **Default Value** text field, simply enter the "Re:" string before the automatically generated mark-up code.



- You need to add two more fields, if they are not displayed in the grid: **idmsg_msg** and **id_init_msg**. To add them, press the **Plus (+)** button on top of the grid and select the fields from the list.



- Configure them as follows:
 - idmsg_msg** - Displayed as Text, Default value: **id_msg** from the *rsOldMessage* recordset. Click the InterAKT Dynamic Data icon (the blue lightning bolt) to select its value from the recordset, without writing any code.
 - id_init_msg** - Displayed as Text, Default value: **id_init_msg** from the *rsOldMessage* recordset. Click the **InterAKT Dynamic Data** icon (the

lightning bolt) to select its value.

- Here is how step 2 of the Insert Record Form Wizard should look like, after configuration:

Step 2/3: Configure the form fields

Set form fields properties

Form fields: + -

Column	Label	Display as	Submit as
date_msg	Date:	Text	Date
subject_msg	Subject:	Text field	Text
content_msg	Content:	Text area	Text
subscribe_msg	Subscribe:	Check box	Checkbox: 1,0
idmsg_msg	Idmsg:	Text	Numeric
id_init_msg	Id_init:	Text	Numeric

Label: Id_init:

Display as: Text

Submit as: Numeric

Default value: {rsOldMessage.id_init_msg}

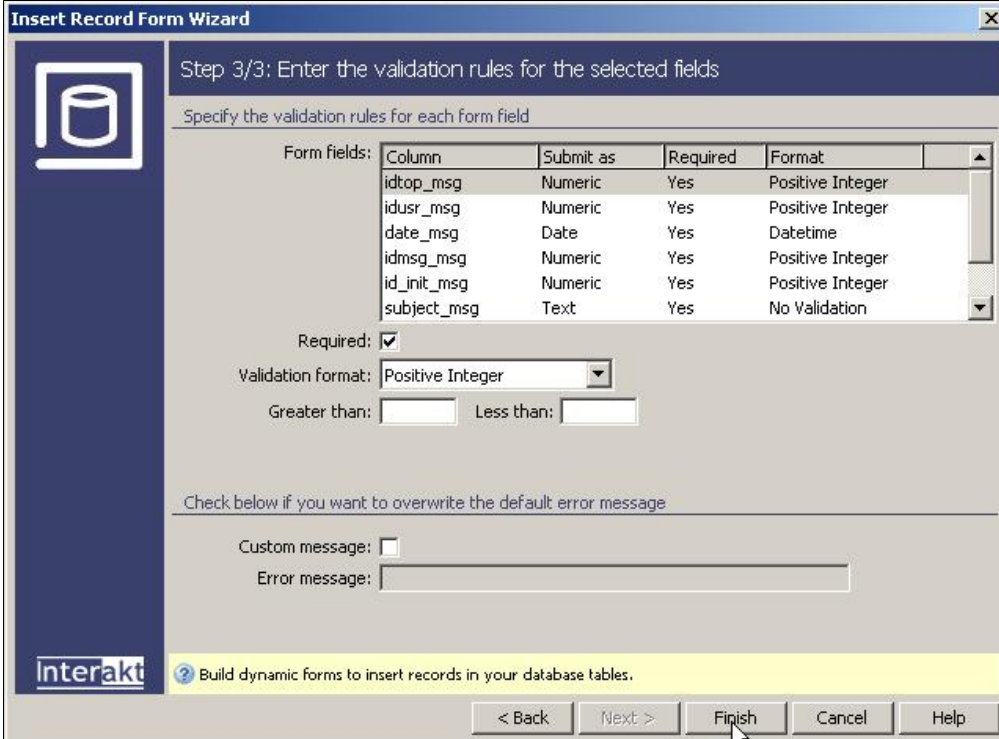
Interakt

Enter the default value for the current form field.

< Back Next > Finish Cancel Help

Click **Next** to proceed to step 3.

- In step 3 of the Insert Record Form Wizard, apply validation rules on the transaction fields. Make all fields required, except the Subscribe check-box. Also, all the foreign keys (*idtop_msg*, *idusr_msg*, *idmsg_msg*, *id_init_msg*) must be positive integers. Finally, the date field should have the **Datetime** format. Here is how step 3 should be configured:



Step 3/3: Enter the validation rules for the selected fields

Specify the validation rules for each form field

Form fields:	Column	Submit as	Required	Format
	idtop_msg	Numeric	Yes	Positive Integer
	idusr_msg	Numeric	Yes	Positive Integer
	date_msg	Date	Yes	Datetime
	idmsg_msg	Numeric	Yes	Positive Integer
	id_init_msg	Numeric	Yes	Positive Integer
	subject_msg	Text	Yes	No Validation

Required:

Validation format: Positive Integer

Greater than: Less than:

Check below if you want to overwrite the default error message

Custom message:

Error message:

Interakt Build dynamic forms to insert records in your database tables.

< Back Next > Finish Cancel Help

10. Once all fields are correctly set up, press the **Finish** button to close the wizard.
11. The wizard will generate the form in the row where you placed the cursor, and all the server behaviors in the Server Behaviors tab. You can change the Insert Transaction's behavior and fields at a later time (if needed) by double-clicking the server behavior with the same name.

Reply to message: {rsOldMessage.subject_msg}

(Display error message)

Idtop:	{text}
Idusr:	{text}
Date:	{text}
Subject:	{text} {Hint} {Error}
Content:	{text} {Hint} {Error}
Subscribe:	<input type="checkbox"/> {Error}
Idmsg:	{text}
Id_init:	{text}

Insert record

12. Since the **idtop_msg**, **idusr_msg**, **date_msg**, **idmsg_msg**, and **id_init_msg** should not appear in the form, select these fields in the **Dreamweaver** page and hit the **Delete** key. They are not removed from the **Insert Transaction**, but only from the HTML form.

The reply to message page is completed, and you can save and upload it to the server now. It cannot be accessed directly, because of the Restrict Access to Page server behavior applied to it. Instead, access the home page, enter your login information, and then post a reply to an existing message.

Reply to message: Adobe purchases Dreamweaver

Subject:	Re:Adobe purchases Dreamweaver
Content:	
Subscribe:	<input type="checkbox"/>

Insert Record

After you complete the *reply_message* page, the basic structure of the Discussion Board is finished. Users can view discussion topics, the messages under each topic, they can register an account with the Board, and authenticate using their login information, and then post new messages or reply to existing messages.

In the next section of the tutorial, you will improve the existing Discussion Board by allowing users subscribe to certain message threads, and receive e-mail notifications when a reply has been posted to one of their messages.

Send reply notification

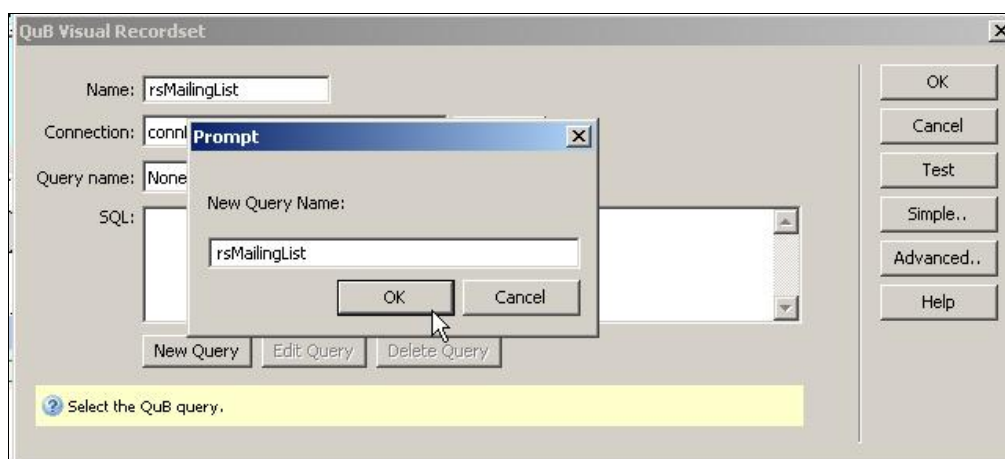
In this final topic of the Discussion Board tutorial, you will enhance the basic application by allowing users to receive an e-mail if a reply has been posted to one of the threads they subscribed to. Each user that checked the Subscribe checkbox when posting a message or a reply should receive an e-mail message when another reply is posted.

In this section, you will modify the *reply_message* page, so that it also sends an e-mail message to all users that have subscribed to the thread. Subscribing to a thread is done when the checkbox in the post or reply message page is checked. When a user checks the **Subscribe** checkbox, the value of the **subscribe_msg** column from the *message_msg* table is set to 1.

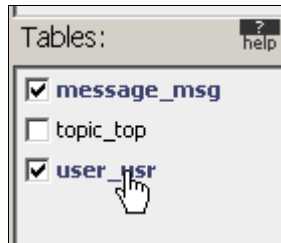
The e-mail notification will be implemented using the Send E-mail To Recipients From Recordset server behavior. To create the recordset that retrieves all subscribed users, you will use MX Query Builder.

To implement this functionality, follow the next steps:

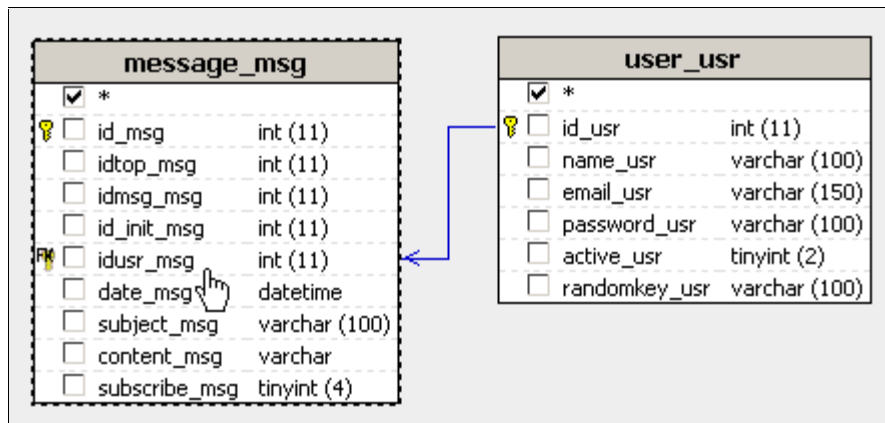
1. Open the *reply_message* page in Dreamweaver. This is where the send e-mail server behavior will be added. As explained when creating this page, the page receives as URL parameters the topic ID, the parent message ID, and the initial message ID. Based on the parent message ID, a recordset was created (*rsOldMessage*) retrieving all message details.
2. The recordset that will contain the e-mail addresses of all subscribed users will have to retrieve its data from two tables: *message_msg* and *user_usr*. To create this recordset, you also need to define these conditions:
 - The **subscribe_msg** column must be equal to 1, to prevent sending notifications to people who are not interested in receiving them.
 - The **id_init_msg** column must be equal to the **id_init_msg** URL parameter, to make sure only replies from the current thread are taken into account.
 - The **id_usr** column must be different from the **kt_login_id** session variable that stores the current user ID. Otherwise, the user will also receive a notification when he replies to one of the messages in the thread.
3. To create the recordset in a simple, visual environment, you will use the **MX Query Builder** again. Create a simple recordset called *rsMailingList* and start the **MX Query Builder** as you did in the view messages page:



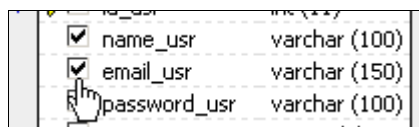
4. Once the **MX Query Builder** web interface opens, add the *message_msg* and *user_usr* tables to the query, by selecting them from the Tables panel.



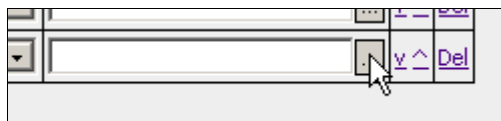
- In the **Database Diagram** drag the `idusr_msg` column from the `message_msg` table and drop it onto the `id_usr` column from the `user_usr` table. This will link the two tables using a LEFT JOIN.



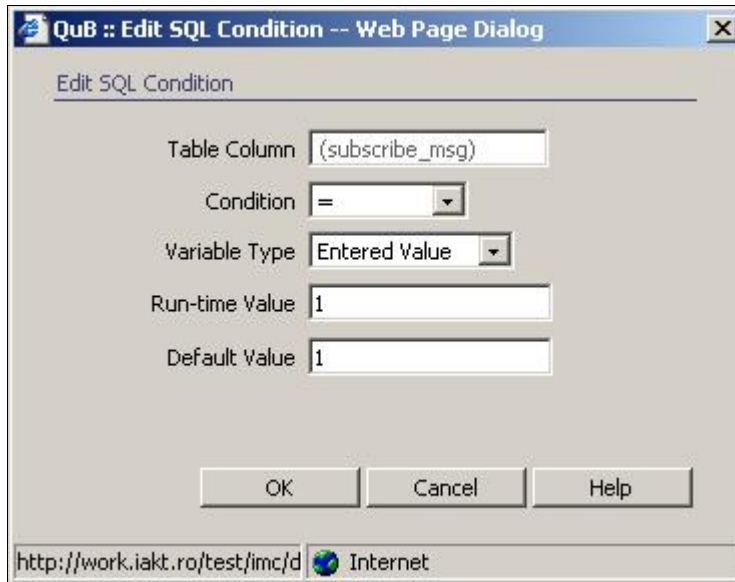
- In the **Database Diagram**, check the table columns you need for your recordset: `id_usr`, `email_usr`, `name_usr`, `id_init_msg` and `subscribe_msg`.



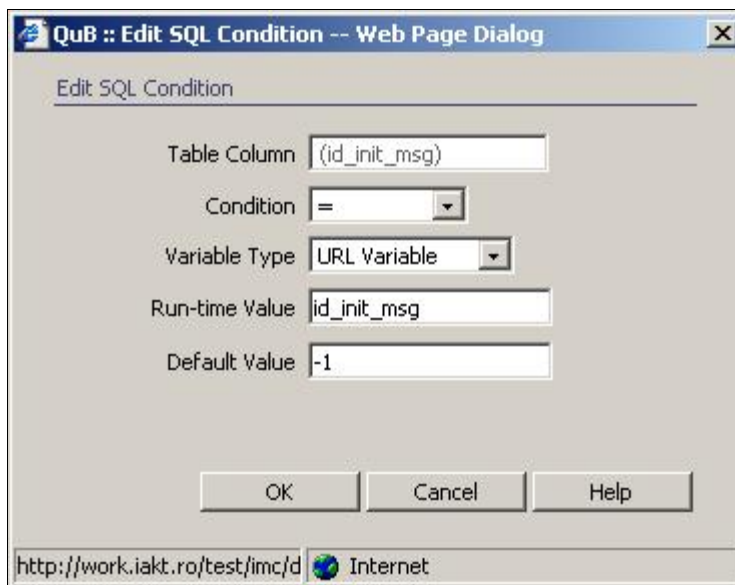
- From the Query Management Panel, define a condition for the `subscribe_msg` column, by clicking the corresponding button next to the Condition text field.



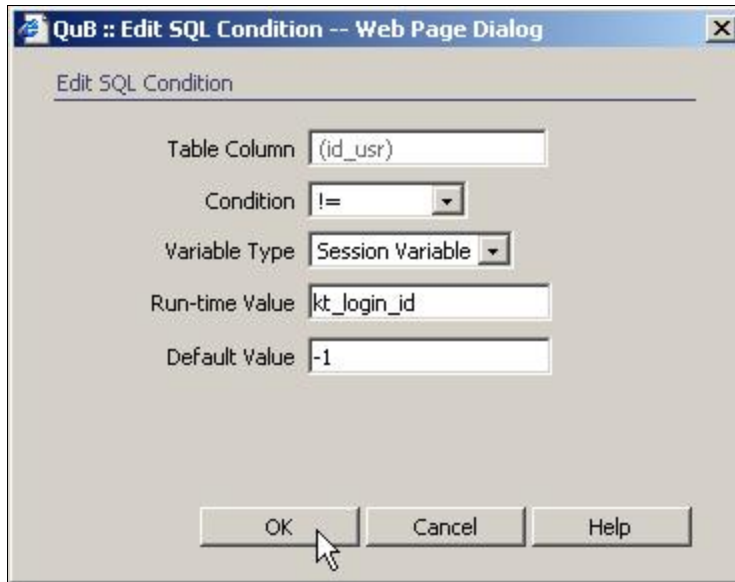
The column should be equal to 1, so you should configure it as in the image below:



- Next define a condition for the **id_init_msg** column: it should be equal to the **id_init_msg** URL parameter:



- Finally, define a condition for the **id_usr** column: it should be different from the **kt_login_id** session variable:



- Once all the columns are selected, and the conditions are set, the QuB3 user interface should look like in the following image:

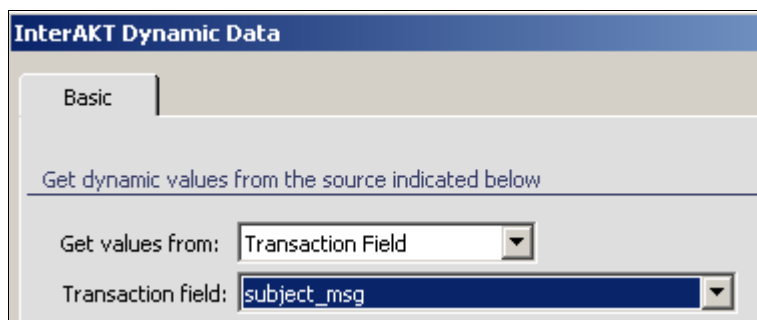
Column Name	Alias	Sort	Output	Grouping	Condition
user_usr.email_usr			<input checked="" type="checkbox"/>		
user_usr.name_usr			<input checked="" type="checkbox"/>		
message_msg.id_init_msg			<input checked="" type="checkbox"/>		=\$_GET["id_init_msg"]
message_msg.subscribe_msg			<input checked="" type="checkbox"/>		=1
user_usr.id_usr			<input checked="" type="checkbox"/>		!= \$_SESSION["kt_login_id"]

- Save the query by clicking the Save button from the **Query List** panel. You can now close the **MX Query Builder** web interface. Back in **Dreamweaver**, press the **Refresh** button to display the new query in the **SQL** text area.



Click **OK** to generate the code for the recordset.

12. Next apply the **Send E-mail To Recipients From Recordset** server behavior. You can access it from the **Server Behaviors tab > + > Mx Kollection > Send E-mail**.
13. Configure the user interface as follows:
 - In the **Recordset** drop-down menu select the *rsMailingList* recordset created at steps 2-11.
 - In the **E-mail to field** menu, select the *email_usr* field.
 - In the **From** field, enter an e-mail address, or leave it at its default value (this value can be changed from InterAKT Control Panel > Define e-mail server settings)
 - In the **Subject** field, enter the text "Reply posted", then add the dynamic value for the subject of the message, by using the **InterAKT Dynamic Data** button. The dynamic value will be retrieved from the transaction field {subject_msg}, as seen here:



- Select to write the body of the e-mail, then write a message in the Content text area. The message can be personalized by including the user name from the rsMailingList recordset, using the InterAKT Dynamic Data.

InterAKT Dynamic Data

Basic

Get dynamic values from the source indicated below

Get values from: Recordset Field

Recordset: rsMailingList

Field: name_usr

The corresponding mark-up code

Mark-up code: {rsMailingList.name_usr}

The corresponding mark-up code will be inserted in the **Content** text area:

Body: Write content

Content: Dear {rsMailingList.name_usr},

- Type the following text after the introductory line:

"A reply has been posted to one of your messages on the Discussion Board. To read the message click"

Also, add a link to allow the user to view the corresponding message thread:

```
<a href = view_messages.php?id_top={idtop_msg}>here</a>
```

To add dynamic data to the link, use the **InterAKT Dynamic Data**.

- Here is how the configured server behavior should look like:

14. Save the page and upload it to the server. When an user will post a reply to a message you're subscribed to, you will receive an e-mail message using the format above.

From:	admin@discussionboard.com
Date:	Wednesday, June 22, 2005 4:54 PM
To:	mzaharia@interaktonline.com
Subject:	Reply posted: Re: The stock market - an opportunity?

Dear Jane, A reply has been posted to one of your messages on the Discussion Board. To read the message click [here](#).

If you click on the link, it will open the `view_message` page, with the correct topic selected.

You have now completed the Discussion Board tutorial. Feel free to improve it in any way you like, using any of the MX Kollection server behaviors or by writing your own code.

Other Resources

Other Dreamweaver Extensions from InterAKT

- KTML
- MX Kart
- MX Site Search
- MX RSS Reader-Writer
- MX Dynamic Table Sorter
- MX Coder Pack
- MX Dynamic Charts
- MX CSS Dynamic Menus

Contact Us

- **Address:**
1-11 Economu Cezarescu ST, AYASH Center, 1st floor
Sector 6, ZIP 060754, Bucharest, Romania
- **Web:** <http://www.interaktonline.com/>
- **E-mail:** contact@interaktonline.com
- **Phone:** +4031 401.68.19 or +4021 312.51.91
- **Fax:** +4021 312.53.12

Copyright

Windows is a trademark of Microsoft, Inc.

Dreamweaver MX is a trademark of Macromedia, Inc.

Redhat is a trademark of Redhat, Inc.

Copyrights and Trademarks

Copyright 2000-2005 by InterAKT Online.

All Rights Reserved. This tutorial is subject to copyright protection.

PHAkt, ImpAKT, NeXTensio, MX Query Builder, Transaction Engine, MX Includes, KHTML, MX Kommerce, MX Kollection, MX Widgets, MX Looper, MX Dynamic Charts, MX CSS Dynamic Menus, MX Tree Menu, MX Form Validation, MX File Upload, MX Send E-mail, MX User Login, MX CSV Import-Export, MX Kart, MX Site Search, MX Dynamic Table Sorter, MX RSS Reader-Writer, MX Coder Pack, MX Dynamic Charts are trademarks of InterAKT Online.

All other trademarks are acknowledged as the property of their respective owners.

This document and the product to which it pertains are distributed under licenses restricting their use, copying, distribution, and decompilation.

No part of this document or of the associated product may be reproduced in any form by any means without prior written authorization of InterAKT Online, except when presenting only a summary of the tutorial and then linking to the InterAKT website.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Send comments and suggestions to products@interaktonline.com



InterAKT Online

Web: <http://www.interaktonline.com/>

E-mail: contact@interaktonline.com

Address: 1-11 Economu Cezarescu ST, AYASH Center, 1st floor, Sector 6, ZIP 060754, Bucharest, Romania

Phone: +4021 312.51.91

Fax: +4021 312.53.12