



IBM Software Group

WebSphere® Message Broker Version 6.1

Route, DatabaseRoute and DatabaseRetrieve nodes



@business on demand.

© 2008 IBM Corporation
Updated January 17, 2008

This presentation will discuss the new route, DatabaseRoute and DatabaseRetrieve nodes in Message Broker Version 6.1

Agenda

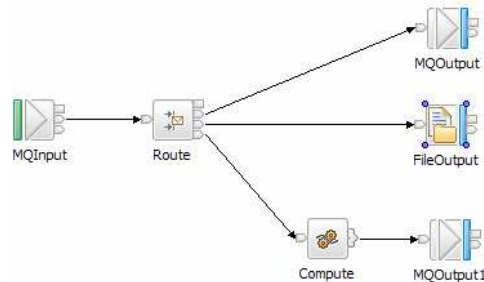
- Route
- DatabaseRoute
- DatabaseRetrieve

This presentation covers the topics shown on this slide.

A recurring theme with the node enhancements in version 6.1 is as much as possible to remove the need for programming with ESQL or with Java™. To that end, these three new nodes have been designed to enable these basic functions without the need to write application code. The routing decisions that are taken are built using the Message Broker toolkit, and provide an intuitive way to create the appropriate expressions.

Route node

- Directs messages that meet certain criteria down different parts of a flow



- Extremely easy to use – “removes the need for writing code”
 - ▶ Uses XPath expressions to identify match criteria
 - ▶ Define additional output terminals to specify multiple expressions
 - ▶ Default terminal if no matches are found

3

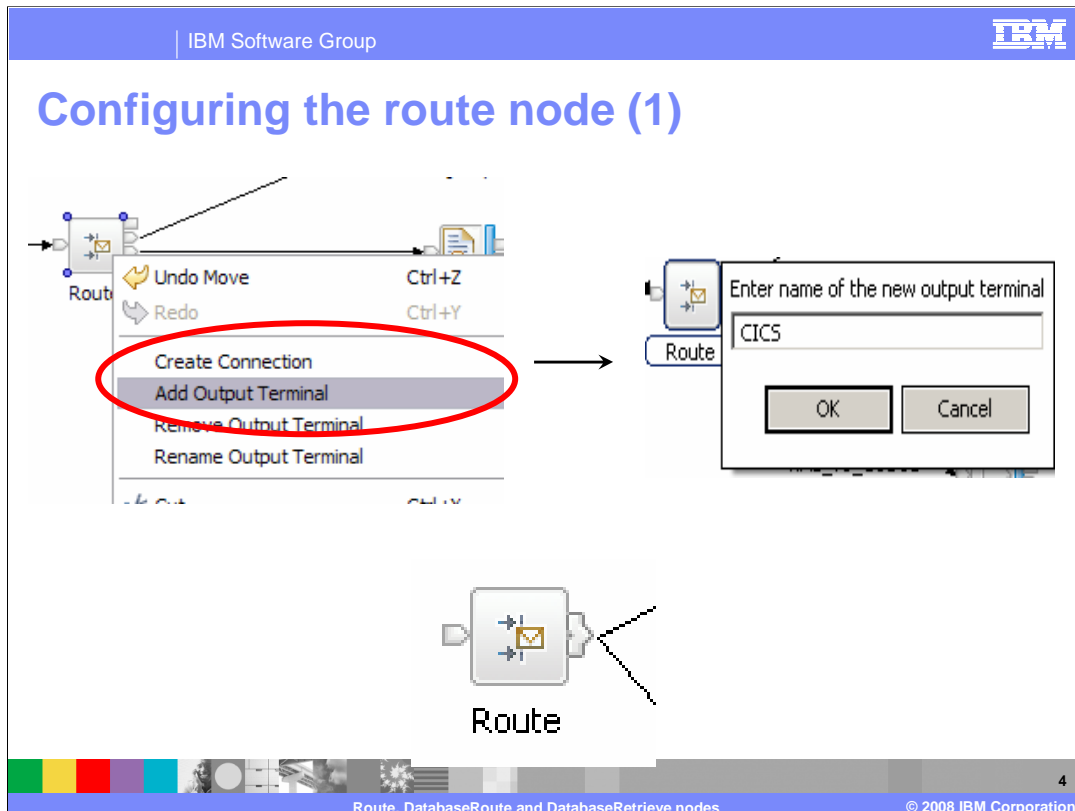
Route, DatabaseRoute and DatabaseRetrieve nodes

© 2008 IBM Corporation

The new Route node provides a graphical technique to enable incoming requests to be examined, and flow sent down the appropriate part of the message flow, depending on the specified criteria.

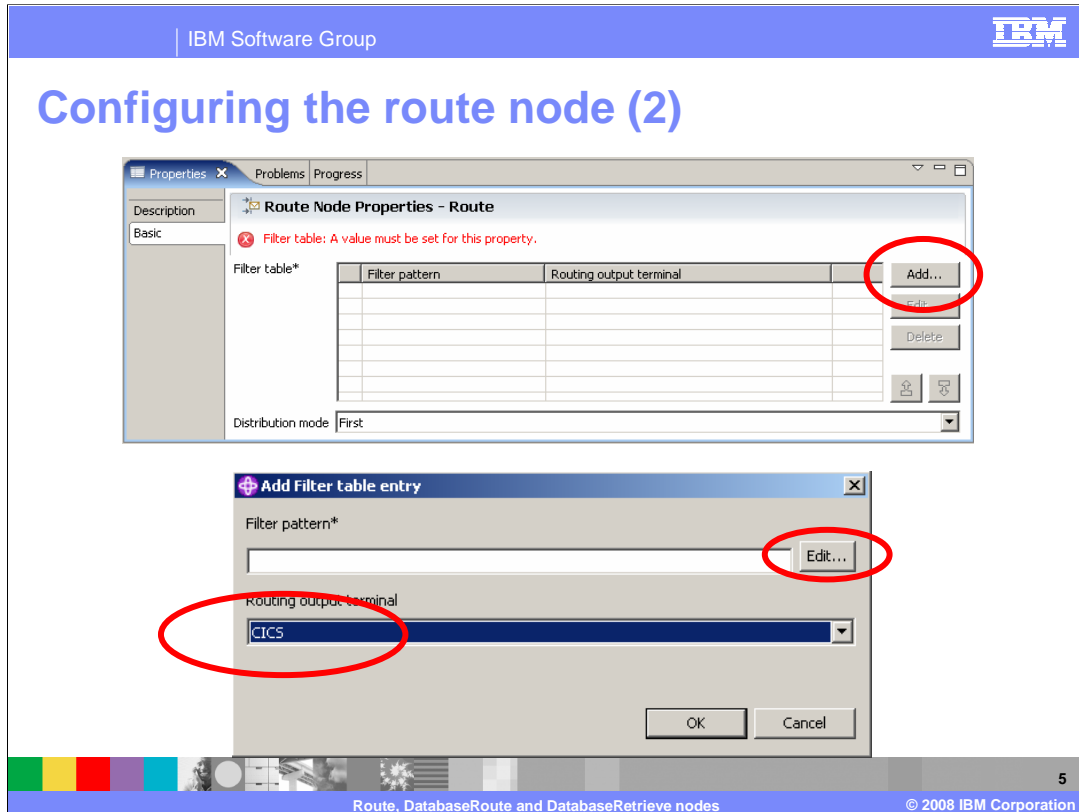
The Route node matches the incoming data to a set of specified criteria. Based on this criteria, the message can be routed to output terminals on the Route node. These terminals can be specified by the application developer, as shown on the next slide.

The filter criteria are specified by the generation of an X-Path statement in the Filter pattern for each required match. An example of this is shown on the next few slides.



To configure the Route node, first drop a node onto the message flow editor. Then, right-click the Route node, and select "Add Output Terminal". When the dialogue box opens, provide a name for the new output terminal. This example creates a new terminal called "CICS".

You can define as many output terminals as you like. This will be determined by the profile of your message flow. When you have more than four output terminals, the node will consolidate all terminals into the combined display, as shown on this slide. To connect these terminals, click on the terminal arrow, select the required terminal, and wire to the next node as needed.



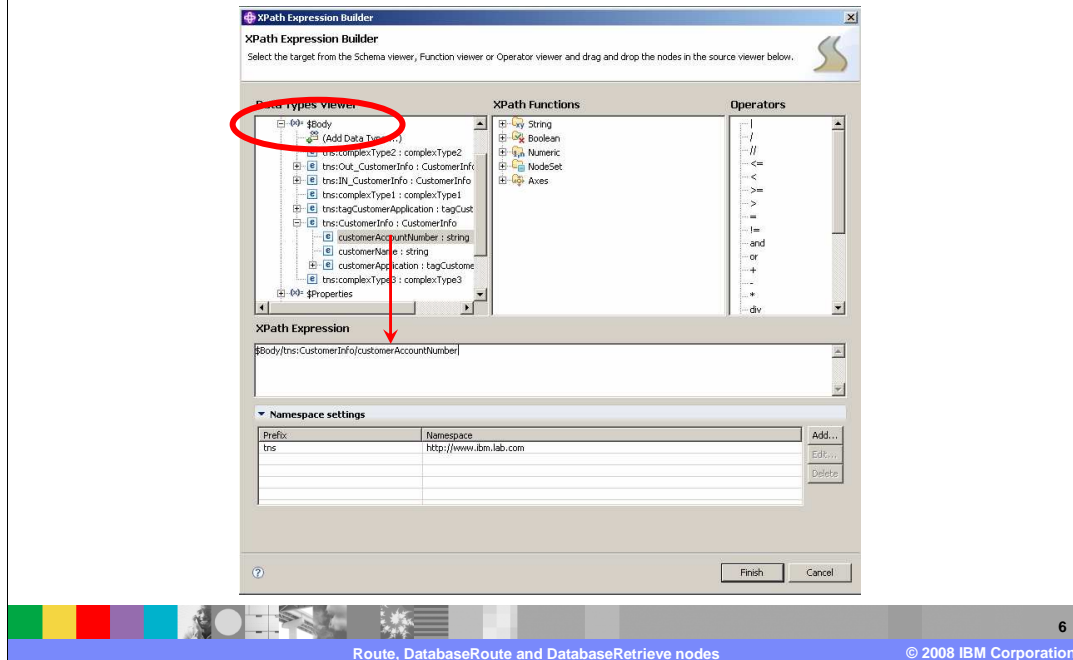
Once all terminals have been defined, you need to specify the filter properties for each condition. The first screen capture shows the properties of the Route node, before any filters have been defined.

To create a filter, click the Add button. This can be done for each of the filter that are required.

When you have clicked the Add button, the second window will open. First, select the appropriate output terminal by using the pull-down. This will be populated with all of the output terminals that you have created on the Route node.

Then click the "Edit" button to specify the filter.

Configuring the route node (3)

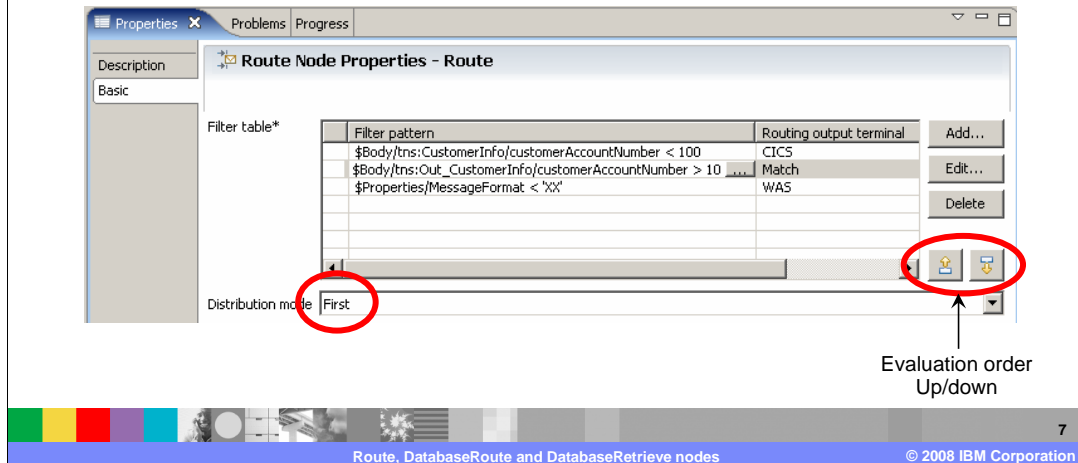


The Edit button brings you to the Expression Builder. There are many ways in which a valid XPath expression can be built.

As a simple example, expand the Variables structure in the “Data Types Viewer”, then expand \$Body (pronounced dollar-body). Then expand an appropriate part of the message tree. When you have located the required part of the message tree, drag this component onto the XPath Expression pane in the middle section of the window. The example on this slide uses the field customerAccountNumber. Then complete the expression, either by manually adding the criteria, or by dragging and dropping the required operator. You can also match fields from other parts of the incoming message, and you are not restricted to the Body of the message.

Configuring the route node (4)

- Distribution mode
 - ▶ Propagate message to the first terminal that matches, or to all that match



Finally, on the Route node properties, you need to specify how you want each criteria to be checked.

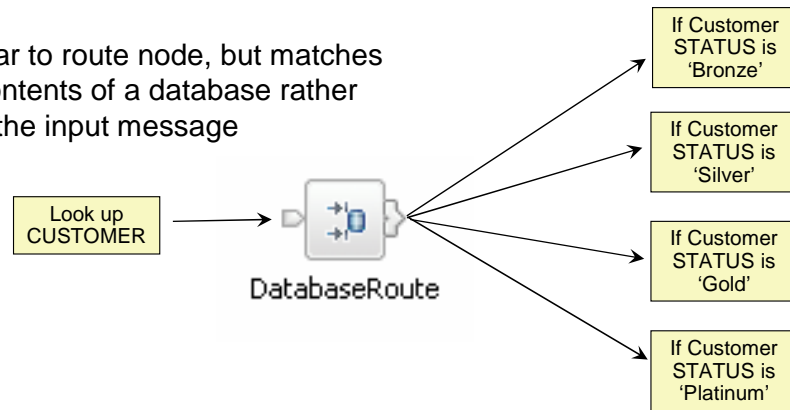
The default value for Distribution Mode is "All". This means that all Filter patterns are checked, and every filter condition that is true results in the message being passed to the specified output terminal. In this case, multiple paths in the message flow will be executed.

If the Distribution Mode is changed to "First", each filter pattern will be evaluated in turn, starting from the top of the list. When the first matching filter is encountered, control will be passed to the specified terminal.

The order in which filter patterns can be evaluated can be changed by using the UP and DOWN buttons. These will be shown in color when there are two or more defined filter patterns, and when one of the filter patterns is highlighted by clicking on the filter.

DatabaseRoute node

- Similar to route node, but matches on contents of a database rather than the input message



CUSTOMER	STATUS
00001	SILVER
00002	GOLD
00003	BRONZE

The DatabaseRoute node provides a similar function to the Route node, but the routing decision is based on the contents of a relational database. In the example shown on this slide, the incoming message is a customer record, containing a customer number. This request needs to be routed to a certain part of the flow, depending on the STATUS of the customer. However, this information is held on a relational database, which needs to be read by the DatabaseRoute node.

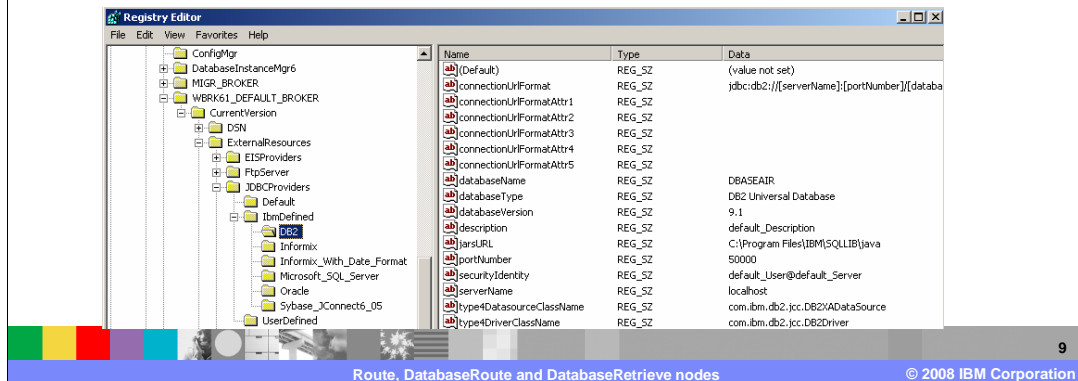
Configuring the DatabaseRoute node (1)

- JDBC configuration – simple example, Windows® only, local installation of DB2®

```
mqsireportproperties <BROKER> -c JDBCProviders -o DB2 -r
```

```
mqsichangeproperties <BROKER> -c JDBCProviders -o DB2 -n databaseName  
-v <DATABASENAME>
```

```
mqsichangeproperties <BROKER> -c JDBCProviders -o DB2 -n serverName -  
v localhost
```



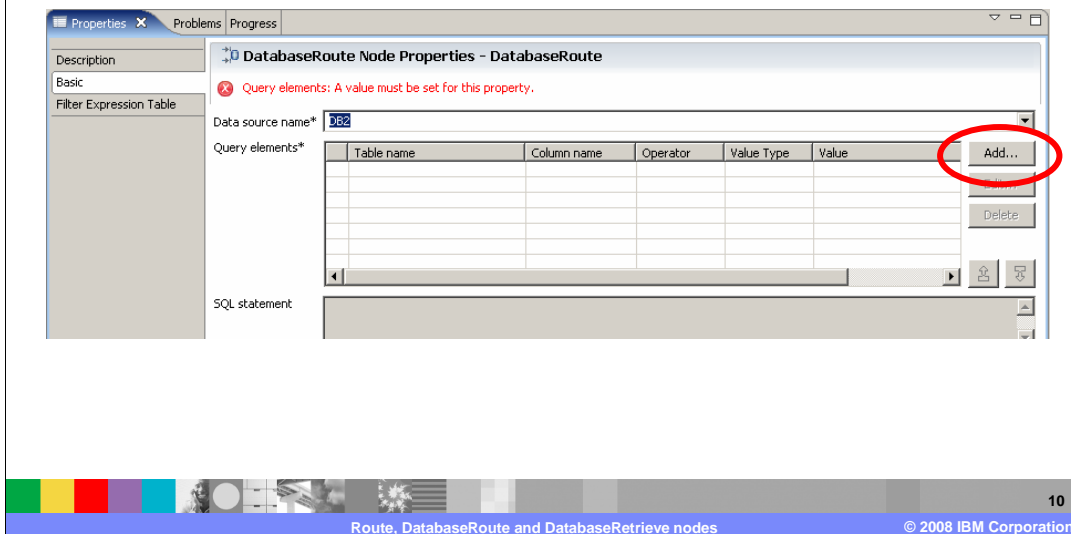
Prior to building a message flow with the DatabaseRoute node, you need to prepare the database environment. This can vary significantly between environments, and depends on both the platform where the broker and database reside, and the database that is being used.

This slide shows a very simple example, where the database is IBM DB2, and where the DB2 database is installed on the same Windows platform as the Message Broker. The JDBC configuration needs to be created within the Broker environment, and this is done using Configurable Services. In this case, the “mqsichange-properties” command is used to set the database name and the location of the database server, so that the Broker can locate the database at runtime. The second and third commands on this slide are used to do this.

In a Windows environment, this JDBC configuration is stored in the Windows registry. In this example, you will see that the databaseName has been set to “DBASE AIR”.

For more complex scenarios, ensure you review the Message Broker Information Center for details on JDBC configuration.

Configuring the DatabaseRoute node (2)

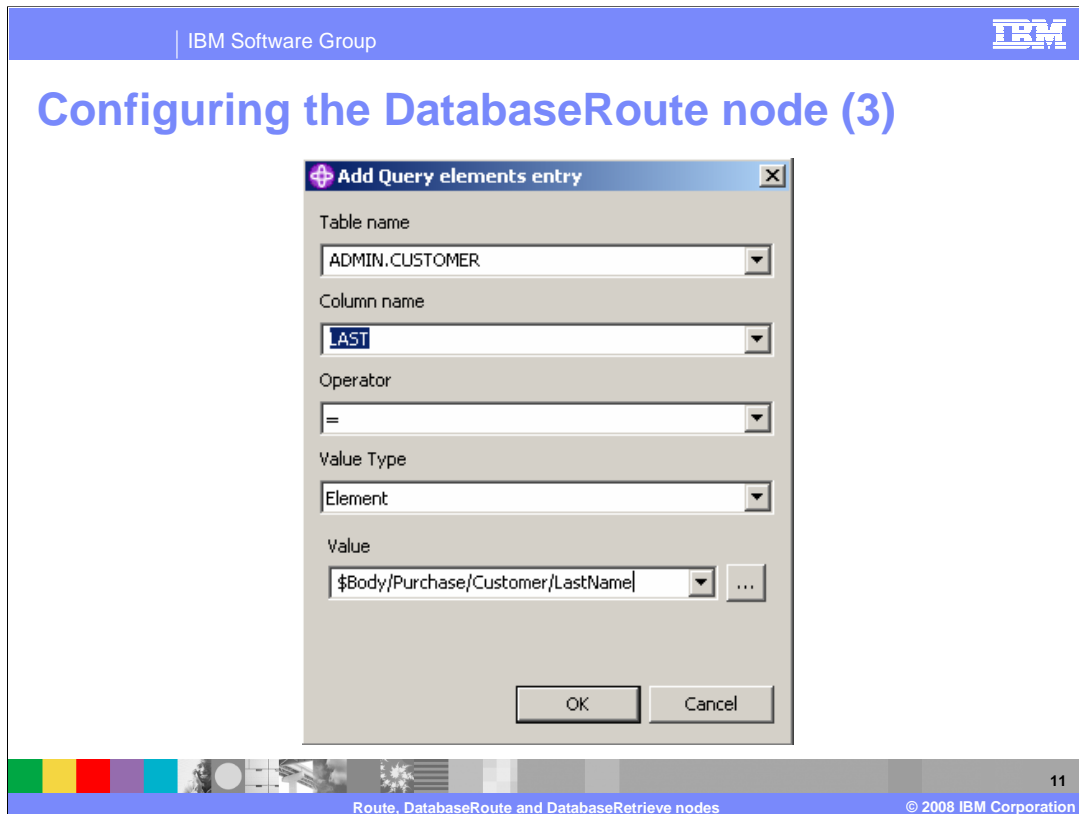


When the DatabaseRoute node is dropped on to the flow, the properties of the node will look like this.

The Basic tab is shown by default, and this is used to specify the database access statement to query the database.

Later, the Filter Expression Tab will be used to specify the routing decision.

Clicking Add on this screen opens a dialogue window, as shown on the next slide.



The Add Query window opens.

The values shown in this example are as follows:

“ADMIN” is the database schema, and CUSTOMER is the database table name.

“LAST” is the column of the database table that the DatabaseRoute node will read.

Operator is a Boolean value which specifies how the required row should be retrieved. This example is requesting a direct match on the element value.

Finally, value contains the XPath expression which identifies the data in the incoming message. This data is used to read the required database row. This value can either be typed in manually, or can be automatically included by clicking the button to the right side of this field, and following the dialogue. This will be a similar dialogue to the Route node.

Click OK to move to the next window.

Configuring the DatabaseRoute node (4)

The screenshot shows the 'DatabaseRoute Node Properties' dialog box. The 'Data source name*' field is set to 'DB2'. The 'Query elements*' table has the following data:

Table name	Column name	Operator	Value Type	Value
ADMIN.CUSTOMER	LAST	=	Element	\$Body/AirlineRequest/

The 'Add...' button is circled in red. The 'SQL statement' field contains the following text:

```
FROM ADMIN.CUSTOMER
WHERE ADMIN.CUSTOMER.LAST = ?
```

This has now created a new query element for the DatabaseRoute node, and the details are shown in the properties. The generated SQL statement is shown at the bottom of these properties.

To ensure the query statement retrieves at least one valid row, there needs to be a query element with a null value type. If you do not do this, the SQL statement will not be valid.

Click Add again, as shown on this slide.

This will again open the Query Elements entry, as shown on the next slide.

Configuring the DatabaseRoute node (5)

Add Query elements entry [X]

Table name
ADMIN.CUSTOMER

Column name
STATUS

Operator
ASC

Value Type
None

Value
None

OK Cancel

On this window, the Table name and column name are as before. However, the Operator value is “ASC”, and the remaining values are defaulted.

Configuring the DatabaseRoute node (6)

DatabaseRoute Node Properties - DatabaseRoute

Data source name* DB2

Table name	Column name	Operator	Value Type	Value
ADMIN.CUSTOMER	LAST	=	Element	\$Body/AirlineRequest/Purchase/Customer/LastName
ADMIN.CUSTOMER	STATUS	ASC	None	None

SQL statement

```
SELECT ADMIN.CUSTOMER.STATUS
FROM ADMIN.CUSTOMER
WHERE ADMIN.CUSTOMER.LAST = ?
ORDER BY ADMIN.CUSTOMER.STATUS ASC
```

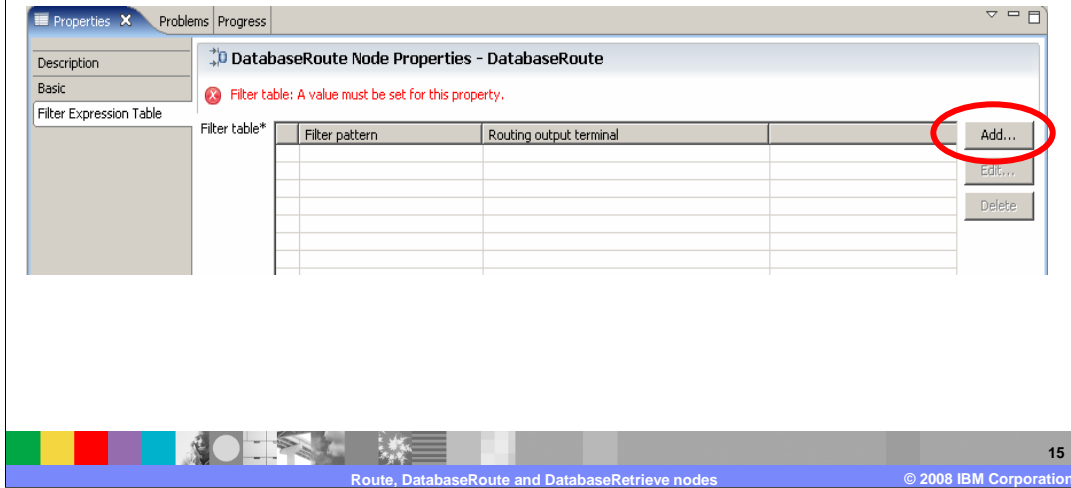
Distribution mode All

Route, DatabaseRoute and DatabaseRetrieve nodes © 2008 IBM Corporation 14

This creates a second entry in the Query Elements table. The resulting SQL statement is shown at the bottom.

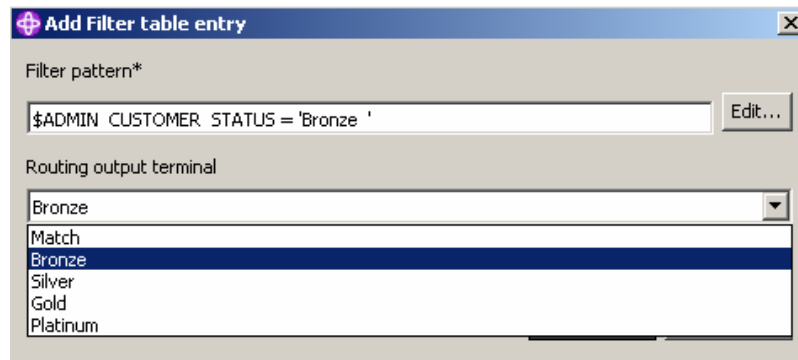
The Query Elements are now complete. Click on "Filter Expression Table" to construct the filter part of the properties. This will show the window on the next slide.

Configuring the DatabaseRoute node (7)



Now click "Add" to create a new filter expression.

Configuring the DatabaseRoute node (8)

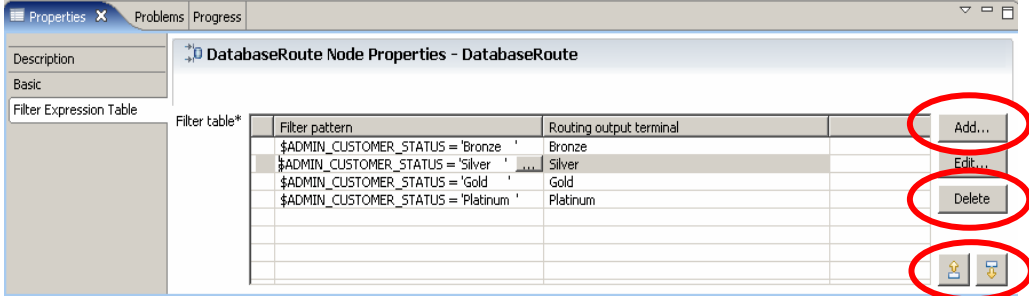


The “Add Filter” dialogue will open. Specify the Filter pattern that is required. In this example, ADMIN is the schema, CUSTOMER is the table, and STATUS is the column. These are all connected using an underscore character, which is not clearly visible on this screen capture.

Using the pull-down, select the appropriate output terminal. This list is generated from the new terminals that were defined earlier.

Click OK to complete the dialogue for one filter pattern.

Configuring the DatabaseRoute node (9)



Properties Problems Progress

DatabaseRoute Node Properties - DatabaseRoute

Description Basic

Filter Expression Table

Filter pattern	Routing output terminal	
\$ADMIN_CUSTOMER_STATUS = 'Bronze '	Bronze	Add...
\$ADMIN_CUSTOMER_STATUS = 'Silver '	Silver	Edit...
\$ADMIN_CUSTOMER_STATUS = 'Gold '	Gold	Delete
\$ADMIN_CUSTOMER_STATUS = 'Platinum '	Platinum	

17

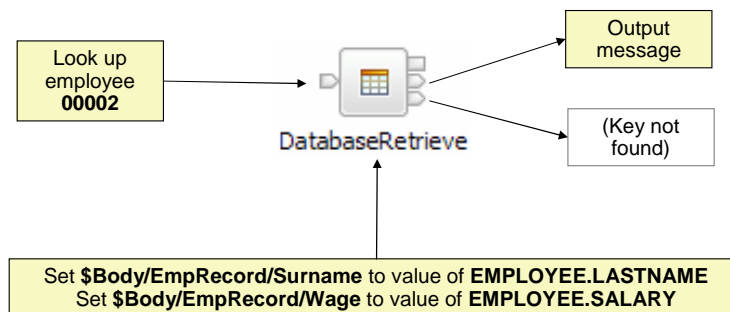
Route, DatabaseRoute and DatabaseRetrieve nodes © 2008 IBM Corporation

If necessary, add further filter patterns by using the Add button. Existing filter patterns can be edited or deleted from this window.

The filter expressions are evaluated in the order shown. If you need to change the order, use the colored buttons on the bottom right hand side of this window.

DatabaseRetrieve node

- Allows data to be retrieved from one or more database tables
- The retrieved data is stored as elements within this node's outgoing message.
- Queries can span multiple tables and select multiple predicates



18

Route, DatabaseRoute and DatabaseRetrieve nodes

© 2008 IBM Corporation

A similar node is the DatabaseRetrieve node. This node enables data to be retrieved from one of more relational databases, and use a similar dialogue to the DatabaseRoute node.

It uses JDBC to connect to the database, so the same base configuration as with the DatabaseRoute node needs to be built.

This example uses the DatabaseRetrieve node to look up the record of employee number 00002. Two values are retrieved from the database, which are then used to populate the Body of the message, which is passed down the message flow. Follow the example shown on the DatabaseRoute node to understand how to configure the node itself.

Configuring the DatabaseRetrieve node [2]

- Alternative terminals
 - ▶ **keyNotFound**: if the query is unsuccessful or the outgoing message is unsuccessfully modified
 - ▶ **failure**: If an exception occurs during the processing of the input message
- Configuring Database access
 - ▶ Store user ID and password information using mqsisetdbparms
 - ▶ Also used by DatabaseRoute

When accessing a database, the requested record may not be found. In this case, or if the message fails to be updated correctly, control will be passed to the “keyNotFound” terminal.

If a more general error occurs, control will be passed to the failure terminal, as normal.

Summary

- Route
 - ▶ Route messages based on fields in message
- DatabaseRoute
 - ▶ Route messages based on database entries
- DatabaseRetrieve
 - ▶ Populate fields using data in a database record

This presentation has covered the new Route, DatabaseRoute and DatabaseRetrieve nodes in Message Broker Version 6.1

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WMB61_IEA_Route_DB_Route.ppt

This module is also available in PDF format at: [./WMB61_IEA_Route_DB_Route.pdf](http://WMB61_IEA_Route_DB_Route.pdf)



You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

DB2 IBM WebSphere

Windows, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

JDBC, Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

© Copyright International Business Machines Corporation 2008. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.