Course Exercises

IBM Tivoli Netcool/Impact 7.1 Administration and Implementation

Course code TN045 ERC 2.1

IBM Training

February 2017 edition

NOTICES

This information was developed for products and services offered in the USA.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

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 and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

TRADEMARKS

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

IT Infrastructure Library is a Registered Trade Mark of AXELOS Limited.

ITIL is a Registered Trade Mark of AXELOS Limited.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

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

UNIX is a registered trademark of The Open Group in the United States and other countries.

Contents

Unit 1 Introduction to IBM Tivoli Netcool/Impact exercises

This unit's exercises focus on Netcool/Impact installation and the initialization of the database to be used for class exercises.

Exercise 1 Installing Netcool/Impact V7.1.0.1 on SUSE Linux

In this exercise, you use IBM Installation Manager to install Netcool/Impact 7.1.0.1 and the Netcool/Impact extensions for Netcool Operations Insight (NOI) 1.4. IBM Installation Manager was installed with the non-root user ID **netcool** during the virtual image configuration. A Netcool/OMNIbus ObjectServer is installed during the virtual image configuration. The ObjectServer user repository is used during the Netcool/Impact installation.

1. Log in to the Impact71_IMG1 virtual image as the user **netcool**, with the password **object00**.
2. Open a command prompt and start the Netcool OMNIbus ObjectServer. Enter the following command:

   ```
   /opt/IBM/tivoli/netcool/omnibus/bin/nco_objserv -name NCOMS &
   ```

3. Verify that the ObjectServer is operational. Enter the following command:

   ```
   /opt/IBM/tivoli/netcool/omnibus/bin/nco_ping NCOMS
   ```

4. Change to the Installation Manager directory:

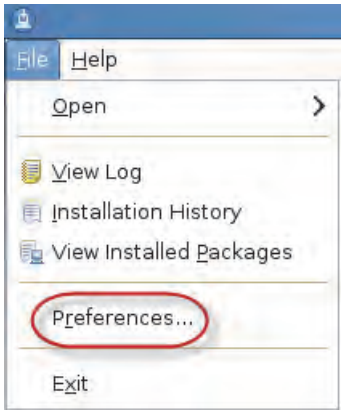
   ```
   cd /home/netcool/IBM/InstallationManager/eclipse
   ```

5. Use the following command to start the Installation Manager tool:

   ```
   ./IBMIM
   ```

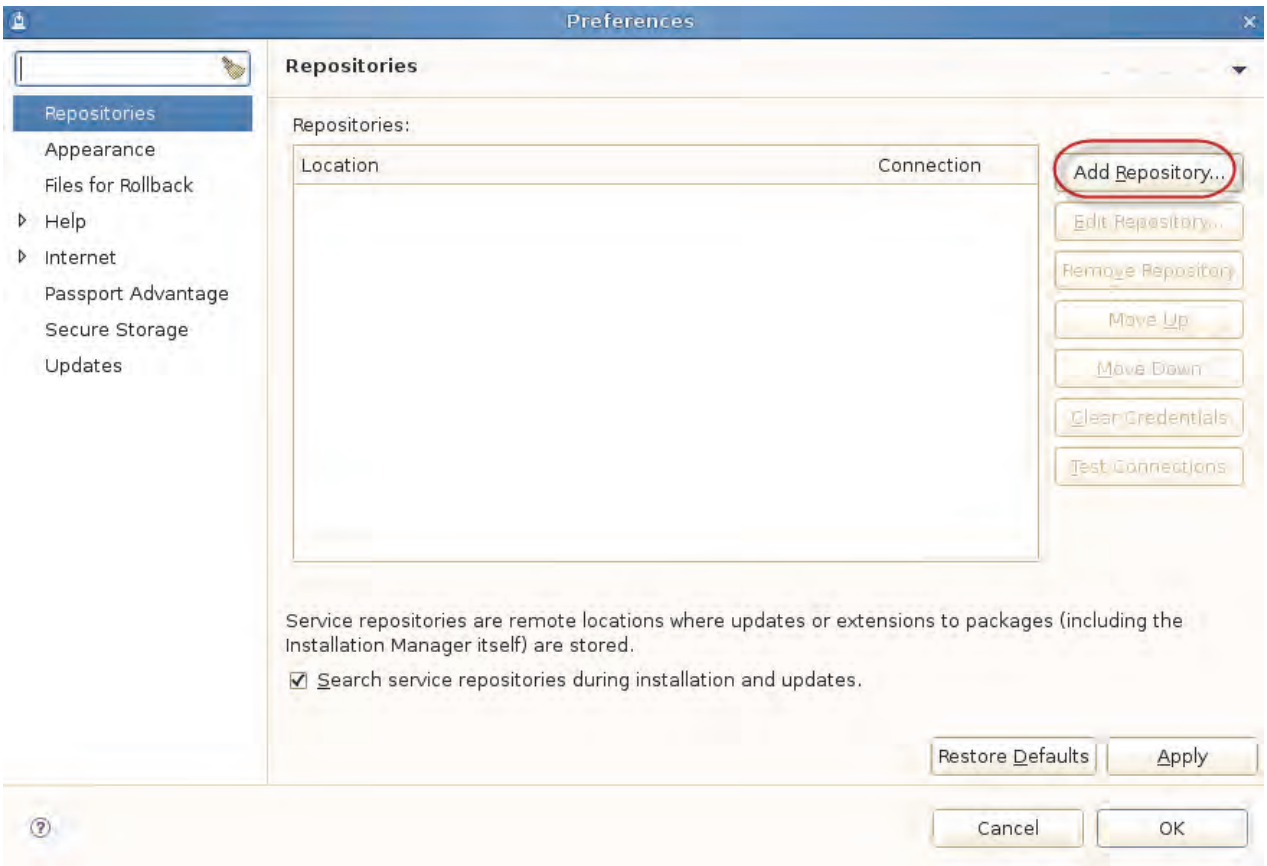
   ```
   netcool@jazz01:~/Desktop> cd /home/netcool/IBM/InstallationManager/eclipse
   netcool@jazz01:~/IBM/InstallationManager/eclipse> ./IBMIM
   ```

6. Configure the installation repositories. Select **File > Preferences**.

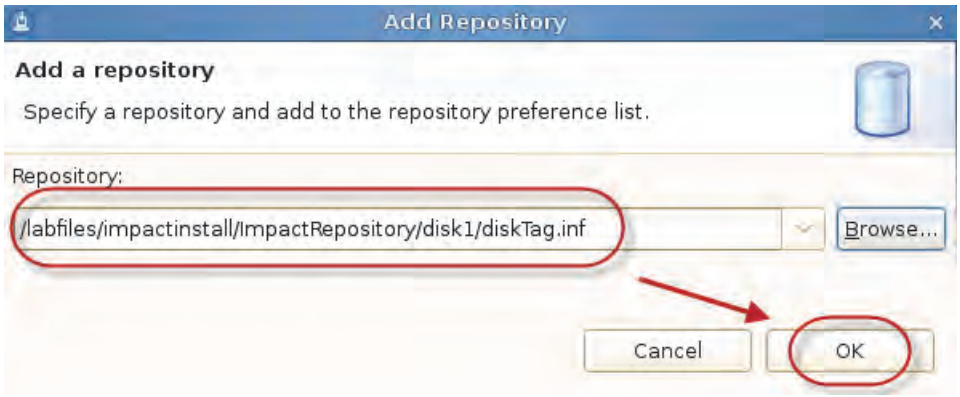


7. Add the repository for the Netcool/Impact 7.1.0.1 installation files. Click **Add Repository**.

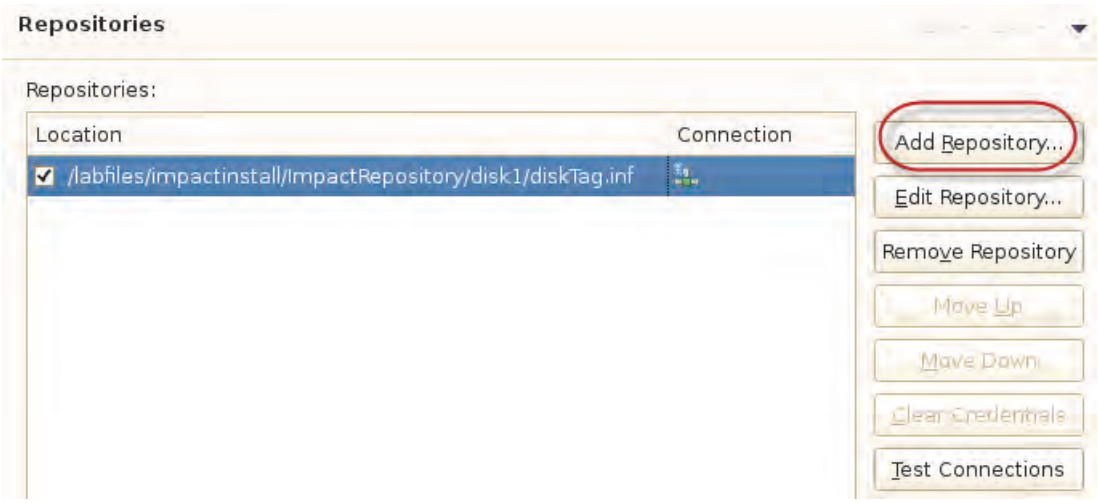
Important: Delete any existing repositories before adding the new ones.



8. Enter **/labfiles/impactinstall/ImpactRepository/disk1/diskTag.inf** in the **Repository** field and click **OK**.



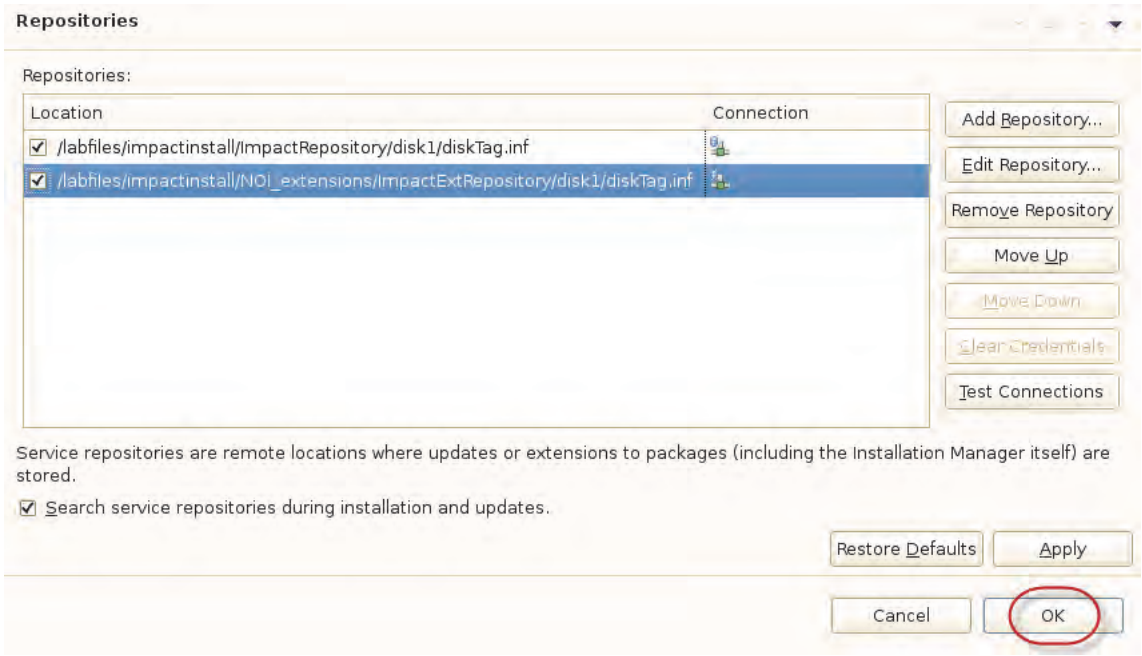
9. Add the repository for the Netcool/Impact 7.1.0.1 installation files. Click **Add Repository**.



10. Enter **/labfiles/impactinstall/NOI_extensions/ImpactExtRepository/disk1/diskTag.inf** in the **Repository** field and click **OK**.

    The two repositories are shown in the Repositories section.

11. Click **OK**.

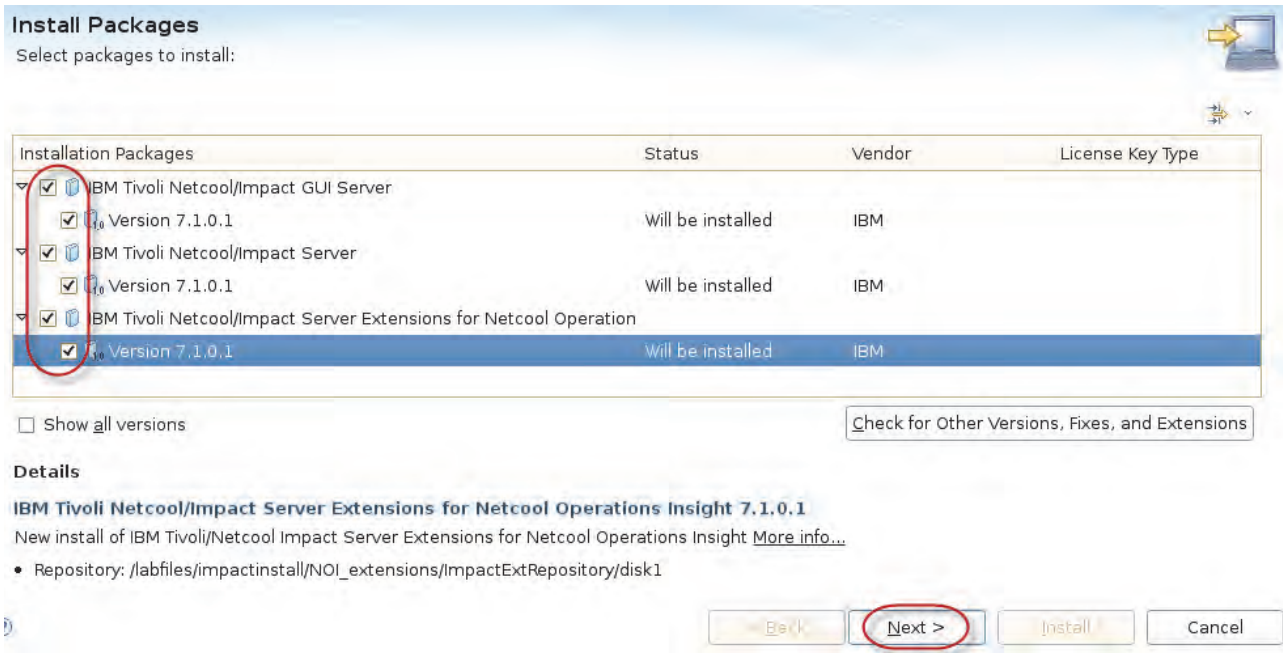


12. Click the icon to the left of **Install**.

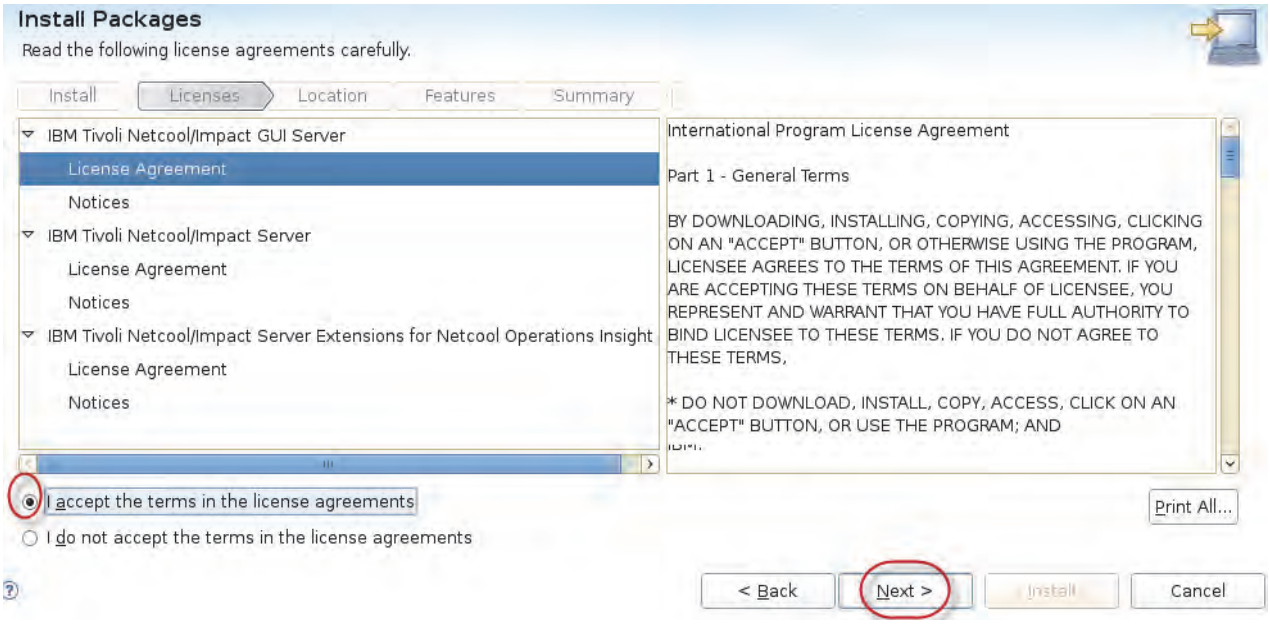
13. Select the following installation packages:
    - IBM Tivoli Netcool/Impact GUI Server
    - IBM Tivoli Netcool/Impact Server
    - IBM Tivoli Netcool/Impact Server Extensions for Netcool Operation

Note: The **Version 7.1.0.1** subentries are automatically selected as each installation package is selected.

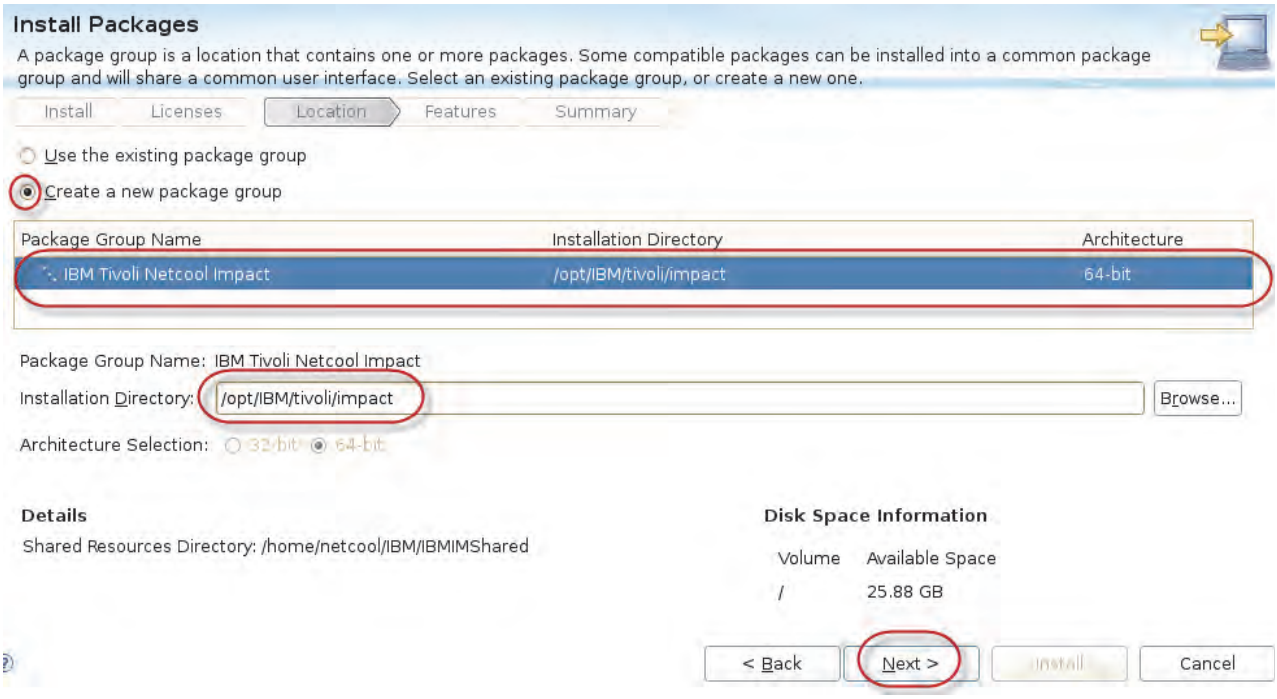
14. Click **Next**.



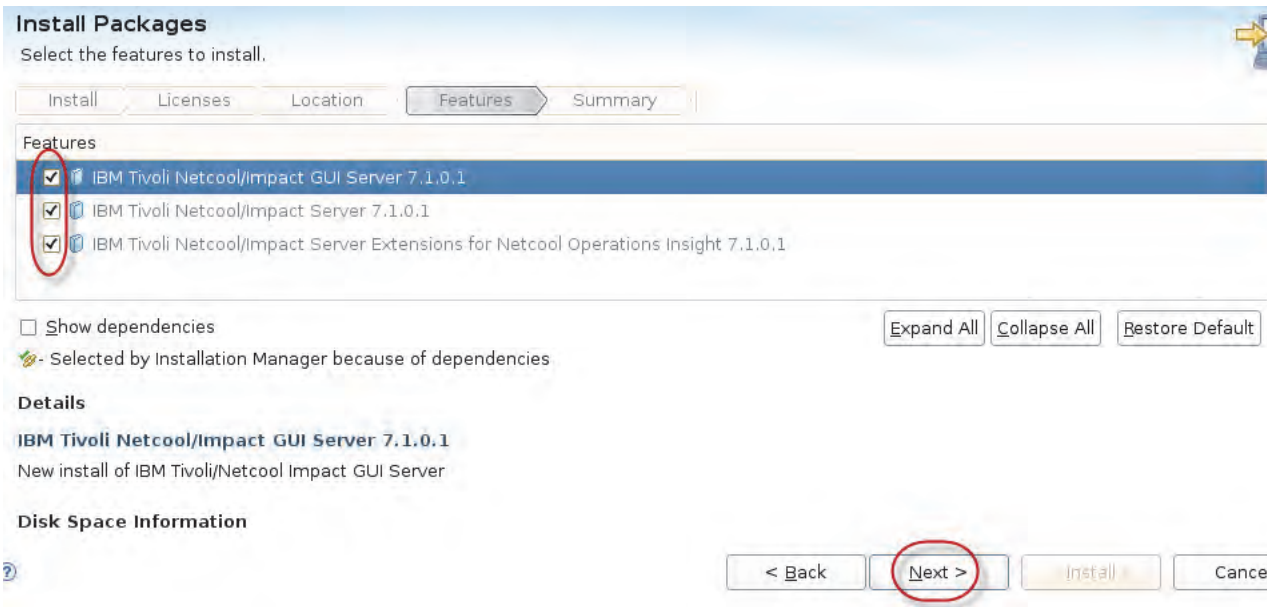
15. Select **I accept the terms in the license agreements** and click **Next**.



16. Use the default values for creating a new package group and the installation directory and click **Next**.



17. Verify that all features are selected and click **Next**.

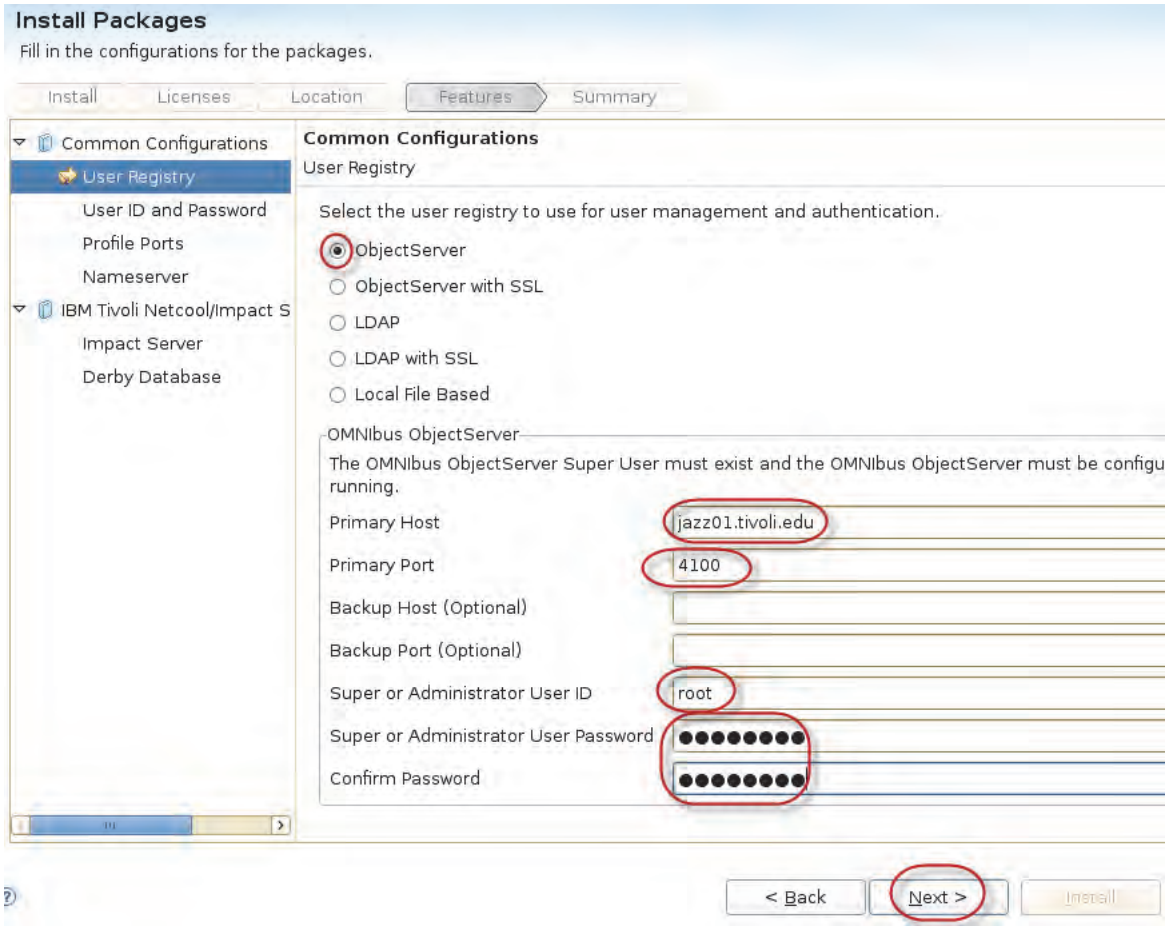


18. You must use the ObjectServer as the user registry. Enter **jazz01.tivoli.edu** in the **Primary Host** field.

19. Enter **4100** in the **Primary Port** field.

20. Enter **root** in the **Super or Administrator User ID** field.

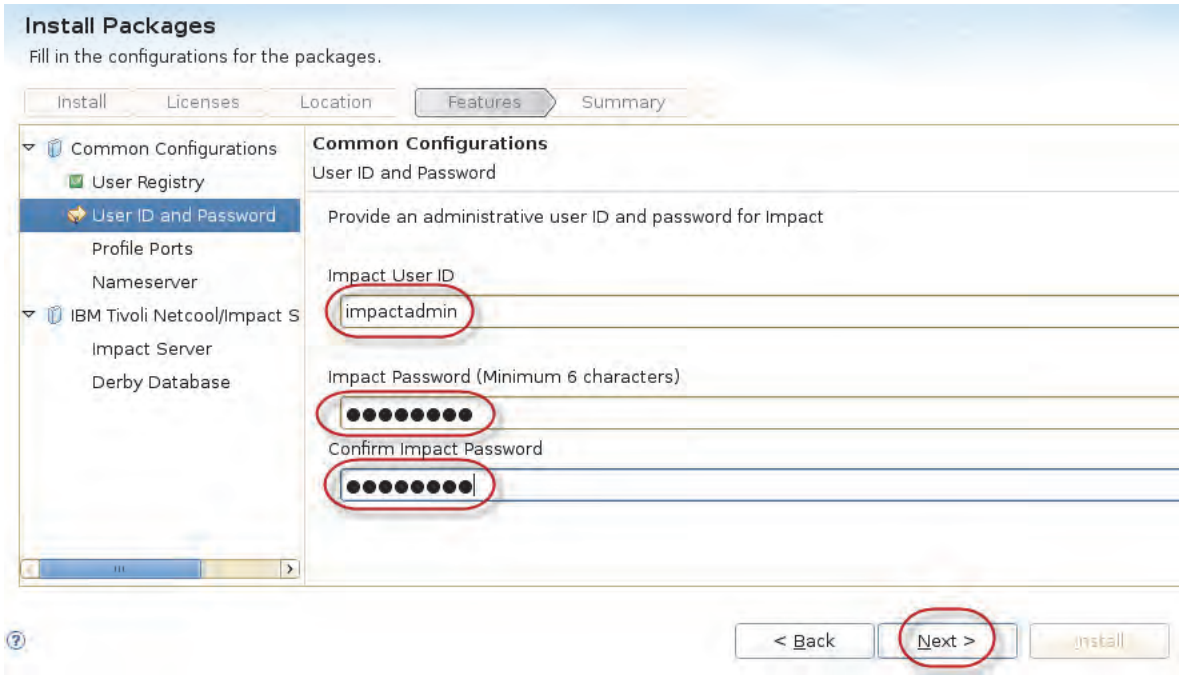
21. Enter **object00** in the password and password confirmation fields and click **Next**.



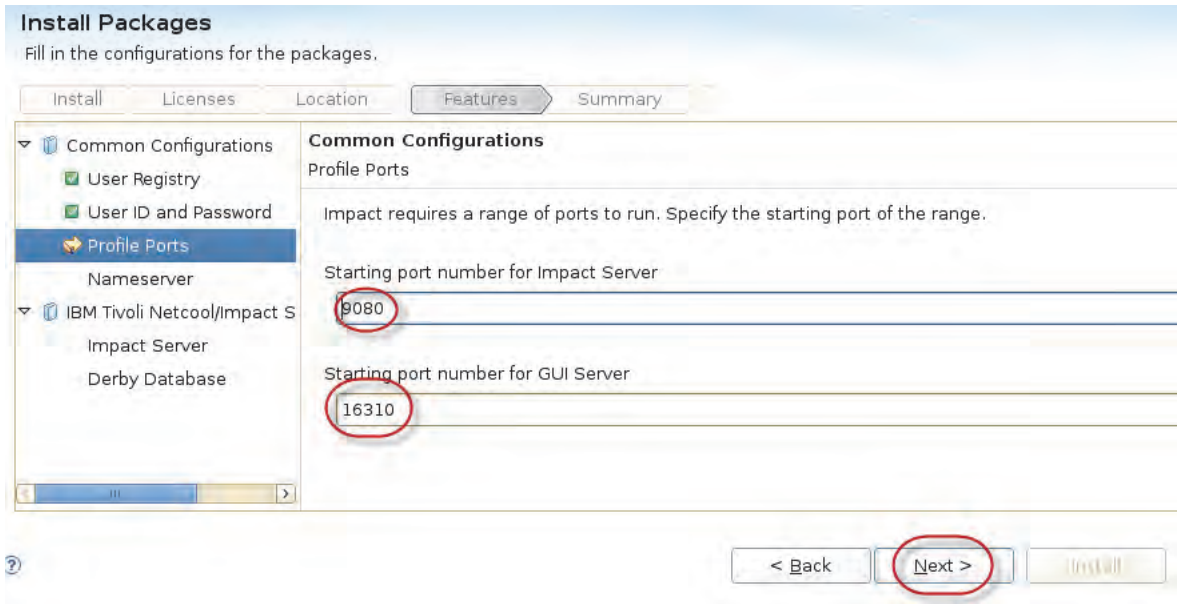
22. Enter **impactadmin** in the **Impact User ID** field.

    The user ID **impactadmin** was created in the ObjectServer repository during the image configuration.

23. Enter **object00** in the password and password confirmation fields and click **Next**.

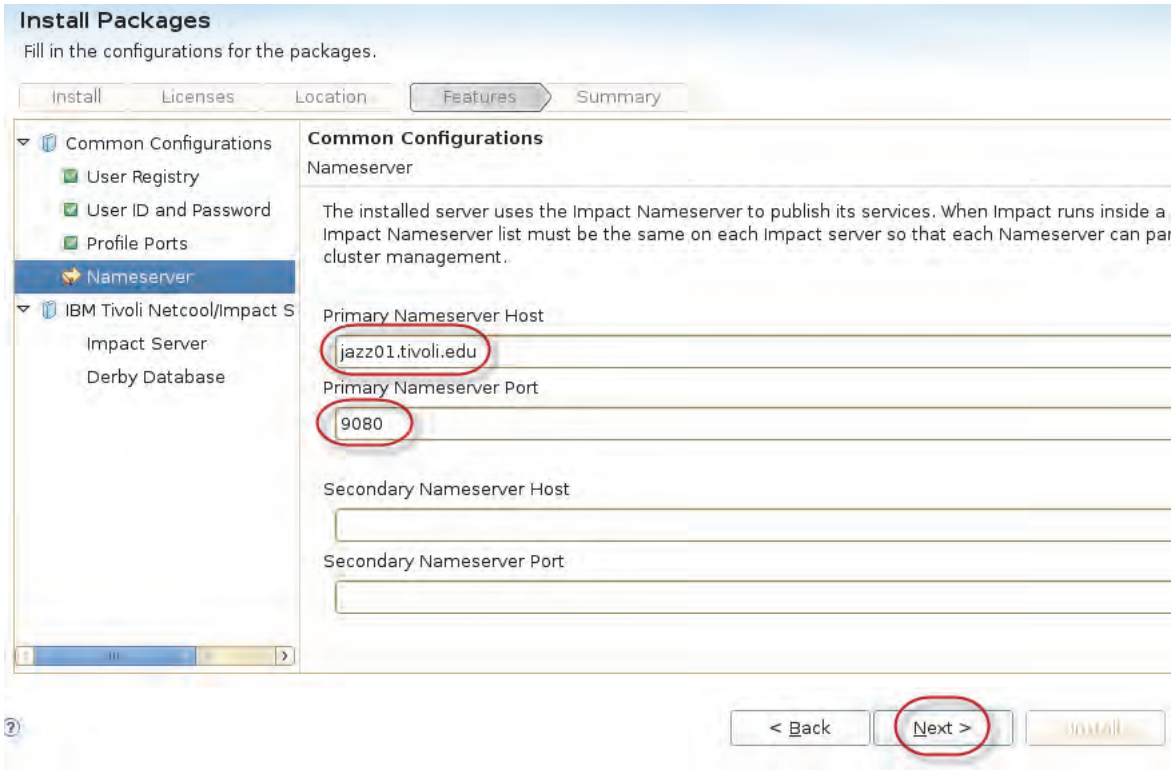


24. Use the default profile port values and click **Next**.

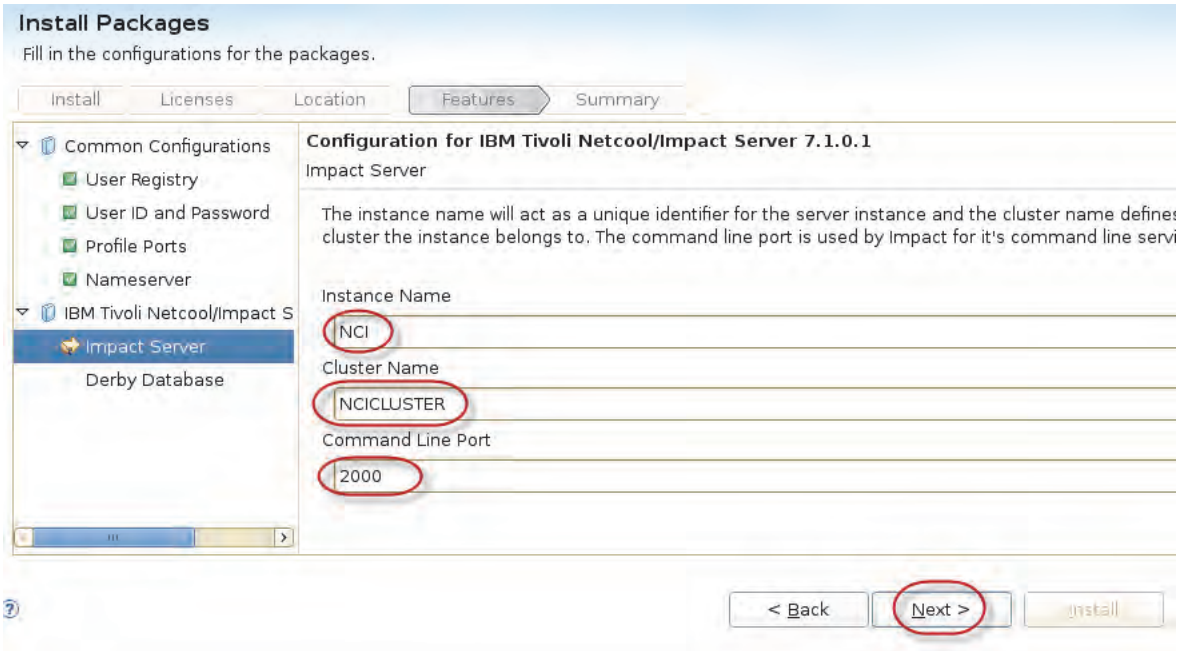


25. Enter **jazz01.tivoli.edu** in the **Primary Nameserver Host** field.

26. Enter **9080** in the **Primary Nameserver Port** field and click **Next**.



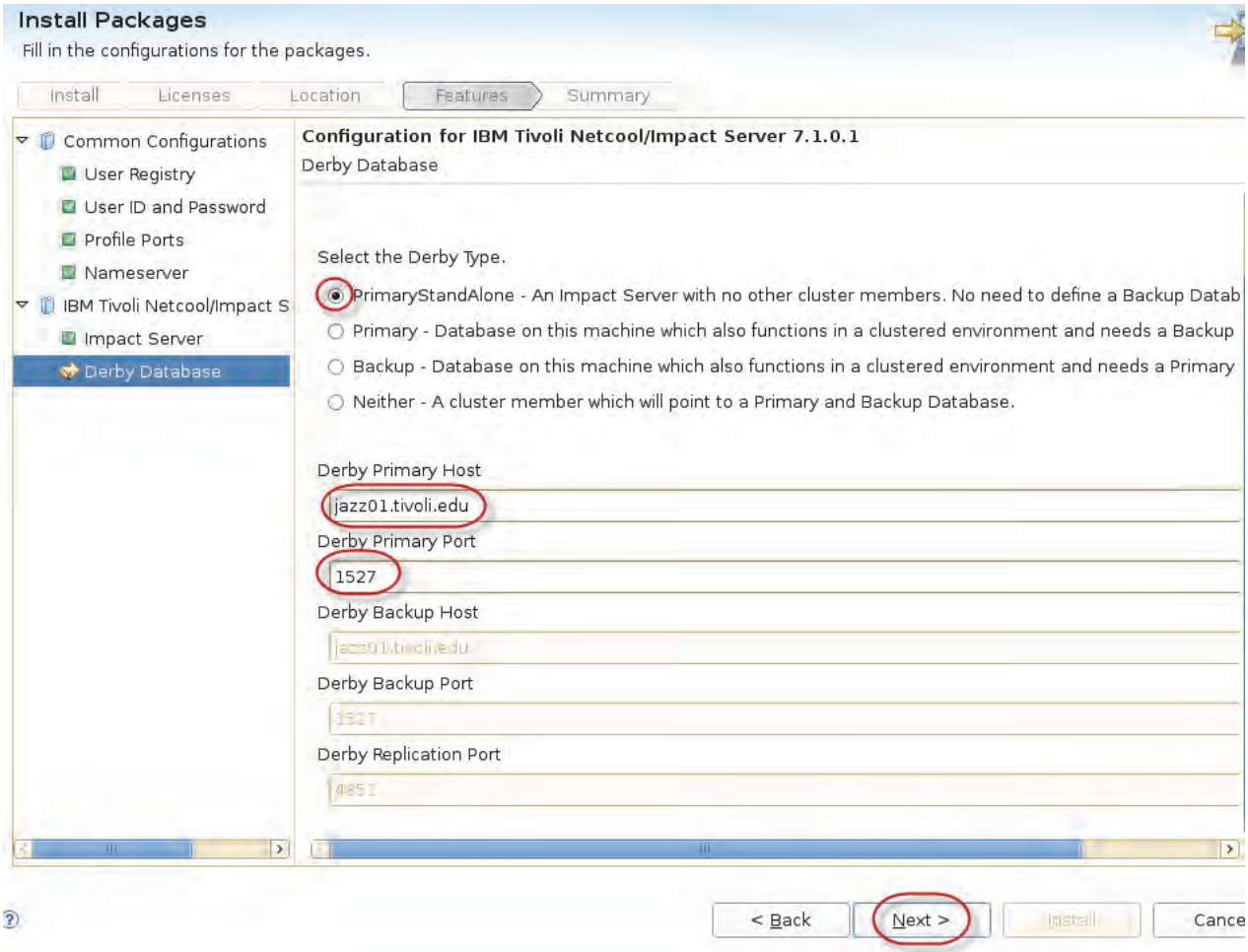
27. Use the default **Instance Name**, **Cluster Name**, and **Command Line Port** values and click **Next**.



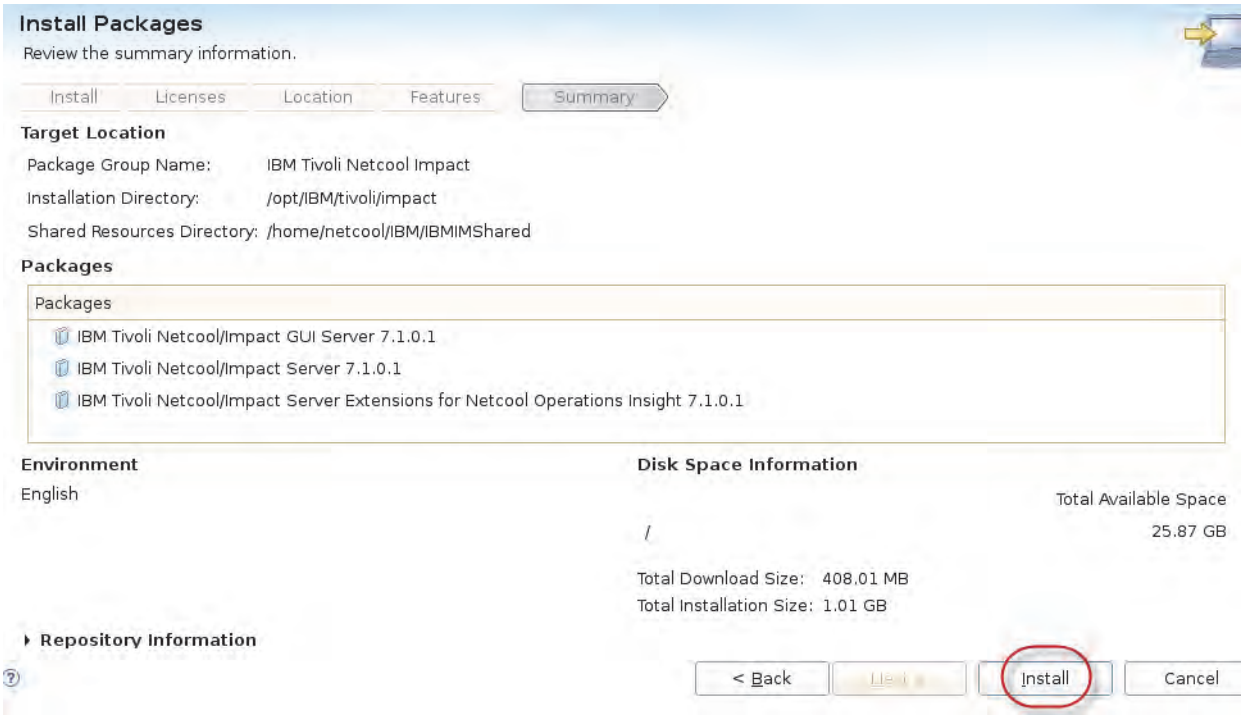
28. Select **PrimaryStandalone** in the **Derby Type** list.

29. Enter **jazz01.tivoli.edu** in the **Derby Primary Host** field.

30. Enter **1527** in the **Derby Primary Port** field and click **Next**.



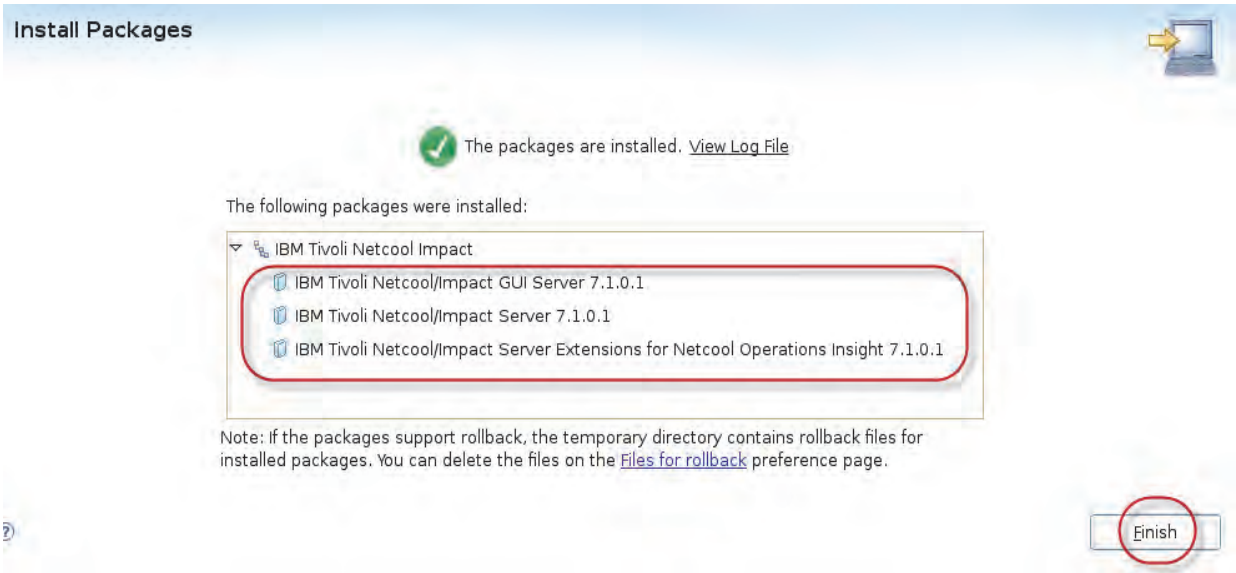
31. Review the installation package summary information and click **Install**.



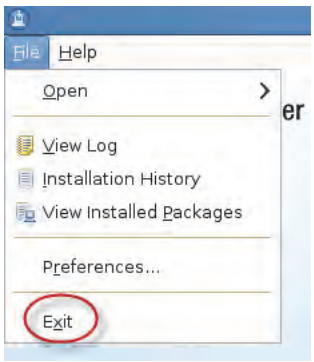
You see a progress bar that shows the component installation progress. Typical install time is approximately 40 minutes.

The Installation Manager tool shows a screen that indicates that all of the packages are successfully installed.

32. Click **Finish**.



33. Close the Installation Manager tool by selecting the **File > Exit** menu.



Exercise 2 Verifying the installation

1. Start the web browser application on the jazz01 virtual image. Click **Computer** in the lower-left of the image desktop and select **Firefox** in the **Applications** tab.

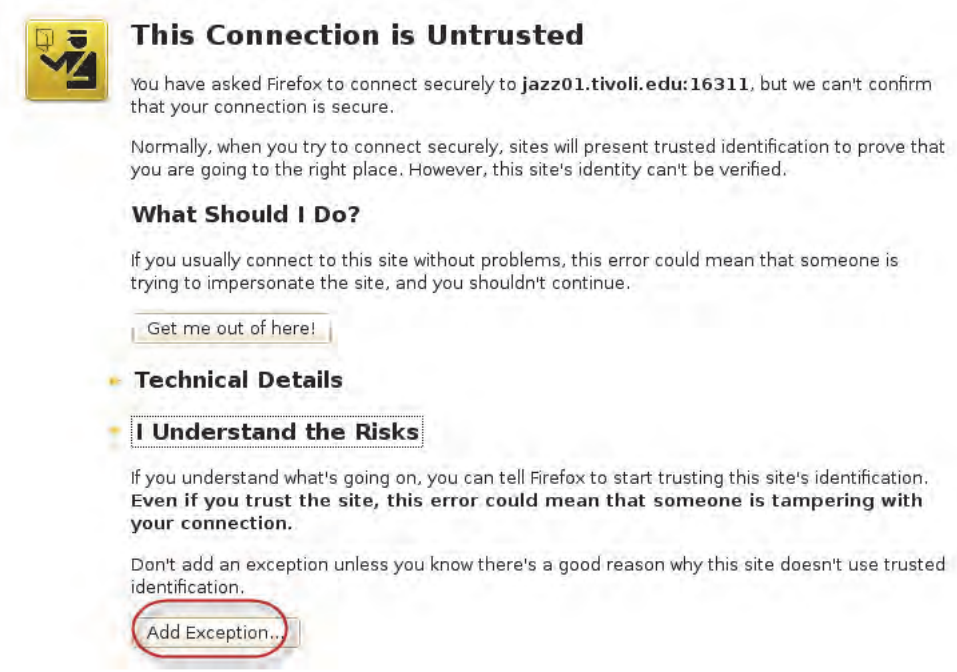
2. Enter the default Netcool/Impact web console address:

   `https://jazz01.tivoli.edu:16311/ibm/console/`

   The web browser opens a certificate warning window.

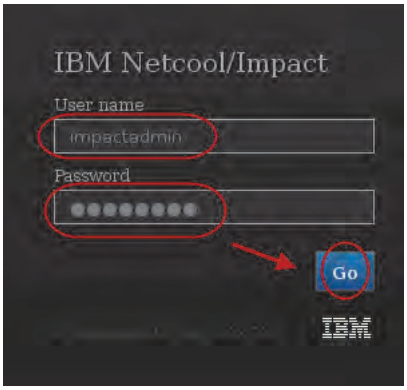
3. Click **I Understand the Risks**.

4. Click **Add Exception**.

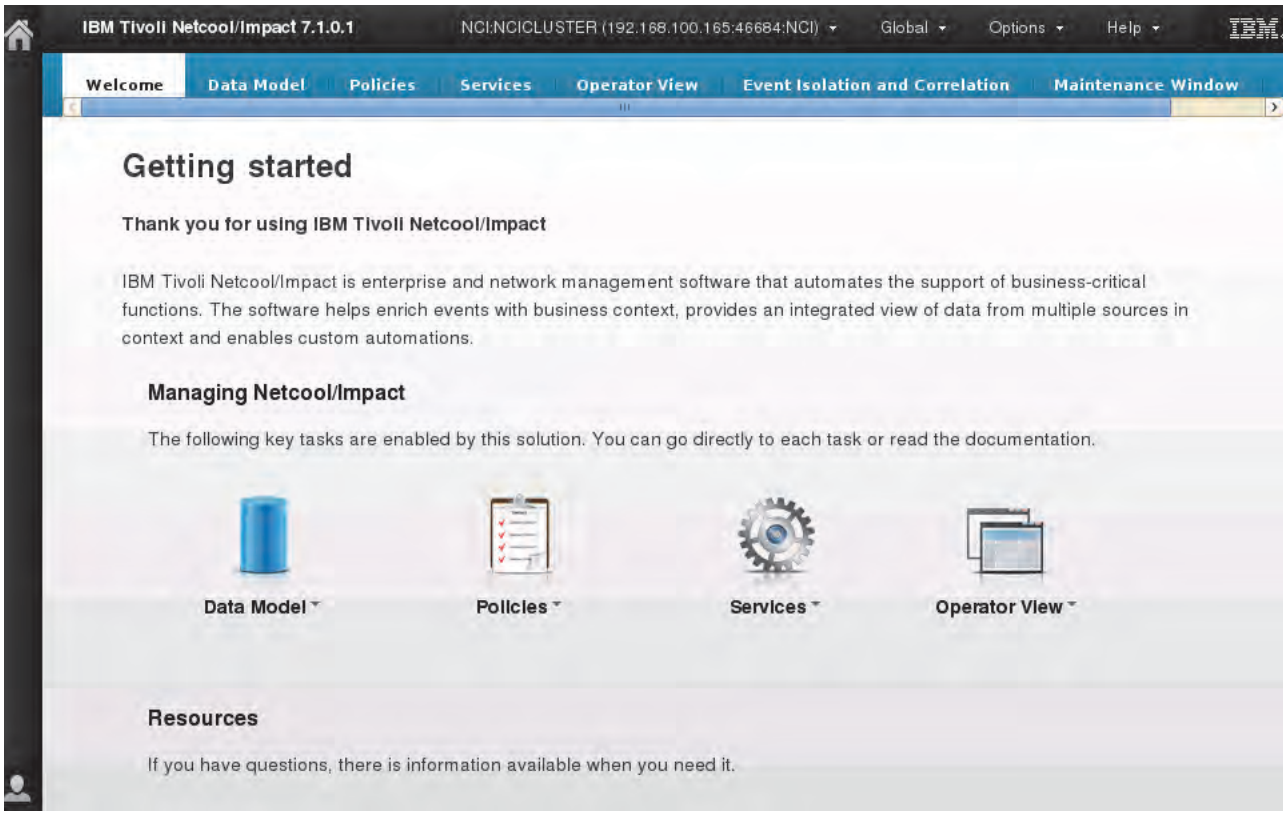


5. Click **Confirm Security Exception** to import and store the application certificate.

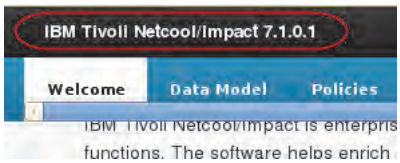
6. Log in to the Netcool/Impact console as the user **impactadmin**, with the password **object00**. Click **Go**.



You see the Getting Started page, indicating that the installation is successful.

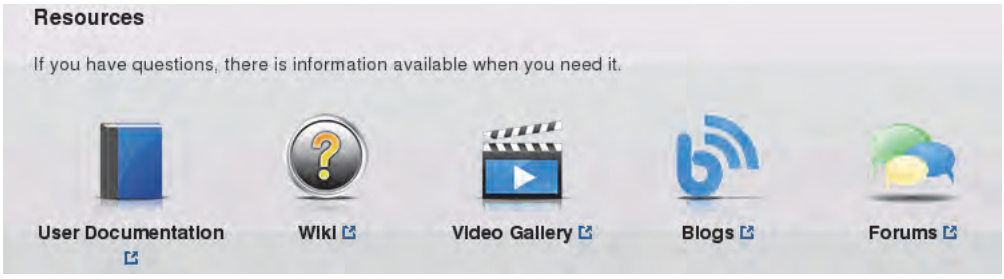


The application version is shown in the upper left of the console.

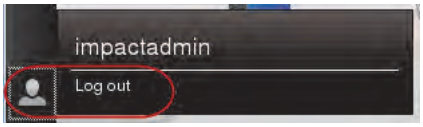


7. View links for the learning resources in the **Resources** section of the Welcome page.

Note: Some links require an Internet connection. The laboratory environment does not have Internet access.



8. Log off the console. Click the person icon in the lower left of the console and select **Logout**.



9. Close the browser. Select the **File > Quit** menu.

Exercise 3 Installing Fix Pack 5

1. Change to the Installation Manager directory:

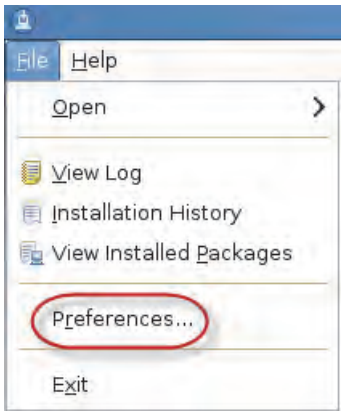
   ```
   cd /home/netcool/IBM/InstallationManager/eclipse
   ```

2. Use the following command to start the Installation Manager tool:

   ```
   ./IBMIM
   ```

   ```
   netcool@jazz01:~/Desktop> cd /home/netcool/IBM/InstallationManager/eclipse
   netcool@jazz01:~/IBM/InstallationManager/eclipse> ./IBMIM
   ```

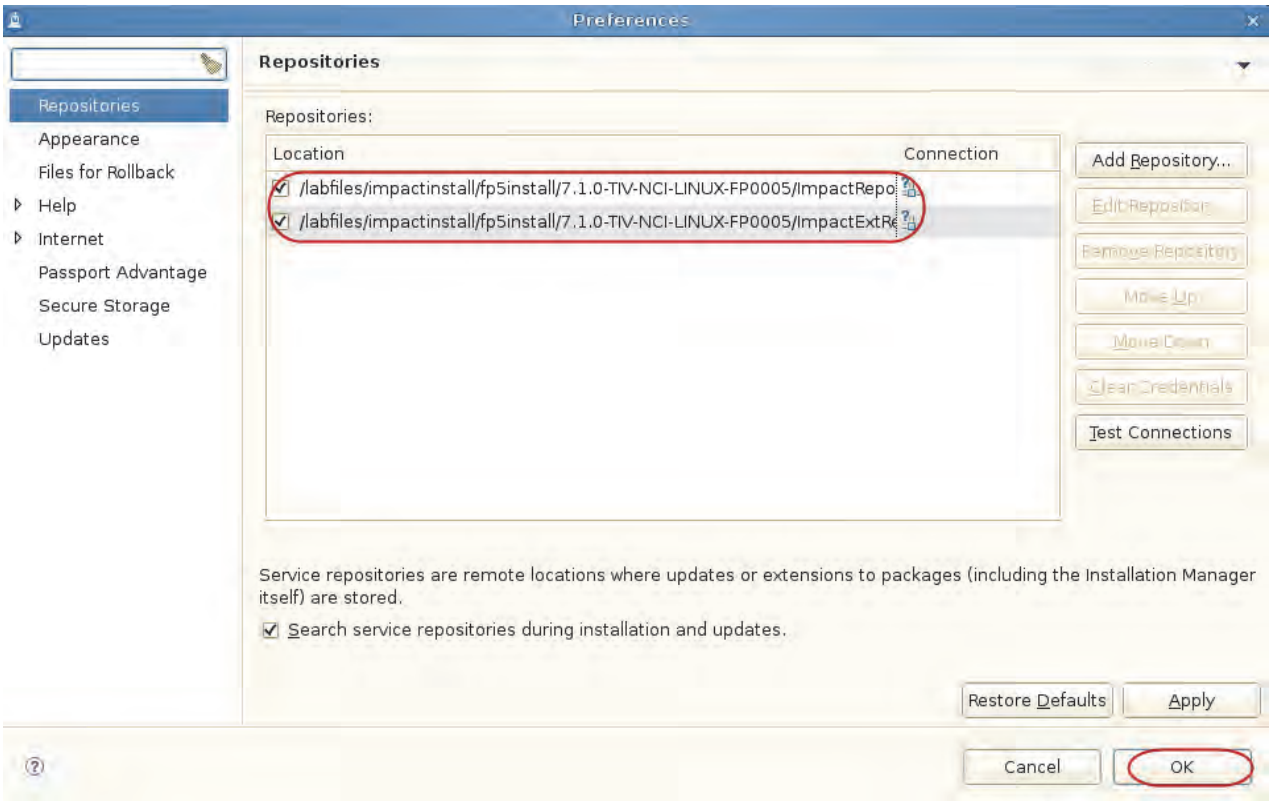
3. Configure the installation repositories. Select **File > Preferences**.



4. Add the repositories for the Netcool/Impact fp5 installation files. Click **Add Repository**.

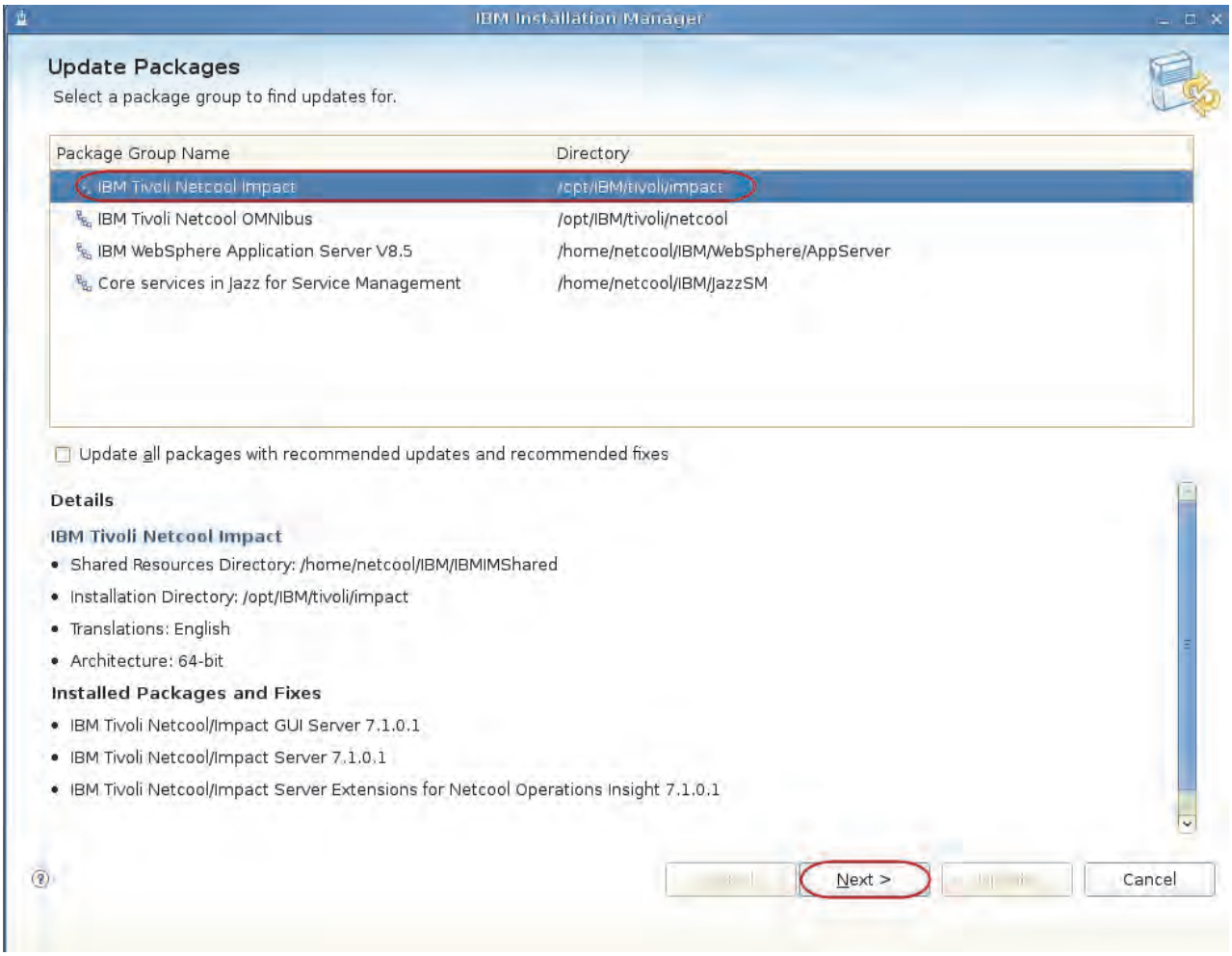
/labfiles/impactinstall/fp5install/ImpactRepository/repository.config
/labfiles/impactinstall/fp5install/ImpactExtRepository/repository.config

5. The two repositories are shown in the Repositories section. Click **OK**.



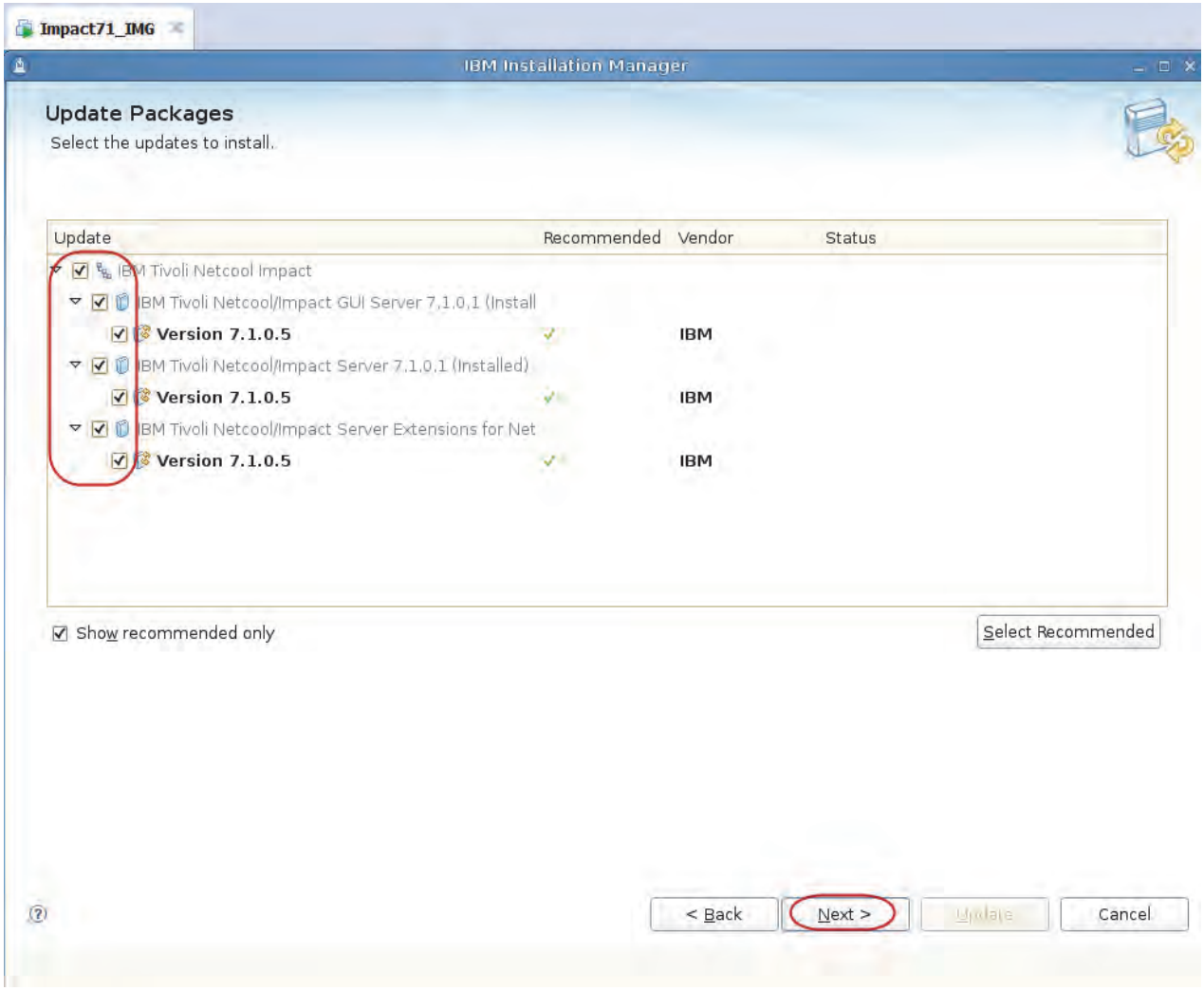
6. Click the icon to the left of **Update**.

7. Select the package to update:
   - IBM Tivoli Netcool/Impact

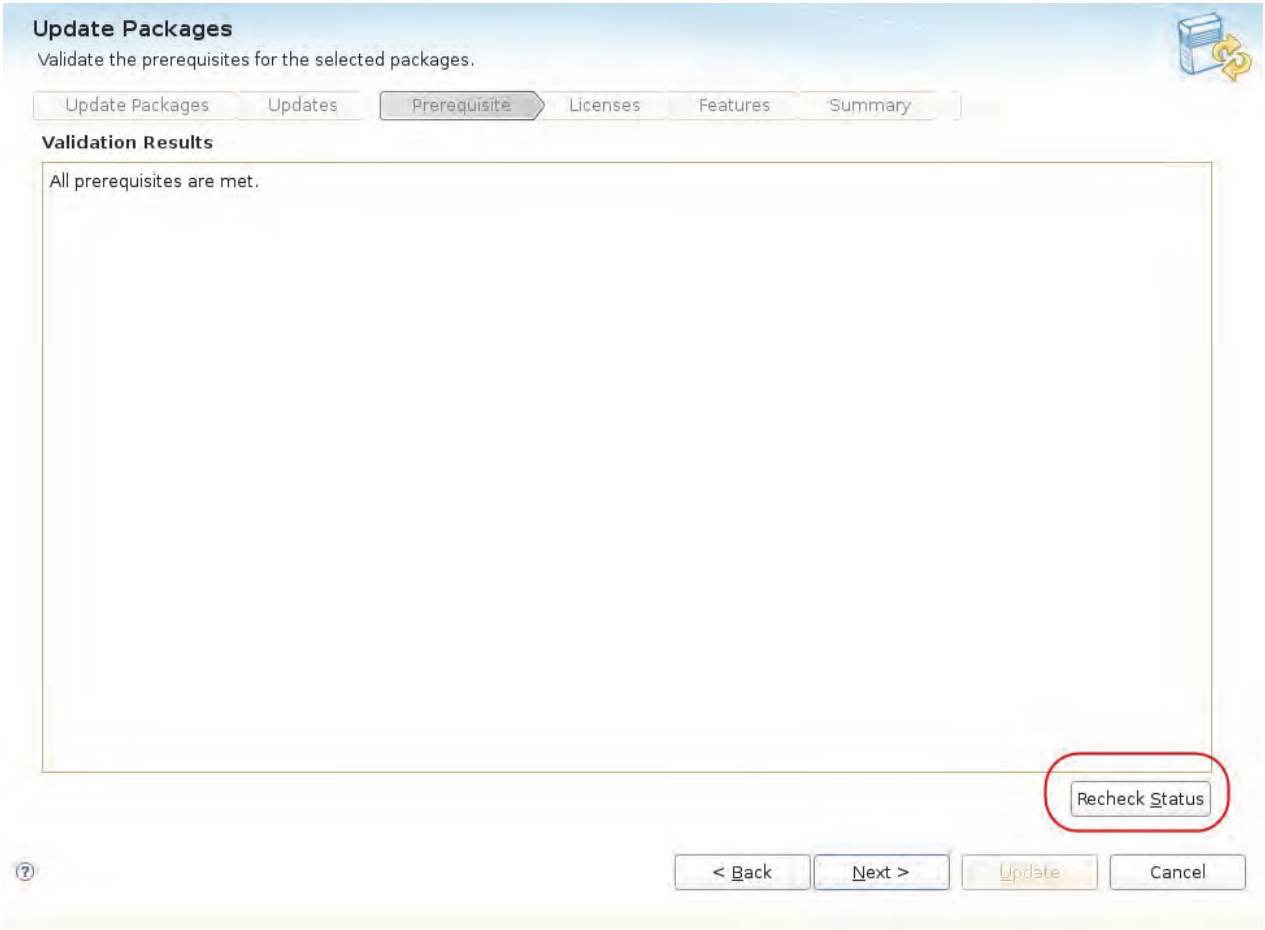


8. Select the following installation packages:
   - IBM Tivoli Netcool/Impact GUI Server
   - IBM Tivoli Netcool/Impact Server
   - IBM Tivoli Netcool/Impact Server Extensions for Netcool Operations Insight

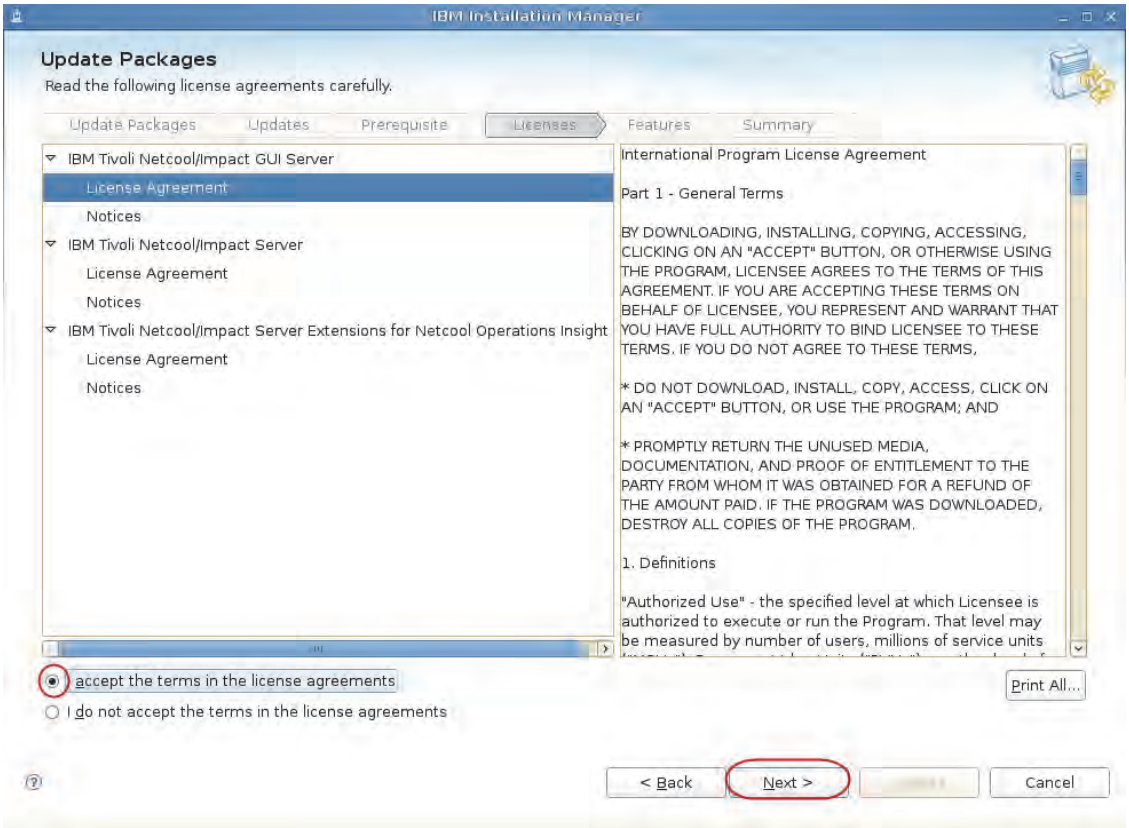
9. Click **Next**.



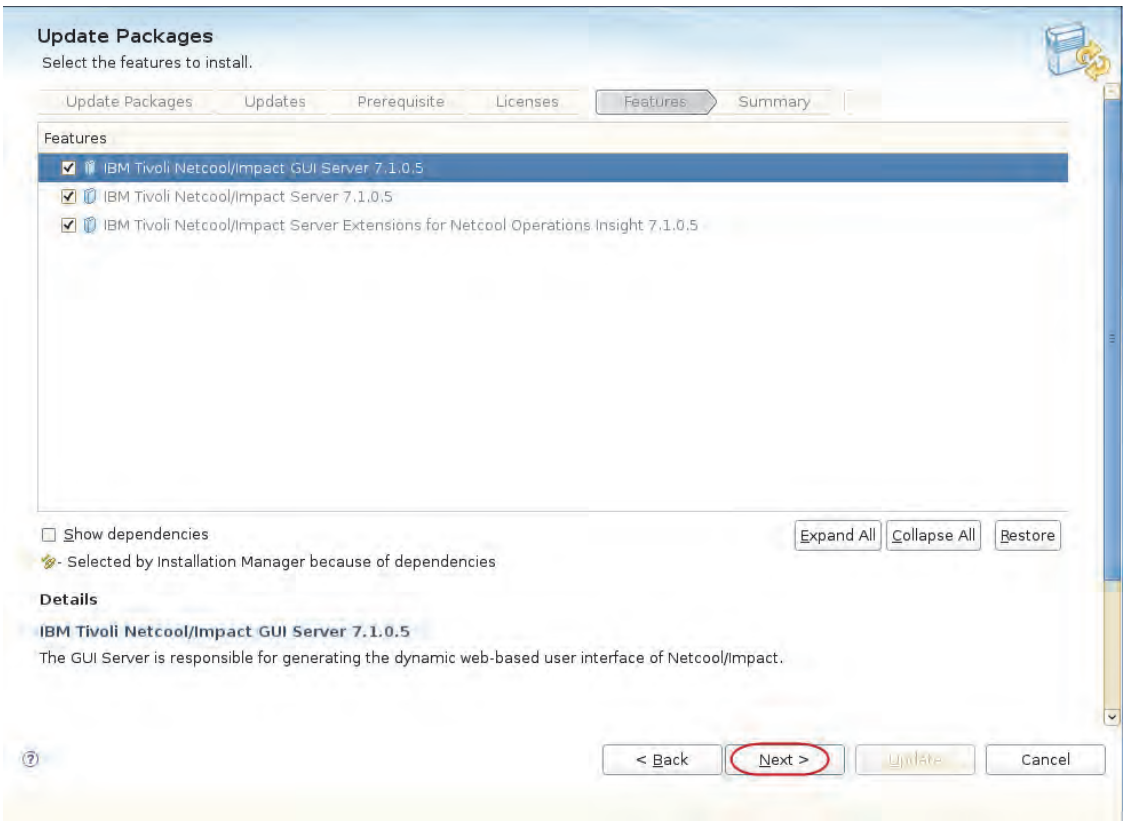
10. Click **Recheck Status** to verify prerequisites.



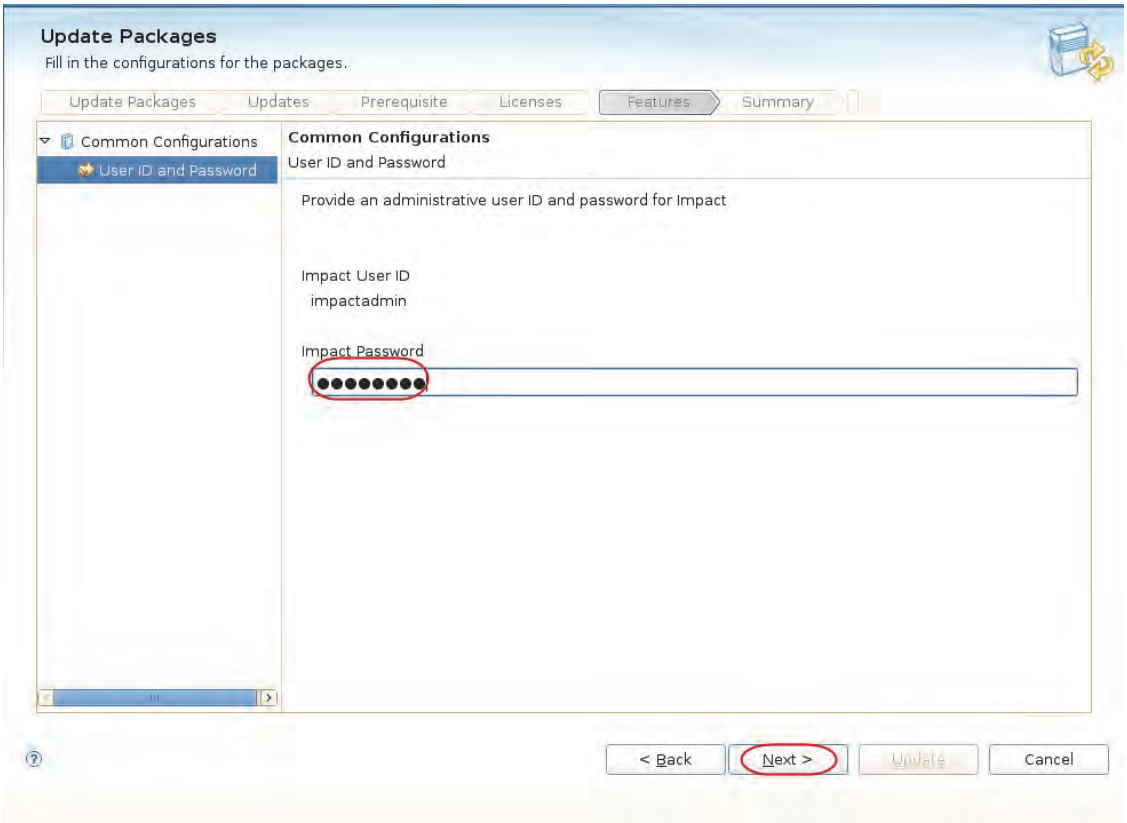
11. Select **accept the terms in the license agreements** and click **Next**.



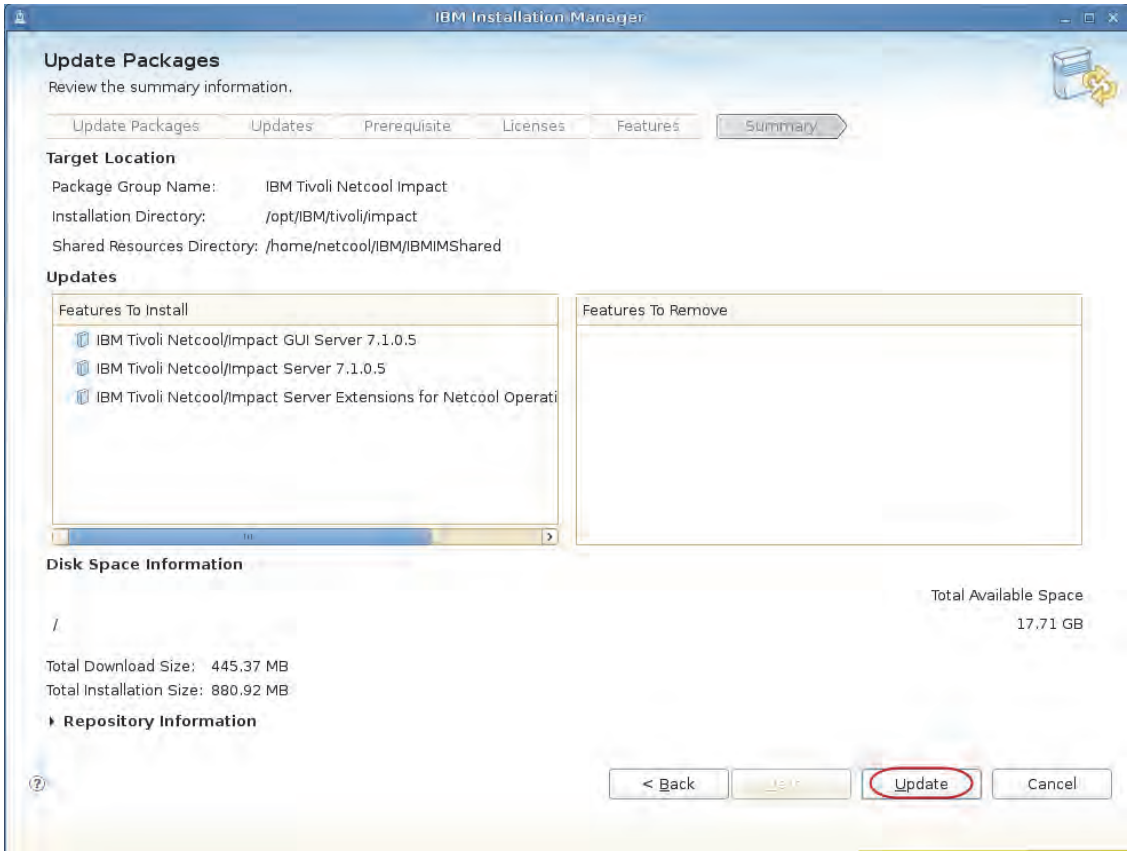
12. Verify that three features are selected. Click **Next**.



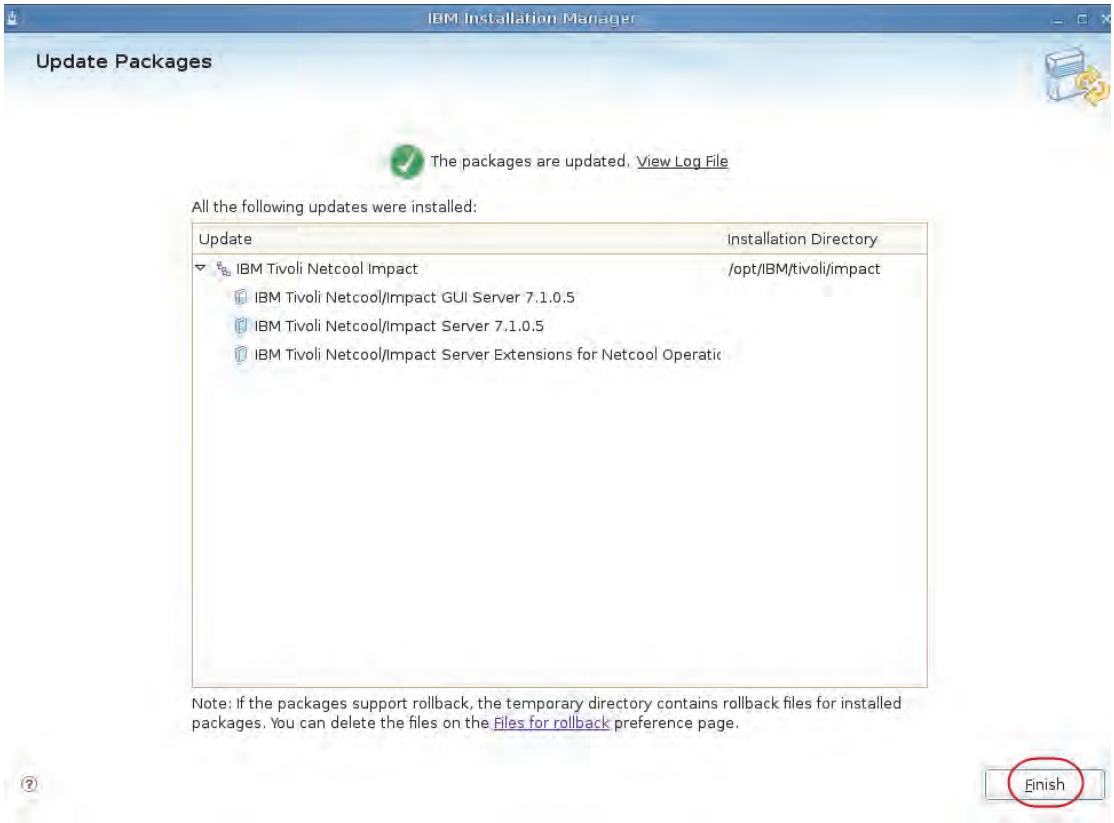
13. Type **object00** in the Impact Password field. Click **Next**.



14. Click **Update** to install fp5.

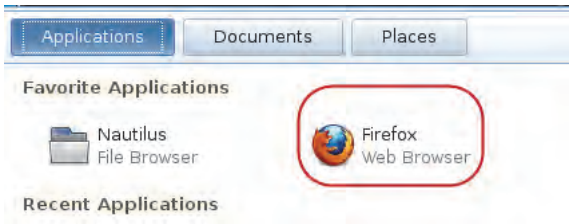


15. Verify that the update completed successfully. Click **Finish**.



Exercise 4 Verifying Fix Pack installation

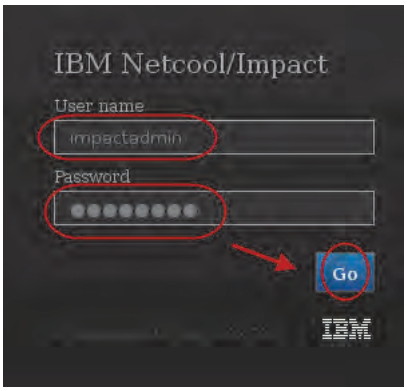
1. Select **Firefox** in the **Applications** tab.



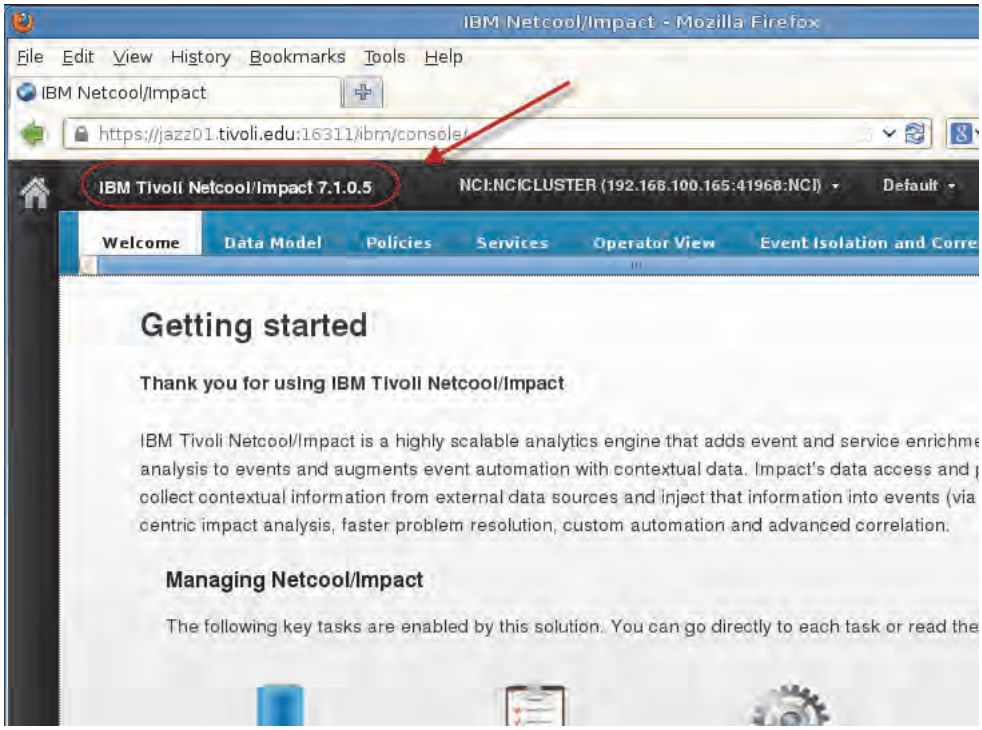
2. Enter the default Netcool/Impact web console address:

   `https://jazz01.tivoli.edu:16311/ibm/console/`

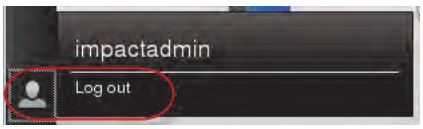
3. Log in to the Netcool/Impact console as the user **impactadmin**, with the password **object00**. Click **Go**.



4. Notice that the product version changed from 7.1.0.1 to 7.1.0.5.



5. Exit Impact. Click the person icon in the lower left side of the console and select **Logout**.



6. Close the browser. Select the **File > Quit** from the menu.

Exercise 5 Initializing the Derby database

Use the Derby database as a data source. The next few steps prepare the database for class exercises.

1. The initialization files, **Impact71DERBYSetup.sql** and **OVTABLESETUP.sql**, are in the **/labfiles** directory. Enter the following commands to move the files to a Netcool/Impact directory:

   ```
   cd /labfiles
   cp Impact71DERBYSetup.sql /opt/IBM/tivoli/impact/bin
   cp OVTABLESETUP_DERBY.sql /opt/IBM/tivoli/impact/bin
   ```

2. Change directory to **/opt/IBM/tivoli/impact/bin** and enter the following command:

   ```
   ./nci_db connect
   ```

3. At the prompt, type the following command:

connect 'jdbc:derby://jazz01.tivoli.edu:1527/ImpactDB;user=impact;password=derbypass';

```
netcool@jazz01:/opt/IBM/tivoli/impact/bin> ./nci_db connect
ij version 10.8
ij> connect 'jdbc:derby://jazz01.tivoli.edu:1527/ImpactDB;user=impact;password=d
erbypass';
ij>
```

4. Run the script by entering the following command at the **ij>** prompt:

   ```
   run 'Impact71DERBYSetup.sql';
   ```

5. Verify the script operations by entering the following command at the **ij>** prompt:

   ```
   select * from inventory;
   ```

```
ij> select * from inventory;
PARTID                         |NODE                           |INTERFACE
                  |STATUS     |TYPE                           |VERSION

--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
----------------
DAL-99e309                     |gambit                         |
                  |1          |Cisco                          |IOS 12.0

DAL-93488f                     |orac                           |1
                  |1          |Sun                            |Freeware

DAL-934r588f                   |wombat                         |1
                  |1          |Sun                            |Freeware

DAL-934r88f                    |muppet                         |1
                  |1          |Sun                            |Freeware

DAL-9345488f                   |moose                          |1
                  |1          |Sun                            |Freeware
```

6. Start the second script at the **ij>** prompt by entering the following command:

```
run 'OVTABLESETUP_DERBY.sql';
```

7. Verify the script operations by entering the following command at the **ij>** prompt:

```
select * from ASSET;
```

```
ij> select * from ASSET;
DEVICE                                                                  |CUSTOMER
                                                        |CUSTCONTACT
                                        |CCPHONE          |CCEMAIL
                                          |OWNER
                        |OPHONE            |OEMAIL

------------------------------------------------------------------------------
------------------------------------------------------------------------------
------------------------------------------------------------------------------
------------------------------------------------------------------------------
------------------------------------------------------------------------------
-----------------------
192.168.1.119                                                           |ACME TELCOM
                                                        |Server Adminperson
                                        |212-555-3331     |saadmin@vacmtel.com
                                          |Device Ownerperson
                        |609-555-1234     |downer@acmtel.com


1 row selected
ij>
```

8. Close the Derby database command interpreter by entering the following command:

```
quit;
```

Exercise 6 Modifying the OMNIbus Object server

1. Confirm that the OMNIbus ObjectServer server is started.
2. Change directory to the following location:

   `/opt/IBM/tivoli/netcool/omnibus/bin/`
3. Enter either of the following commands:

   `./nco_ping NCOMS`

   or

   `ps -ef | grep nco`
4. If the OMNIbus server is *not* running, start OMNIbus with the following command:

   `/opt/IBM/tivoli/netcool/omnibus/bin/nco_objserv NCOMS &`
5. Use the **nco_ping NCOMS** command to verify that the ObjectServer is running.
6. Verify that the **addfields.sql** script exists in the **/labfiles** directory. Enter the following command:

   `cat /labfiles/addfields.sql`

```
netcool@jazz01:/opt/IBM/tivoli/impact/bin> cat /labfiles/addfields.sql
ALTER TABLE alerts.status ADD COLUMN PartNumber VARCHAR(64);
ALTER TABLE alerts.status ADD COLUMN Department VARCHAR(64);
ALTER TABLE alerts.status ADD COLUMN SLA VARCHAR(64);
ALTER TABLE alerts.status ADD COLUMN WorkingHours VARCHAR(64);
ALTER TABLE alerts.status ADD COLUMN ImpactFlag1 INTEGER;
ALTER TABLE alerts.status ADD COLUMN ImpactFlag2 INTEGER;
ALTER TABLE alerts.status ADD COLUMN ImpactFlag3 INTEGER;
ALTER TABLE alerts.status ADD COLUMN ImpactServiceFlag INTEGER;
ALTER TABLE alerts.status ADD COLUMN IMSent INTEGER;
go
```

7. Change directory to **/opt/IBM/tivoli/netcool/omnibus/bin**.
8. Start the **addfields.sql** script with the following command:

`./nco_sql -server NCOMS -user root -password 'object00' < /labfiles/addfields.sql`

```
netcool@jazz01:/opt/IBM/tivoli/impact/bin> cd /opt/IBM/tivoli/netcool/omnibus/bi
n
netcool@jazz01:/opt/IBM/tivoli/netcool/omnibus/bin> ./nco_sql -server NCOMS -use
r root -password 'object00' < /labfiles/addfields.sql
(0 rows affected)
```

The message **0 rows affected** indicates that the script commands were processed.

Note: If the nco_sql command indicates errors in the file, use the command **dos2unix addfields.sql** to convert the text file to the proper format.

Unit 2 The Netcool/Impact user interface exercises

This unit focuses on starting components, stopping components, and using projects.

Exercise 1 Starting the Netcool/Impact components

1. To confirm that the OMNIbus server is running, change to the following directory:

   ```
   cd /opt/IBM/tivoli/netcool/omnibus/bin/
   ```

2. Enter either of the following commands:

   ```
   ./nco_ping NCOMS
   ```

   OR

   ```
   ps -ef | grep nco
   ```

3. If the OMNIbus server is *not* started, enter the following command:

   ```
   /opt/IBM/tivoli/netcool/omnibus/bin/nco_objserv -name NCOMS &
   ```

4. Use the **nco_ping NCOMS** command to verify that the object server is running.

5. Verify the Netcool/Impact server, enter the following command:

   ```
   /opt/IBM/tivoli/impact/bin/startImpactServer.sh
   ```

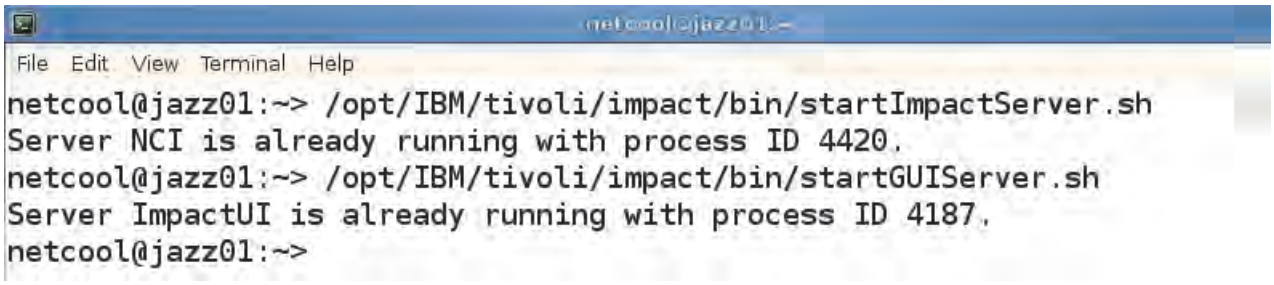
```
netcool@jazz01:/opt/IBM/tivoli/impact/bin> /opt/IBM/tivoli/impact/bin/startImpactSe
rver.sh
Starting server NCI.
Server NCI started with process ID 11295.
Started Impact Server NCI
```

6. To start the Netcool/Impact GUI server, enter the following command:

   ```
   /opt/IBM/tivoli/impact/bin/startGUIServer.sh
   ```

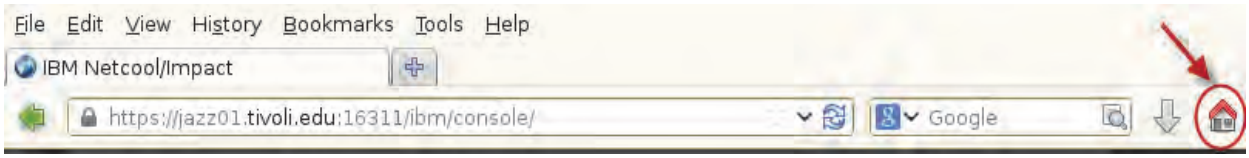
```
netcool@jazz01:/opt/IBM/tivoli/impact/bin> /opt/IBM/tivoli/impact/bin/startGUIServe
r.sh
Starting server ImpactUI.
Server ImpactUI started with process ID 11694.
Started GUI Server ImpactUI
```

If the servers are already running the commands will display the process ID similar to the following:

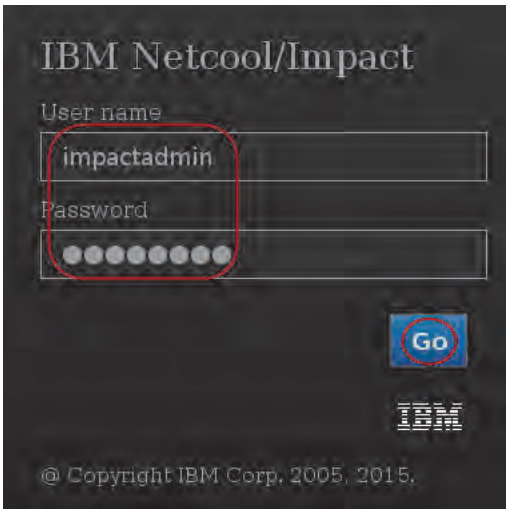


Exercise 2 Creating a project

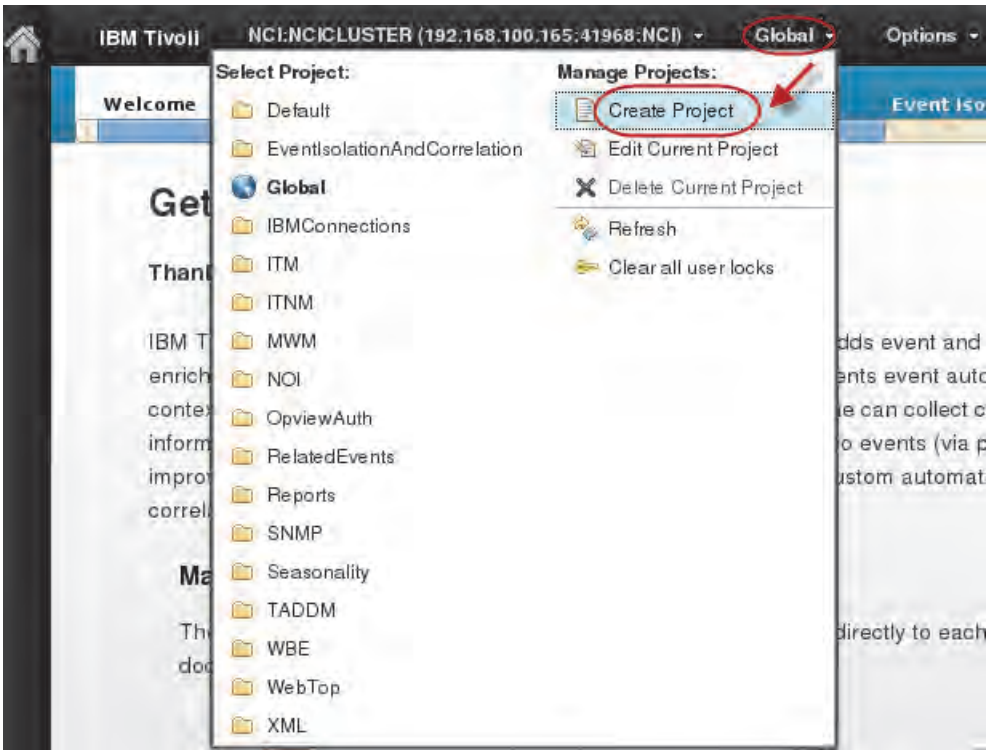
1. Open the Firefox web browser to log into the Netcool/Impact console. Start the browser and click the home page icon in the upper right.



2. Enter **impactadmin** in the **User name** field,**object00** in the **Password** field, and click **Go**.



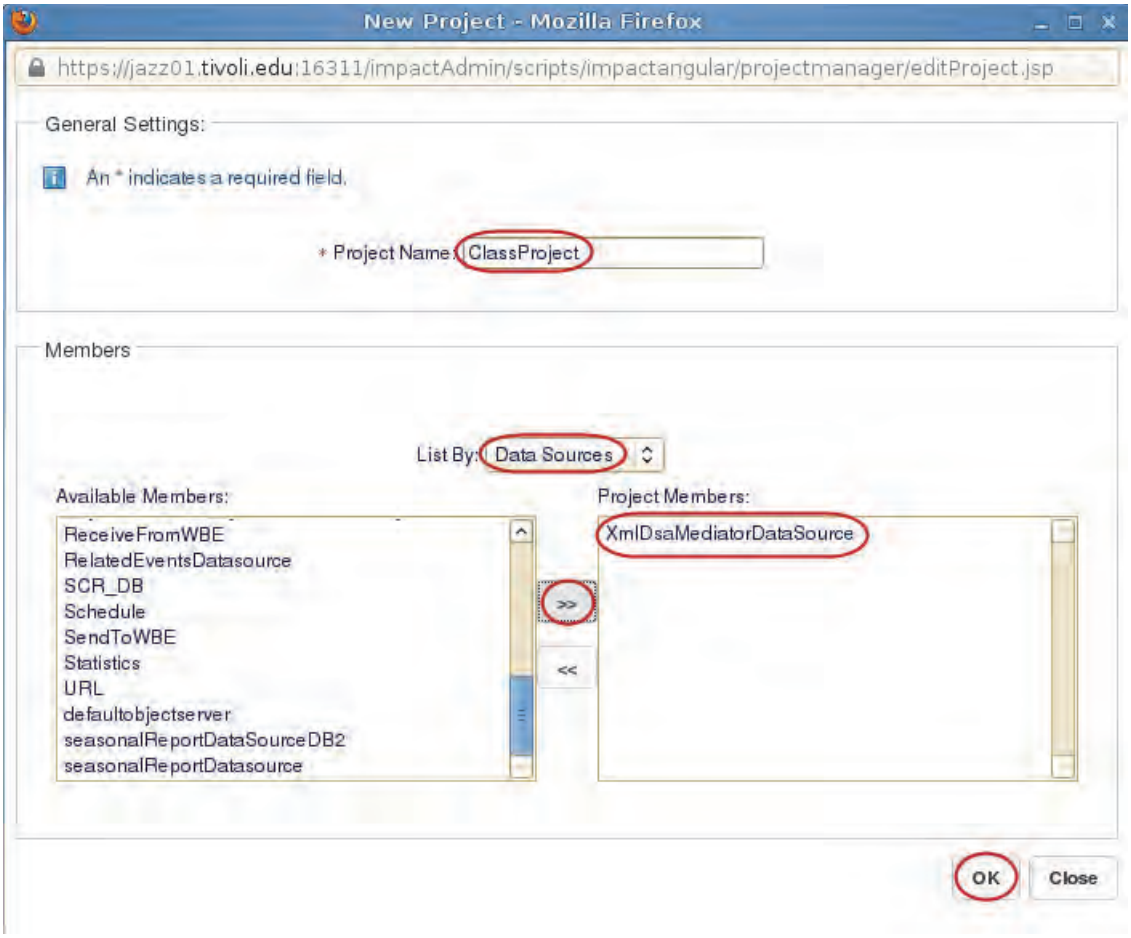
3. Create a Netcool/Impact project. Click the project list menu at the top of the console and click **Create Project** in the **Manage Projects** section.



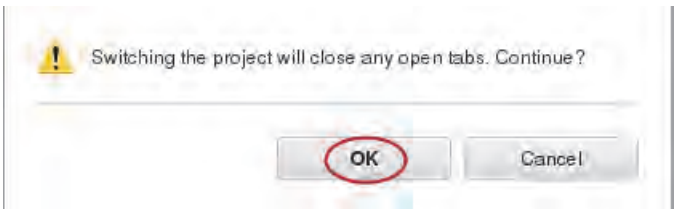
A project configuration window opens.

4. Enter **ClassProject** in the **Project Name** field in the **General Settings** section.

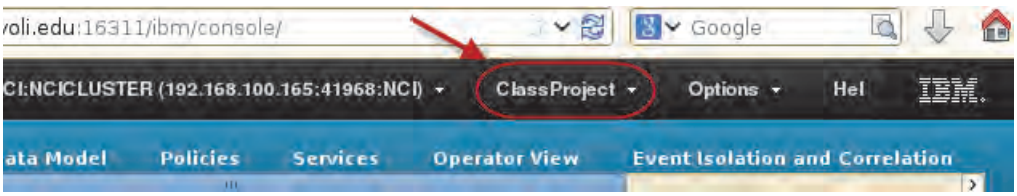
5. In the **Member Selection** section, complete the following steps:
   a. Select **Data Sources** in the **List By** menu.
   b. Scroll down in the **Global Repository** list, click **XmlDsaMediatorDataSource**, click **>>** to add it to the **Project Members** list and click **OK**.



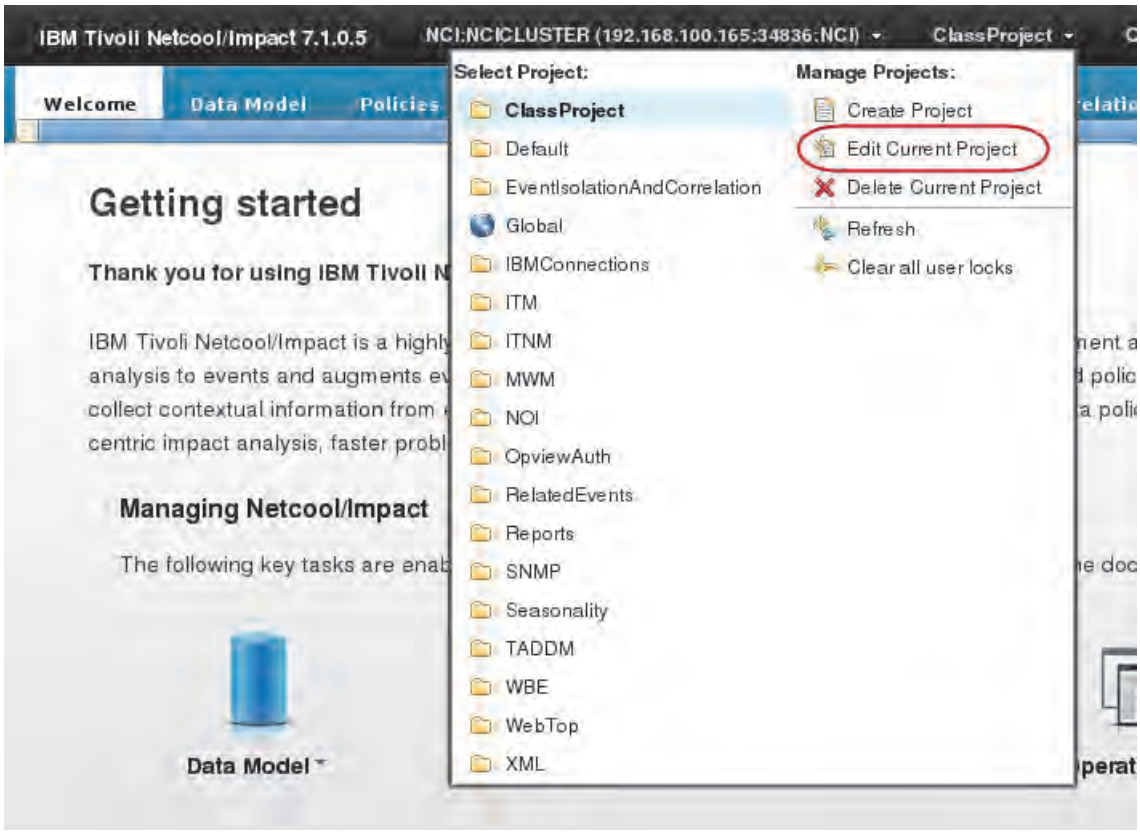
6. Click **OK** in the confirmation window.



ClassProject is shown as the current project in the project menu at the top of the console.

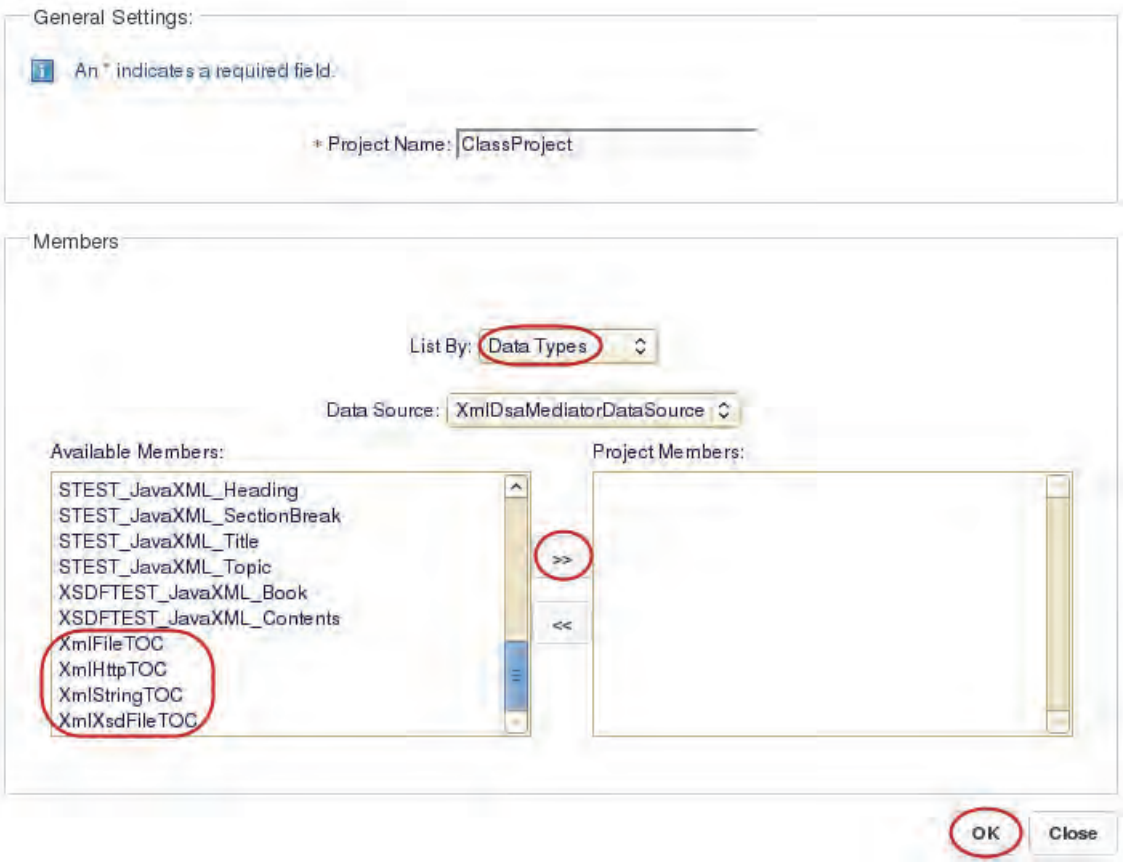


7. Edit the ClassProject configuration. Click the **ClassProject** menu and click **Edit Current Project** in the **Manage Projects** section.

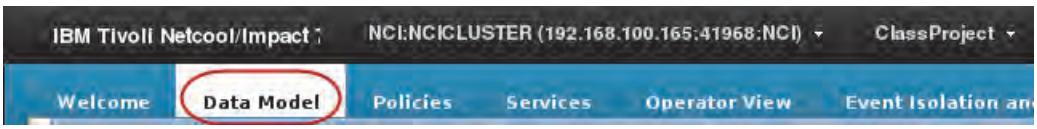


8. Select **Data Types** in the **List By** menu. Because **XmlDsaMediatorDataSource** is the only data source in the project, it is automatically shown in the Data Source menu.

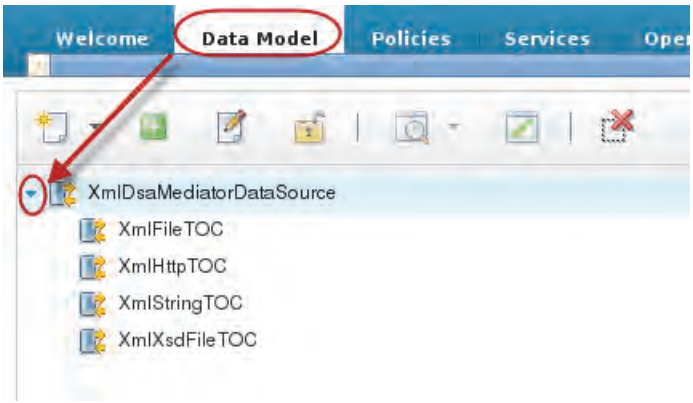
9. Add all of the available data types to the data source. Select all data **xml** data types in the **Available Members** list, click **>>** to add them to **Project Members**, and click **OK**.



10. Click the **Data Model** tab in the toolbar at the top of the console to show the list of data sources in the current project.

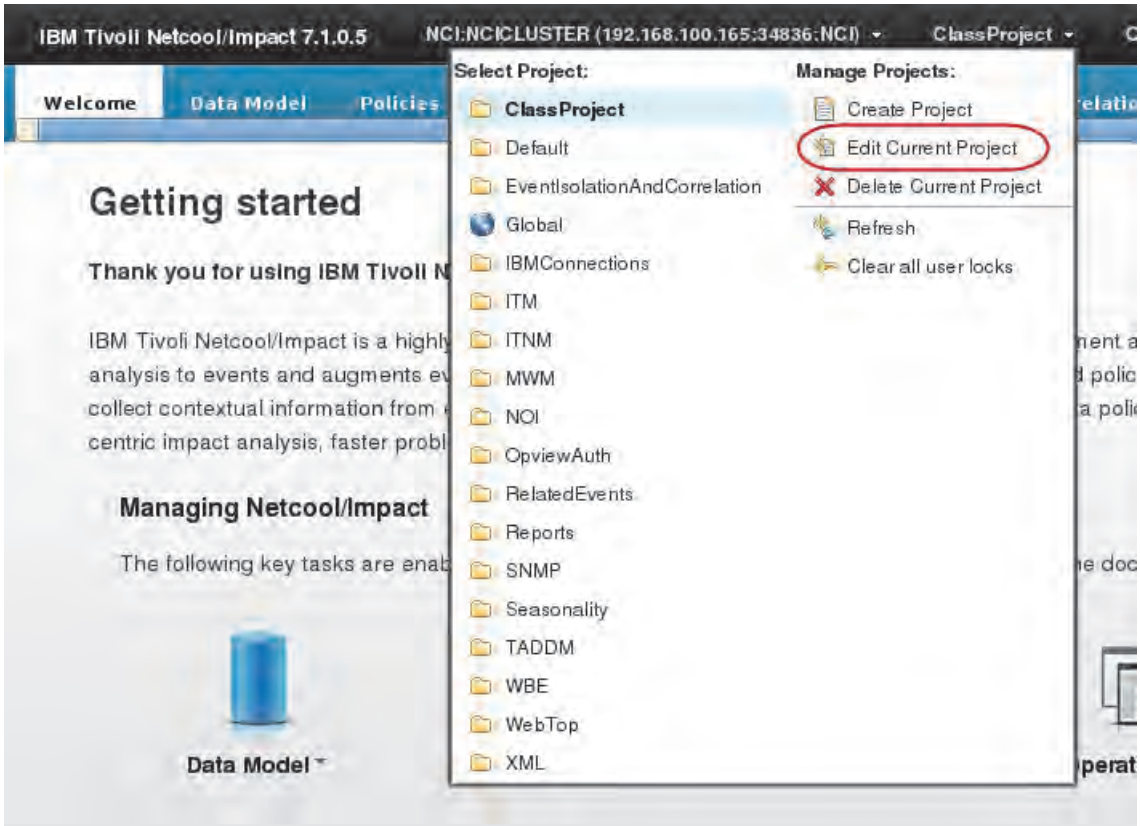


11. Click the arrow to the left of **XmlDsaMediatorDataSource** to show the list of configured data types.

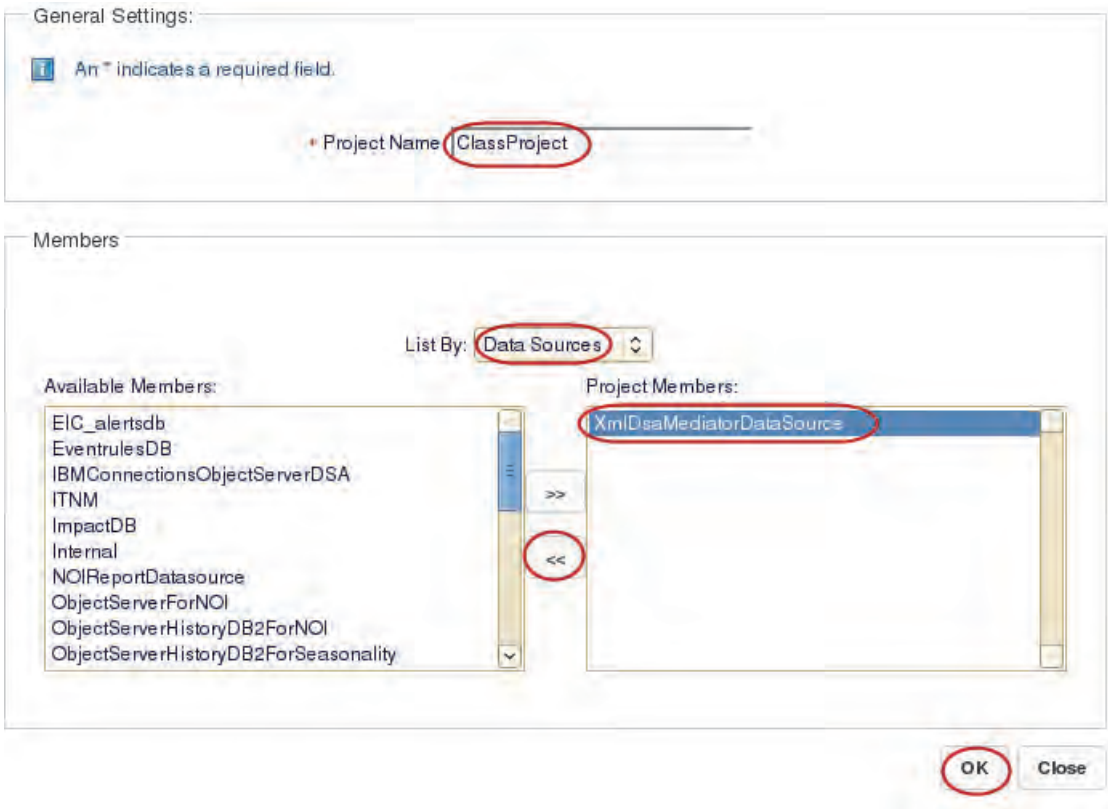


Exercise 3 Removing project members

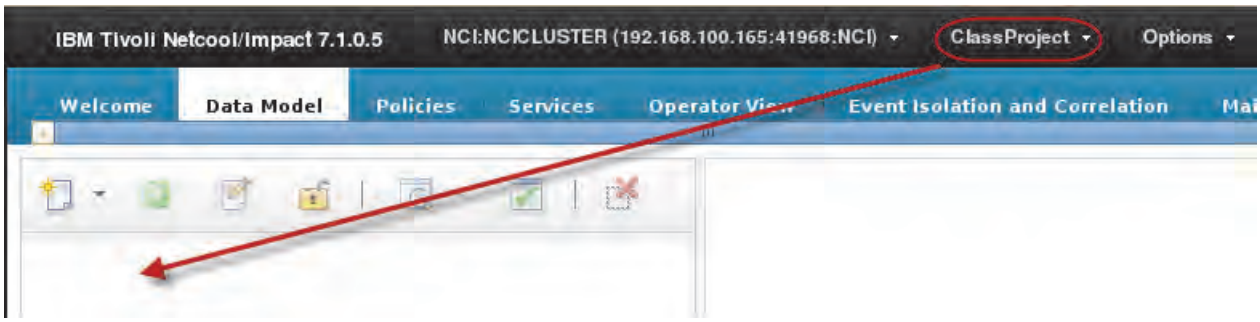
1. Edit the ClassProject project. Click **ClassProject** in the project menu and select **Edit Current Project**.



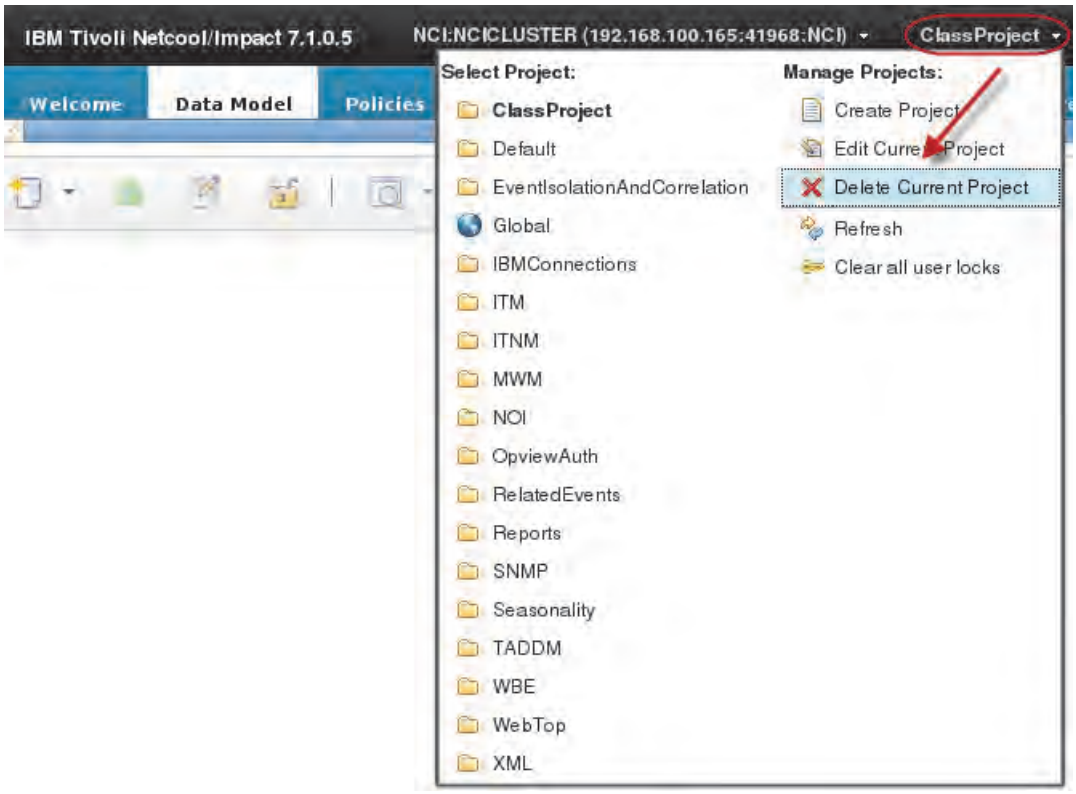
2. Remove the XmlDsaMediatorDataSource data source from the project.
   a. Select **Data Sources** in the **List By** section.
   b. Click **XmlDsaMediatorDataSource** in the **Project Members** list.
   c. Click the double arrows (**<<**) and click **OK**.



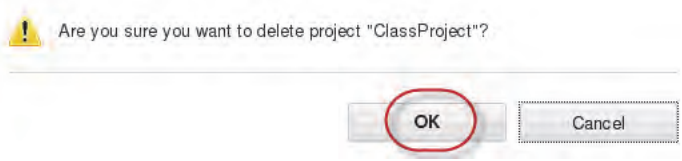
The console indicates that there is no data to show in the **Data Model** tab.



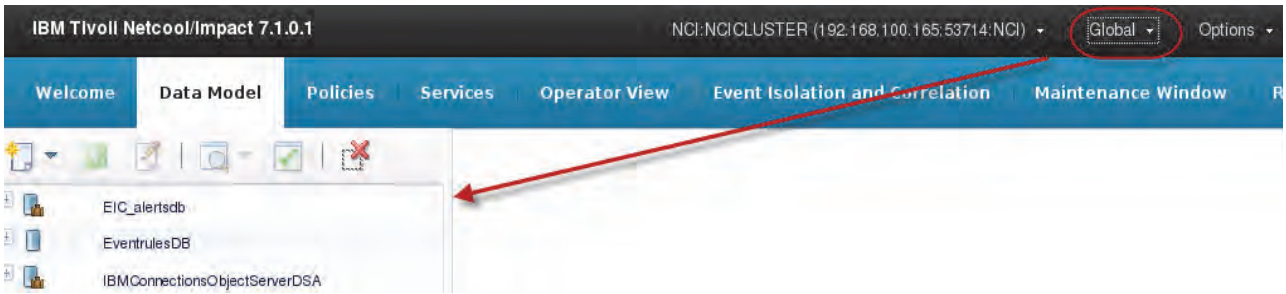
3. Delete the ClassProject project. Click **ClassProject** in the project menu and select **Delete Current Project**.



4. Click **OK** in the confirmation window.



ClassProject is removed from the project menu and the current project reverts to **Global**.



Unit 3 The Netcool/Impact data model exercises

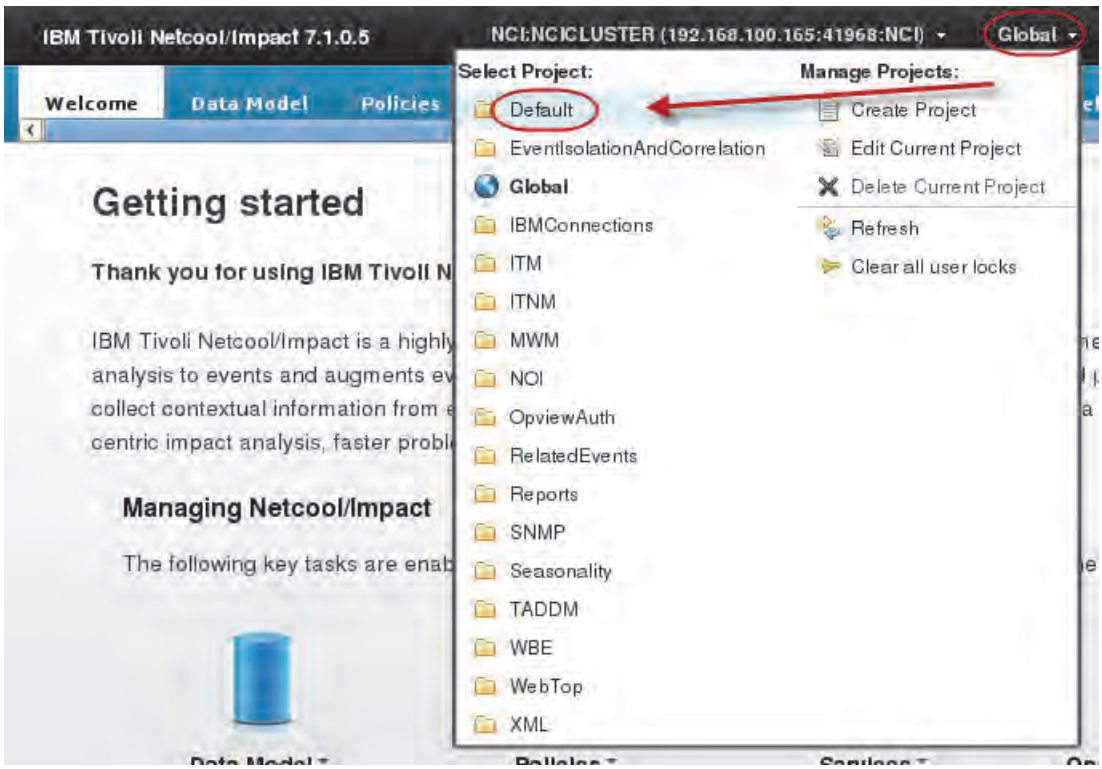
The focus of this unit is the Netcool/Impact Data Model.

Exercise 1 Creating the HS_TRAIN data source

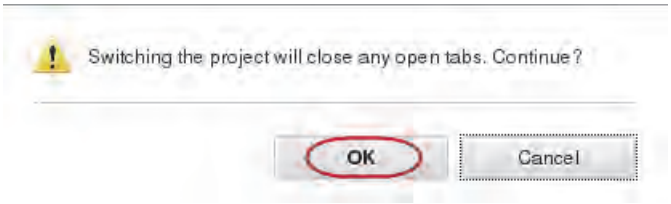
You must create a data source definition for the database that holds the sample data. The sample data was loaded into the Derby database during an earlier exercise.

1. Log on to the Netcool/Impact console as the user **impactadmin** and the password **object00**.
2. Select **Default** in the project menu.

Note: You work in the **Default** project for the remainder of the class.

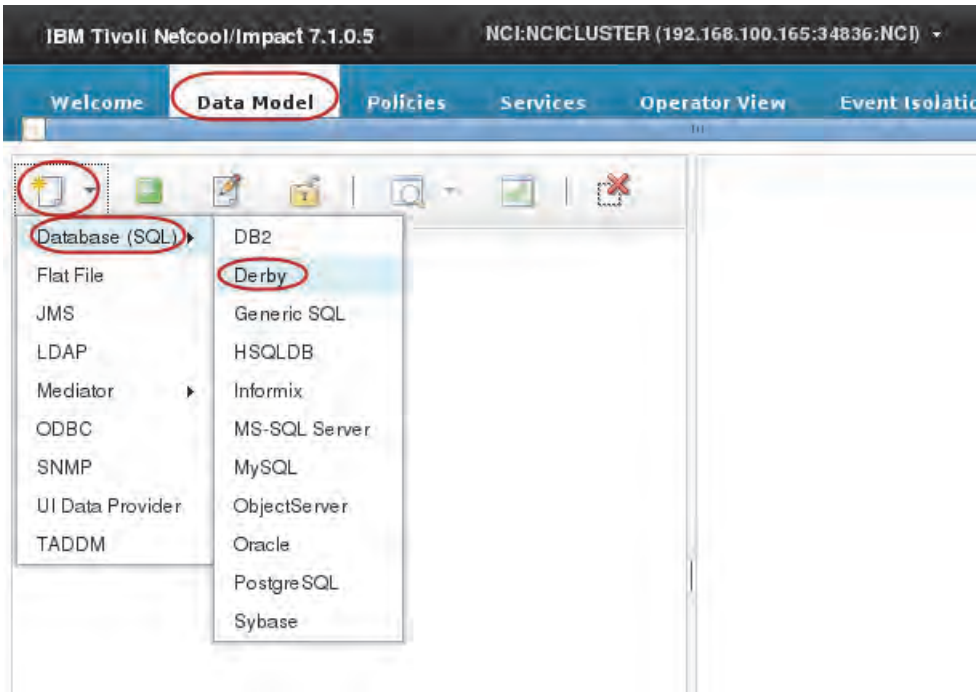


3. Click **OK** in the confirmation window.



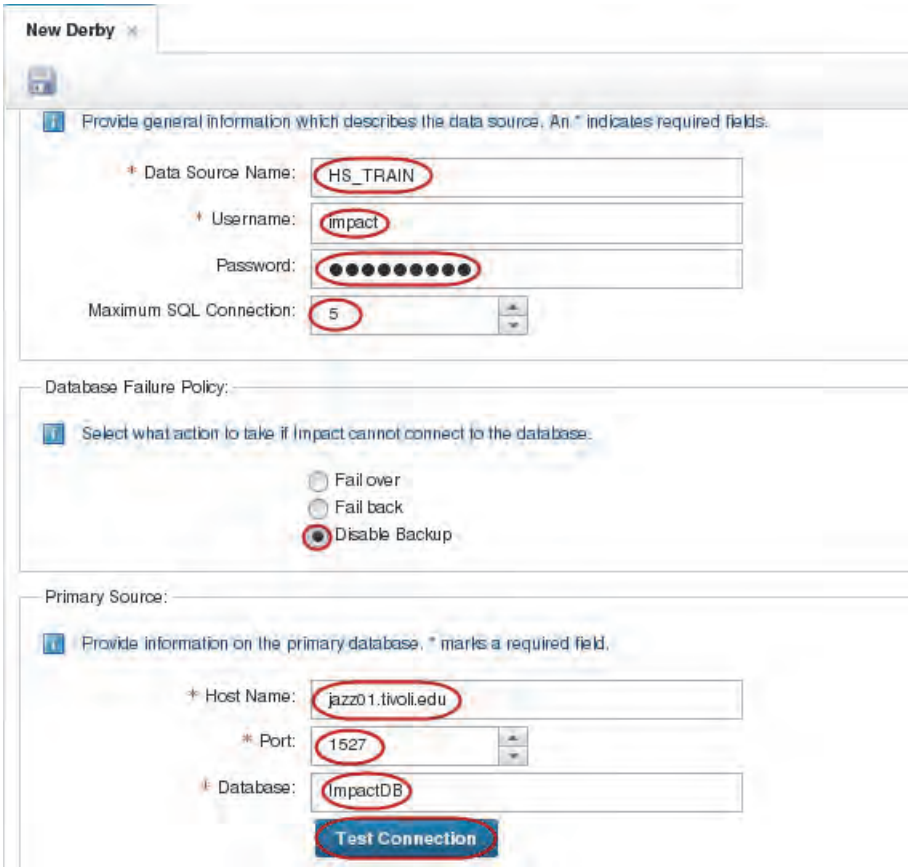
4. Click **Data Model** to open the **Data Model** page.

5. Click the **New Data Source** icon and select **Database (SQL) > Derby**.

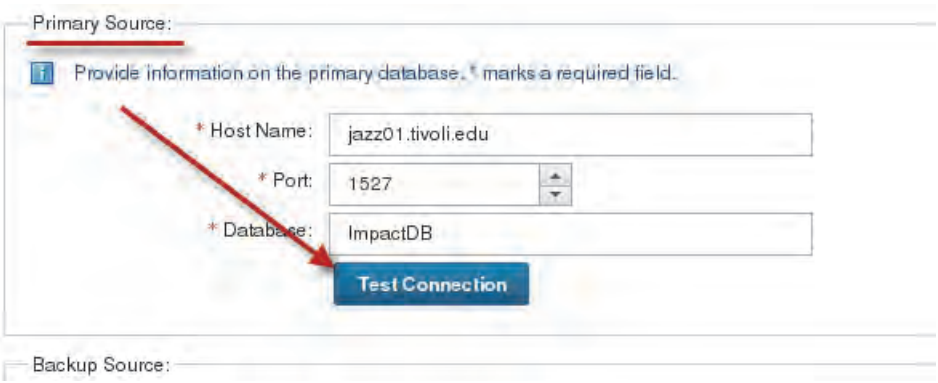


6. On the right side of the page, configure the **General Settings, Database Failure Policy,** and the **Primary Source** panels. Use the following information:

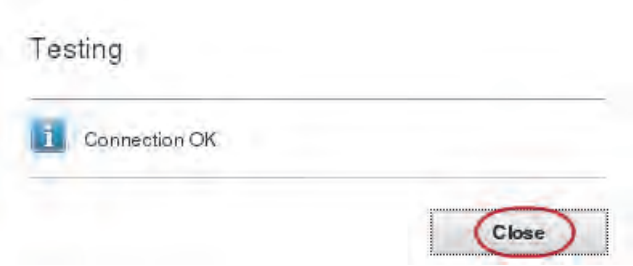
- General Data
  - Data Source Name **HS_TRAIN**
  - Username **impact**
  - Password **derbypass**
  - Maximum SQL Connection **5**
- Database Failure Policy **Disable Backup**
- Primary Source
  - Host Name **jazz01.tivoli.edu**
  - Port **1527**
  - Database **ImpactDB**



7. In the **Primary Source** section, click **Test Connection**.



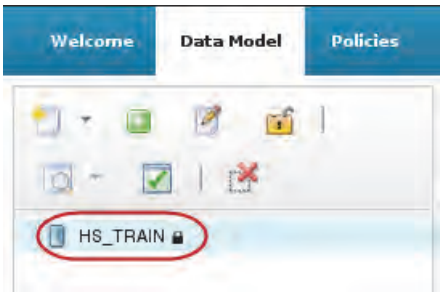
8. When the Information window opens, click **OK**.



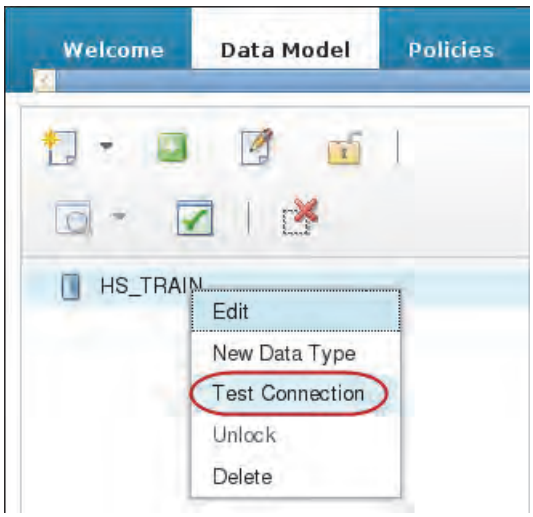
9. Click the **Save** icon to save the data source configuration.

10. Click **Close** if the window does not clear.

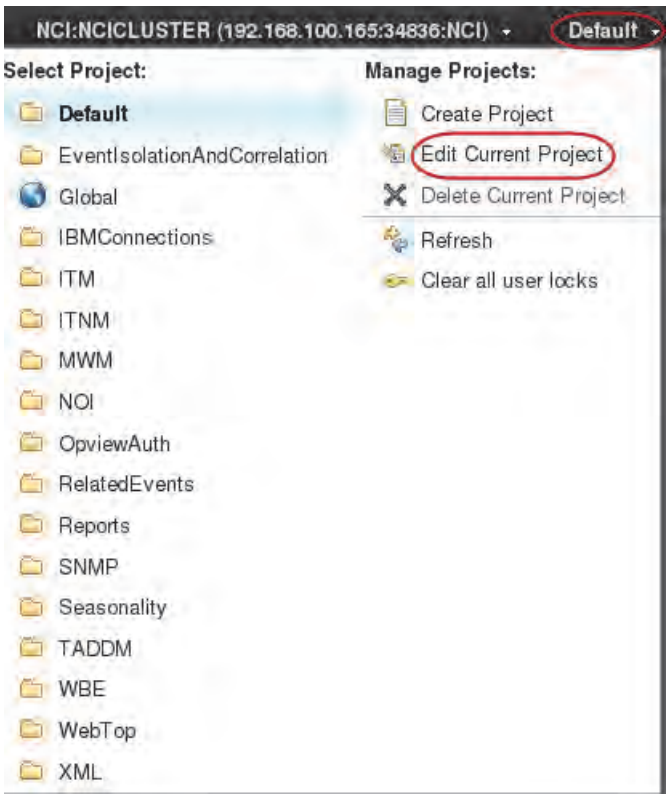
You can edit the data source definition after you save it. While the data source is opened in edit mode, the GUI shows a lock symbol to the right of the **HS_TRAIN** data source.



You can also test the connection by right-clicking the data source name and selecting **Test Connection**.

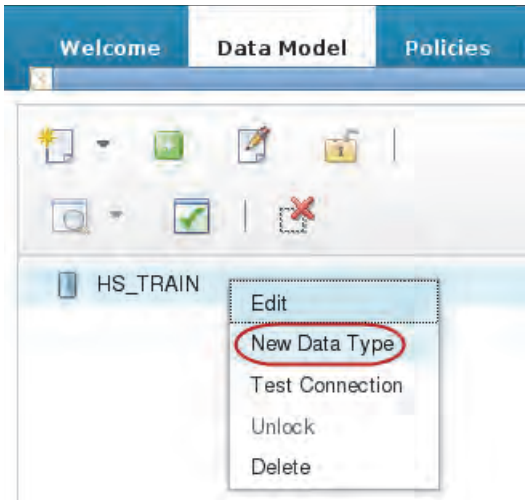


Note: If the **HS_TRAIN** data source is not shown in the **Default** project, check the Global Repository by clicking the **Edit Project** icon. Add the **HS_TRAIN** data source to the **Default** project.



Exercise 2 Creating data types for the HS_TRAIN data source

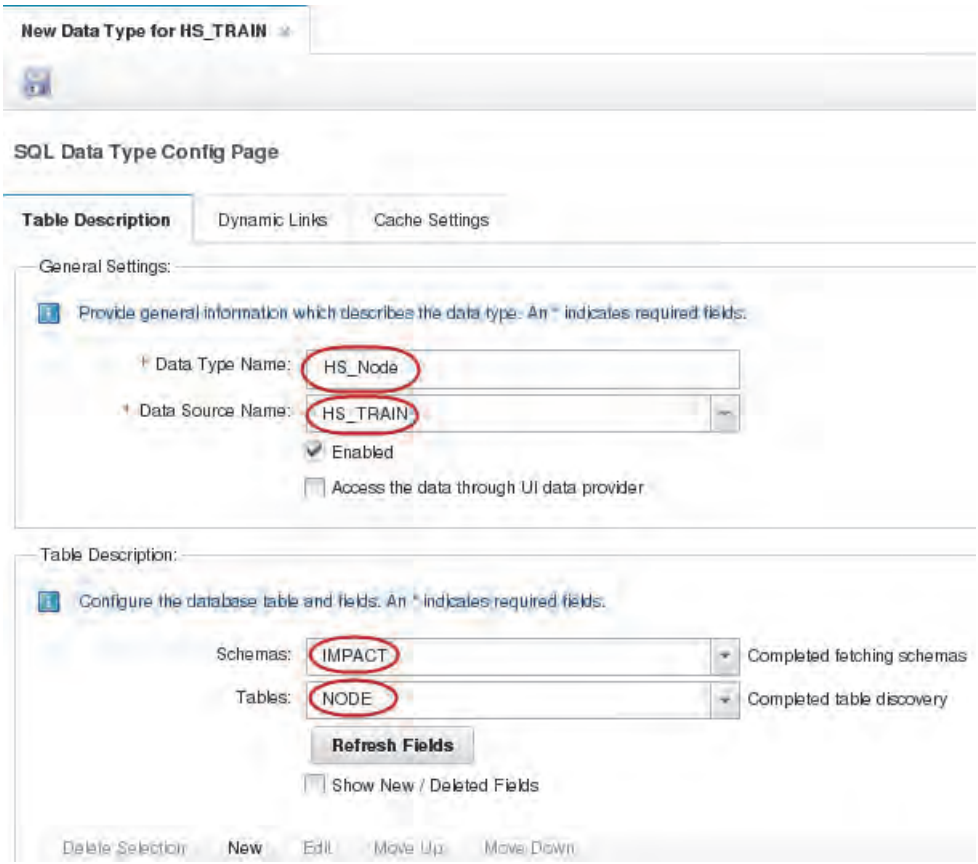
1. Right-click the **HS_TRAIN** data source name, and select **New Data Type**.



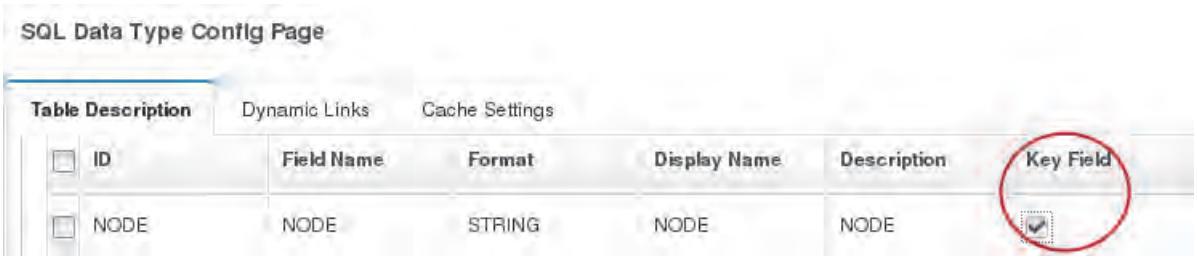
   On the right side of the browser window, the data type configuration panel opens.

2. Enter **HS_Node** for the **Data Type Name**.
3. Select **IMPACT** from the **Base Table** list.
4. To the right of the **Base Table** list, select **NODE** from the list on the right.

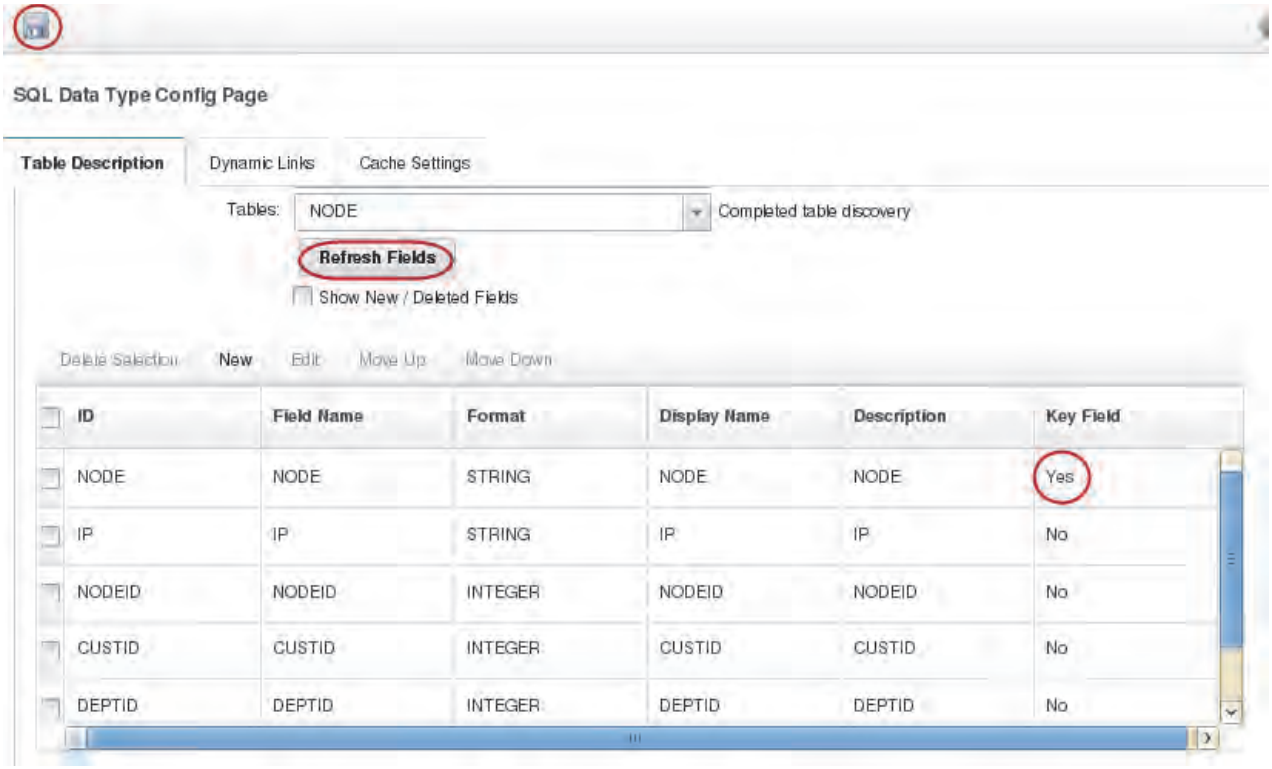
Important: If an error occurs in this section clear the browser cache and try again.



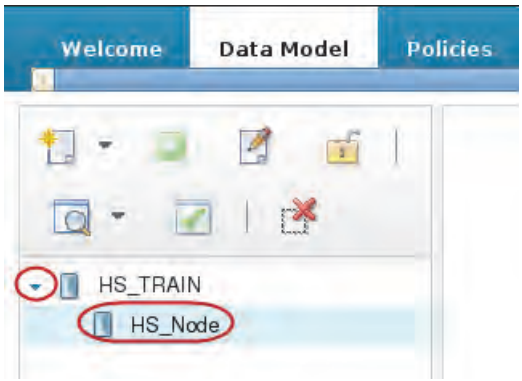
5. Click **Refresh Fields** to display the table structure.
6. Select **NODE** as the **Key Field**. Double click **No** to display the checkbox. Click again to show the check-mark.



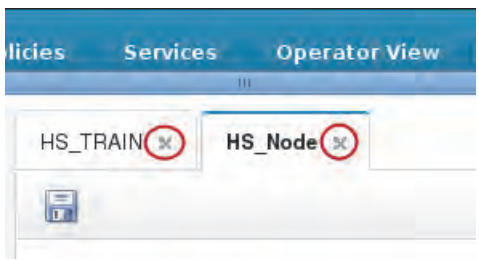
7. Click away from the row to show **Yes**. Click the **Save** icon.



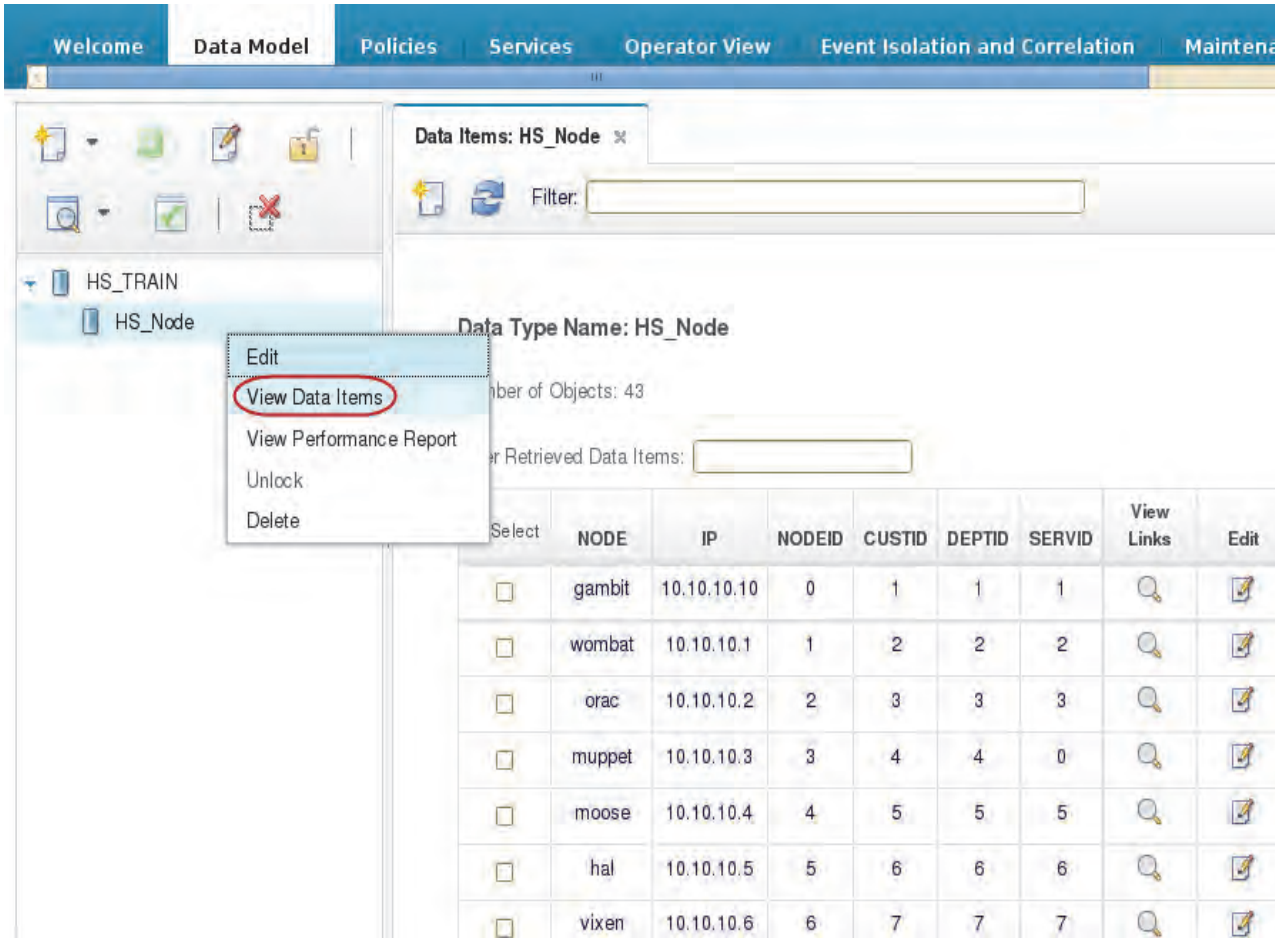
8. Verify that **HS_Node** is listed under the data source **HS_TRAIN** in the data model.



9. Close the open page tabs in the right side of the console. Click the close symbol (**X**) in the **HS_TRAIN** and **HS_Node** tabs in the right side of the page.



10. Verify that the data type definition is functional by right-clicking **HS_Node** and selecting **View Data Items**. Table values display in the right pane for the **HS_Node** data type.

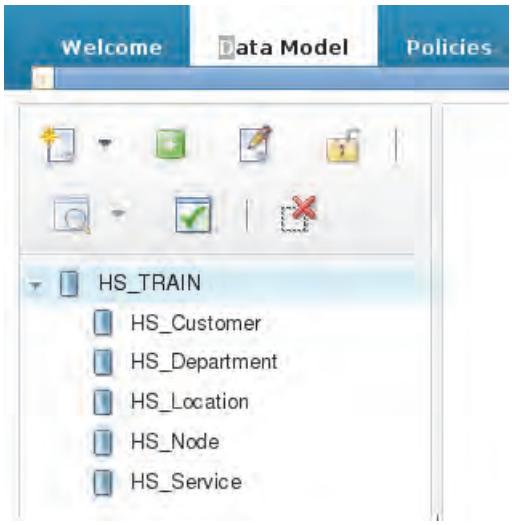


11. Create four additional data types for the **HS_TRAIN** data source by repeating the previous steps for each new data type.

    Use the data type names and associated keys from the following table.

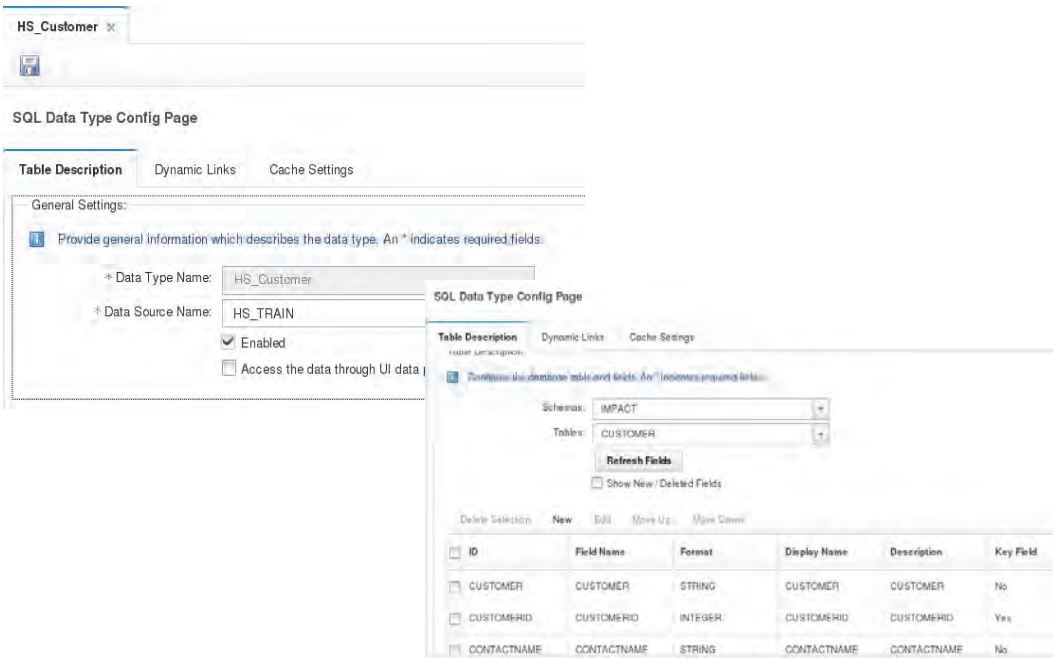
Data Type Name	Base Table Description	Key
HS_Customer	IMPACT/CUSTOMER	CUSTOMERID
HS_Department	IMPACT/DEPARTMENT	DEPTID
HS_Location	IMPACT/LOCATION	LOCATIONID
HS_Service	IMPACT/SERVICE	SERVICEID

After completion of the data types, the **Data Sources and Types** panel looks like the following screen image.

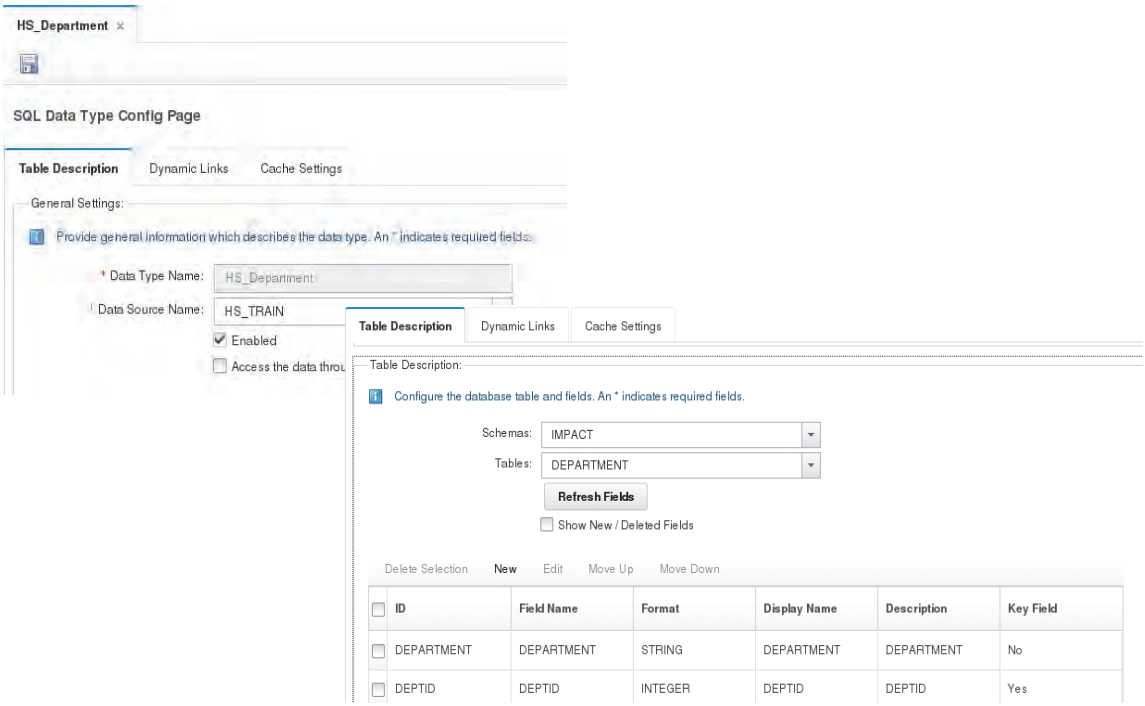


The **HS_Customer**, **HS_Department**, **HS_Location**, and **HS_Service** tables definitions in the **HS_TRAIN** data source are shown in the following screen captures.

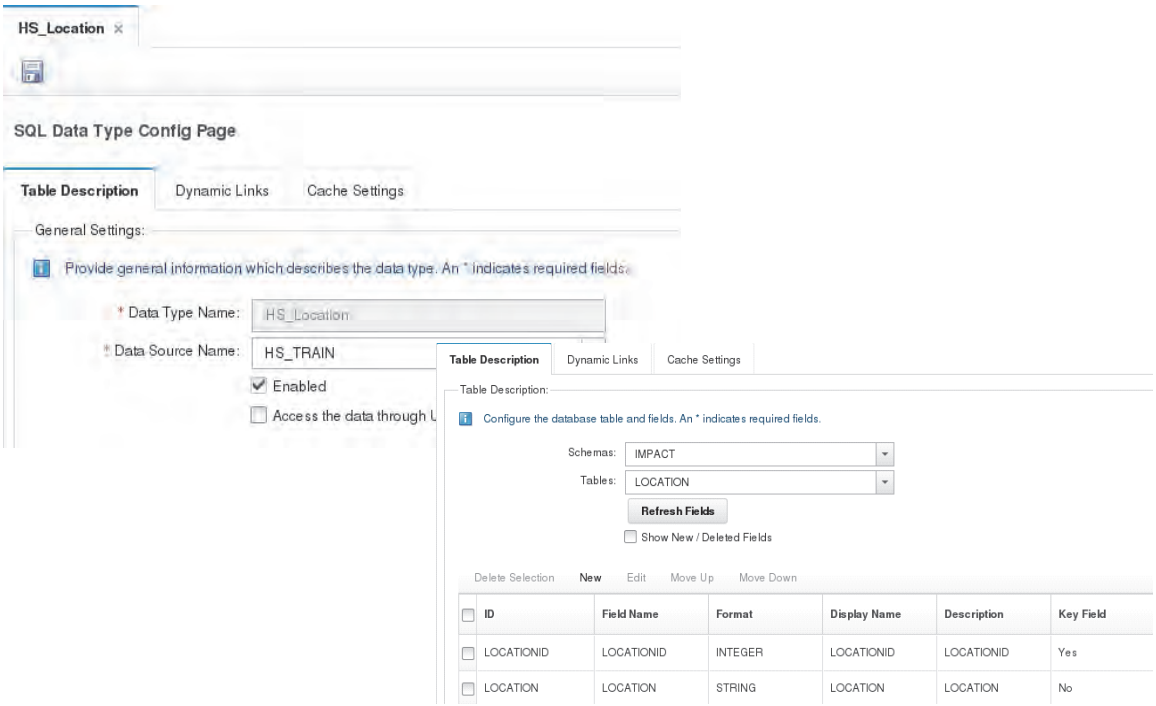
The data type definition for the **HS_Customer** table looks like this example:



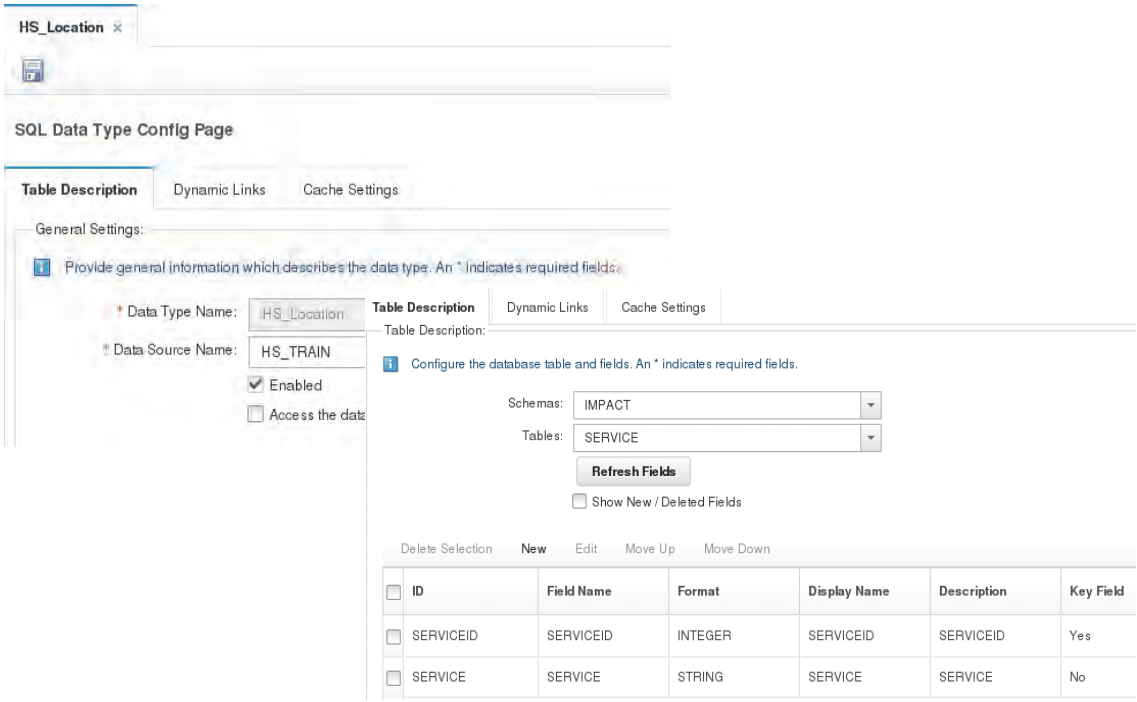
The data type definition For the **HS_Department** table looks like this example:



The data type definition for the **HS_Location** table looks like this example:



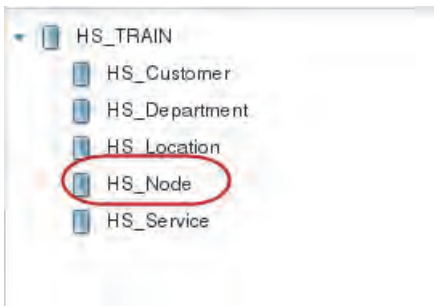
The data type definition for the **HS_Service** table looks like this example:



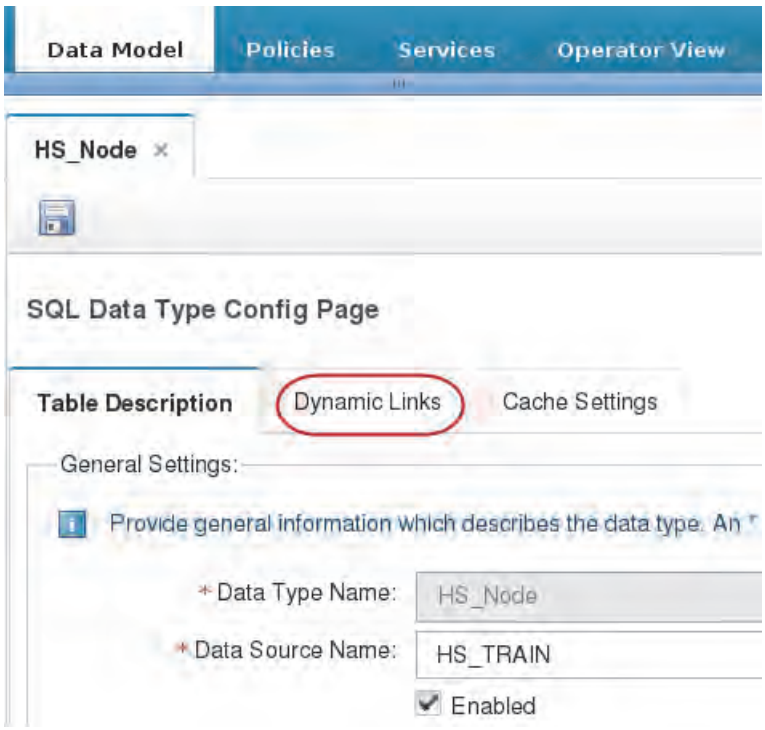
Exercise 3 Creating dynamic links

In this exercise, you create four dynamic links. The first link connects the source table **HS_Node** to the target table **HS_Customer,** using **CUSTID** as the foreign key.

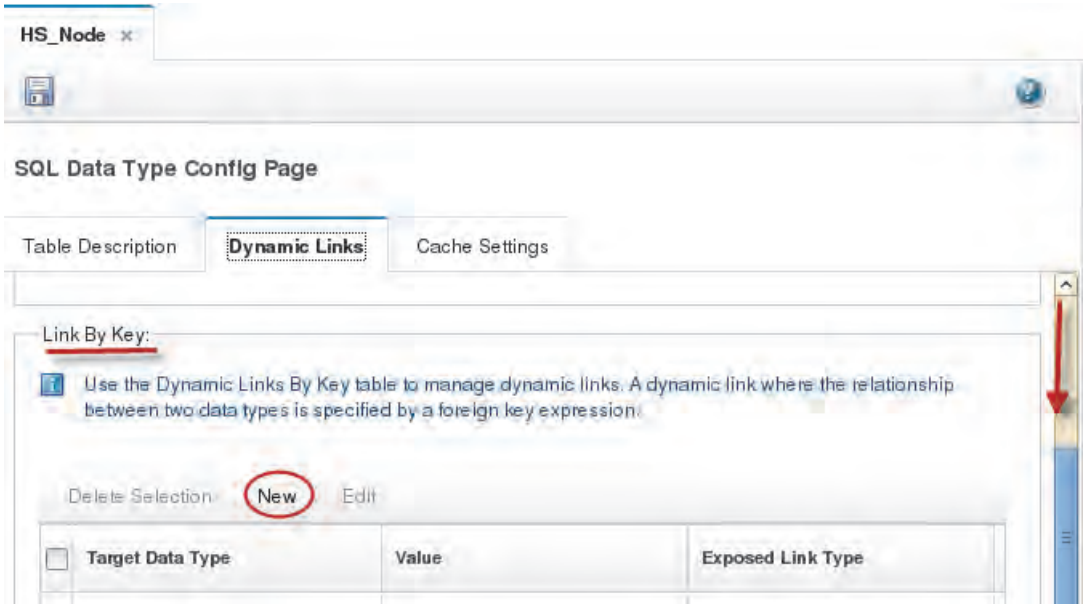
1. Double-click **HS_Node** to edit the data item.



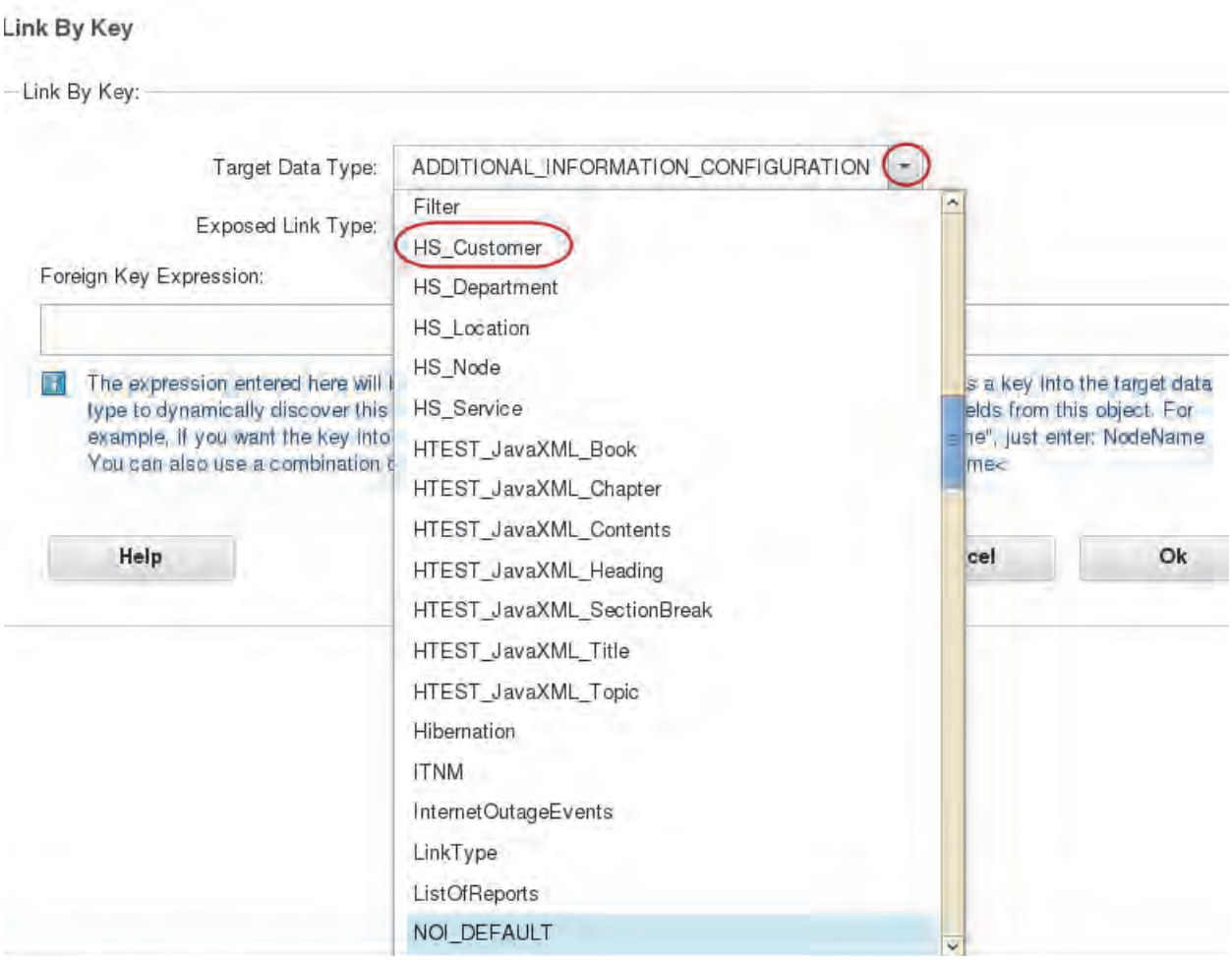
2. Click the **Dynamic Links** tab.



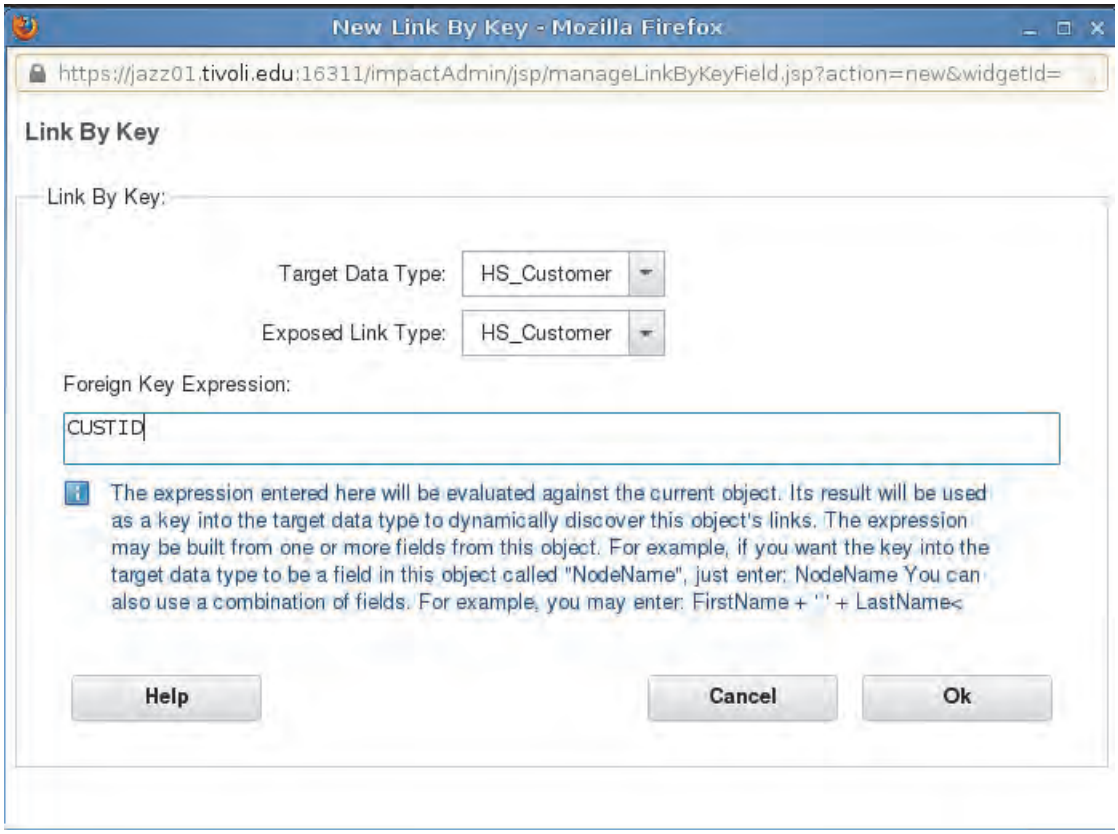
3. Scroll to the **Link By Key** section**.** Click **New** to create the first entry in the table.



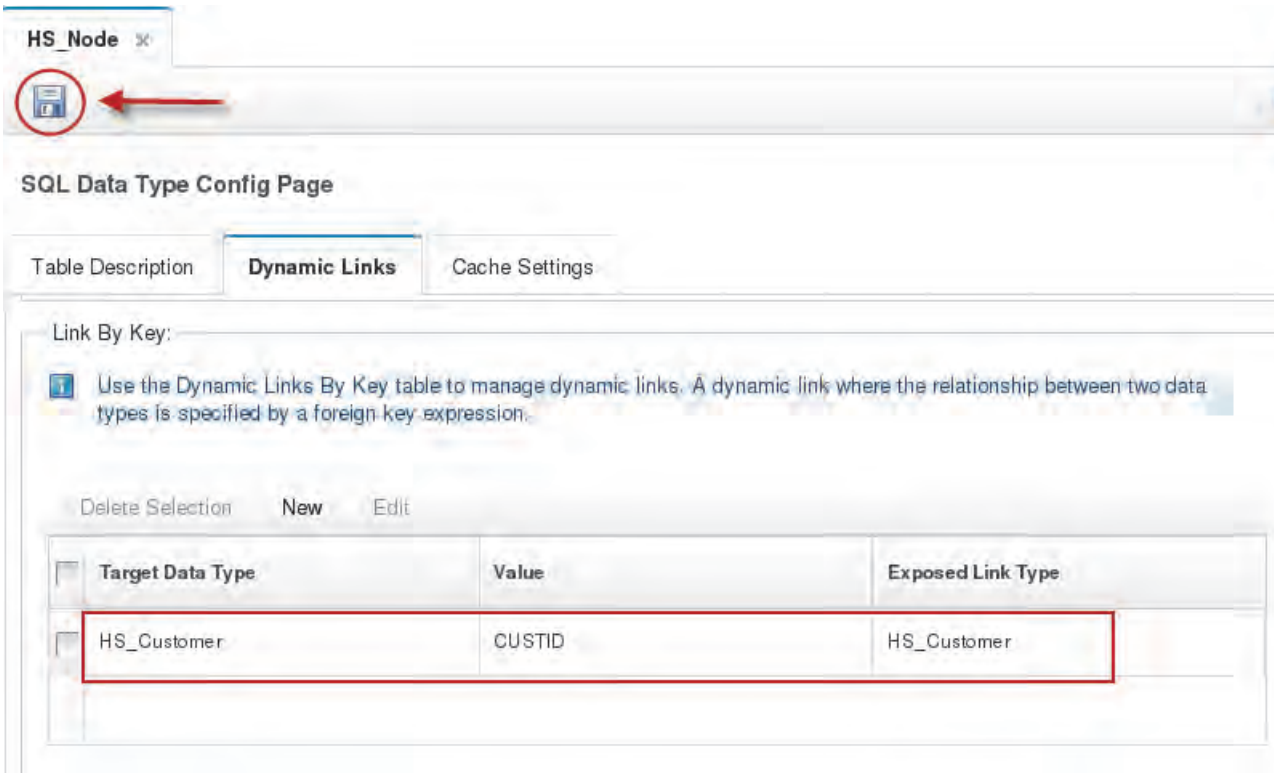
4. Select **HS_Customer** in the **Target Data Type** menu.



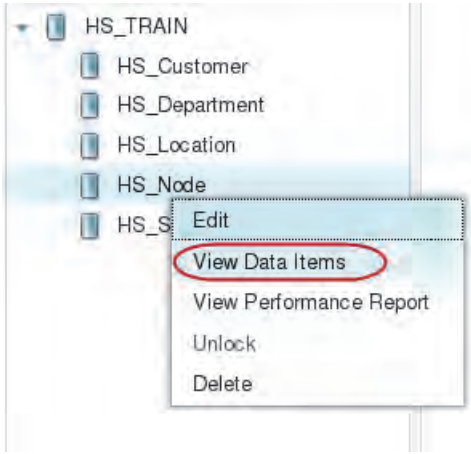
5. Type **CUSTID** in the **Foreign key expression** field. Click **OK** to save the link description.



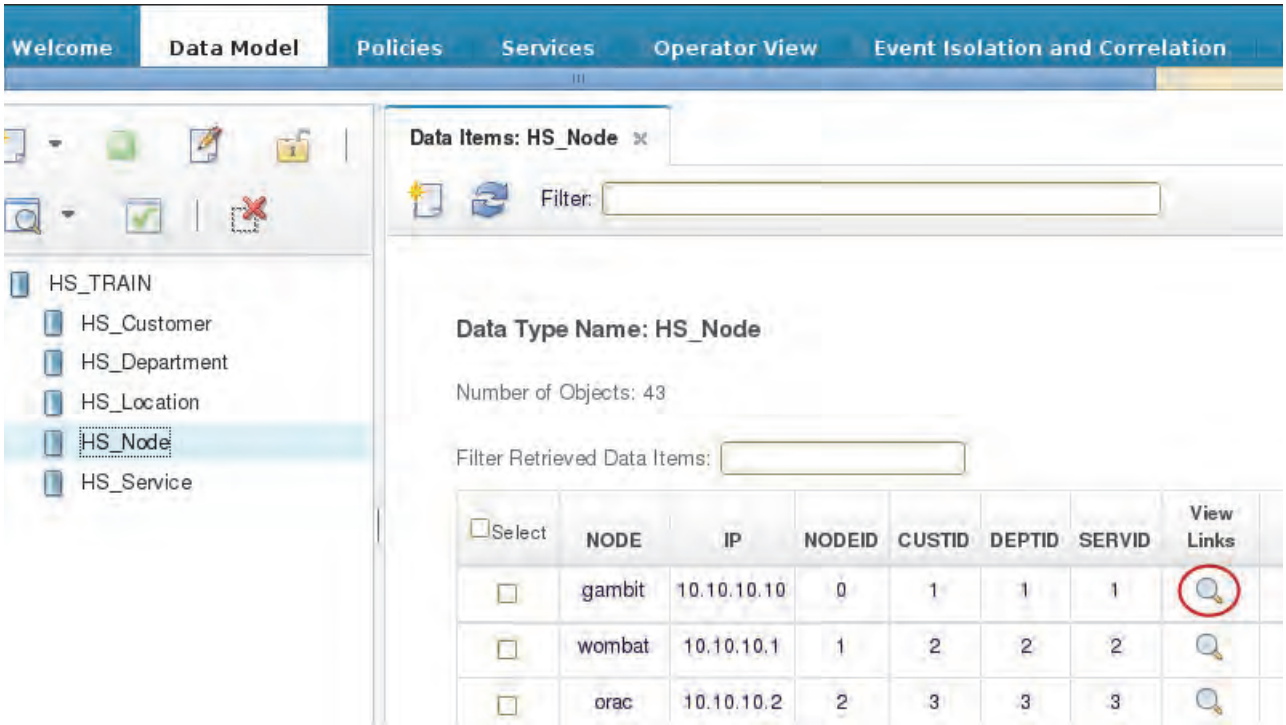
6. Click the **Save** icon.



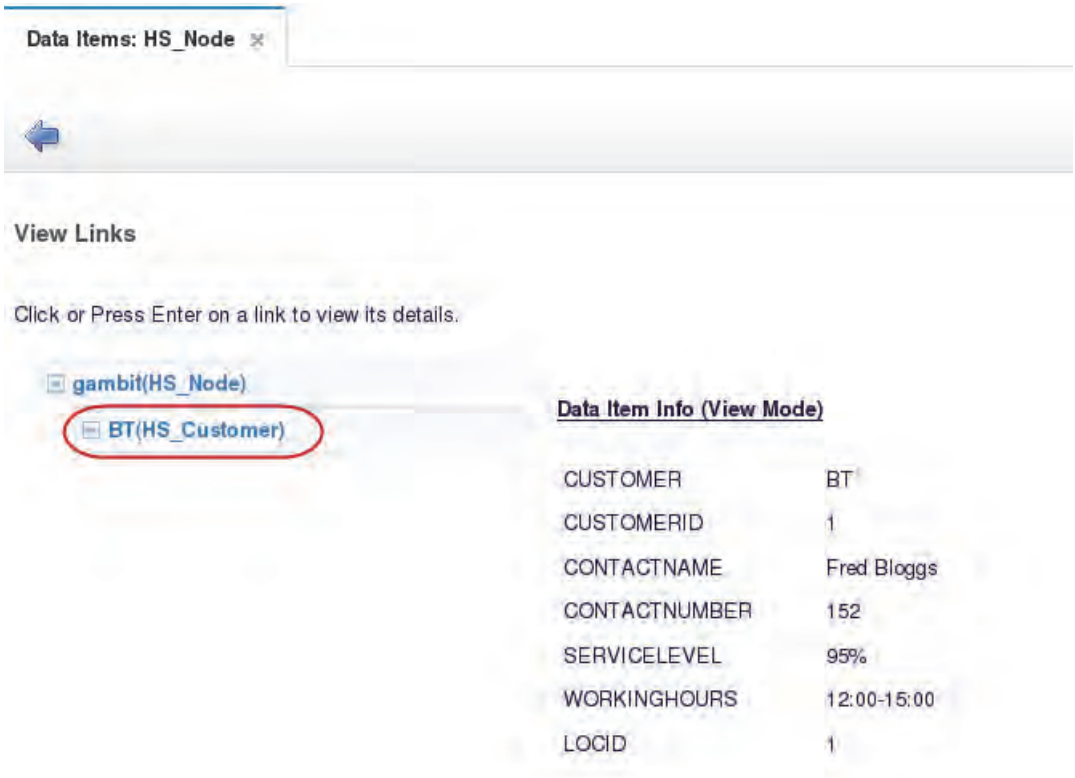
7. Verify that the link works by right-clicking the HS_Node data type name and selecting **View Data Items**.



8. Click the magnifying glass icon next to the **gambit** entry to view the linked data item information.



The resulting screen should look like the following example.

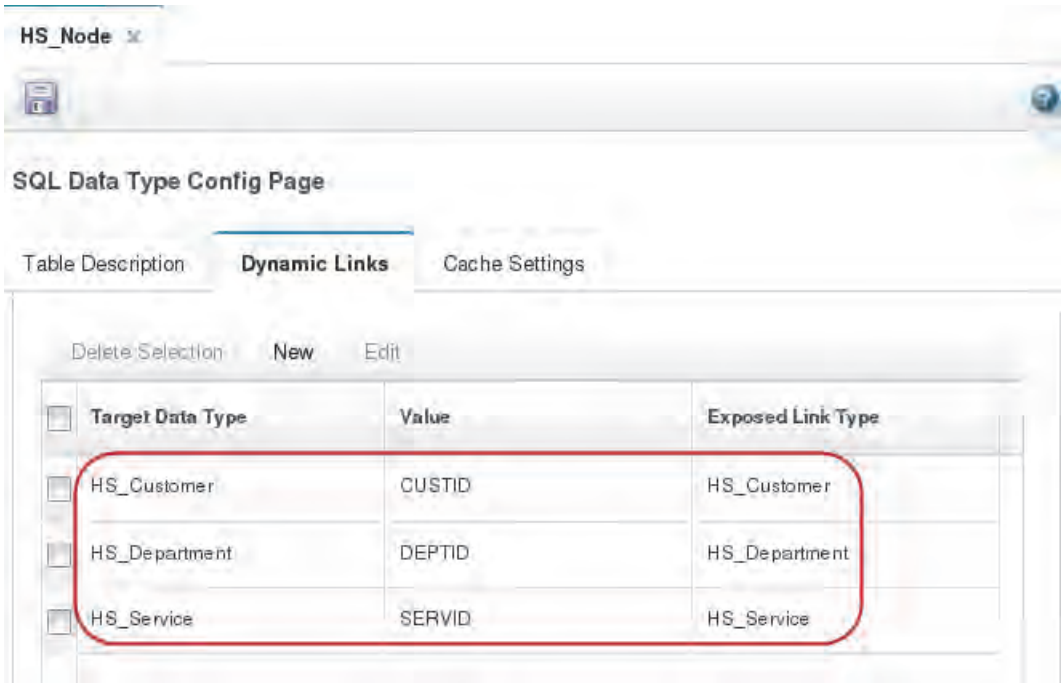


9. Repeat the steps in this exercise for the remaining link table entries.

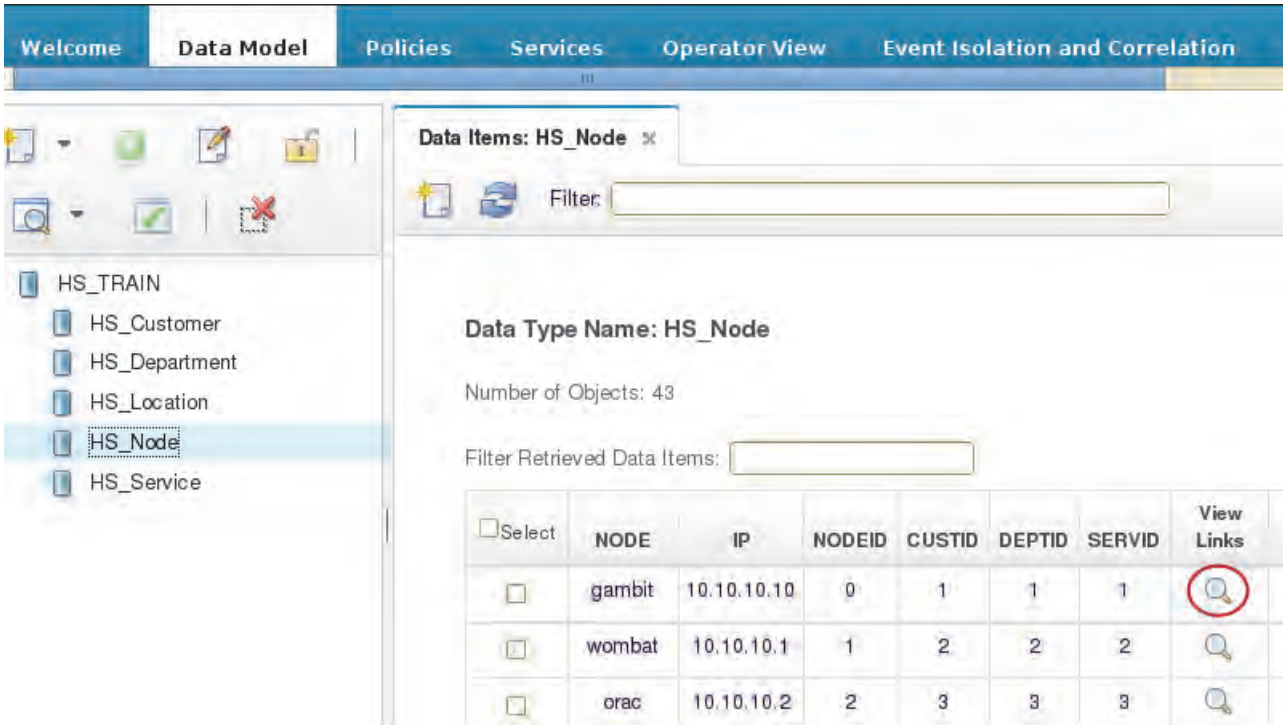
Table 1 Link information

Link Source Table	Link Target Table	Foreign Key
HS_Node	HS_Department	DEPTID
HS_Node	HS_Service	SERVID
HS_Customer	HS_Location	LOCID

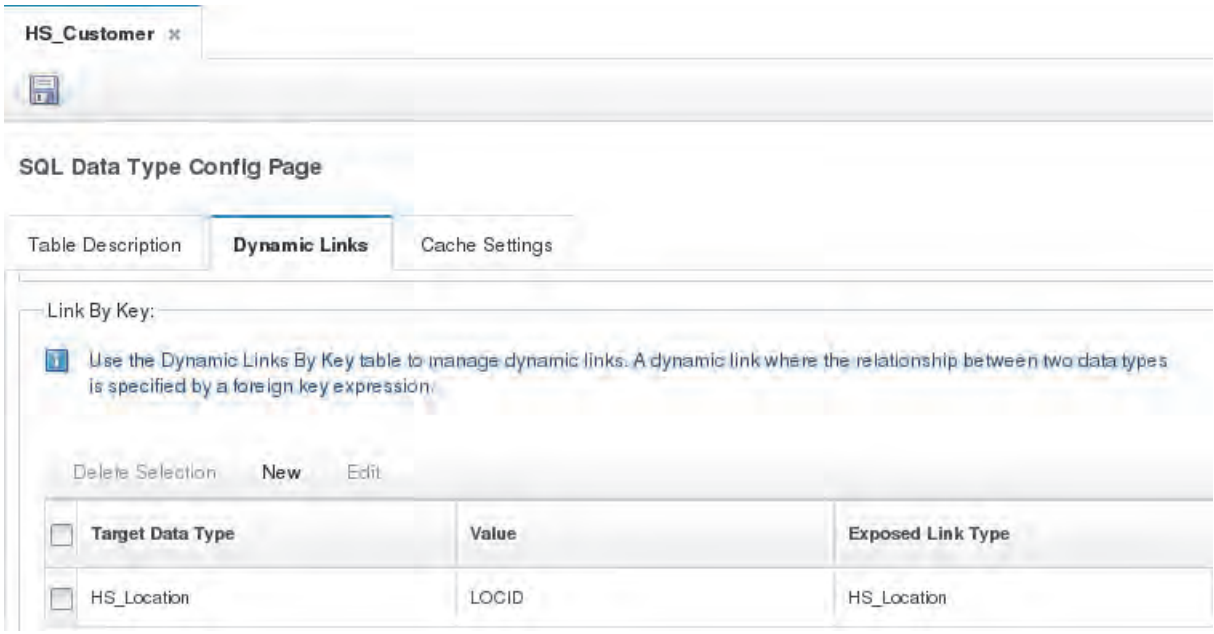
The following graphic shows the resulting link definitions for **HS_Node**.



10. Click the link icon to show the **HS_Node** link data for the **gambit** entry.



The following graphic shows the resulting link for **HS_Customer** after the dynamic link to **HS_Location** has been created.



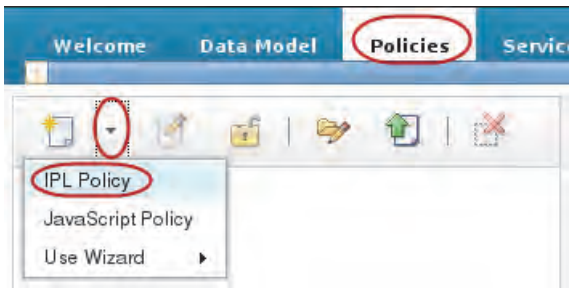
11. Display data for the **HS_Node** data item. Click the link icon to show the **HS_Customer** link data for the **gambit** entry. Expand **HS_Customer** to see the link to the **HS_Location** table.

Unit 4 Policies exercises

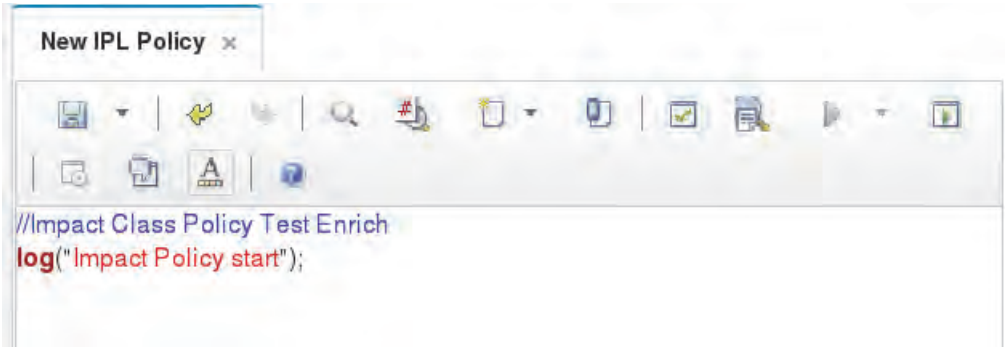
This unit's exercises are an introduction to writing Netcool/Impact policies.

Exercise 1 Creating a policy

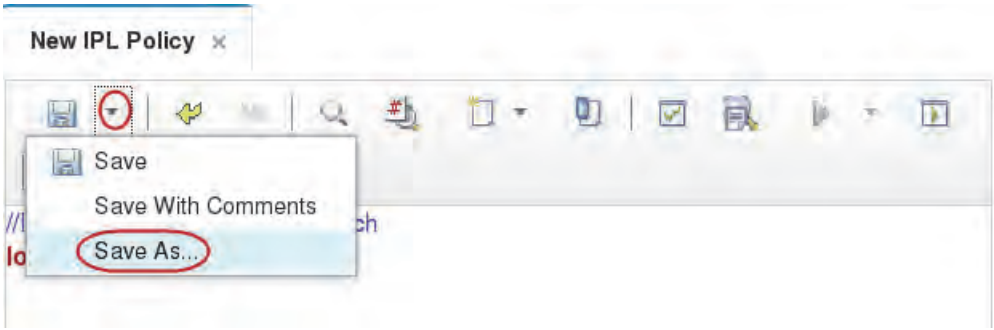
1. Click **Policies** to open the **Policies** tab. Click the **New Policy** icon. Select **IPL Policy** in the list.



2. Modify the policy as shown in the screen capture.
   log ("Impact Policy start");

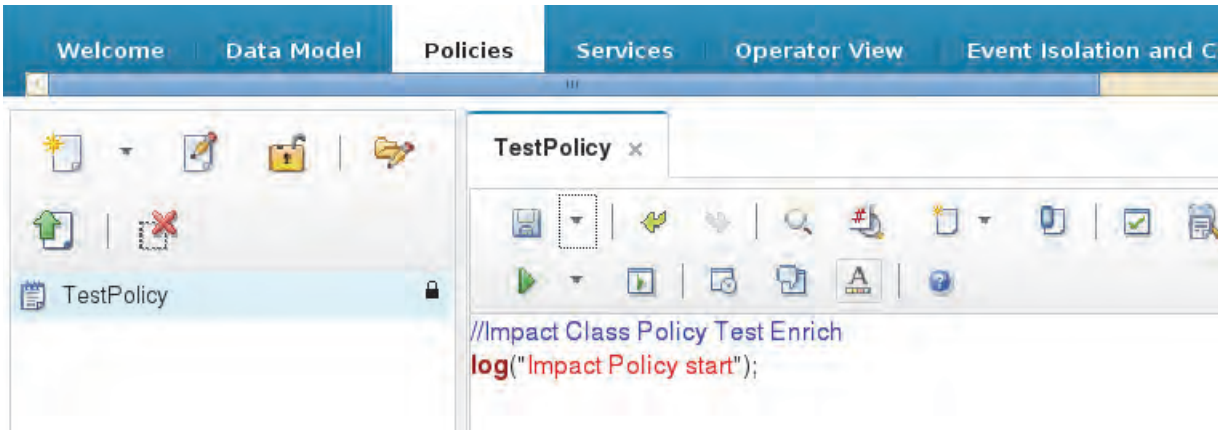


3. Click the arrow next to the **Save** icon and select **Save As** in the list.



4. Type the **TestPolicy** in the **Policy Name** field. Click **Save As**.

The new policy is displayed in the Policies panel.

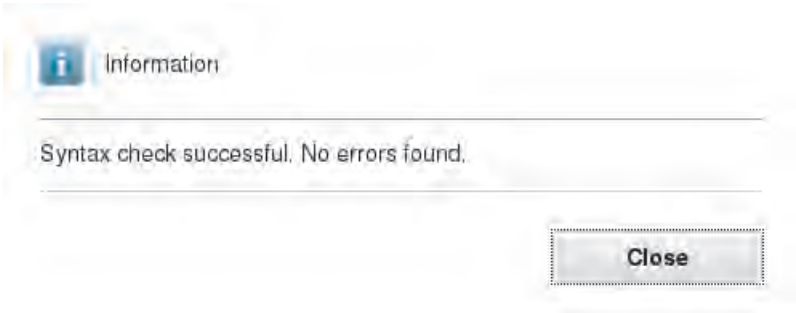


Exercise 2 Creating a context policy

1. Edit the **TestPolicy** just created.
2. Type the following policy into the policy editor after the log statement:

```
PrblmTktAssign = NewObject();
PrblmTktAssign.Name = "John Smith";
PrblmTktAssign.Phone = "18001234567";
PrblmTktAssign.Severity = "2";
PrblmTktAssign.Status = "open";
Log(PrblmTktAssign.Name + "/" + PrblmTktAssign.Phone + "/" +
PrblmTktAssign.Severity + "/" + PrblmTktAssign.Status);
```

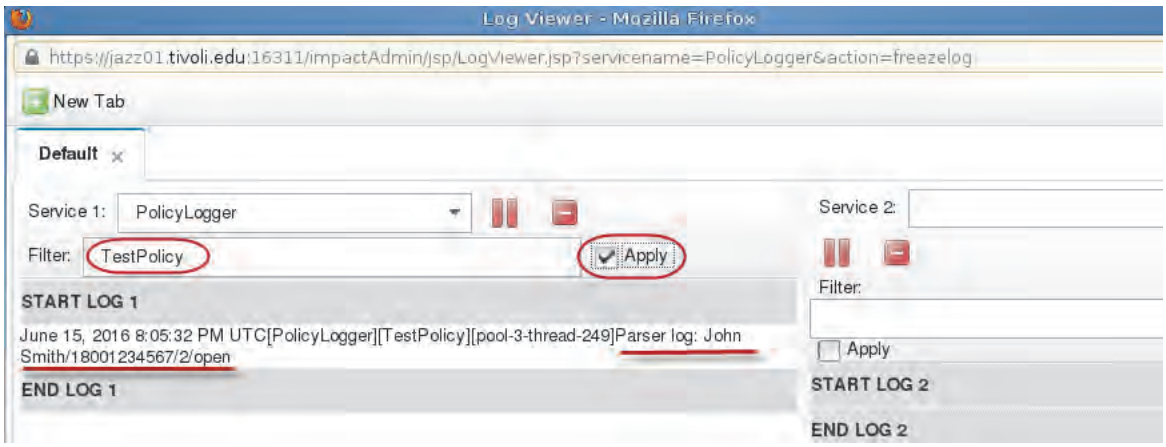
3. Click the **Check Syntax** icon to check for policy errors.
4. Click **OK** to close the window.



5. Click the **Save** icon .
6. Click the right-pointing arrow to test the policy. Click **OK** to close the window.

7. Click the **View Policy Log** icon to view the policy results.

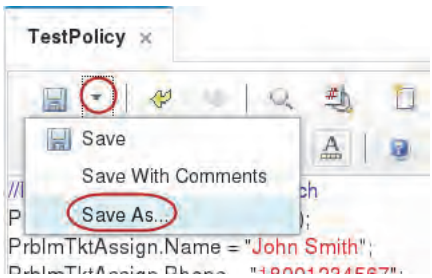
8. Create a Log filter.
   - Type **TestPolicy** in the **Filter** field.
   - Check **Apply** to isolate the results.
   - Click the minus sign “-” in the red box to clear the screen.
   - Leave the **Log Viewer** window open.
   - Execute TestPolicy one more time from the **Policy** panel.
   - The resulting **Log Viewer** output should look similar to the following example.



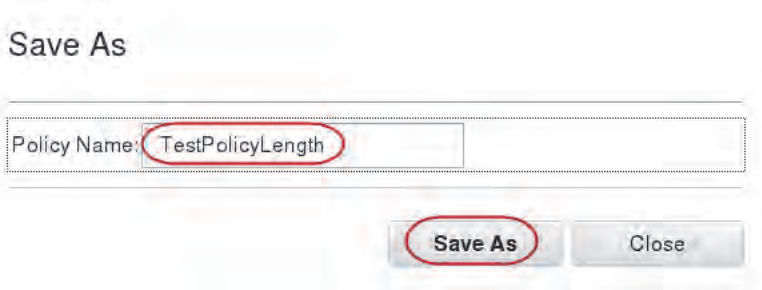
Important: You have to set the filter before you run the policy.

Exercise 3 Using the length function

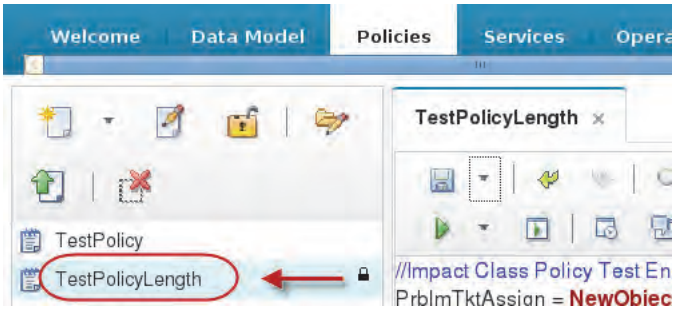
1. Open **TestPolicy** in the policy editor.
2. Click the arrow next to the **Save** icon and select **Save As** from the list.



3. Type **TestPolicyLength** in the **Policy Name** field and click **OK**.



4. Verify that the new policy is displayed in Policies list as **TestPolicyLength**.

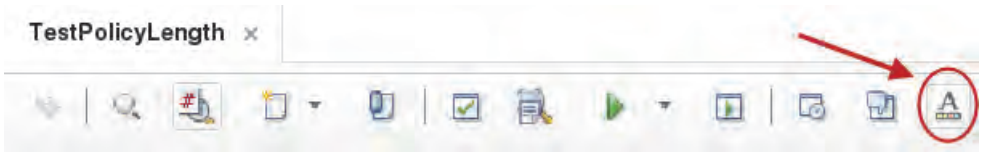


5. Modify the **TestPolicyLength** policy. Delete the existing lines of code, and add the following lines, also shown in the screen capture. Click the **Save** icon.

```
//Length Test
log("Impact Policy Start");
DataType="HS_Location";
Filter=("City='New York'");
CountOnly=False;
MyNodes=GetByFilter(DataType,Filter,CountOnly);
NumNodes=Length(MyNodes);
Log("Length Test Policy" + "," + NumNodes);
```

Note: The quotes display correctly when you type in the policy editor. Cutting and pasting from a pdf document may not provide the desired results.

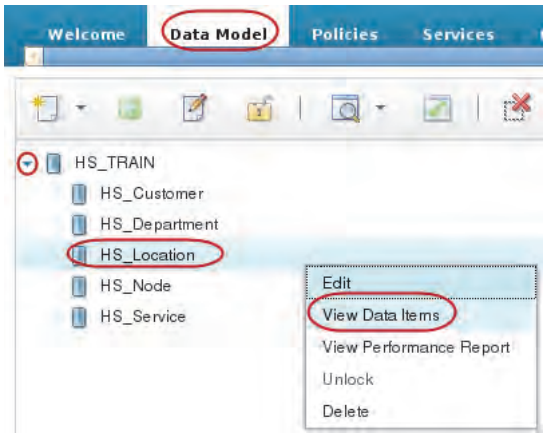
6. Click the icon to turn on syntax highlighting in the policy.



In the policy editor blue, green and red colors make it easier to follow the policy flow.

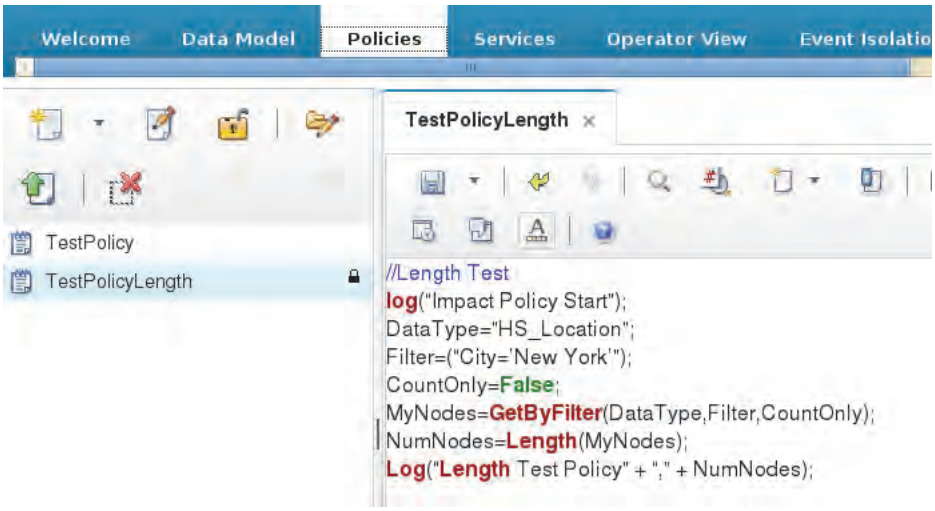
```
//Length Test
log("Impact Policy Start");
DataType="HS_Location";
Filter=("City='New York'");
CountOnly=False;
MyNodes=GetByFilter(DataType,Filter,CountOnly);
NumNodes=Length(MyNodes);
Log("Length Test Policy" + "," + NumNodes);
```

7. Verify the filter used in the **GetByFilter** expression. Click **Data Model** to open the **Data Model** tab.Expand **HS_TRAIN**. Right-click **HS_Location** and select **View Data Items**.



8. Enter the **Filter** expression as shown in the screen capture.

The resulting policy window looks similar to the screen capture.

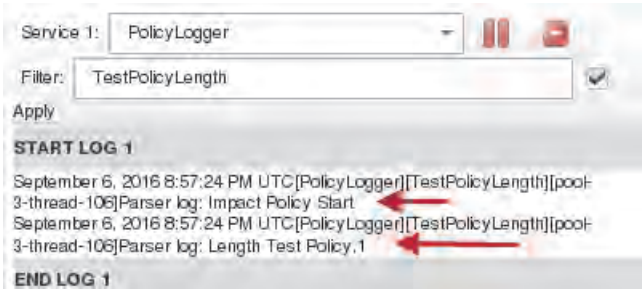


9. Click the **Check Syntax** icon to check for policy errors. Perform a syntax check and save the policy.

10. Click the right-pointing arrow to test the policy. Click **OK** to close the window.

11. Click the **Policy Log** icon to view the policy results.

PolicyLogger results for this policy look similar to the following screen capture.

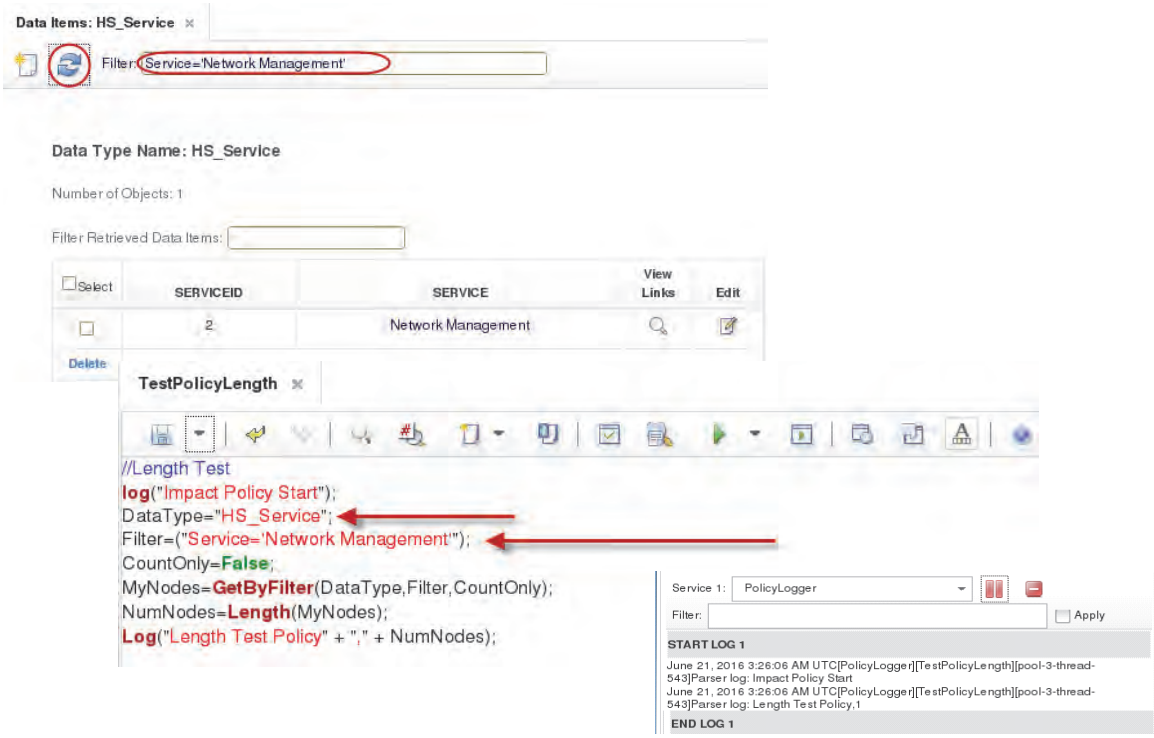


12. Modify the **TestPolicyLength** policy to test a filter for the **Service** table.
    Use the following information to build the filter expression:

    ```
    DataType: HS_Service    Filter:Service='Network Management'
    ```

    If the **HS_Service** table returns data using the filter expression, you can be confident when you add the filter expression to the policy.

    – Test the filter expression using the **Data Model** for the **HS_Service** table.
    – Modify the policy to add the filter.
    – Run the policy, then view the log to see the results.
    – The results should look similar to the following example.

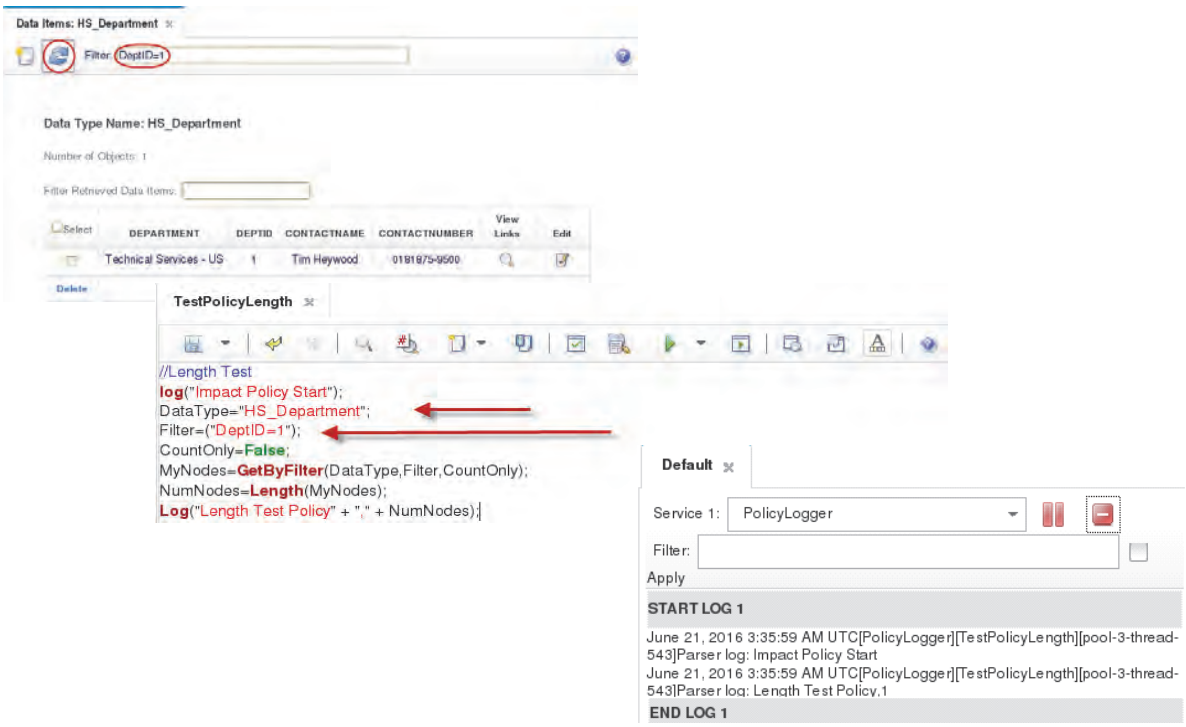


13. Modify the policy again to test a filter for the **HS_Department** table. Run the policy, and view the log to see the results.

Use the following information to build the filter expression:

```
DataType: HS_Department Filter:DeptID=1
```

The results should show as follows:



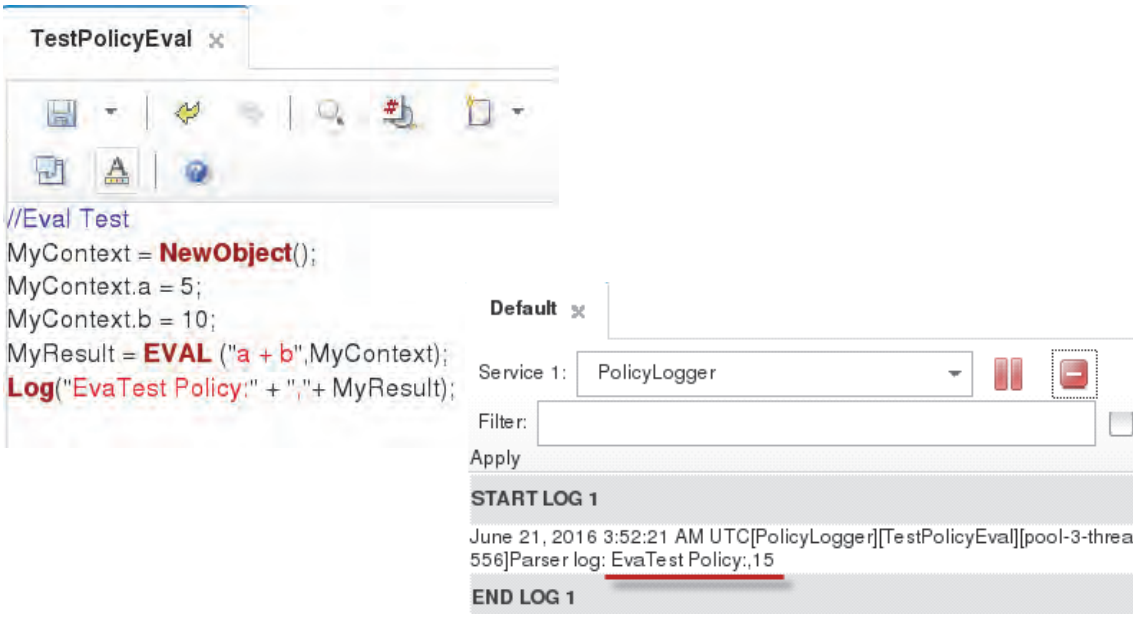
Exercise 4 Using the Eval function

1. Create the **TestPolicyEval** policy using the steps described in the previous exercises. Enter the policy data as displayed.

```
//Eval Test
MyContext = NewObject();
MyContext.a = 5;
MyContext.b = 10;
MyResult = EVAL ("a + b",MyContext);
Log("EvaTest Policy:" + ","+ MyResult);
```

2. Test the **TestPolicyEval** policy and verify the results.

The result looks similar to the following screen capture.



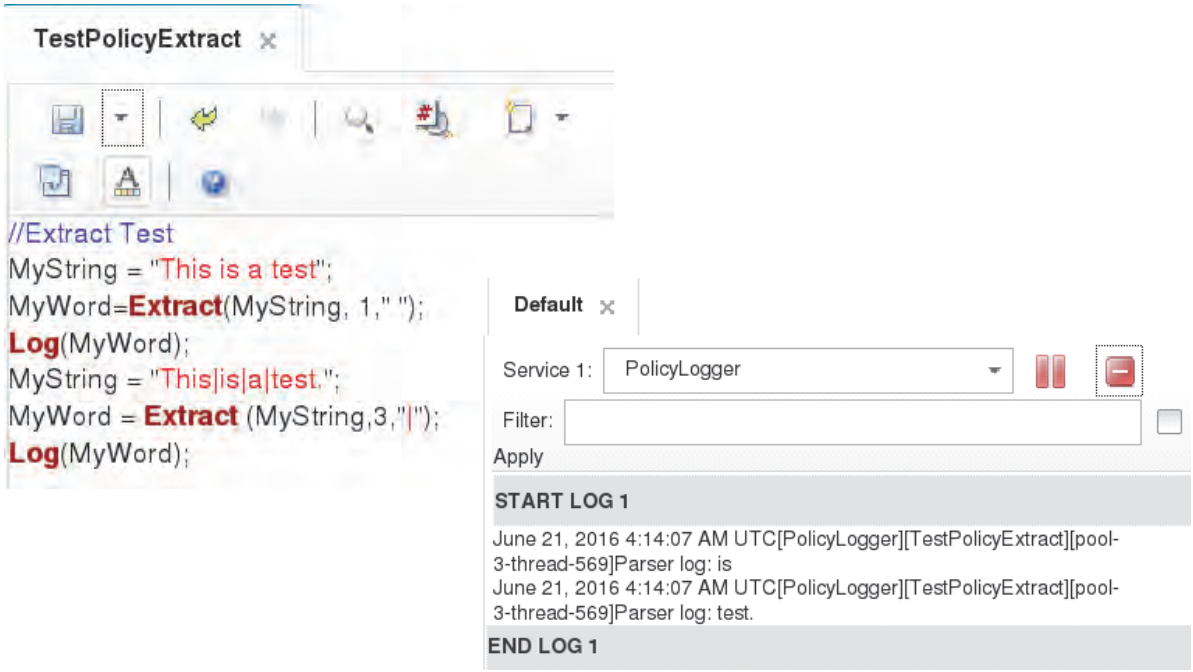
Exercise 5 Using the extract function

1. Create the **TestPolicyExtract** policy using the steps from the previous exercises. Enter the policy data as shown in the following screen capture.

```
//Extract Test
MyString = "This is a test";
MyWord=Extract(MyString, 1," ");
Log(MyWord);
MyString = "This|is|a|test.";
MyWord = Extract (MyString,3,"|");
Log(MyWord);
```

2. Test the **TestPolicyExtract** policy and verify the results.

The result looks similar to the following screen capture.



Unit 5 Services exercises

This unit has no student exercises.

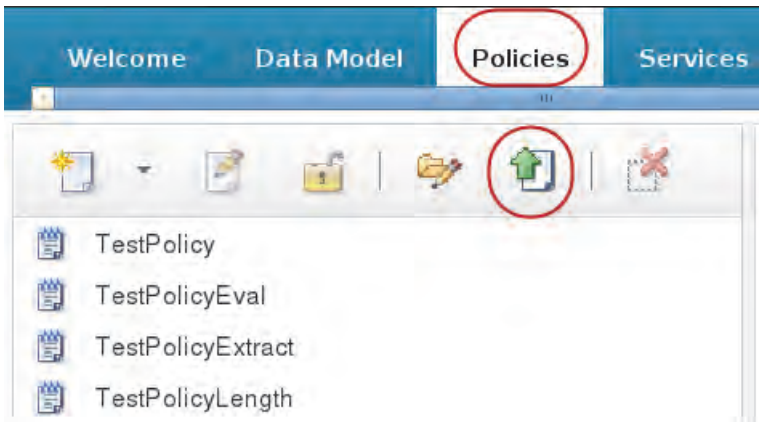
Unit 6 The Enrichment policy exercises

This unit focuses on the Enrichment Policy.

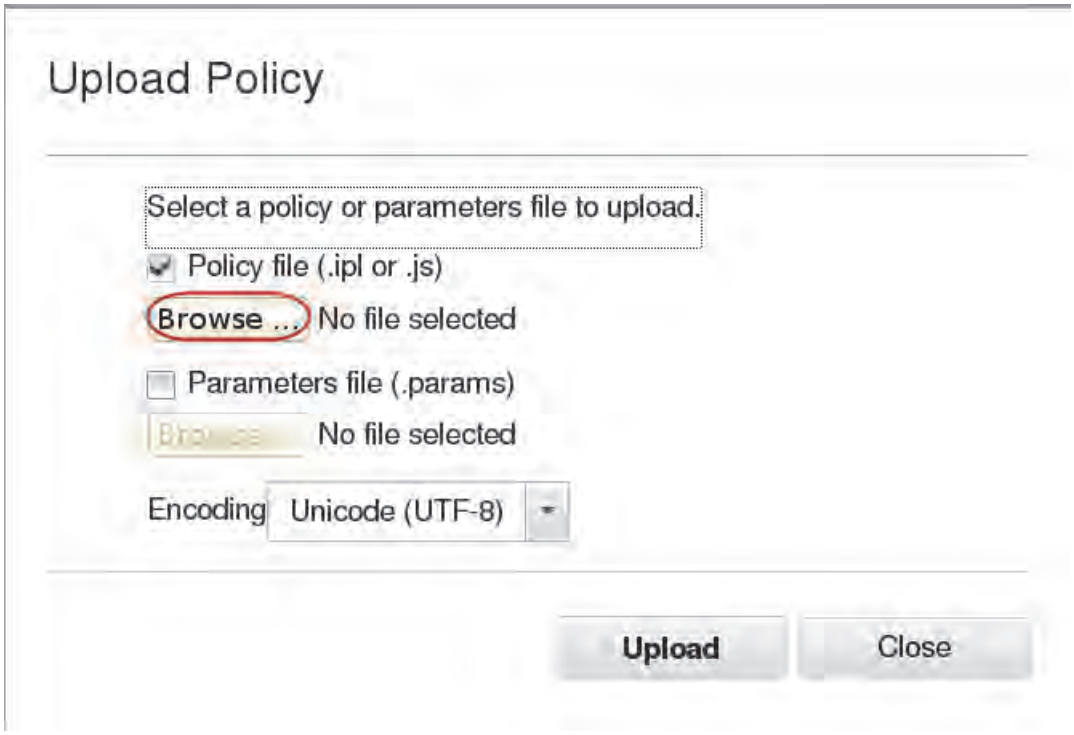
Exercise 1 Importing a policy with the policy uploader

In this exercise, you use the policy uploader to import a policy. The policy illustrates basic event enrichment by using the Impact Programming Language functions **GetByLinks** and **GetByKey**.

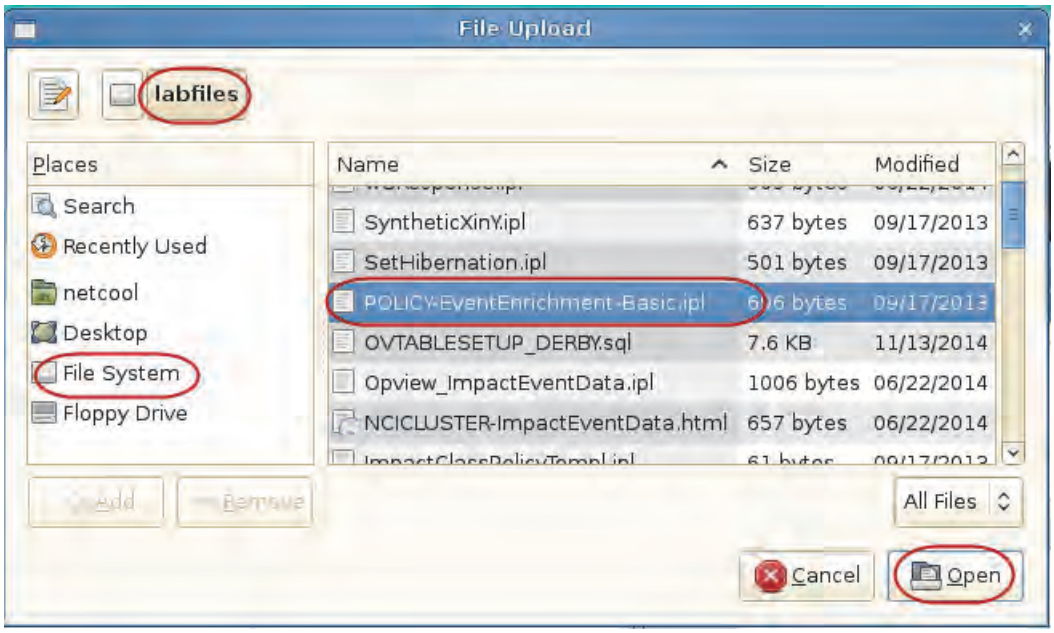
1. In the navigation tree, click **Policies** to open the **Policies** tab.
2. Click the **Upload a Policy File** icon.



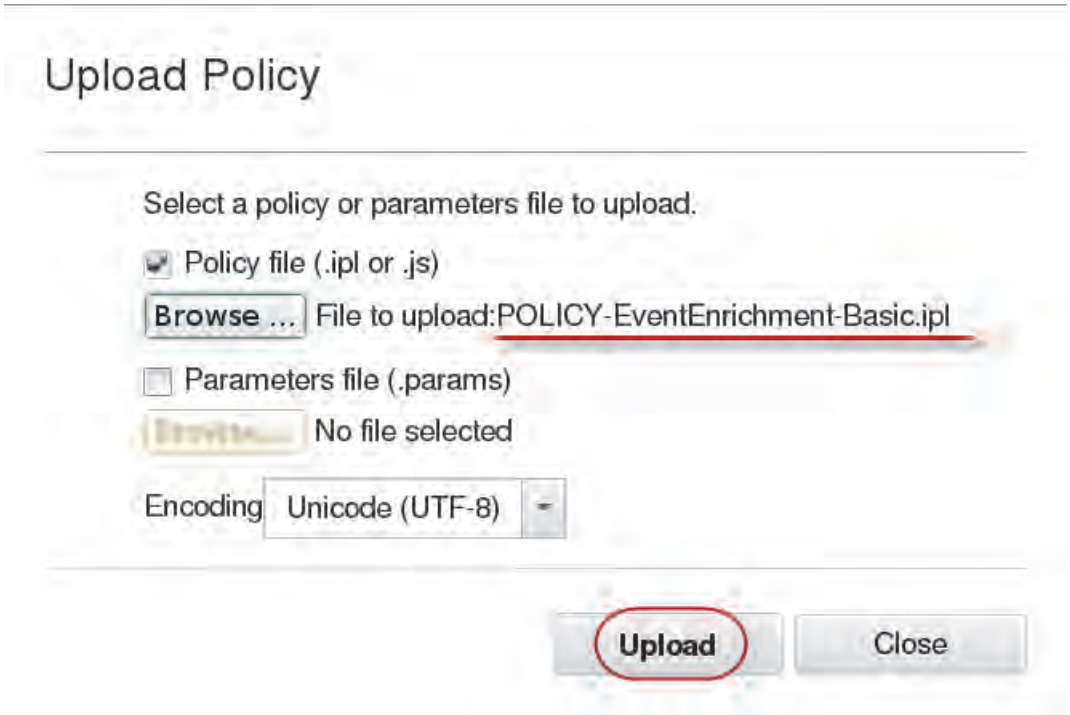
3. Click **Browse** to navigate to the **/labfiles** directory.



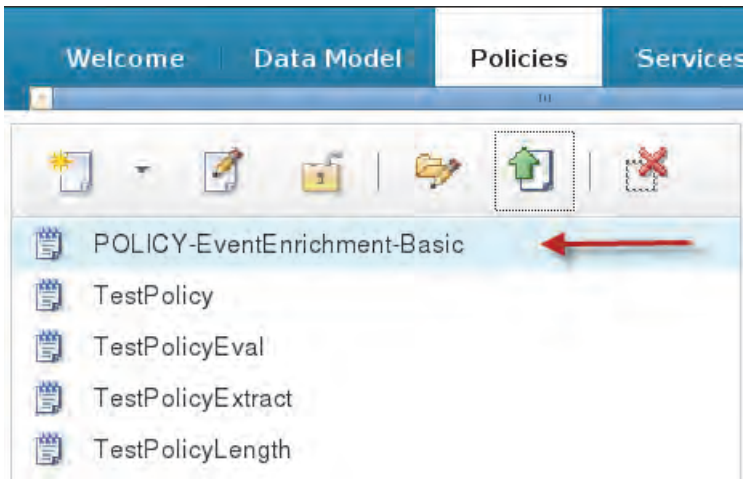
4. Highlight the **POLICY-Event_Enrichment-Basic.ipl** policy. Click **Open** to load the policy and close the window.



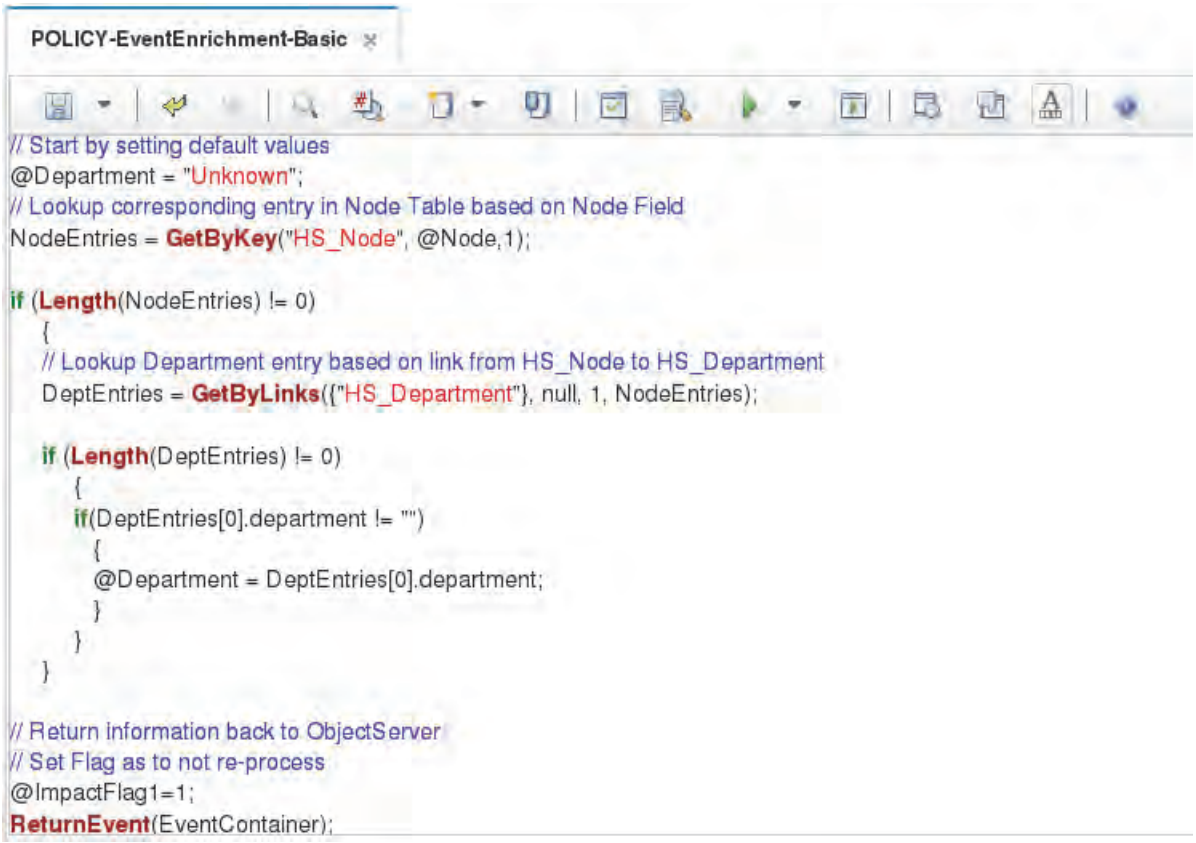
5. Click **Upload** to import the selected policy.



The policy is added to the **Policies** list as shown in the following display.



6. Right-click the **POLICY-EventEnrichment-Basic** policy. Select **EDIT** from the list to view the policy contents.Click **Save.**

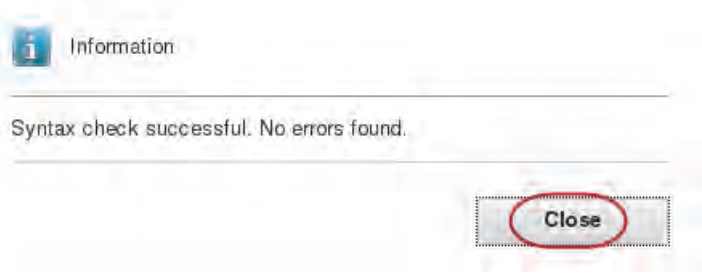


7. Determine the best location for each log statement. Edit the policy **POLICY-EventEnrichment-Basic** and add each of the following log statements**.**

```
// Start by setting default values
log("***Policy Start***");
log("Serial from the Event Container=======>"+@Serial);
log("Department set in the policy=======>"+@Department);
log("Node from the Event Container=======> "+@Node);
log("NodeEntries: Result of Lookup in HS_Node=======> "+NodeEntries);
log("Dept Entries: Result of lookup in HS_Department =======>"+DeptEntries);
log("Department Output to be sent to OMNIbus in the
EventContainer========>"+@Department);
```

8. Click the check syntax icon to verify there are no errors. 

9. Click **OK** to close the window. Save the policy.



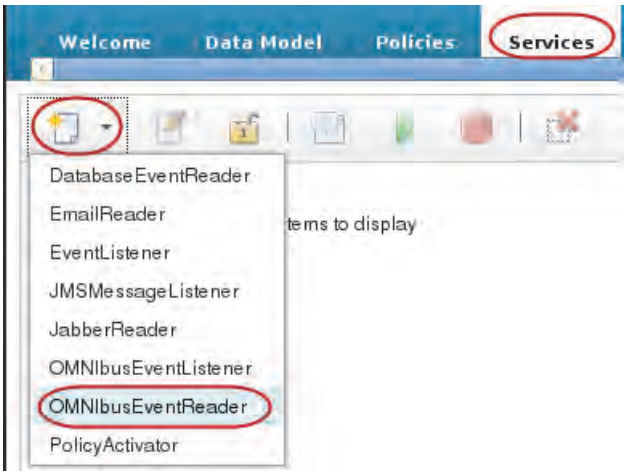
The resulting policy should look similar to the following display.

POLICY-EventEnrichment-Basic

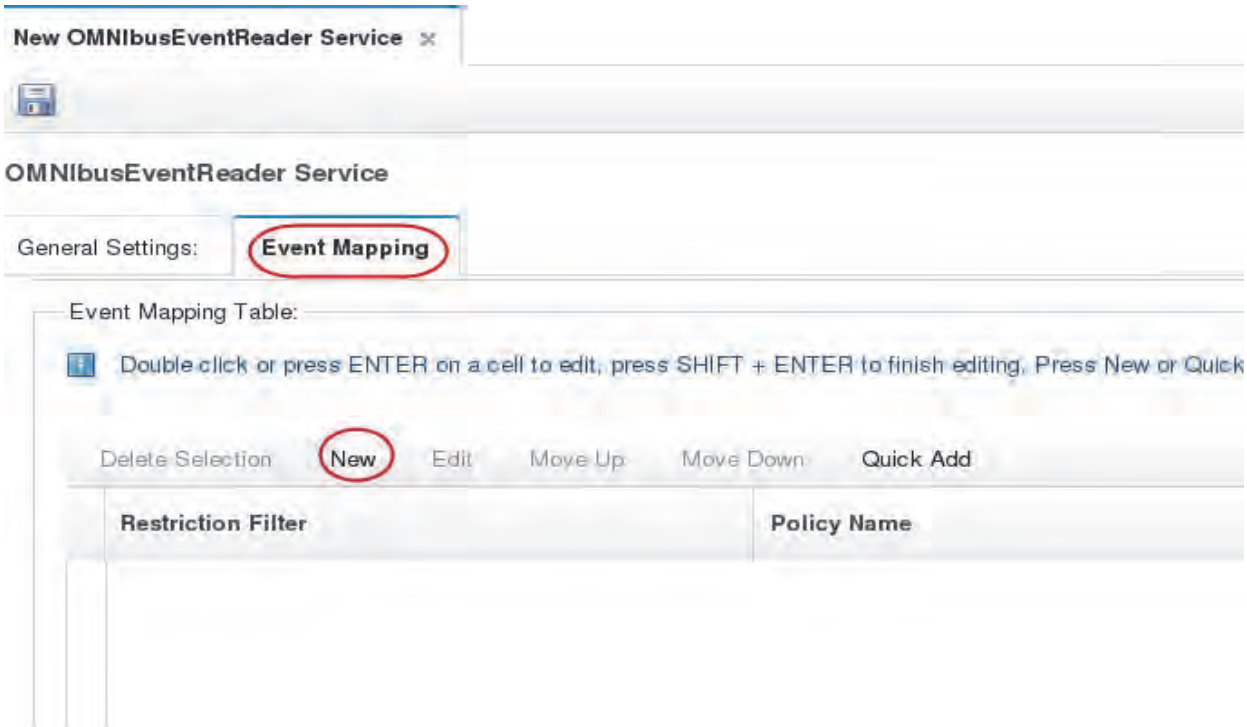
```
// Start by setting default values
log("***Policy Start***");
log("Serial from the Event Container========>"+@Serial);
@Department = "Unknown";
log("Department set in the policy========>"+@Department);
log("Node from the Event Container========> "+@Node);
// Lookup corresponding entry in Node Table based on Node Field
NodeEntries = GetByKey("HS_NODE", @Node,1);
log("NodeEntries: Result of Lookup in HS_Node========> "+NodeEntries);
if (Length(NodeEntries) != 0)
   {
   // Lookup Department entry based on link from HS_Node to HS_Department

DeptEntries = GetByLinks({"HS_Department"}, null, 1, NodeEntries);
log("Dept Entries: Result of lookup in HS_Department ========>"+DeptEntries);
   if (Length(DeptEntries) != 0)
      {
      if(DeptEntries[0].department != "")
        {
        @Department = DeptEntries[0].department;
        }
      }
   }
log("Department Output to be sent to OMNIbus in the EventContainer=========>"+@Department);
// Return information back to ObjectServer
// Set Flag as to not re-process
@ImpactFlag1=1;
ReturnEvent(EventContainer);
```

Exercise 2 Configuring event mapping in the OMNIbus Event Reader service

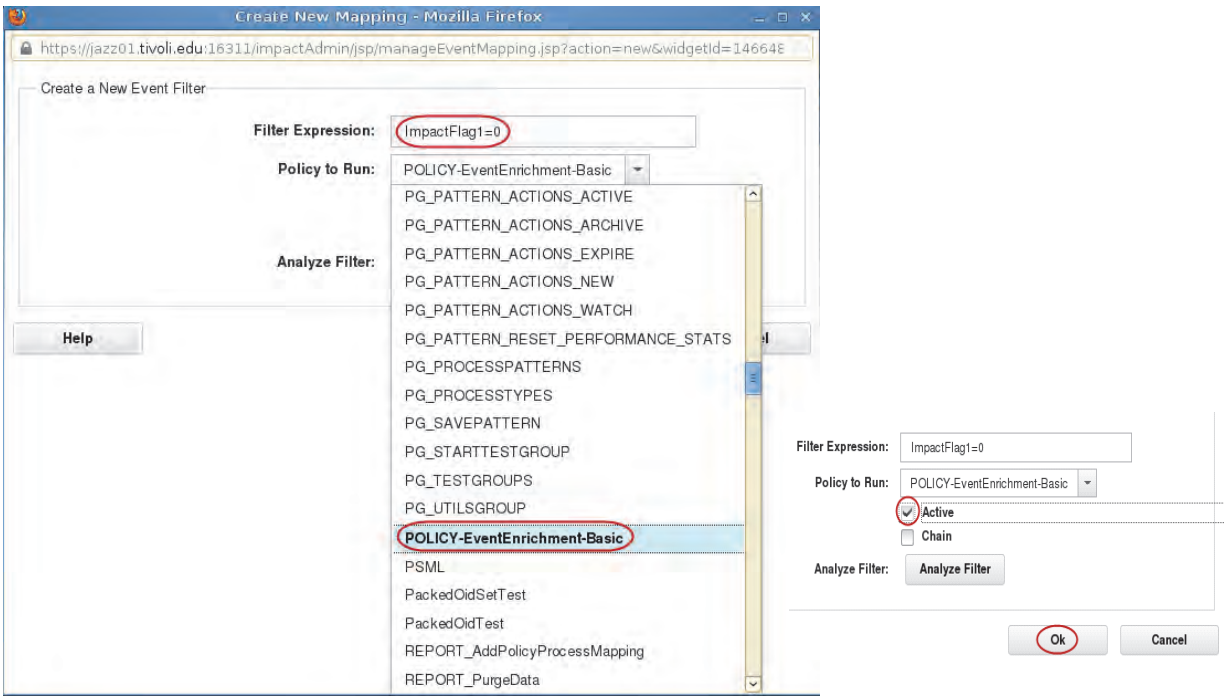
1. Click **Services** to open the **Services** tab. Click the **Create New Service** icon. Select **OMNIbusEventReader** in the list.



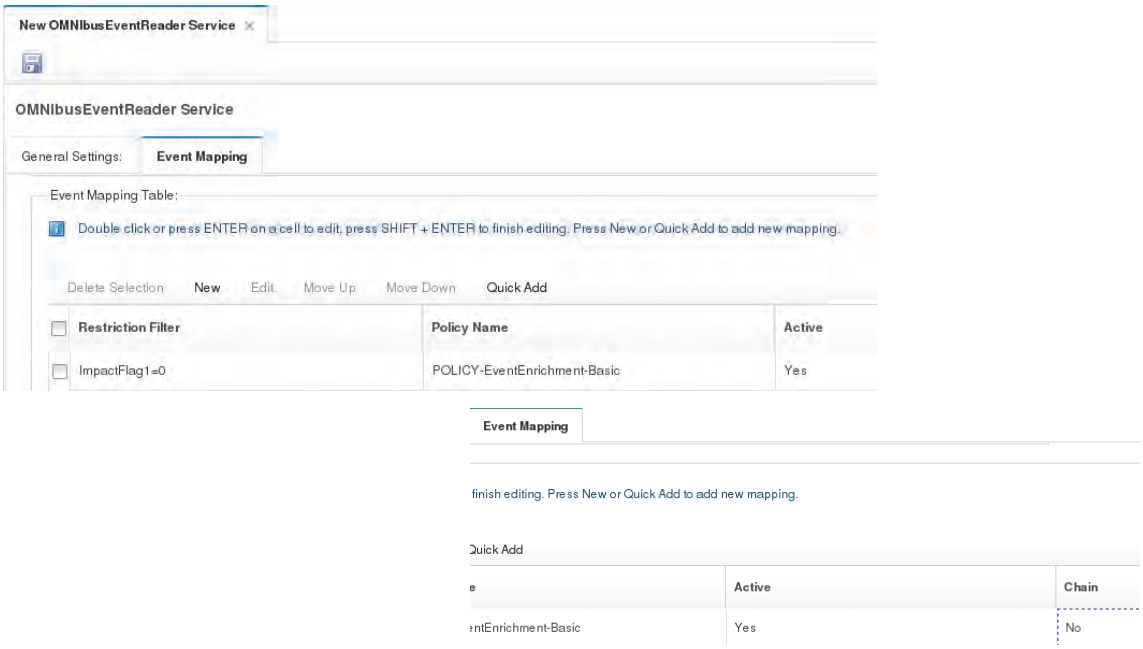
2. Click the **Event Mapping** tab. Scroll down to the **Event Mapping Table**. Click **New** to create a new mapping for the policy.



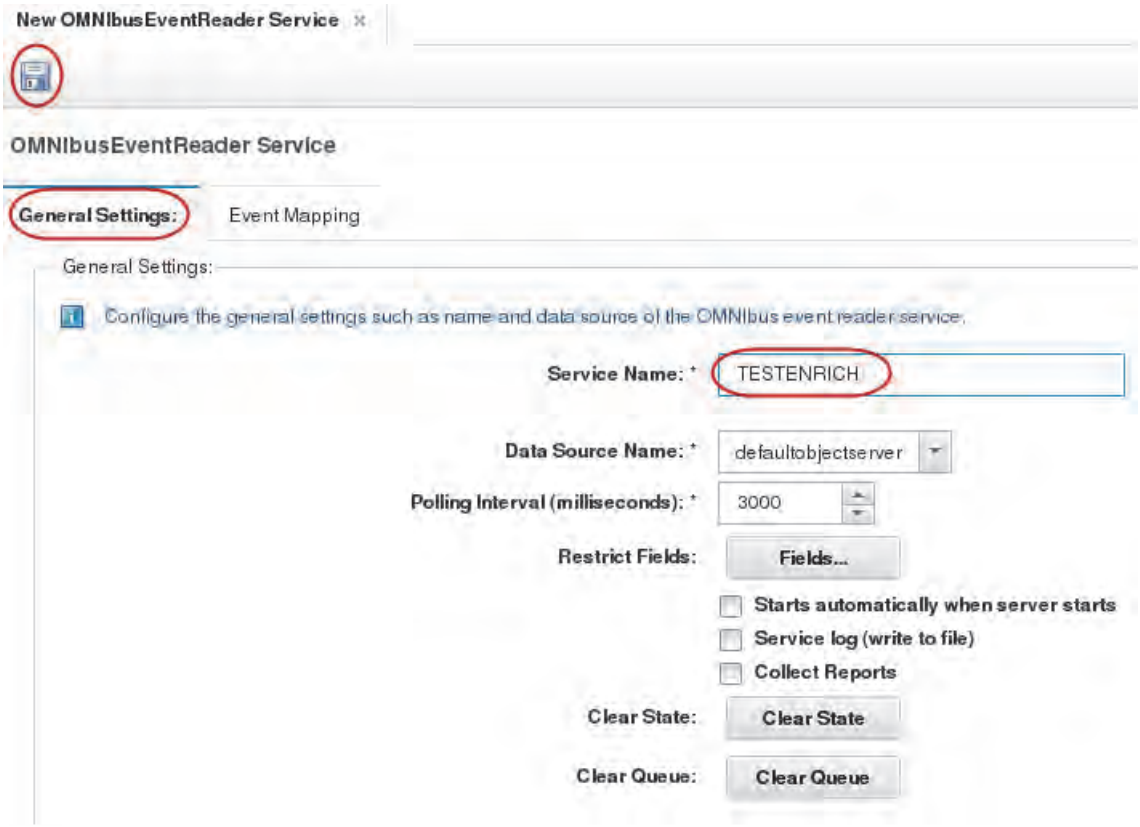
3. In the **Edit Event Filter** window, perform the following steps:
   a. In the **Filter Expression field**, type **ImpactFlag1=0**.
   b. Select **POLICY-EventEnrichment-Basic** from the **Policy to Run** list.
   c. Select the **Active** check box**.** Click **OK** to close the window.



The policy shows in the mapping table as shown in the following screen capture. When this service starts, the **POLICY-EventEnrichment-Basic** policy runs. The filter selects events to pass to the policy that have a zero value in the alerts.status field; **ImpactFlag1=0**.



4. Click the **General Settings** tab. Type **TESTENRICH** as the service name. Click the **Save** icon.



The new service shows in the **Services** list as shown in the following screen capture.

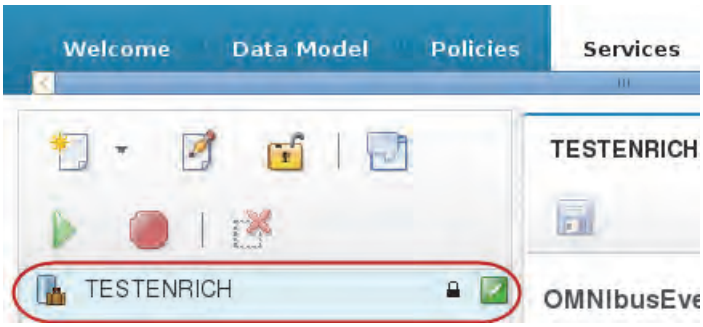
5. Double-click the GNOME Terminal icon on the desktop.

6. To generate OMNIbus events, start the *Simnet Probe* by typing the following commands from the command line as shown:

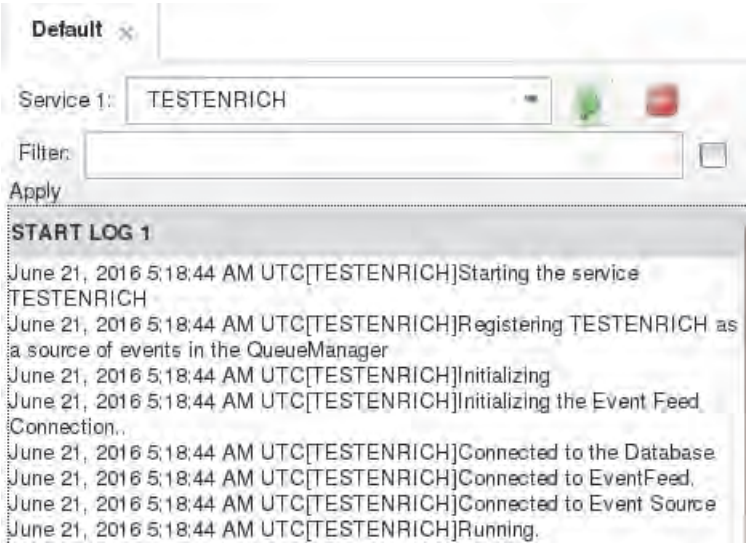
```
cd /opt/IBM/tivoli/netcool/omnibus/probes
./nco_p_simnet &
```

7. Start the **TESTENRICH** service by clicking the **Start Service** icon.

The check mark indicates the service is running.

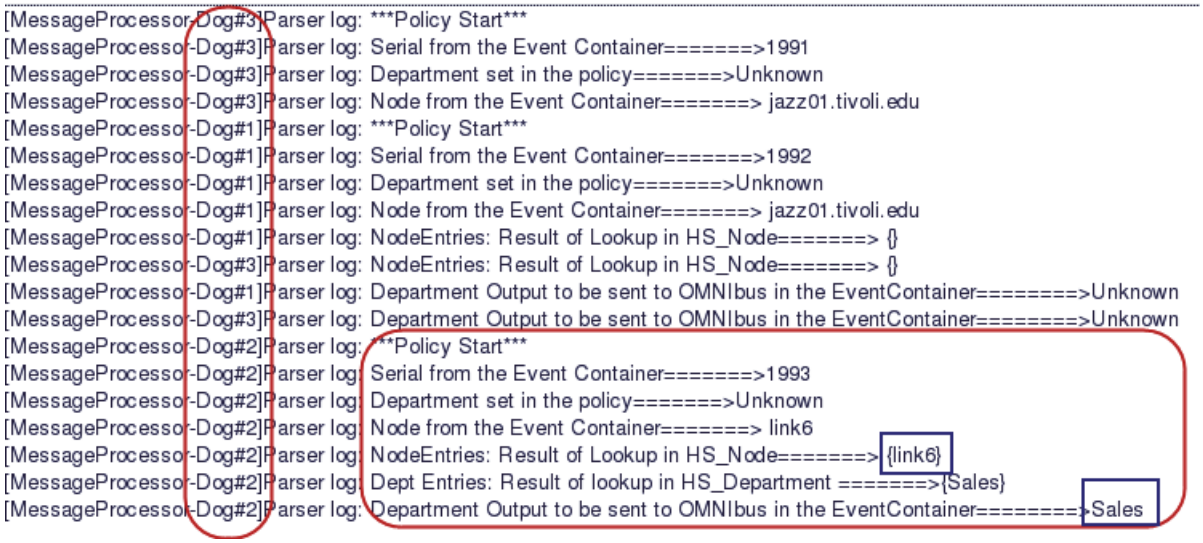


8. Click the icon to view the Service Log  . Verify messages in the Service Log output.

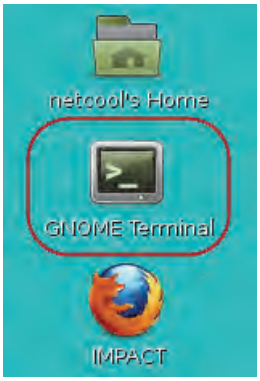


Exercise 3 Verifying the results of the POLICY-Event_Enrichment-Basic policy

1. Edit the **POLICY-Event_Enrichment-Basic** policy window. Click the **Policy Log** icon to view results of the log statements.



2. Double-click the **GNOME Terminal** icon on the desktop.



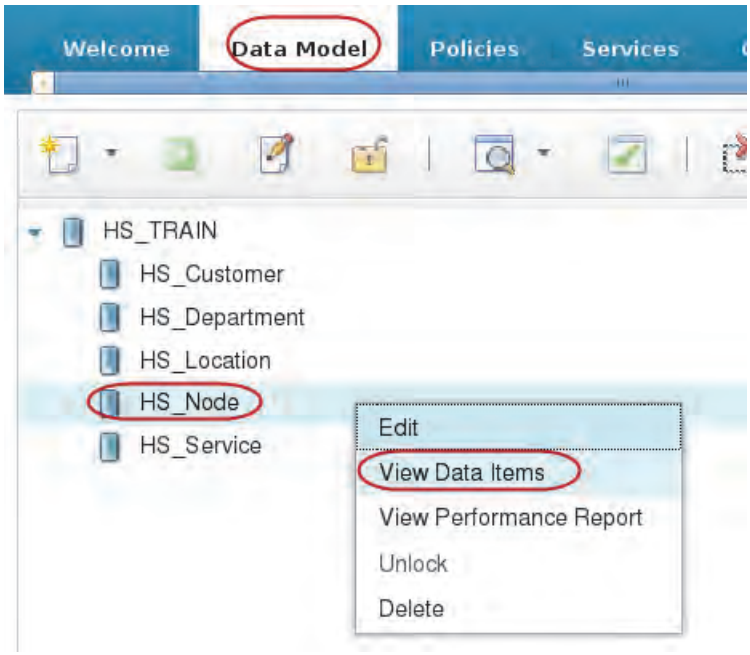
3. Run the script to verify the updates to the OMNIbus **alerts.status** table using the following commands:

```
cd /opt/IBM/tivoli/netcool/omnibus/bin
./nco_sql -server NCOMS -user root -password object00 -i /labfiles/OMNIbusQuery
```

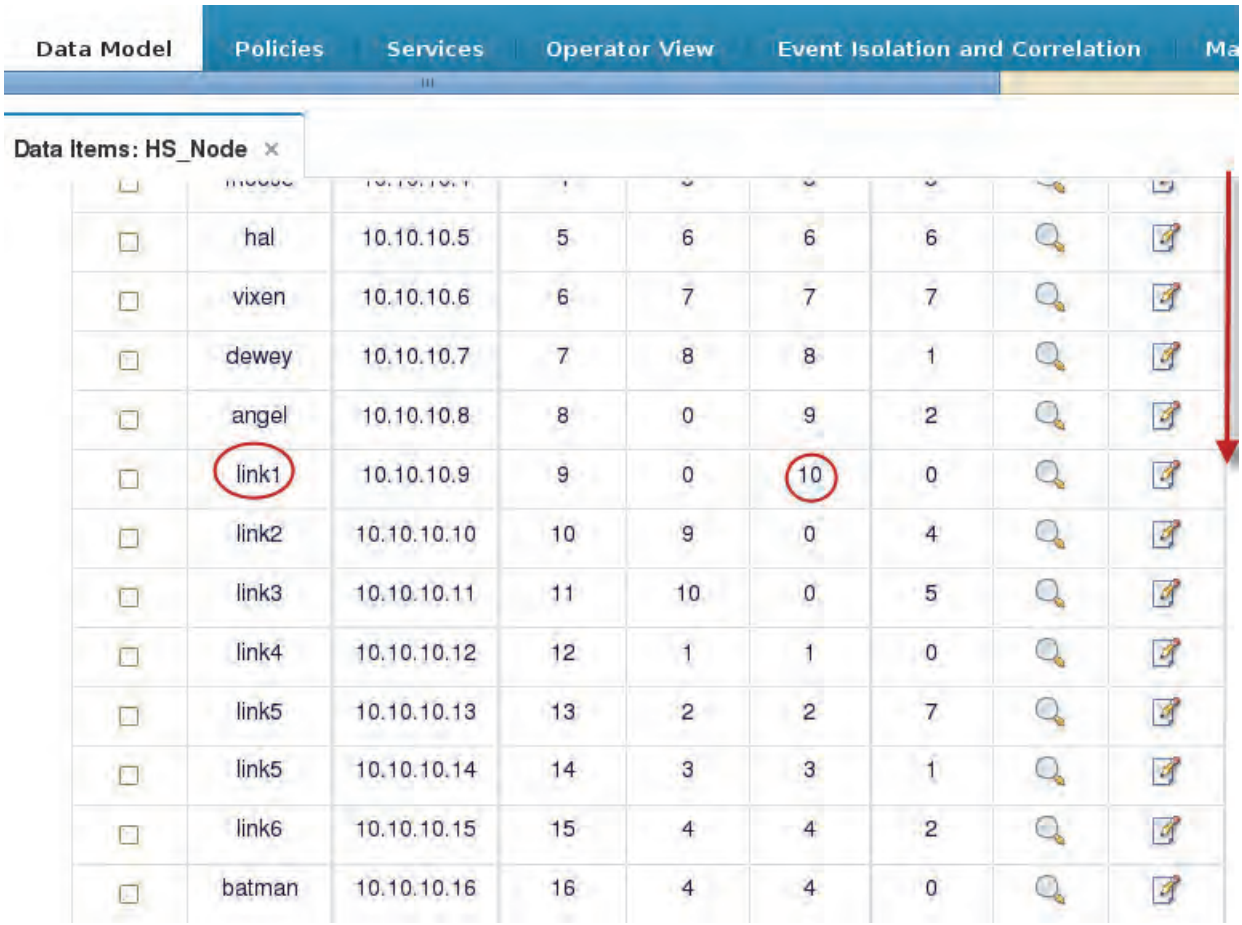
The **Node** and **Department** entries in OMNIbus should look similar to the following display.

Note: The number of rows affected varies depending on how long the probe has been running.

4. Compare the results to the table data. Click the **Data Model** tab. Right-click **HS_Node** and select **View Data Items**.



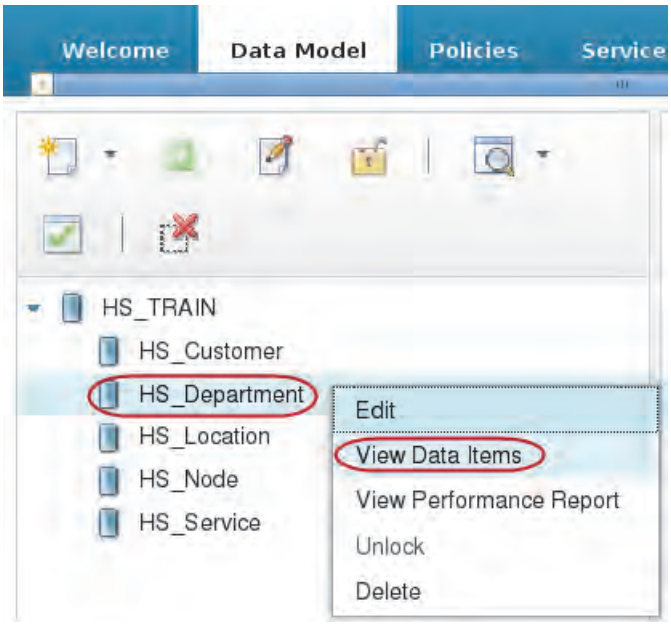
The **HS_Node** entries are shown similar to the display.



☐	hal	10.10.10.5	5	6	6	6		
☐	vixen	10.10.10.6	6	7	7	7		
☐	dewey	10.10.10.7	7	8	8	1		
☐	angel	10.10.10.8	8	0	9	2		
☐	link1	10.10.10.9	9	0	10	0		
☐	link2	10.10.10.10	10	9	0	4		
☐	link3	10.10.10.11	11	10	0	5		
☐	link4	10.10.10.12	12	1	1	0		
☐	link5	10.10.10.13	13	2	2	7		
☐	link5	10.10.10.14	14	3	3	1		
☐	link6	10.10.10.15	15	4	4	2		
☐	batman	10.10.10.16	16	4	4	0		

Recall the links between tables created earlier. The HS_Node table uses **DEPTID** to link to the HS_Department table.

5. Compare the results to the table data. Click the **Data Model** tab. Right-click **HS_Department** and select **View Data Items**.



The **HS_Department** entries are shown similar to the display.

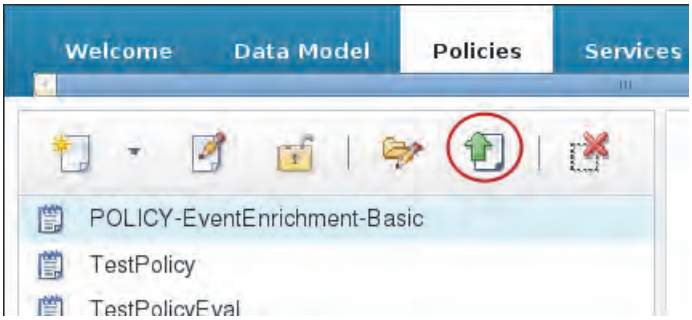
Data Items: HS_Department

Select	DEPARTMENT	DEPTID	CONTACTNAME	CONTACTNUMBER	View Links	Edit
☐	Technical Services - US	1	Tim Heywood	0181875-9500		
☐	Technical Services - EMEA	2	Laurent Marchal	0181875-9500		
☐	Finance	3	Anthony Isaacs	0181875-9500		
☐	Sales	4	Mike Donohue	0181875-9500		
☐	Marketing	5	Evan Birkhead	0181875-9500		
☐	Technical Support	6	Mike Foster	0181875-9500		
☐	Development	7	Adam Kerrison	0181875-9500		
☐	Maintenance	8	Jim Bleecker	0181875-9500		
☐	Product Management	9	Peter Shearan	0181875-9500		
☐	Training Department	10	Sarah Cambridge	0181875-9500		

Exercise 4 Adding functions to the enhanced event enrichment policy

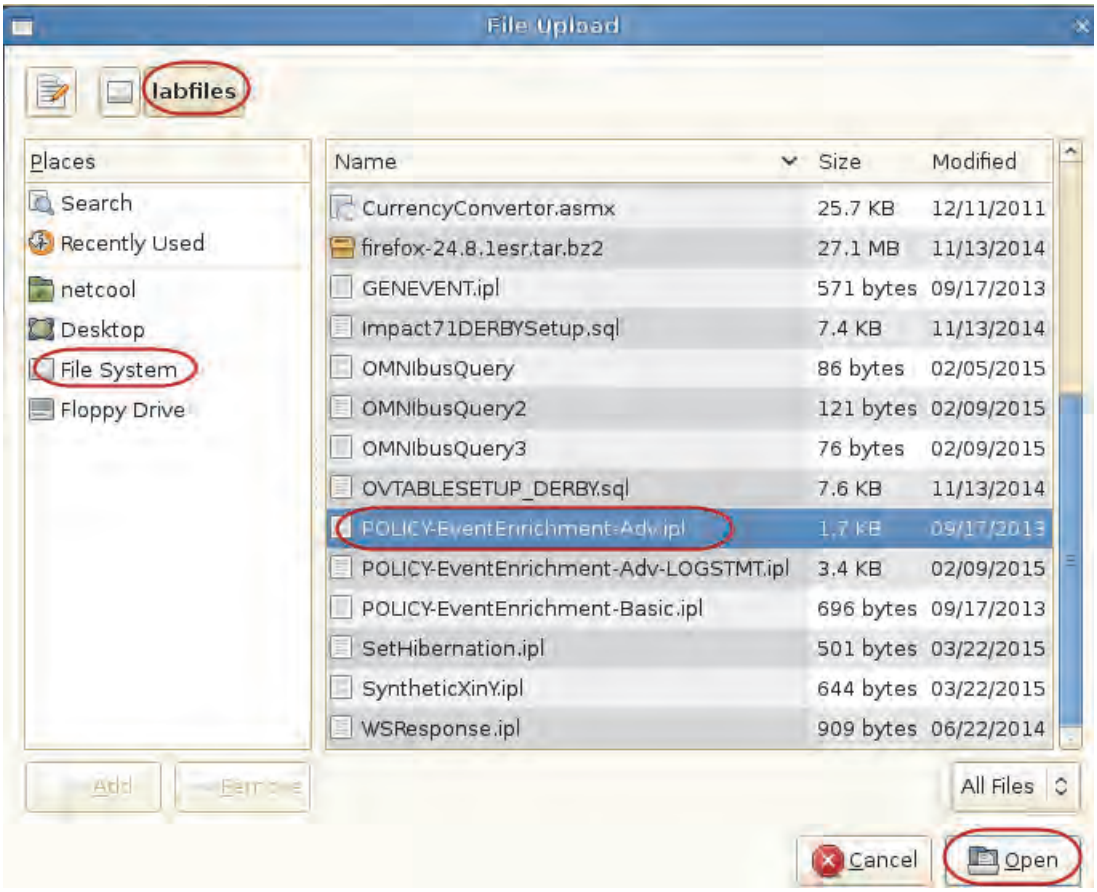
In this exercise, you add parsing and enrichment functions to the Event Enrichment policy created in the previous exercise. You populate the additional OMNIbus alert.status fields **Location**, **SLA**, **WorkingHours**, and **Customer**. You implement **Severity** and **Event Suppression**.

1. Click the **Upload a Policy File** icon.



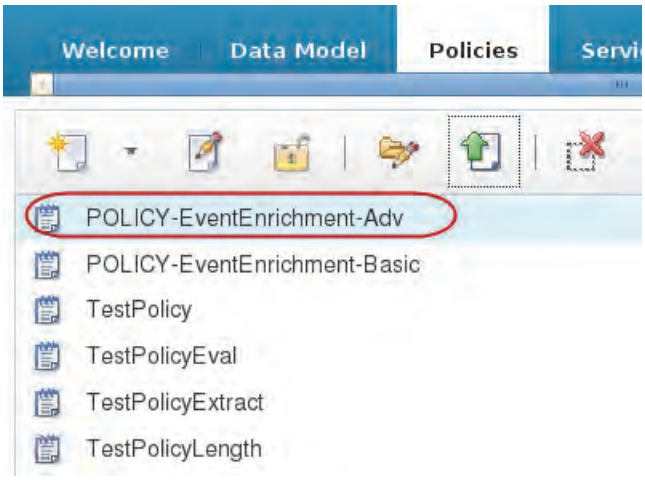
2. Click **Browse** to navigate to the **/labfiles** directory.

3. Click **File System>labfiles** and select the **POLICY-EventEnrichment-Adv** policy. Click **Open**.



4. Click **Upload** to import the policy.

The policy is added to the **Policies** list as shown in the following screen capture.



5. Modify the policy and add **Log** statements to verify the policy flow. See the examples in the following screen capture.

Note: Pay close attention the bracket ({}) placements. Be sure to check syntax.

a. Log statements: Policy Start, Input Key, GetByKey HS_Node

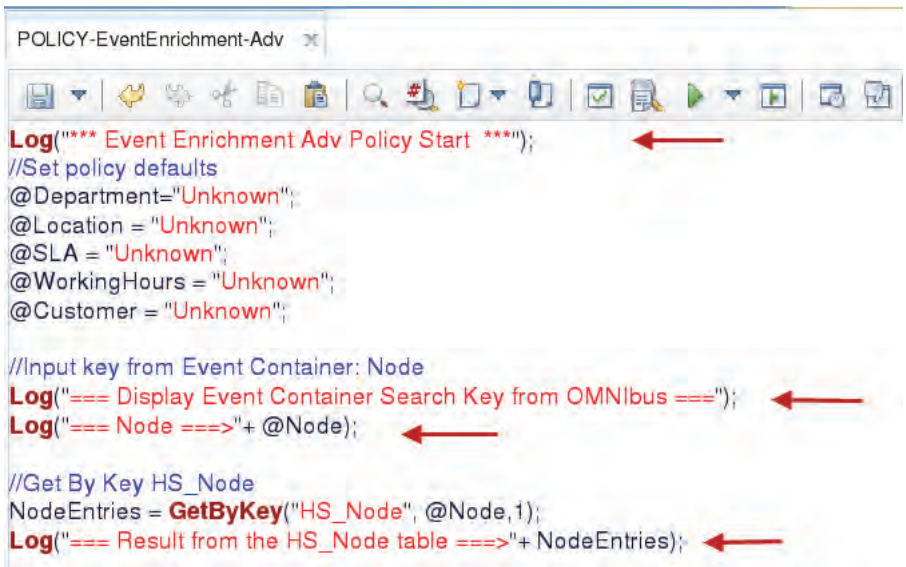


Figure 1 Policy logic screen 1

b. Log statements: GetByLinks HS_Department and HS_Customer, Process Department

```
//Get By Links
if (Length(NodeEntries) > 0) {
  @Service = NodeEntries[0].Service;
  Log("===  @Service value set from HS_Node ===>"+@Service);
  DeptEntries = GetByLinks({"HS_Department"}, null, 1, NodeEntries);
  Log("===  Result from HS_Department search  ===>"+DeptEntries);
  CustomerEntries = GetByLinks({"HS_Customer"}, null, 1, NodeEntries);
  Log("===  Result from HS_Customer search  ===>"+CustomertEntries);
}
//Process Department
if(Length(DeptEntries) > 0) {
  if(DeptEntries[0].department != "") {
    @Department = DeptEntries[0].department;
    Log("===  @Department value set from link ===>"+@Department);
}
}
```

Figure 2 Policy logic screen 2

c. Log Statements: Process Customer

```
//Process Customer
if (Length(CustomerEntries) > 0) {
  @Customer = CustomerEntries[0].customer;
  Log("===  @Customer value set from link ===>"+@Customer);
  @SLA = CustomerEntries[0].servicelevel;
   Log("===  @SLA value set from link ===>"+@SLA);
  SLAValue=rextract(@SLA,"([0-9]+)\%");
   Log("===  SLAValue from extract ===>"+SLAValue);
     if(SLAValue != "") {
     if(int(SLAValue) <=95) {
         @Severity=4;
         }
         else {
         @Severity=5;
         }
        Log("===  @Severity value calculated ===>"+@Severity);
        }
```

Figure 3 Policy logic screen 3

d. Log statements: Working Hours Calculation

```
//Set working hours only if CustomerEntries is grater than 0
  @WorkingHours = CustomerEntries[0].workinghours;
 Log("===  @WorkingHours set from HS_Customer ===>"+@WorkingHours);
if(@WorkingHours != "24 Hours") {
   WHTime = @WorkingHours;
   Log("===  WHTime value calculated ===>"+WHTime);
   StartMinute = int(rextract(WHTime,".*:([0-9]+)-"));
   Log("===  StartMinute ===>"+StartMinute);
   StartTime = int(rextract(WHTime,"([0-9]+):.*-"))*60 + StartMinute;
   Log("===  StartTime value calculated ===>"+StartTime);
   EndMinute = int(rextract(WHTime,"-.*:([0-9]+)"));
   Log("===  EndMinute value calculated ===>"+EndMinute);
   EndTime = int(rextract(WHTime,"-([0-9]+):"))*60 + EndMinute;
   CurrentMinute = int(localtime(@LastOccurrence,"mm"));
   Log("===  CurrentMinute value calculated ===>"+CurrentMinute);
   CurrentTime = int(localtime(@LastOccurrence,"HH"))*60 + CurrentMinute;
    Log("===  CurrentTime value calculated ===>"+CurrentTime);
if((CurrentTime < StartTime) or (CurrentTime > EndTime)) {
 @SuppressEscl = 4;
}
 Log("===  @SuppressEscl  value set ===>"+@SuppressEscl);
}
```

Figure 4 Policy logic screen 4

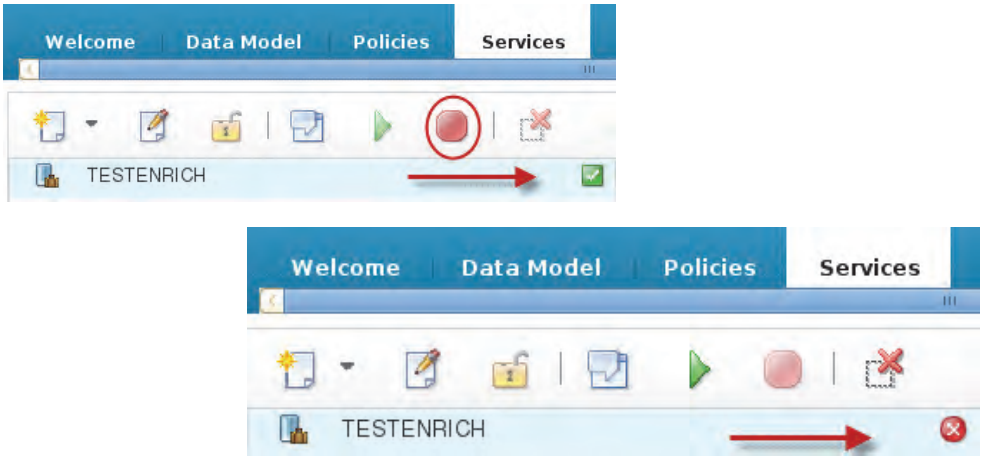
e. Log statements: GetByLinks HS_Location, set ImpactFlag2, Return Event

```
//Set Location only if CustomerEntries is greater than 0
//Get By Links
 LocationEntries = GetByLinks({"HS_Location"}, null, 1, CustomerEntries);
 Log("===  LocationEntries from HS_Location ===>"+LocationEntries);
   if (Length(LocationEntries) > 0) {
      temploc= LocationEntries[0].location;
       Log("===  temploc value calculated ===>"+temploc);
      location=rextract(temploc,".*:(.*):.*") + " " + rextract(temploc,".*:.*:(.*)$");
     @Location = location;
      Log("===  @Location value calculated ===>"+@Location);
}
//end bracket below for CustomerEntries check
}
//Set Impact reprocess flag
@ImpactFlag2 = 1;
ReturnEvent(EventContainer);
```

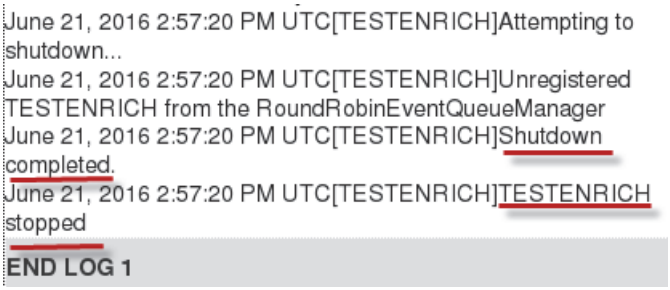
Figure 5 Policy logic screen 5

Exercise 5 Testing the enhanced enrichment policy

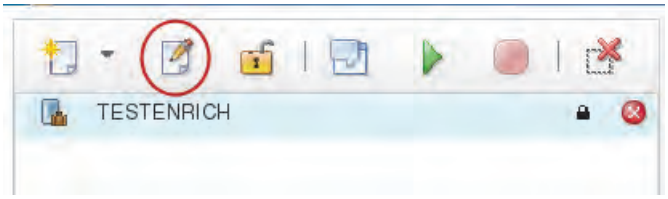
1. Stop the **TESTENRICH** service by clicking the red **Stop Service** button. The green check box changes to a red circle with an *x* when the server stops.



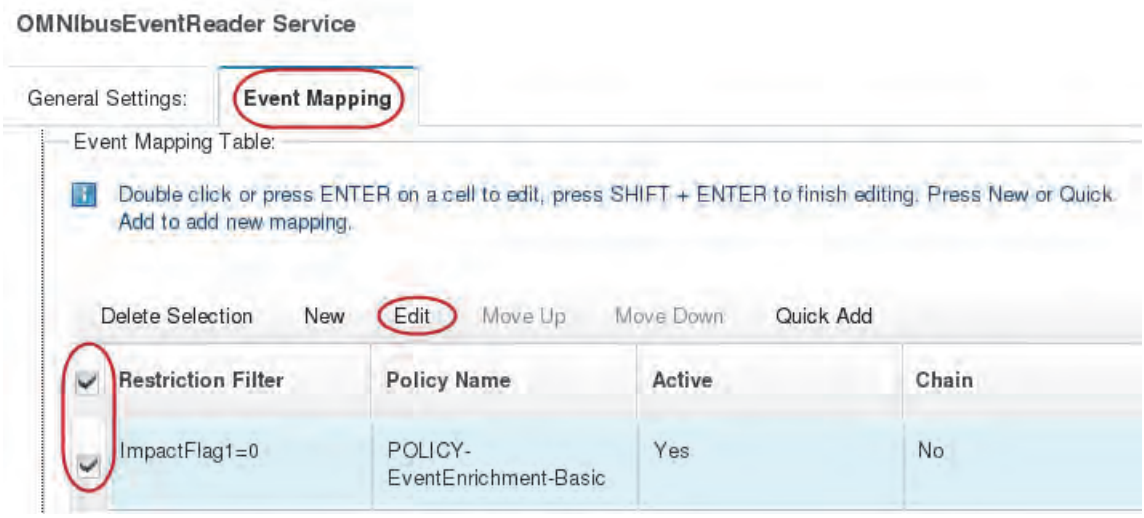
2. Click the Service Log icon to verify the server has stopped. 
   The shutdown messages should look similar to the following display.



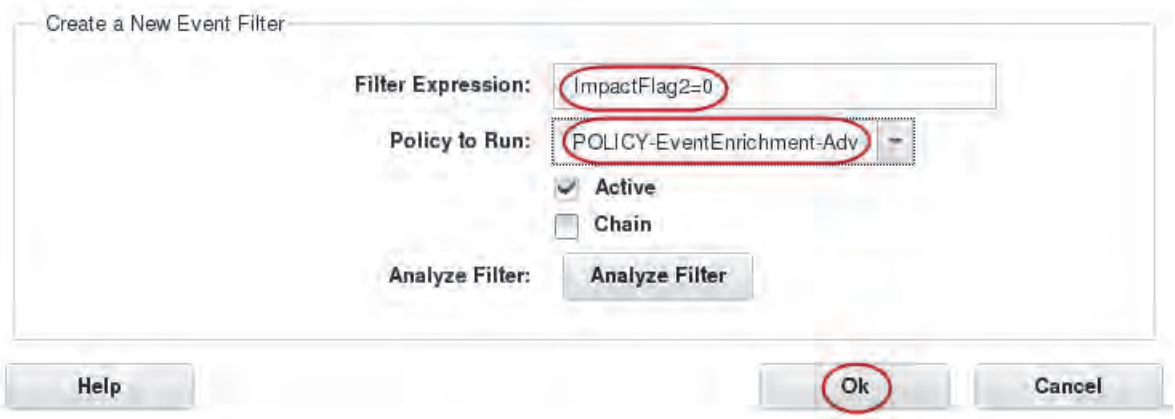
3. Edit the **TESTENRICH** service by clicking the pencil icon.



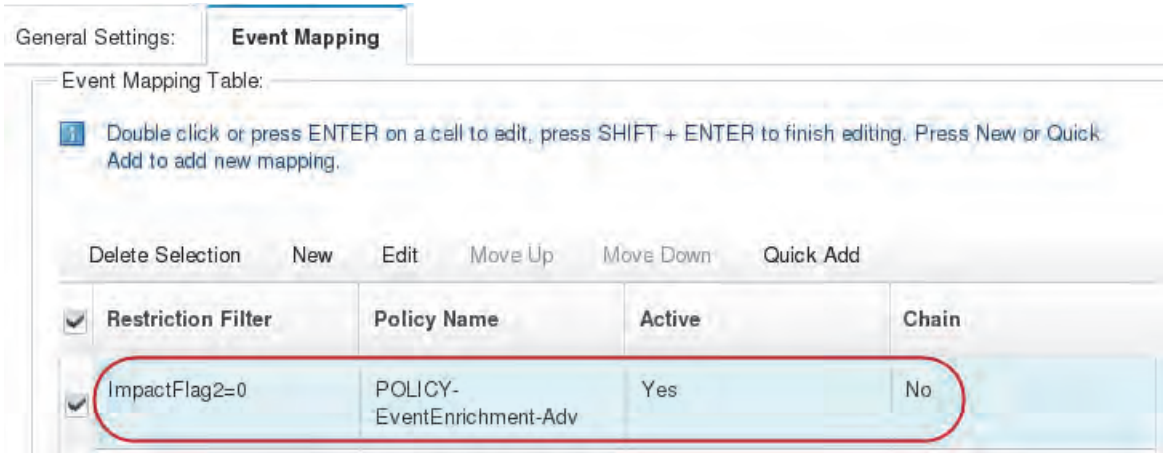
4. Click the **Event Mapping** tab. Scroll down to the **Event Mapping Table**. Select **Policy-EventEnrichment-Basic**. Click **Edit**.



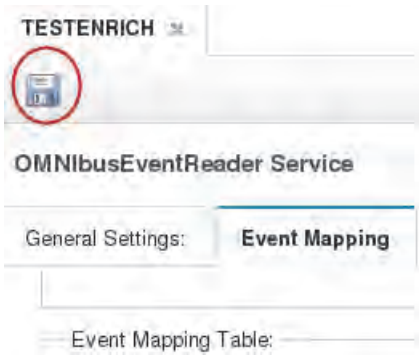
5. Change the Filter Expression to use **Impact Flag2= 0**. Select **POLICY-EventEnrichment-Adv** from the **Policy to Run** drop down list**.** Verify that the **Active** checkbox is checked. Click **OK.**



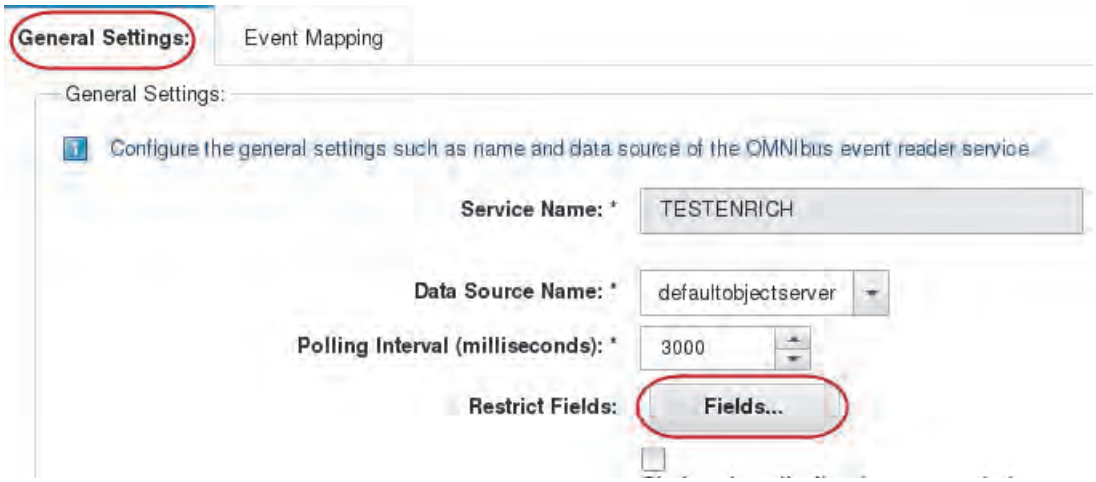
The policy shows in the mapping table as shown in the following screen capture. When this service starts, the **POLICY-EventEnrichment-Adv** policy runs. The filter selects events to pass to the policy that have a zero value in the alerts.status field; **ImpactFlag2=0**.



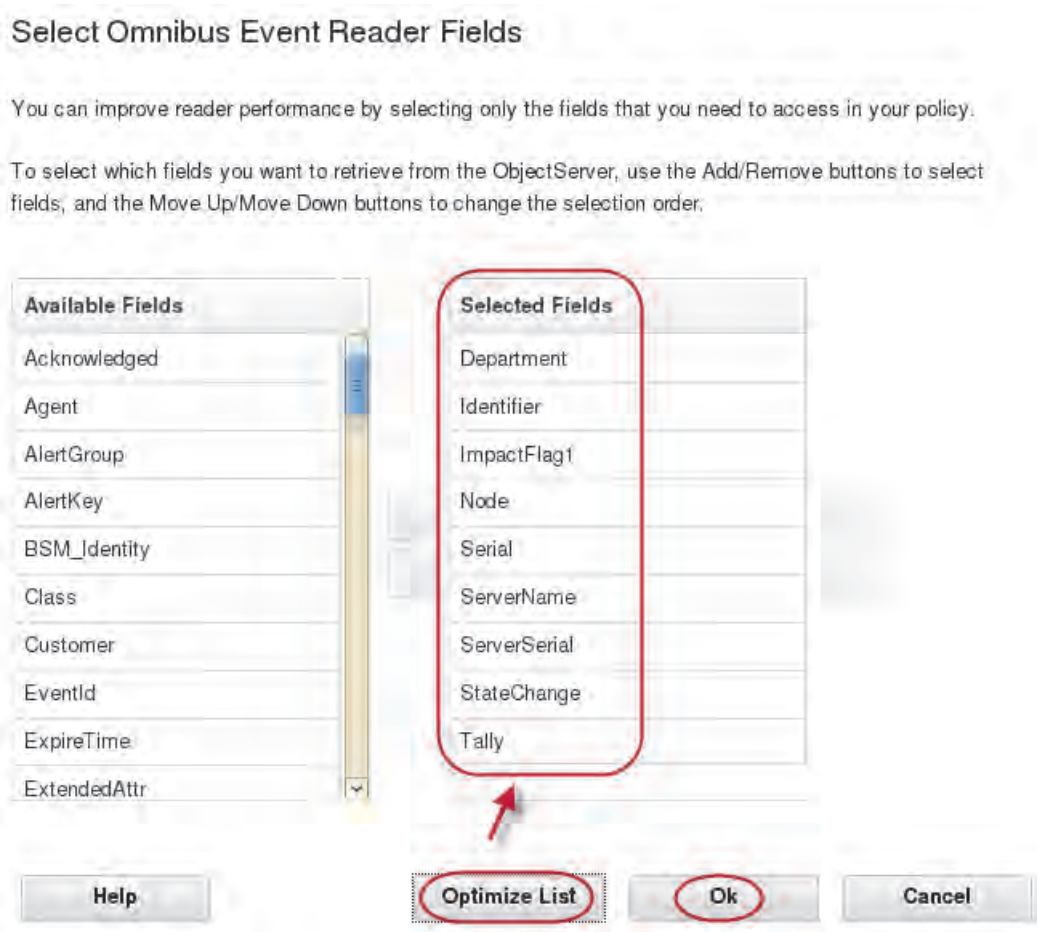
6. Save the changes. Click the Save icon.



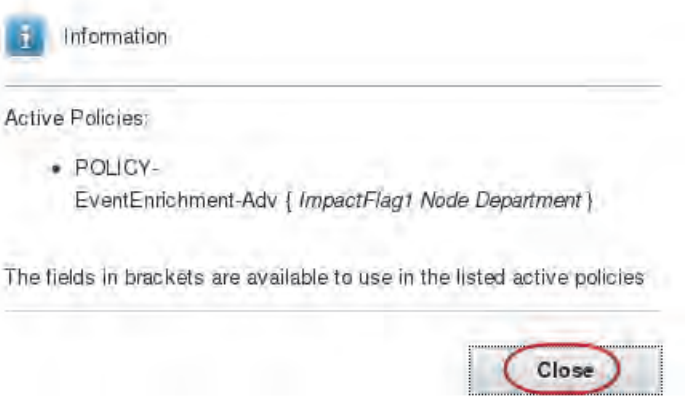
7. On the **General Settings** tab, click **Fields**.



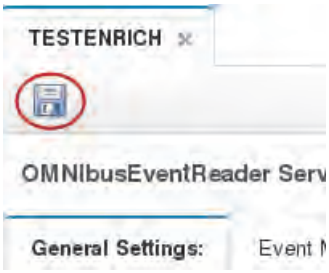
8. Click **Optimize List** to select the OMNIbus fields needed for this service.



9. **Active Policies** displays the new policy **POLICY-EventEnrichment-Adv**. Click **Close.**



10. Save the changes. Click the Save icon.



Exercise 6 Verifying the results of the POLICY-EventEnrichment-Adv policy

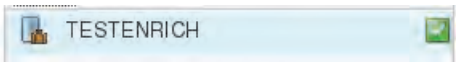
Important: Review the policy logic using log statements in the Policy Log and check the table values in OMNIbus.

For example, verify that the following statements are true:

```
If the event Severity = 4, the SLA value <= to 95, and
If the event Severity = 5, the SLA value is > 95.
```

1. Start the **TESTENRICH** service by clicking the **Start Service** icon.

The started service will show a green checkbox.



2. Open the **POLICY-Event_Enrichment-Adv** policy window. Click the **Policy Log** icon to view results of the log statements.

The log statements should look similar to the following screen capture.

```
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === Result from HS_Customer search ===>NULL
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === @Department value set from link ===>Sales
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === @Customer value set from link ===>Microsoft
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === @SLA value set from link ===>93%
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === SLAValue from extract ===>93
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === @Severity value calculated ===>4
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === @WorkingHours set from HS_Customer ===>9:00-17:30
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === WHTime value calculated ===>9:00-17:30
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === StartMinute ===>0
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === StartTime value calculated ===>540
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === EndMinute value calculated ===>30
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === CurrentMinute value calculated ===>8
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === CurrentTime value calculated ===>1028
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]Parser log: === @SuppressEscl value set ===>0
PolicyLogger][POLICY-EventEnrichment-Adv-LOGSTMT][MessageProcessor-Dog#8]
```

3. Run the script to verify the updates to the OMNIbus **alerts.status** table using the following commands:

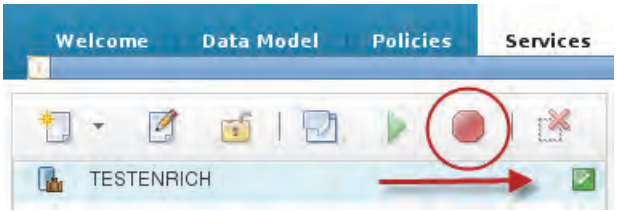
```
cd /opt/IBM/tivoli/netcool/omnibus/bin
./nco_sql -server NCOMS -user root -password object00 -i /labfiles/OMNIbusQuery2
```

The **Node**, **Department, Severity, SLA**, **WorkingHours**, and **Location** entries in OMNIbus should look similar to the following screen capture.

```
File  Edit  View  Terminal  Help
netcool@jazz01:/opt/IBM/tivoli/netcool/omnibus/bin> ./nco_sql -server NCOMS -user root
object00 -i /labfiles/OMNIbusQuery2
 Node
        Department
        Severity
        SLA
        WorkingHours
        Location
-------------------------------------------------------------------
-------------------------------------------------------------------
-----------
-------------------------------------------------------------------
-------------------------------------------------------------------
-------------------------------------------------------------------
 jazz01
        Product Management
                2
        96%
        24 Hours
        Frankfurt Building 3
 jazz01
        Product Management
                0
        96%
        24 Hours
        Frankfurt Building 3

(36 rows affected)
netcool@jazz01:/opt/IBM/tivoli/netcool/omnibus/bin>
```

Note: The number of rows affected varies depending on how long the probe has been running. Also, if there are problems generating data using the restriction filter delete the TESTENRICH service and create TESTENRICH2 with the same parameters.

4. Stop the **TESTENRICH** service.



Unit 7 Controlling policy execution sequence exercises

This unit focuses on event locking and policy chaining.

Exercise 1 Observing chaining and thread locking

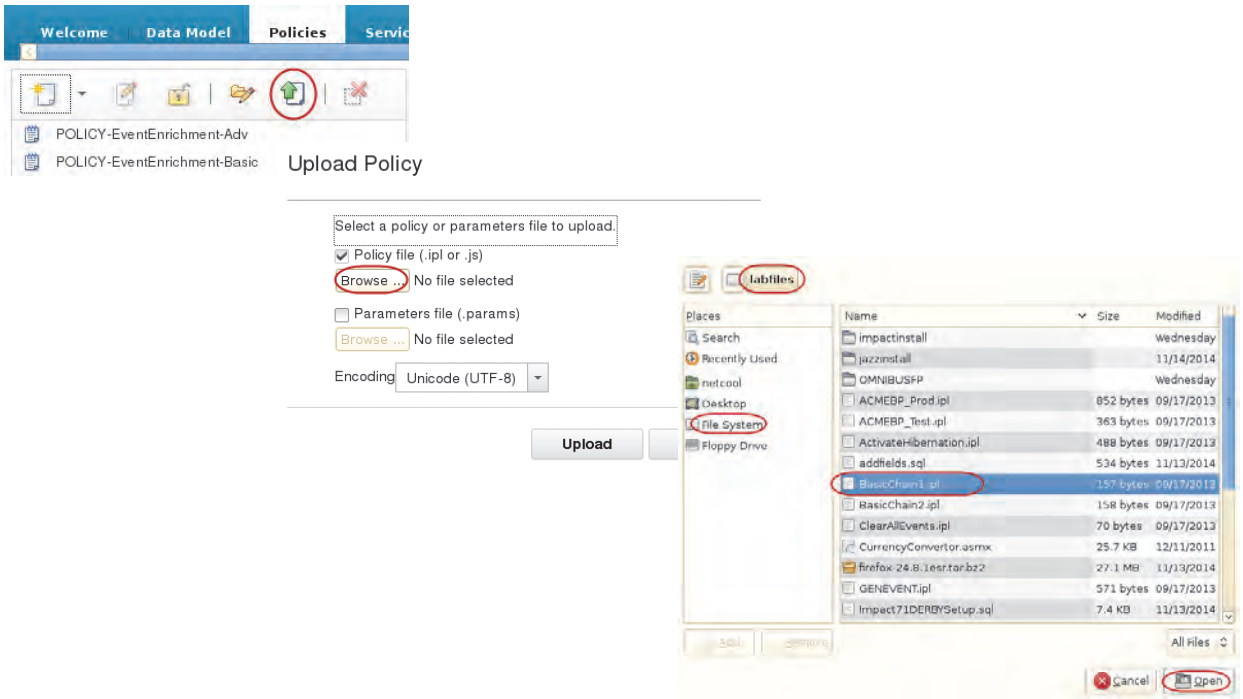
In this exercise, you observe the behavior of policies when implementing thread processing control and policy chaining. You import policies and then configure TESTENRICH for two new event mappings. The first event mapping runs **BasicChain1**. The second event mapping runs **BasicChain2**.

1. Stop the Simnet Probe by issuing the following commands to locate the process ID and stop the process.

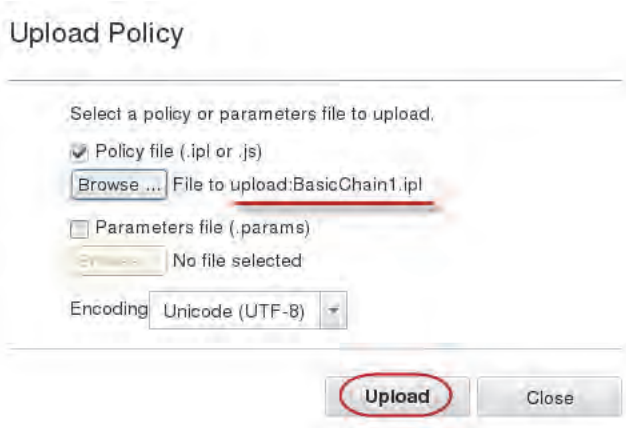
```
ps -ef | grep nco
kill <process id>
```

```
netcool@jazz01:~/Desktop> ps -ef | grep nco
netcool   4256     1  0 Jun20 ?        00:06:15 /opt/IBM/tivoli/netcool/omnibus/
platform/linux2x86/bin64/nco_objserv
netcool   4315     1  1 Jun20 ?        00:38:21 /opt/IBM/tivoli/impact/sdk/jre/b
in/java -javaagent:/opt/IBM/tivoli/impact/wlp/bin/tools/ws-javaagent.jar -Djava.
awt.headless=true -XX:MaxPermSize=256m -Xms512M -Xmx1200M -Dclient.encoding.over
ride=UTF-8 -Dhttps.protocols=SSL_TLSv2 -jar /opt/IBM/tivoli/impact/wlp/bin/tools
/ws-server.jar ImpactUI
netcool   4557     1  4 Jun20 ?        01:33:03 /opt/IBM/tivoli/impact/sdk/jre/b
in/java -javaagent:/opt/IBM/tivoli/impact/wlp/bin/tools/ws-javaagent.jar -Djava.
awt.headless=true -XX:MaxPermSize=256m -Xms512M -Xmx1200M -Dclient.encoding.over
ride=UTF-8 -Dhttps.protocols=SSL_TLSv2 -jar /opt/IBM/tivoli/impact/wlp/bin/tools
/ws-server.jar NCI
netcool  15769 15659  0 05:18 pts/0    00:00:42 /opt/IBM/tivoli/netcool/omnibus/
platform/linux2x86/probes64/nco_p_simnet
netcool  19553 19536  0 18:00 pts/1    00:00:00 grep nco
netcool@jazz01:~/Desktop> kill -9 15769
```

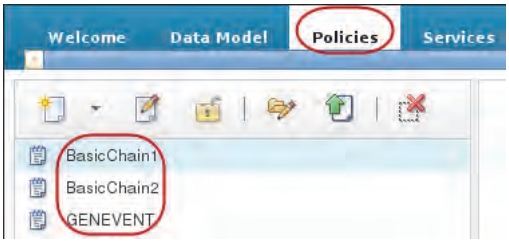
2. Click the **Upload Policy** icon. Click **Browse** to locate **BasicChain1.ipl** in the **labfiles** directory. Click **Open**.



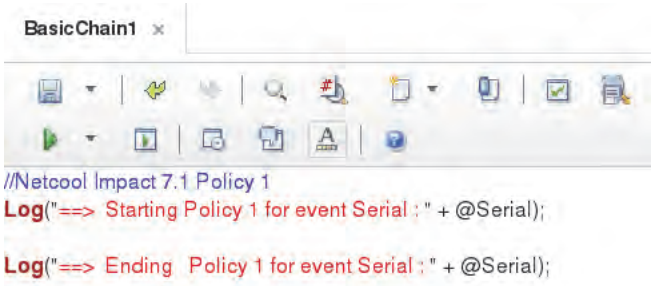
3. Click **Upload** to add the policy to the Polices list.



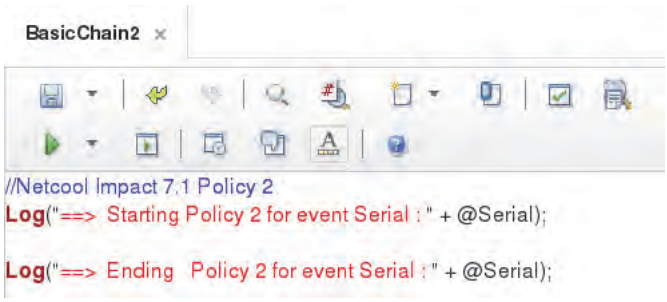
4. Repeat the steps for **BasicChain2.ipl**, and **Genevent.ipl**.
   The policies are added to the Policies list.



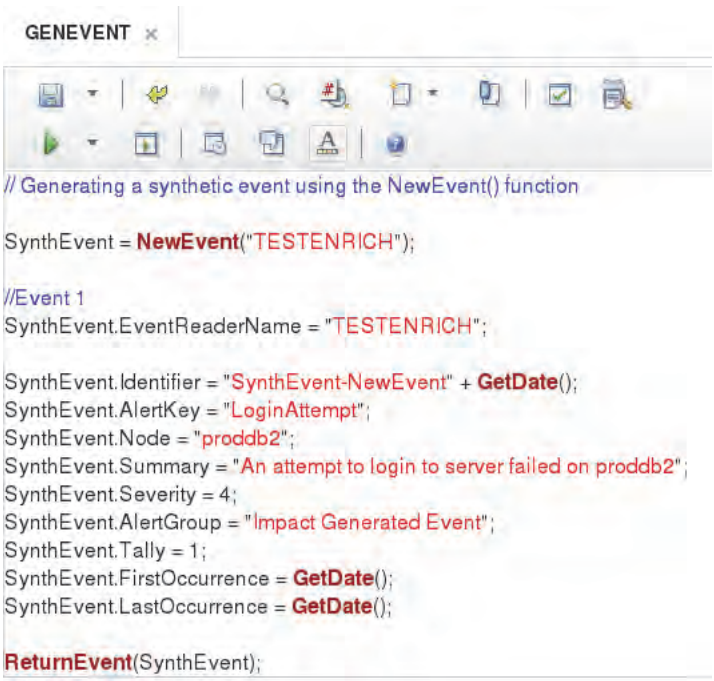
BasicChain1.ipl



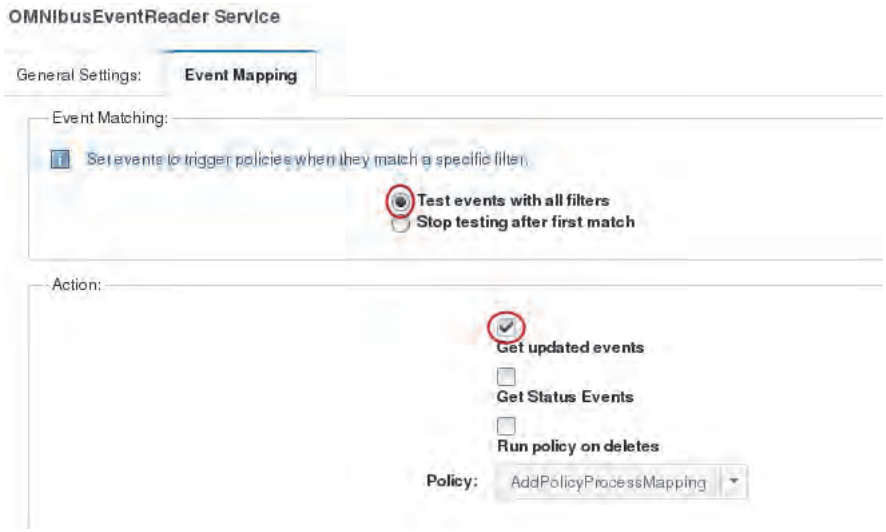
BasicChain2.ipl



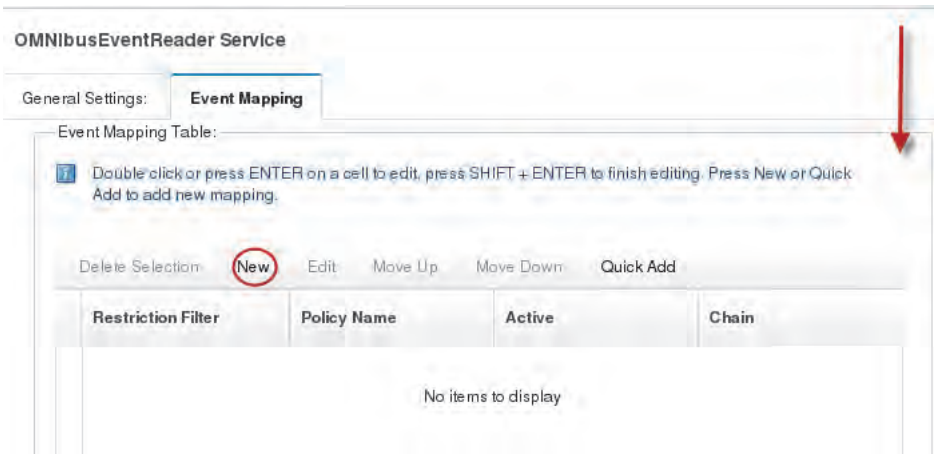
GENEVENT.ipl



5. Modify the TESTENRICH service and create new mappings for the uploaded policies. Delete any existing policies in the mapping table.



6. Scroll down to the **Event Mapping Table** and click the **New** button.



Note: Be sure that **Event Locking** is not selected in the Event Locking section.

The **Edit Event Filter** window opens.

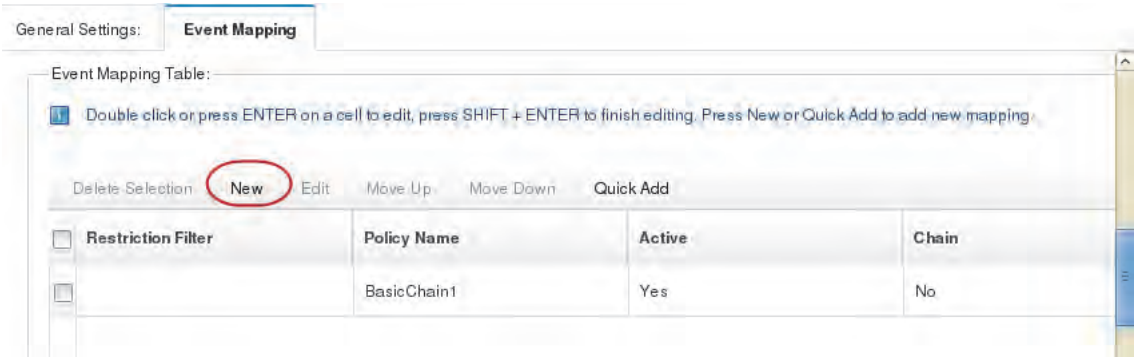
a. Leave the **Filter Expression** field blank.

b. From the **Policy to Run** list, select **BasicChain1**.

c. Check the **Active** box.

d. Click **OK**.

7. Click the **New** button.



The **Edit Event Filter** window opens.

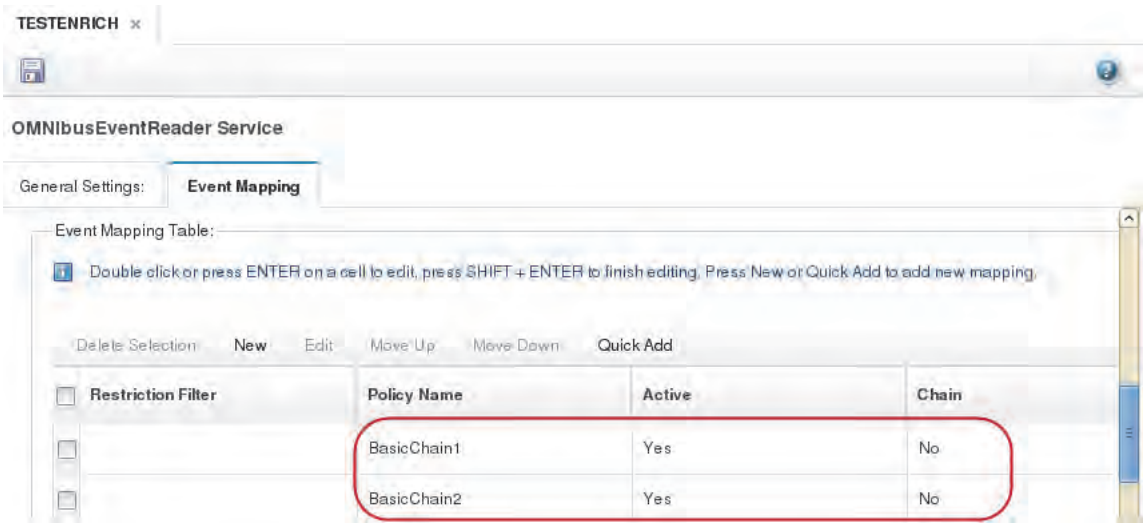
a. Leave the **Filter Expression** field blank.

b. From the **Policy to Run** list, select **BasicChain2**.

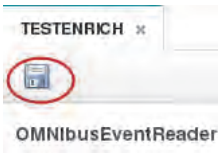
c. Check the **Active** box.

d. Click **OK**.

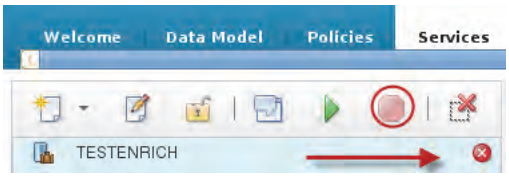
The resulting mappings are listed as shown in the following example.



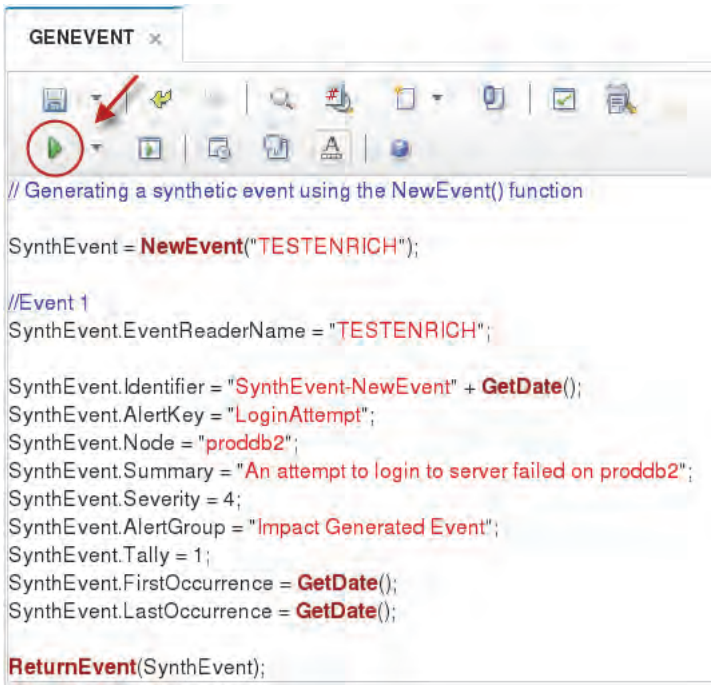
8. Save the changes. Click the **Save** icon.



9. Stop the **TESTENRICH** service.



10. Select the **GenEvent** policy in the **Policies** list. Click the green arrow to execute the policy.



11. Click **Close**.

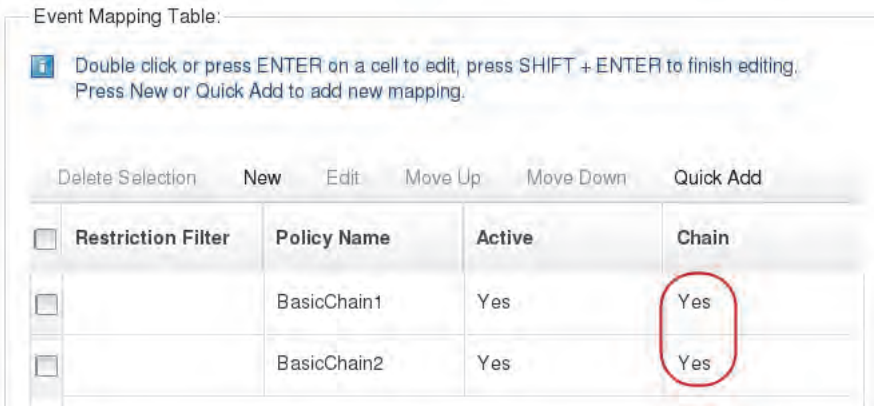
12. Start the **TESTENRICH** service.

13. View the output in the policy logger For the GENEVENT policy.

Notice how Policy 2 starts before Policy 1 is finished with the event.

```
Parser log: ==>  Starting Policy 1 for event Serial : 16
Parser log: ==>  Starting Policy 1 for event Serial : 182
Parser log: ==>  Starting Policy 2 for event Serial : 182
Parser log: ==>  Ending   Policy 1 for event Serial : 182
Parser log: ==>  Ending   Policy 1 for event Serial : 16
Parser log: ==>  Ending   Policy 2 for event Serial : 182
```

14. Modify the **TESTENRICH** service and select the **Chain** check box next to each policy as shown in the example:



15. Save the changes. Stop the **TESTENRICH** service. Start the service. Clear the policy log. Execute the **GenEvent** policy in the policy editor. View the policy log.

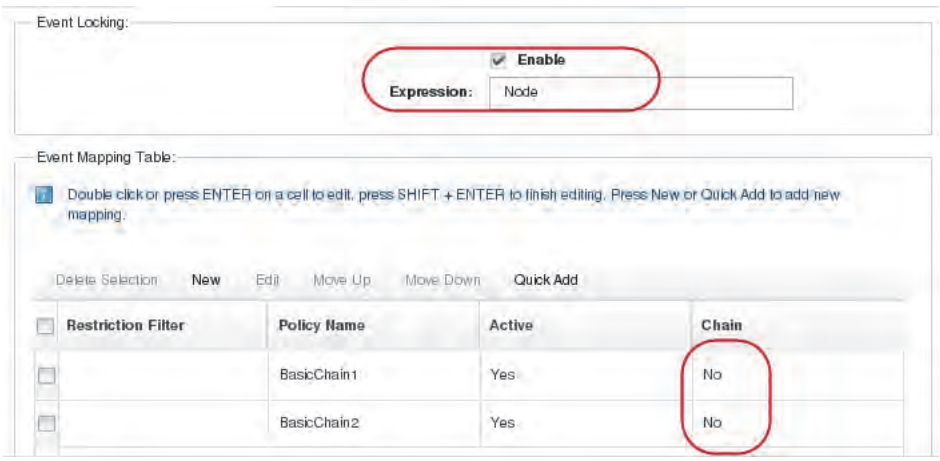
```
|Parser log: ==>  Starting Policy 1 for event Serial : 430
|Parser log: ==>  Ending   Policy 1 for event Serial : 430
|Parser log: ==>  Starting Policy 2 for event Serial : 3
|Parser log: ==>  Ending   Policy 2 for event Serial : 3
|Parser log: ==>  Starting Policy 2 for event Serial : 430
|Parser log: ==>  Ending   Policy 2 for event Serial : 430
```

This time Policy 1 ends before Policy 2 starts for the same serial.

Exercise 2 Configuring thread locking

Using event locking, the multi-threaded Event Reader sorts incoming events based on alert fields. It then processes them one category at a time in the order sent by the Event Reader.

1. Configure event locking on the **TESTENRICH Event Mapping Configuration** window for **BasicChain1** and **BasicChain2**. Select the **Enable** check box and, for this lab, type **Node** in the **Expression** field.



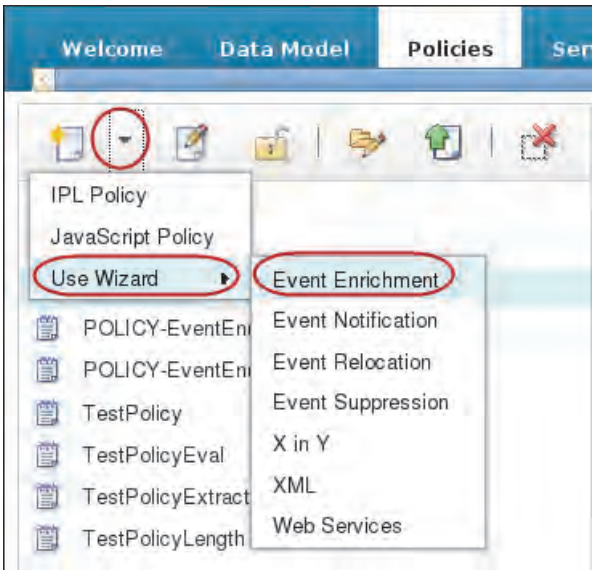
2. Restart the **TESTENRICH** service. Execute the **GenEvent** policy two or three times. View the policy log.

Unit 8 Policy wizards exercises

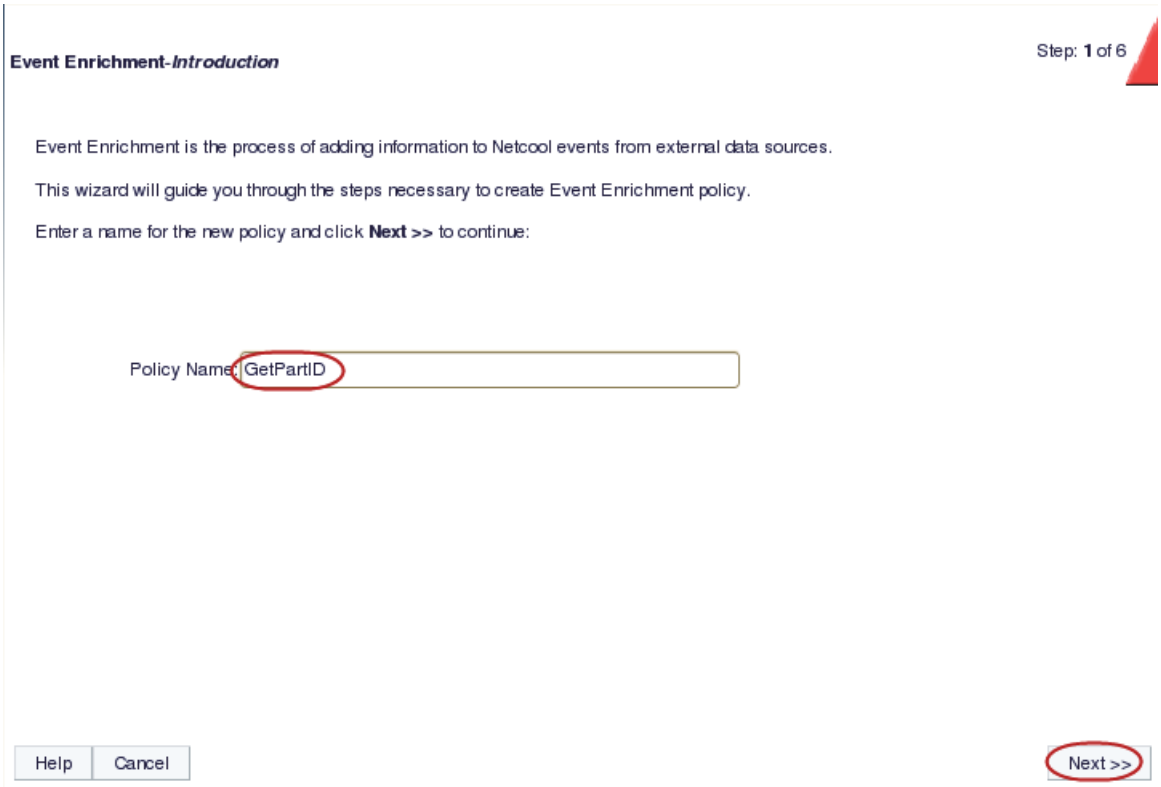
In this unit, you work with the policy wizard.

Exercise 1 Creating an enrichment policy using the wizard

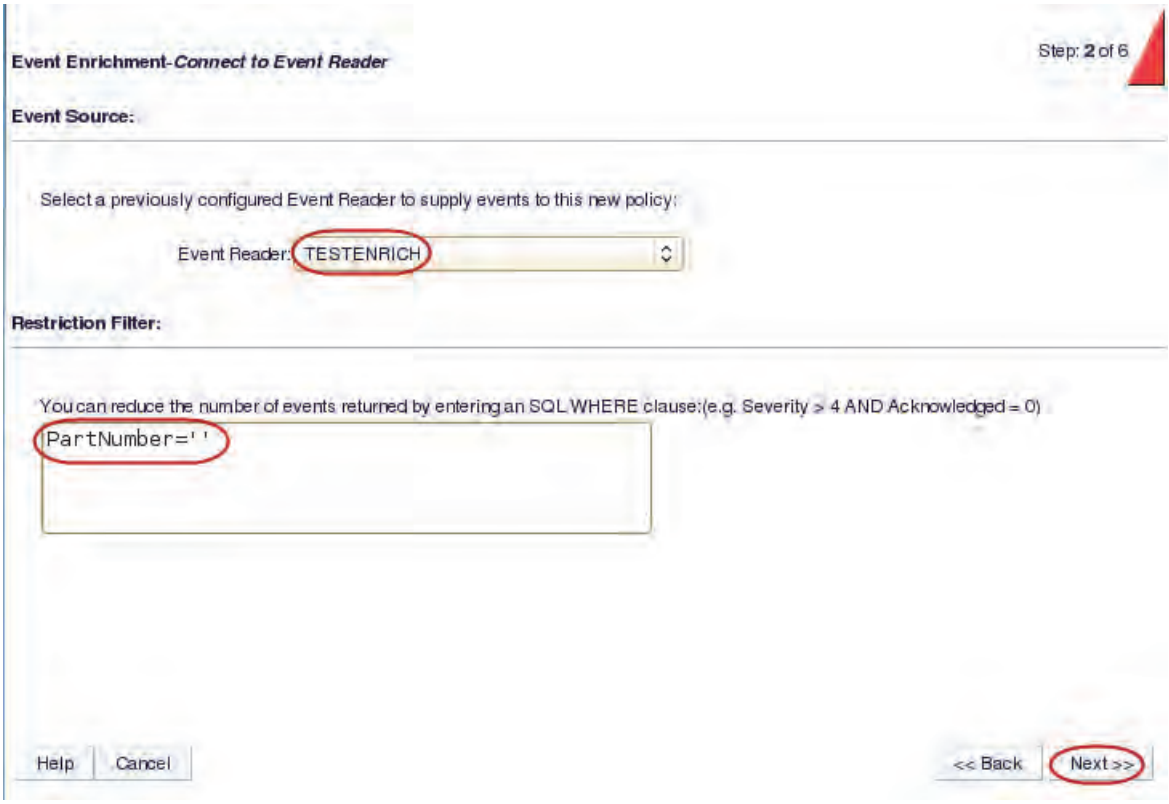
1. Click **Policies** to open the **Policies** tab.
2. Click the **New Policy** icon. Select **Use Wizard > Event Enrichment** in the list.



3. Type **GetPartID** in the **Policy Name** field. Click **Next**.

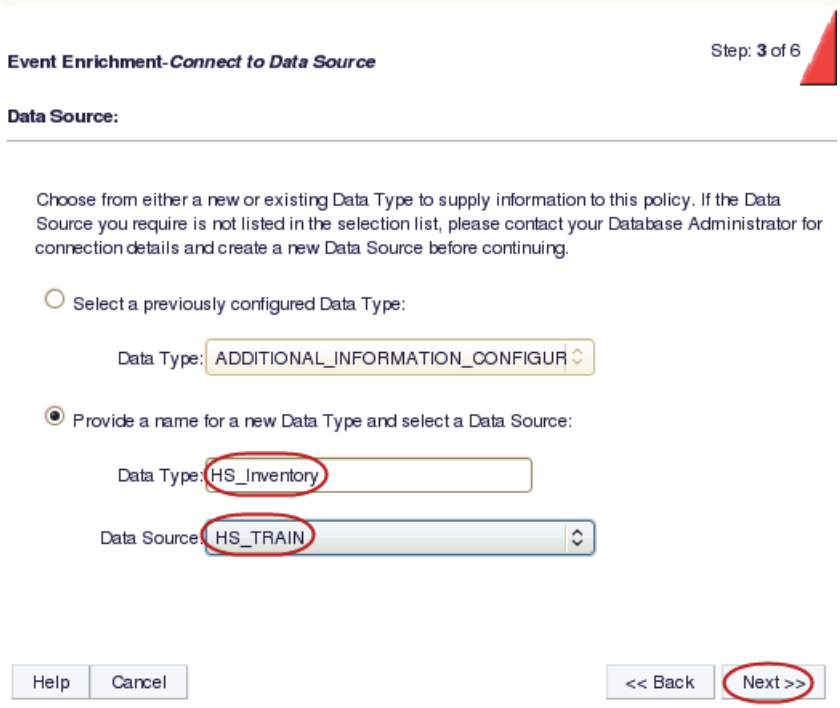


4. Select **TESTENRICH** as the Event Reader. Type **PartNumber= ''** (two single quotation marks) in the **Restriction Filter** field. The restriction clause is set against fields in the Event Reader. Click **Next**.



5. Create a new data type by performing the following steps:
   a. Type **HS_Inventory** in the **Data Type** field.
   b. Select **HS_TRAIN** from the **Data Source** list.

c. Click **Next**.



6. In the **Data Source Base Table** section, select **IMPACT** and **INVENTORY** in the **Base Table** lists to specify the table name that is being accessed from the **HS_TRAIN** source.

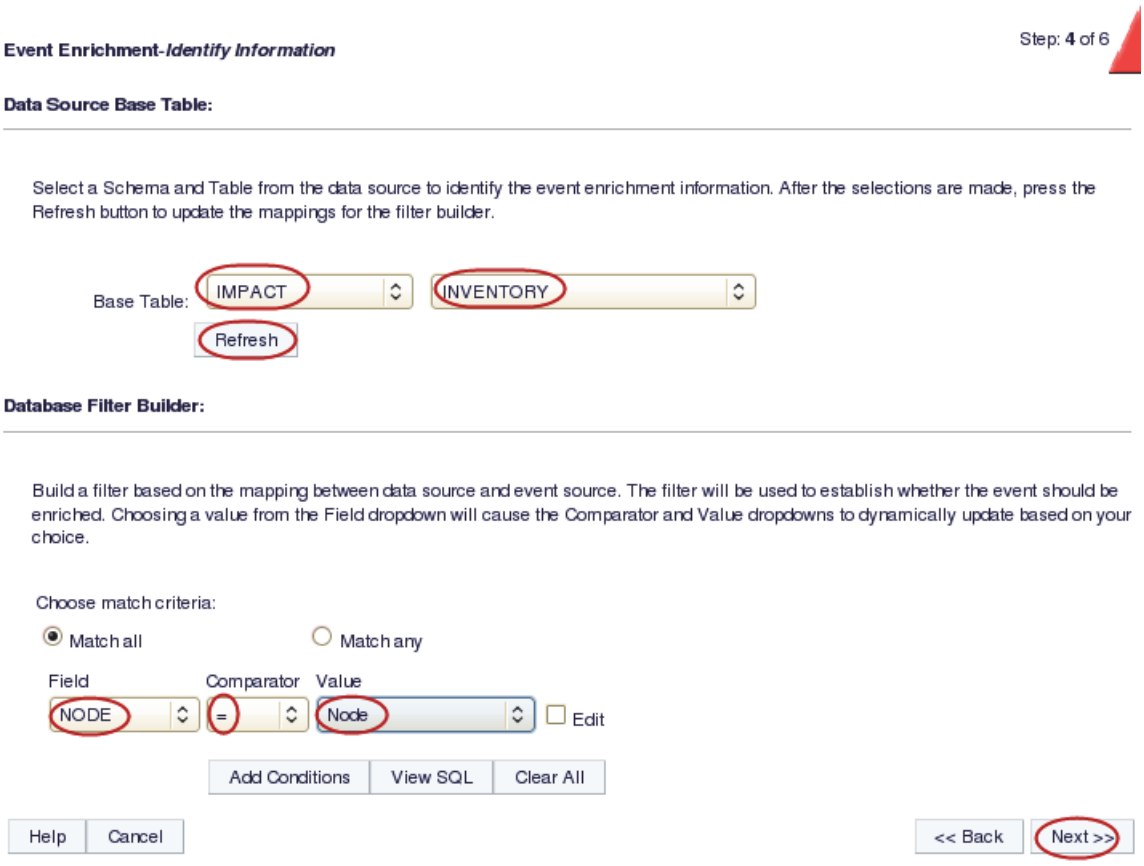
   a. Click **Refresh**.

   b. In the **Database Filter Builder** section, specify the following values for the database query.

Field	Value
Field	NODE
Comparator	=
Value	Node

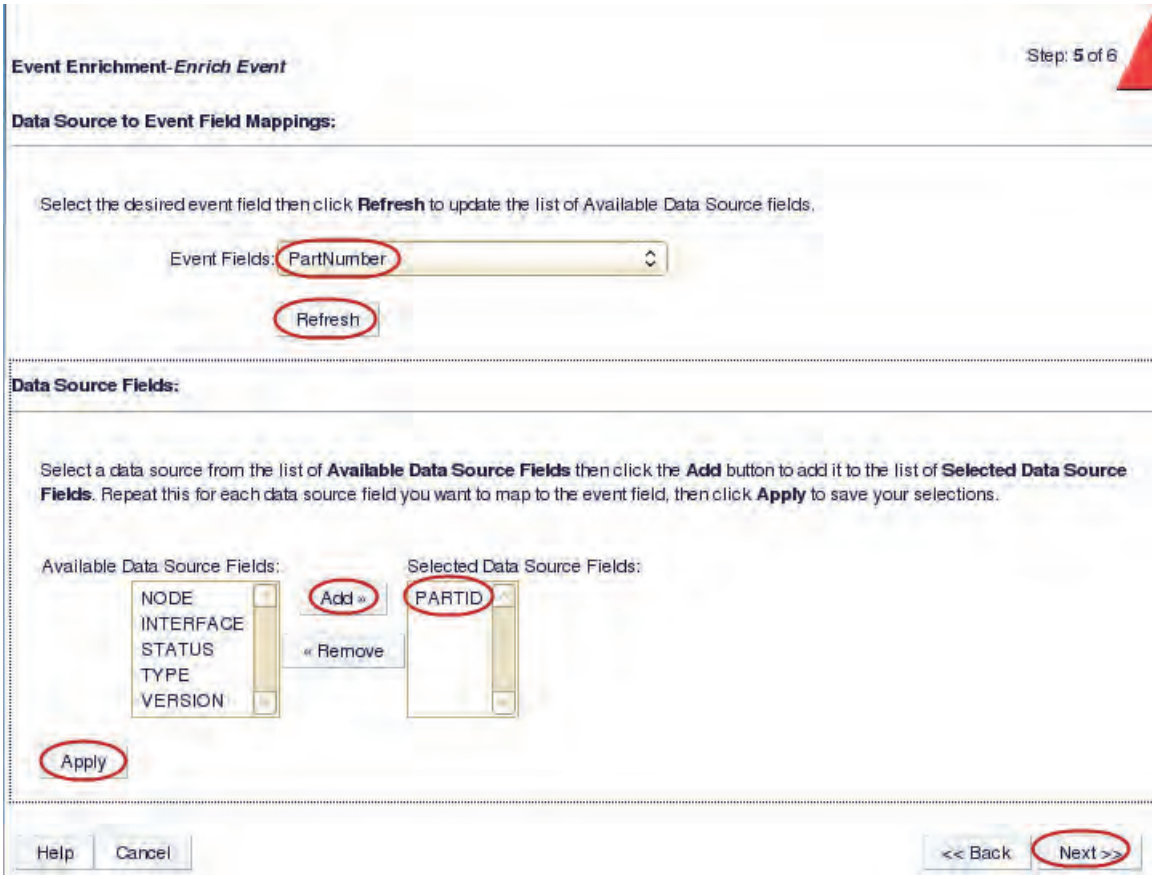
   c. Click **Next**.

These settings specify the field in the table that is matched against the field in the ObjectServer. In this case, **NODE** from the HS_Inventory table matches against **Node** from the ObjectServer.



If no fields are displayed in the list, most likely the ObjectServer source used for the Event Reader is not available, or it is configured improperly.

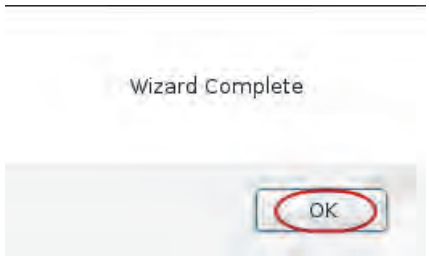
7. Select **PartNumber** from the **Event Fields** list. Click **Refresh**. Select **PARTID** in the **Available Data Source Fields** list, and click **Add**. PARTID moves to the **Selected Data Source Fields** list. Click **Apply.** Click **Next.**



Note: PARTID is from the inventory table.

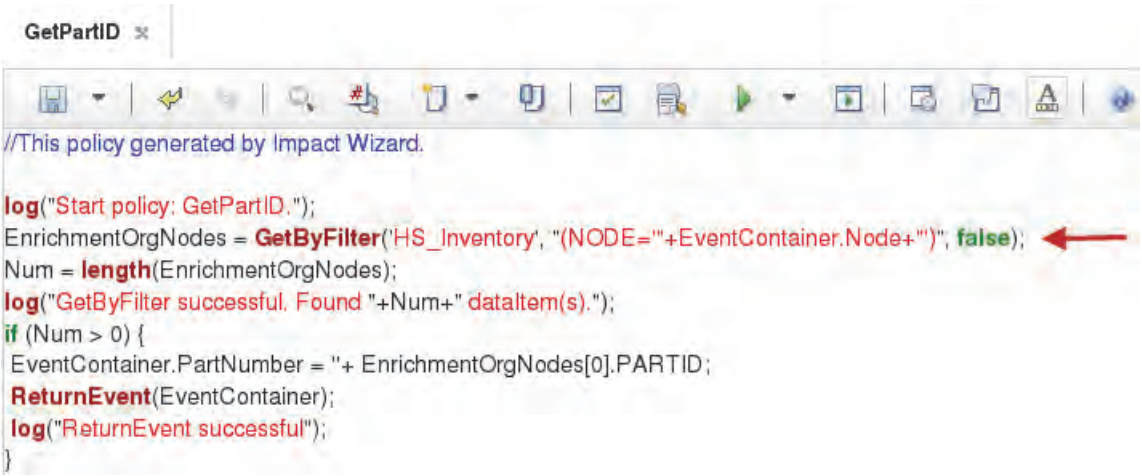
8. When the Summary and Finish window opens, click **Apply**. Click **Finish** to create the policy.

9. Click **OK** to close the window.

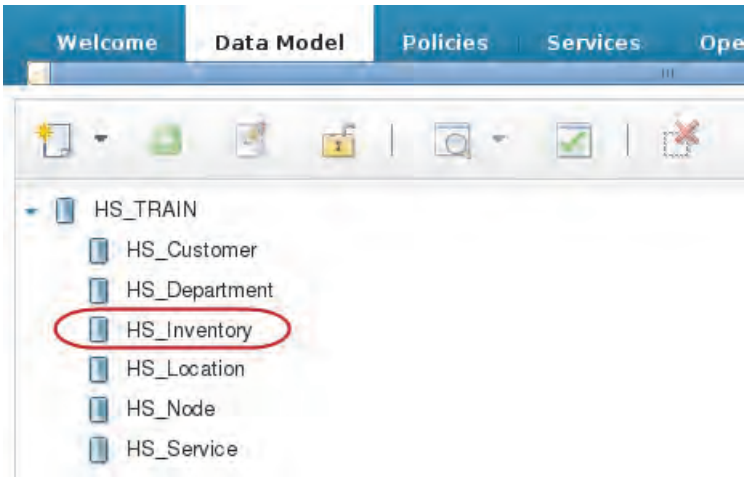


The new **GetPartID** policy opens in the policy window.

10. Click the **Save** icon.

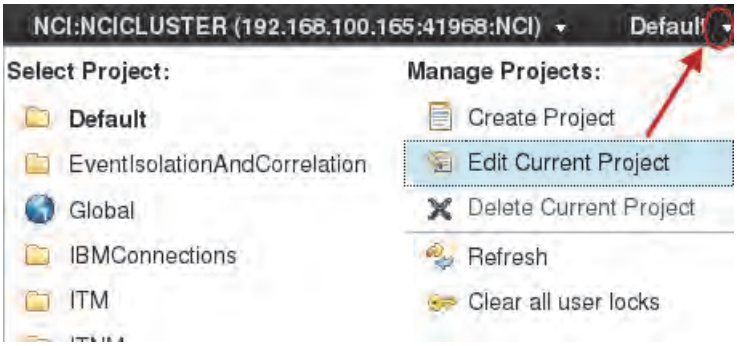


11. Click **Data Model** and expand the Data Source **HS_TRAIN.**

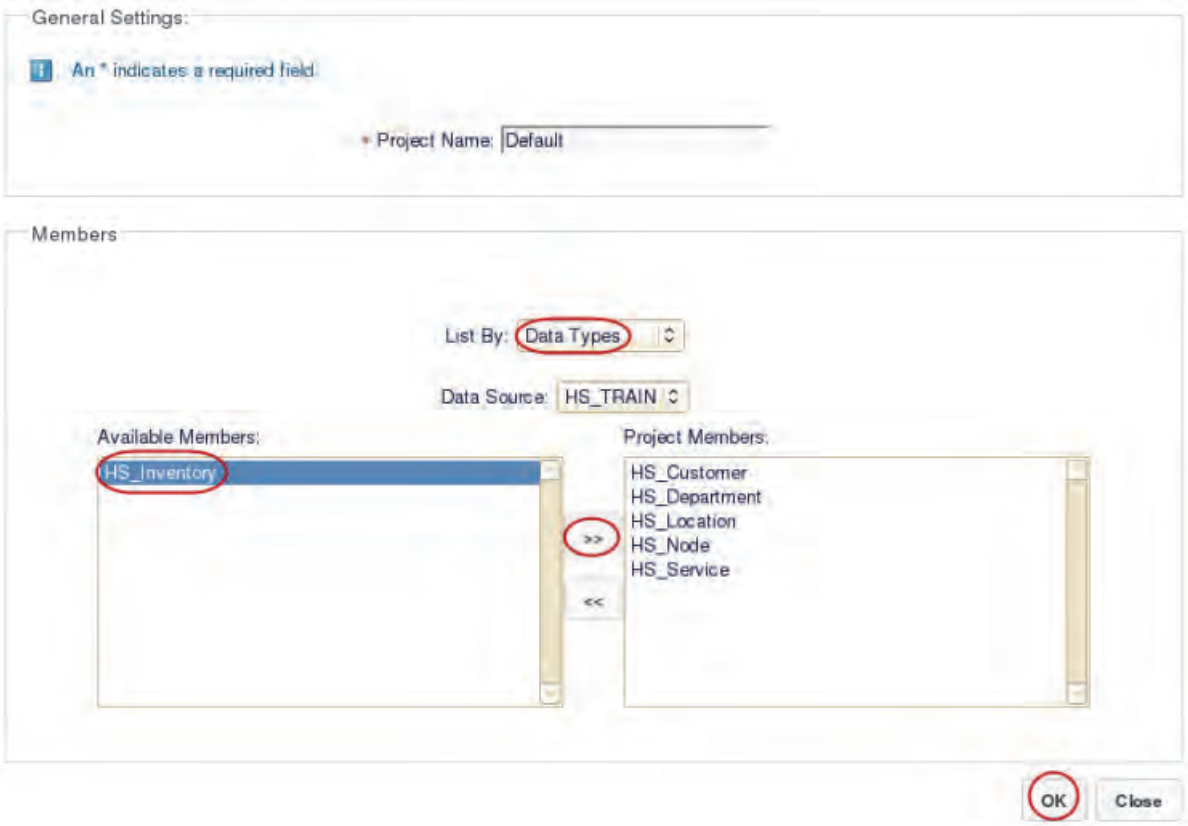


If **HS_Inventory** is not listed, add it from the Global Repository.

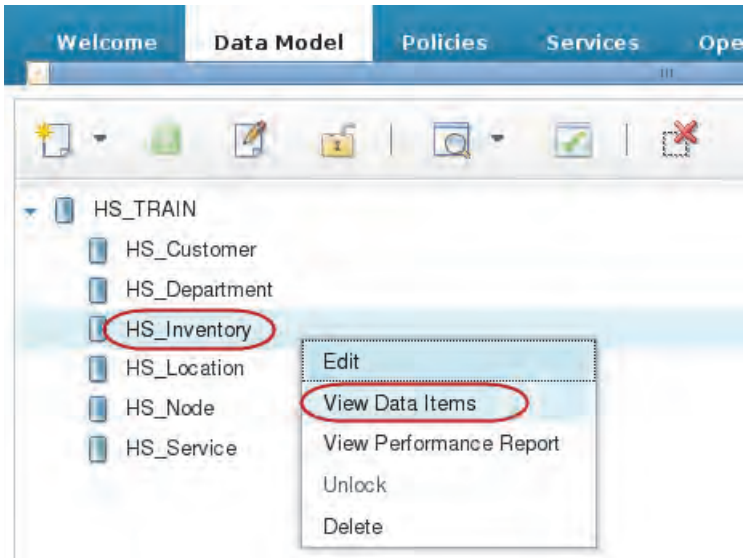
i. Edit the Default project.



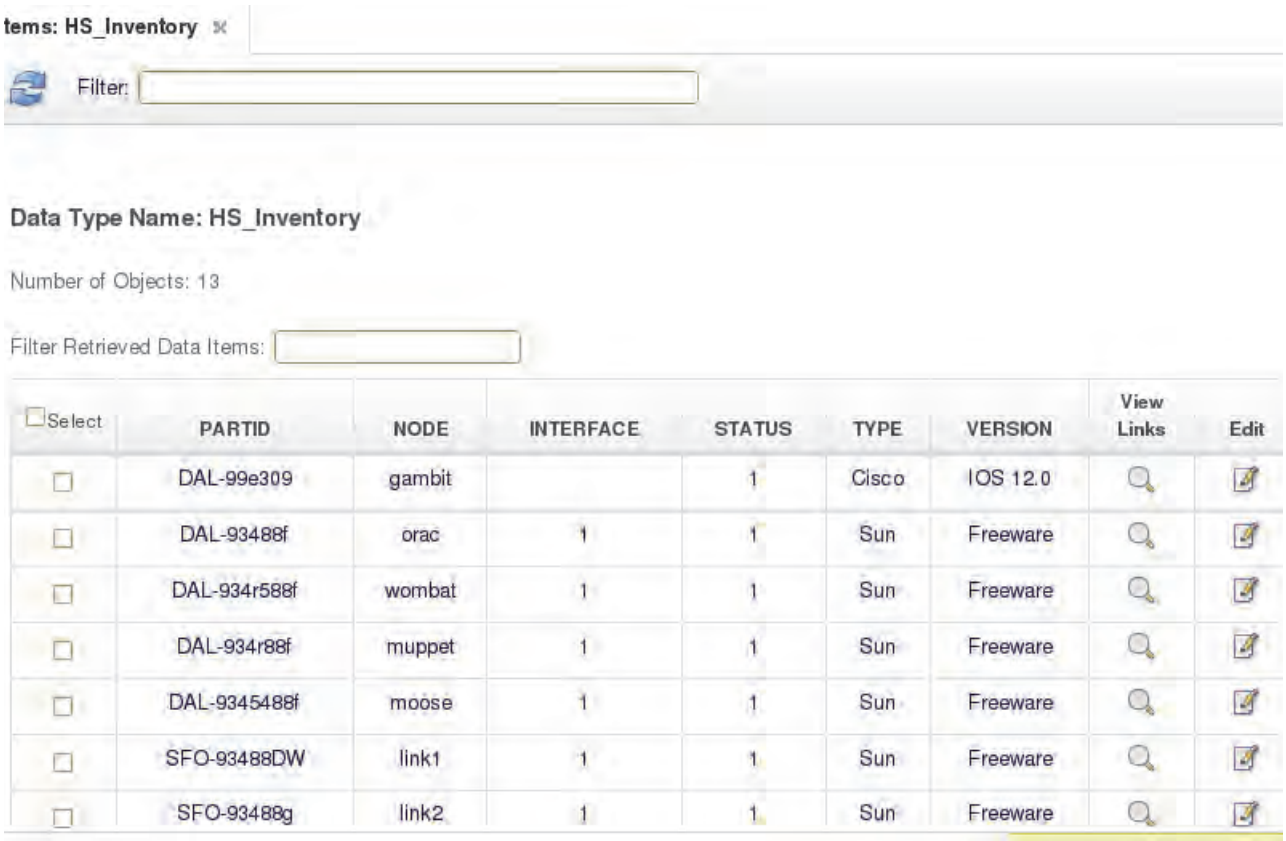
ii. List By **Data Types**. Select **HS_Inventory** and add it to the **Default** project. Click **OK**.



12. Right-click **HS_Inventory** and select **Edit** to view the table description. Right-click **HS_Inventory** and select **View Data Items** to see field values.



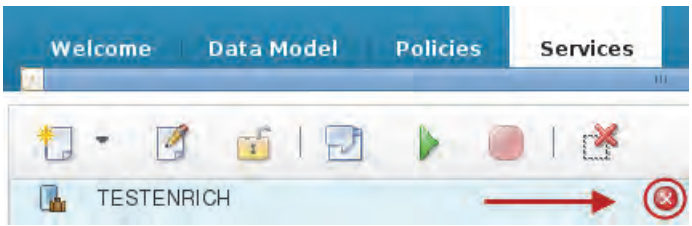
The table should look similar to the following screen capture.



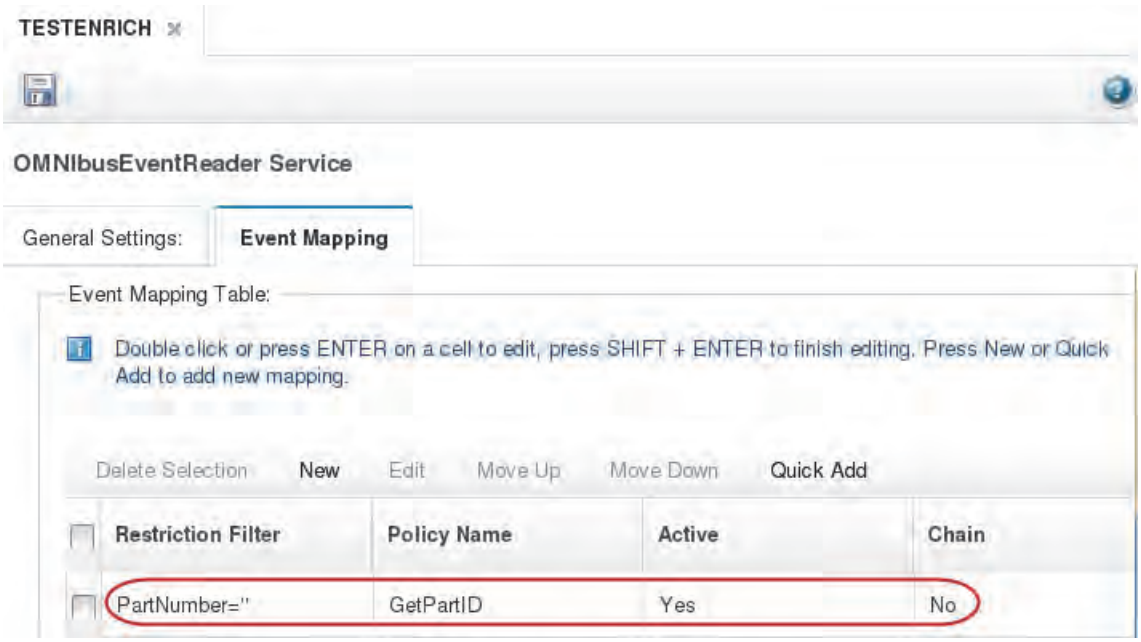
Select	PARTID	NODE	INTERFACE	STATUS	TYPE	VERSION	View Links	Edit
☐	DAL-99e309	gambit		1	Cisco	IOS 12.0		
☐	DAL-93488f	orac	1	1	Sun	Freeware		
☐	DAL-934r588f	wombat	1	1	Sun	Freeware		
☐	DAL-934r88f	muppet	1	1	Sun	Freeware		
☐	DAL-9345488f	moose	1	1	Sun	Freeware		
☐	SFO-93488DW	link1	1	1	Sun	Freeware		
☐	SFO-93488g	link2	1	1	Sun	Freeware		

Notice the values for **PARTID** and **NODE**.

13. Verify that the **TESTENRICH** service is stopped.



14. Edit the service and make sure that there is only one policy **GetPartID**, defined to the service. Save any changes.



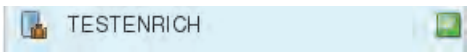
15. Double-click the **Gnome Terminal** icon on the desktop.

16. Start the Simnet Probe using the following commands:

```
cd /opt/IBM/tivoli/netcool/omnibus/probes/
./nco_p_simnet &
```

Note: The TESTENRICH service was configured by the wizard to include the GetPartID policy and the policy filter.

17. Start the TESTENRICH service.



18. View the policy log.

```
[PolicyLogger][GetPartID][MessageProcessor-Dog#1]Parser log: GetByFilter successful. Found 1 dataItem(s).
[PolicyLogger][GetPartID][MessageProcessor-Dog#1]Parser log: ReturnEvent successful
[PolicyLogger][GetPartID][MessageProcessor-Dog#1]Parser log: Start policy: GetPartID.
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: GetByFilter successful. Found 0 dataItem(s).
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: Start policy: GetPartID.
[PolicyLogger][GetPartID][MessageProcessor-Dog#1]Parser log: GetByFilter successful. Found 0 dataItem(s).
[PolicyLogger][GetPartID][MessageProcessor-Dog#1]Parser log: Start policy: GetPartID.
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: GetByFilter successful. Found 0 dataItem(s).
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: Start policy: GetPartID.
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: GetByFilter successful. Found 0 dataItem(s).
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: Start policy: GetPartID.
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: GetByFilter successful. Found 0 dataItem(s).
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: Start policy: GetPartID.
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: GetByFilter successful. Found 0 dataItem(s).
[PolicyLogger][GetPartID][MessageProcessor-Dog#2]Parser log: Start policy: GetPartID.
[PolicyLogger][GetPartID][MessageProcessor-Dog#1]Parser log: GetByFilter successful. Found 1 dataItem(s).
[PolicyLogger][GetPartID][MessageProcessor-Dog#1]Parser log: ReturnEvent successful
```

19. Type the following commands to verify **Node** and **PartNumber** updates in OMNIbus.

```
cd /opt/IBM/tivoli/netcool/omnibus/bin/
./nco_sql -server NCOMS -user root -password object00 -i
/labfiles/OMNIbusQuery3
```

```
netcool@jazz01:/opt/IBM/tivoli/netcool/omnibus/bin> ./nco_sql -server NCOMS -use
r root -password object00 -i /labfiles/OMNIbusQuery3
 Node
         PartNumber
 ----------------------------------------------------------------
         ----------------------------------------------------------------
 link3
         LON-93488ef
 link2
         SFO-93488g
 link3
         LON-93488ef
 link2
         SFO-93488g
 link1
         SFO-93488DW
 link5
         LGA-393488f
 link4
         LON-943488f
 link5
         LGA-393488f
 link4
         LON-943488f
 link1
         SFO-93488DW

(10 rows affected)
netcool@jazz01:/opt/IBM/tivoli/netcool/omnibus/bin>
```

Unit 9 Notification policies exercises

In this unit, you configure a notification policy.

Exercise 1 Notification

This exercise uses the Netcool/Impact email service to send a notification. Validate the send and receive email function on your machine using the **mail** command as follows:

1. To log in as the **root** user, type **su - root** from the command prompt.
2. Type **object00** when prompted for the password.
3. At the command prompt, type the following text:
   mail netcool@jazz01.tivoli.edu
4. When prompted for a subject, type the following text:
   Network Status
5. Press Enter. Type the following text:
   The server in the security room is down.
6. To send the message, type a period (**.**) on a line by itself and press the Enter key. Type **exit** to switch back to the **netcool** user.

```
Directory: /home/netcool/Desktop
Tue Jun 21 21:11:07 UTC 2016
netcool@jazz01:~/Desktop> su - root
Password:
jazz01:~ # mail netcool@jazz01.tivoli.edu
Subject:  Network Status
The server in the security room is down.
.
EOT
jazz01:~ # exit
logout
You have mail in /var/spool/mail/netcool
netcool@jazz01:~/Desktop> mail
Heirloom mailx version 12.2 01/07/07.  Type ? for help.
"/var/spool/mail/netcool": 1 message 1 new
>N  1 root@jazz01.tivoli Tue Jun 21 21:17   18/647   Network Status
?
```

7. To view messages, at the command prompt type **mail.** Hit the enter key again to display the details.

```
?
Message  1:
From root@jazz01.tivoli.edu  Tue Jun 21 21:17:56 2016
X-Original-To: netcool@jazz01.tivoli.edu
Delivered-To: netcool@jazz01.tivoli.edu
Date: Tue, 21 Jun 2016 21:17:56 +0000
To: netcool@jazz01.tivoli.edu
Subject:  Network Status
User-Agent: Heirloom mailx 12.2 01/07/07
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
From: root@jazz01.tivoli.edu (root)

The server in the security room is down.

?
```

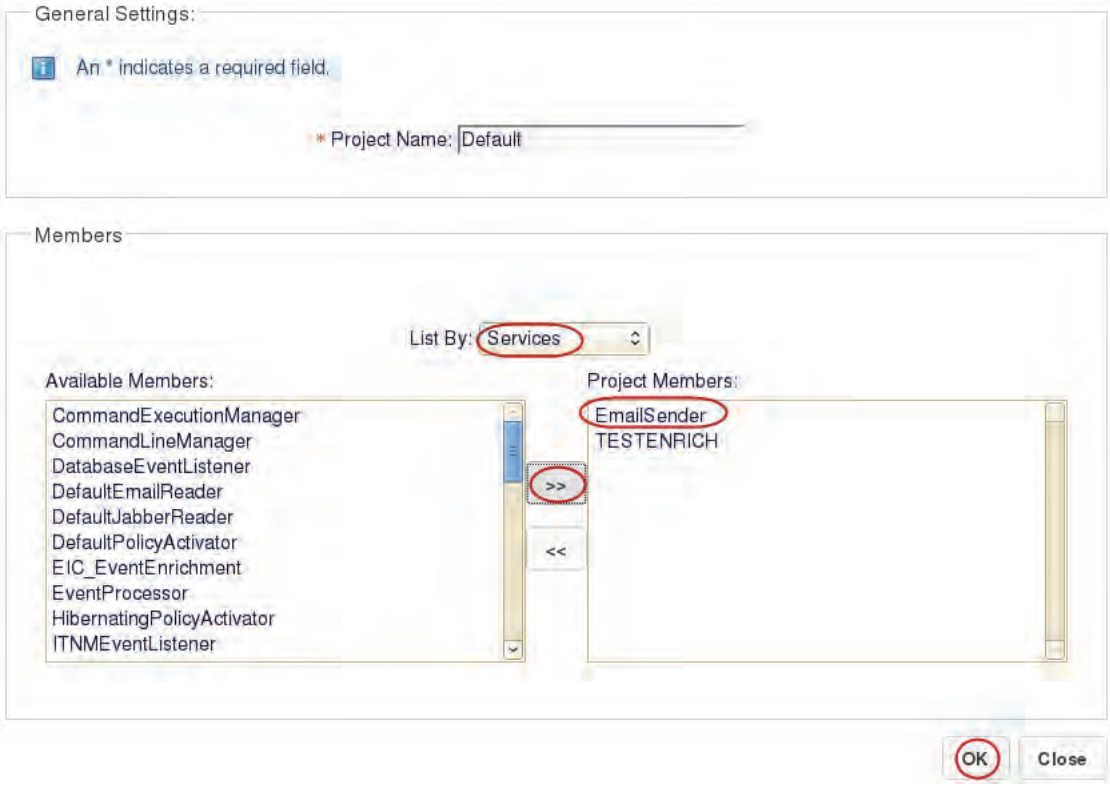
8. Repeat the steps in this exercise as the **netcool** user. Type a response to the original email as shown in the example slide.

   Repeating the steps verifies that the netcool user **netcool** can send mail to the **root** user at **root@jazz01.tivoli.edu**.

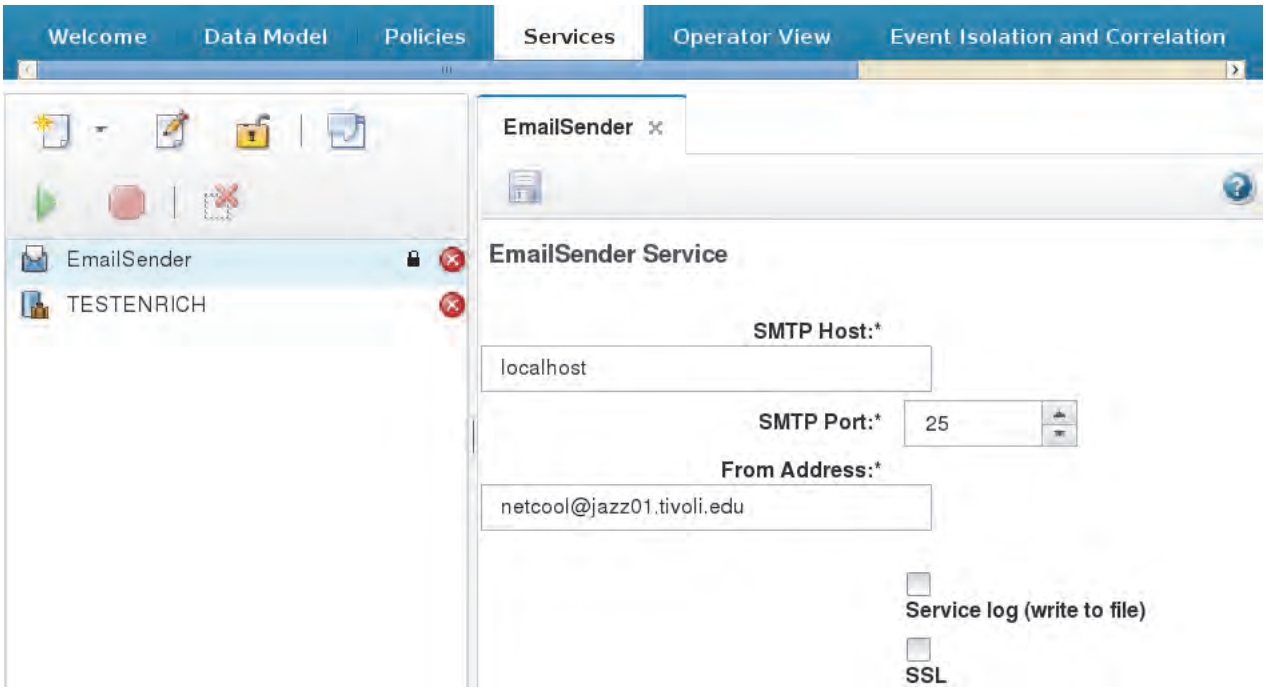
```
netcool@jazz01:~/Desktop> mail root@jazz01.tivoli.edu
Subject: re:Network Status
We are working on the problem.
.
EOT
You have new mail in /var/spool/mail/netcool
netcool@jazz01:~/Desktop> su - root
Password:
jazz01:- # mail
Heirloom mailx version 12.2 01/07/07.  Type ? for help.
"/var/spool/mail/root": 1 message 1 new
>N  1 netcool@jazz01.tiv Tue Jun 21 21:35   18/650   re:Network Status
?
Message  1:
From netcool@jazz01.tivoli.edu  Tue Jun 21 21:35:53 2016
X-Original-To: root@jazz01.tivoli.edu
Delivered-To: root@jazz01.tivoli.edu
Date: Tue, 21 Jun 2016 21:35:53 +0000
To: root@jazz01.tivoli.edu
Subject: re:Network Status
User-Agent: Heirloom mailx 12.2 01/07/07
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
From: netcool@jazz01.tivoli.edu (Netcool User)

We are working on the problem.

?
```

Exercise 2 Configuring the EmailSender service

1. Add the **EmailSender** service to the **Default** project.



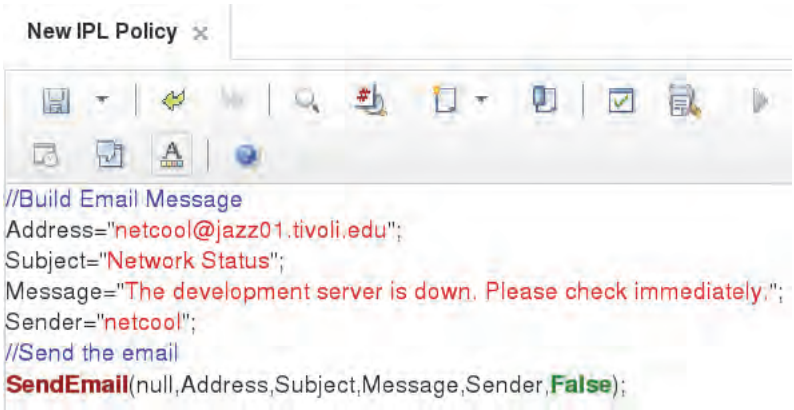
2. Edit the **EmailSender** service. Set the fields as shown in the following screen capture. Click the **Save** icon.



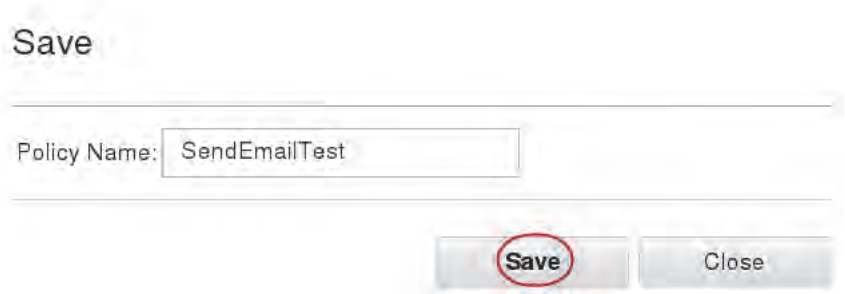
Exercise 3 Creating the policy

1. Create a new IPL policy and add the lines as displayed.

```
//Build Email Message
Address="netcool@jazz01.tivoli.edu";
Subject="Network Status";
Message="The development server is down. Please check immediately.";
Sender="netcool";
//Send the email
SendEmail(null,Address,Subject,Message,Sender,False);
```



2. Click the **Save** icon. Type **SendEmailTest** as Policy Name.



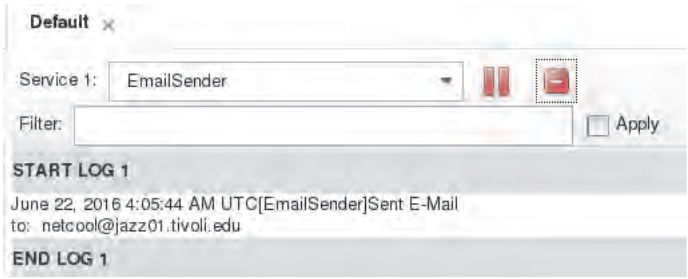
3. Check the syntax icon. Click **Close**.

4. Click the **Run** icon to test the policy. Change the **Subject** line in the policy and run the policy a few more times.
5. Click **OK** to continue each time.

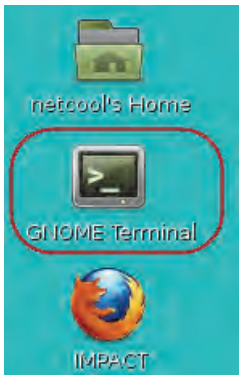
Exercise 4 Verifying SendEmailTest results

1. Click the **View Service Log** icon.

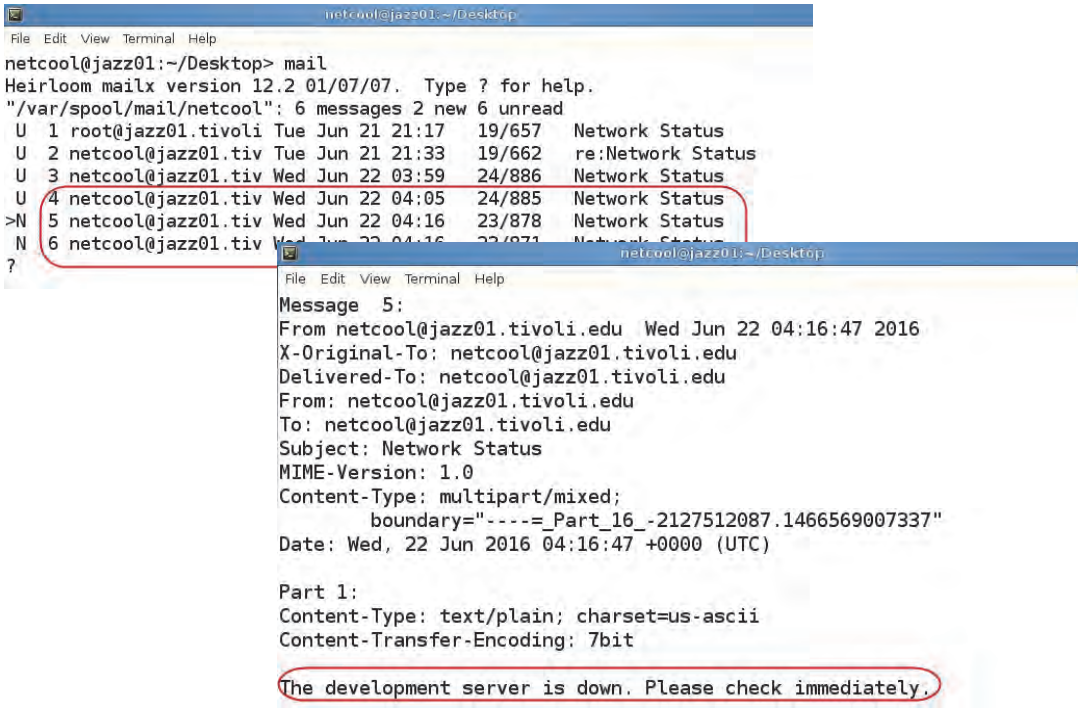
2. Verify that the **EmailSender** service log contains a **SendEmailTest** message.
   a. On the **Services** tab, click the **EmailSender** service.
   b. Click the **View Service Log** icon.



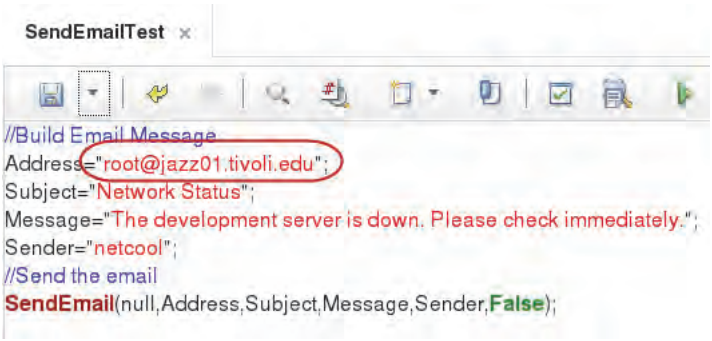
3. Open a GNOME Terminal.



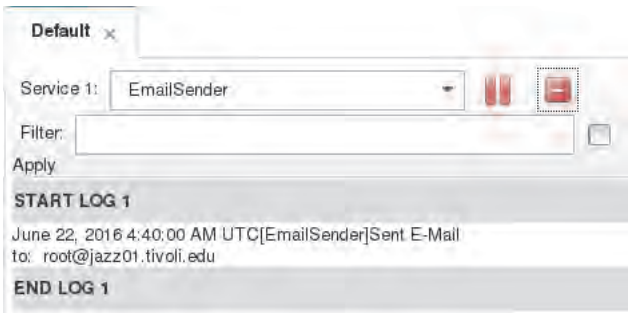
4. Type **mail** command in a terminal window to view the new messages for the **netcool** user. In this example, there are six messages. Press the **Enter key** to see new messages.



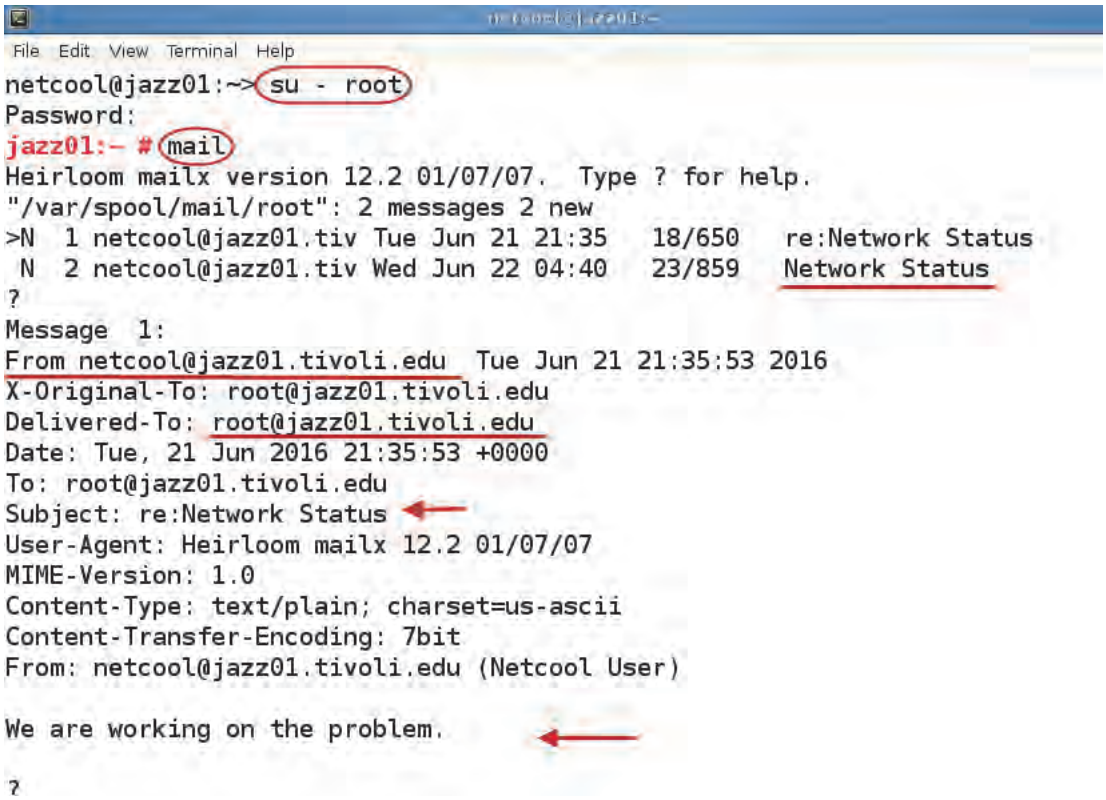
5. Modify the policy to send an email to the **root** user.



6. View the **EmailSender** log results.



7. Open a terminal session and change to the **root** user. Use the **Mail** command to view the email messages. The results will be similar to what is displayed below.



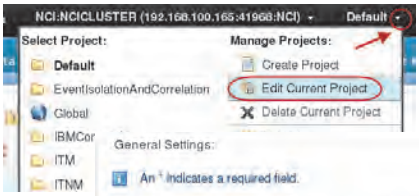
Unit 10 Reports exercises

In this unit, you set up report collection, run several reports, and display the configuration documenter.

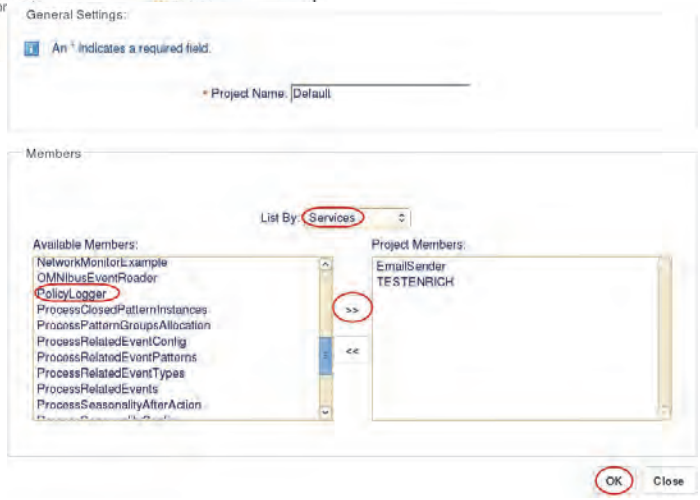
Exercise 1 Enabling report collection

In this lab exercise, you use the Netcool/Impact preinstalled reporting tools to generate reports for policies created in class. Reporting data is generated during policy execution if the **Collect Reports** feature is enabled.

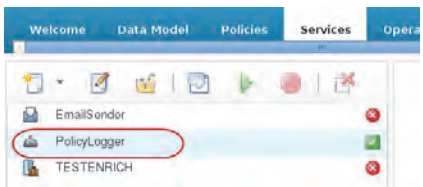
1. Click **Services** to open the **Services** tab.
2. If the **PolicyLogger** is not listed in the **Services** list, add the **PolicyLogger** service to the **Default** project as shown.



Click the **Default** project and **Edit Current Project**

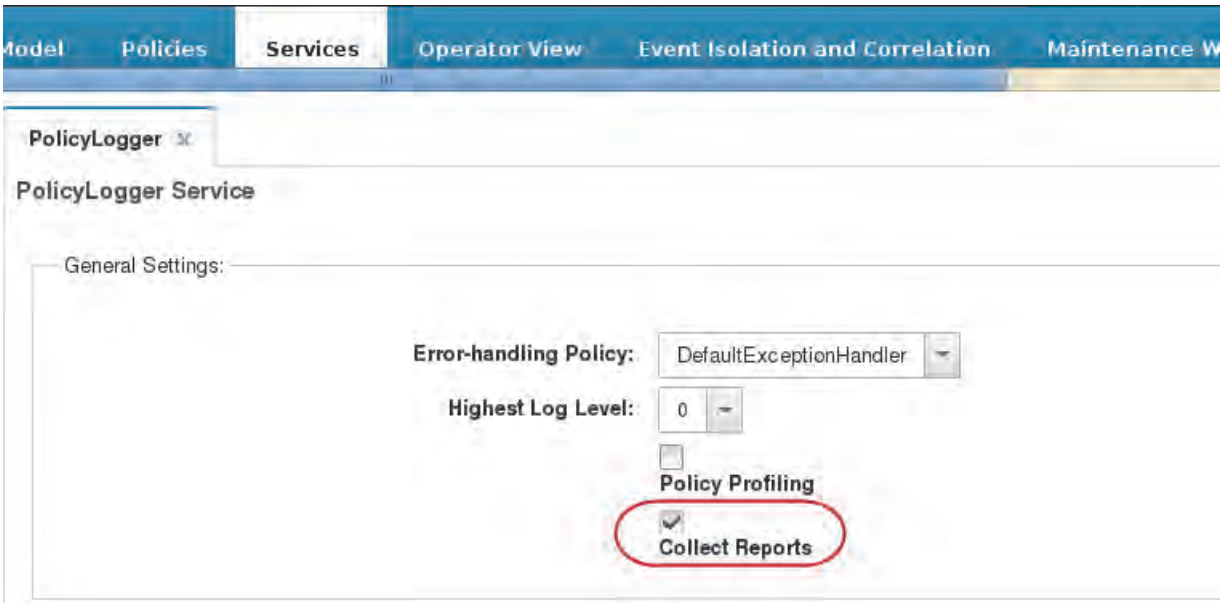


List by **Services** and add the **Policy Logger** Service to the **Project Members** list

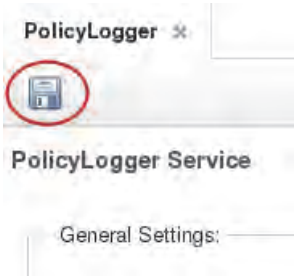


The **Policy Logger** Service is added to the **Services** list

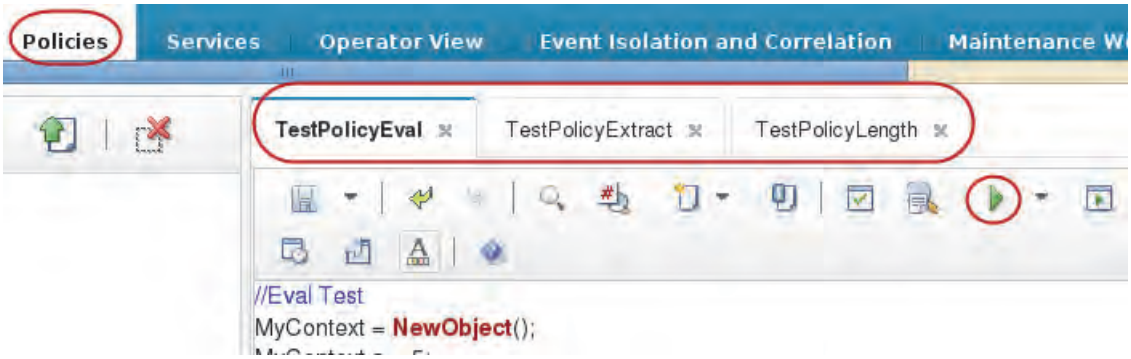
3. Edit the **PolicyLogger** configuration either by double-clicking the **Policy Logger** entry in the **Services** list or by right-clicking the entry and selecting **EDIT.** Select the check box to enable **Collect Reports** as shown in the display**.**



4. Click the **Save Service** icon.



5. Open the **Policies** window and run three policies that you created in **Unit 4: Policies Exercises** by clicking the green arrow for each. Run each policy three to five times to create reporting data.

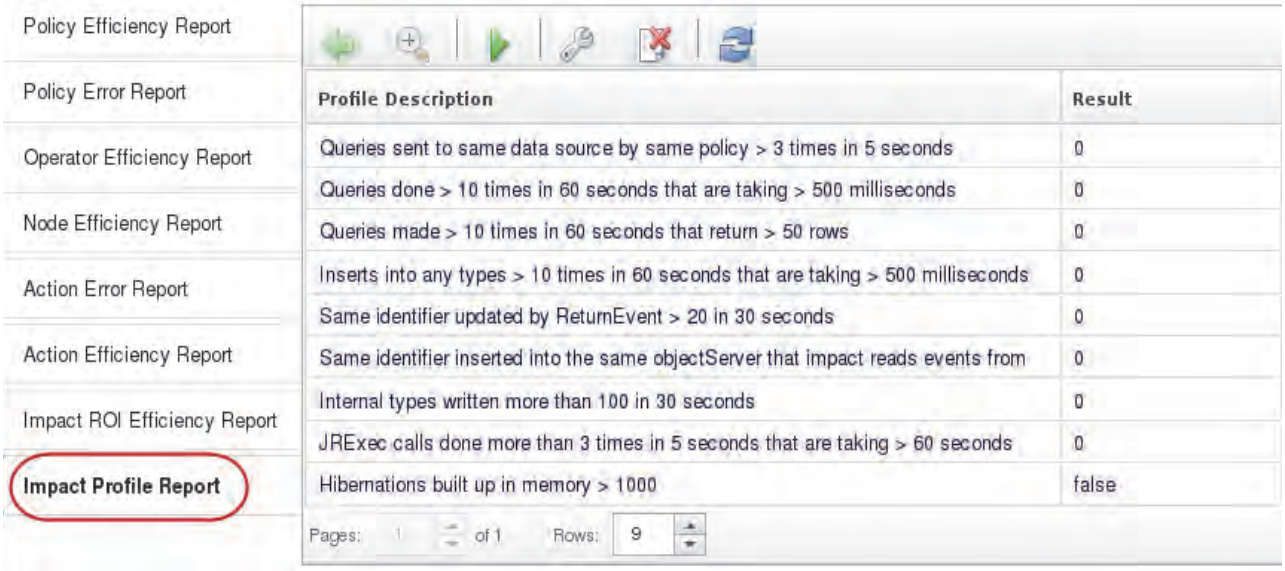


Exercise 2 Creating the Impact Profile report

1. Click the **Reports** tab on the far right.

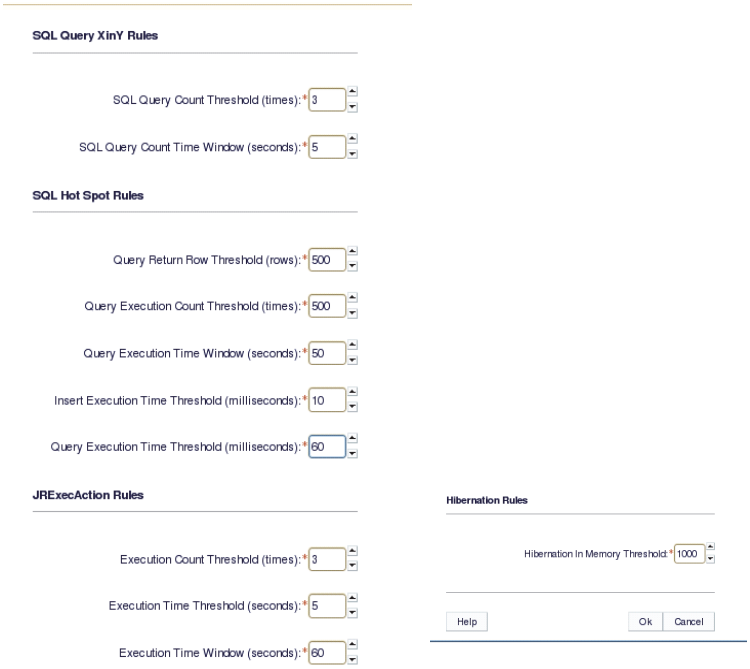
The **Impact Profile Report** provides information about the efficiency of the Netcool/Impact server configuration. The detailed view shows information about SQL queries, policies that issued the queries, type of action, data sources queried, and metrics.

2. Select the **Impact Profile Report**.



3. Click the wrench icon to **Open Configuration** window.

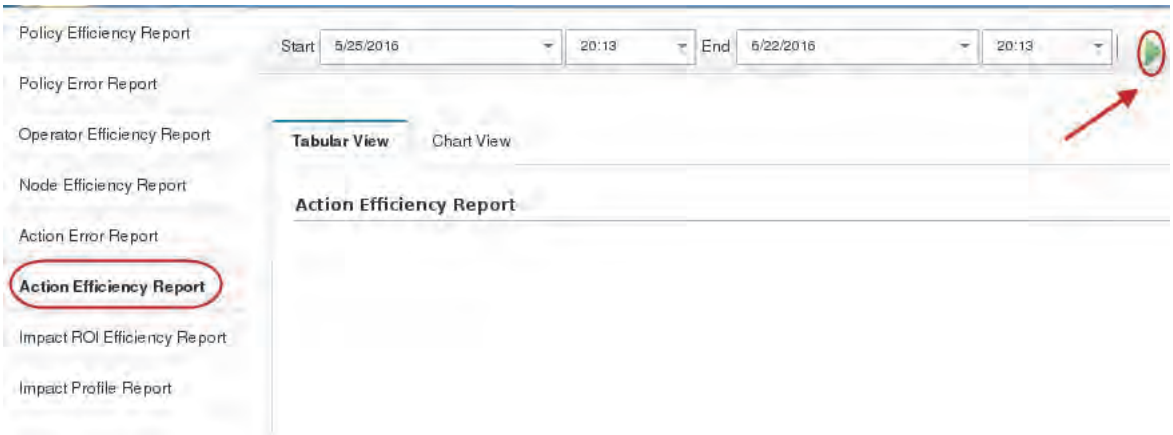
The **Impact Profile Report Rules Editor** lists all the queries you can use to generate an **Impact Profile Report**. The current configuration is as displayed. This report is useful if you want to evaluate sql efficiency in an existing Netcool/Impact production environment.



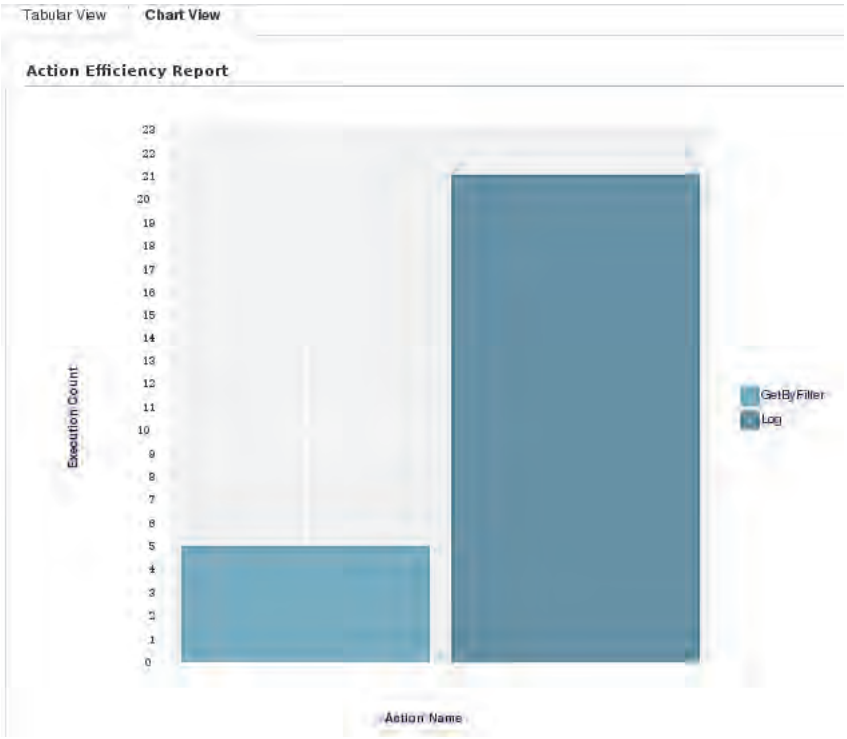
Exercise 3 Creating an Action Efficiency report

The Action Efficiency report shows the total number of actions that were processed over a selected time range. You use this report to determine how many actions the Impact server performed and which actions are used most often.

1. Select the **Action Efficiency Report**. Set the **Start** and **End** dates to include yesterday and tomorrow. Click the green arrow to run the report.

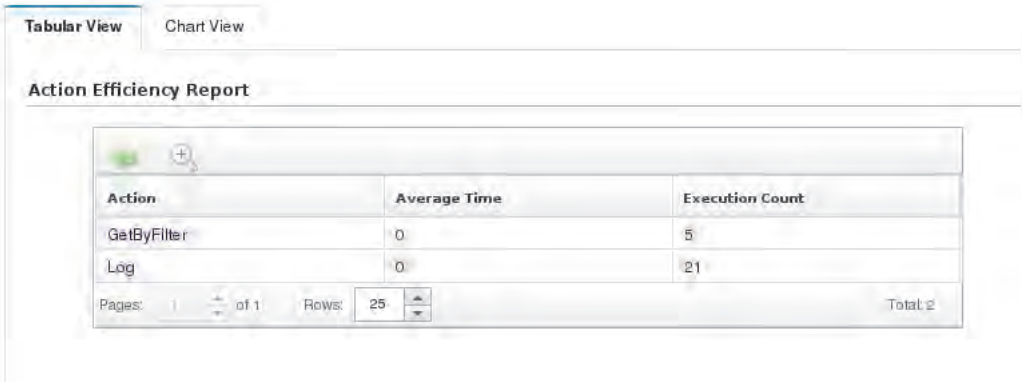


The tabular view and the chart view shows how many times each action has been performed by policies in the Impact server.



2. Click the **Tabular View** tab.

   The tabular view contains a table that shows how many times an action was run and the average time it took to process the action.

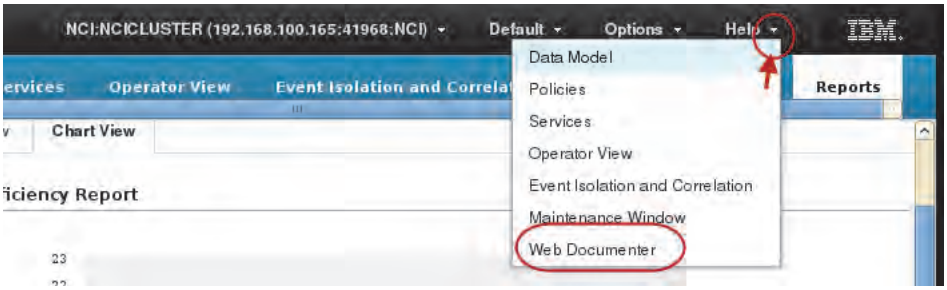


Note: The policies have Average Times equal to 0 because they run very quickly, and use very little resource.

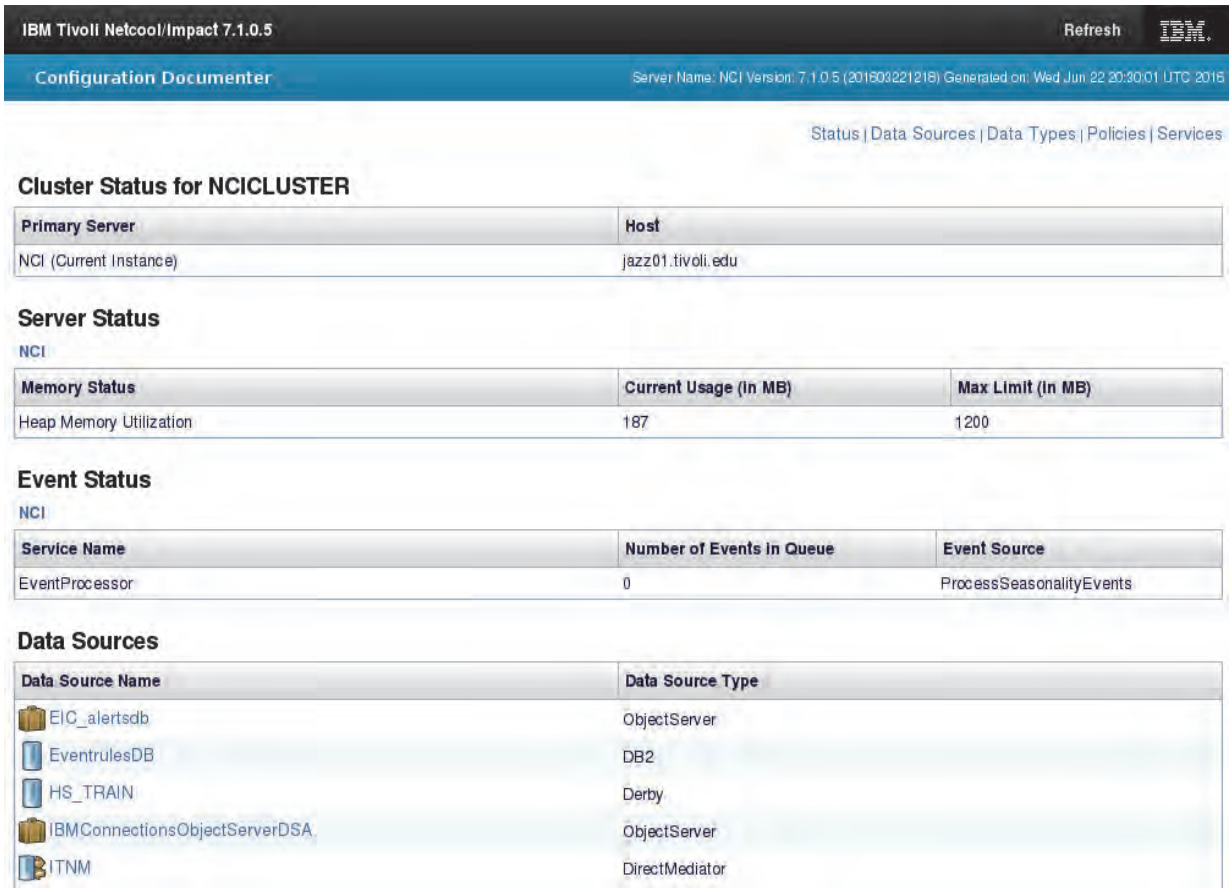
Exercise 4 Viewing the configuration documenter contents

In Netcool/Impact v7 the **Configuration Documenter** is also referred to as the **Web Documenter**, and is accessible from the **Help** menu.

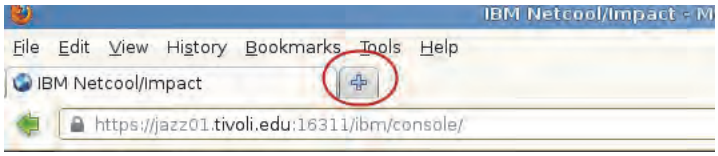
1. Two options are available to access the tool:
   a. Open in the GUI. Click **Help** and select **Web documenter** in the list to start the tool.



The Configuration Documenter pane is displayed.



b. Minimize the **Configuration Documenter** window. To open in a separate window. Click the plus sign.



2. Cut and paste or type the URL into the **New Tab**. Click the green arrow next to the URL to display the Configuration Documenter page.

`https://jazz01.tivoli.edu:16311/documenter/WebDoc.jsp?clusterName=NCICLUSTER`

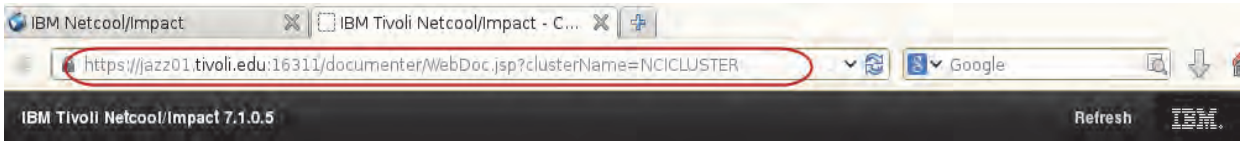
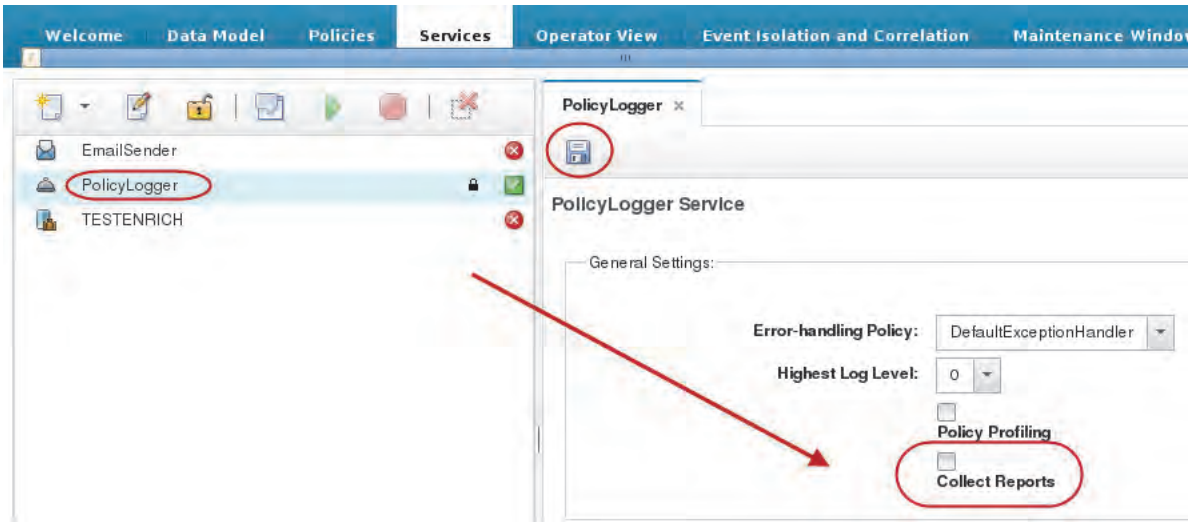


Figure 1 Data Model selection in earlier releases

Exercise 5 Disable reporting

Collecting data for reports can be resource intensive. Turn off reporting when reports are not required.

1. Click the **Services** tab to list all services.
2. Edit the **PolicyLogger** service
3. Delete the **Collect Reports** check mark to disable reporting.
4. Click the **Save Service** icon.



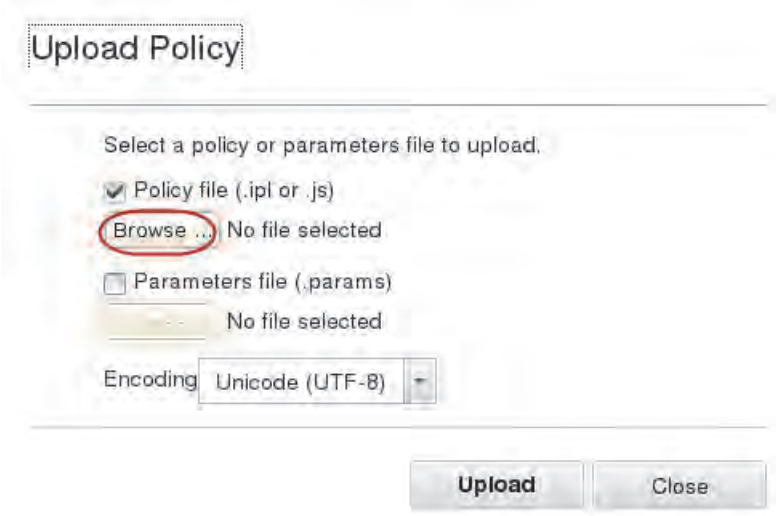
Unit 11 Operator views exercises

This exercise is an elementary introduction to the operator view. An operator view is created with an information panel and an action button, which runs a policy that clears all events in the event list.

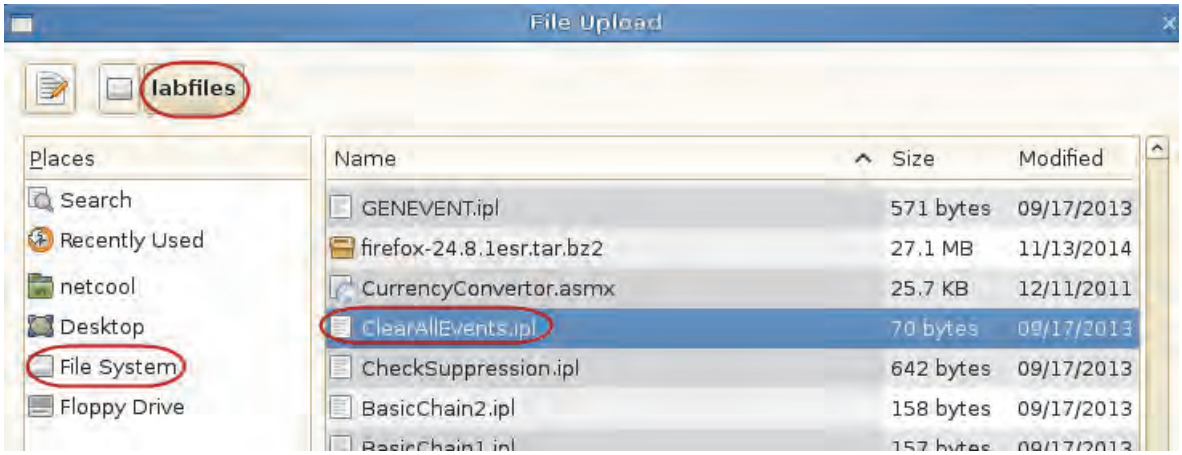
Exercise 1 Creating a basic operator view

Uploading the policy for the Action panel

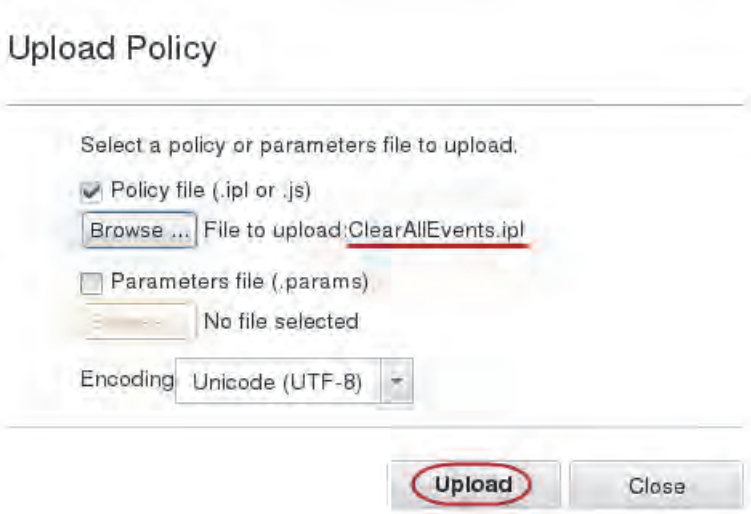
1. Click **Policies** to open the **Policies** tab. From the policy menu, click the **Upload a Policy File** icon  .
2. The **Upload Policy** box opens. Click **Browse**.



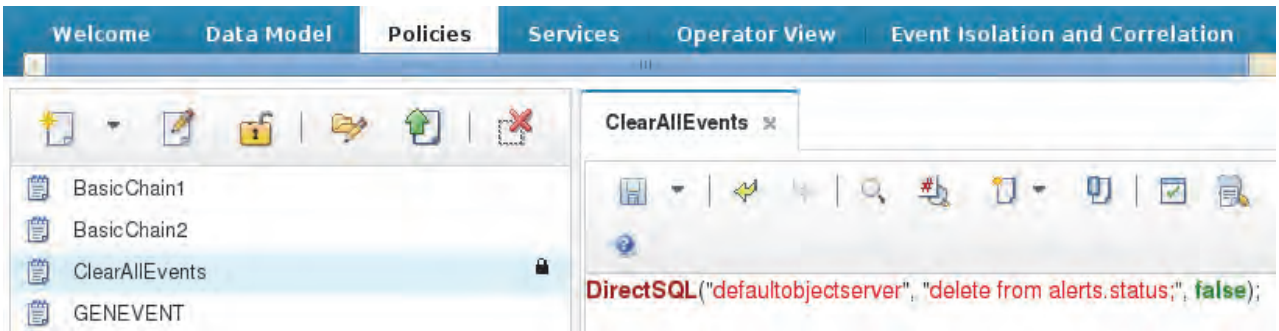
3. Locate the **/labfiles/ClearAllEvents.ipl** file.



4. With **ClearAllEvents.ipl** in the policy file window, click **Upload** to import the policy.



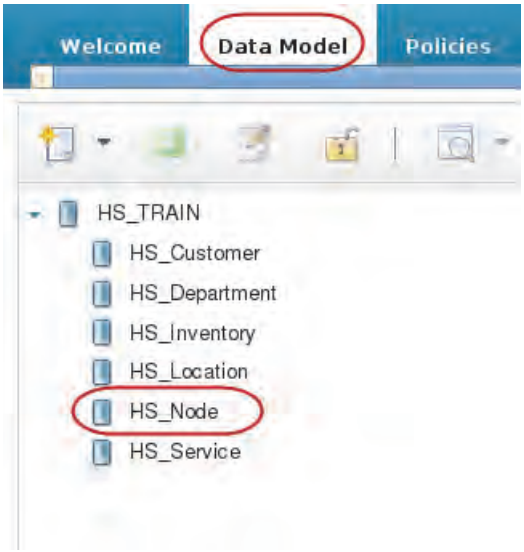
The **ClearAllEvents** policy uses the Netcool/Impact DirectSQL function so that Netcool/Impact policies issue SQL statements against a data source. The policy deletes all events in the ObjectServer.



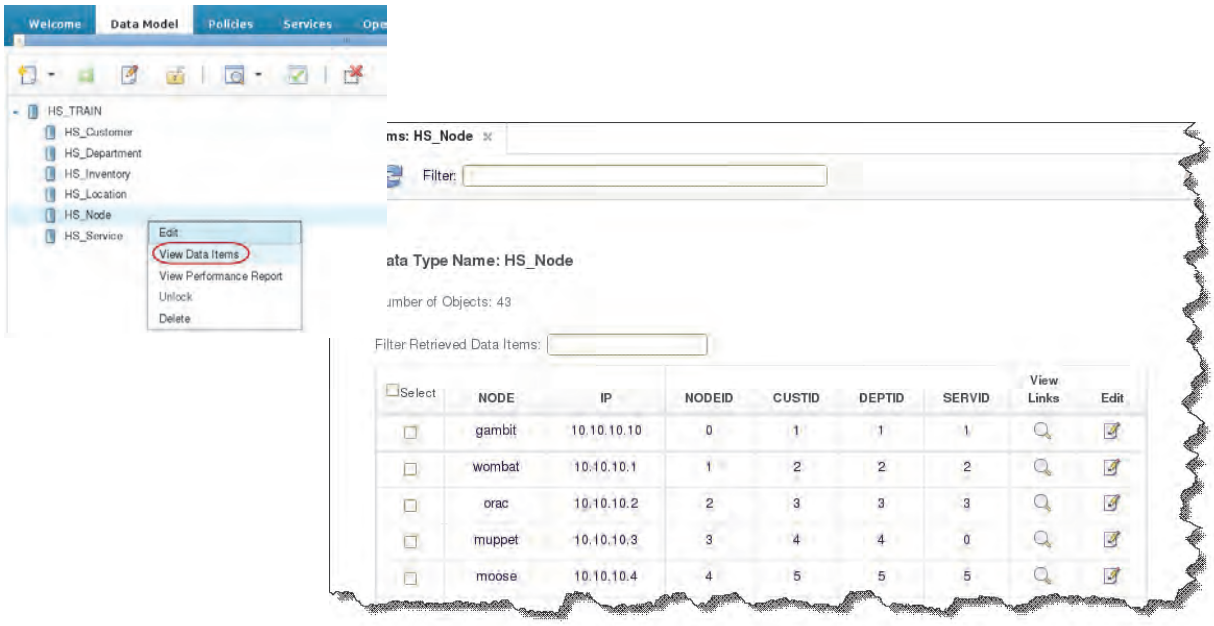
Creating and configuring the data source

Create a data source definition for the database that holds the sample data for the operator view. The sample data was loaded into the Derby database. You created the **HS_TRAIN** data source and the **HS_Node** data type earlier.

1. Verify that **HS_Node** is listed under the data source **HS_TRAIN** in the **Data Model** tab.



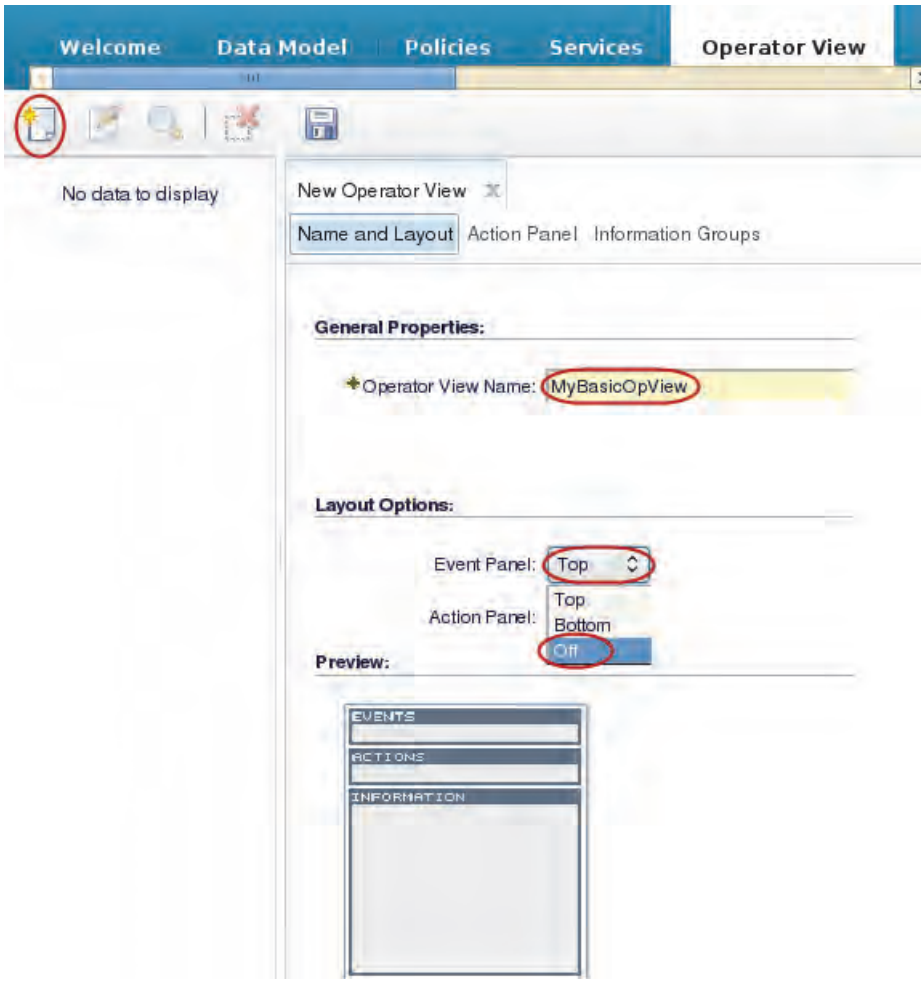
2. Verify the data type definition by right-clicking **HS_Node** and selecting **View Data Items**.



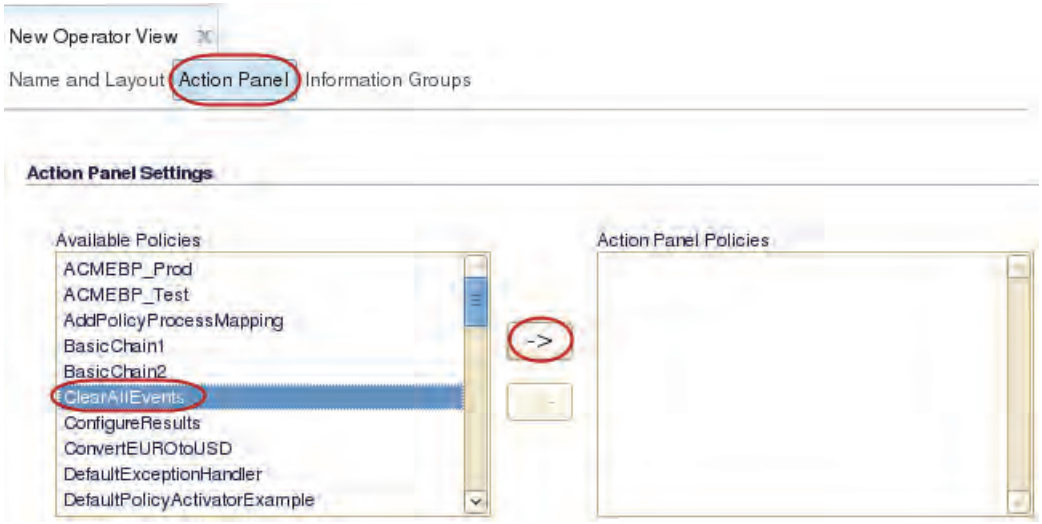
Creating the operator view

Create the basic operator view.

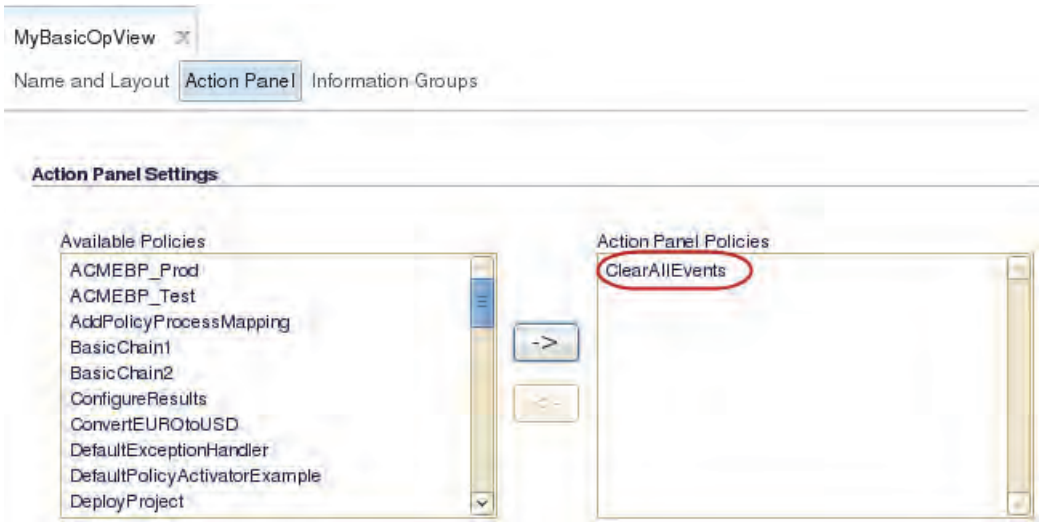
1. Click **Operator Views** to display the **Operator View** tab. Click the **New Operator View** icon.
2. Configure the new operator view as follows:
   - Operator View Name: **MyBasicOpView**
   - Event Panel: **Off**



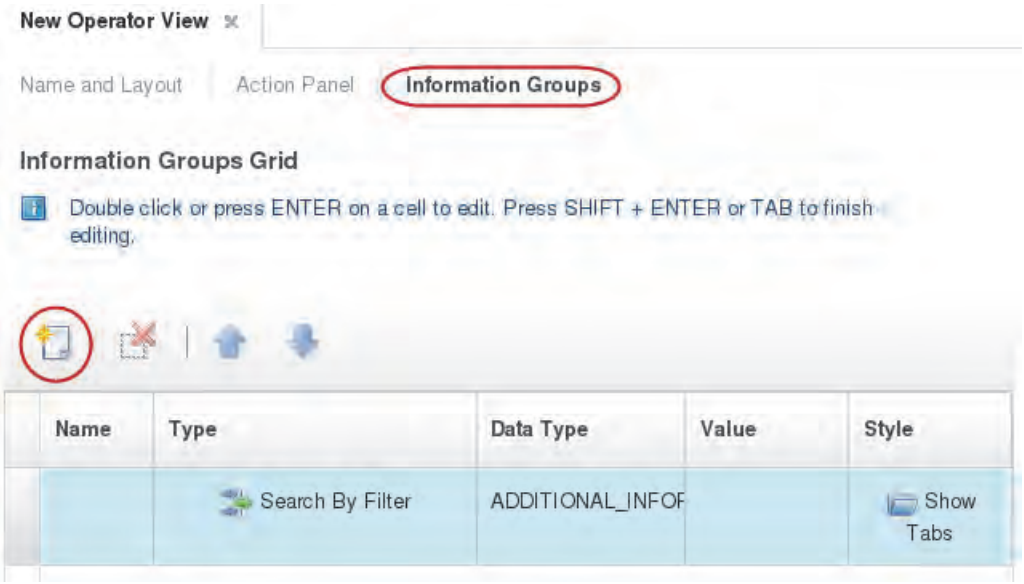
3. Click the **Action Panel** tab and move **ClearAllEvents** from the **Available Policies** list to the **Action Panel Polices** list.



The resulting Action Panel Policies window opens.

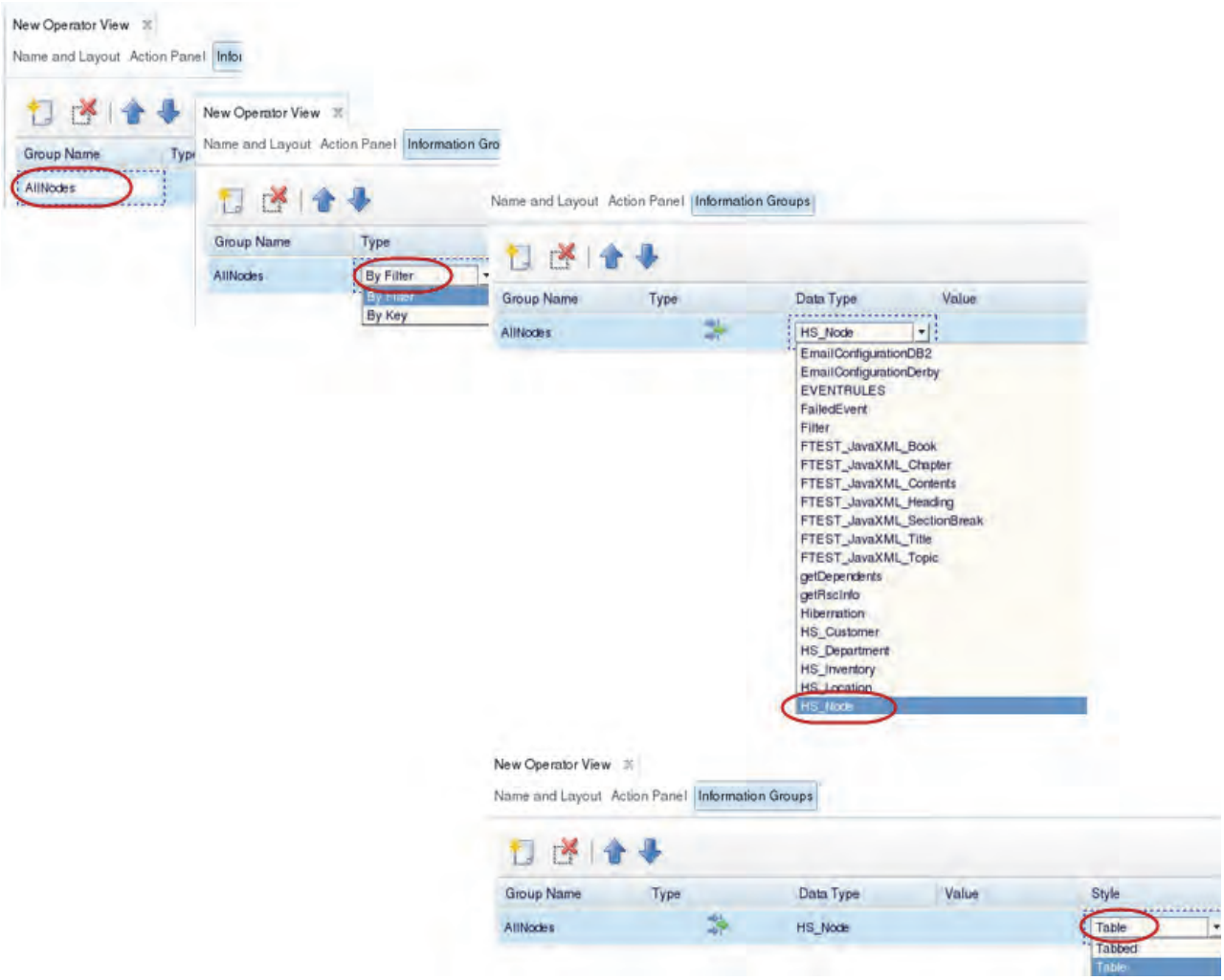


4. Click the **Information Groups** tab. Click the **New Information Group** icon.

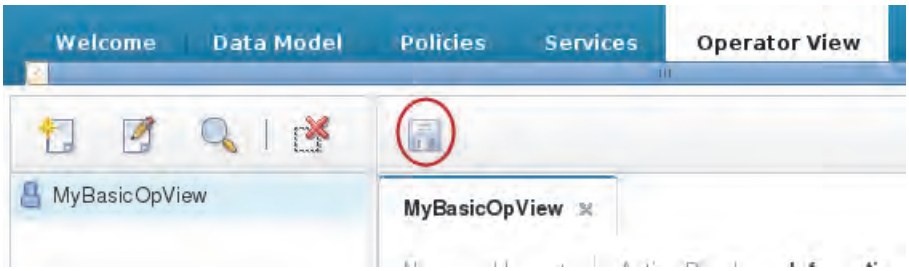


5. Specify the following information by double-clicking the dark blue bar under group name, type, data type, and style to edit the entries:
   - Group Name: **AllNodes**
   - Type: **Filter**
   - Data Type: **HS_Node**
   - Style: **Table**

The results are like the following example.

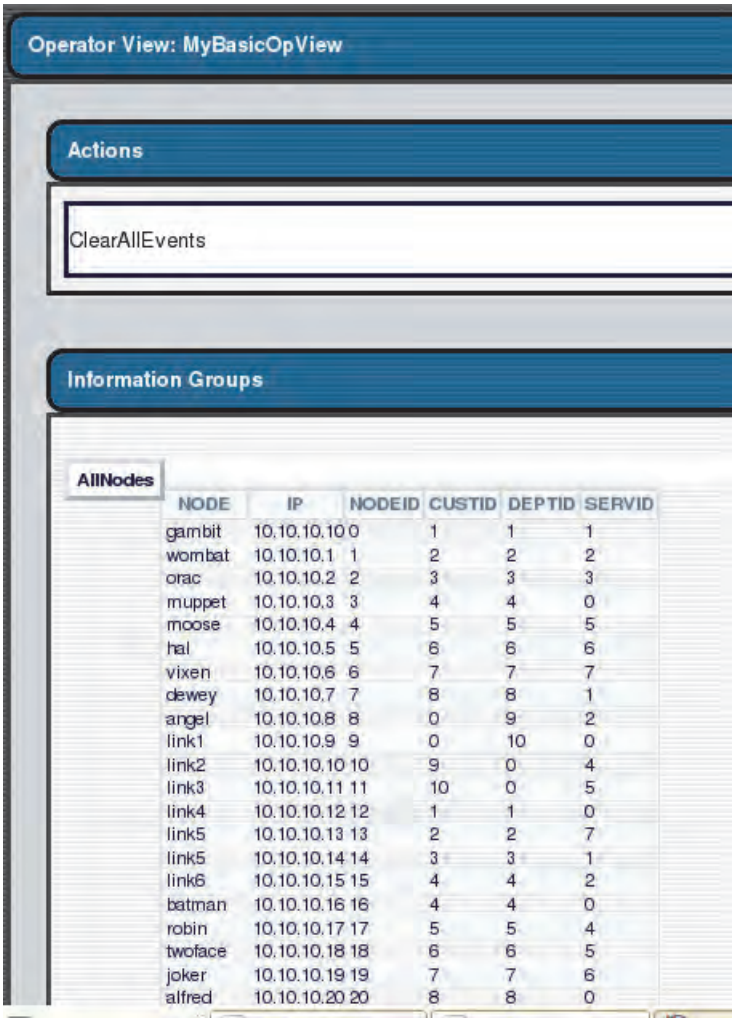


6. Click the icon to save the operator view definition.



7. Click the **View Operator View Display** (magnifying glass) icon to display the operator view data. 

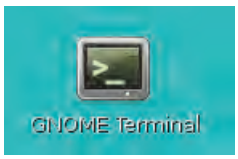
The operator view **MyBasicOpView** opens as shown:



NODE	IP	NODEID	CUSTID	DEPTID	SERVID
gambit	10.10.10.10	0	1	1	1
wombat	10.10.10.1	1	2	2	2
orac	10.10.10.2	2	3	3	3
muppet	10.10.10.3	3	4	4	0
moose	10.10.10.4	4	5	5	5
hal	10.10.10.5	5	6	6	6
vixen	10.10.10.6	6	7	7	7
dewey	10.10.10.7	7	8	8	1
angel	10.10.10.8	8	0	9	2
link1	10.10.10.9	9	0	10	0
link2	10.10.10.10	10	9	0	4
link3	10.10.10.11	11	10	0	5
link4	10.10.10.12	12	1	1	0
link5	10.10.10.13	13	2	2	7
link5	10.10.10.14	14	3	3	1
link6	10.10.10.15	15	4	4	2
batman	10.10.10.16	16	4	4	0
robin	10.10.10.17	17	5	5	4
twoface	10.10.10.18	18	6	6	5
joker	10.10.10.19	19	7	7	6
alfred	10.10.10.20	20	8	8	0

Note: You can also view the operator view using this URL:
https://jazz01.tivoli.edu:16311/opview/displays/NCICLUSTER-MyBasicOpView.html

8. On the Linux desktop, click the **GNOME Terminal** icon.

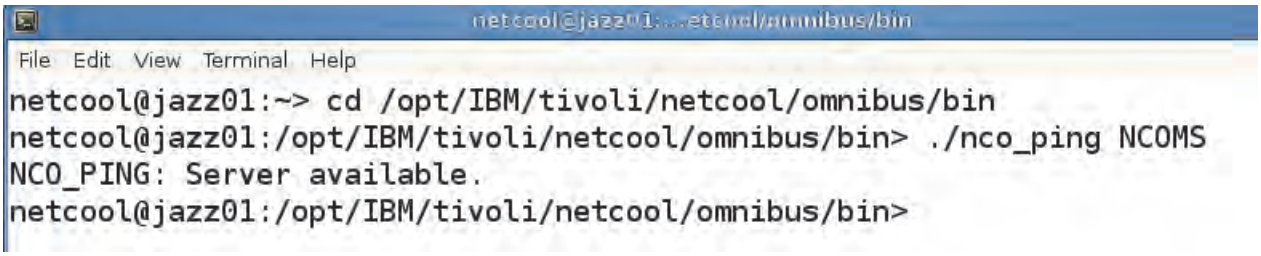


9. Go to the Netcool/OMNIbus **bin** directory by typing the following command:

```
cd /opt/IBM/tivoli/netcool/omnibus/bin
```

10. Verify that Netcool/OMNIbus is started with the **nco_ping** command:

    ./nco_ping NCOMS &



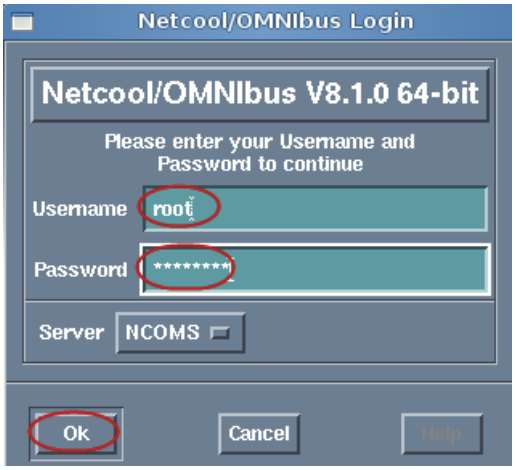
    If the server is unavailable, start it with the following command:

    `./nco_objserv &`

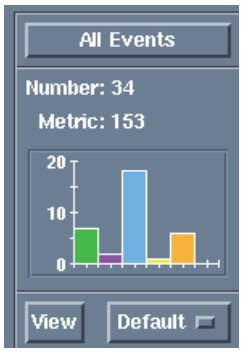
11. Verify that there are existing alerts. Open the Netcool/OMNIbus event list with the following command:

    `./nco_event &`

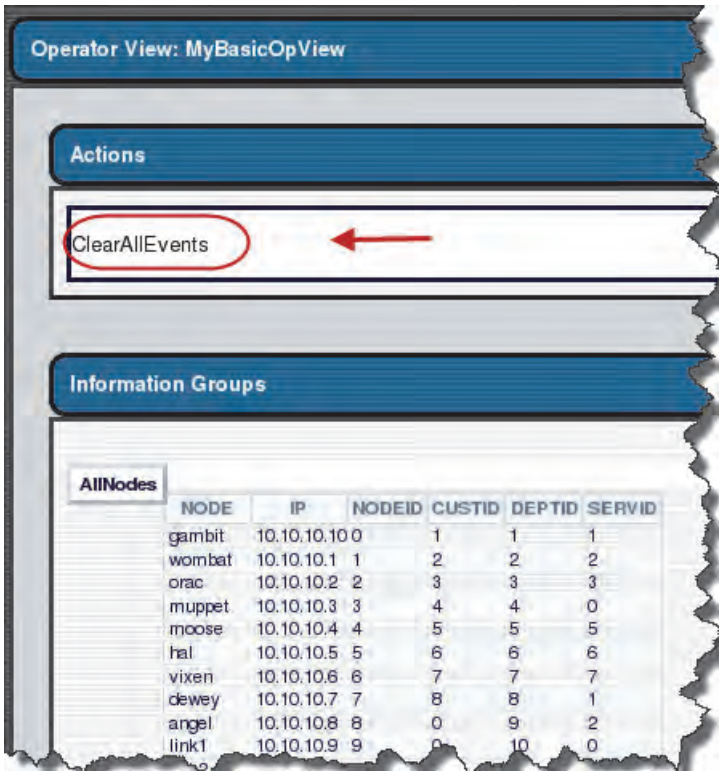
12. When prompted, type **root** as the user and **object00** for the password. Click **Ok**.



13. Expand the screen to show the alert group **All Events** to view the existing events.

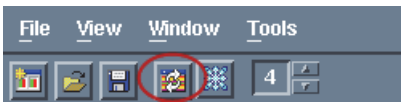


14. To test the **Actions** link in the operator view, click **ClearAllEvents** to clear all Netcool/OMNIbus events.



15. Click **Ok** to close the window.

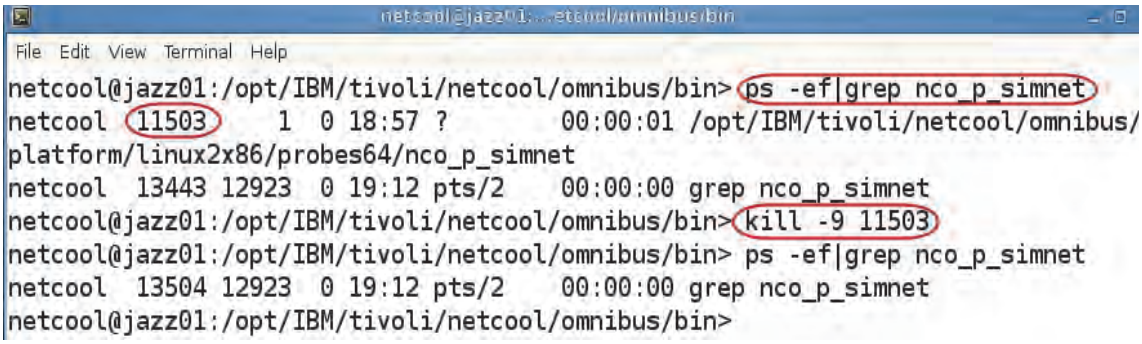
16. Verify that this action occurred in the Netcool/OMNIbus **Event Viewer**. Be sure that you refresh the view in Netcool/OMNIbus.



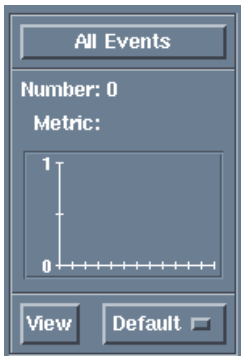
17. From a terminal session, make sure the Simnet Probe is not running to see all the events clear.
    a. To stop the simnet process if it is running, type the commands as shown here.

```
ps -ef|grep nco_p_simnet
kill -9 11503
```

Note: Use the process id on your system to replace 11503.



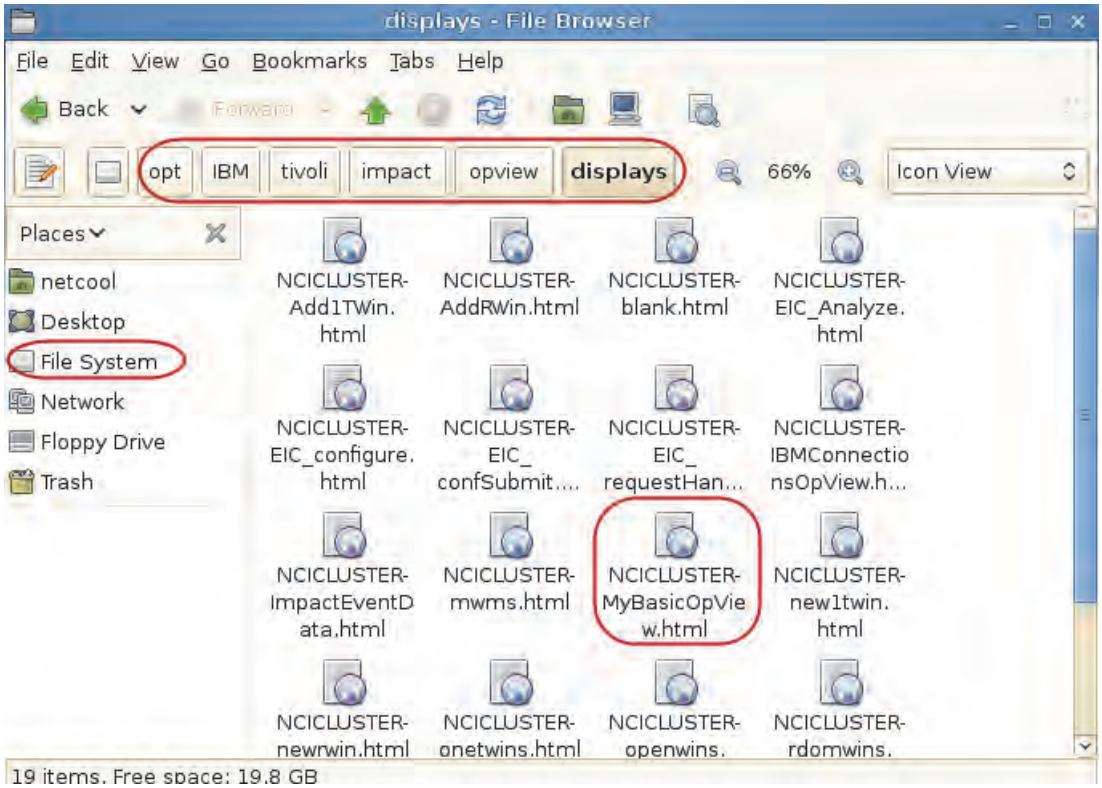
b. Click **ClearAllEvents** in the **Operator View.** Refresh **All Events** in OMNIbus**.** The resulting view in OMNIbus is as shown in the following screen capture:



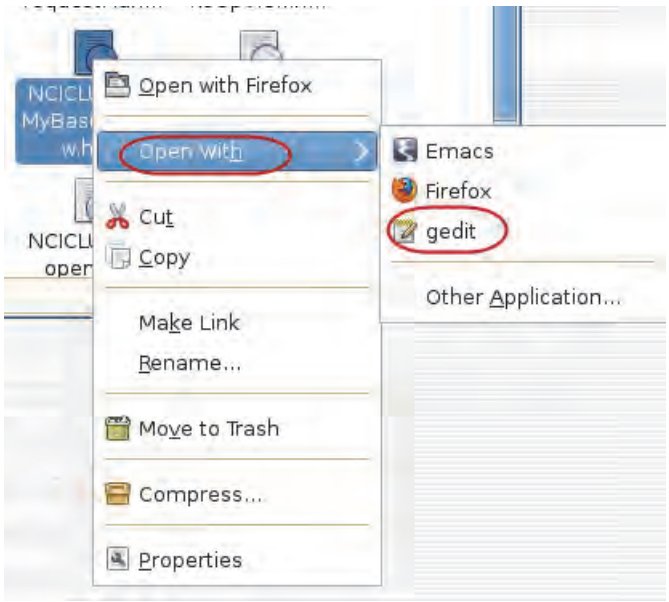
After the operator view is created, a policy is automatically created with the name **OpView_MyBasicOpview**. A corresponding HTML file called **NCICLUSTER-MyBasicOpView.html** is created in **$NCHOME/opview/displays**.

18. Double click the icon shown below. Examine the **NCICLUSTER-MyBasicOpView.html** file in the following directory:

    /opt/IBM/tivoli/impact/opview/displays

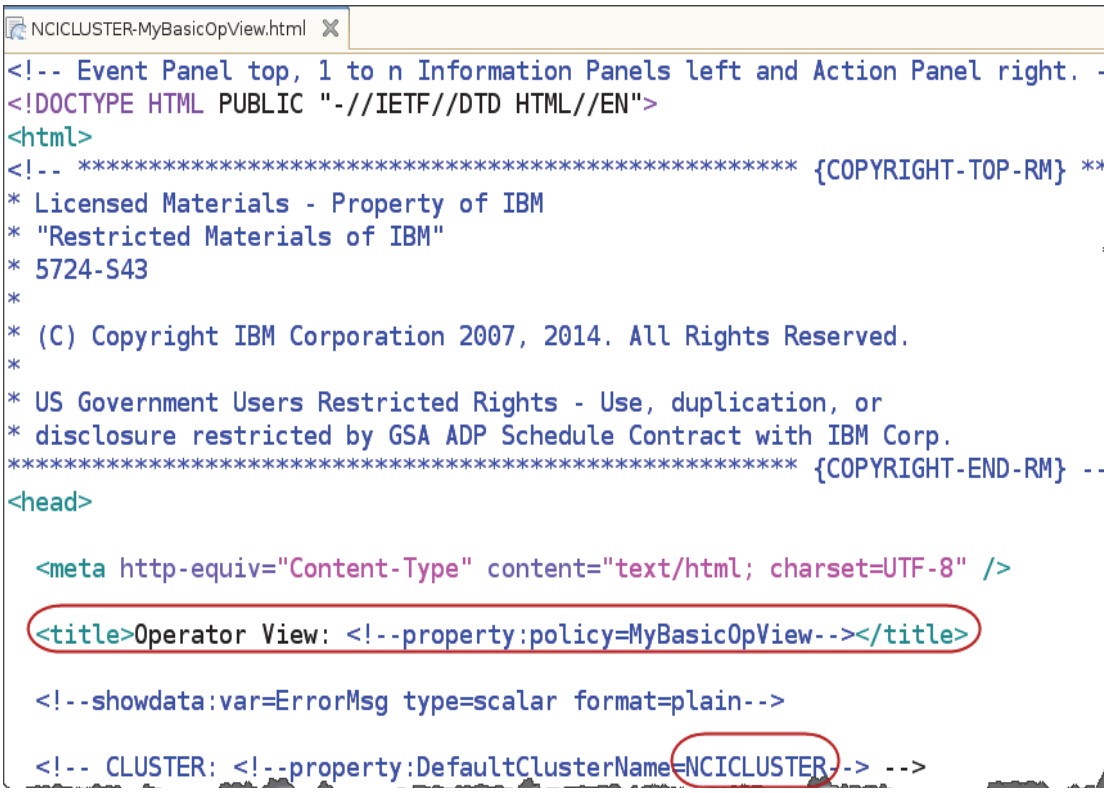


19. Right-click the file icon and select **gedit** to edit the file.



Locate the operator view smart tags that are responsible for the appearance and content of the newly created **MyBasicOpView**.

Note the two property smart tags: one to define the Netcool/Impact policy behind the web page and one to identify the Netcool/Impact cluster. The smart tags are HTML comments.



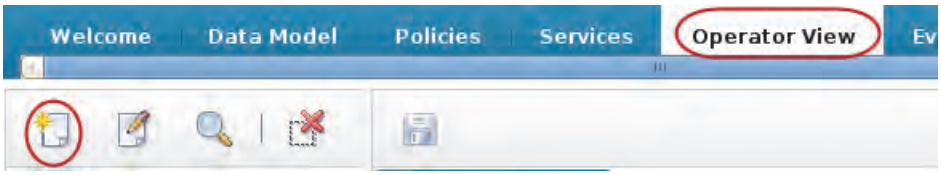
Exercise 2 Creating an advanced (custom) operator view

Netcool/Impact operator views have two primary components: *a policy* (or policies) that retrieve data fields from one or more sources, and *an HTML page*. When the HTML page is opened, it starts the Netcool/Impact policy. The policy contains the required data fields, which reference smart tags in the HTML page.

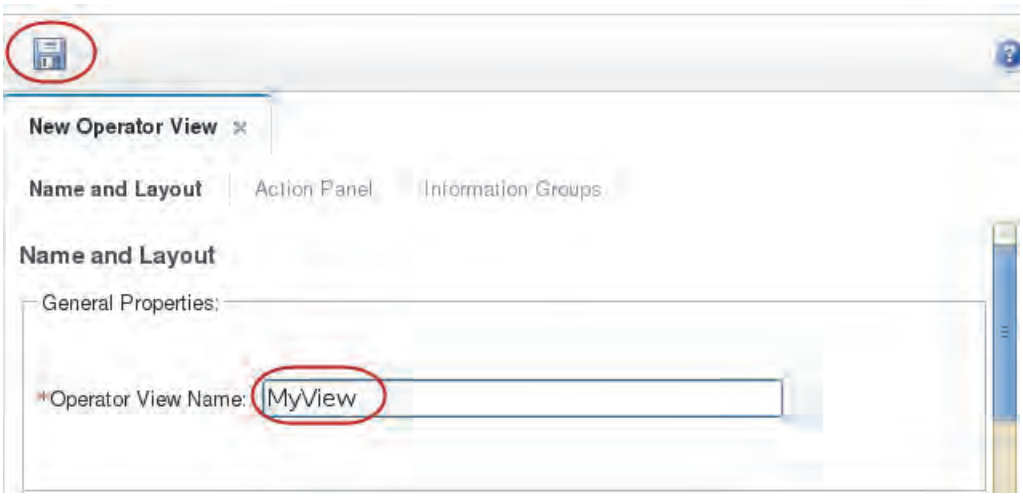
This document does not cover the detailed mechanics of HTML pages. The exercise is designed to introduce the process required to produce an advanced operator view.

Creating the base operator views

1. Click **Operator Views** to display the **Operator View** tab. Click the **New Operator View** icon.

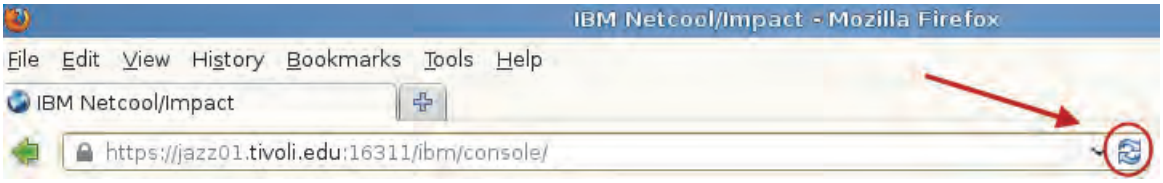


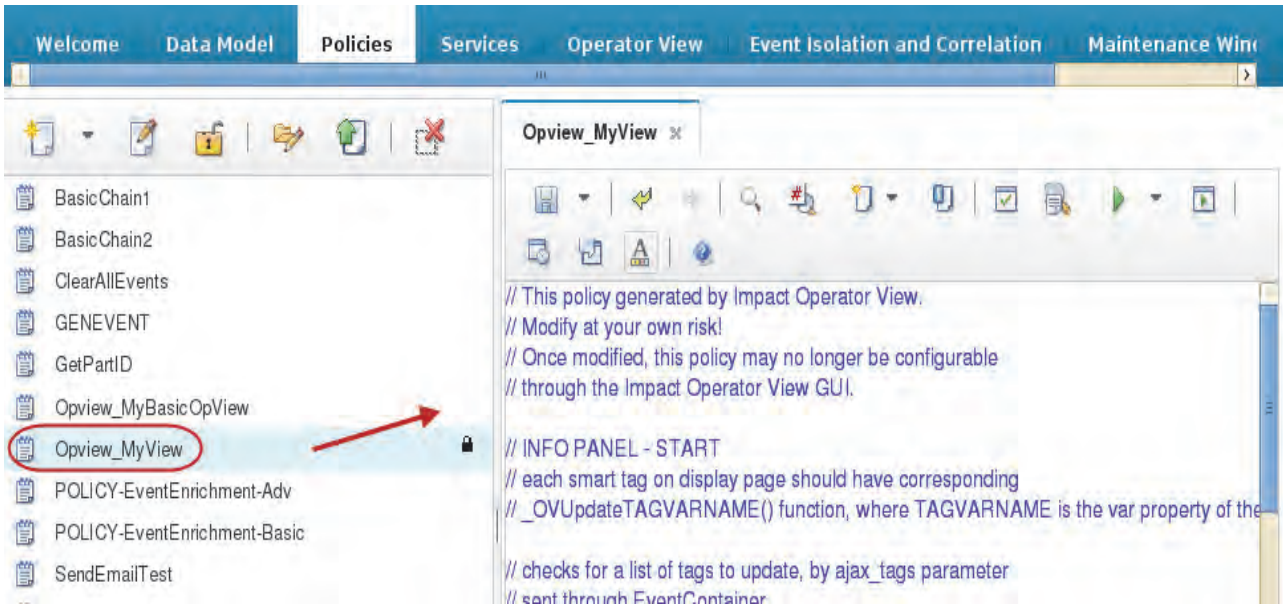
2. Create the first operator view and name it **MyView**. Save it without any customization.



3. When you save the operator view, a policy named **Opview_MyView** is automatically generated and is displayed in the list of policies. Select the new policy in the list to display it in the policy editor, as shown in the following figure.

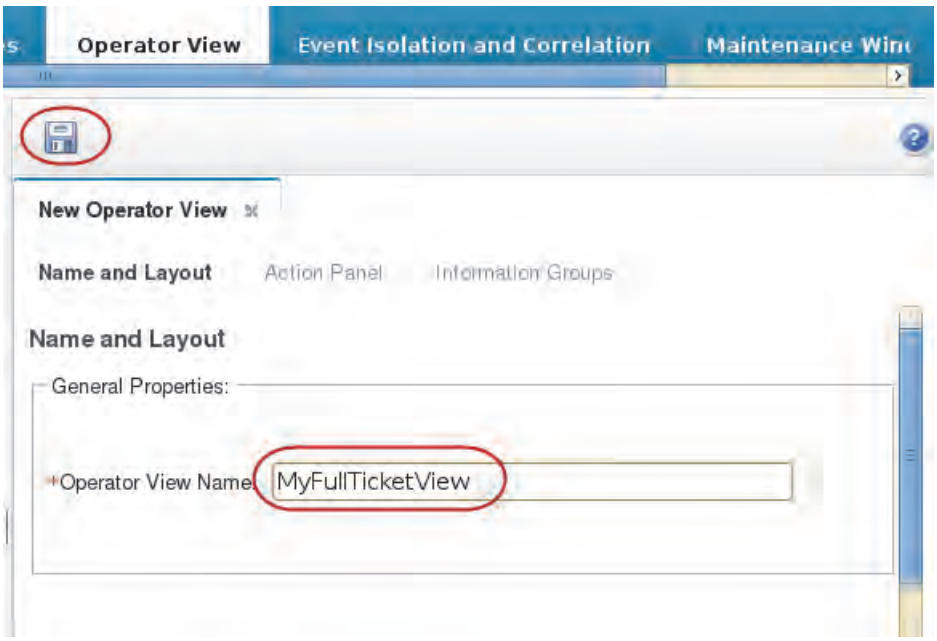
Note: If the operator view policy does not open, refresh the browser.





In addition to the policy, an HTML page is also automatically generated. This page is located in **/opt/IBM/tivoli/impact/opview/displays** and is named **NCICLUSTER-MyView.html**.

4. Create another operator view called **MyFullTicketView**. Save it without customization.



5. When you save it, another policy is automatically generated, and it shows in the list of policies as **Opview_MyFullTicketView.** The corresponding HTML file is also created.

Sample database

The data that is used for this operator view exercise is already loaded into the Derby database. These four tables are required for this lab exercise:

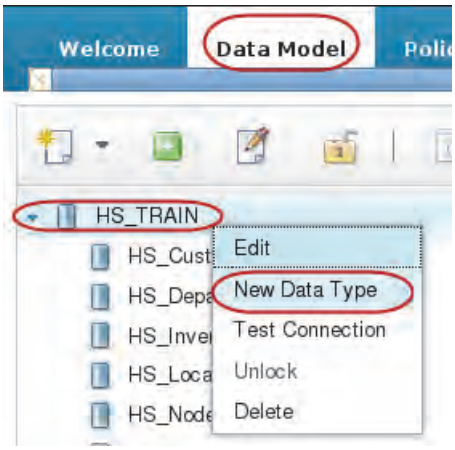
- ASSET
- SERVAPPS
- SERVINFO
- TICKET

Creating the data source and data types required by the policies

The policies that are used in the operator views require the following data source and data type definitions:

- Data source

Because the sample tables were installed in the Derby database and there is already a data source definition for it, you can use the same data source, **HS_TRAIN**.



- Data types

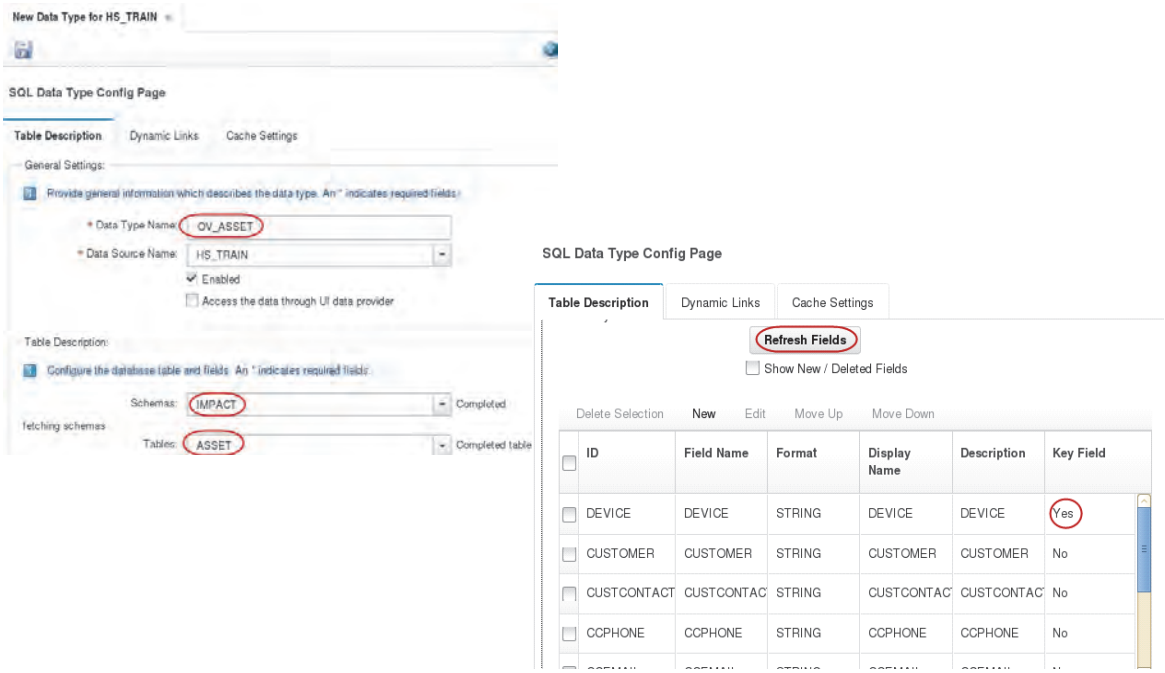
  You need a data type definition for each of the four sample tables mentioned previously.

1. Create four data types under the data source **HS_TRAIN** using **Schema: IMPACT**. Use the following information for other specifications. Remember to click **Refresh** to see the fields in the table. Don't forget to save your changes.

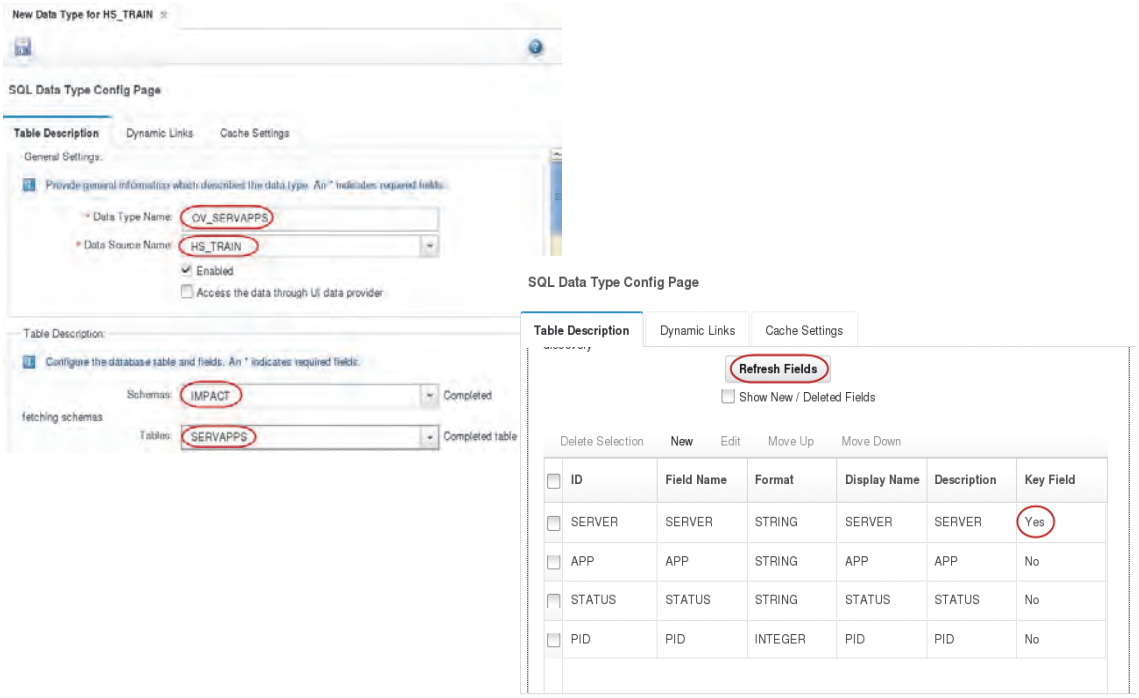
Data Type Name	Key field	Table
OV_ASSET	DEVICE	ASSET
OV_SERVAPPS	SERVER	SERVAPPS
OV_SERVINFO	SNAME	SERVINFO
OV_TICKET	TICKETNO	TICKET

The resulting data model is like the following examples:

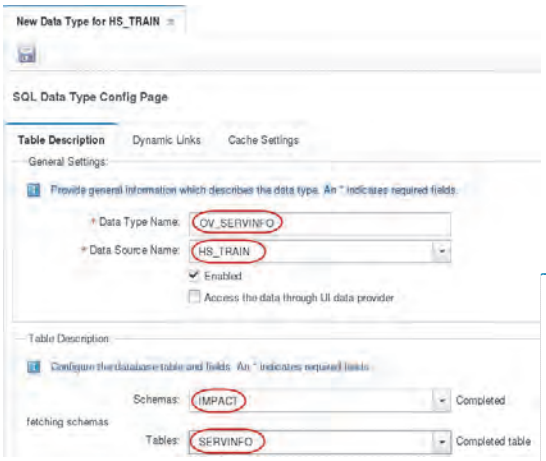
OV_ASSET

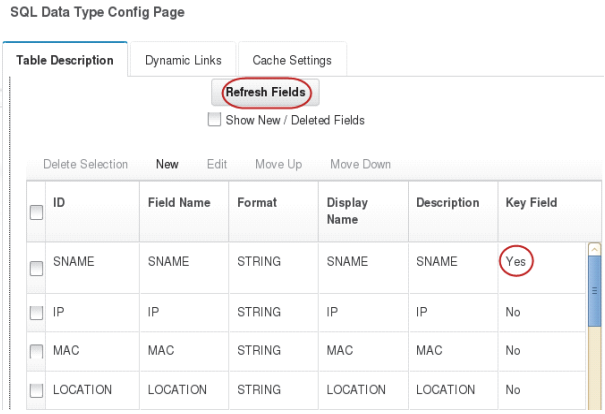


OV_SERVAPPS

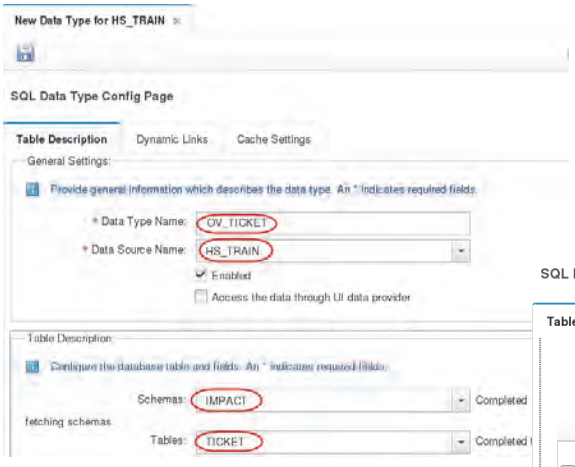


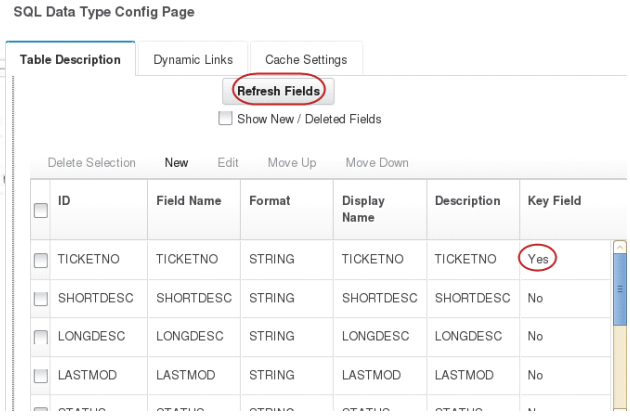
OV_SERVINFO



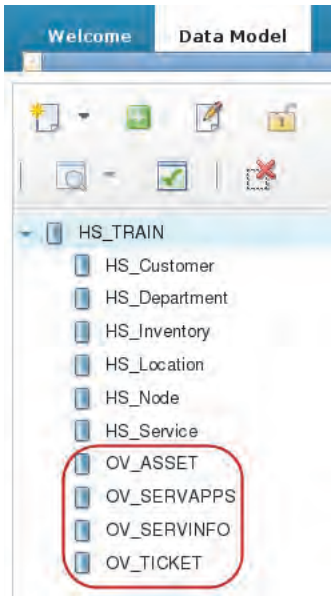


OV_TICKET





Resulting Data Model



Building the web pages

At the beginning of this exercise, you created two operator views:

- MyView
- MyFullTicketView

The process of generating those operator views created Netcool/Impact policies and HTML pages. Typically, these provide the starting point for the customization that is required to produce the views. That process consists of modifying the HTML pages to include smart tags that reference data that is populated by Netcool/Impact. You revise the policies to retrieve data from various sources (data types) and produce the data used to populate the smart tags.

For this exercise, you replace the base files that were automatically generated by the creation of the two operator views with the corresponding HTML pages, and completed policies in the **labfiles** directory.

Policies

Two policies are provided. One policy is used to populate the base dashboard. The second policy is used to retrieve ticket detail and to populate an additional area on the dashboard. Each of these policies is provided in Netcool/Impact Policy Language (IPL) format. The easiest way to get these policies into Netcool/Impact is to copy and paste the policy text.

The policies can be found in this location:

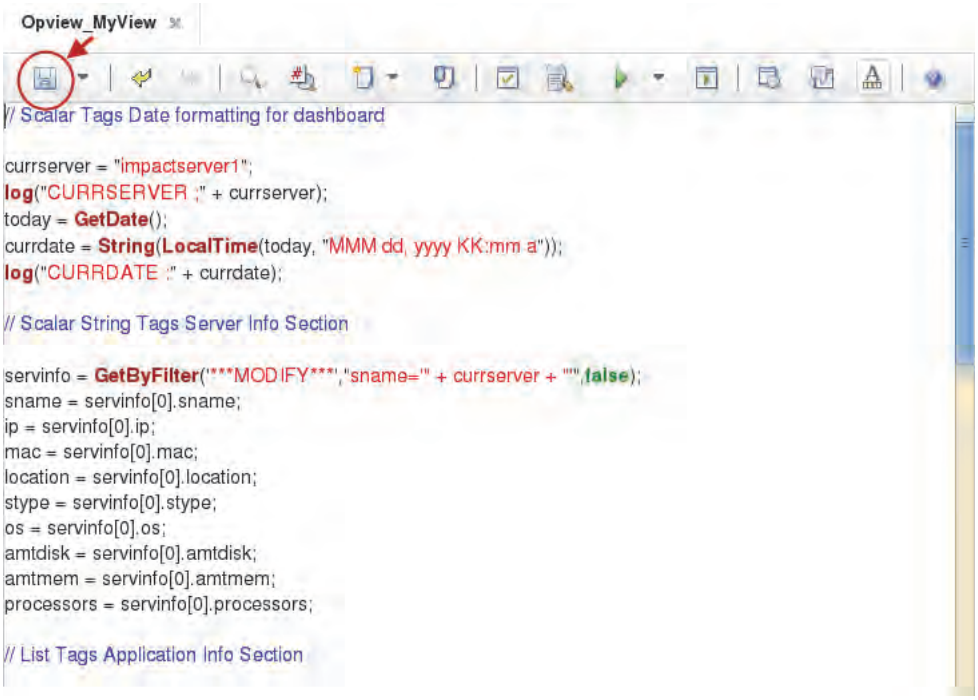
```
/labfiles/policies
   NCI_Opview_OVGuide.ipl
   NCI_Opview_FullTicket.ipl
```

1. Open the **Opview_MyView** policy in the Netcool/Impact policy editor. You use this policy to populate the contents of the base dashboard.

2. Delete all the policy contents from the editor window. Click the **Save** icon.

3. In a separate terminal window on the operating system (or upload the policy into the policy editor), edit the policy file, **/labfiles/policies/NCI_Opview_OVGuide.ipl**.

4. Select and copy the entire contents of the **NCI_Opview_OVGuide.ipl** policy, and paste it into the empty **Opview_MyView** policy editor window. Save the **Opview_MyView** policy.



Modifying the policies

Change the ***MODIFY*** entries in the policies as described in the next few steps.

1. Locate the following lines in the policy and change the data type name from ***MODIFY*** to the following name:

```
servinfo = GetByFilter('OV_SERVINFO',"sname='" + currserver + "'",false);
```

```
// Scalar Tags Date formatting for dashboard

currserver = "impactserver1";
log("CURRSERVER :" + currserver);
today = GetDate();
currdate = String(LocalTime(today, "MMM dd, yyyy KK:mm a"));
log("CURRDATE :" + currdate);

// Scalar String Tags Server Info Section

servinfo = GetByFilter('OV_SERVINFO',"sname='" + currserver + "'",false);
sname = servinfo[0].sname;
ip = servinfo[0].ip;
mac = servinfo[0].mac;
```

```
apps = GetByFilter('OV_SERVAPPS',"server='" + currserver + "'",false);
```

```
// List Tags Application Info Section

apps = GetByFilter('OV_SERVAPPS',"server='" + currserver + "'",false);
appcount = Length(apps);
log("APPCOUNT :" + appcount);
i = 0;
```

```
asset = GetByFilter('OV_ASSET',"device ='" + ip + "'",false);
```

```
asset = GetByFilter('OV_ASSET',"device ='" + ip + "'",false);

custemail = asset[0].ccemail;
owneremail = asset[0].oemail;
ec_buttitle = "Email Customer";
```

```
tickets = GetByFilter('OV_TICKET',"catid ='" + ip + "'",false);
```

```
// OrgNode CustomTable Type Ticket Info View Section

tickets = GetByFilter('OV_TICKET',"catid ='" + ip + "'",false);
```

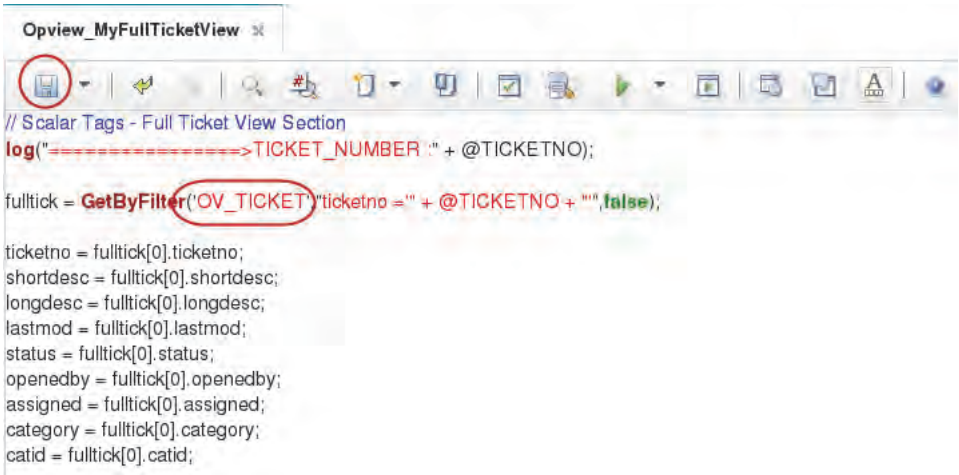
2. Check the syntax, and save the changes to the policy.
3. Open the **Opview_MyFullTicketView** policy in the Netcool/Impact policy editor.
4. Delete all of the policy text in the editor window. Click the Save icon.

5. View and edit **/labfiles/policies/NCI_Opview_FullTicket.ipl** and copy its contents into the empty policy in the Policy Editor window.

   This policy is used to retrieve ticket detail information when you select a **view** button on the dashboard. It contains a data type name that you must change as follows.

   ```
   fulltick = GetByFilter('OV_TICKET',"ticketno ='" + @TICKETNO + "'",false);
   ```

6. Save the changes to the policy, which are like the following example.



HTML pages

The HTML files are located in the **labfiles/html** directory. As was mentioned previously, HTML smart tags are used to place Netcool/Impact policy data into the web page. After completing this lab exercise, time permitting, go back and modify the HTML one panel at a time to see what changes.

In the first step in this exercise, you generated two operator views:

- Opview_MyView
- Opview_MyFullTicketView

That process also produced two corresponding HTML files:

```
/opt/IBM/tivoli/impact/opview/displays
   NCICLUSTER-MyView.html
   NCICLUSTER-MyFullTicketView.html
```

In the next step, you replace each of these files with the corresponding file from the **labfiles** directory. The replacement files are in this location:

```
/labfiles/html
   NCICLUSTER-OVGuide.html
   NCICLUSTER-FullTicket.html
```

1. From a command line, execute these commands to copy each file from the **labfiles** directory and rename them:

```
cd /labfiles/html
cp NCICLUSTER-OVGuide.html
/opt/IBM/tivoli/impact/opview/displays/NCICLUSTER-MyView.html
cp NCICLUSTER-FullTicket.html
/opt/IBM/tivoli/impact/opview/displays/NCICLUSTER-MyFullTicketView.html
```

Each of these HTML files requires additional modifications. These changes revise the policy references.

2. Use **gedit** to edit **/opt/IBM/tivoli/impact/opview/displays/NCICLUSTER-MyView.html**:
   a. Locate the following lines at the top of this file:

```
<!--
       <!--property:policy=***MODIFY***-->
       <!--property:DefaultClusterName=NCICLUSTER-->
-->
```

   b. Change the policy name reference:

```
   <!--property:policy=MyView-->
```

   c. Locate the following line:

```
action_policy="***MODIFY***"
```

   d. Change as follows:

```
   action_policy="MyFullTicketView"
```

   e. Save the changes.

3. Edit **/opt/IBM/tivoli/impact/opview/displays/NCICLUSTER-MyFullTicketView.html**.
   a. Locate the following line near the top of this file:

```
     <!--property:policy=***MODIFY***-->
```

   b. Change as follows:

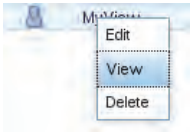
```
     <!--property:policy=MyFullTicketView-->
```

   c. Save the changes.

Viewing the advanced operator view

The advanced operator view configuration is complete and ready to test.

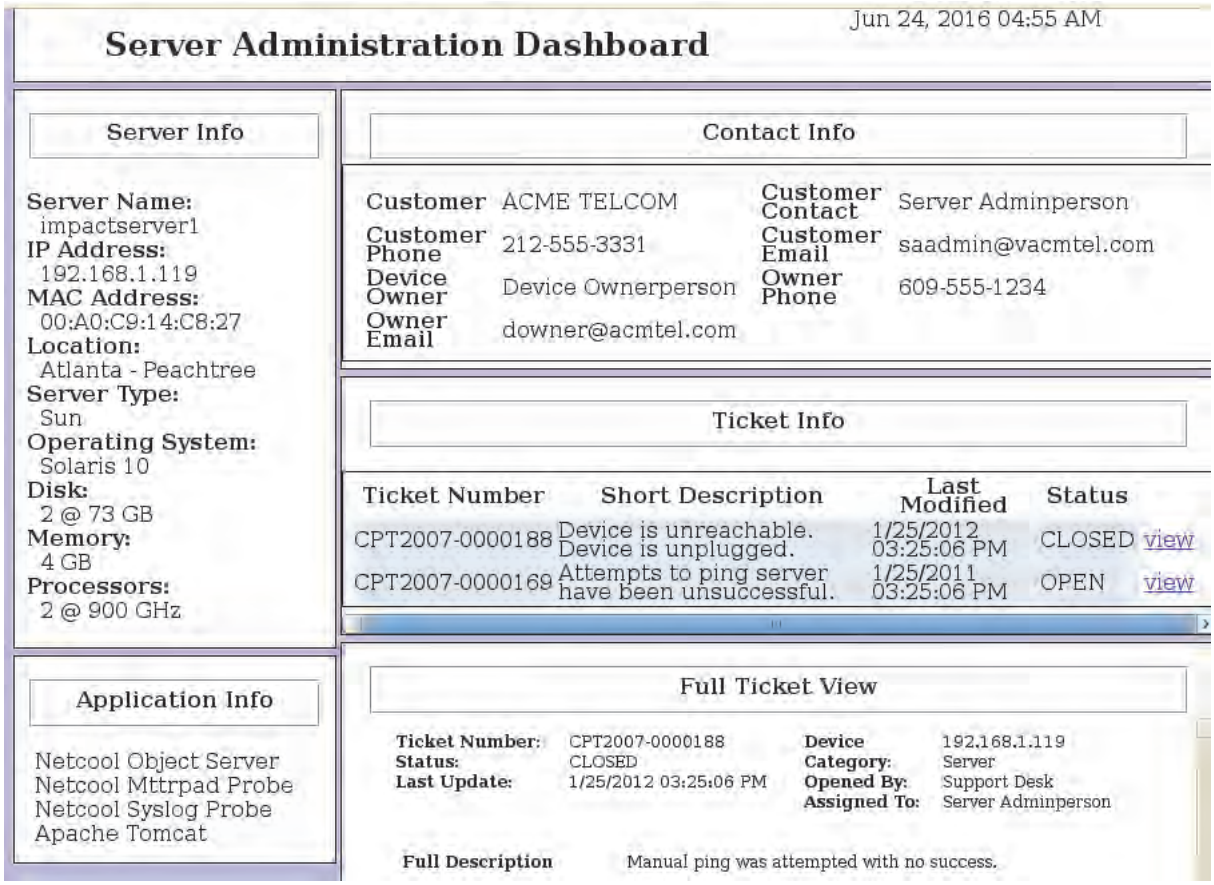
1. Click **Operator View** to display the **Operator View** tab.
   Highlight and right-click **MyView**. Select **View**.



   The operator view is also accessible from a browser.

2. Type the following URL:
   https://jazz01.tivoli.edu:16311/opview/displays/NCICLUSTER-MyView.html

3. Click the **view** link to the right of the ticket number to display the **Full Ticket View**. The results are similar to the following example.



When the window initially opens, the ticket detail in the lower portion of the dashboard is empty. Selecting the **view** action link shows the corresponding detail in the **Full Ticket View** panel.

This operator view page starts a policy that retrieves data for a specific device, **impactserver1**. A review of the **MyView** policy shows that the server name value was hard coded as **currserver = ”impactserver1”;**. You can revise the policy to replace the **currserver** constant with a variable. You can then use the variable to launch the operator view in context.

The **MyView** HTML file refers to the **Opview_MyView** Netcool/Impact policy. That policy retrieves data from four data tables and passes selected fields into the HTML page with smart tag references. The **view** link to the right of each ticket on the dashboard opens the **MyFullTicketView** HTML page for that particular ticket. This page references the **Opview_MyFullTicketView** policy. That policy retrieves detail data regarding the corresponding ticket. Those details are passed back to the HTML page using smart tag references.

View the **MyView HTML** file to examine the different smart tags that are associated with each of the panels of the finished main operator view page.

Unit 12 Working with web services exercises

In these exercises, you work with the Web Services wizard.

The Web Services wizard performs the following actions:

- Parses the WSDL file
- Creates the necessary JAR package
- Determines the available methods and parameters for making requests against the web services application
- Dynamically loads the Java classes into the Netcool/Impact server without requiring a restart of the Netcool/Impact server

The resulting policy contains the necessary parameters to invoke the user-selected method, invokes the method, waits for the response, and stores the returned information, which is now ready for additional processing by the policy.

Exercise 1 Creating a policy using the Web Services wizard

In this exercise, you build a policy to request a web services application to determine the conversion rate from Euro to US dollar. Typically, an Internet connection is required to test policies accessing public web services. However, the Internet will not be required for this exercise. You build a sample return result in order to examine web services, and Jazz for Service Management components.

Building the Web Service policy

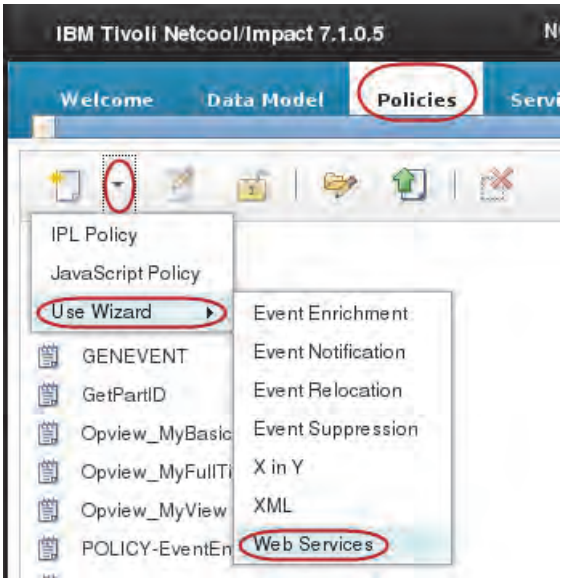
Note: The Netcool/Impact URL is https://jazz01.tivoli.edu:16311/ibm/console/.

1. Log in as **impactadmin** with password **object00**. Click **Go**.

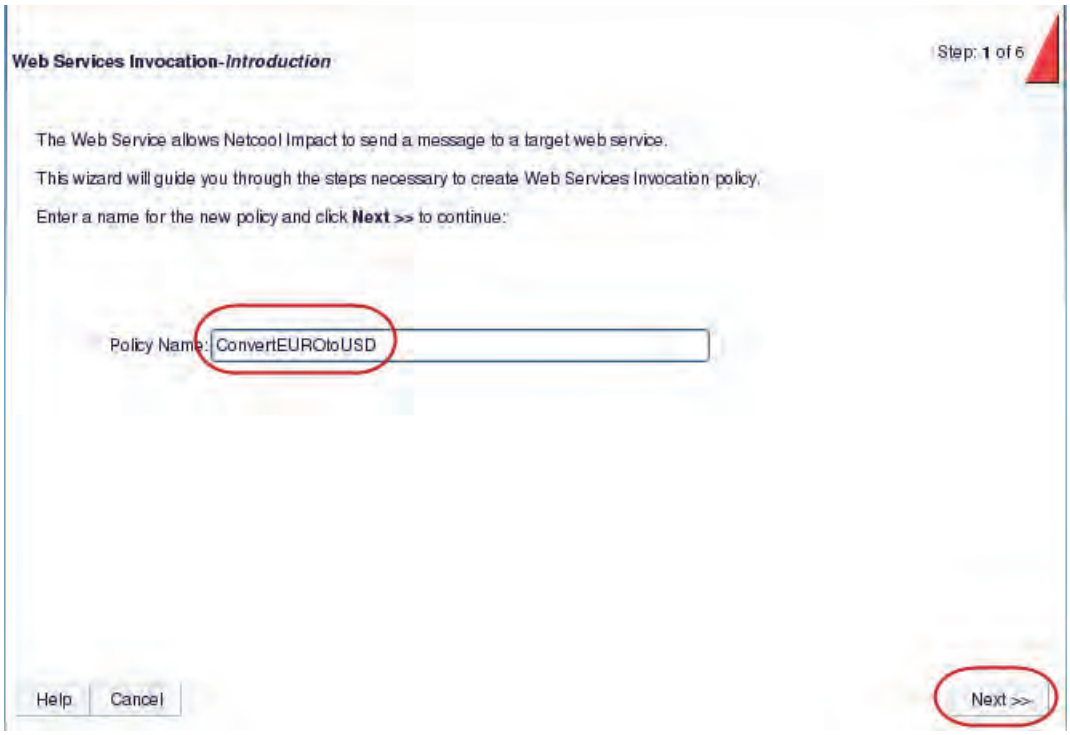
   a. Click **Policies** to display the policy editor screen.

   b. Click the **New Policy** icon . Scroll to **Use Wizard**.

   c. Click **Web Services.**



2. At the **Web Services Invocation** window, enter the policy name **ConvertEUROtoUSD**. Click **Next**.



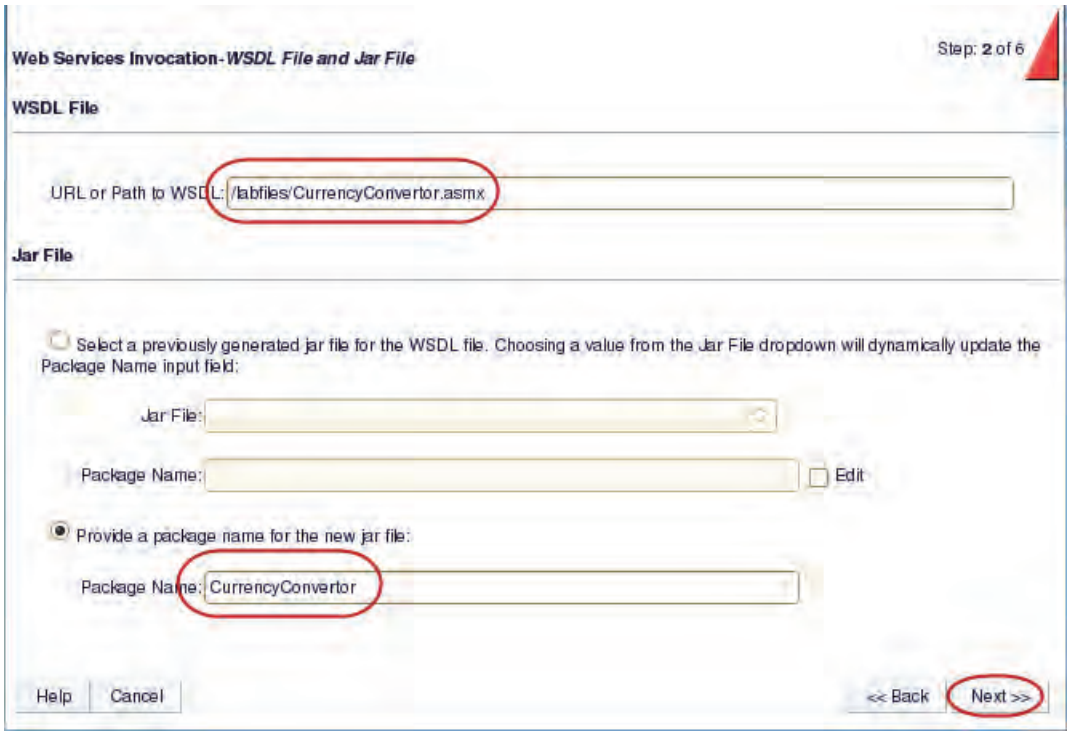
3. Follow these steps for the WSDL File and Client Stub configuration:
   a. At Step 2 under **WSDL File**, specify **the URL or Path to WSDL** as **/labfiles/CurrencyConvertor.asmx**.

      This file was downloaded to the local disk from the URL **http://www.webservicex.net/CurrencyConvertor.asmx?WSDL**.

      If the system has Internet connection, you can specify this URL at this step in the policy creation instead of pointing to the local copy.

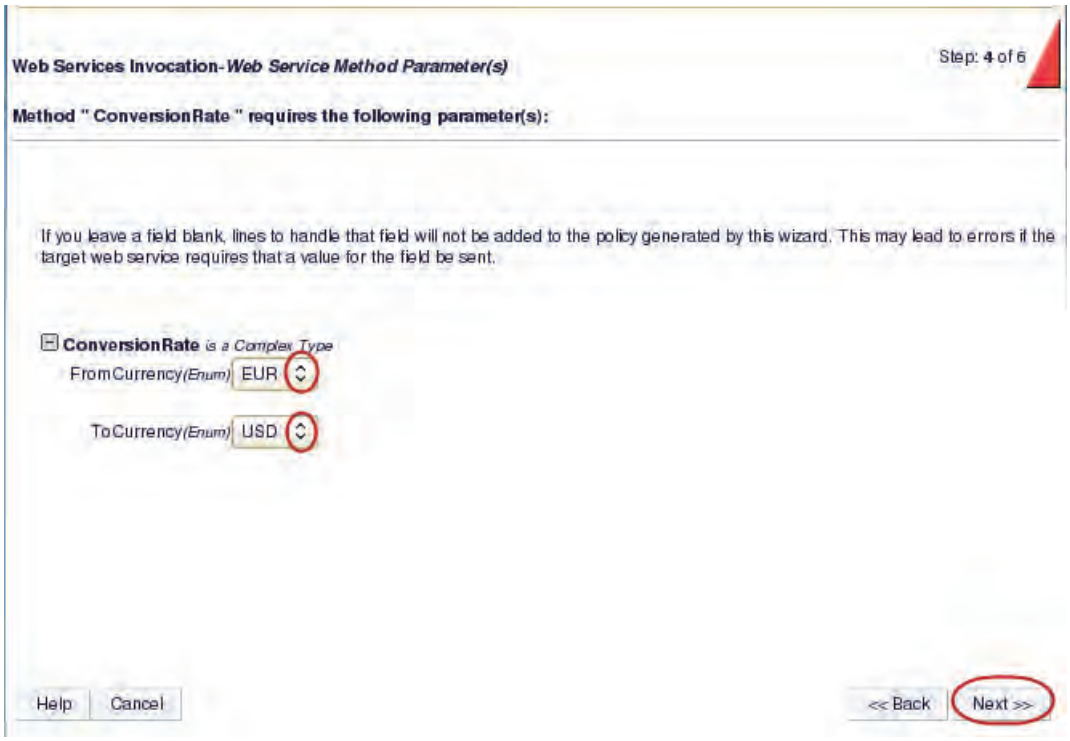
   b. Under **Client Stub**, select **Provide a package name for the new client stub**. Give it a package name of **CurrencyConvertor**. This name is the name of the JAR file that is built to

support the Web Services API for this particular **CurrencyConvertor** application. Click **Next**.



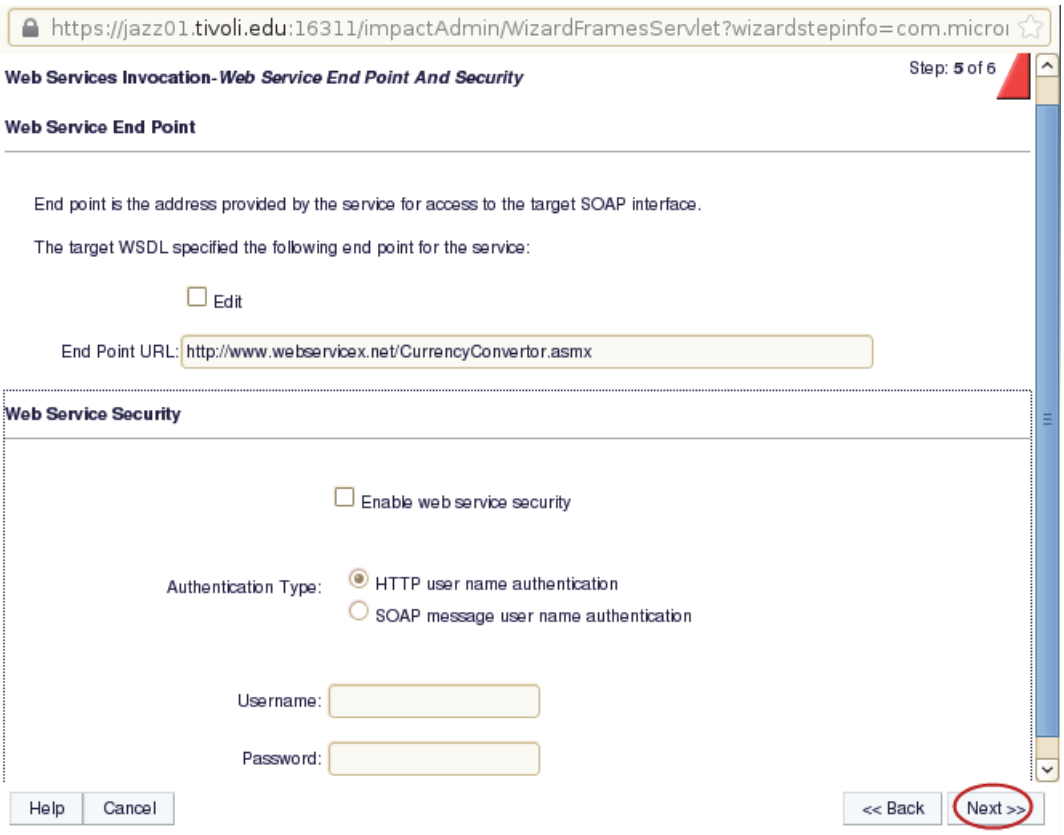
4. At Step 3 in the GUI, accept the default values for the **Web Service**, **Web Service Port Type**, and **Web Service Method**. Click **Next**.

5. At Step 4 in the GUI, expand **ConversionRate** and specify the **FromCurrency** and **ToCurrency** values. The hard-coded choices, which are taken from the WSDL file, are presented in the drop-down menus. Click **Next**.

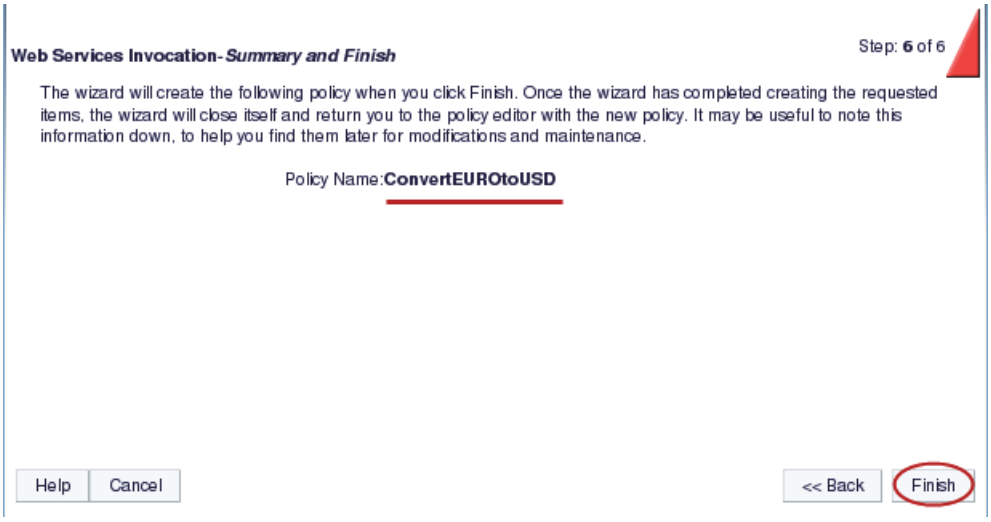


6. At Step 5 in the GUI, note the **End Point URL**, the access point of the target SOAP interface.

Do not change this entry. It was derived from the WSDL file that you designated in Step 2. Do not select **Enable web service security**. Click **Next.**



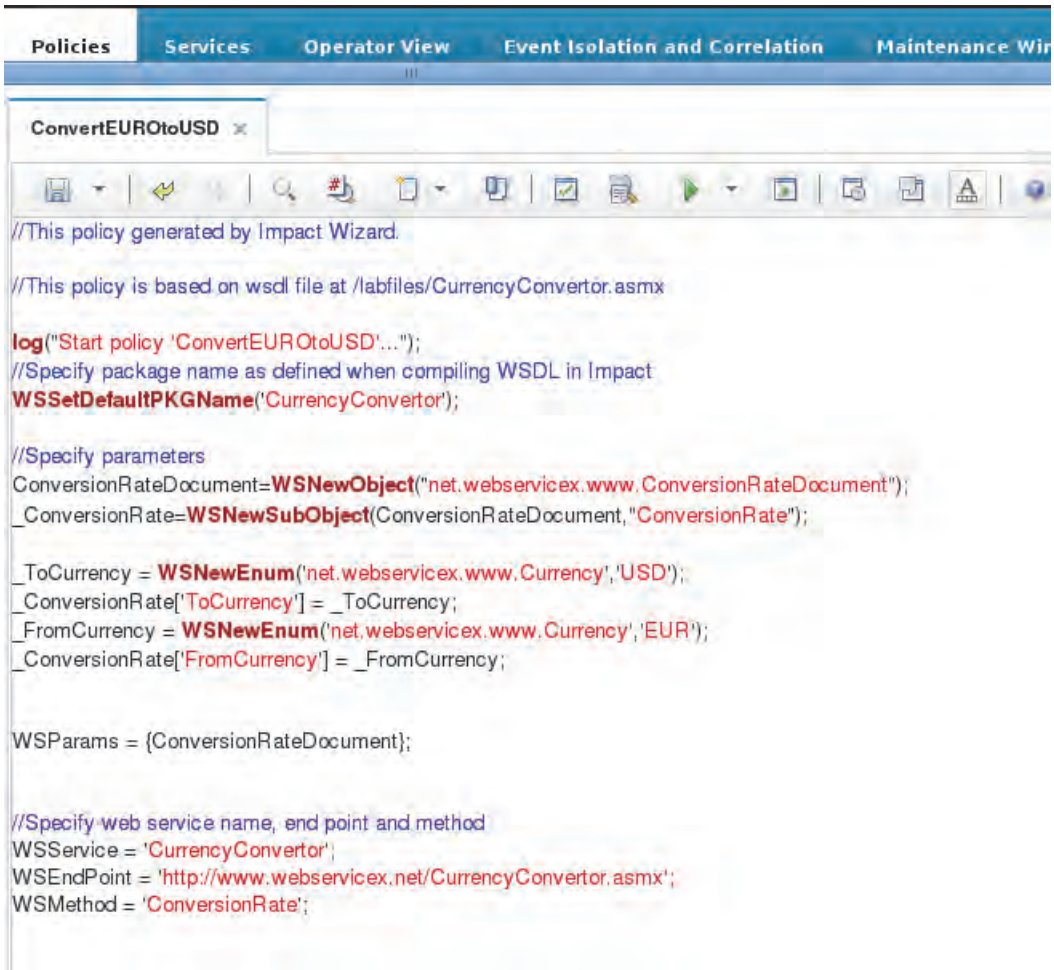
7. At the Summary and Finish screen, click **Finish**.



8. When the Wizard Complete screen opens, click **OK**.

9. If the screen is not visible or closes, at the bottom of the session, click **IBM Netcool/Impact - Mozilla Firefox** to redisplay the screen.

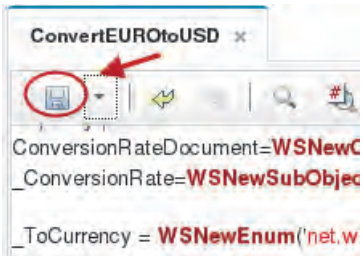
The policy that is displayed in the policy editor is like the following example.



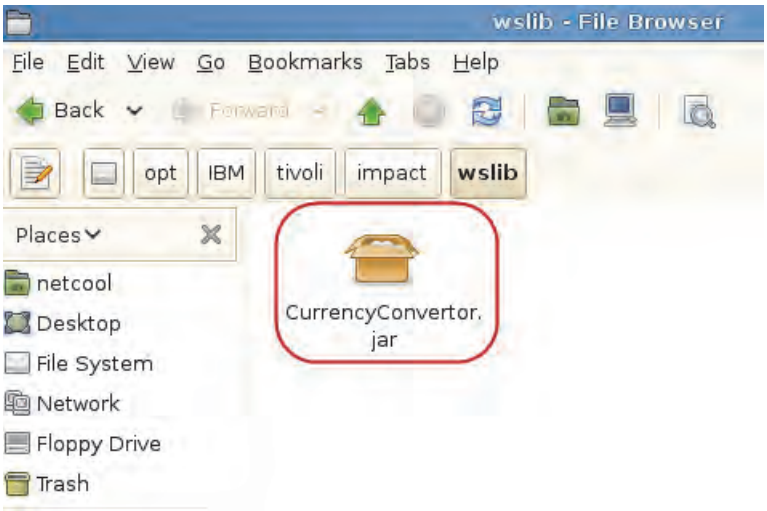
10. Examine the policy to see how the web services call **WSInvokeDL** was configured.

```
//Specify web service name, end point and method
WSService = 'CurrencyConvertor';
WSEndPoint = 'http://www.webservicex.net/CurrencyConvertor.asmx';
WSMethod = 'ConversionRate';


log("About to invoke Web Service call ConversionRate ......");

WSInvokeDLResult = WSInvokeDL(WSService, WSEndPoint, WSMethod, WSParams);
log("Web Service call ConversionRate return result: " +WSInvokeDLResult);
```

11. Save the policy created by the wizard. Click the **Save** icon.



The Web Services wizard created a JAR file and placed in **/opt/IBM/tivoli/impact/wslib/CurrencyConvertor.jar**.

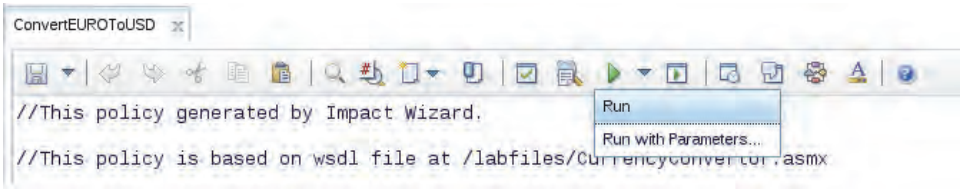


The Netcool/Impact server is configured by default to look in this location for application JAR files. They are automatically deployed into the Netcool/Impact server application.

Testing the policy

Test the new policy in the policy editor.

1. While editing the policy **ConvertEUROToUSD**, click the **Run** icon.



If an Internet connection is not available, the Error window opens.

2. Click the **Policy Log** icon to see your results. If the policy executed successfully, in the PolicyLogger you see the **ConversionRateResponse** from the **CurrencyConvertor** application, like the following example. The word **double** is replaced by the result of the conversion.

```
<soap:Envelope>
 <soap:Body>
 <ConversionRateResponse xmlins="http://www.webserviceX.NET/">
   <ConversionRateResult>double</ConversionRateResult>
 <ConversionRateResponse>
</soap:Body>
</soap:Envelope>
```

Note: Web Services can change over time. Be sure to verify that the soap response tags are the same.

Exercise 2 The web service response

In this exercise, for clarity you upload a second policy to parse the web service result.

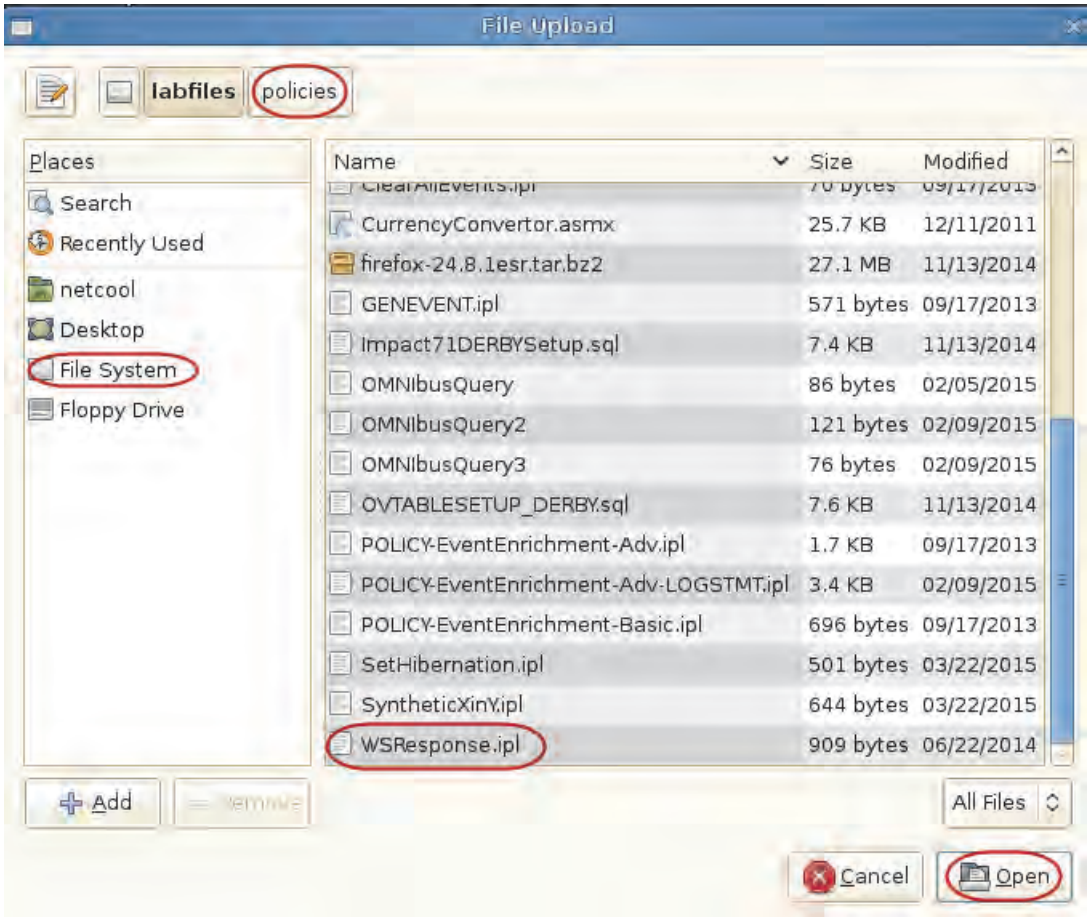
Parsing the response

In this case, the returned result contains one variable that is surrounded by several soap tags. To continue our testing without an Internet connection you can create a similar structure within a policy.

1. Click **Policies** to open the policy editor window.
2. Click the **Upload** icon.

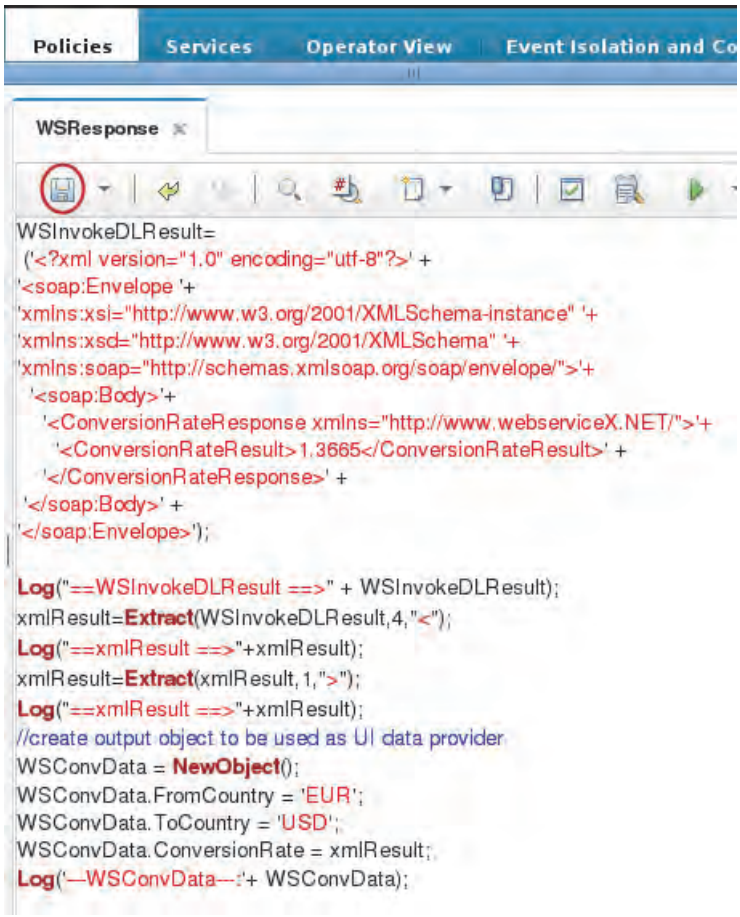
3. When the **Upload a Policy File** window opens, click **Browse.**

4. Highlight the **File System>labfiles** directory, select **WSResponse.ipl**, and click **Open.**



5. Click **Upload**.

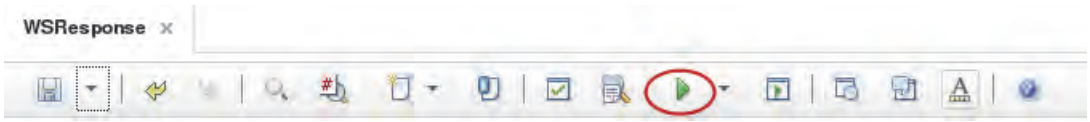
6. Double-click **WSResponse** in the left panel to open the policy. The resulting policy should look like the following screen captures. Click the **Save** icon.



This policy accomplishes three tasks:

- The **WSInvokeDLResult** field contains the soap response that would have been returned from the CurrencyConvertor web service.
- The Extract function is used to parse the web service result.
- The NewObject function is used to create UI provider data for Jazz for Service Management.

7. Click the **Run** icon to execute the policy.



8. Click **Close** in the **Information** window.

9. Click **View Policy Log** icon to view the results.

At the bottom of the log entry you should see the following message:

```
---WSConvData---:"Created by parser"=(ConversionRate=1.3665, ToCountry=USD,
FromCountry=EUR)
```

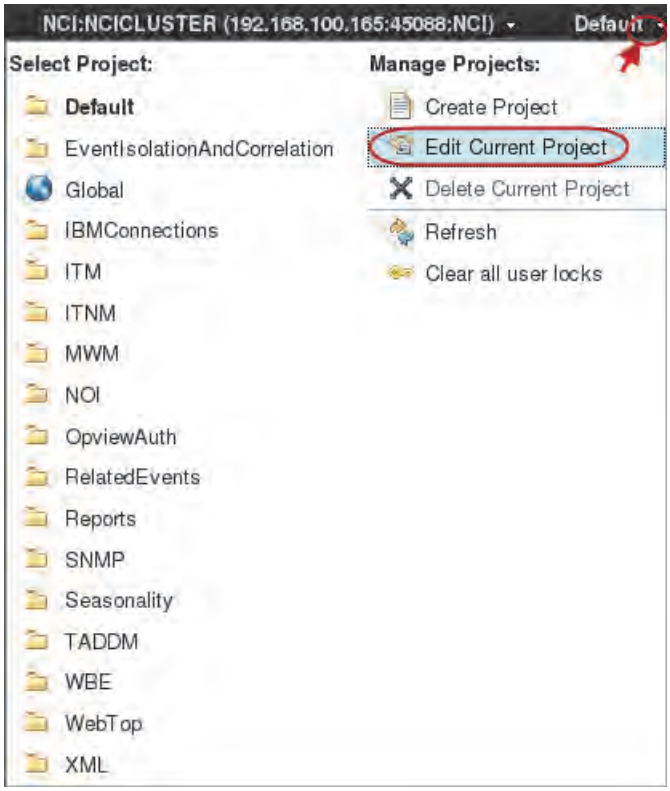
Unit 13 Hibernation, X in Y, and synthetic events exercises

This exercise is an introduction to advanced topics such as Hibernation, X in Y processing, and Synthetic Events.

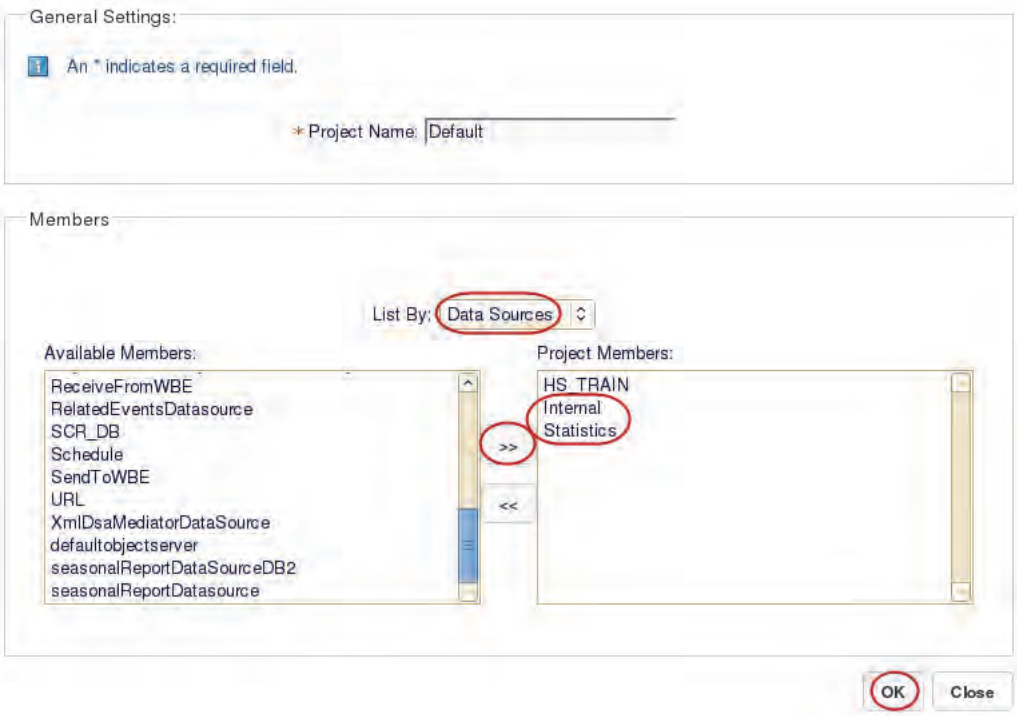
Exercise 1 Hibernations

In this exercise, you experiment with hibernations.

1. Edit the **Default** project.

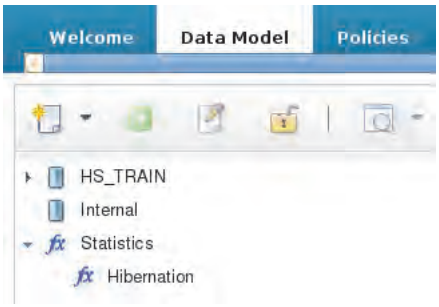


2. Add the **Internal** and **Statistics** Data Sources to the **Project Members** list. Click **OK**



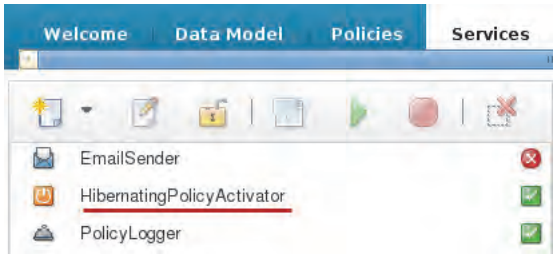
3. Select **Statistics** as the Data Source, List By **Data Types**, and move the **Hibernation** data type to the **Project Members**. Click **OK**.

The result should be similar to this display.

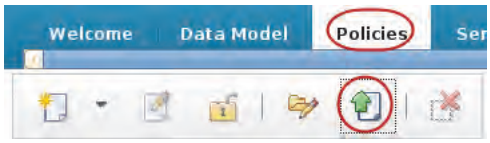


4. Select **Services** and add the **HibernatingPolicyActivator** from the Global Repository to the Project Members list. Click **OK**.

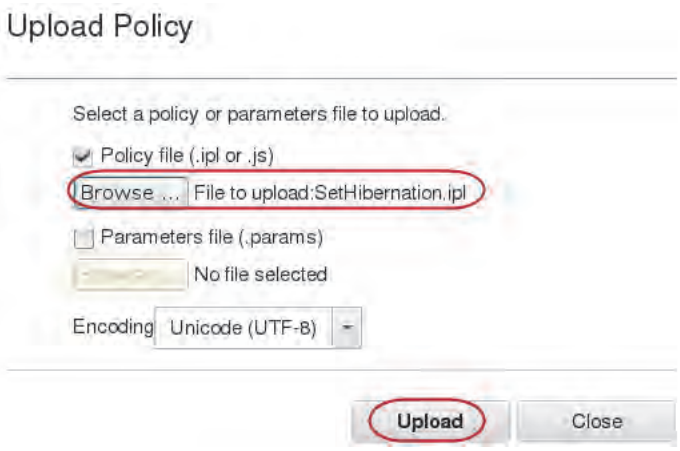
The result should be similar to this display.



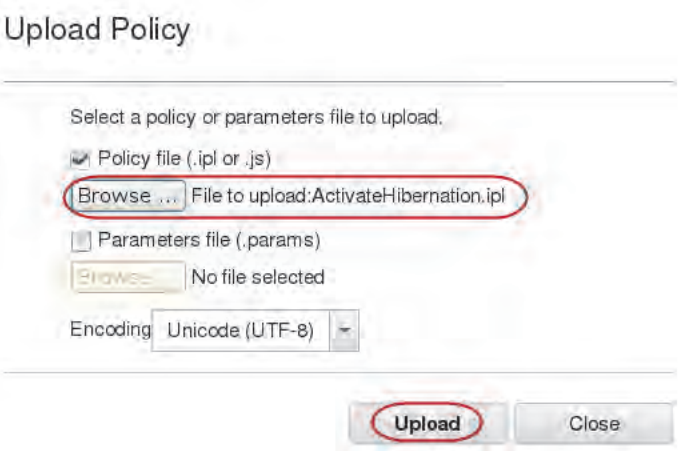
5. Click the **Policies** tab and use the policy uploader to import the **SetHibernation.ipl** and **ActivateHibernation.ipl** policies from the **labfiles** directory.



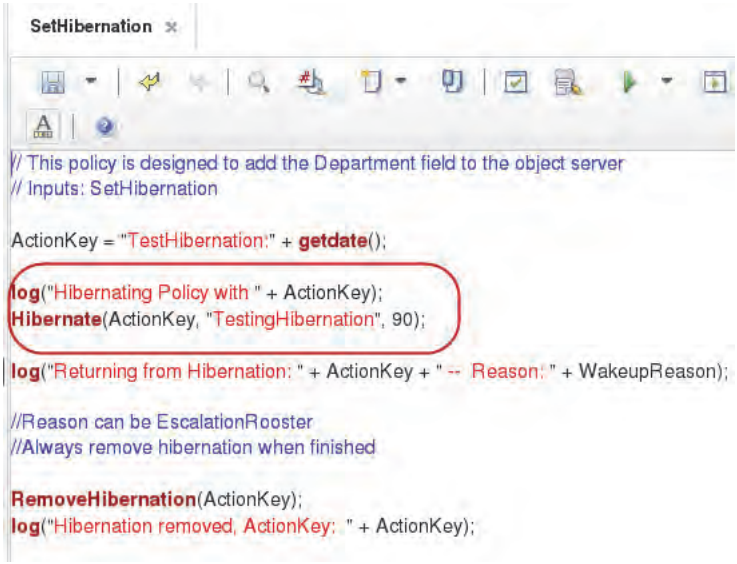
SetHibernation.ipl



ActivateHibernation.ipl

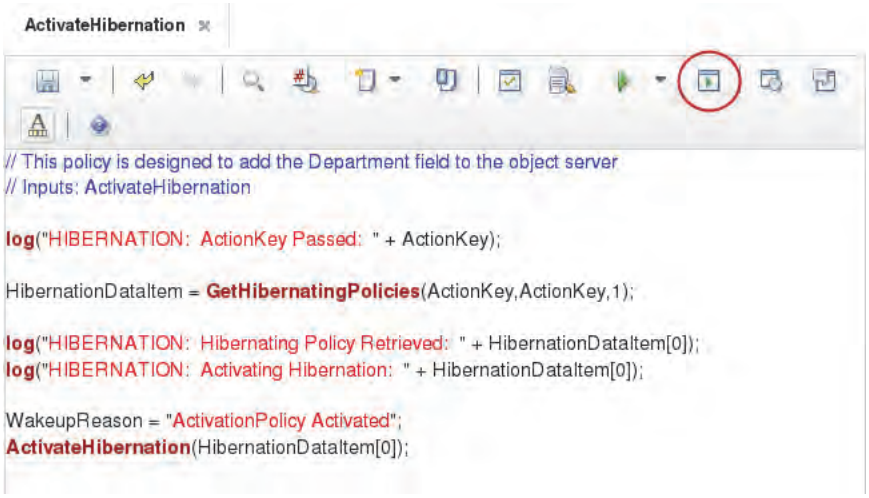


6. The **SetHibernation** policy hibernates a policy uniquely identified by the **ActionKey** for 90 seconds. Verify this parameter in the policy where shown.



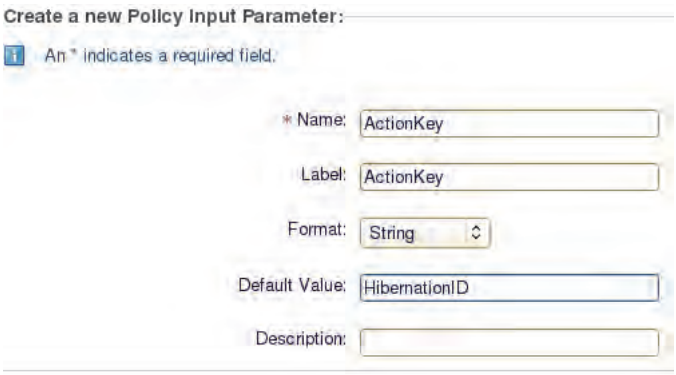
Hibernations are removed from hibernation in one of several ways: the user removes it through the GUI action (Wakeup Hibernation), the hibernation period expires, or the **ActivateHibernation** policy is triggered. The reason or method for waking up from hibernation is noted in the PolicyLogger log.

7. To test the **ActivateHibernation** policy, click the **Configure Policy Settings** icon.



8. Under **Policy Input Parameters** click the **New** button.

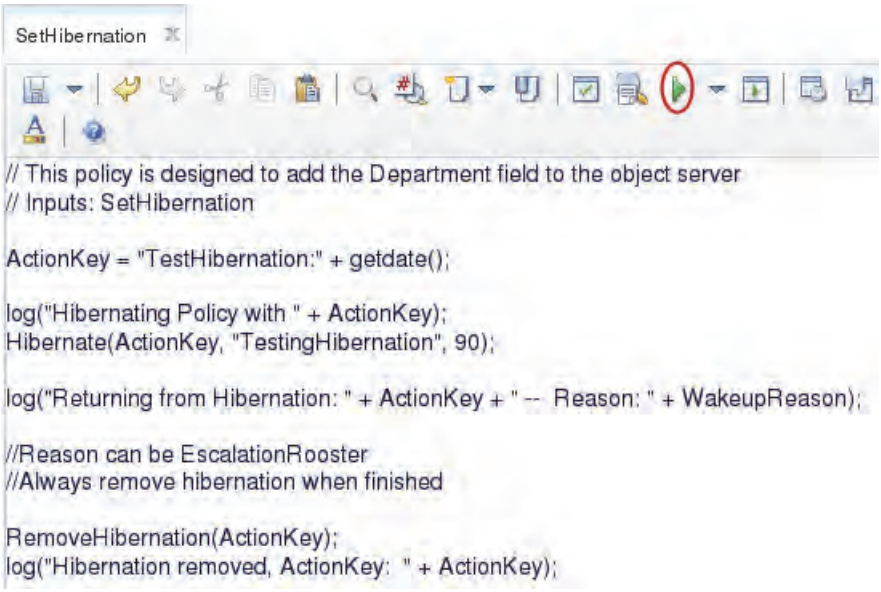
9. Specify the new runtime parameter **ActionKey** as shown, and click **OK**.



The resulting entry looks similar to the following example.

10. Scroll to the bottom to click **OK**.

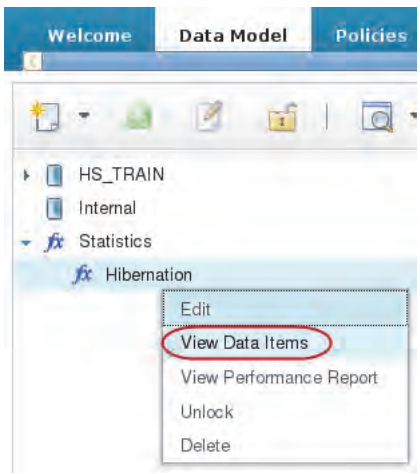
11. Open the **SetHibernation** policy and run the policy five or six times.



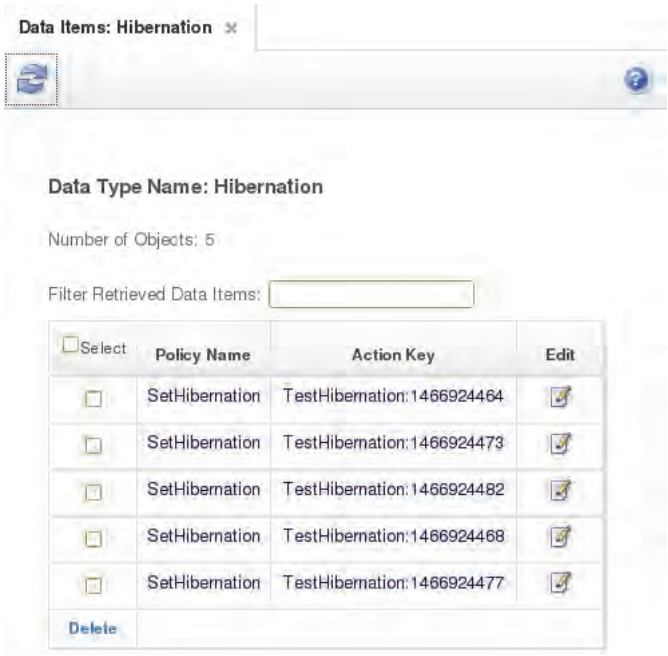
12. Log entries will look similar to the display:

```
June 26, 2016 6:55:24 AM UTC[PolicyLogger][SetHibernation][pool-3-thread-119]Parser log: Hibernating Policy with TestHibernation:1466924124
June 26, 2016 6:56:13 AM UTC[PolicyLogger][SetHibernation][pool-3-thread-119]Parser log: Hibernating Policy with TestHibernation:1466924173
June 26, 2016 6:56:19 AM UTC[PolicyLogger][SetHibernation][pool-3-thread-119]Parser log: Hibernating Policy with TestHibernation:1466924179
June 26, 2016 6:56:24 AM UTC[PolicyLogger][SetHibernation][pool-3-thread-119]Parser log: Hibernating Policy with TestHibernation:1466924184
June 26, 2016 6:56:29 AM UTC[PolicyLogger][SetHibernation][pool-3-thread-119]Parser log: Hibernating Policy with TestHibernation:1466924189
June 26, 2016 6:56:35 AM UTC[PolicyLogger][SetHibernation][pool-3-thread-119]Parser log: Hibernating Policy with TestHibernation:1466924195
June 26, 2016 6:56:44 AM UTC[PolicyLogger][SetHibernation][pool-3-thread-119]Parser log: Hibernating Policy with TestHibernation:1466924204
June 26, 2016 6:56:55 AM UTC[PolicyLogger][SetHibernation][pool-3-thread-119]Parser log: Returning from Hibernation: TestHibernation:1466924124
-- Reason: EscalationRooster
June 26, 2016 6:56:55 AM UTC[PolicyLogger][SetHibernation][pool-3-thread-119]Parser log: Hibernation removed,
ActionKey: TestHibernation:1466924124
```

13. View the data items for the **Hibernation** data type and look at the hibernations just started. You may have to run the SetHibernation policy a few more times to catch entries here.

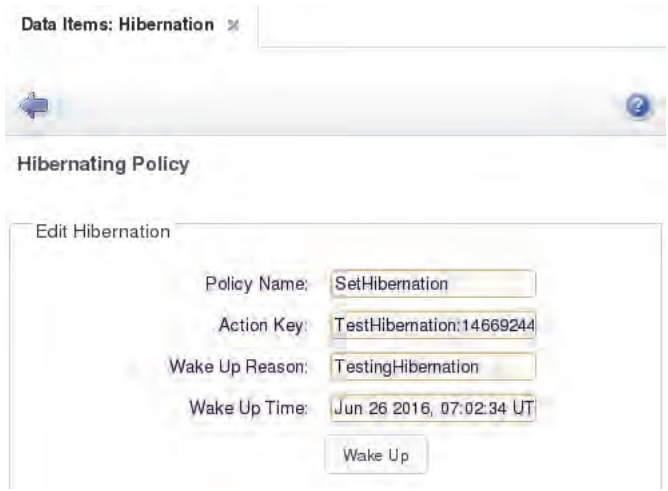


14. Click the pencil icon to edit one of the hibernations.



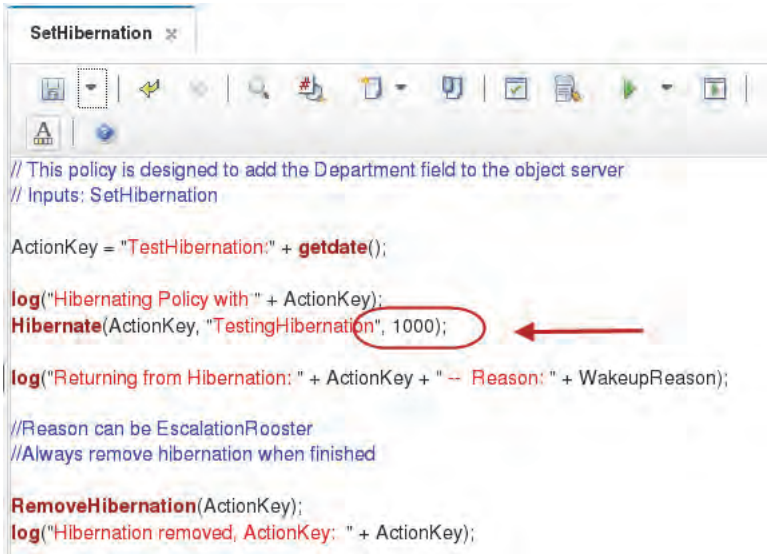
15. Write down the **Action Key** information for the hibernation you selected.

_____

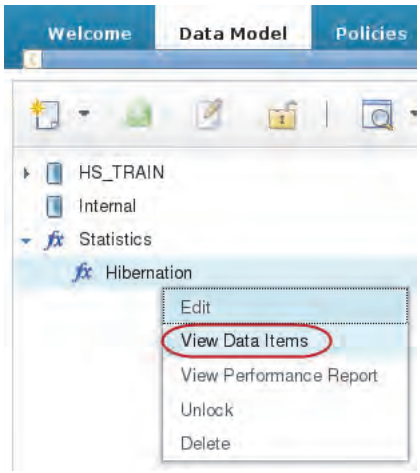


16. Review the log find the entries related to the Action Key that you selected.The reason for the return from hibernation is **EscalationRooster**.

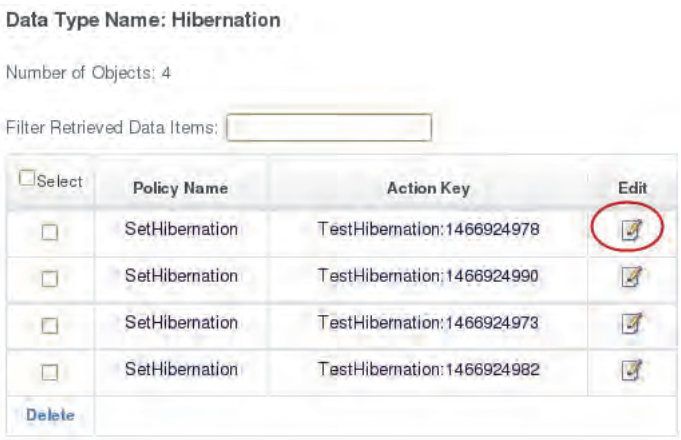
17. Modify the **SetHibernation** policy and replace the time value with **1000**. Check the syntax, and save the policy. Set new hibernations by clicking the green arrow 5 or 6 times.



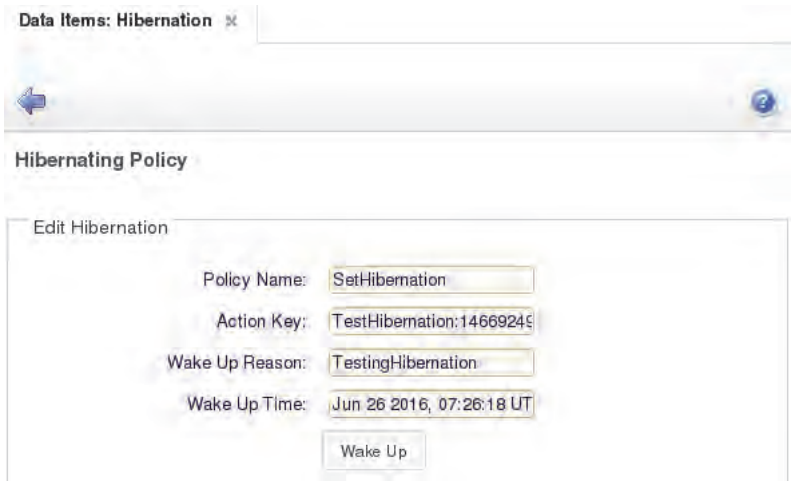
18. View the data items for the **Hibernation** data type and look at the new hibernations.



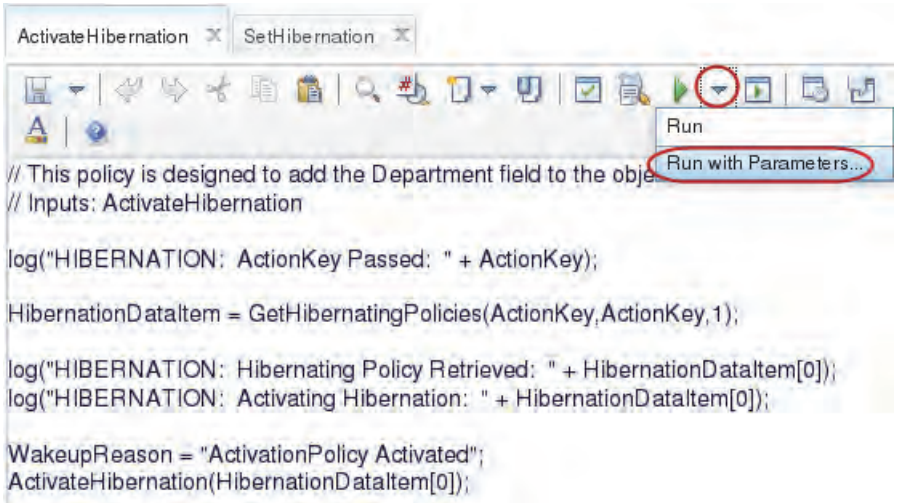
19. Select and edit a hibernation.



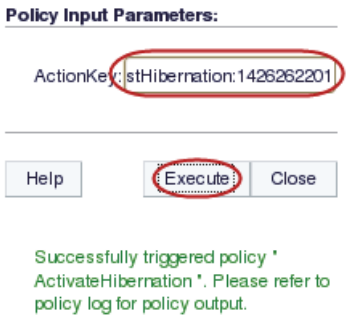
20. Write down the **Action Key** information.



21. Edit the **ActivateHibernation** policy and select **Run With Parameters**.

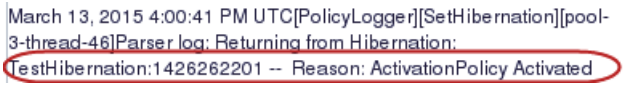


22. Enter the **ActionKey** that you recorded earlier, including **TestHibernation:**. Click **Execute** to run the policy. Click **Close** when complete.



23. View the PolicyLogger log. Find the entries for the preceding action.

Note: The reason for **Returning from Hibernation** is logged as **ActivationPolicy Activated** which was set in the **ActivateHibernation** policy.



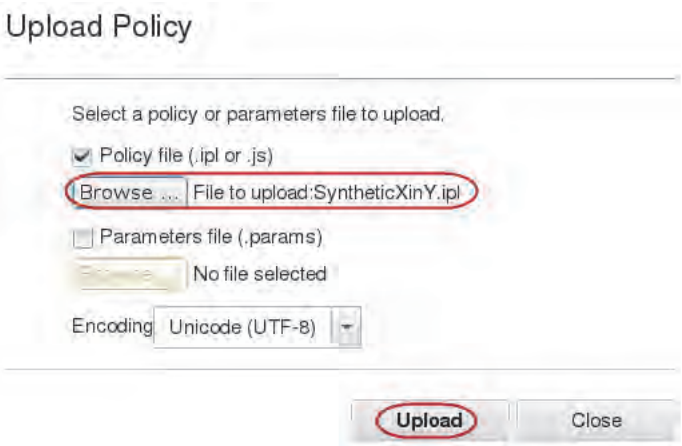
Eventually the hibernation interval for the remaining hibernations expires. When this interval expires, the log file entries indicate that the reason for returning from hibernation is **EscalationRooster.**

After a few minutes, the **Data Item** list for the data type **Hibernation** no longer lists entries.

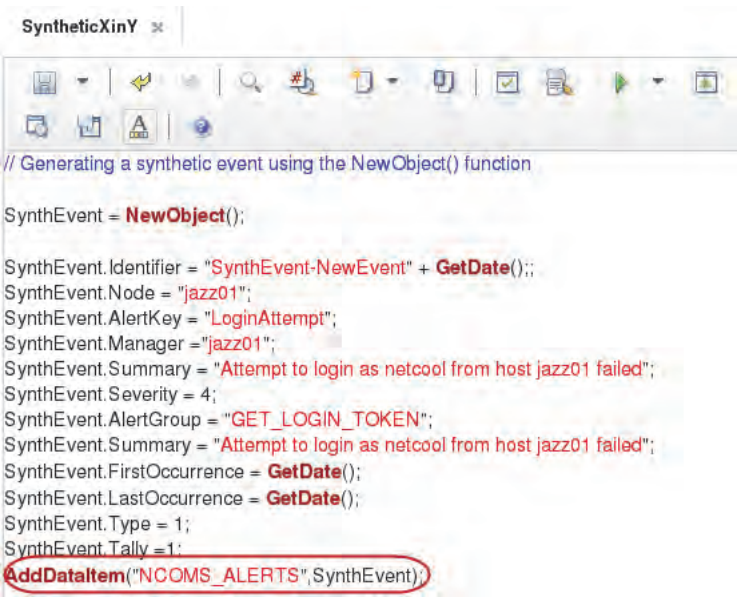
Exercise 2 Using a policy to create synthetic events

In this exercise, you use a policy to create synthetic events in the **alerts.status** table in OMNIbus.

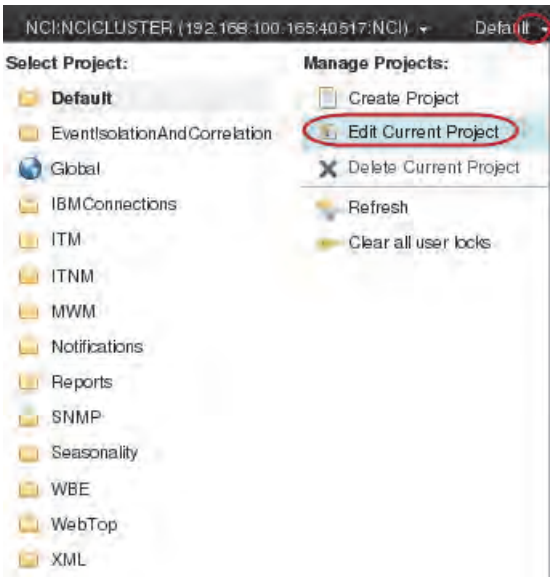
1. Use the **Policy Uploader** to upload the **SyntheticXinY.ipl** policy from the **labfiles** directory.



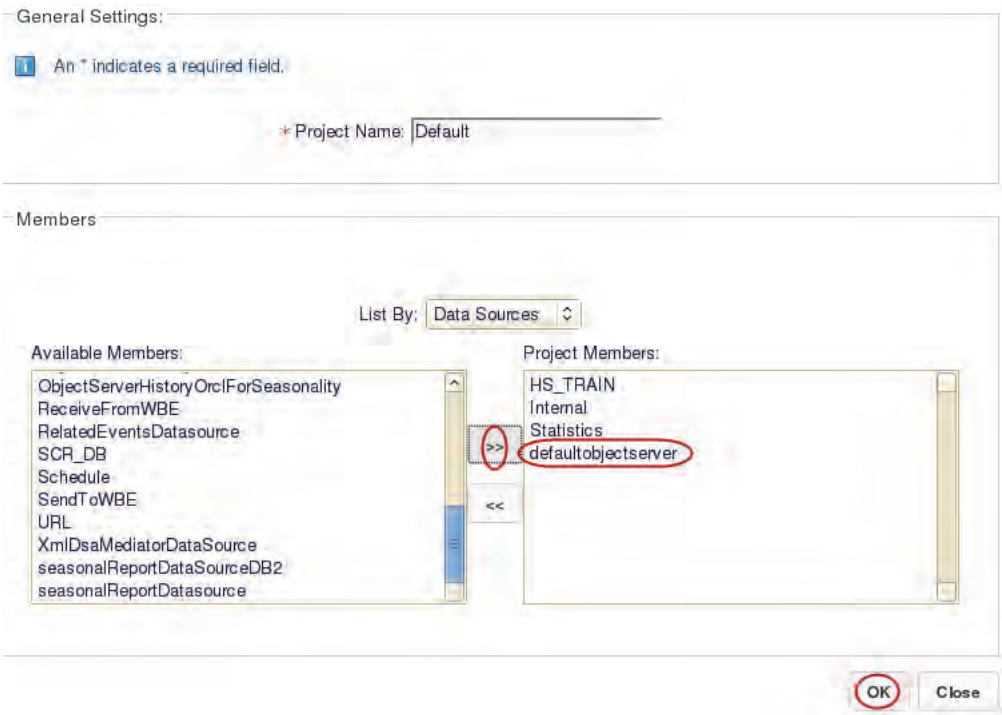
2. Edit the policy in the policy window. Examine the field settings for the **SynthEvent NewObject** object and the **AddDataItem** function. Click **Save**.



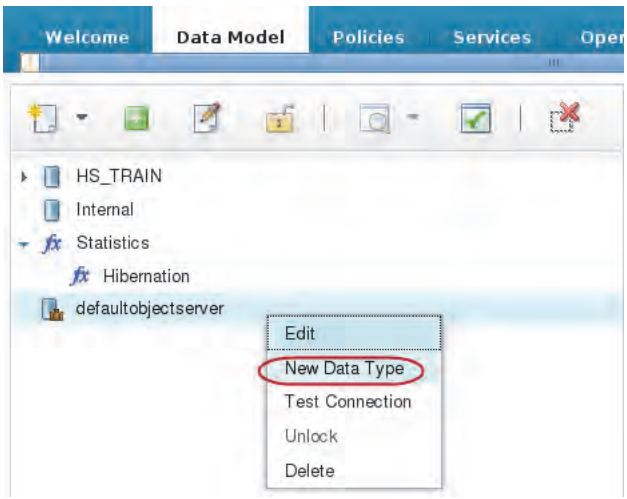
3. Click **Data Model** to open the **Data Model** tab. If **defaultobjectserver** is not listed, add it to the **Default** Project.
   a. Edit the **Default** project.



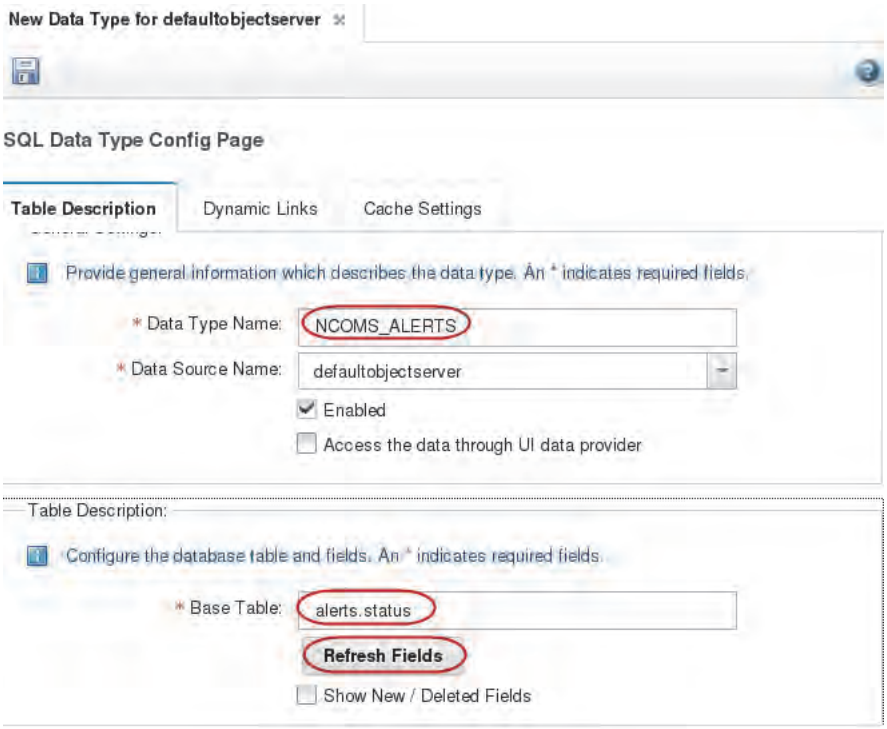
   b. Select List By **Data Sources**, and add the **defaultobjectserver** to the Project Members. Click **OK**.



4. Right-click the **defaultobjectserver** entry in the list. Select **New Data Type.**

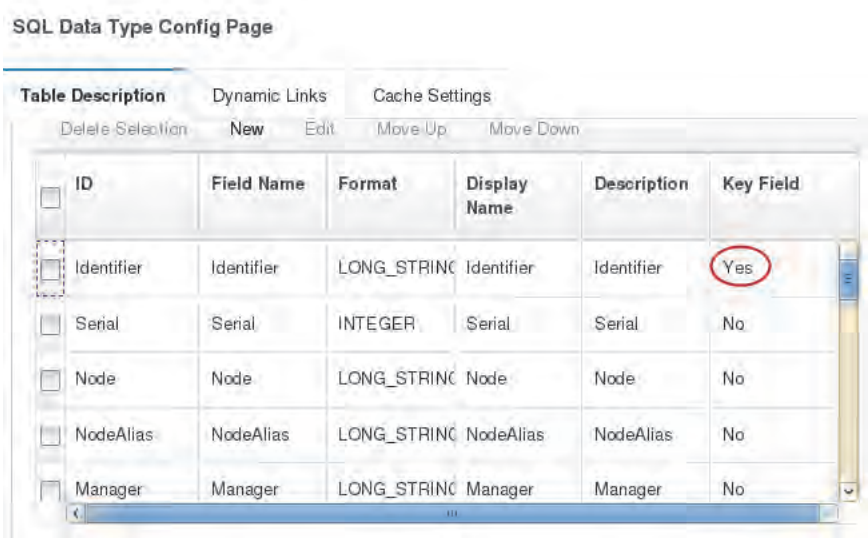


5. Create the **NCOMS_ALERTS** data type as depicted in the following screen capture. The **Base Table** entry is **alerts.status.**



Note: Remember to click the **Refresh** button.

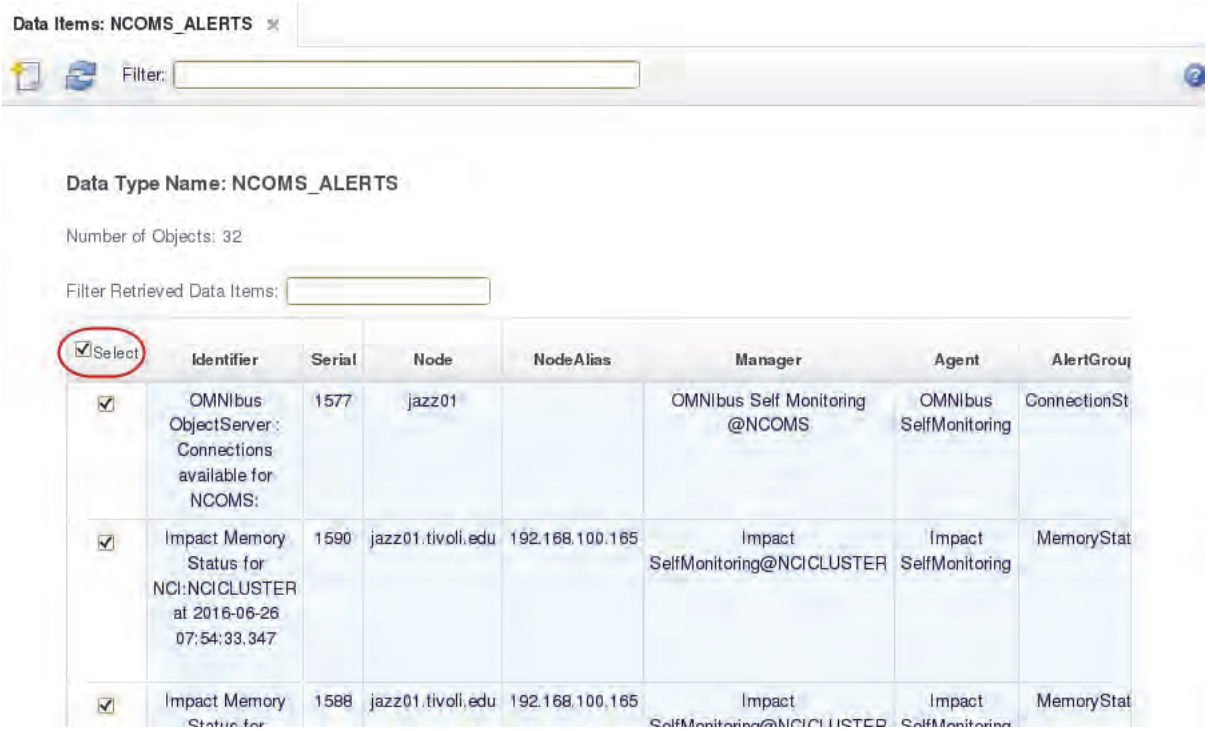
6. Scroll down to display the schema. Select Identifier as the Key Field.



7. Click **Save**. Close the window.

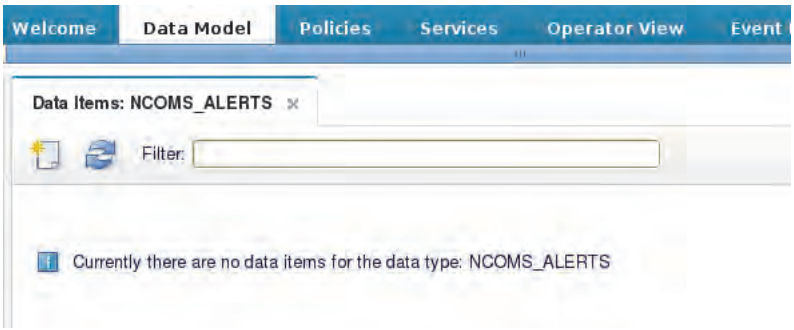
Exercise 3 Testing synthetic events

1. Click **View Data Items** for the **NCOMS_ALERTS** data type. Click **Select**. Scroll to the bottom and to click **Delete** to remove all items from the list.

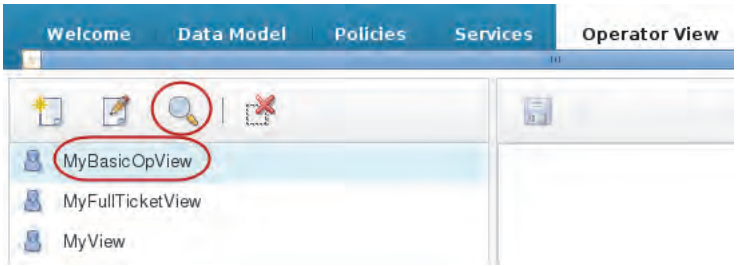


The resulting message opens, as shown in the example. If events continue to show, turn off the Simnet Probe.

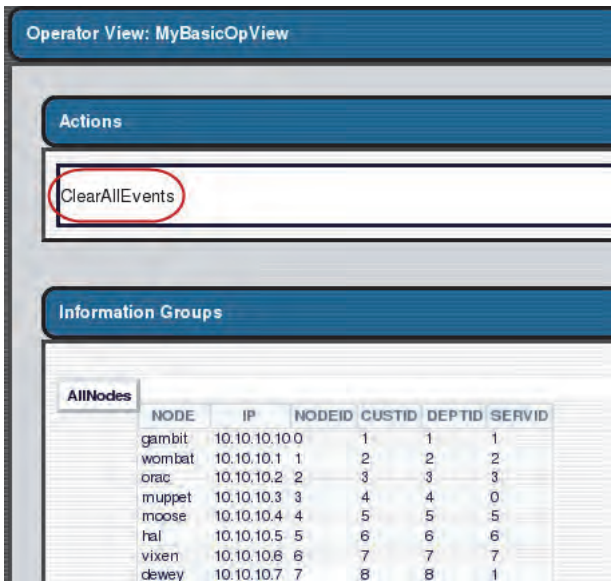
Note: To stop the Simnet Probe, locate the processid for *nco_p_simnet*. Use `ps -ef | grep nco_p_simnet` as described earlier. Issue the **kill <*processid*>** command to stop the process.



2. **MyBasicOpView** can also be used to delete object server events. Click the Operator View tab. Single click to highlight **MyBasicOpView**.
3. Click the hour glass to display the operator view.



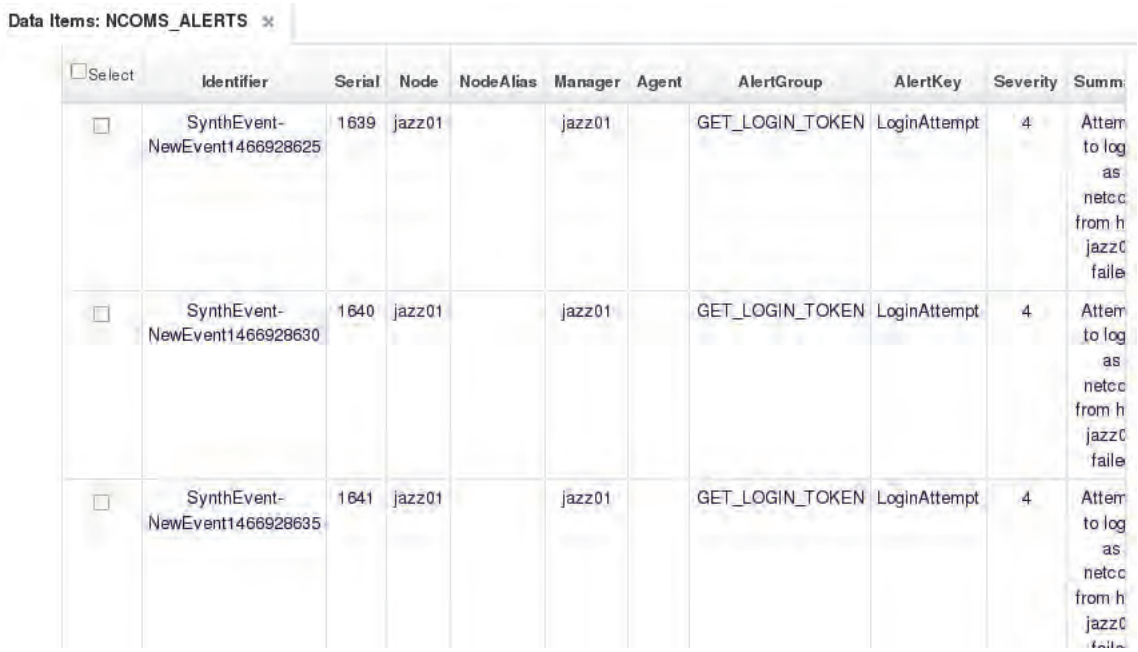
4. Click the Action **ClearAllEvents**.



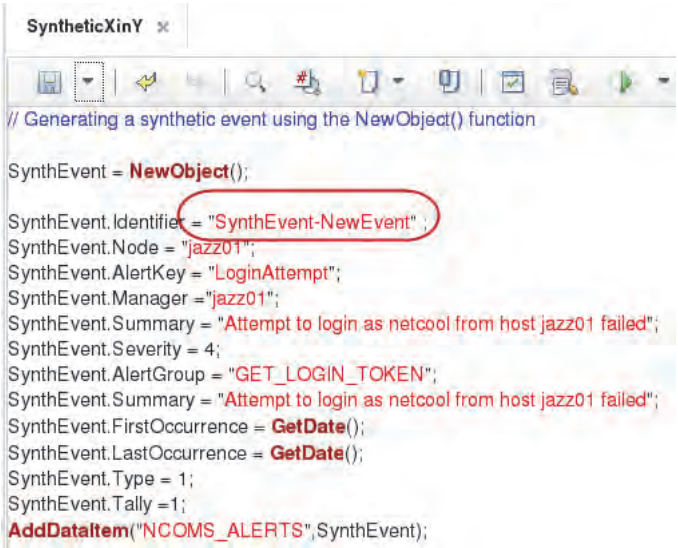
5. Click **Ok** to continue.

6. Edit the **SyntheticXinY.ipl** policy.

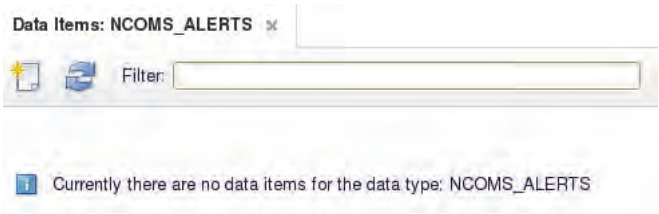
7. Click the green arrow to execute the **SyntheticXinY.ipl** policy three times. **View Data Items** again to see the new **NCOMS_ALERTS** events. Click the **Refresh** icon if events do not immediately show in the list.



8. Navigate back to the **Policies** window. Edit the **SyntheticXinY.ipl** policy and remove the **GetDate()** function from the **SynthEvent.Identifier** line as shown in the example. Click the **Check Syntax** icon. Click **Save**.

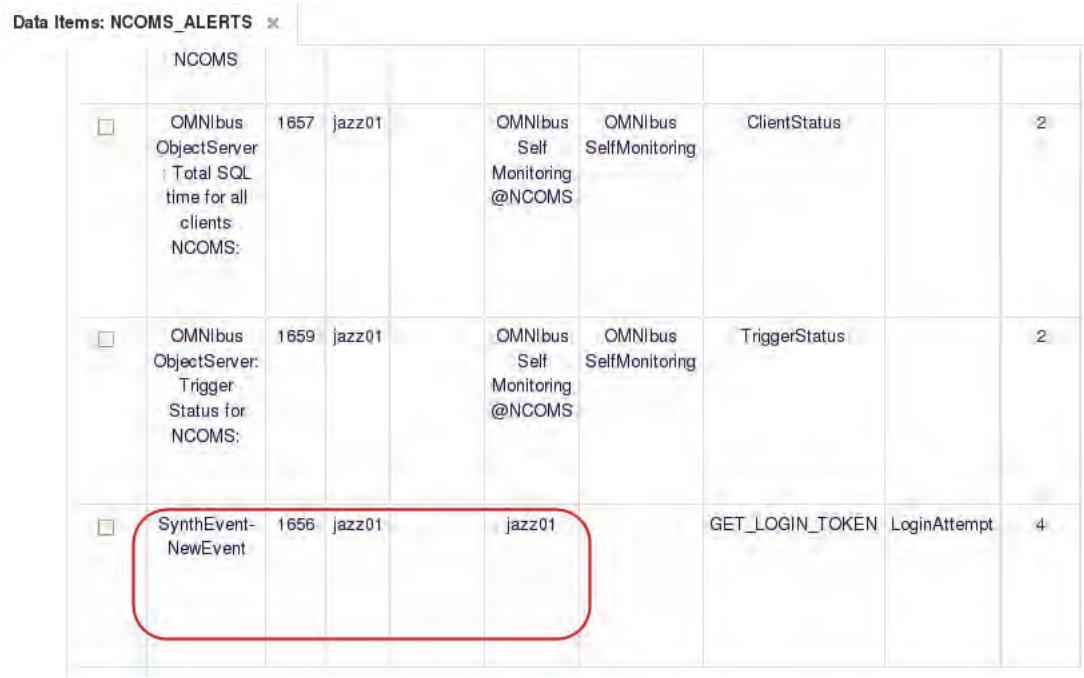


9. Clear the events in the **NCOMS_ALERTS** table.



10. Click the **Run** icon to execute the **SyntheticXinY.ipl** policy four times.
11. Click **View Data Items** to see the new **NCOMS_ALERTS** events.

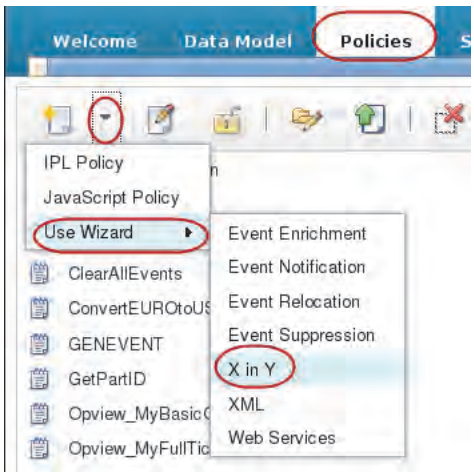
Hint: With deduplication, only one Synthetic event is displayed. The **GetDate** function created unique entries before it was removed.



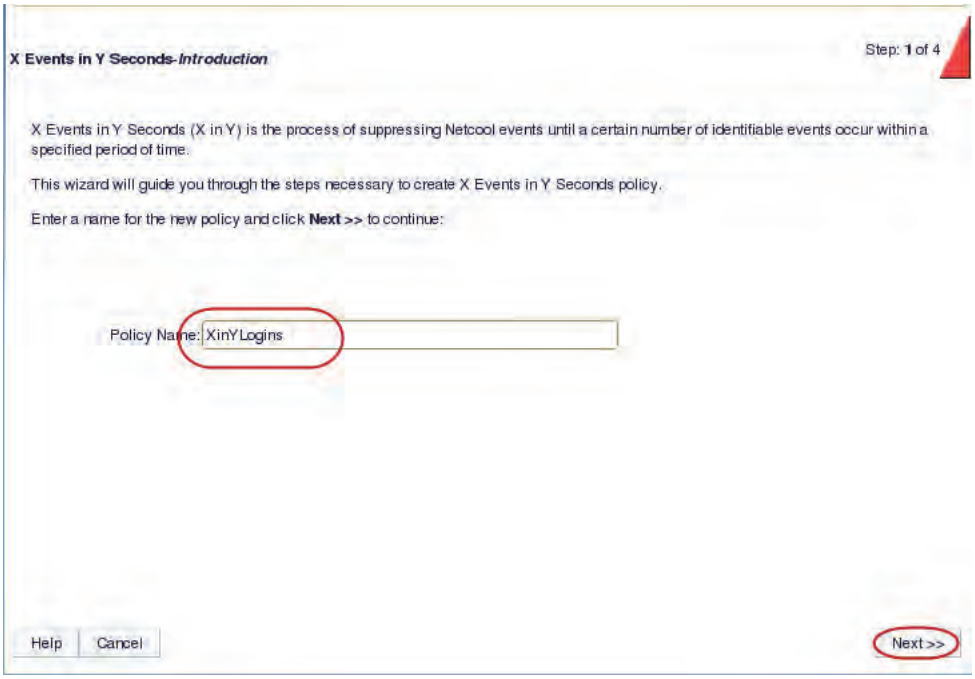
Exercise 4 X in Y policy

X in Y policies monitor the number of occurrences of an event within a sliding period of time. This policy detects five failed ObjectServer attempts that occur within 30 seconds. When the 30-second threshold is exceeded, the event summary is updated to indicate that the threshold is exceeded.

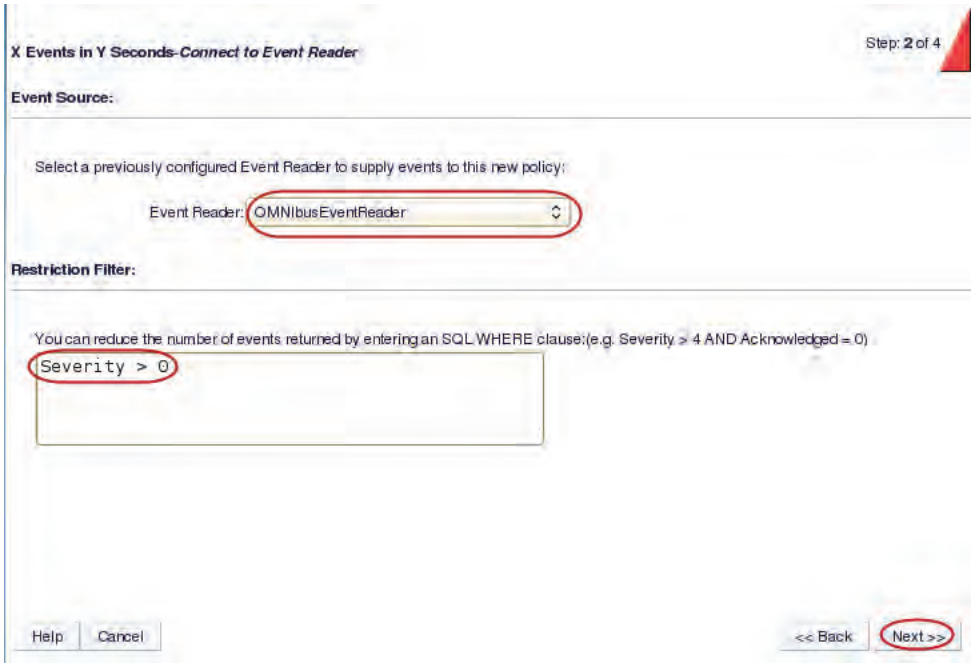
1. Click **Policies** to open the **Policies** tab. Click the **New Policy** icon. Select **Use Wizard > X in Y**.



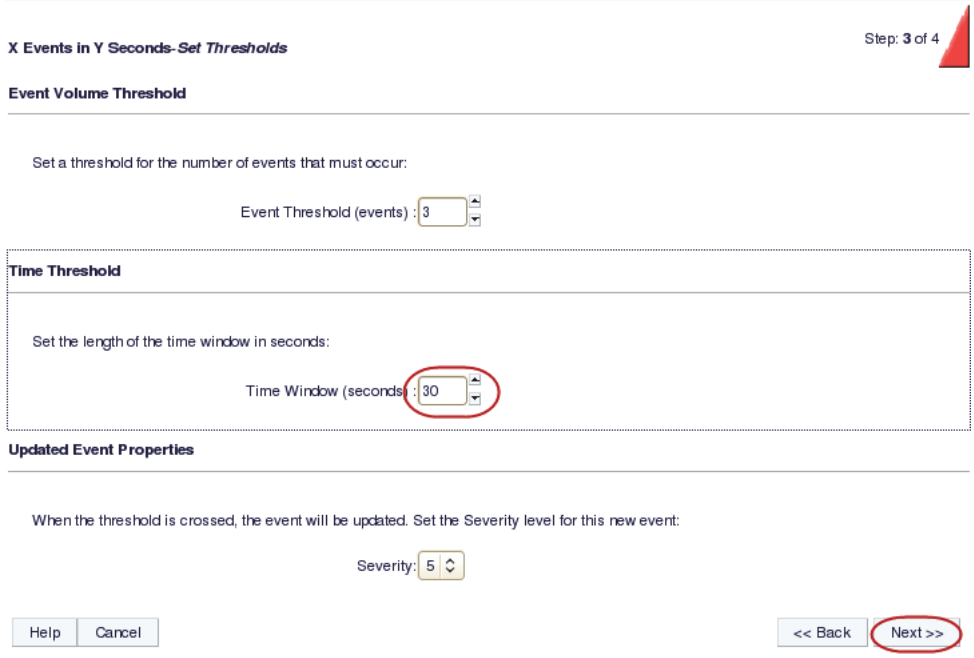
2. In the **Policy Name** field, type the value **XinYLogins**. Click **Next.**



3. Select **OMNIbusEventReader as the** event reader. In the **Restriction Filter** field, type **Severity > 0.** Click **Next.**

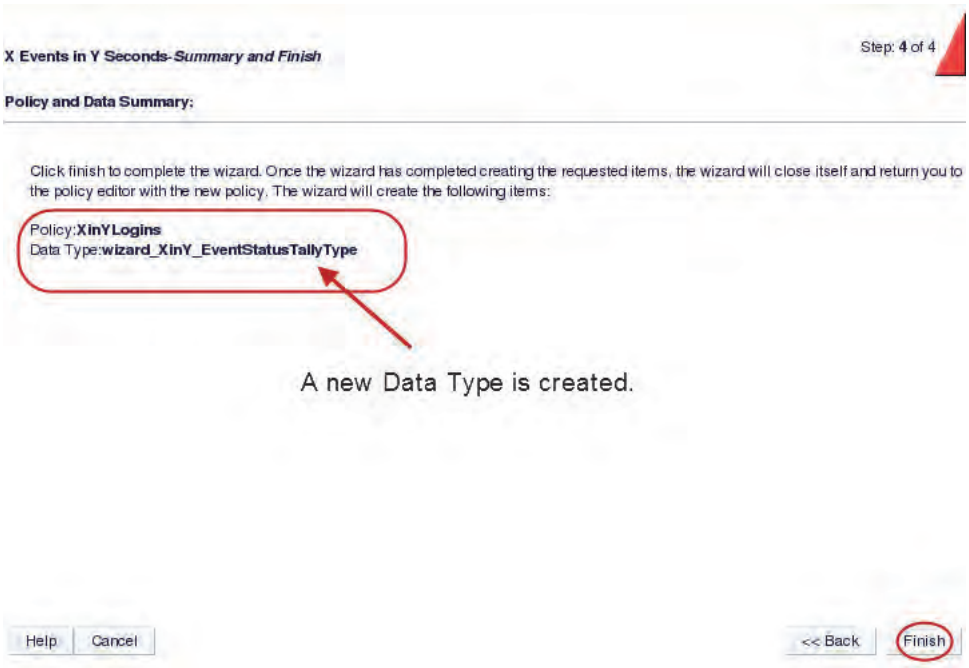


4. Set the **Event Threshold** to **3**. Set the **Time Window** to **30**. Click **Next**.

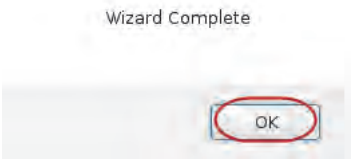


Notice the data type automatically created for the policy to use.

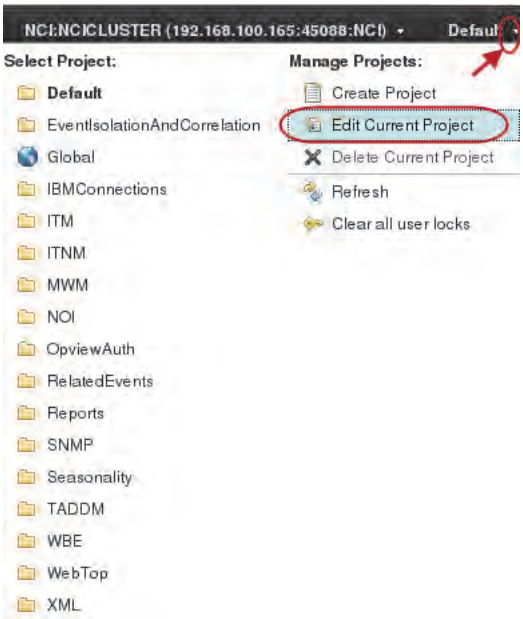
5. Click **Finish**.



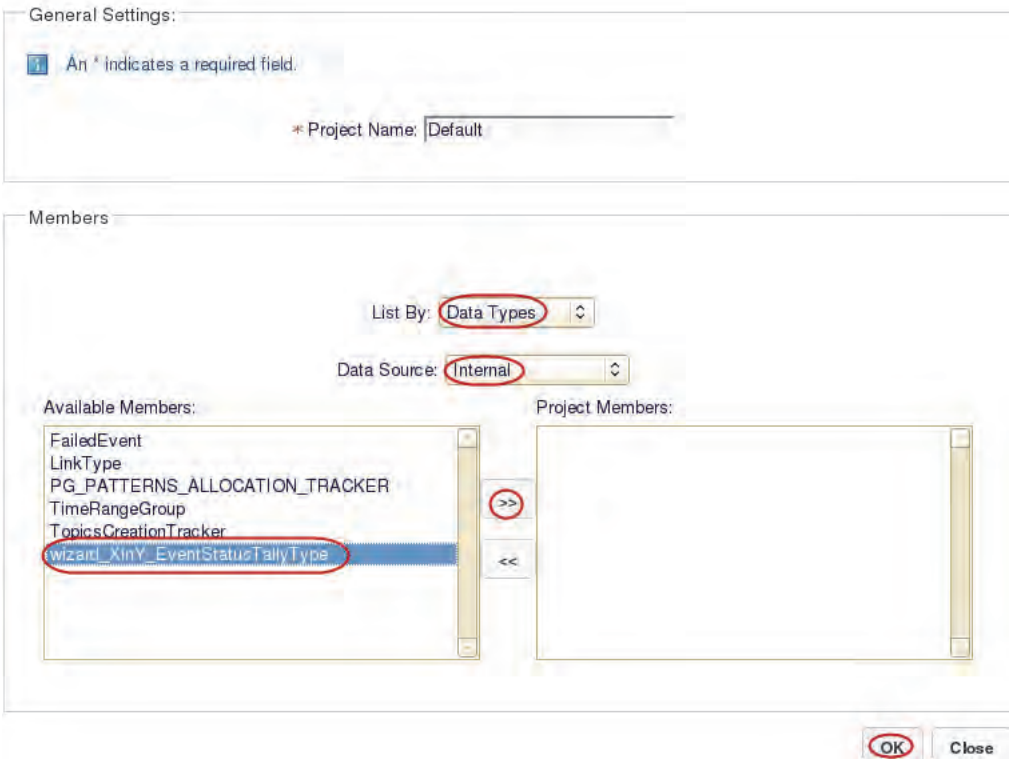
6. Click **OK** to continue.



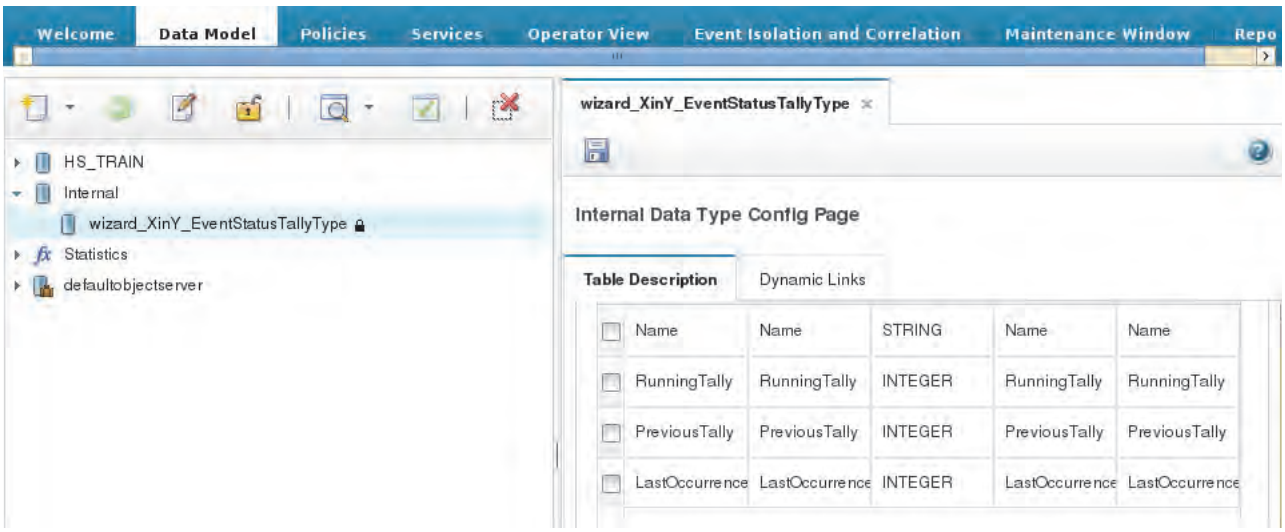
7. Edit the **Default** project.



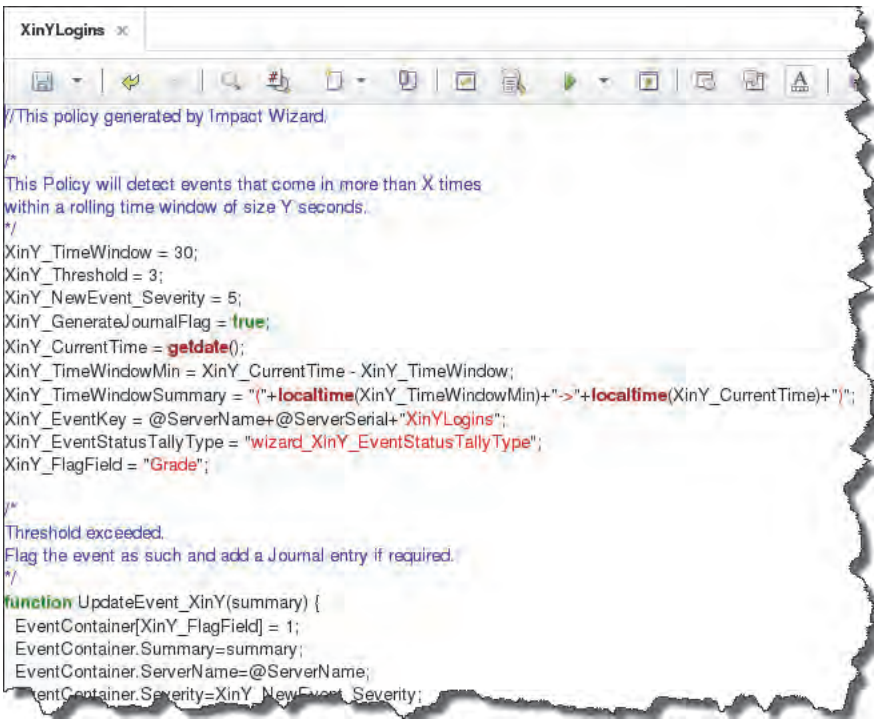
8. Add the **wizard_XinY_EventStatusTallyType** Data Type created by the wizard to the **Internal** Data Source as shown in the example.



9. Select the **Internal** Data Source and examine the **wizard_XinY_EventStatusTallyType** Data Type.



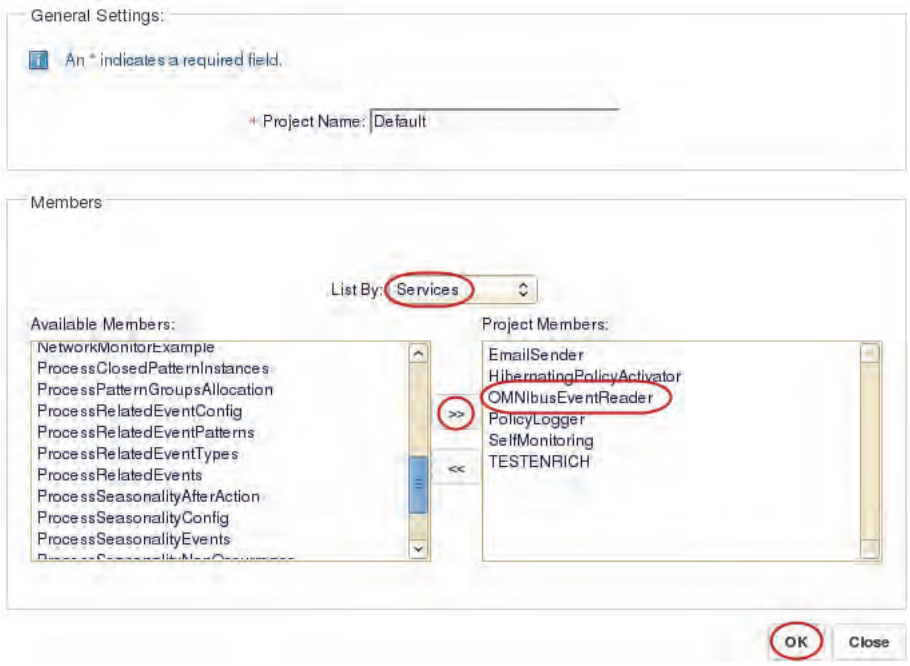
10. View the policy **XinYLogins** in the **Policies** window.



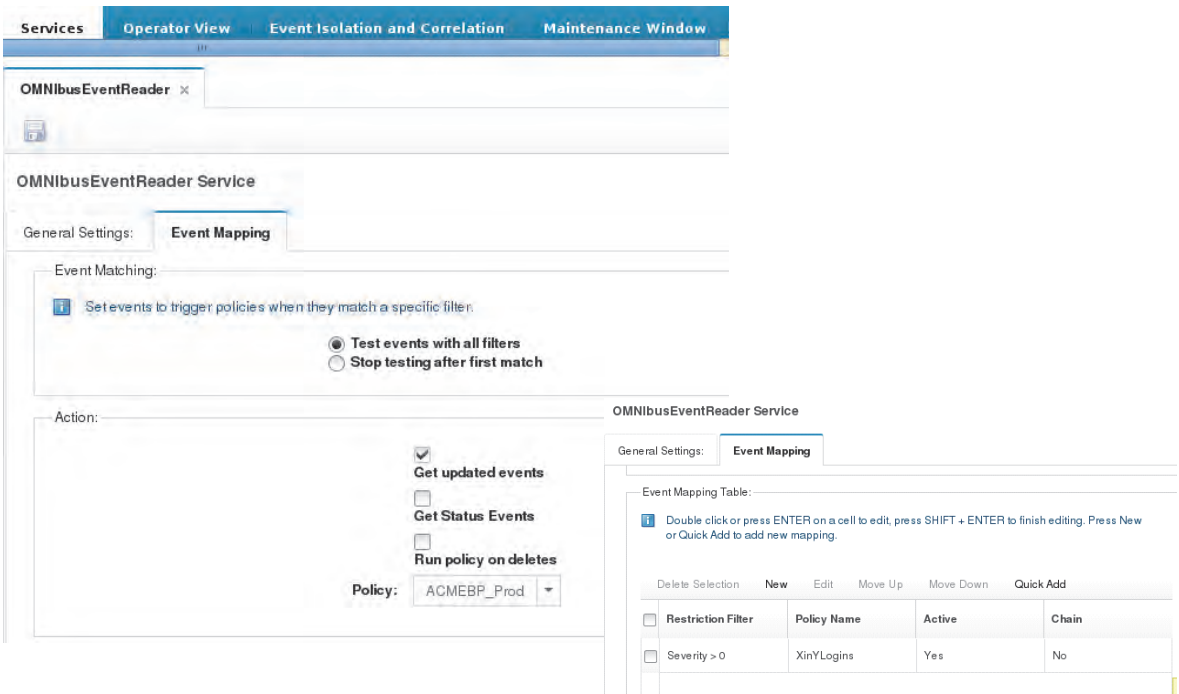
Note: Log statements assist in debugging the policy.

Exercise 5 Testing the policy

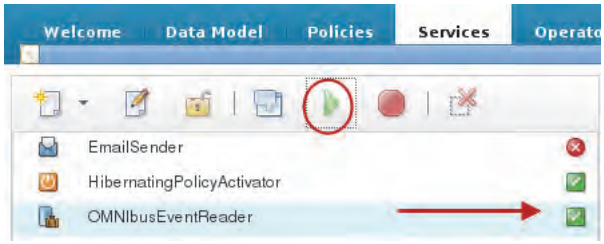
1. Add the **OMNIbusEventReader** service to the **Default** project, as shown in the example.



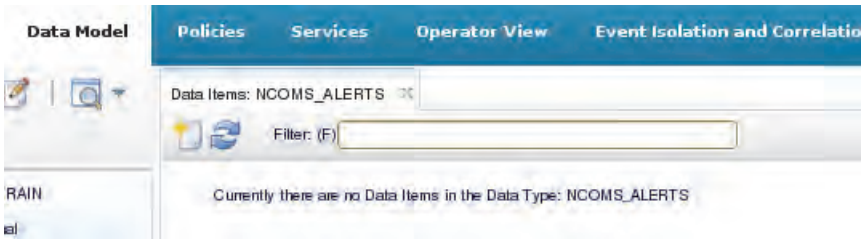
2. Click the **Services** tab and Edit **OMNIbusEventReader** general settings**.** Notice the mapping tab is already configured.



3. Start the **OMNIbusEventReader** service.



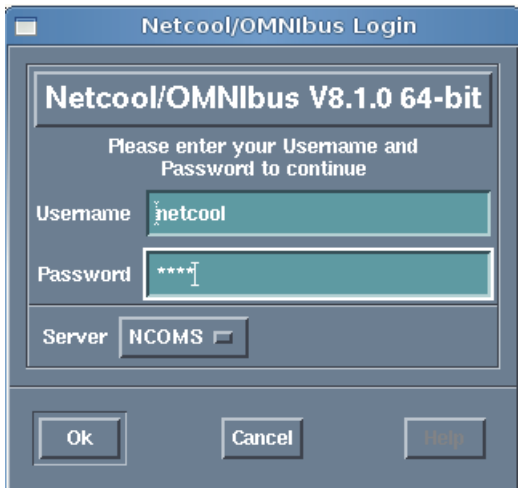
4. Delete the rows from the **NCOMS_ALERTS** table**.**



5. Test the policy by generating three login errors within 10 seconds. From a GNOME Terminal, type the following commands:

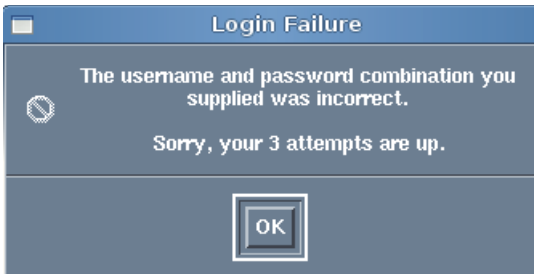
```
cd /opt/IBM/tivoli/netcool/omnibus/bin
./nco_event
```

6. When the login window opens, type **netcool** as the user name, and **test** as the password. Click **OK** three times quickly.



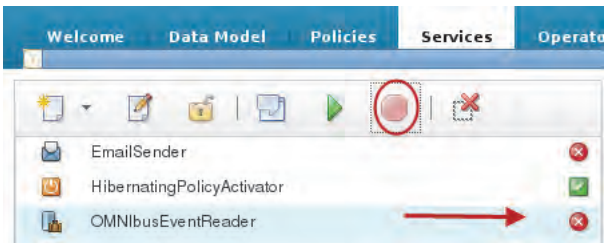
An error in OMNIbus error occurs, and the **XinY** policy generates an threshold event.

7. Click **OK** to continue.



A successful test displays **XinYLogins Threshold Exceeded: 3 >= 3** as part of the **Summary** field in **NCOMS_ALERTS** table**.**

8. Stop the OMNIbusEventReader service in the Services panel.

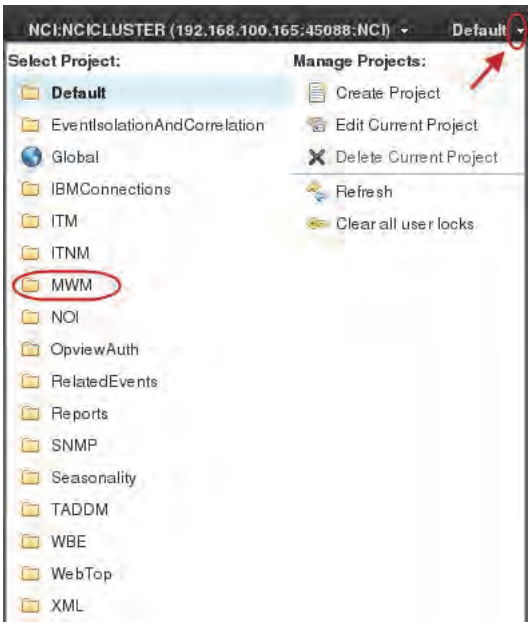


Unit 14 Maintenance window management exercises

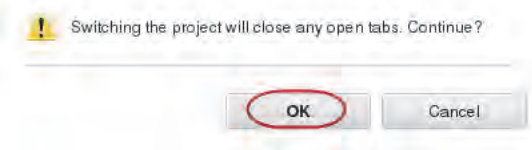
In this exercise, you experiment with the Maintenance Window Management service.

Exercise 1 Maintenance window management

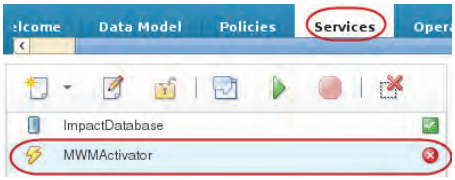
1. Switch to the MWM project.



2. Click **OK**.

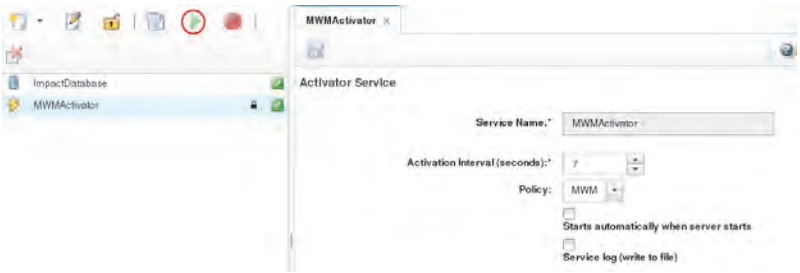


3. Click the **Services** tab. Start the **MWMActivator** service as shown in the following screen capture.

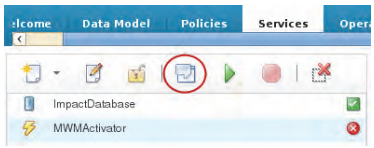


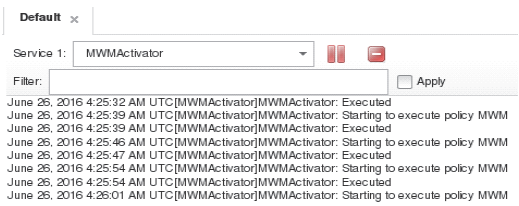
Double click **MWMActivator** to view the service configuration

Click the green arrow to start the service.

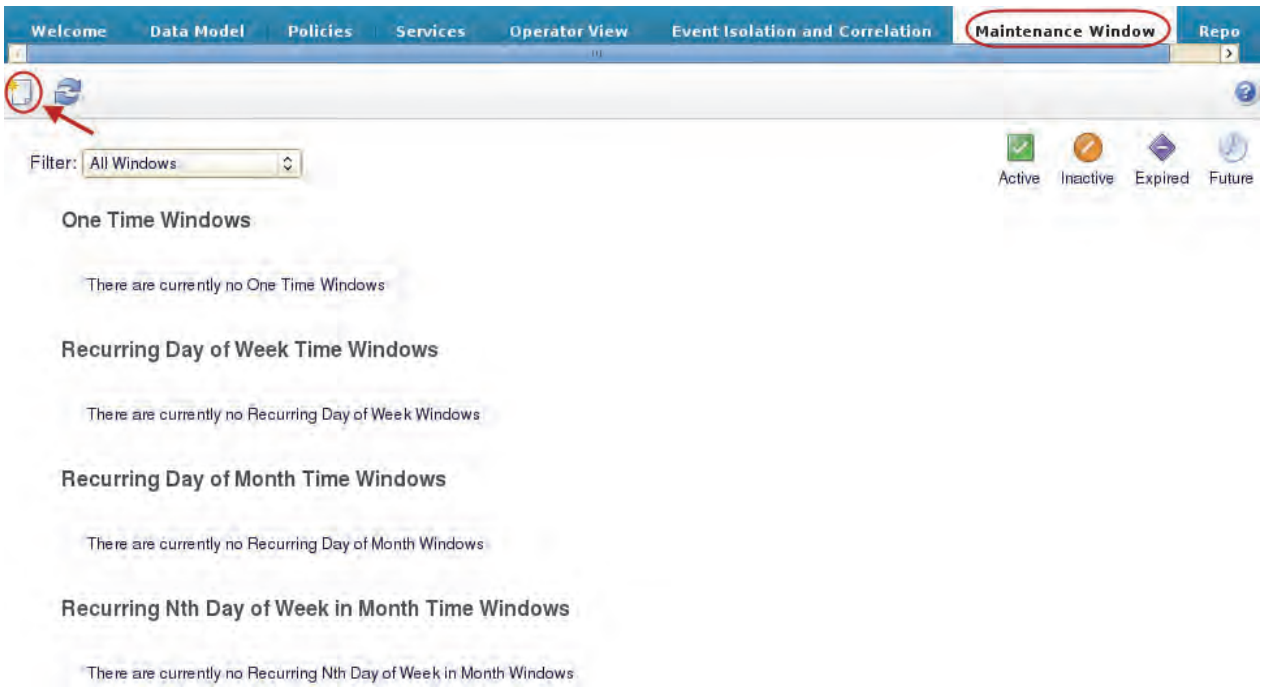


Click the view log icon.

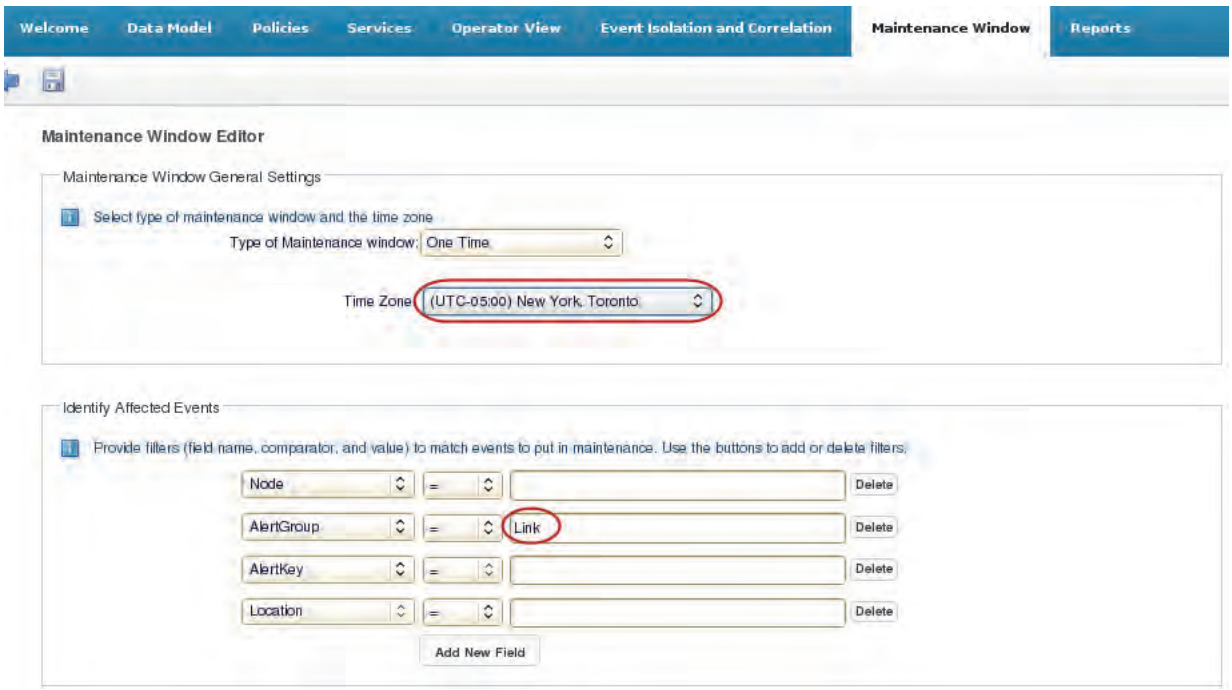




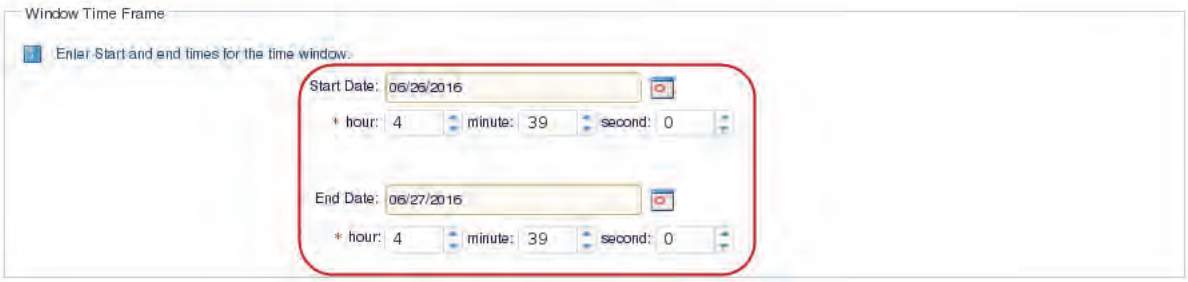
4. Click the **Maintenance Window** tab to open the home screen. Click the icon to create a new maintenance window.



5. Click **Time Zone** and select to **New York, Toronto**.Type **Link** in the **AlertGroup** field.

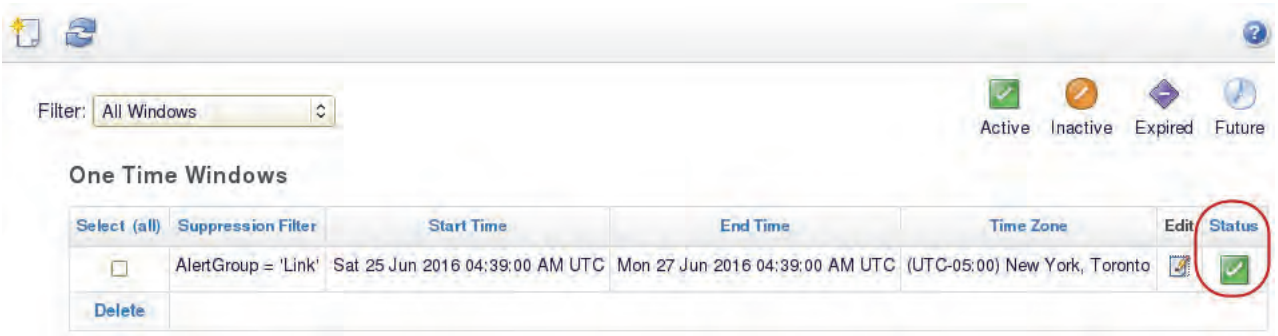


6. Scroll down to **Set the Window Time Frame**.
   - Set the **Start Time** to yesterday's date and the **End Time** to tomorrow's date.
   - Scroll to the top of the page to click the **Save** button.



7. Click the Back arrow to view the configured service.

The green checkbox icon indicates a **Active** maintenance window.



8. Double-click the GNOME Terminal icon on the desktop.

9. To generate OMNIbus events, start the Simnet Probe by typing the following commands from the command line as shown:

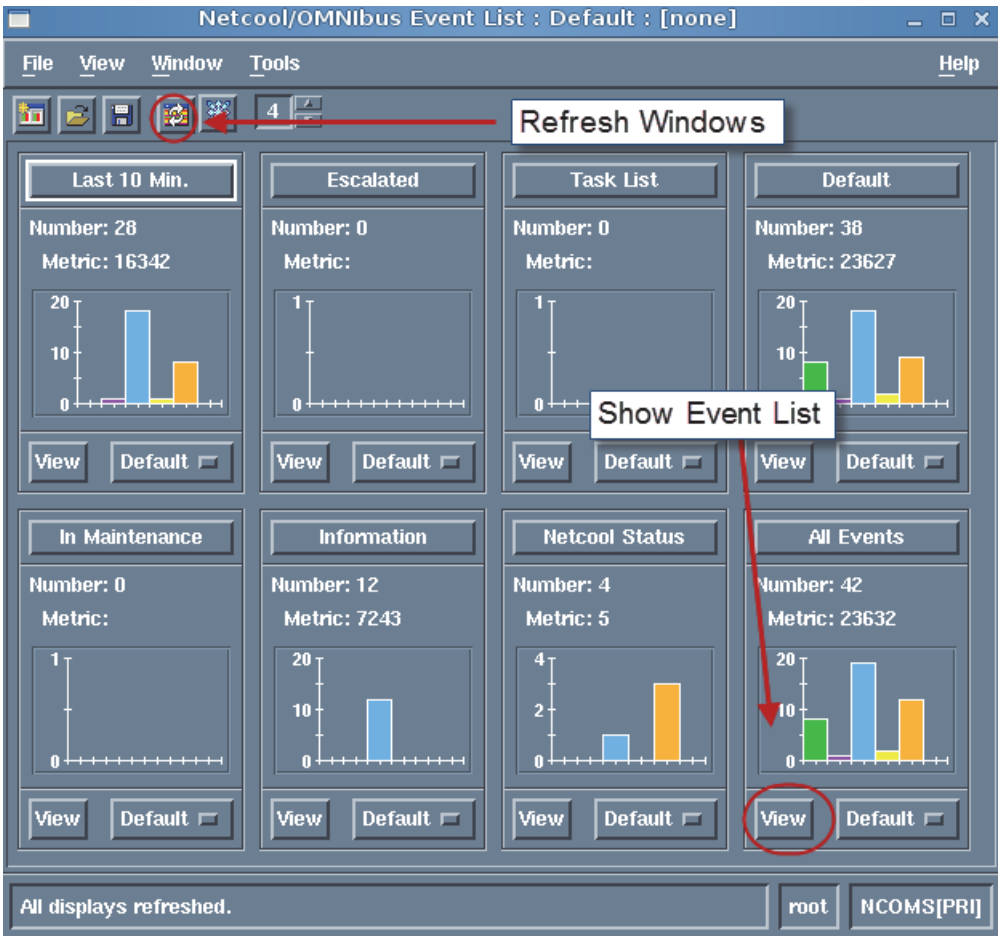
```
cd /opt/IBM/tivoli/netcool/omnibus/probes
./nco_p_simnet &
```

Note: Recall that **SuppressEscl** = 6 indicates that the event is in maintenance.

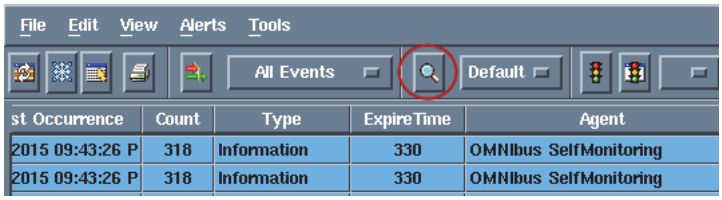
10. To check the **SuppressEscl** field in OMNIbus type the following commands to start the OMNIbus Event Viewer.

```
cd /opt/IBM/tivoli/netcool/omnibus/bin
./nco_event
```

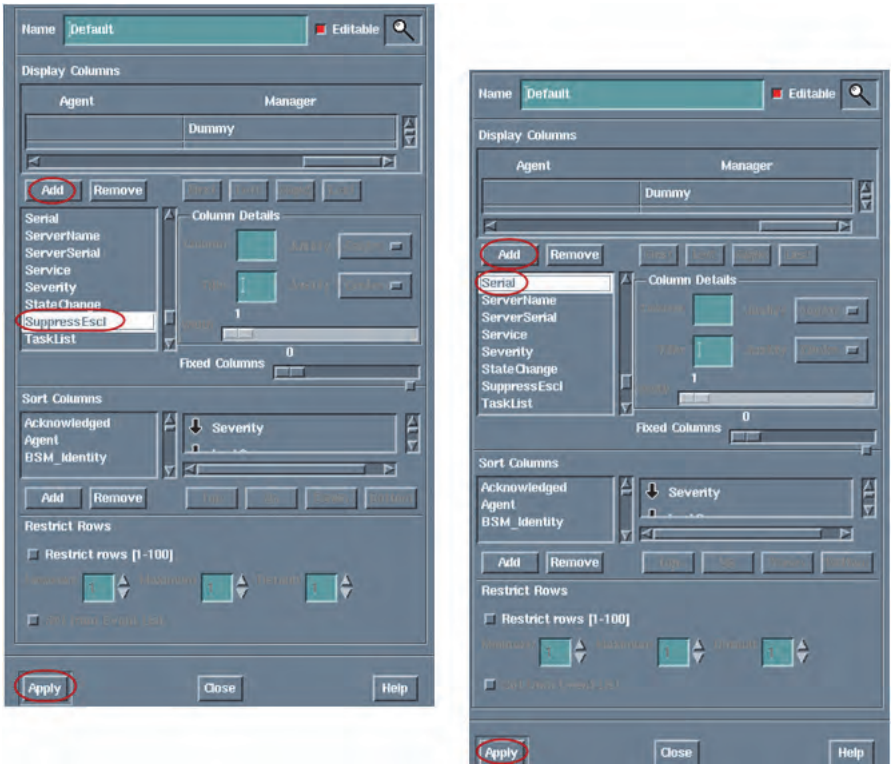
11. Click the **Refresh Windows** button to see the metrics change. Click **View** to see the event list.



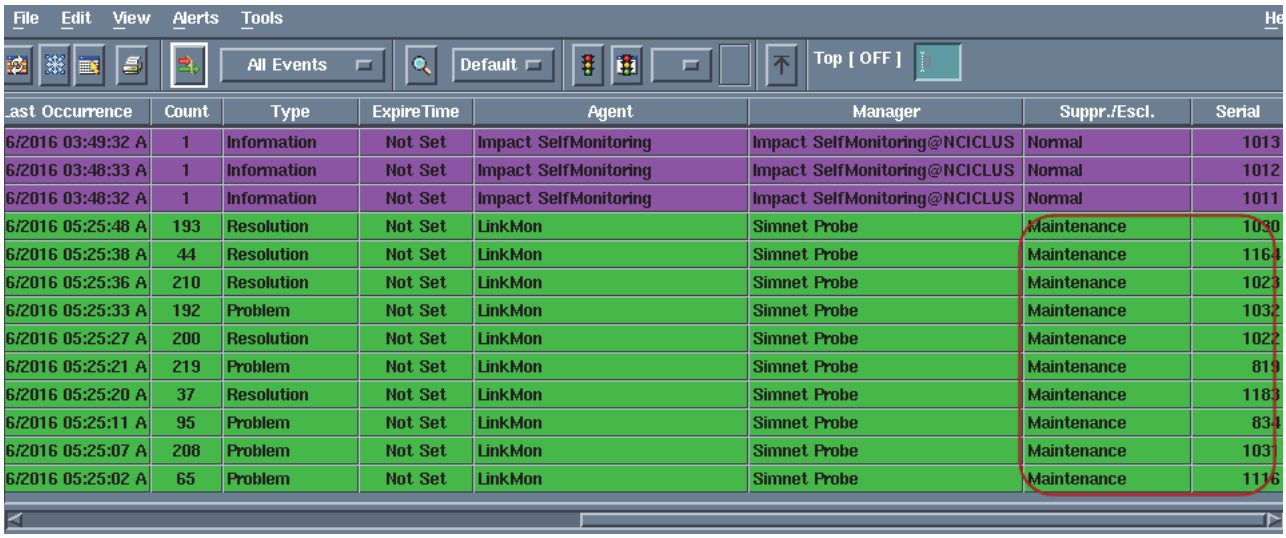
12. Click the magnifying glass.



13. Select the **SuppressEscl** and the **Serial** fields. Click **Add** to include the fields in the list and click **Apply** to make the changes.

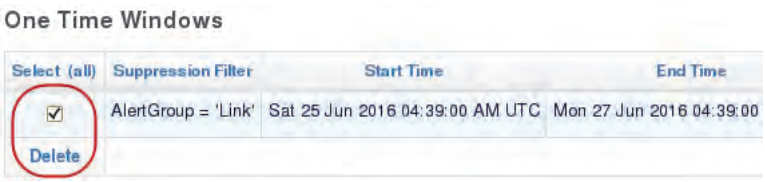


14. Scroll to the right and to the bottom to see events in Maintenance and Serial numbers.

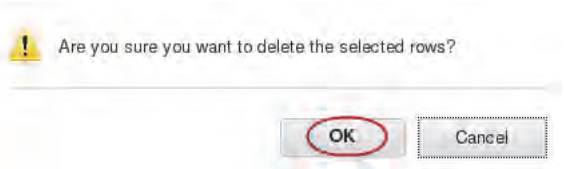


Last Occurrence	Count	Type	ExpireTime	Agent	Manager	Suppr./Escl.	Serial
6/2016 03:49:32 A	1	Information	Not Set	Impact SelfMonitoring	Impact SelfMonitoring@NCICLUS	Normal	1013
6/2016 03:48:33 A	1	Information	Not Set	Impact SelfMonitoring	Impact SelfMonitoring@NCICLUS	Normal	1012
6/2016 03:48:32 A	1	Information	Not Set	Impact SelfMonitoring	Impact SelfMonitoring@NCICLUS	Normal	1011
6/2016 05:25:48 A	193	Resolution	Not Set	LinkMon	Simnet Probe	Maintenance	1030
6/2016 05:25:38 A	44	Resolution	Not Set	LinkMon	Simnet Probe	Maintenance	1164
6/2016 05:25:36 A	210	Resolution	Not Set	LinkMon	Simnet Probe	Maintenance	1023
6/2016 05:25:33 A	192	Problem	Not Set	LinkMon	Simnet Probe	Maintenance	1032
6/2016 05:25:27 A	200	Resolution	Not Set	LinkMon	Simnet Probe	Maintenance	1022
6/2016 05:25:21 A	219	Problem	Not Set	LinkMon	Simnet Probe	Maintenance	819
6/2016 05:25:20 A	37	Resolution	Not Set	LinkMon	Simnet Probe	Maintenance	1183
6/2016 05:25:11 A	95	Problem	Not Set	LinkMon	Simnet Probe	Maintenance	834
6/2016 05:25:07 A	208	Problem	Not Set	LinkMon	Simnet Probe	Maintenance	1031
6/2016 05:25:02 A	65	Problem	Not Set	LinkMon	Simnet Probe	Maintenance	1116

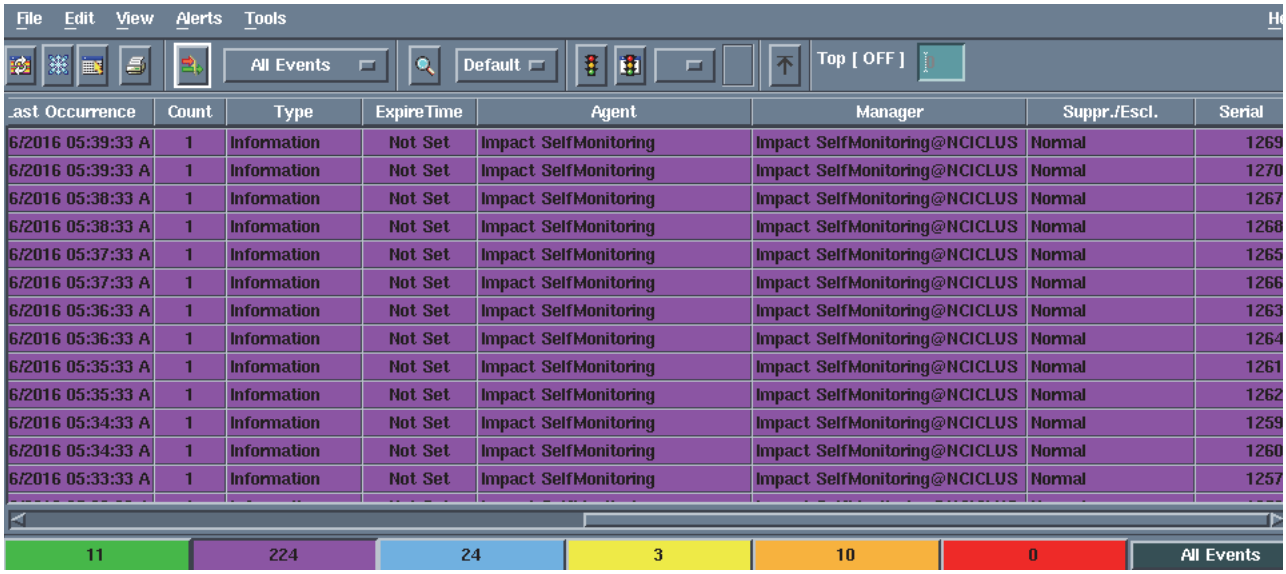
15. Select the table entry and click delete to remove the maintenance window.



16. Confirm the delete. Click **OK**.



17. Refresh the event list to see the change. The **SuppressEscl** field should not have maintenance events.

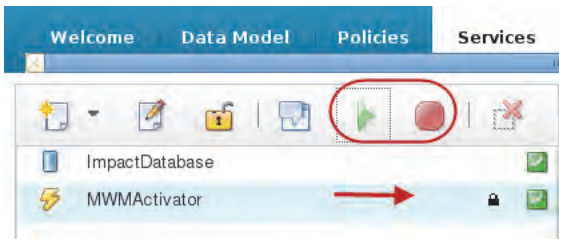


Exercise 2 Modifying Maintenance Window Management properties

1. Open the **MWM_Properties** policy.
2. Set **clearFlag** to **FALSE**.
3. Click the **Check Syntax** icon. Click **Save**.

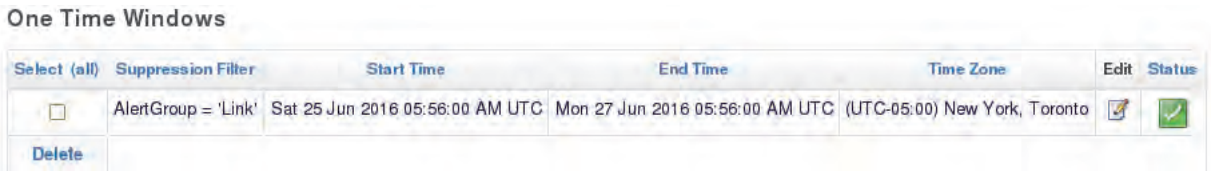
   When the **clearFlag** variable is set to **FALSE**, the events are left tagged as in maintenance after the maintenance window expires. Click the **Check Syntax** icon. Click **Save**.

4. Stop and start the **MWMActivator** service.



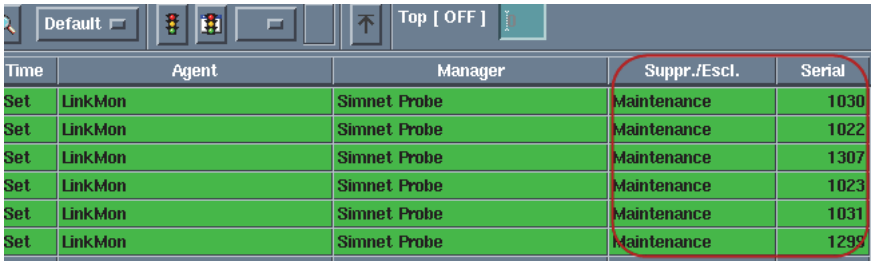
5. Recreate the One Time maintenance from the earlier exercise. Set the **Window Time Frame**, set the **Start Time** to yesterday's date and the **End Time** to tomorrow's date.

The result should look as follows:

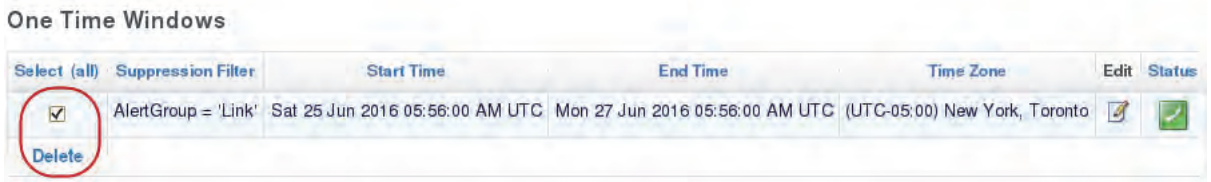


Maintenance events reappear.

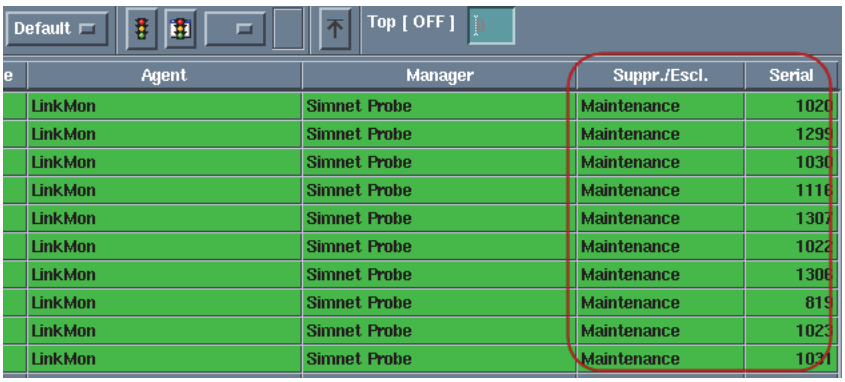
6. View the results in the OMNIbus event viewer.



7. Force the maintenance window to end by deleting the table entry.



8. Refresh and view the results in the OMNIbus event viewer. Notice that the **SuppressEscl** status is unchanged.



Unit 15 Command-line tools and self-monitoring exercises

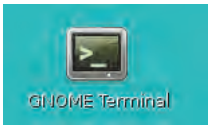
In this unit you experiment with policy encryption, and self-monitoring.

Exercise 1 Encrypting a policy

In this exercise, you experiment with policy encryption. Encryption serves two primary purposes:

- Software providers or business partners can use encryption to create policies and solutions for clients and customers without handing over intellectual property.
- An administrator can use encryption to lock modifications to policies in a production or change-sensitive environment.

1. Double-click the **Gnome Terminal** icon.



2. Type the following command to navigate to the **/opt/IBM/tivoli/impact/bin** directory.

```
cd /opt/IBM/tivoli/impact/bin
```

3. Using the following encryption parameters to construct the encryption command:

   Netcool/Impact Server: **NCI**

   Encryption Password: **ProdSecure1**

   Input Unencrypted Policy: **ACMEBP_Test.ipl**

   Output Encrypted Policy: **ACMEBP_Prod.ipl**

Policy Location Directory: **labfiles**

ACMEBP_Test.ipl

```
//Impact Class Policy
log("ACMEBP Policy Start");
PrblmTktAssign = NewObject();
PrblmTktAssign.Name = "John Smith";
PrblmTktAssign.Phone = "18001234567";
PrblmTktAssign.Severity = "2";
PrblmTktAssign.Status = "open";
Log(PrblmTktAssign.Name + "/" + PrblmTktAssign.Phone + "/" +
PrblmTktAssign.Severity + "/" + PrblmTktAssign.Status);
log("ACMEBP Policy End");
```

Figure 1 ACMEBP_Test Policy Contents

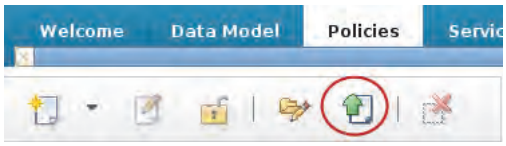
4. Run the **nci_encryptpolicy** command.

```
./nci_encryptpolicy NCI ProdSecure1 /labfiles/ACMEBP_Test.ipl
/labfiles/ACMEBP_Prod.ipl
```

The results look similar to the following example:

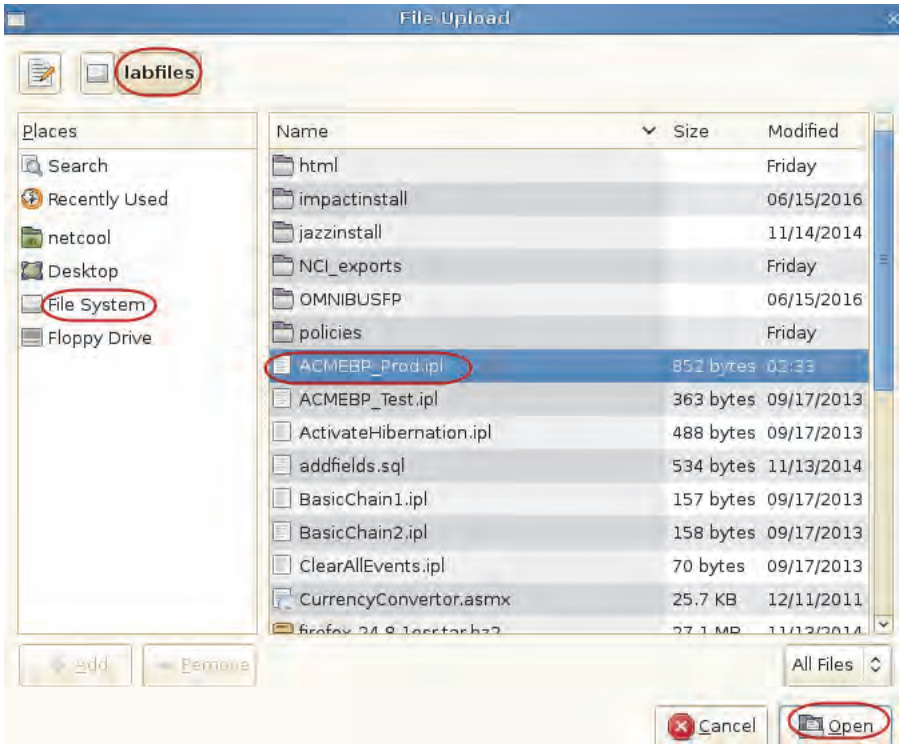
```
File  Edit  View  Terminal  Help
netcool@jazz01:~> cd /opt/IBM/tivoli/impact/bin
netcool@jazz01:/opt/IBM/tivoli/impact/bin> ./nci_encryptpolicy NCI ProdSecure1 /
labfiles/ACMEBP_Test.ipl /labfiles/ACMEBP_Prod.ipl
netcool@jazz01:/opt/IBM/tivoli/impact/bin>
```

5. Upload the **ACMEBP_Prod.ipl** policy into the policy editor using the **Upload a Policy File** icon

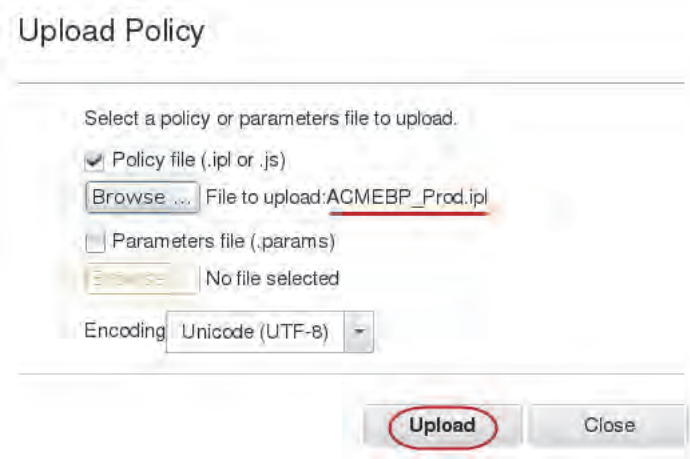


6. Locate the encrypted file. Click **Browse**.

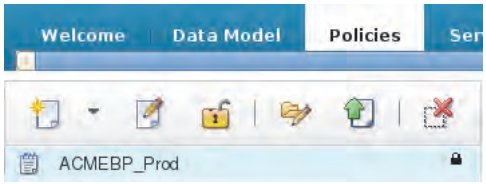
7. Click **File System>labfiles.** Select **ACMEBP_Prod.ipl**, click **Open**.



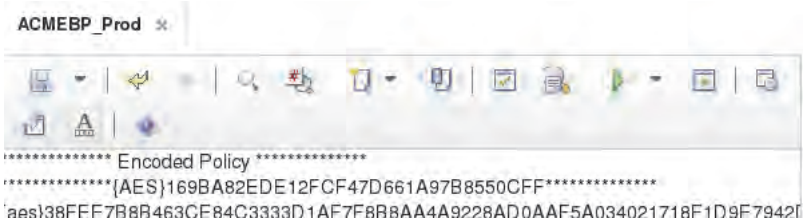
8. Click **Upload**.



After the encrypted policy is imported, you can use it in services such as event readers the same way that unencrypted policies are used.

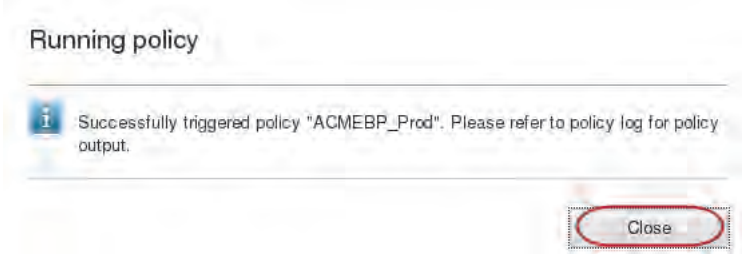


However, when viewing the policy, the user sees only encrypted code.



9. Trigger the **ACMEBP_Prod** policy from the policy editor. Click the **Run Policy** icon.

10. Click **Close** to exit the window.



11. View the results in the policy logger.

```
[PolicyLogger][ACMEBP_Prod][pool-3-thread-57]Parser log: ACMEBP Policy Start
[PolicyLogger][ACMEBP_Prod][pool-3-thread-57]Parser log: John Smith/18001234567/2/open
[PolicyLogger][ACMEBP_Prod][pool-3-thread-57]Parser log: ACMEBP Policy End
```

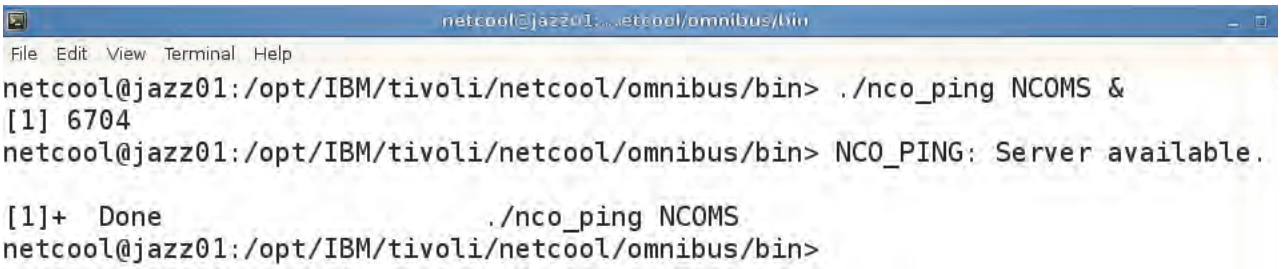
Exercise 2 Enabling self-monitoring

In this exercise, you enable self-monitoring using the command-line manager and the GUI.

1. Double-click the **Gnome Terminal** icon on the desktop.

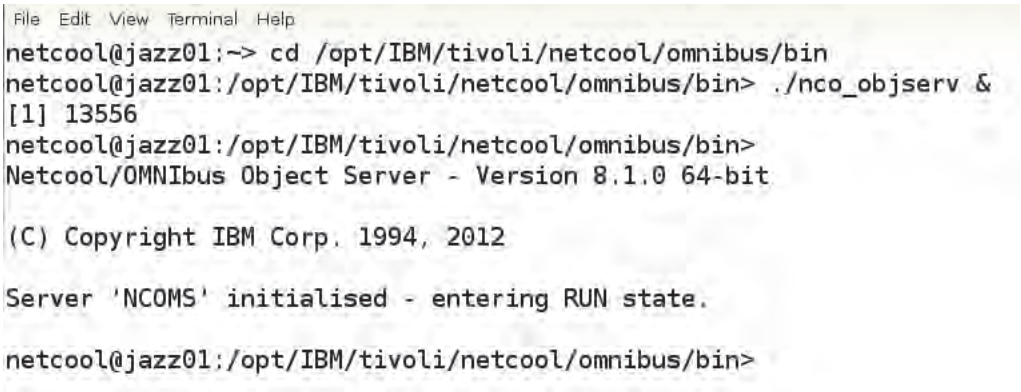
2. Ping the OMNIbus server.

```
cd /opt/IBM/tivoli/netcool/omnibus/bin
./nco_ping NCOMS &
```



If the server is unavailable, use **nco_objserv** to start the server.

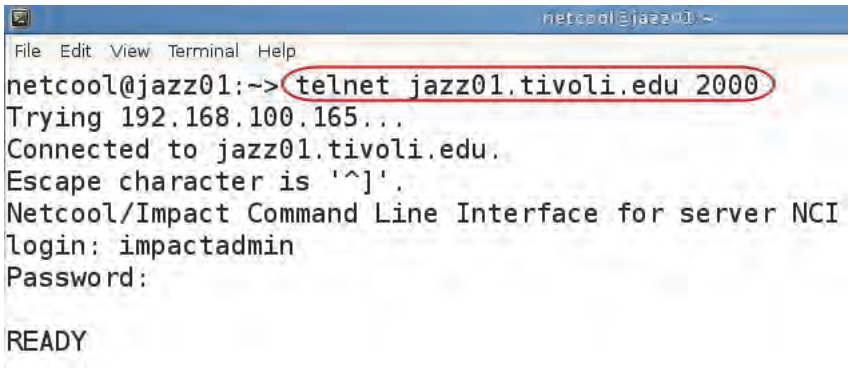
```
cd /opt/IBM/tivoli/netcool/omnibus/bin
./nco_objserv &
```



3. Start the simnet probe using the nco_p_simnet command.

```
cd /opt/IBM/tivoli/netcool/omnibus/probes
./nco_p_simnet &
```

4. From a command prompt, start the command-line manager (telnet) using the following parameters:
   - Server: **jazz01.tivoli.edu**
   - Port: **2000**
   - User: **impactadmin**
   - Password: **object00**



5. Enable Cluster Status Monitoring using the following command:

```
UPDATE Service SET EnableClusterStatus = true WHERE Name = 'SelfMonitoring';
```

The results resemble the following screen capture:

```
READY
UPDATE Service SET EnableClusterStatus = true WHERE Name = 'SelfMonitoring';
Command Executed
READY
```

6. Start the self-monitoring service in the command-line manager by typing the following command:

   ```
   UPDATE Service SET Running = true WHERE Name = 'SelfMonitoring';
   ```

   The results resemble the following screen capture:

   ```
   UPDATE Service SET Running = true WHERE Name = 'SelfMonitoring';
   Command Executed
   READY
   ```

   If the OMNIbus server is not running, the following error occurs:

   ```
   READY
   UPDATE Service SET Running = true WHERE Name = 'SelfMonitoring';
   Execution Failed, check netcool.log for details.
   READY
   ```

7. Enable Queue Status Monitoring using the following command:

   ```
   UPDATE Service SET EnableQueueStatus = true WHERE Name = 'SelfMonitoring';
   ```

   The results resemble the following screen capture:

   ```
   UPDATE Service SET EnableQueueStatus = true WHERE Name = 'SelfMonitoring';
   Command Executed
   READY
   ```

8. Issue the following command to see if Queue Status Monitoring is enabled:

   ```
   SELECT IsQueueStatusEnabled FROM Service WHERE Name = 'SelfMonitoring';
   ```

   The results resemble the following screen capture:

   ```
   READY
   SELECT IsQueueStatusEnabled FROM Service WHERE Name = 'SelfMonitoring';

   STARTRECORD
   IsQueueStatusEnabled
   ----------------------
   false
   ENDRECORD

   READY
   ```

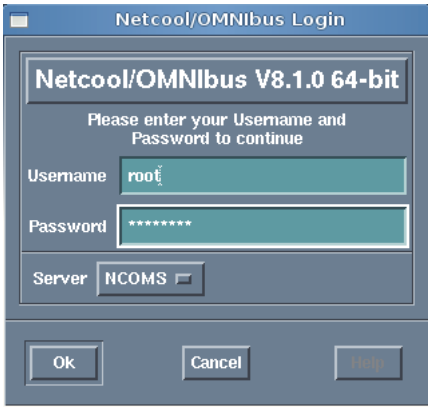
9. Log in to OMNIbus to view self-monitoring events. Go to the following directory:

   ```
   cd /opt/IBM/tivoli/netcool/omnibus/bin
   ```

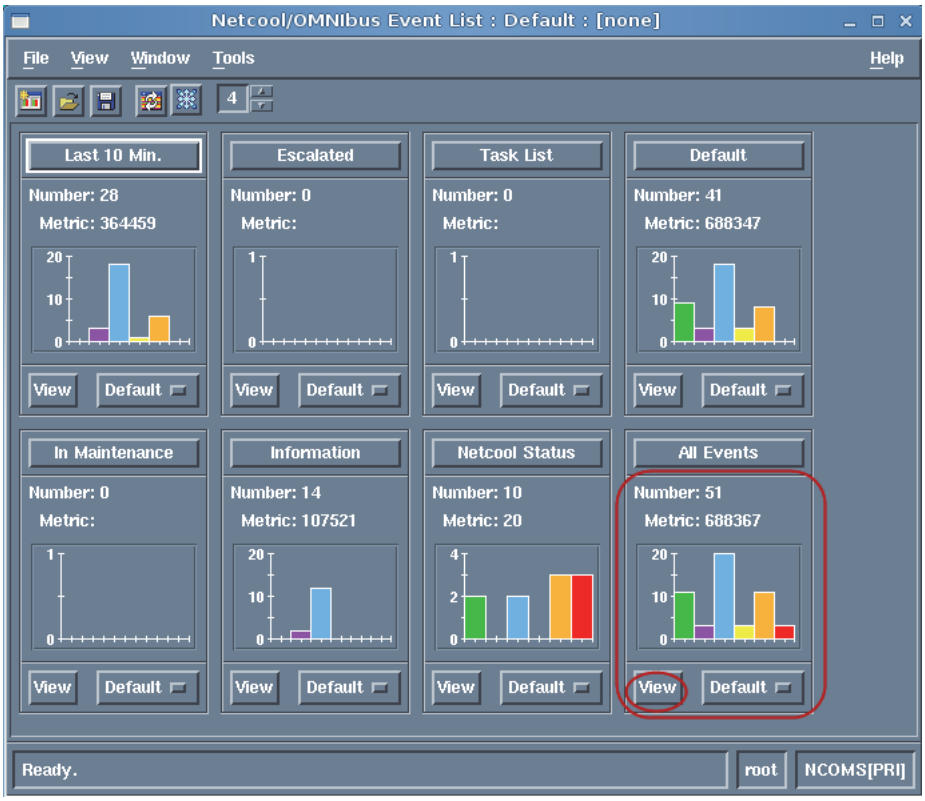
10. Start OMNIbus using the following command:

```
./nco_event
```

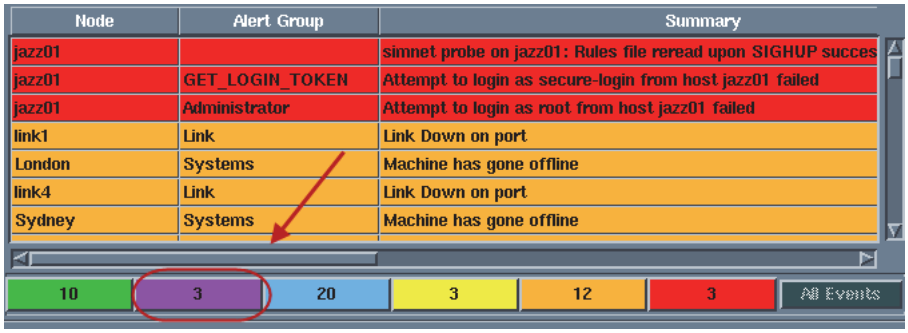
11. Log in with the user name **root**, password **object00**.



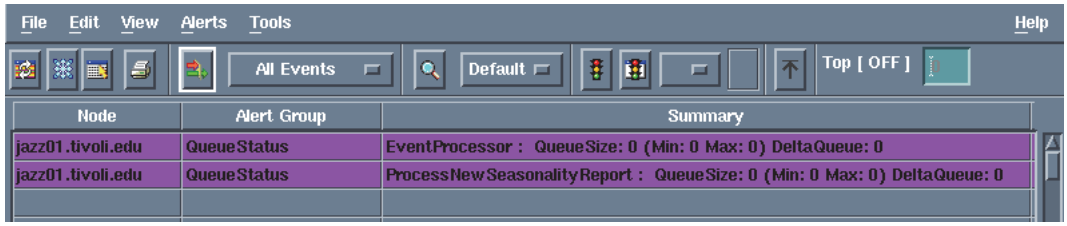
12. Locate the **All Events** box and click **View**.



13. Click the purple tab at the bottom of the window.



14. View the events, and locate the entry for **Queue Status** as displayed:



The self-monitoring service sends Cluster Status events to the ObjectServer under the following conditions:

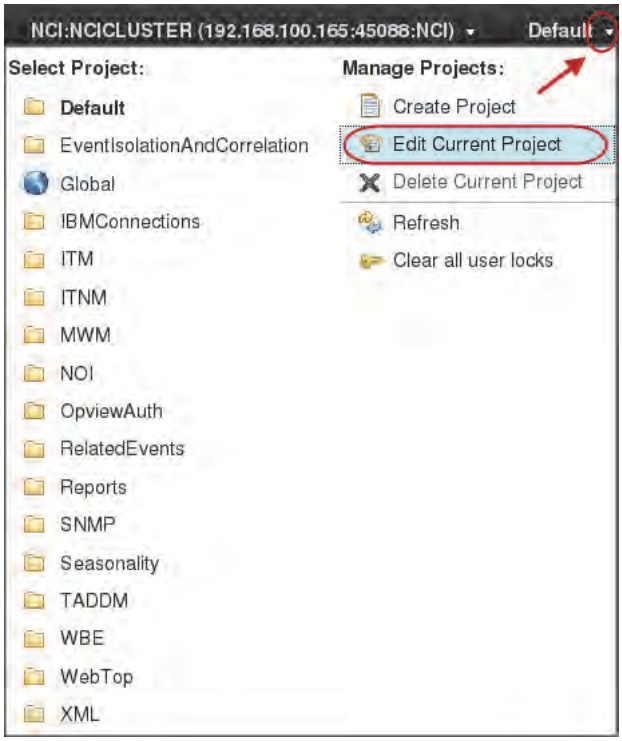
– A secondary cluster member becomes unavailable.

– A secondary cluster member assumes the primary role.

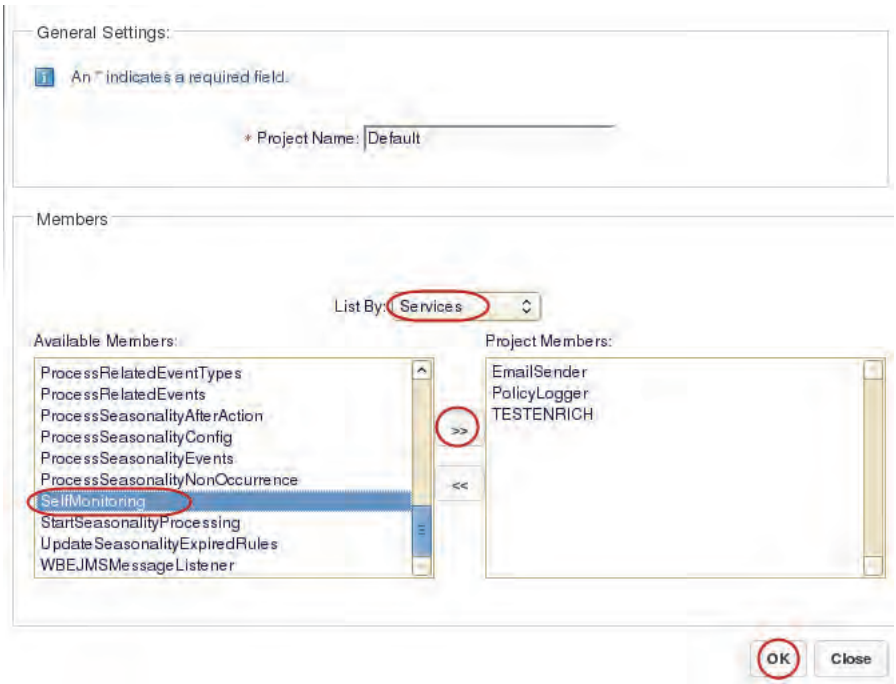
– A secondary cluster member detects that the primary has become unavailable.

– A secondary cluster member accepts a new primary server.

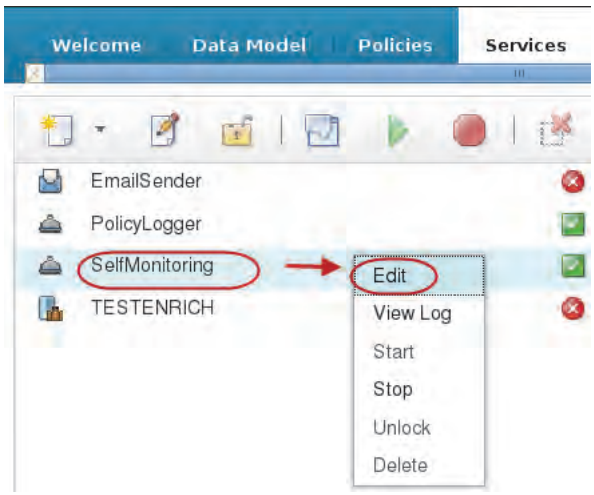
15. To add the **SelfMonitoring** service to the **Default** project, select **Edit Current Project**.



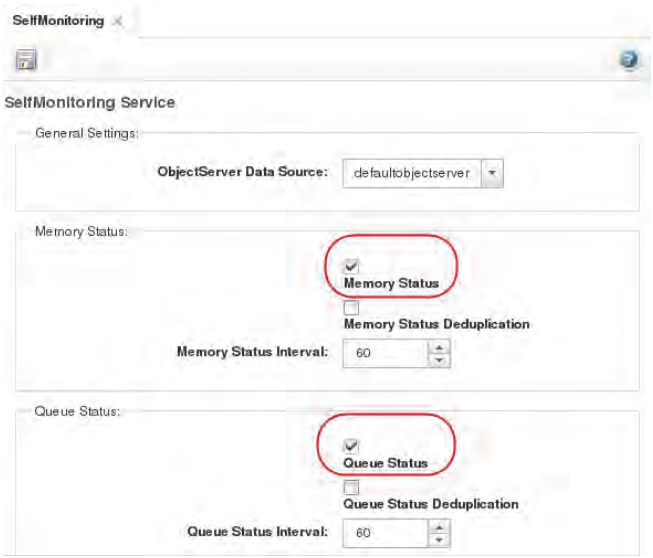
16. Add the **SelfMonitoring** service to the **Default** project. Select **List By Services** and select **SelfMonitoring** from **Available Members**. Click the arrows (>>) to add to **Project Members**. Click **Ok**.



17. Click the **Services** tab. Edit **SelfMonitoring**.

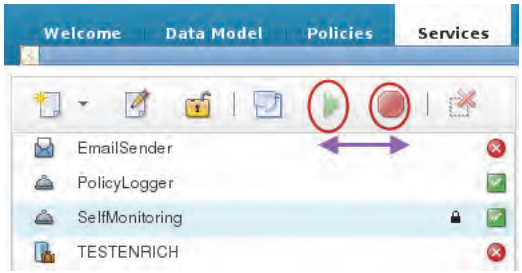


18. Edit the **SelfMonitoring** service configuration screen as shown in the following screen capture. Click the **Save service** icon.



Note: The events are generated every 60 seconds.

19. Stop and start the **SelfMonitoring** service from the GUI.



20. View the changes in the OMNIbus event viewer. Notice the new events for **MemoryStatus**.

QueueStatus	EventProcessor : QueueSize: 0 (Min: 0 Max: 0) DeltaQueue: 0
MemoryStatus	Impact's Heap using 118M out of 1200M, Free System Memory Available: 3183M
QueueStatus	EventProcessor : QueueSize: 0 (Min: 0 Max: 0) DeltaQueue: 0
MemoryStatus	Impact's Heap using 233M out of 1200M, Free System Memory Available: 3181M
QueueStatus	EventProcessor : QueueSize: 0 (Min: 0 Max: 0) DeltaQueue: 0
MemoryStatus	Impact's Heap using 197M out of 1200M, Free System Memory Available: 3182M
QueueStatus	EventProcessor : QueueSize: 0 (Min: 0 Max: 0) DeltaQueue: 0

21. Type **exit** to close the telnet connection.

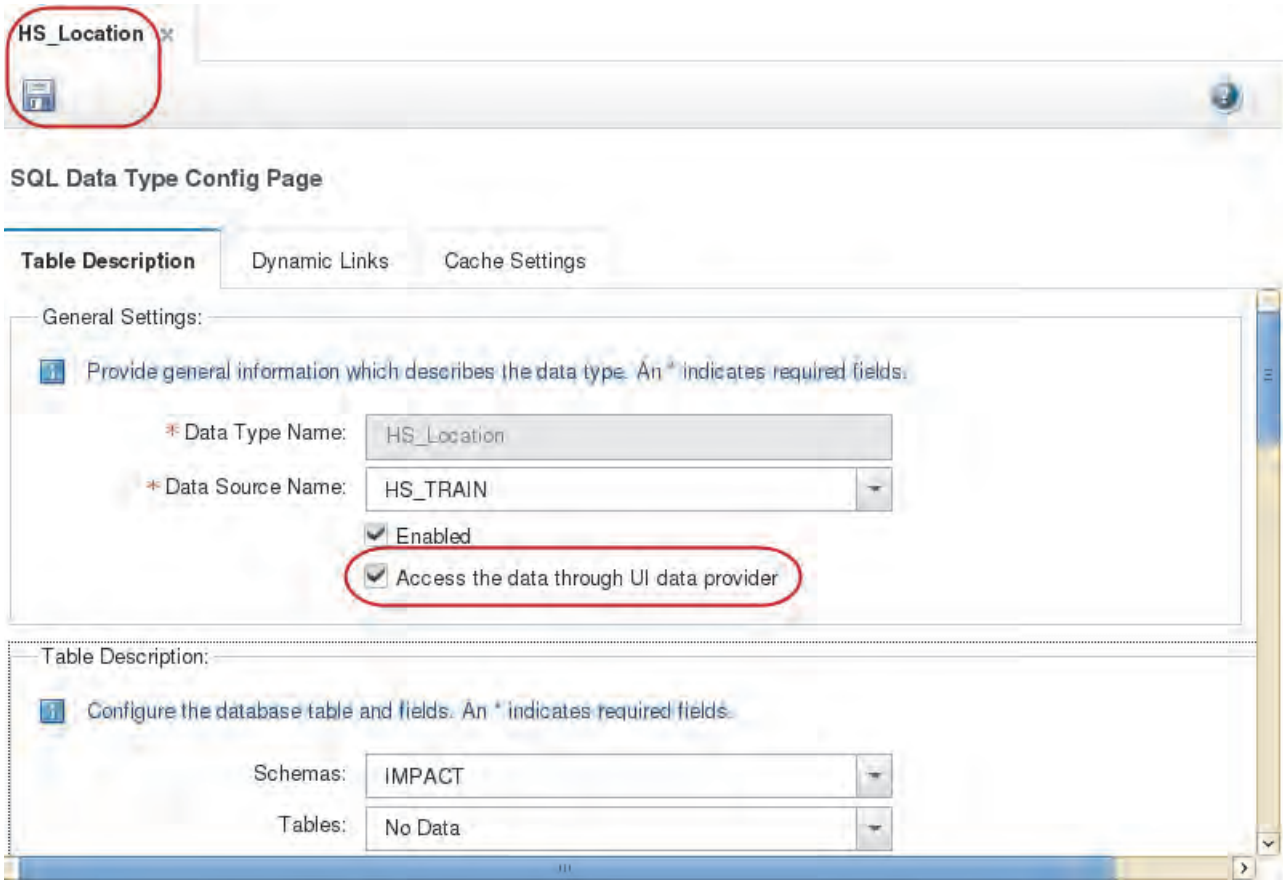
```
READY
exit
Connection closed by foreign host.
```

Unit 16 The Netcool/Impact UI data provider exercises

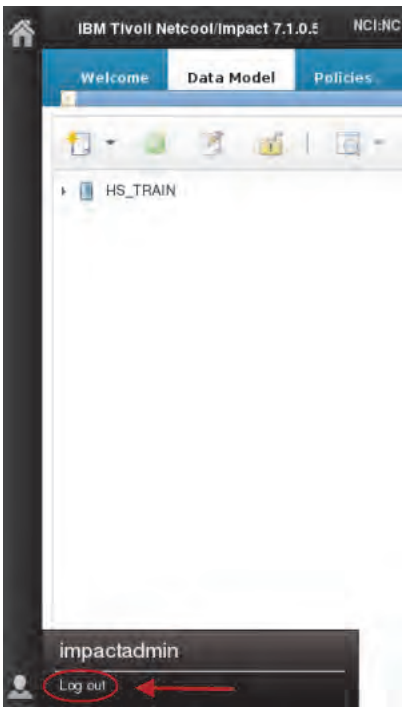
In this unit, a Netcool/Impact policy passes data to the IBM Service Management Hub to populate dashboard widgets.

Exercise 1 Configuring a data source

1. Edit the **HS_Location** Data Type (HS_TRAIN Data Source) and select the check box for **Access the data through UI data provider**. Click **Save**.



2. Click **Log out** to exit Impact.



Exercise 2 Building a widget

1. Open a GNOME Terminal on the desktop and type the startup command for JazzSM.

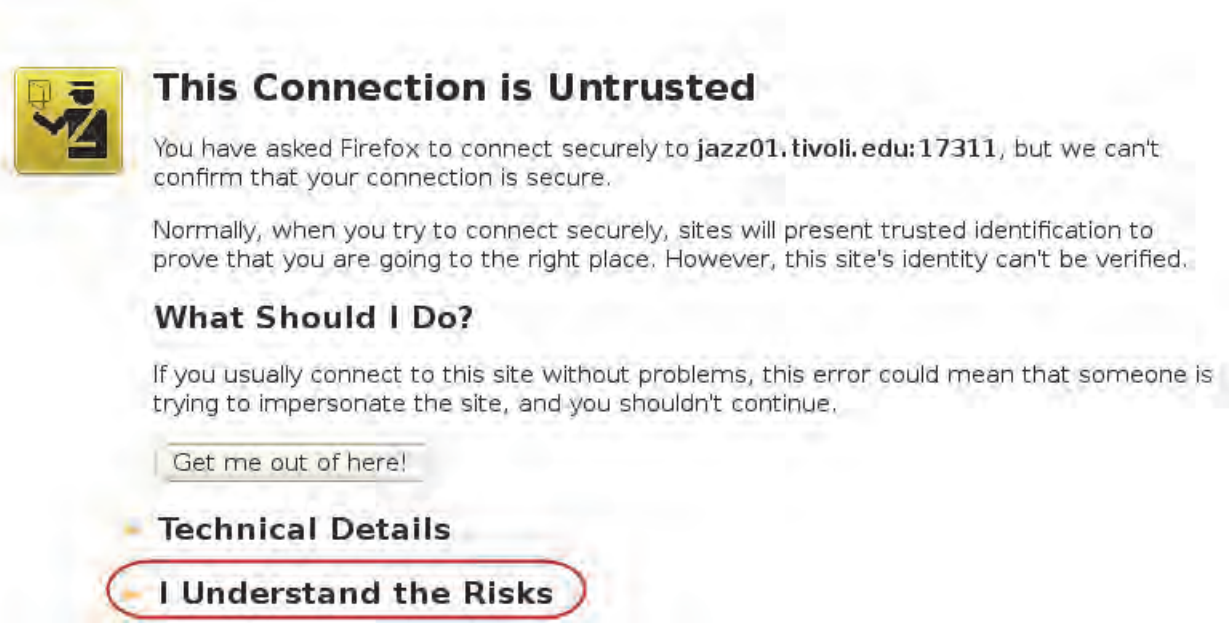

```
cd /home/netcool/IBM/JazzSM/profile/bin
./startServer.sh server1
```

   Wait for the JazzSM server to start.

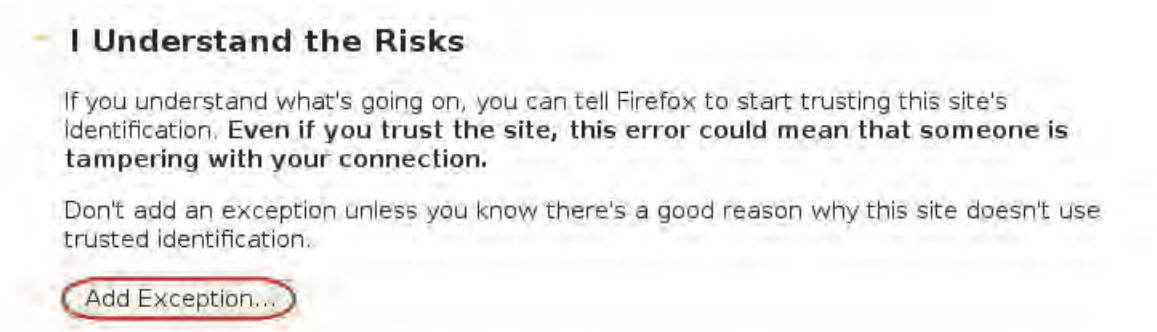
2. Open a Firefox browser and connect to the following URL:

   `http://jazz01.tivoli.edu:17310/ibm/console`

3. Accept the connection. Click **I understand the Risks.**

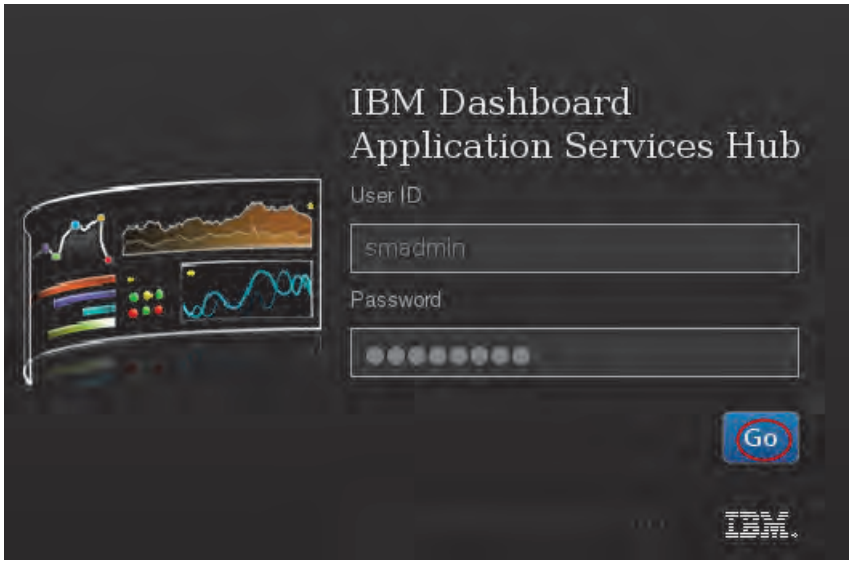


4. Click **Add Exception**.



5. Add security exception. Click **Confirm Security Exception**.

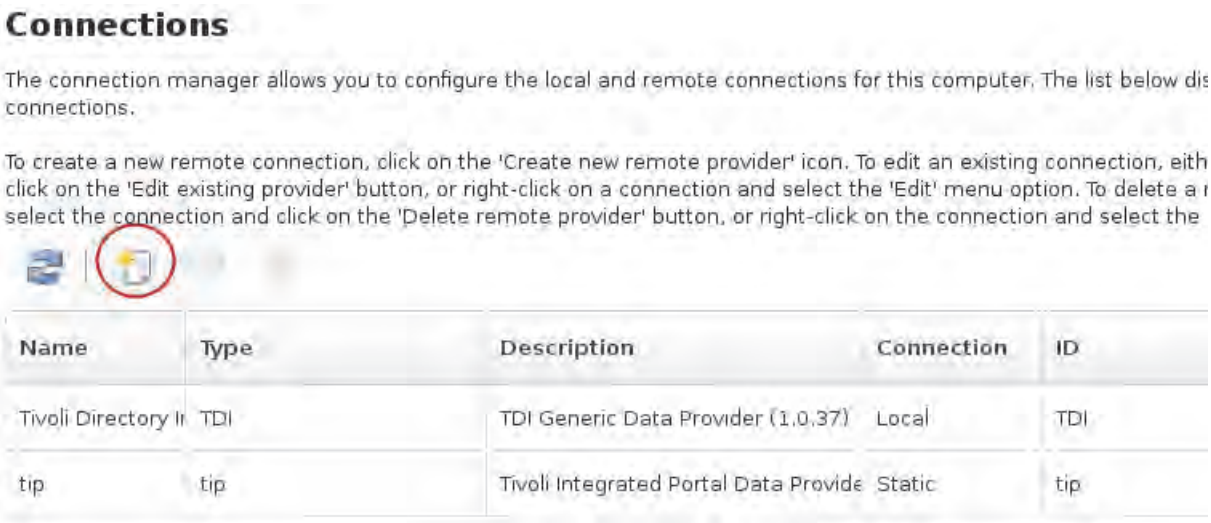
6. Type **smadmin** and **object00** as the user ID and password.



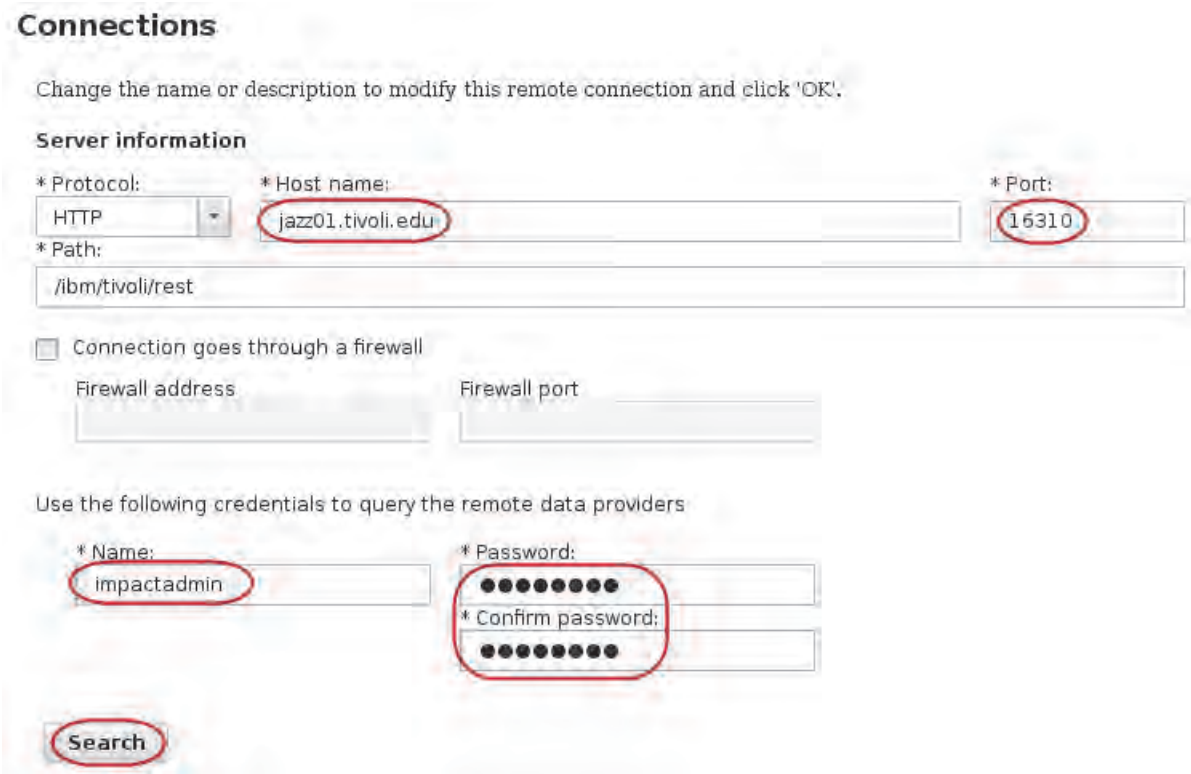
The welcome screen provides options for videos, tours, and community connections.

7. Create the connection to Impact. Click the console settings icon. Click **Connections**.

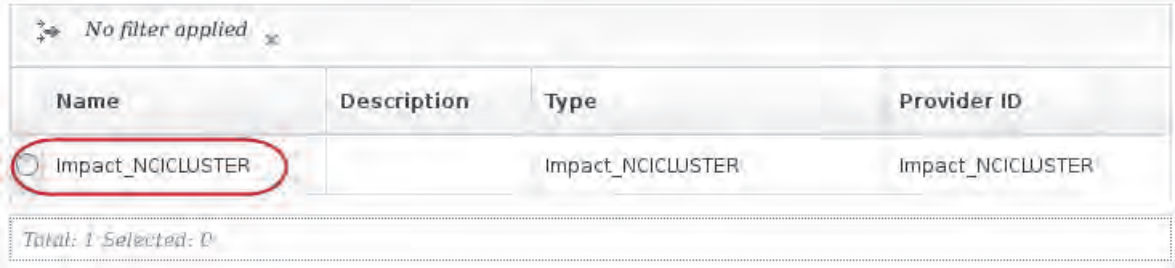
8. Establish a connection. Click the **Create a new Remote Provider** icon.



9. Enter the connection information as displayed in the example. Click **Search**.
   - Host Name: jazz01.tivoli.edu
   - Port:16310
   - Name: impactadmin
   - Password: object00
   - Confirm Password: object00



10. Scroll down to select the table entry. Click **Impact_NCICLUSTER**.



The connection information will automatically display as shown in the example.

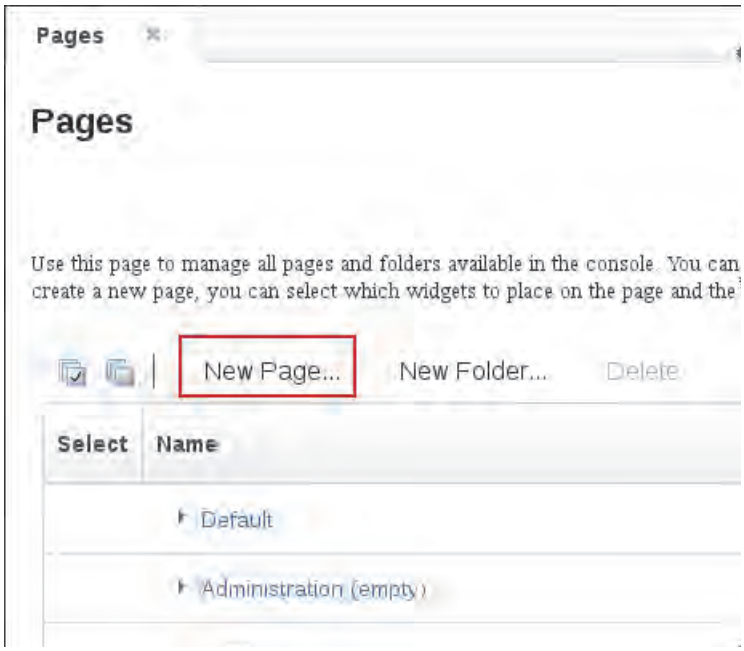
11. Click **OK**.
    - Name: Impact_NCICLUSTER
    - Description: Impact_NCICLUSTER

12. Examine the **Impact_NCICLUSTER** table entry. Verify it has **Working** as status.

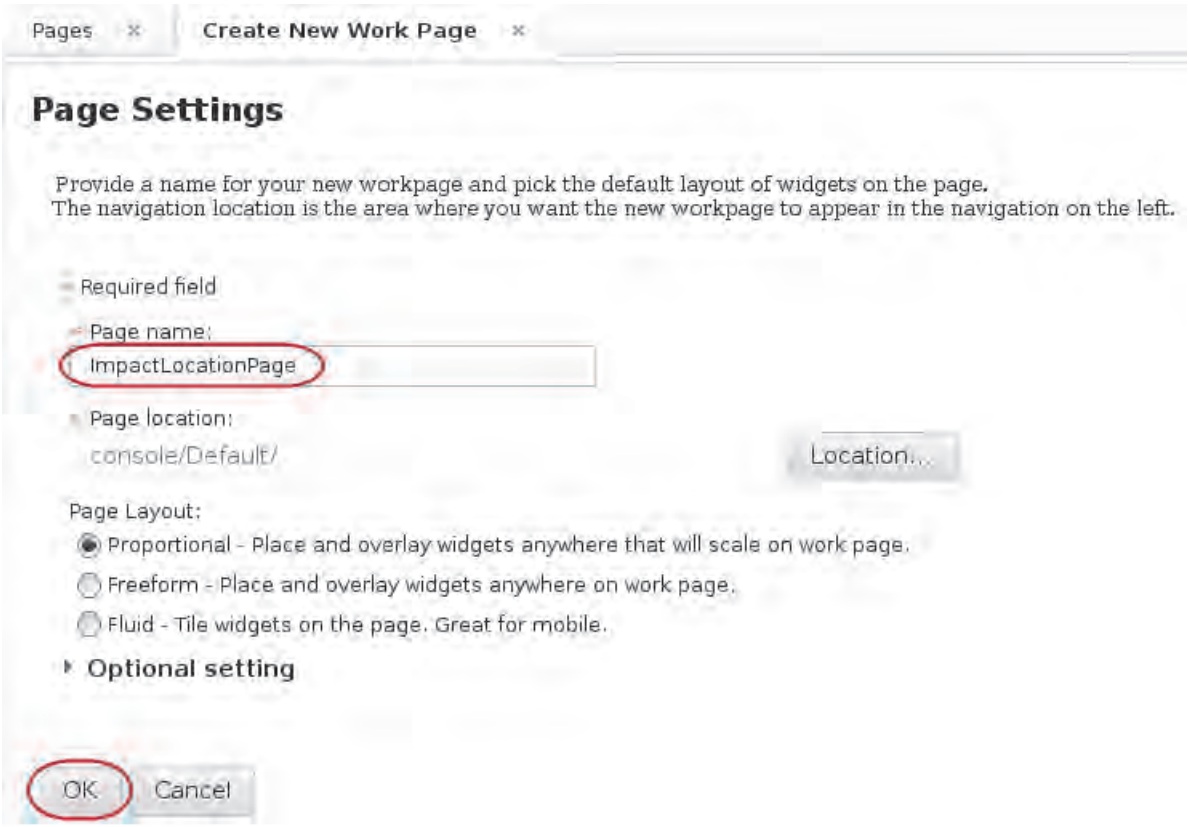
Name	Type	Description	Connection	ID	Status
Impact_NCICLUSTER	Impact_NCICLUSTER	Impact_NCICLUSTER	Remote	Impact_NCICLUSTER.jazz01.tivoli.edu	Working
Tivoli Directory Integr	TDI	TDI Generic Data Prov	Local	TDI	No data r
tip	tip	Tivoli Integrated Porta	Static	tip	Working

13. Build a new page by clicking **Console Settings > Pages** as shown.

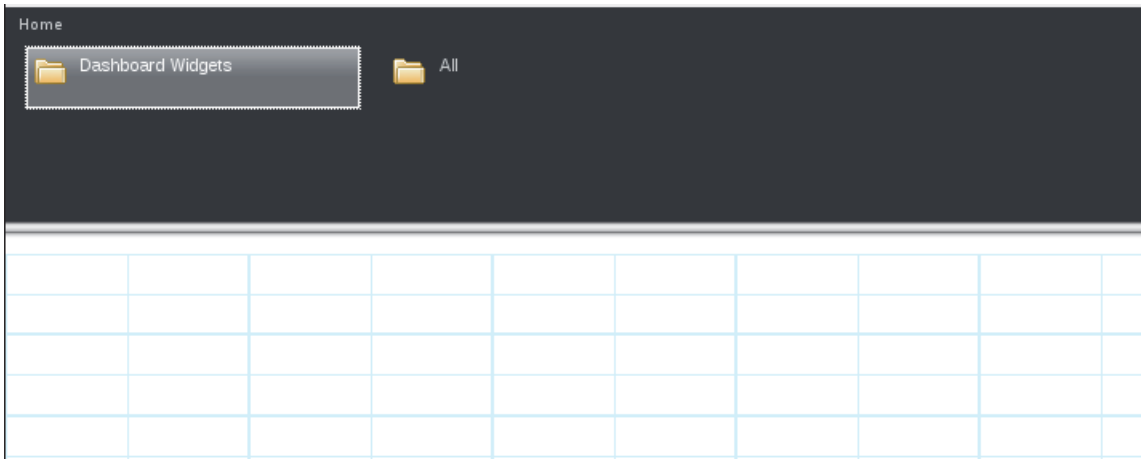
14. Click **New Page**.



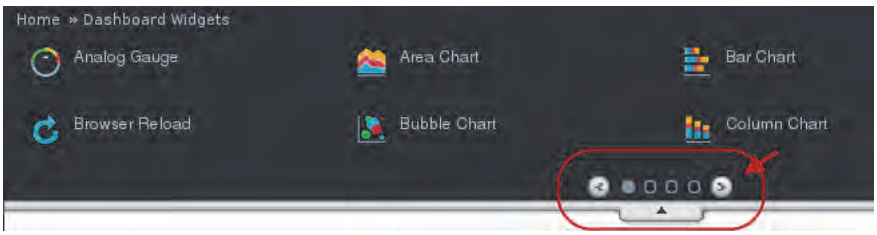
15. Type **ImpactLocationPage** in the **Page name** field and click **OK**.



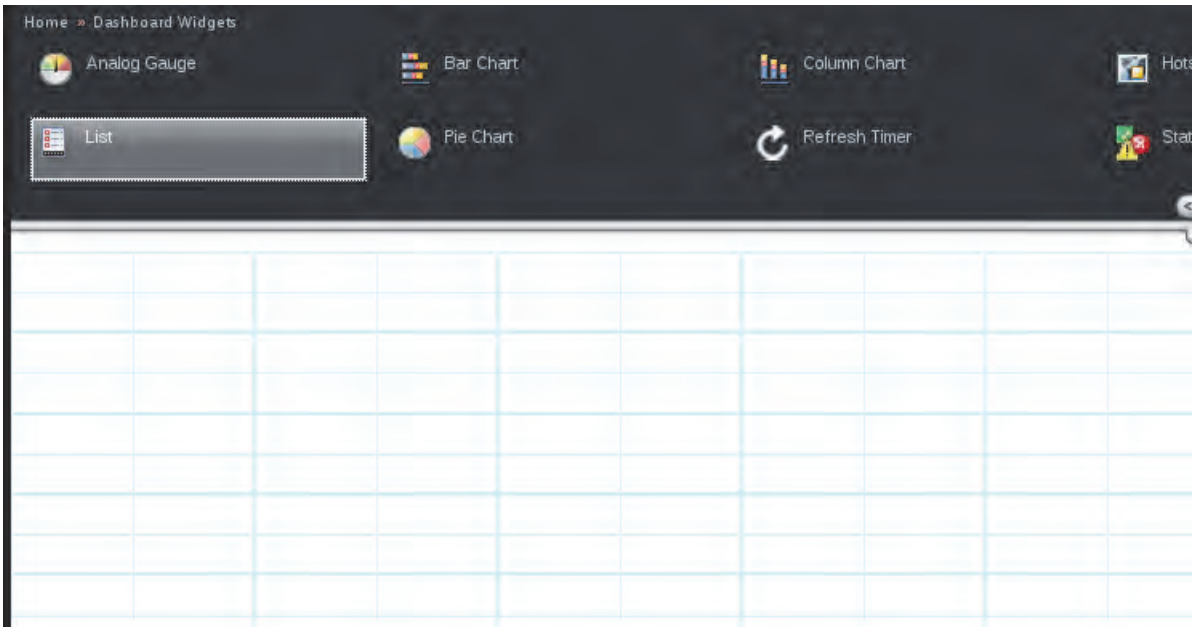
16. Click **Dashboard Widgets**.



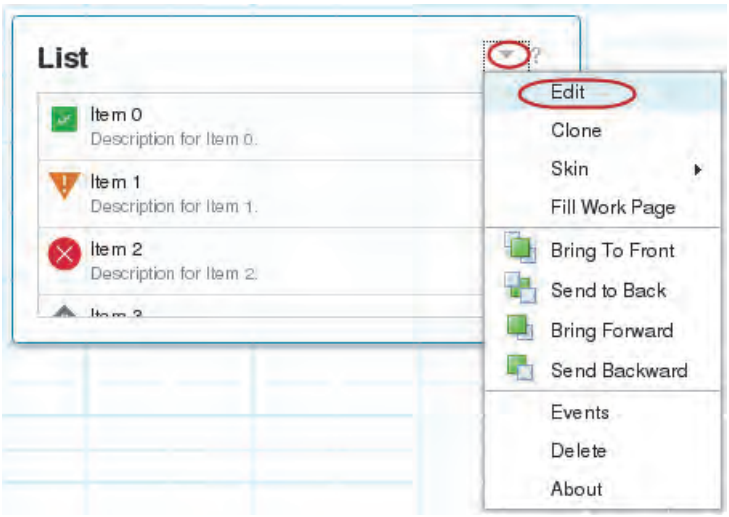
17. Find the **List** widget. Use the arrow to scroll right.



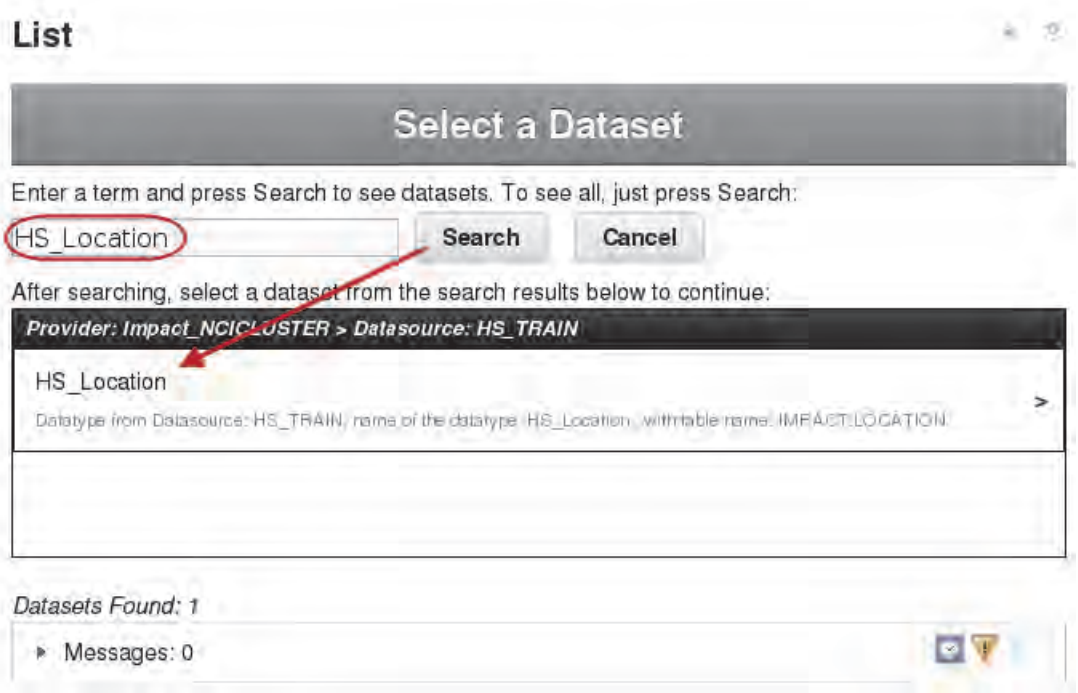
18. Drag the **List** widget to the page.



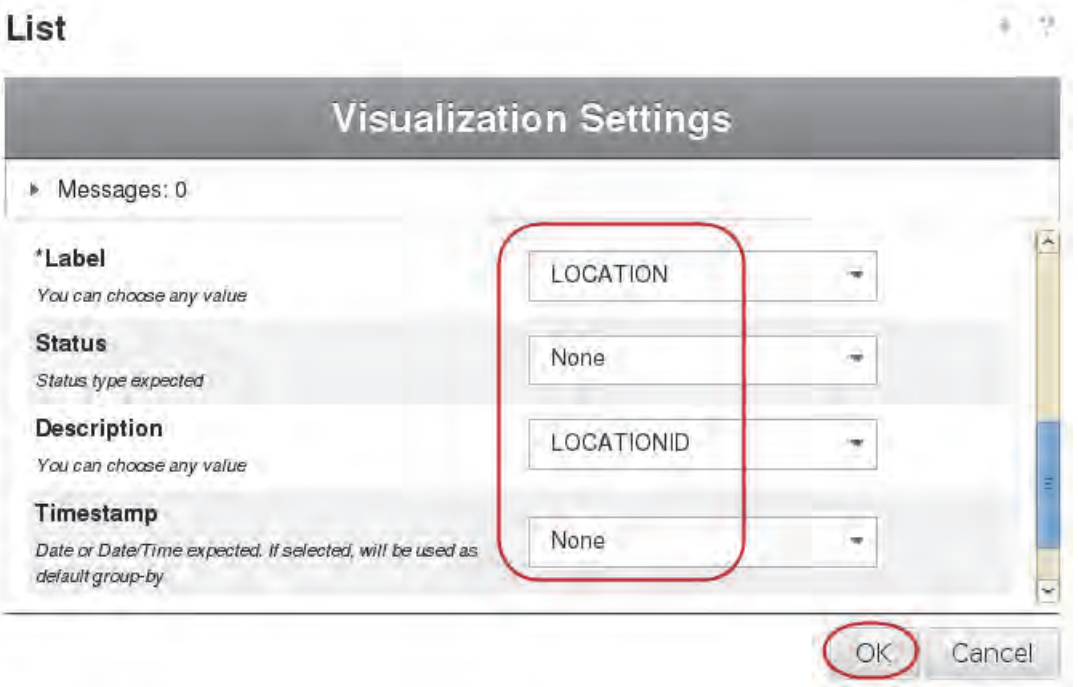
19. Click the arrow to edit the widget.



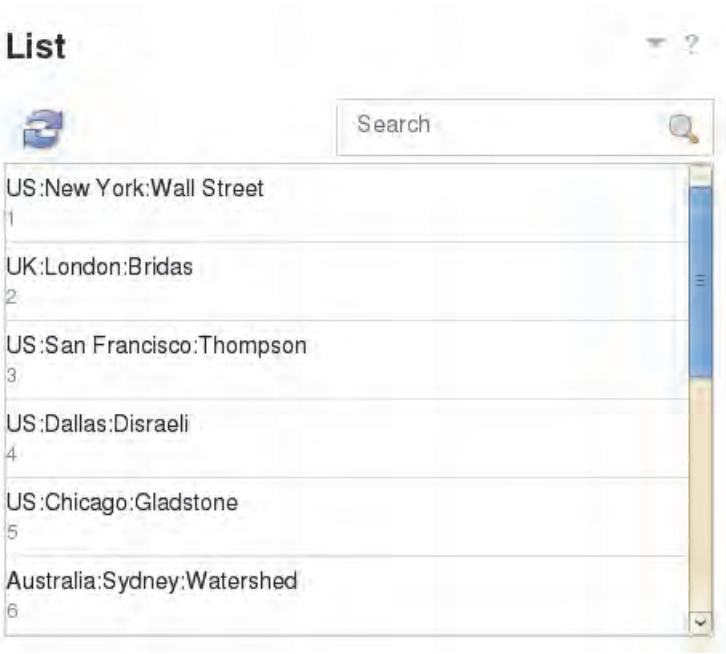
20. Type **HS_Location** in the search field. Click **HS_Location** in the list to select the data set.



21. Map the visualization attributes to the data set columns as shown. Click **OK**.

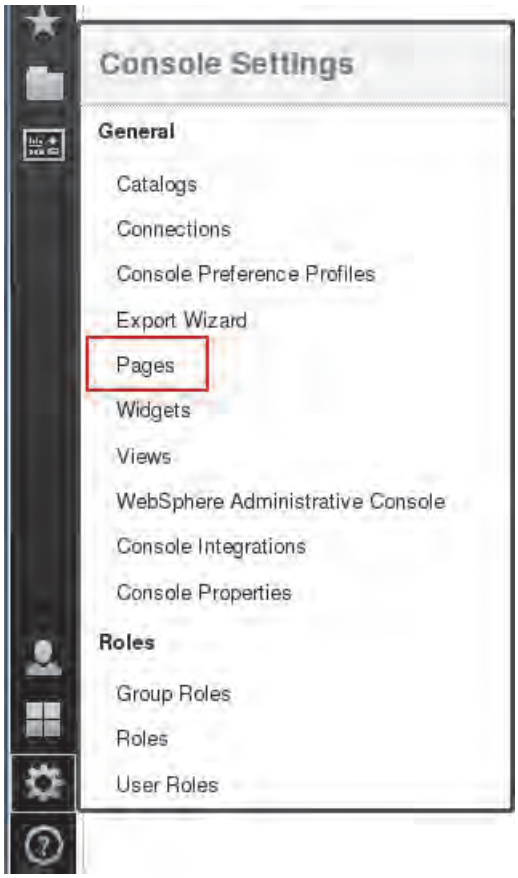


The resulting data is as shown in this screen capture.

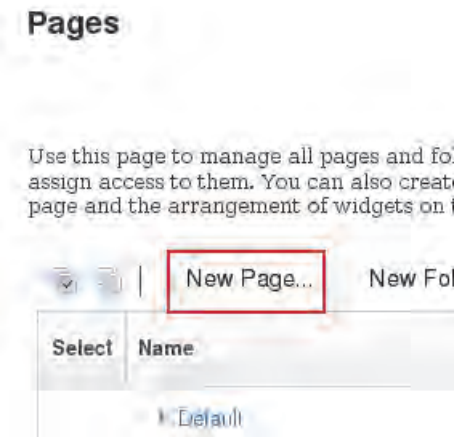


Exercise 3 Opening the Operator View using the Web Widget

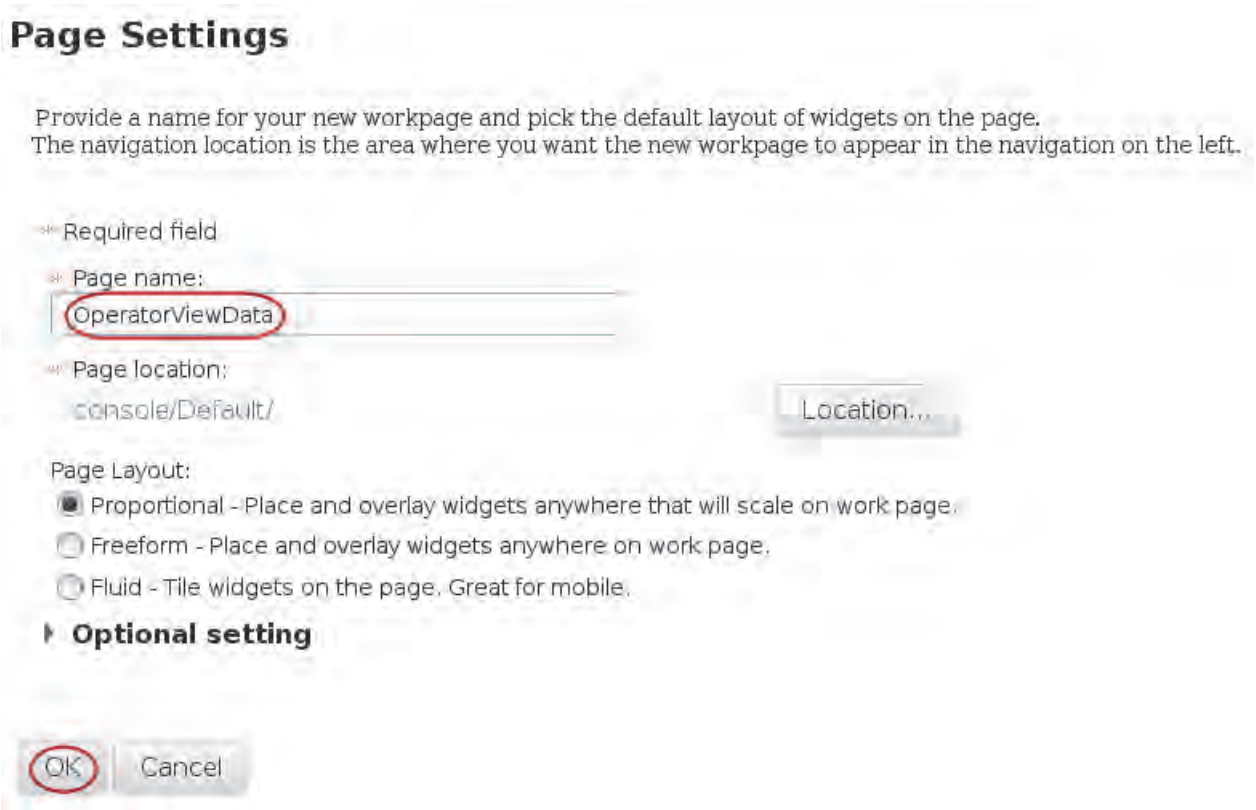
1. Build a new page by clicking **Console Settings > Pages** as shown.



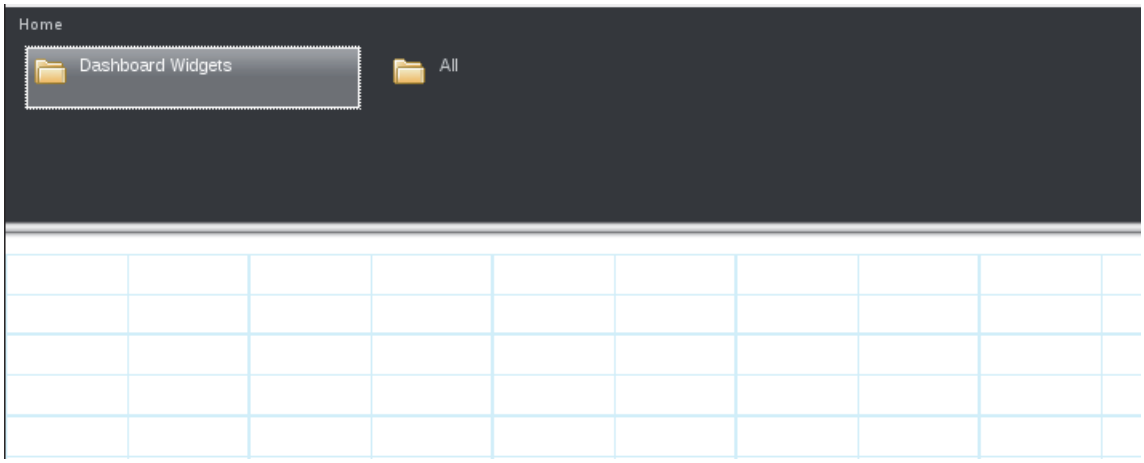
2. Click **New Page**.



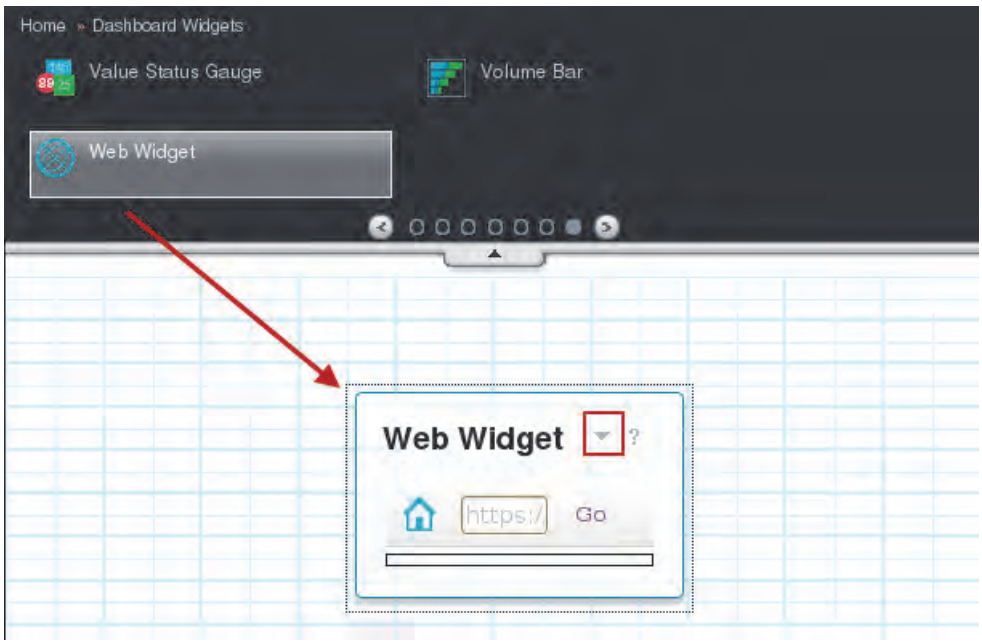
3. Type **OperatorViewData** in the **Page name** field and click **OK**.



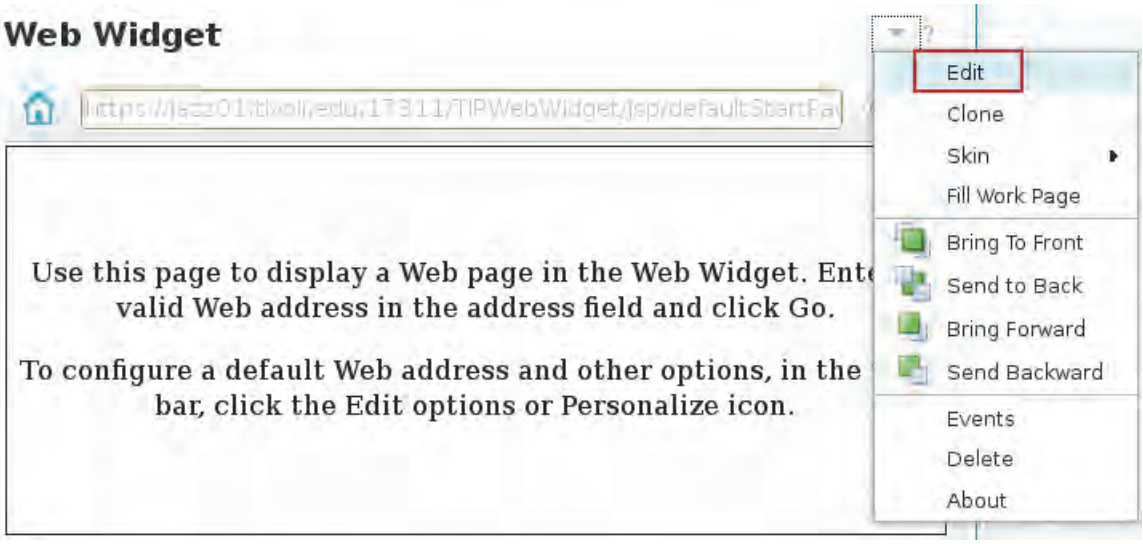
4. Click **Dashboard Widgets** to expand it.



5. Scroll to the right. Drag the **Web Widget** to the page.

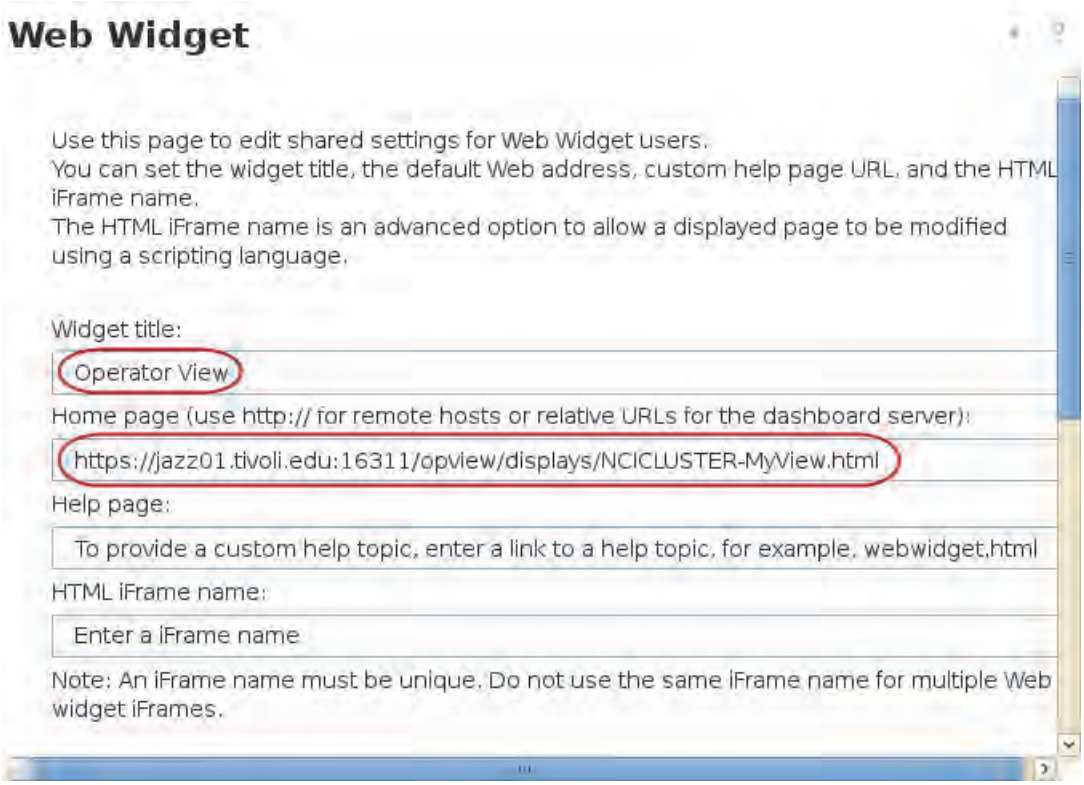


6. Click the arrow, and select **Edit**.

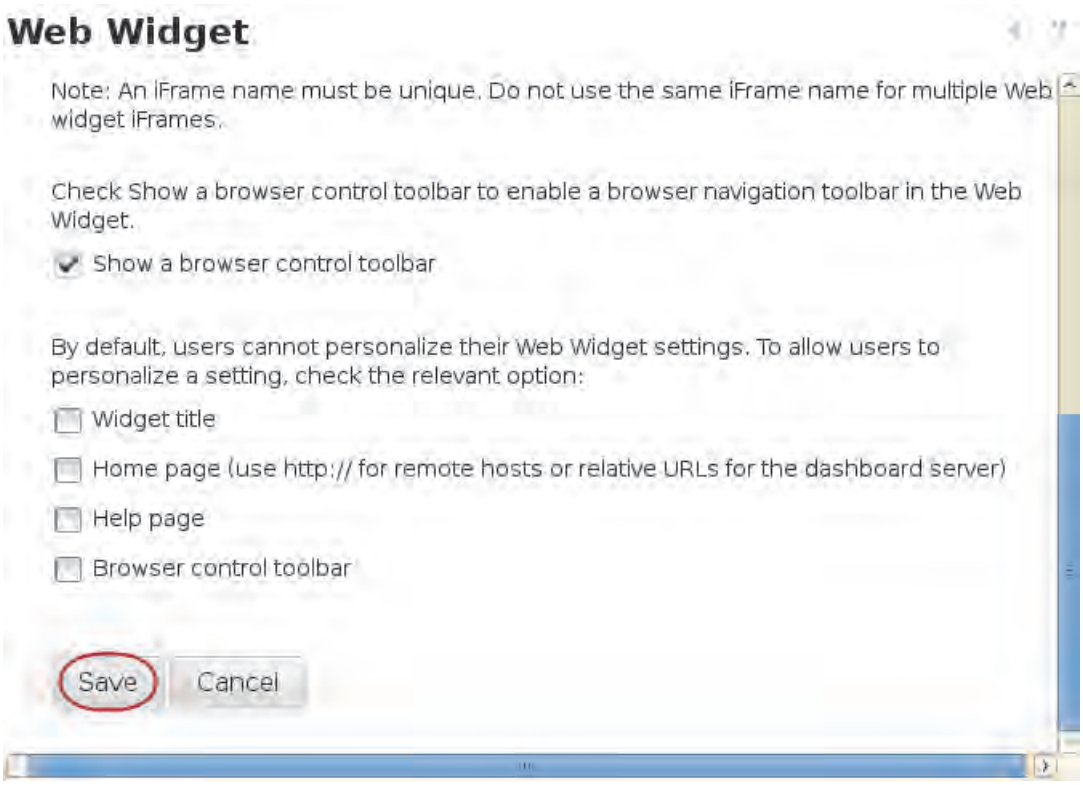


7. Enter the following data:

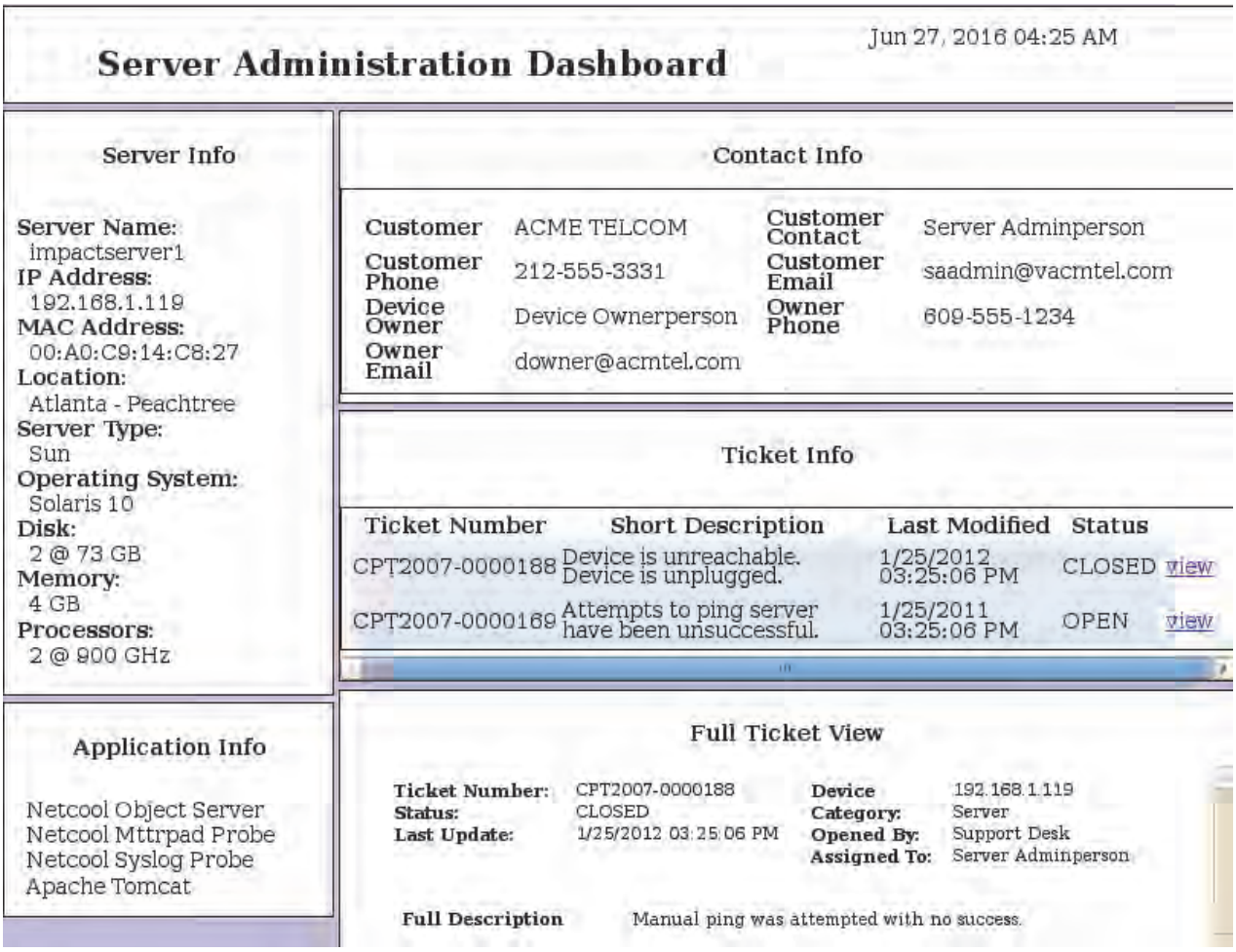
```
Widget Title: Operator View
Home Page (Operator View URL):
https://jazz01.tivoli.edu:16311/opview/displays/NCICLUSTER-MyView.html
```



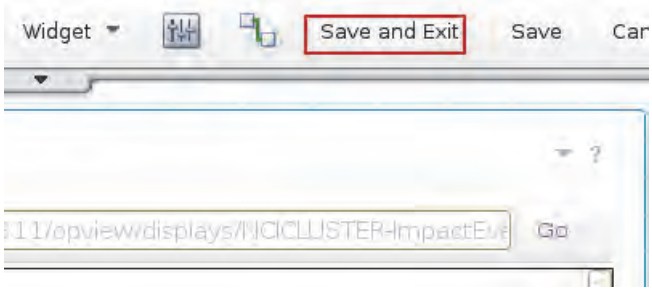
8. Scroll down to click **Save**.



The Web Widget shows in the Operator View URL.



9. Click **Save and Exit**.

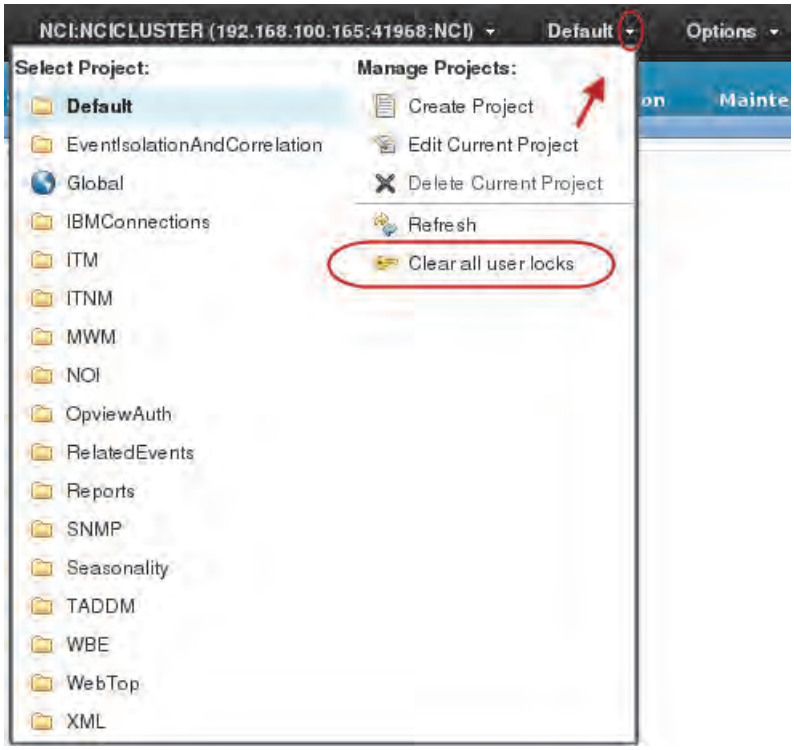


Unit 17 Server utilities exercises

In these exercises, you work with server utilities.

Exercise 1 Exporting Netcool/Impact server

1. Click the arrow next to **Default.** Select **Clear All user locks** to unlock the objects in Netcool/Impact server.



2. Click **OK** to close the window.

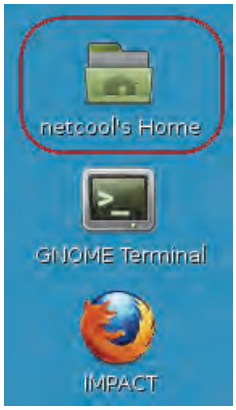
3. To export the Netcool/Impact objects from the server, open a Gnome Terminal and type the following commands.

```
cd /opt/IBM/tivoli/impact/bin/
./nci_export NCI /labfiles/NCI_exports/
```

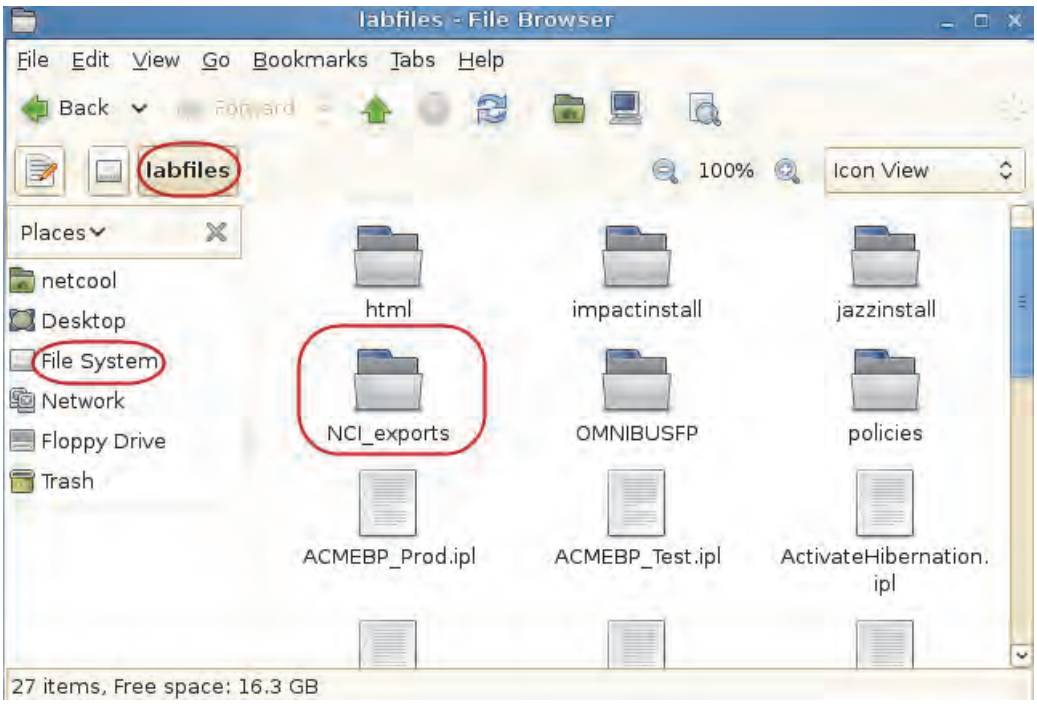
```
netcool@jazz01:/opt/IBM/tivoli/impact/bin> ./nci_export NCI /labfiles/NCI_export
s/
nci_export of NCI to /labfiles/NCI_exports completed.
netcool@jazz01:/opt/IBM/tivoli/impact/bin>
```

Note: **NCI_exports** is a directory. Make sure that the **NCI_exports** directory does not exist before you run the **export** command; otherwise the command will fail.

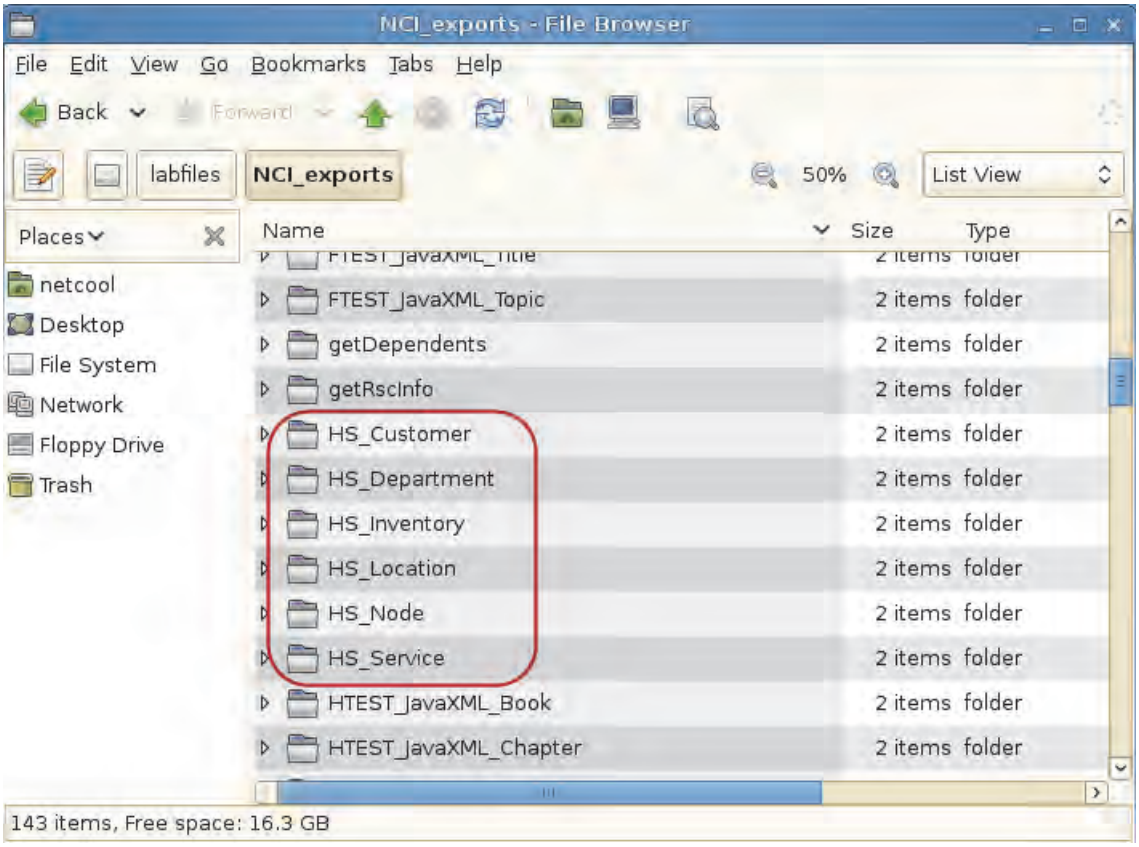
4. Open a file browser on your desktop.



5. View the contents of the labfiles/**NCI_export** directory.



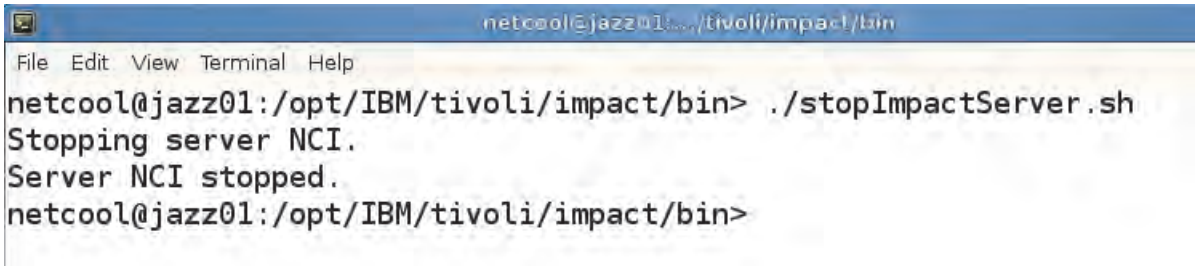
6. Verify that your previously created policies, data sources, and other elements exist in the directory.



7. Stop the Impact server using the following commands.

```
cd /opt/IBM/tivoli/impact/bin/
./stopImpactServer.sh
```

8. Look for the following messages in the output:



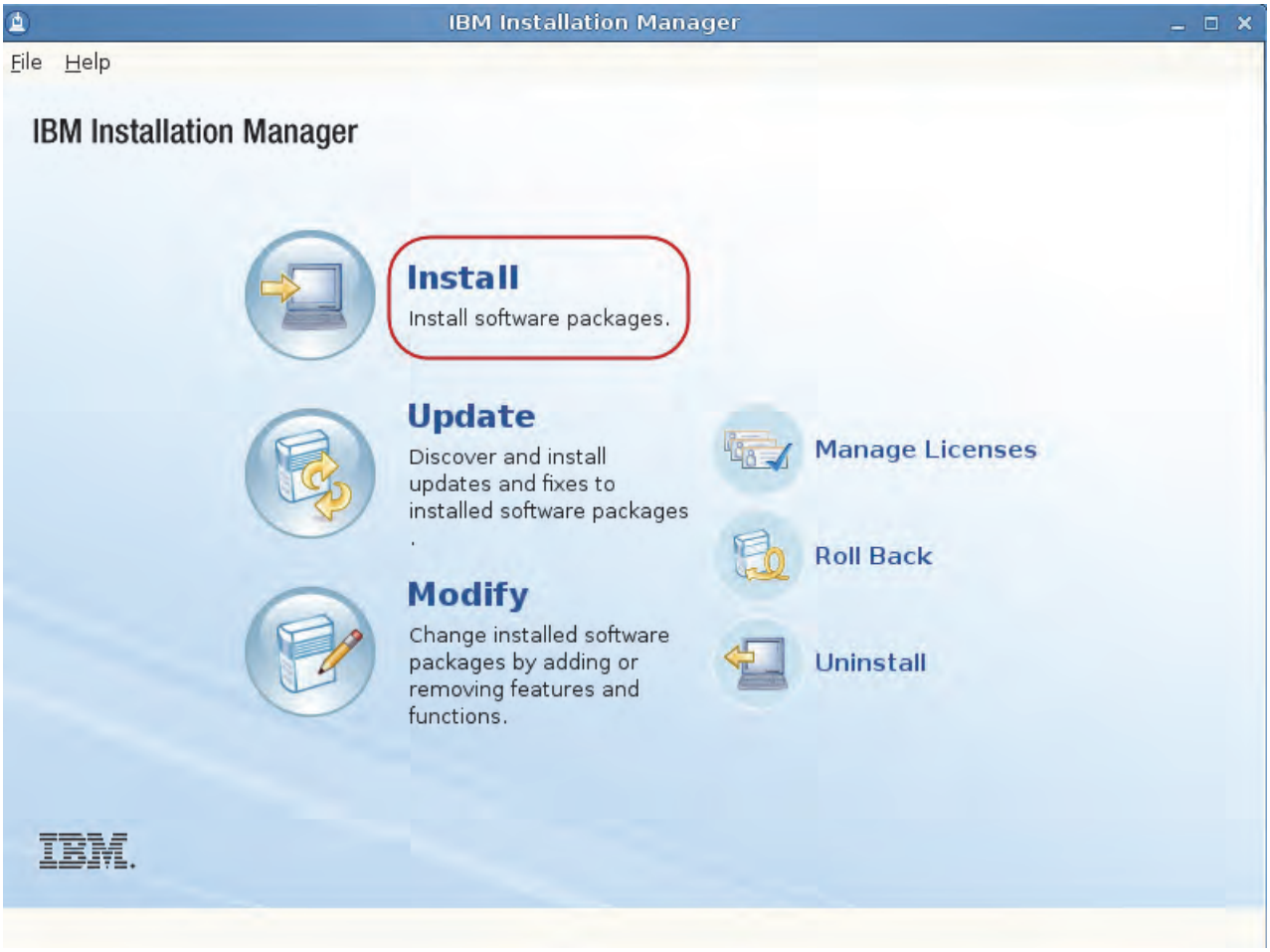
Exercise 2 Creating a second server instance

1. Use the following command to start the Installation Manager tool:

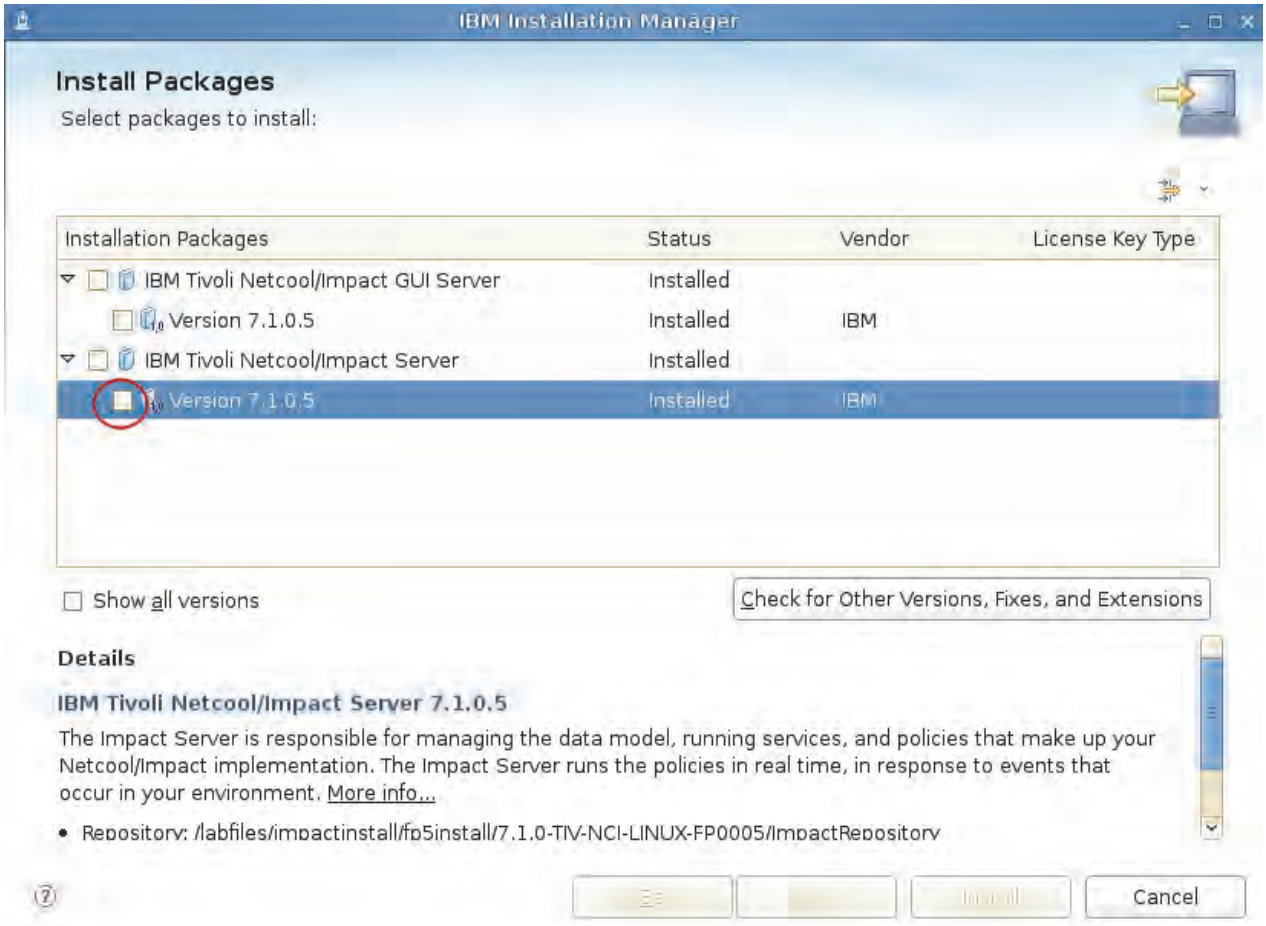
   ```
   cd /home/netcool/IBM/InstallationManager/eclipse/
   ./IBMIM
   ```

2. Add the installation repositories. Select **File > Preferences** add the following package:

   `/labfiles/impactinstall/ImpactRepository/disk1/diskTag.inf`

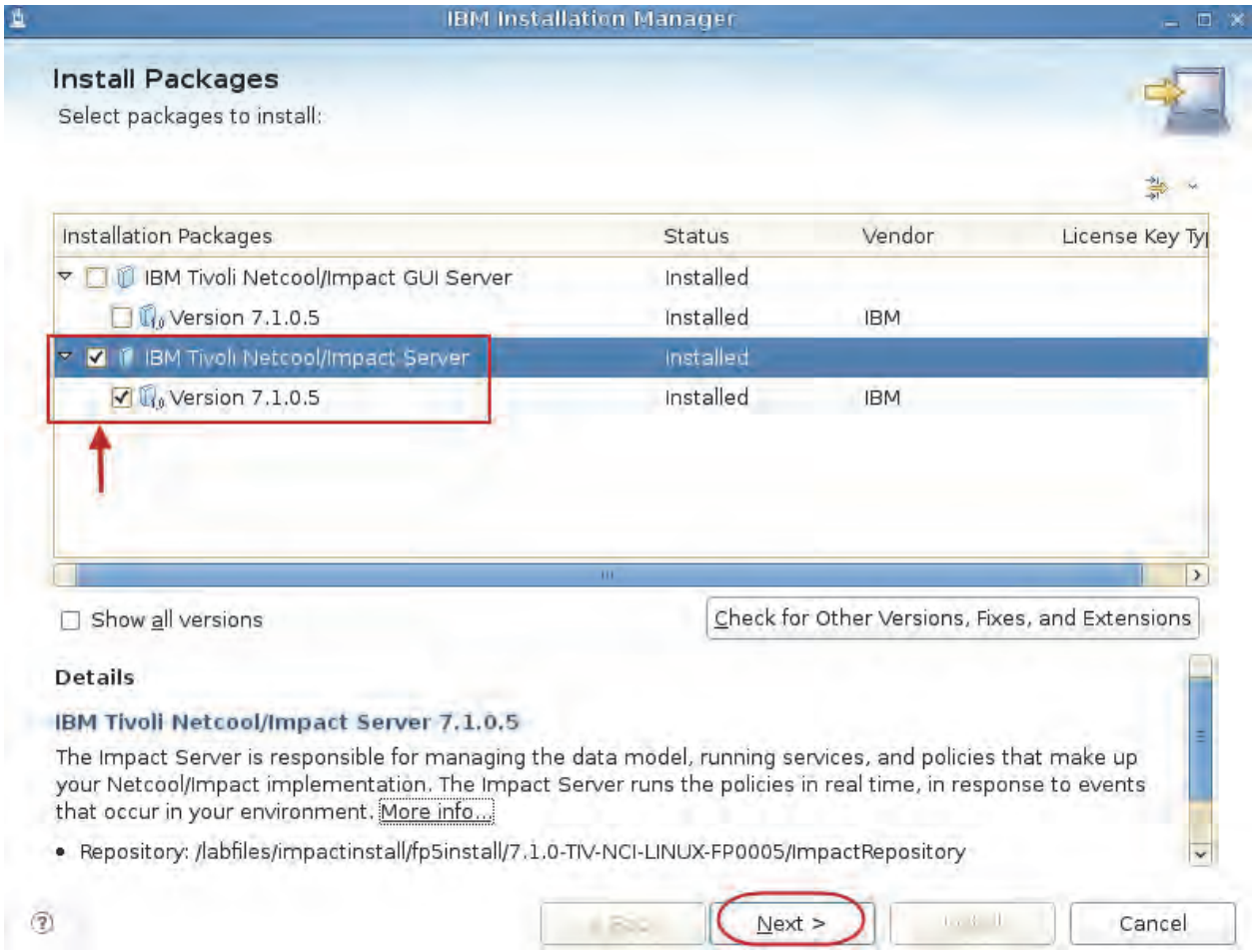


3. Select IBM Tivoli Netcool/Impact Server version 7.1.0.5.

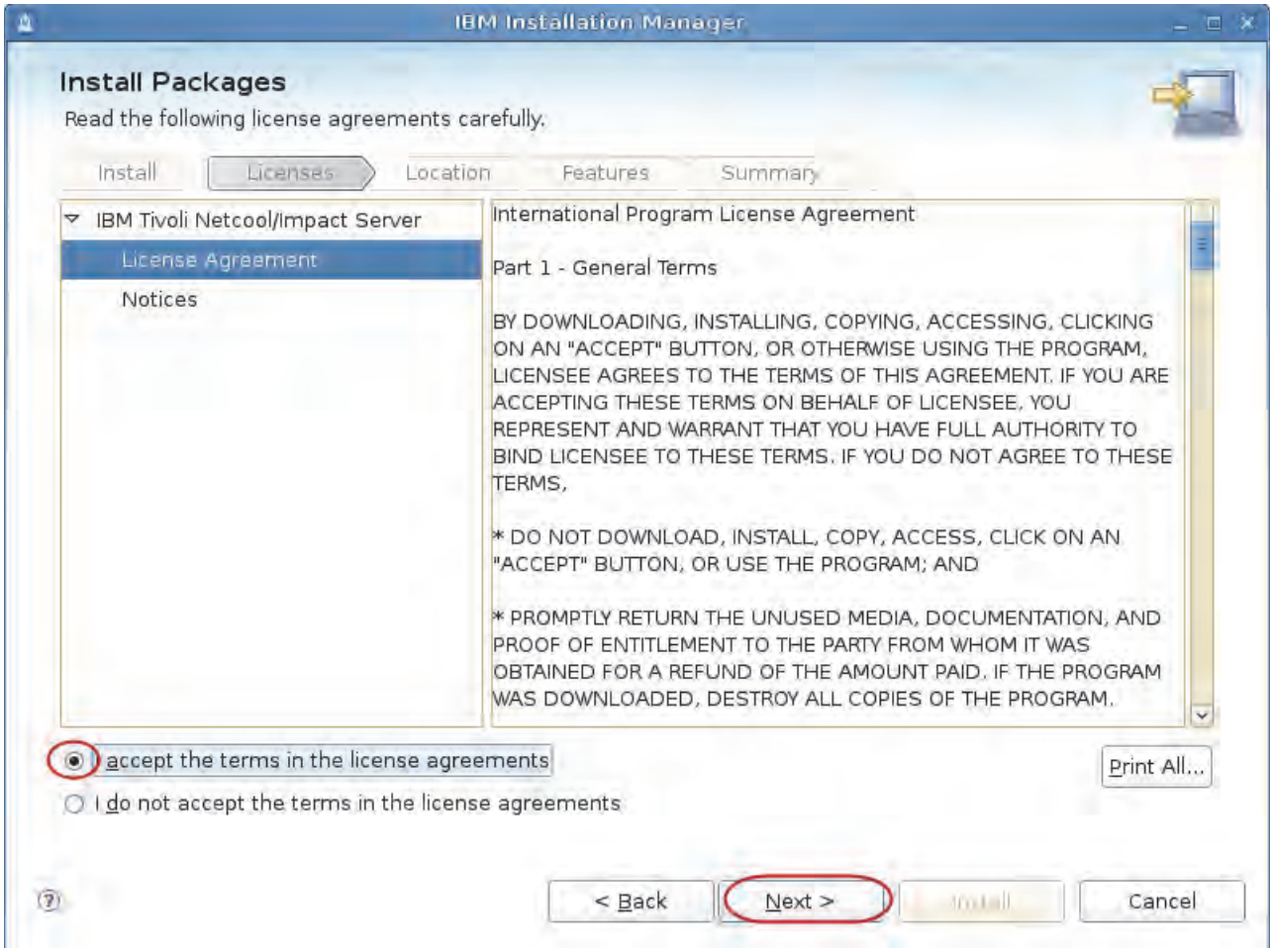


4. Click **Continue** to install the package to a new group.

5. Install packages. Click **Next**.

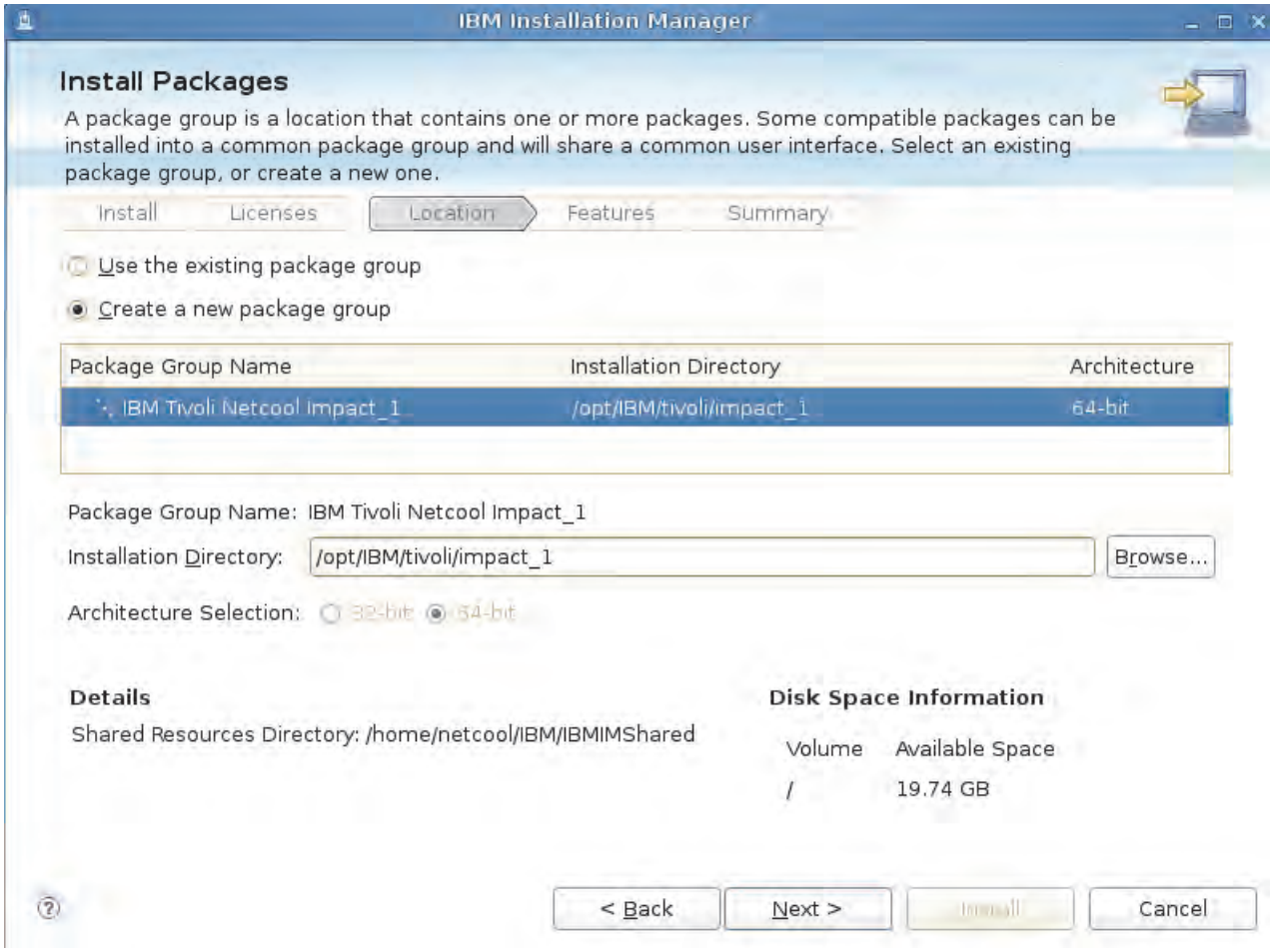


6. Accept the license agreement. Click **Next**.

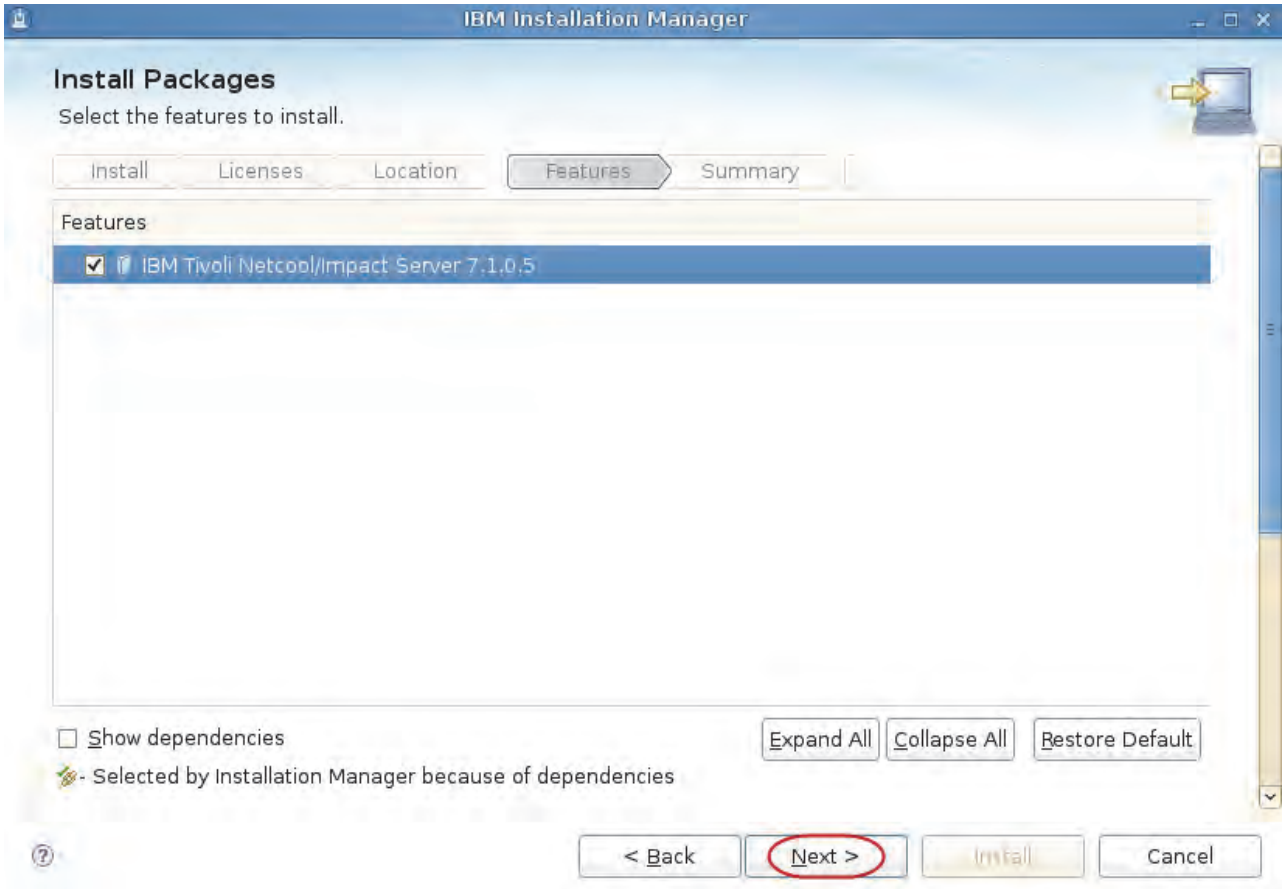


Notice the Installation Directory.

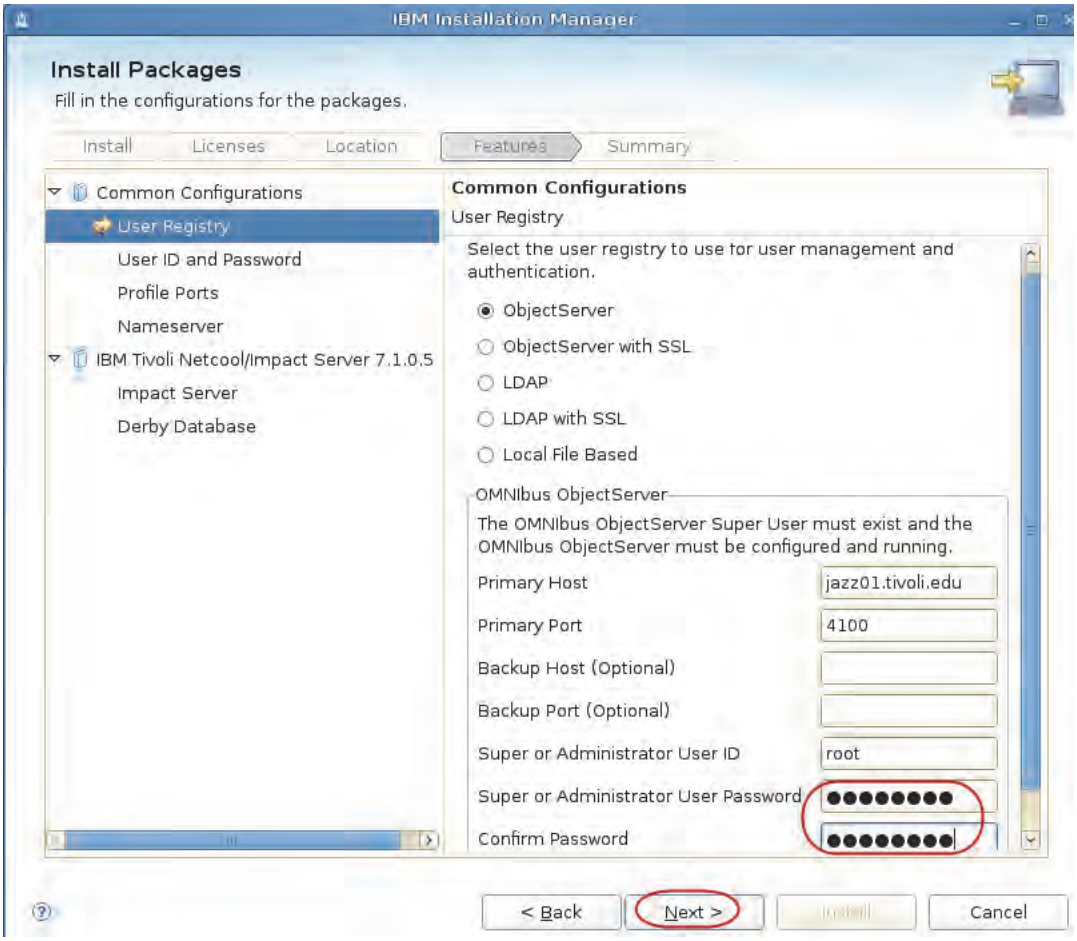
7. Click **Next**.



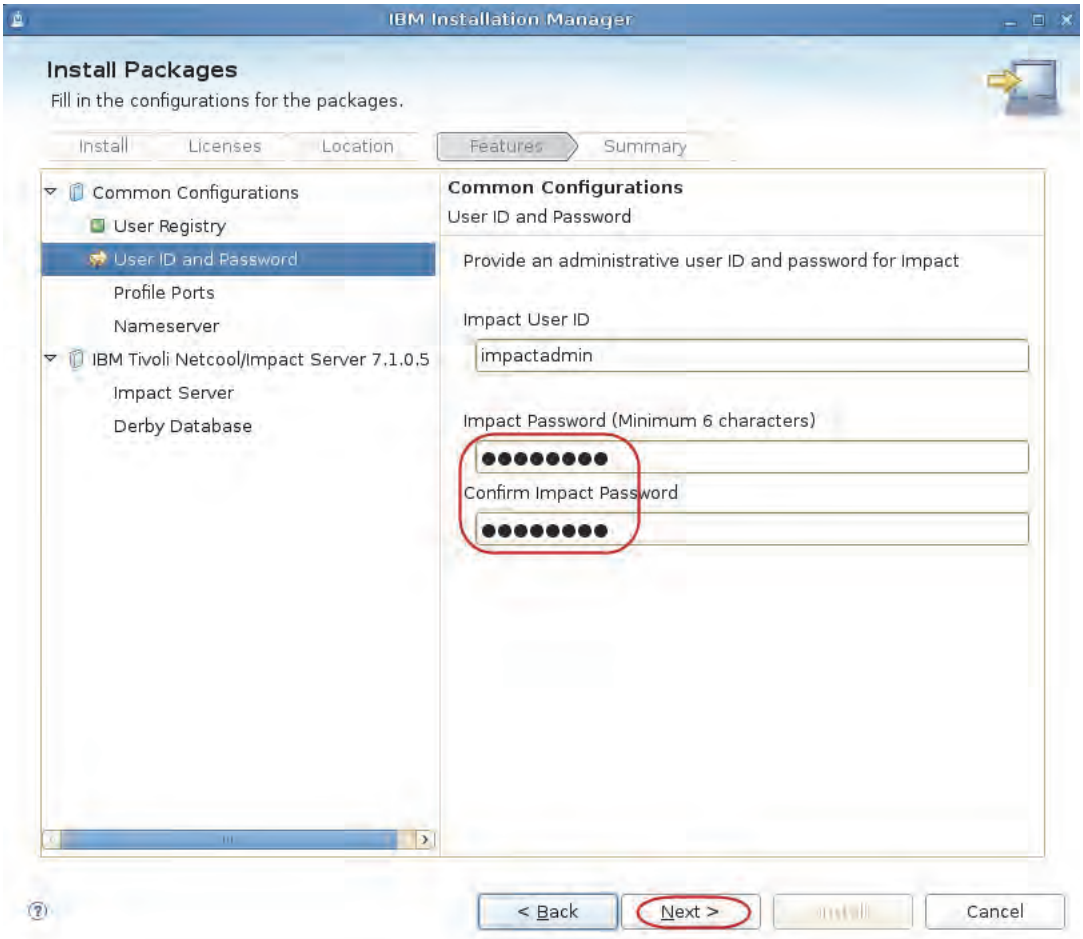
8. Accept the defaults and click **Next**.



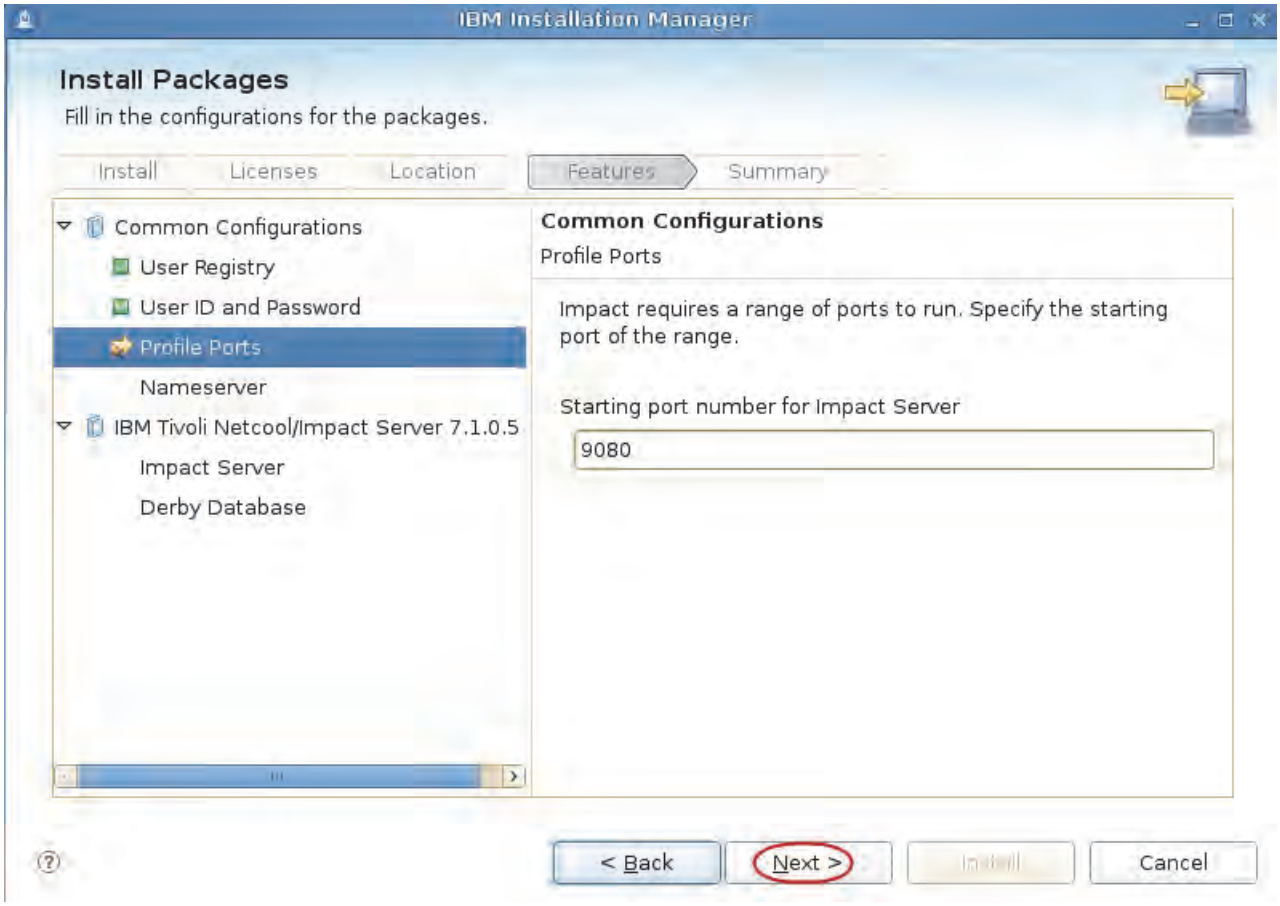
9. Type the password **object00**. Click **Next**.



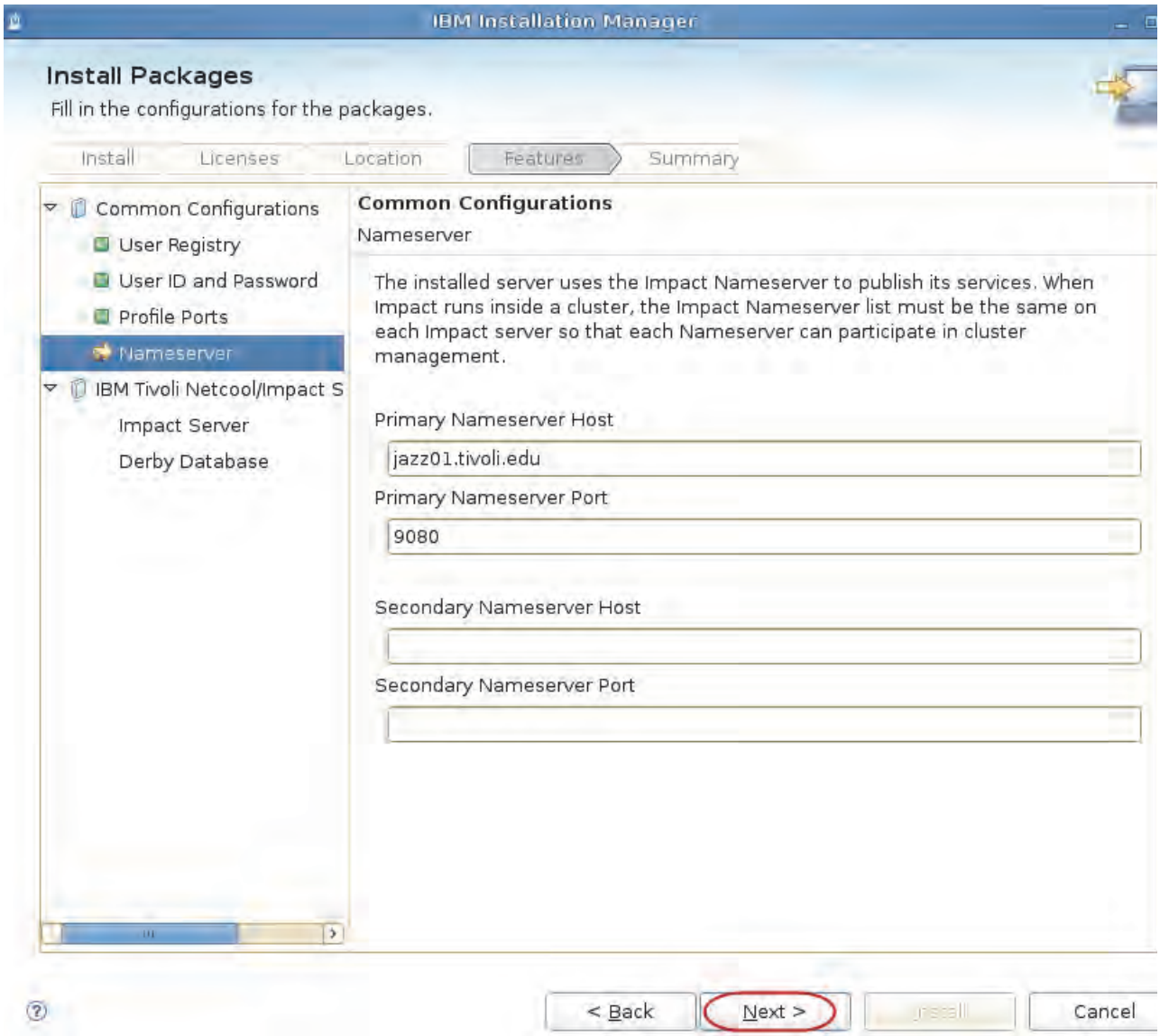
10. Type **object00** as the Impact password.



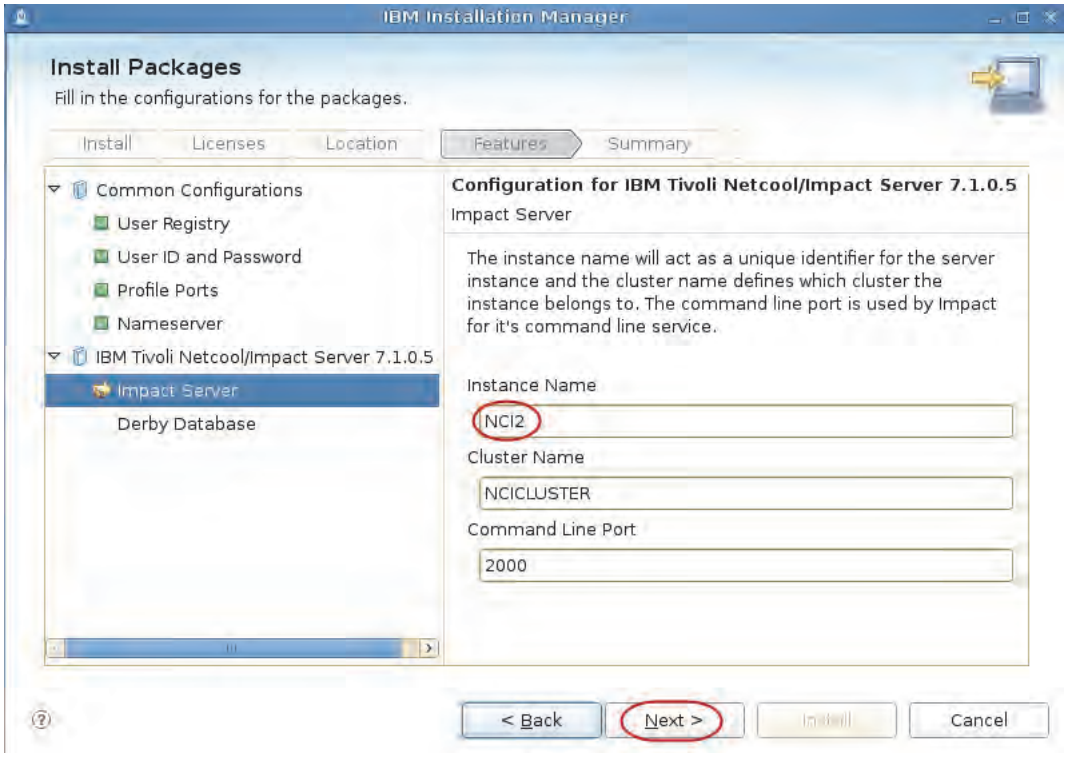
11. Accept the default port and click **Next**.



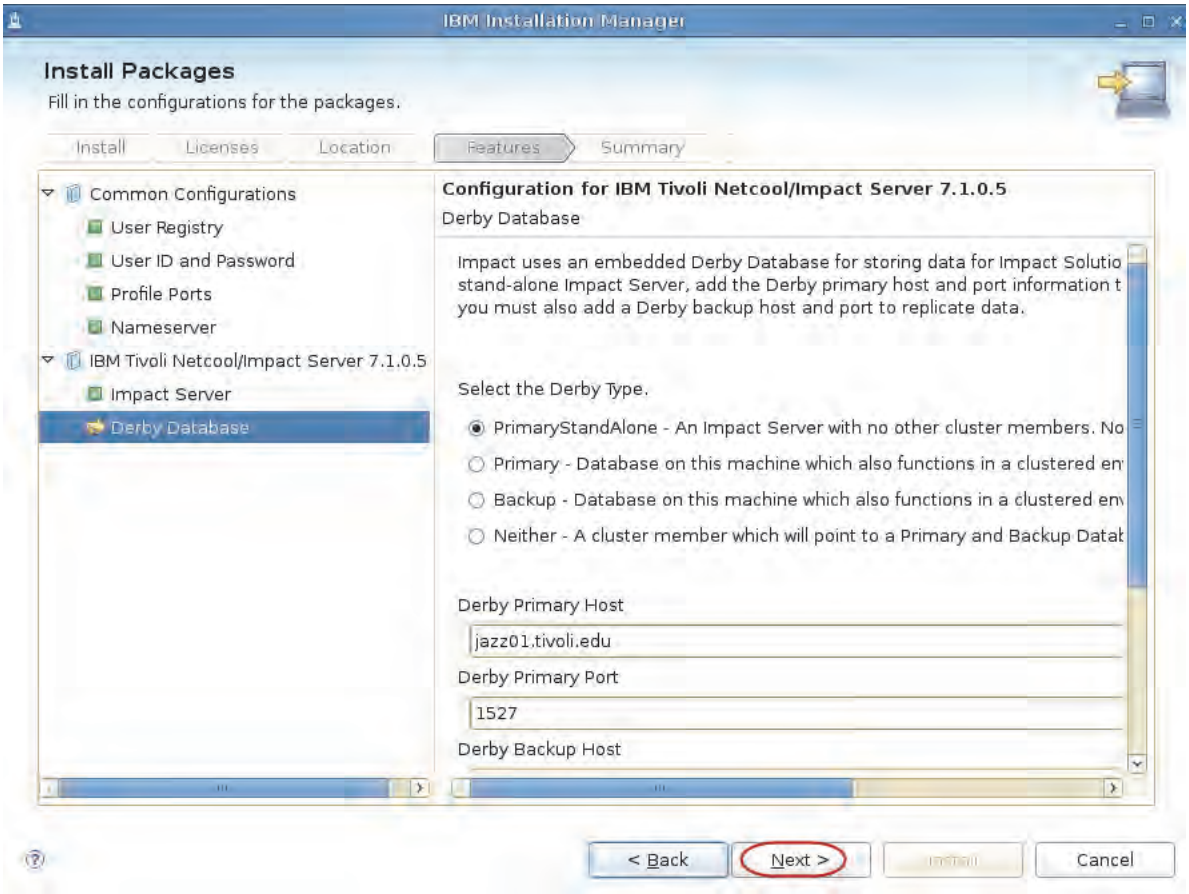
12. Accept the defaults. Click **Next**.



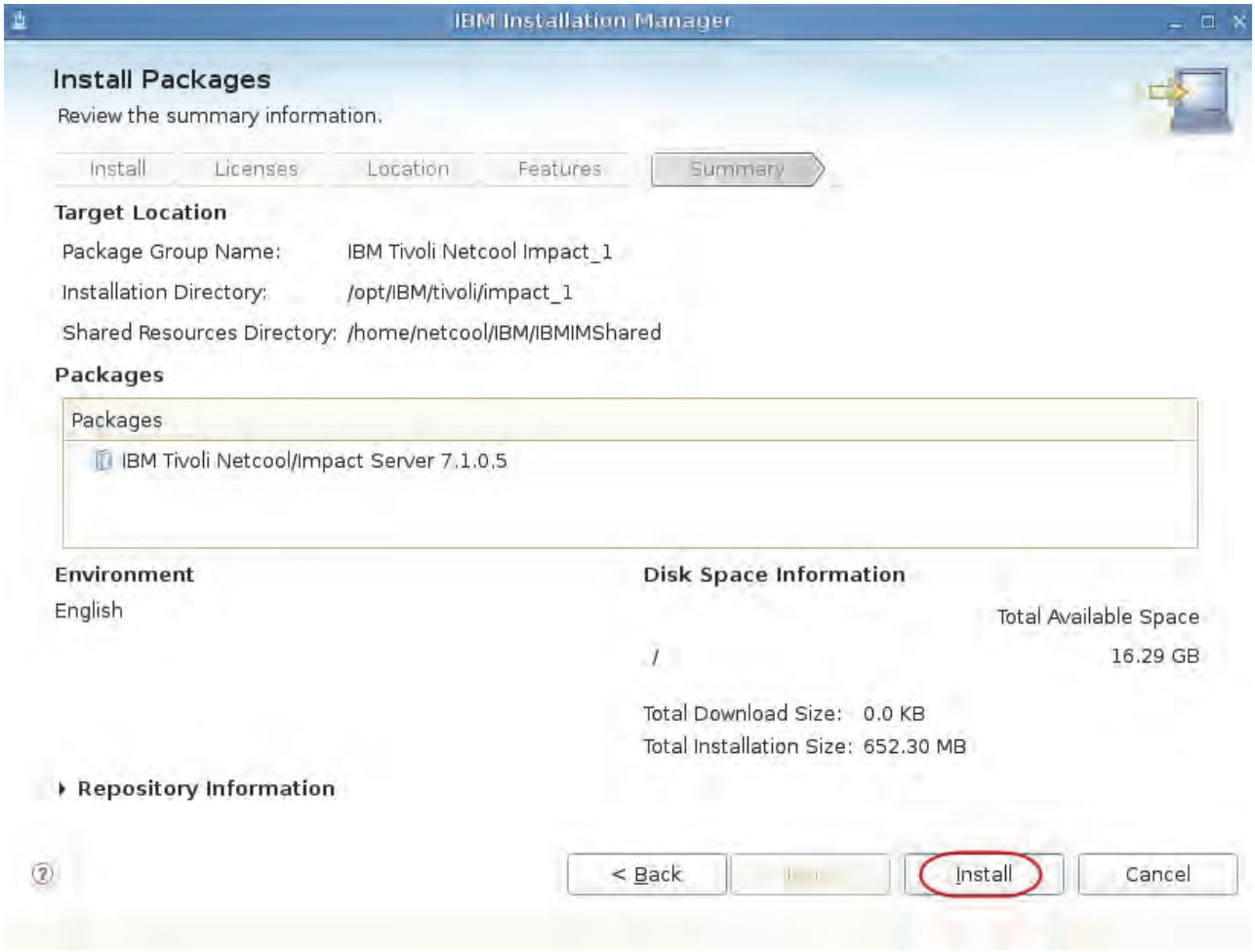
13. Type **NCI2** as the new Instance Name. Click **Next**.



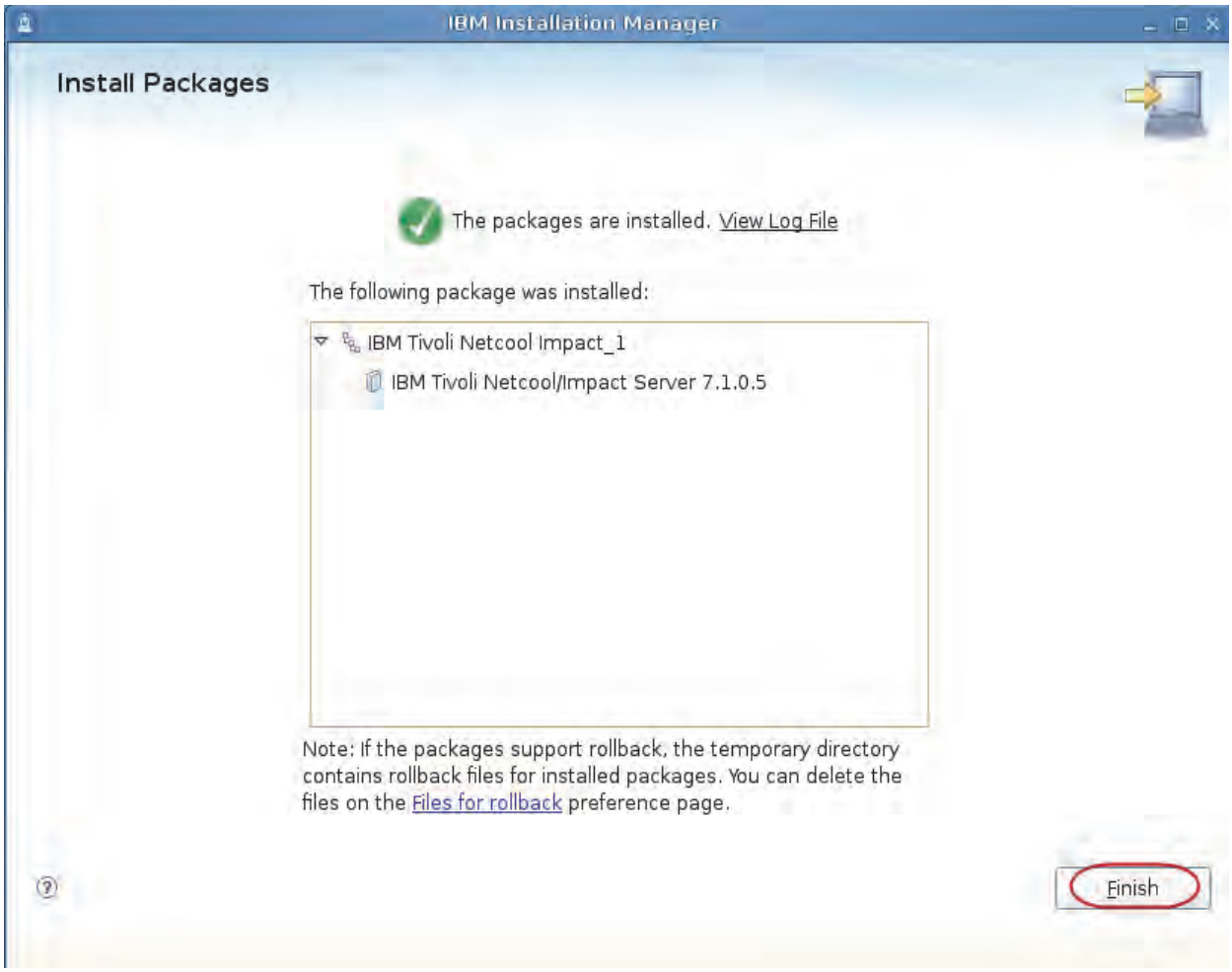
14. Accept the defaults and click **Next**.



15. Click **Install**.



16. Click **Finish** to complete the install.



17. Verify OMNIbus is running, type the following commands:

```
cd /opt/IBM/tivoli/netcool/omnibus/bin/
./nco_objserv&
```

18. To verify the GUI server is running, type the following commands:

```
cd /opt/IBM/tivoli/impact/bin/
./startGUIServer.sh
```

```
netcool@jazz01:/opt/IBM/tivoli/impact/bin> ./startGUIServer.sh
Starting server ImpactUI.
Server ImpactUI started with process ID 6457.
netcool@jazz01:/opt/IBM/tivoli/impact/bin>
```

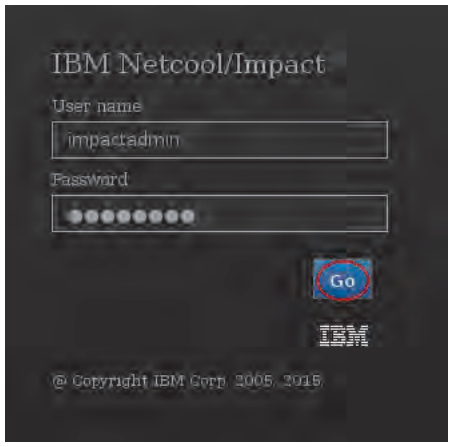
19. To verify the **NCI2** instance has started, type the following commands:

```
cd /opt/IBM/tivoli/impact_1/bin/
./startImpactServer.sh
```

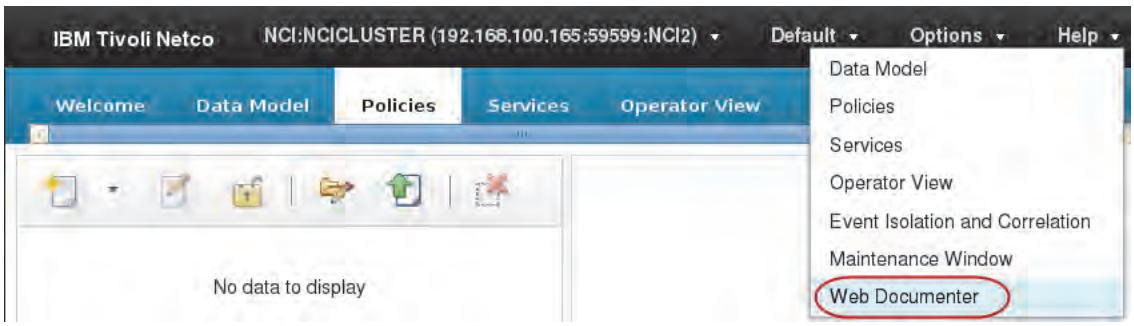
```
netcool@jazz01:/opt/IBM/tivoli/impact_1/bin> ./startImpactServer.sh
Server NCI2 is already running with process ID 31733.
netcool@jazz01:/opt/IBM/tivoli/impact_1/bin>
```

20. Click the IMPACT icon on the desktop.

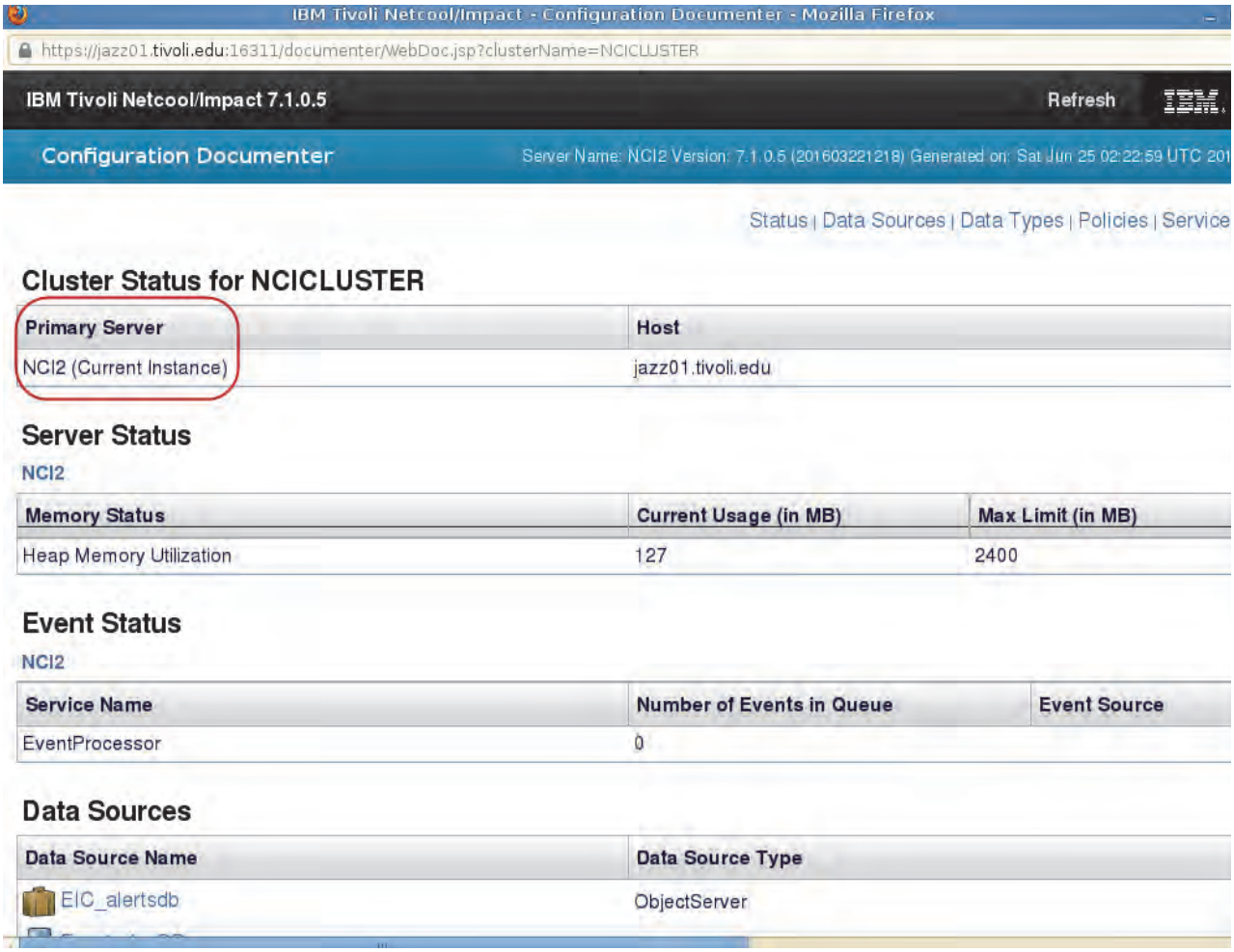
21. Type the User name **impactadmin**, and Password **object00.**



22. Click the **Help** tab and select **Web Documenter** to open the **Configuration Documenter**.



23. Notice the **Primary Server** name. Close the Configuration Documenter.



24. To verify the primary server using the command line, enter the following commands:

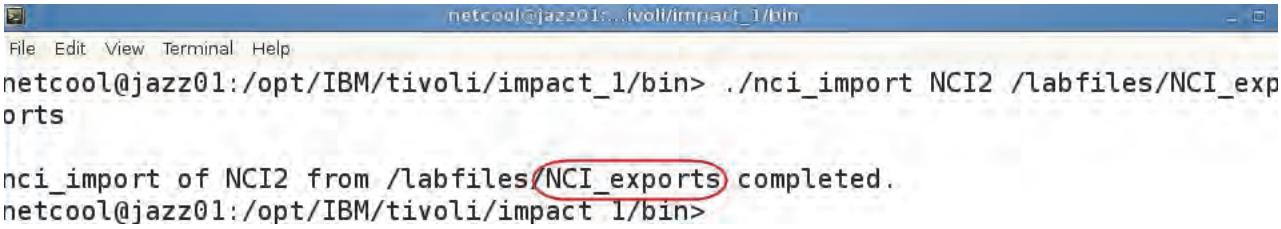
```
cd /opt/IBM/tivoli/impact_1/bin/
./nci_get_primary http jazz01.tivoli.edu 9080 NCICLUSTER
```

```
Press [ENTER] for options
http://jazz01.tivoli.edu:9080/nameserver/services-->
Get Binding

Enter product: Enter service: *** Getting binding
    > getBinding(productbase, servicebase)
        productbase: Impact
        servicebase: NCICLUSTER
    < Result returned
        binding: 192.168.100.165:59599:NCI2
    DONE


Press [ENTER] for options
http://jazz01.tivoli.edu:9080/nameserver/services-->
Quit

*** Exiting

netcool@jazz01:/opt/IBM/tivoli/impact_1/bin>
```

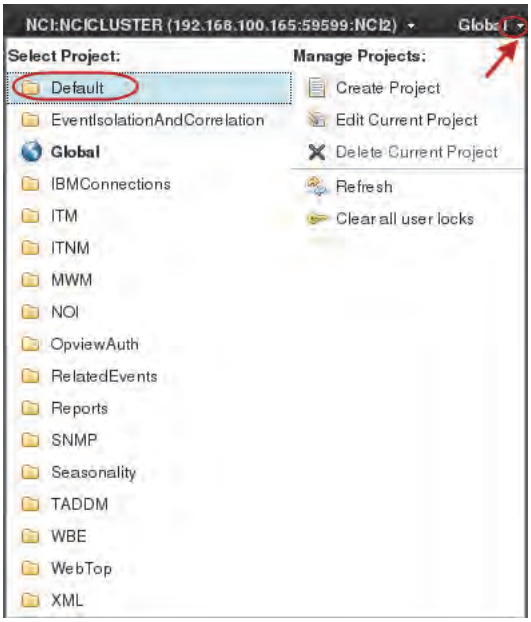
Exercise 3 Importing Netcool/Impact Server

1. Use the import command to load the policies, data model, and services using the following command.

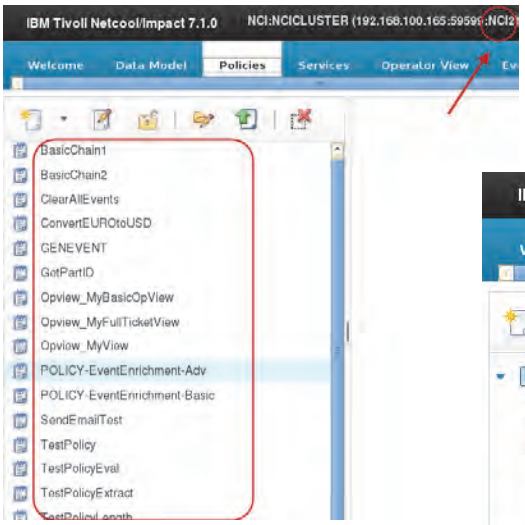
```
cd /opt/IBM/tivoli/impact_1/bin/
./nci_import NCI2 /labfiles/NCI_exports
This process will take a few minutes to complete.
```

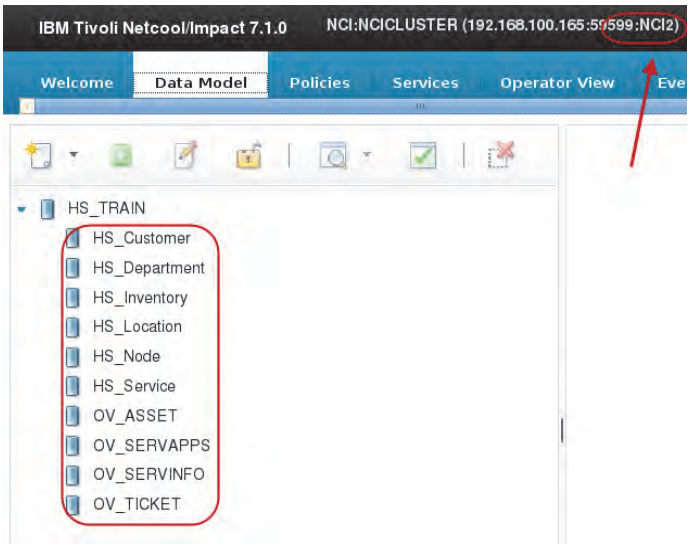


2. Switch to the **Default** project, if the **Global** project is showing.



3. Verify that the Policies, Data Model, Services, and Operator Views have been imported properly from the backup.



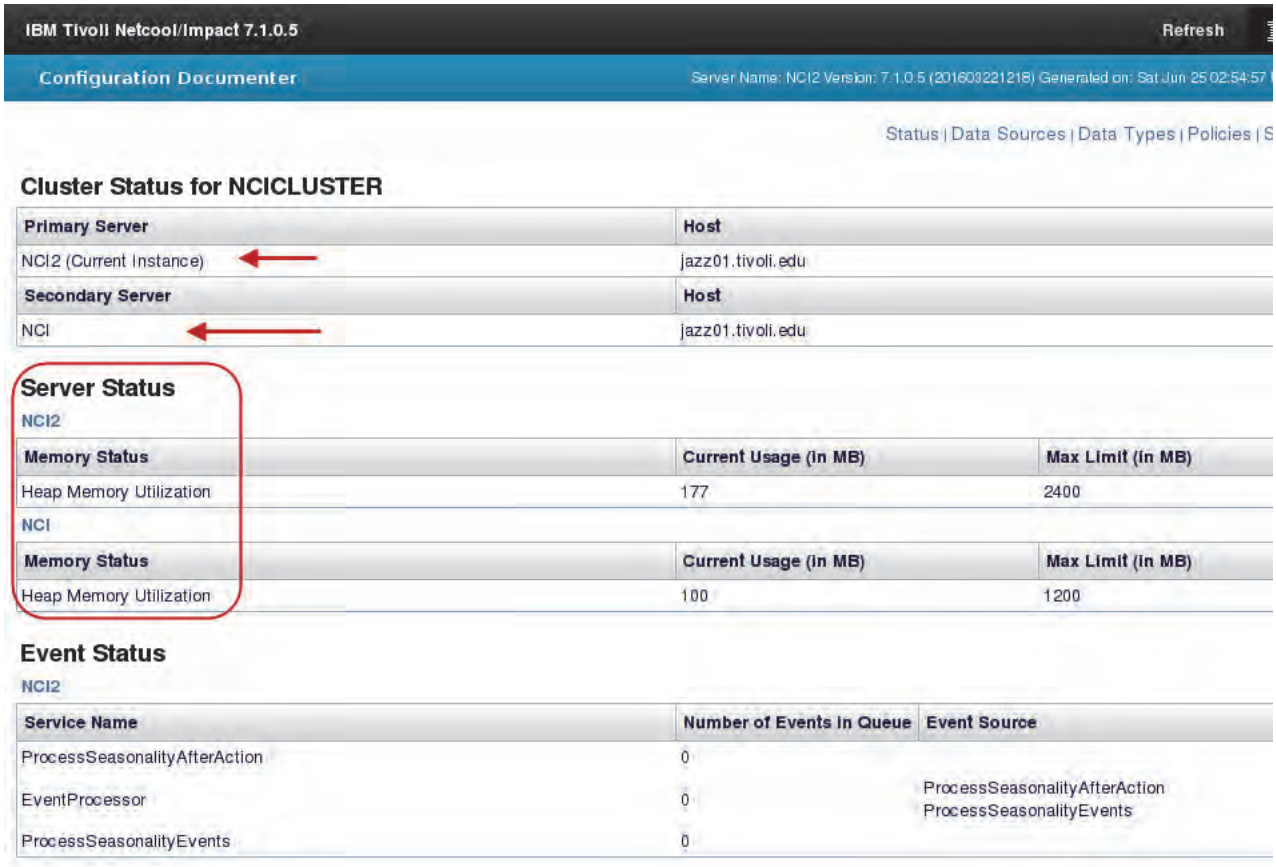


4. Restart the **NCI** server instance using the following commands.

```
cd /opt/IBM/tivoli/impact/bin/
./startImpactServer.sh
```

```
netcool@jazz01:/opt/IBM/tivoli/impact/bin> ./startImpactServer.sh
Starting server NCI.
Server NCI started with process ID 14210.
netcool@jazz01:/opt/IBM/tivoli/impact/bin>
```

5. Click the **Help** tab and select **Web Documenter** to display the **Configuration Documenter**. Notice that **NCI** is now the secondary server.



IBM Training