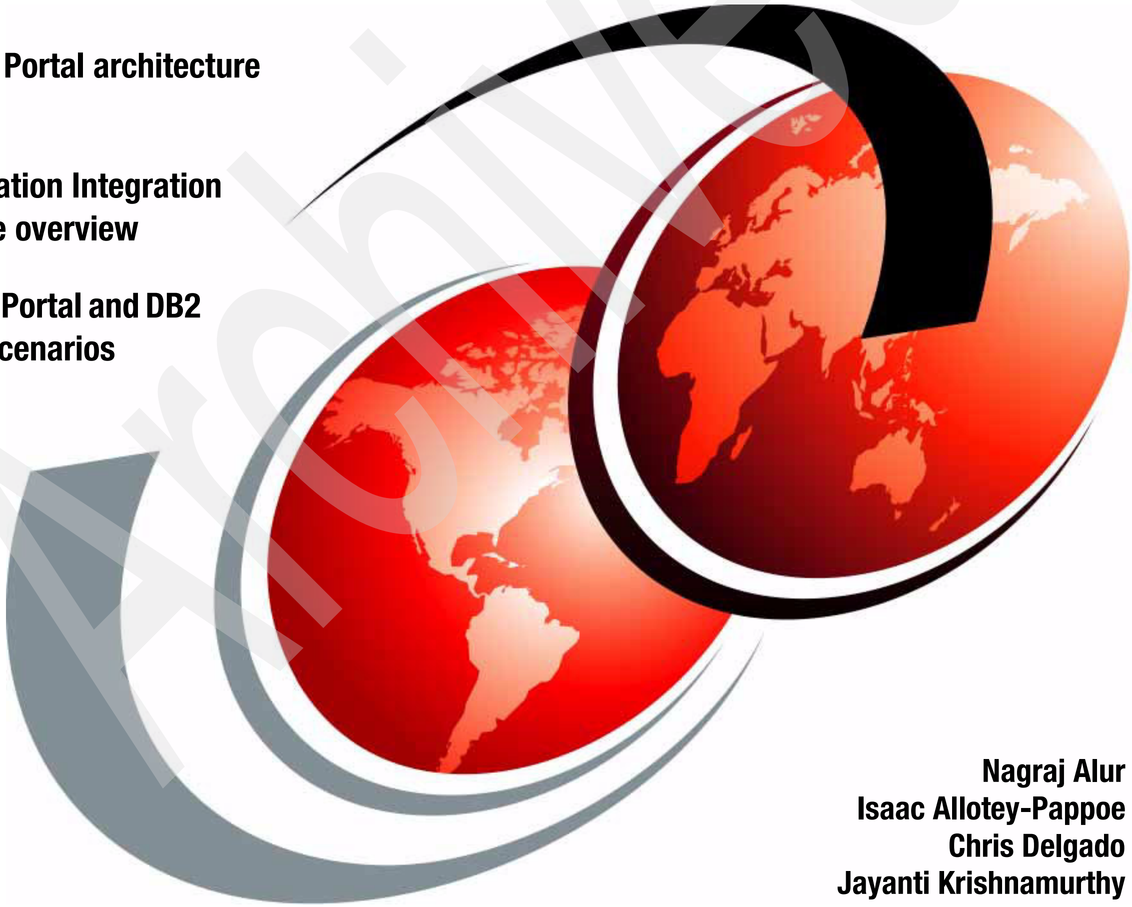
IBM

WebSphere Portal and DB2 Information Integrator:
A Synergistic Solution

WebSphere Portal architecture overview

DB2 Information Integration architecture overview

WebSphere Portal and DB2 II synergy scenarios



Nagraj Alur
Isaac Allotey-Pappoe
Chris Delgado
Jayanti Krishnamurthy

Redbooks

ibm.com/redbooks

International Technical Support Organization

WebSphere Portal and DB2 Information Integrator: A Synergistic Solution

March 2004

SG24-6433-00

Note: Before using this information and the product it supports, read the information in "Notices" on page xvii.

First Edition (March 2004)

This edition applies to Version 1, Release 1, Modification 2 of IBM DB2 Information Integrator (product number 5724-C74).

Contents

Figures

Tables

Examples

Notices

This information was developed for products and services offered in the U.S.A.

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 give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

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 Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites 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.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

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

ibm.com®
iSeries™
z/OS®
AIX®
Domino Designer®
Domino.Doc®
Domino®
DB2 Connect™
DB2 Universal Database™
DB2®
DRDA®
Informix®
Intelligent Miner™
IBM®
IMS™
Lotus Discovery Server™
Lotus Notes®
Lotus®
NetVista™
Notes®
OS/390®
QuickPlace®
Redbooks™
Sametime®
SecureWay®
Tivoli®
WebSphere®
Redbooks (logo) ™

The following terms are trademarks of other companies:

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

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

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

Other company, product, and service names may be trademarks or service marks of others.

Preface

This IBM® Redbook will help you design, develop, and manage portal applications in a WebSphere® Portal and DB2® Information Integrator environment.

Attention: The target audience of this book is primarily WebSphere Portal administrators and portlet developers responsible for developing portals that integrate content from geographically distributed diverse content repositories. Therefore, it does not cover WebSphere Portal installation, configuration of data sources in WebSphere Application Server, or best practices associated with portlet development — all of which should be familiar to the target audience.

For readers interested in these topics, please refer to *IBM WebSphere Portal V5 Handbook,* SG24-6098-00, and "Portlet Development Best Practices and Coding Guidelines" March 17th 2003 by Marshall Lamb available at:
`ftp://207.25.253.53/1/wsdd/pdf/PortletCodingGuidelines.pdf`

This book is organized as follows:

- **Chapter 1** provides a general introduction to portals, categorizes the different types and documents their evolution. It also discusses the challenges of delivering effective portals in emerging on-demand environments, and identifies the requirements of a portal framework for rapid development and deployment of portals.
- **Chapter 2** provides a general overview of the functionality and main components of WebSphere Portal.
- **Chapter 3** discusses the business needs driving the need for information integration, and IBM's response to this demand with its DB2 Information Integration family of products. It introduces the IBM DB2 Information Integration family of products and focuses on DB2 Information Integrator since this redbook discusses its synergy with WebSphere Portal.
- **Chapter 4** describes the inherent synergy between WebSphere Portal and DB2 Information Integrator (DB2 II), discusses key considerations in integrating DB2 II in different WebSphere Portal topologies, and provides tips for successful deployment.
- **Chapter 5** describes a hypothetical portal scenario involving a financial services company named Cotton-Wood Financial Services (CFS) that

leverages DB2 II functionality. It describes custom portlets that leverage DB2 II to access diverse data sources in a typical environment.

- **Appendix A** describes a typical installation of DB2 Information Integrator on AIX® and Windows® platforms.
- **Appendix B** describes DB2 II configuration of data sources used in the CFS portal.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

Nagraj Alur is a Project Leader with the IBM International Technical Support Organization, San Jose Center. He has more than 28 years of experience in DBMSs, and has been a programmer, systems analyst, project leader, consultant, and researcher. His areas of expertise include DBMSs, data warehousing, distributed systems management, and database performance, as well as client/server and Internet computing. He has written extensively on these subjects and has taught classes and presented at conferences all around the world. Before joining the ITSO in November 2001, he was on a 2-year assignment from the Software Group to the IBM Almaden Research Center, where he worked on Data Links solutions and an eSourcing prototype.

Isaac Allotey-Pappoe is a Senior IT Specialist in Sweden. He has more than 8 years of experience in information technology. He holds a degree in Computer Science from the Kwame Nkrumah University of Science and Technology in Ghana, and an MBA from Durham University in England. His areas of expertise include Internet computing, object-oriented programming, and WebSphere.

Chris Delgado is a Consultant in Atlanta, GA working with clients in the areas of WebSphere, Java™, and relational databases. With nearly 10 years experience, he has worked as a programmer, administrator, and architect specializing in server-side application development for distributed systems. Chris is an IBM Certified Specialist in several areas of WebSphere and DB2 including WebSphere Portal, WebSphere Commerce, WebSphere Application Server, and DB2 UDB. Chris holds a Bachelor of Arts degree from Loyola University, New Orleans.

Jayanthi Krishnamurthy is an IT Architect providing enterprise solutions to government organizations. She works for the Government Services Division of Radiant Systems, USA. She has more than 8 years of experience in Information Technology. She holds a degree in Electronics and Communication Engineering from the University of Madras, India. Her areas of expertise include Business

Intelligence, WebSphere Portal, WebSphere Application Server, Web Services, and J2EE.

We would especially like to thank Benjamin Wilde and Jacques Labrie for their great support and contributions to this project. We would also like to thank Paolo Bruni (SG24-7052), Rufus Credle (SG24-6098), and Junius (Buck) Stearns (SG24-7011) for allowing us to borrow heavily from their redbooks.

Walter Alvey
Carol Greenstreet
Jacques Labrie
Cindy Saracco
Laura Stewart
Benjamin Wilde
Kevin Winterfield
IBM Silicon Valley Laboratory

Rufus Credle
Junius (Buck) Stearns
International Technical Support Organization, Raleigh Center

Paolo Bruni
Maritza M. Dubec
Emma Jacobs
International Technical Support Organization, San Jose Center

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

`ibm.com/redbooks/residencies.html`

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  ibm.com/redbooks

- Send your comments in an Internet note to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. QXXE Building 80-E2
  650 Harry Road
  San Jose, California 95120-6099

1

Introduction to portals

In this chapter we introduce portals, categorize the different types, and document their evolution. We also discuss the challenges of delivering effective portals in emerging on-demand environments, and identify the requirements of a portal framework for rapid development and deployment of portals.

The topics covered are:

- What are portals
- Types of portals
- Evolution of portals
- A portal framework

1.1 What are portals

Portals have been around since the early days of the World Wide Web. Whether they knew it or not, everyone who has ever used the Internet has probably used a portal. Generally synonymous with *entrance* or *gate*, a portal is often the page we see when we use a browser to visit a company's Web site to perform tasks such as searching for information, or executing a stock trade.

The definition of a portal has evolved to now mean an integrated and personalized Web-based interface that provides the user with a single point of access to a wide variety of data, knowledge, and services — at any time and from anywhere using any Web-enabled client device.

Such a tool considerably enhances user productivity, thereby resulting in significant cost savings and improved customer satisfaction.

A portal is therefore a single interface that provides convenient access to everything a user needs to get his/her tasks done, regardless of where the information needed to get them done exists. Whether to search for and buy a book, access an account balance and make a transfer, or update personal information in the human resources (HR) system at work, the portal brings everything together in one virtual place.

Similar to a workstation desktop, a portal displays a variety of information and services in a single, consistent, user-friendly interface. Sometimes referred to as a *Web top*, a portal can be the major starting point or anchor site that users visit when they connect to the Web. However, unlike a traditional desktop, a Web-based portal is accessible through a wide range of Web-enabled client devices.

Figure 1-1 is an example of a portal on a typical Web browser.

Figure 1-1 Example of a portal on a typical Web browser

The fundamental characteristics of a portal include information aggregation from local and remote heterogeneous data sources, targeted and personalized information, accessibility from anywhere at any time, and single sign-on capabilities.

What differentiates a portal from a Web site includes the following capabilities:

- A single point of access to all resources associated with the portal domain
- A personalized user experience
- Federated access to hundreds of data types and repositories, both aggregated and categorized
- Collaboration and user integration capabilities
- Integration with applications and workflow systems

These capabilities are critical to improving user productivity and communications, and enhancing the user experience.

Figure 1-2 summarizes the total context of portals in terms of information, services, and infrastructure.

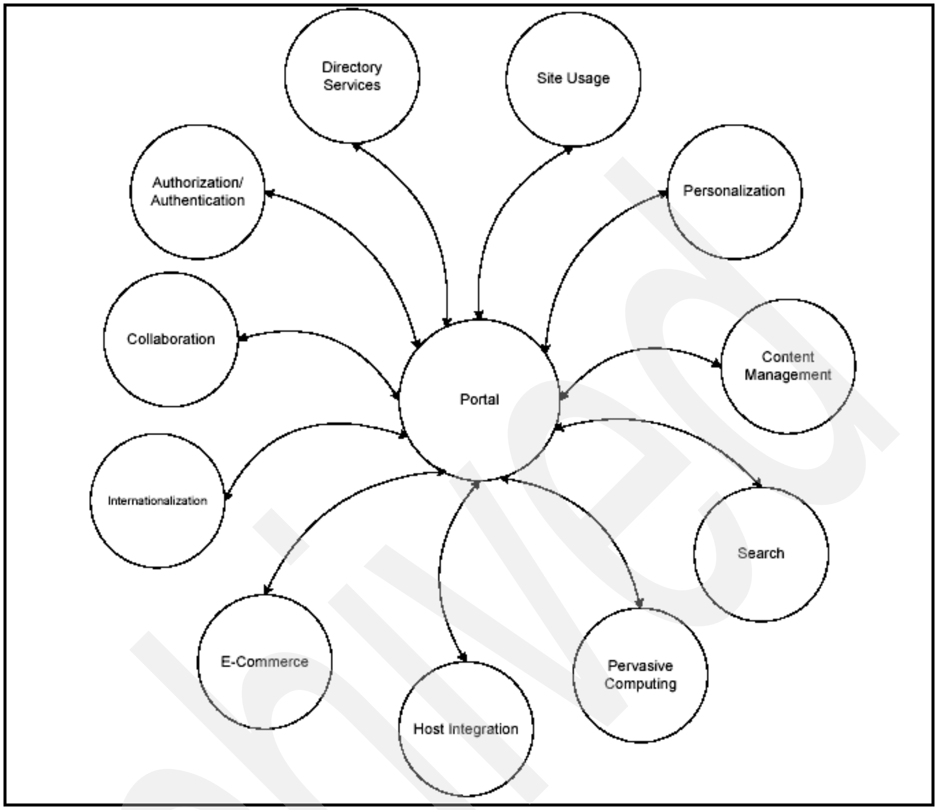


Figure 1-2 Portal context diagram

1.2 Type of portals

Portals tend to be categorized based on the role of users accessing the portal, and the kinds of information and services they use. Portals are categorized as follows:

- **Business-to-Consumer (B2C) portals**

  This type of portal (also known as an extended enterprise portal or *extranet portal*) is associated with CRM [1] and provides consumers with direct access to a variety of content; for example, product manuals, product availability, and price lists.

[1] Customer Relationship Management

Portal customers might also purchase products, check order status, and communicate with customer support. Like any other portal, a B2C portal is usually tailored to match customer's needs.

1. **Business-to-Business (B2B) portals**

   B2B portals are also an extended enterprise portal, and participate in supply chain management (SCM) by providing personalized access to business information by suppliers, resellers, and distributors.

   A typical B2B portal might provide a business partner with access to purchase orders, invoices, statements, and confirmations. Application integration is also required to integrate business processes in procurement, billing, manufacturing, and distribution areas.

2. **Business-to-Employee (B2E) portals**

   B2E portals (also known as *intranet portals*) generally serve as a means to aggregate and disseminate corporate information and services to an organization's employees. There are two basic types of B2E portals:

   – **Employee portals** provide access to relevant content such as company news, HR information, search engines, sources of expertise, reports, and other types of information generally applicable to all employees.

     These portals can enable employees to communicate and collaborate through chat rooms, discussion groups, etc. Typically, an employee portal also allows for self-service, where an employee can sign up for classes or HR benefits, change personal information, etc.

   – **Knowledge worker portals** that are aimed a particular role or set of roles such as sales.

     These portals often integrate content in order to support a particular process or processes. For example, an automotive technician might require resources from a number of applications such as service history, calendar scheduling, or parts availability.

3. **Public or mega portals**

   Sometimes called *Internet portals*, these portals focus on addressing large audiences. There are two major types of public portals as follows:

   – **General public portals** address the entire Internet as opposed to a specific community; for example Yahoo, Google, and Excite.

   – **Industrial or vertical portals** are focused on specific narrow audiences such as retailers, manufacturers, or finance; for example, `http://www.hbs.edu/projfinportal/` which is intended to be a reference guide for students, researchers, and practitioners seeking to obtain information about project finance.

Figure 1-3 summarizes the categories of portals and the types of users who use them. Public portals are not included here as they apply to the entire spectrum of users.

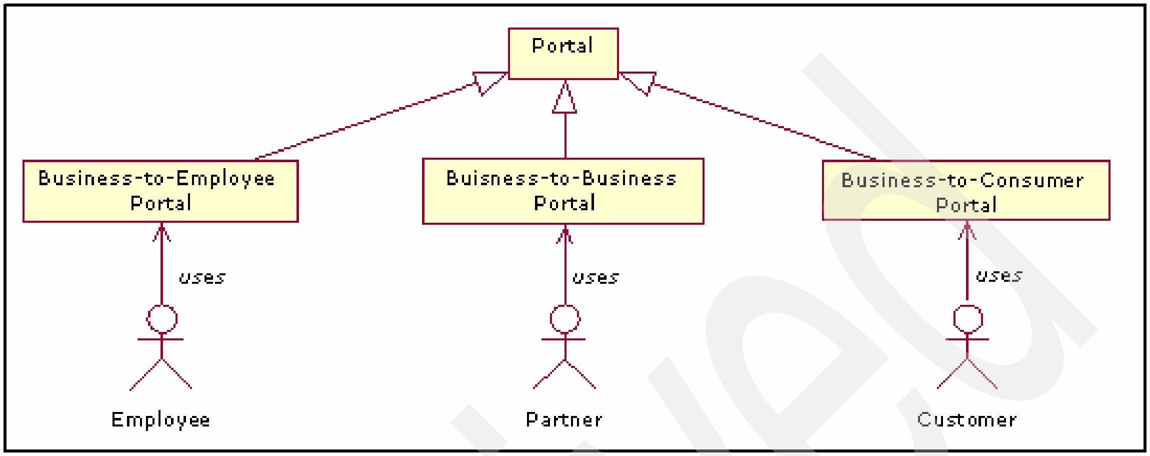


Figure 1-3 Categories of portals and who uses them

> **Important:** A specific portal solution tends to be a hybrid implementation of the different types of portals discussed. Since there can be many types of users within an organization, there can be many hybrid portal solutions supporting these users.

1.3 Evolution of portals

Portals have evolved through three generations as follows:

- **First generation portals**

  First generation portals were focused on providing static Web content, Web documents, and live feeds. Examples of first generation portals are Yahoo, Lycos, and Excite. They are mostly an aggregation of content.

  In corporate environments, they provided a single interface to corporate information distributed throughout the enterprise, and include information such as company news, employee contact information, company policy documents, and other key Web links.

4. **Second generation portals**

   Second generation portals expanded first generation portals by adding a focus on specific information, applications, and collaboration as follows:

   – Integration at the data level such as data warehouses

– Providing services such as customer service for call centers, and handling of expense claims, pension plans, stock purchase plans, and health care benefits

– Collaboration capabilities for teams to work in a virtual office. Collaboration portals provide content management services (the mining and organization of related information) along with collaborative services that allow users to chat, e-mail, share calendars, and define user communities. Collaboration is typically at the intranet level.

5. **Third generation portals**

   The emerging third generation portals are intended to address full-function e-business as shown in Figure 1-4. This involves real-time data access and end-to-end process integration not only within an organization, but also with partners, suppliers, and customers.

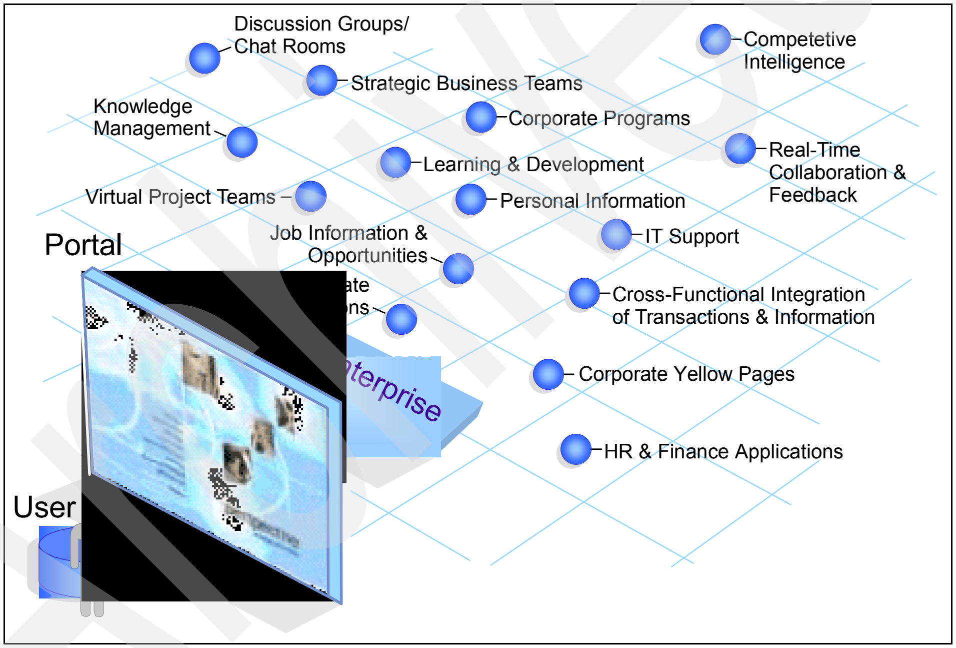


Figure 1-4 e-Business needs

The significant characteristics of the third generation of portals are as follows:

– Integration of application servers provide a single point of integration for content and applications as well as collaborative services.

– Access from multiple types of devices to address the diverse user communities in need of services. They offer the richest set of content and application choice through a single user interface to a diverse community, including browsers and pervasive devices.

– Support automated personalization through rules engines.

> **Attention:** Critical to the further evolution of portals is the availability of a portal framework for common services.

1.4 Portal framework

All portals share common characteristics such as the following:

- Aggregating geographically distributed structured[2] and unstructured[3] information, and services into one place. Information accessed can include syndicated content[4] supplied specifically for the purpose of reuse and integration with other material.

  Services provided include the following:

  – **Collaborative services** (also known as *communication services*) allow people to chat, locate expertise, share calendars, participate in discussion groups, use white boards, etc.

  – **Content management services** provide search, tracking, and data mining capabilities.

  – **Self-service** (also known as *transactional services*) enable users to “interact” with systems directly without going through an intermediary such as a customer representative or salesperson. This allows users to buy products, schedule meetings, check account balances, enroll for classes, etc.

6. Supporting personalization for a group or an individual.

   One of the more interesting aspects of a portal is its ability to be different things to different people. One of the ways it does this is through personalization and customization.

   – **Personalizing** a portal involves choosing the content that should be displayed to a particular individual. Often this is chosen automatically based on business rules such as the user’s role in an organization. For example, when salespeople sign into the system, they are automatically presented with a list of new products.

[2] Data that is been organized (often hierarchically by keywords) in such a way as to facilitate easy searching. A library card catalog is a good example. Structured data often includes reports, analyses, canned queries, and other types of “business intelligence.”

[3] Data that is often hard to search, and is unorganized data residing outside a database. Unstructured data can be text, audio, video, or graphics and take the form of office documents, memos, e-mails, meeting minutes, or any other such “knowledge.”

[4] Data that is commercially available material such as news reports, stock quotes, cartoons, and trivia.

– **Customizing** a portal involves choosing how the portal looks (also known as *theme* and *skin*), what the navigation model will be, and where on the portal screen the content should be displayed (layout). A portal can even be “branded” to appear different to different types of users.

Personalization and customization allow a portal to target a specific community of users such as customers, partners, or employees. Some portals may even be “individualized” to the preferences of a particular user.

7. Being accessible at anytime from anywhere using a standard Web browser

   Since a portal is Internet-based, it is accessible at anytime from anywhere using a standard Web browser. The advent of numerous Web-enabled devices such as cell phones and PDAs has enabled the portal to be extremely versatile and useful in a variety of settings.

All portals require a scalable infrastructure, a flexible and powerful presentation framework, and a framework for building portal components easily. Each requires a high degree of personalization so that the most relevant information is delivered to the user, enabling a more productive interactive experience, and encouraging user loyalty to the portal.

Depending on the nature and sensitivity of the information involved, some portals may require a greater degree of security, including specialized forms of authentication and access control. Depending on the size of the user base, some portals might require very high availability and scalability. Consumer (B2C) portals generally allow users to enroll themselves and manage their own accounts. Conversely, enterprise portals (B2B and B2E) often require integration with existing user databases or enrollment systems.

To enable the rapid development and deployment of portals, vendors such as IBM have developed a portal framework based on the concept of portlets. These are Java-based reusable user interface components that process requests and generate dynamic content.

A portal framework simplifies the development and maintenance of portal sites by supporting the following capabilities:

- The portal page structure is defined only once.
- Portlets are defined independently.
- Portlets can be changed without impacting the overall portal page design.
- Targeting of multiple browsers and mobile devices is easier.

The following subsections briefly describe the key elements of a portal framework that includes portlets, the portlet container, portal services, and the portal server.

1.4.1 Portlets

A portlet is the key building block in most portal frameworks. As mentioned earlier, portlets are Java-based reusable user interface components that process requests and generate dynamic content. Executing in a runtime environment called a *portlet container*, portlets present their content in a window-like display on a portal page as shown in Figure 1-5. Similar to a window on a desktop, the portlet window has a title bar that contains controls that allow the user to expand (maximize) and shrink (minimize) the application.

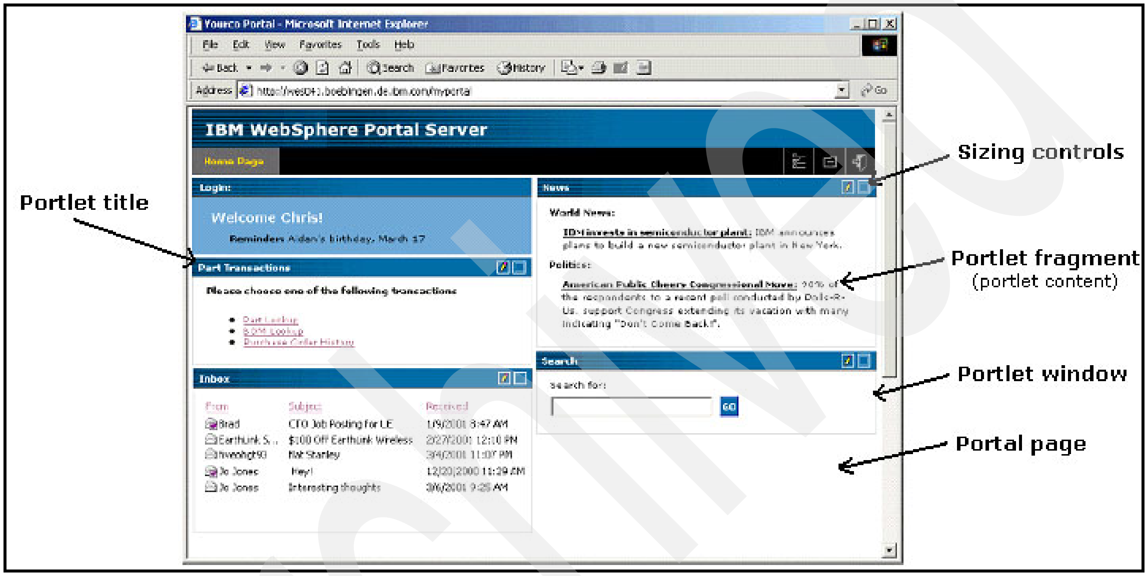


Figure 1-5 Elements of a portal page

Web clients interact with a portlet using the standard request/response paradigm. For a given request cycle, each portlet generates specific content called a *fragment*. Each fragment represents a small portion of markup (for example, HTML or XHTML) that is aggregated with other fragments to form the complete response document.

> **Important:** A portlet is visible on a portal page as a single small window. Each portal page can have many portlets. The portlet is the content *inside* the window, not the window itself.

1.4.2 Portlet container

Most portal frameworks provide the runtime execution environment for the portlets called a *portlet container* (see Figure 1-6).

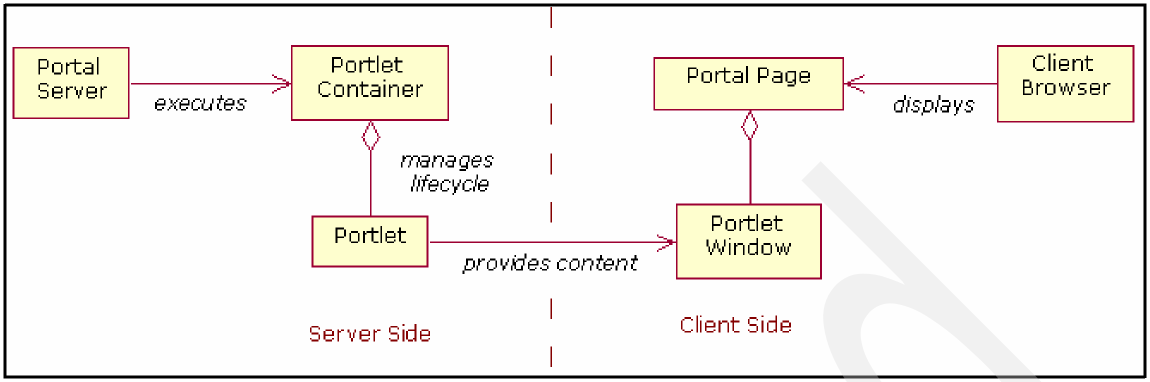


Figure 1-6 Depicts the context in which a portlet exists

The portlet container is responsible for instantiating, invoking, and destroying the portlets it hosts in response to requests it receives from the portal server (discussed in 1.4.4, "Portal server" on page 12). Content aggregation is not a function associated with the portlet container, but rather with the portal or portal server.

1.4.3 Portal services

Portlets rely on the portlet container to provide the necessary infrastructure to support a portal environment. The portal infrastructure provides the core sets of services required by the portlets including the following:

- **Personalization services** enable the portlet to make use of rules engines and user profile information to modify content in order to make a user's visit to the portal more productive and satisfying.
- **Event notification services** enable portlets to respond to various requests in a fashion that is de-coupled from the portal environment.
- **Communication services** provide portlet-to-portlet communication.
- **Content management services** facilitates connections to virtually any content or application source.
- **Search services** support heterogeneous searches across many data sources.
- **Collaboration services** enable users to communicate and participate in "communities of interest."
- **User and group management services** allow users to enroll at the portal and to self-manage their own preferences and account information.

- **Page transformation services** provide support for a wide variety of client devices.
- **Other services** provide or manage:
  – User profile and other types of persistent data
  – Security and access control services, including user authentication and authorization
  – Performance monitoring, load balancing, and content caching

1.4.4 Portal server

The portal server is a specialized application server that provides business logic for a portal application. Typically built on top of an J2EE-compliant application server, the portal server provides development and runtime infrastructure for the portal. A portal server often works in conjunction with a Web server to process a client request as shown in Figure 1-7.

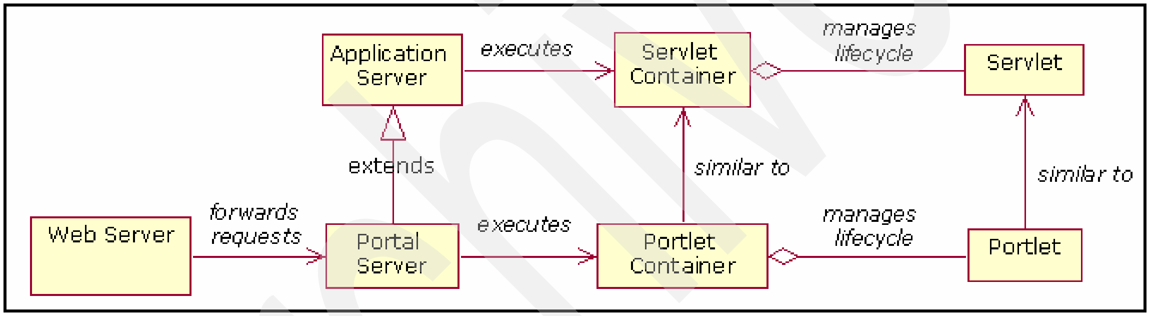


Figure 1-7 Portal server extends Application Server to support portal applications

Whereas a servlet can be viewed as a means of extending the functionality of the application server, a portlet can be seen as a way to extend the functionality of the portal server.

In summary, portal servers provide a server to aggregate content, a scalable infrastructure, and a framework to build portal components and extensions.

Figure 1-8 depicts the flow of a typical portal request processing scenario.

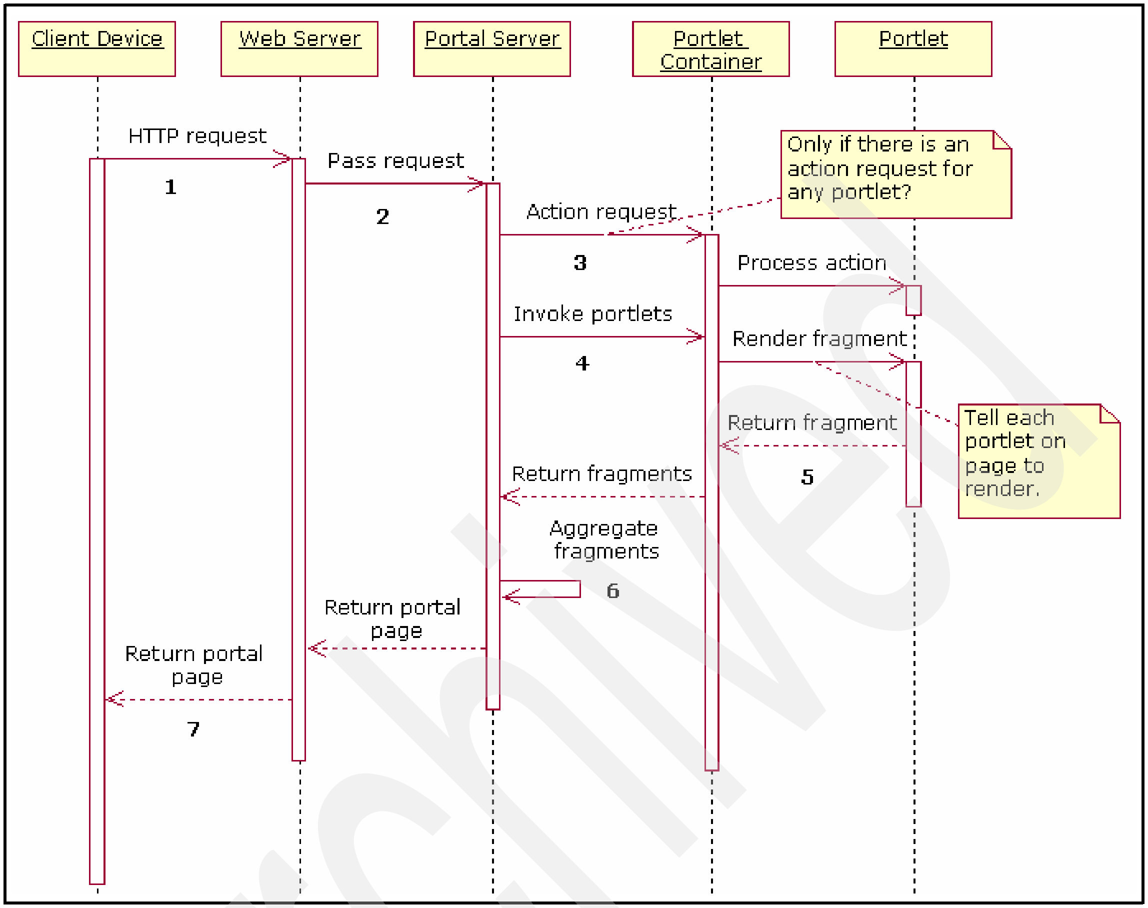


Figure 1-8 Typical portal page request processing scenario

This scenario is initiated when the user requests the portal page from the client device:

1. A client device (for example, a browser, or PDA) makes an HTTP request for the portal page to the Web server.
2. The Web server recognizes the request for a portal page and forwards the request to the portal server.
3. The portal server determines if the request contains an action targeted to a portlet on the portal page. If so, the portal requests the portlet container to invoke the portlet to process the action.
4. The portal server requests the portlet container to invoke the portlets associated with the portal page.

5. The portlet container requests each portlet associated with the portal page to render a fragment of dynamic content, and each does so after accessing the various data sources necessary.

> **Important:** If the data sources are disparate and geographically distributed, the portlet developer needs to use a variety of native APIs to access these data sources if a federated capability such as DB2 Information Integrator is not available.
>
> This can be a significant challenge depending upon the diversity of data sources involved.

6. The fragments are returned to the portal server, where they are aggregated to form the portal page.
7. The portal page is returned to the client device for display.

The portal infrastructure provides the core sets of services required by the portlets including:

- Access to user profile information
- A framework for portlets to participate in events
- A framework to communicate with other portlets
- Access to remote content
- Access to credentials
- A framework for storing persistent data

2

WebSphere Portal architecture overview

In this chapter we provide a general overview of the functionality and main components of WebSphere Portal.

It is aimed at a DB2 II audience.

The topics covered are:

- WebSphere Portal overview
- WebSphere Portal architecture
- WebSphere Portal Family
- WebSphere Portal components

For more information about WebSphere Portal refer to:

- WebSphere Portal Infocenter for IBM WebSphere Portal for Multiplatforms:
  `http://publib.boulder.ibm.com/pvc/wp/500/ent/en/InfoCenter/index.html`
- The IBM Redbook *IBM WebSphere Portal V5 Handbook*, SG24-6098-00 at:
  `http://publib-b.boulder.ibm.com/Redbooks.nsf/9445fa5b416f6e32852569ae006bb65f/b865c47e5d4e074e85256da20057b66a?OpenDocument`

2.1 WebSphere Portal overview

IBM WebSphere® Portal allows people to interact with the on demand world in a personalized way by automatically accessing dynamic information of interest. Users can quickly execute business processes across critical applications, as well as collaborate with portal users inside and outside your organization. These capabilities help an organization improve employee productivity, cut costs, and strengthen relationships with customers and trading partners.

WebSphere Portal consists of middleware, applications (called portlets), and development tools for building and managing secure B2B, B2C and B2E portals. WebSphere Portal's extensible framework allows the end user to interact with enterprise applications, people, content, and processes. WebSphere Portal can deliver Web content to WAP-enabled devices and i-Mode phones, as well as to various Web browsers.

Attention: WebSphere Portal functionality is a superset of the portal framework requirements defined in 1.4, "Portal framework" on page 8.

Figure 2-1 shows the default homepage of IBM WebSphere Portal V5.

Figure 2-1 WebSphere Portal Default Home Page

WebSphere Portal enables the following:

- Helps build scalable and reliable B2B, B2C and B2E portals
- Delivers a single point of personalized interaction with applications, content, processes, and people for a unified user experience
- Allows users to view, search, create, convert, and edit basic documents, spreadsheets, and presentations from within the portal
- Provides powerful collaboration capabilities such as instant messaging, team workplaces, people finder, and e-meetings
- Enables quick portal integration with backend systems through portlet builders.

Figure 2-2 shows the Administration page of IBM WebSphere Portal V5.

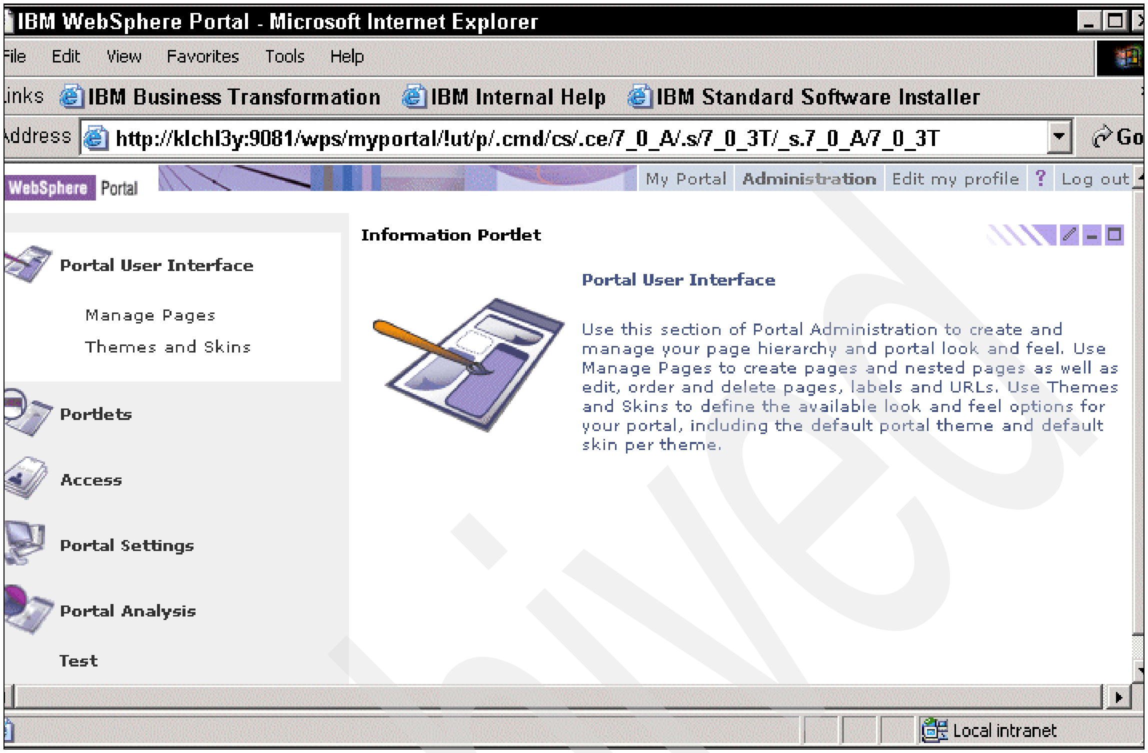


Figure 2-2 WebSphere Portal Administration Page

Customization is a key feature of WebSphere Portal.

As a WebSphere Portal administrator, you can customize the WebSphere Portal to meet the specific needs of your organization, users, and user groups as follows:

- You can adapt the look and feel of the portal to fit the standards of your organization, and customize page content for users and groups in accordance with business rules and user profiles.
- End users such as business partners, customers, and employees can further customize their own views of the portal. End users can add portlets to pages and arrange them as they want, and control portlet color schemes. By aggregating portlets in one place and giving end users the power to customize their own desktops, WebSphere Portal gives end users a means for doing business efficiently and with high satisfaction.

The WebSphere Portal framework lets you plug in new features or extensions called *portlets* as follows:

- WebSphere Portal ships a rich set of standard portlets, including portlets for displaying syndicated content, transforming XML, and accessing search engines and Web pages. Portlets for accessing Lotus® Notes®, Microsoft® Exchange, and instant messaging are also included.
- Several third-party portlets are also available. Examples include Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) portlets.
- WebSphere Portal also ships an API and a toolkit that portlet developers can use to create custom portlets.

2.2 WebSphere Portal architecture

WebSphere Portal's architecture supports portal application development and execution, and provides services for Authentication and Authorization though WebSphere Member Services.

The WebSphere Portal core architecture comprises Presentation Services, the Portal Infrastructure, and the Portal Services. WebSphere Portal takes advantage of the strong platform infrastructure provided by WebSphere Applications Server.

Figure 2-3 describes the overall architecture of WebSphere Portal.

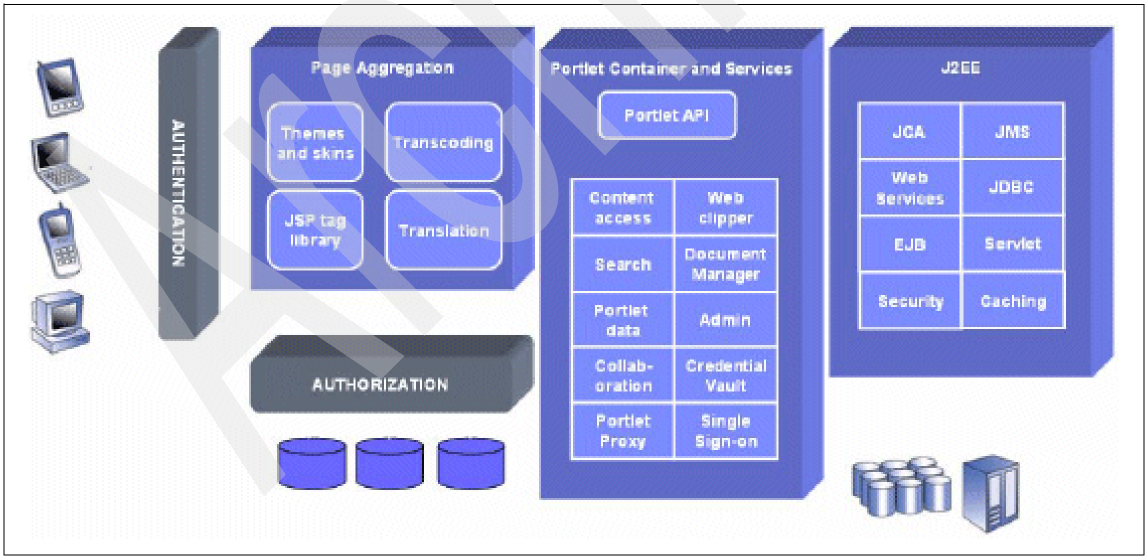


Figure 2-3 WebSphere Portal architecture

WebSphere Portal provides a pure Java engine called the WebSphere Portal engine whose main responsibility is to aggregate content from different sources and serve the aggregated content to multiple devices. The WebSphere Portal engine also provides a framework that allows the presentation layer of the portal to be decoupled from the portlet implementation details. This allows the portlets to be maintained as discrete components.

Figure 2-4 describes the components of the WebSphere Portal engine.

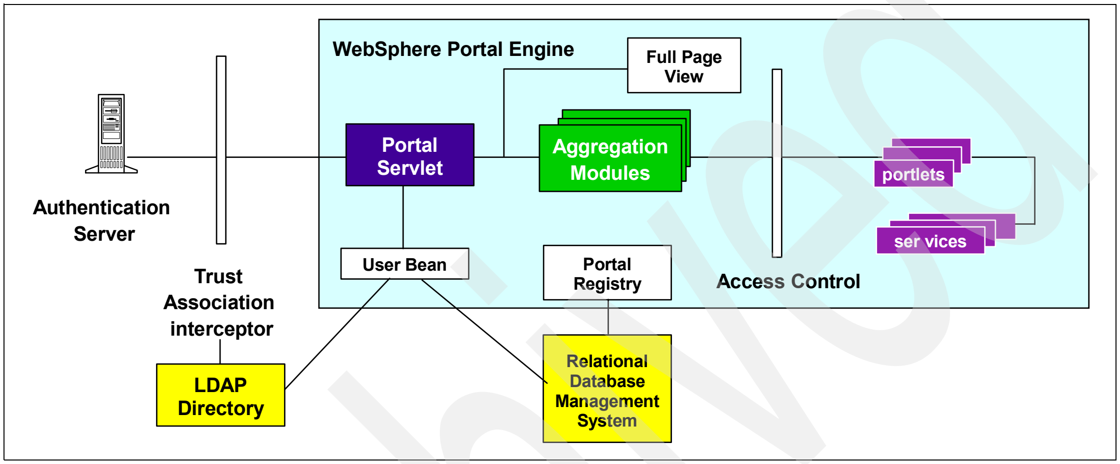


Figure 2-4 WebSphere Portal engine

- The *Authentication Server* is a third party authentication proxy server that sits in front of the WebSphere Portal engine. Access to portlets is controlled by checking access rights during page aggregation, page customization, and other access points.
- From a user's perspective, a *portlet* is a window in the portal that provides a specific service or information; for example, a calendar or news feed. From an application development perspective, portlets are pluggable modules that are designed to run inside a portlet container of a portal server.

  The portlet container provides a runtime environment in which portlets are instantiated, used, and finally destroyed. The portlet container is not a stand-alone container like the servlet container. Instead, it is implemented as a thin layer on top of the servlet container and reuses the functionality provided by the servlet.

  Portlets rely on the portal infrastructure to access user profile information, participate in window and action events, communicate with other portlets, access remote content, look up credentials, and store persistent data. The portlet API provides standard interfaces for these functions. WebSphere Portal V5 supports the following two portlets' APIs in the portlet environment:

- IBM portlet API
- JSR 168 API

8. The *Portal Servlet* is the main component of the WebSphere Portal engine. The portal servlet handles the requests made to the portal. The portal requests are handled in two phases as follows:

    a. The first phase allows portals to send event messages between themselves.

    b. In the second phase, the appropriate aggregation module for the requesting device renders the overall portal page by collecting information from all the portlets on the page, and adding standard decorations such as title bars and edit buttons.

9. **Portal Services** are components that WebSphere Portal uses to extend portal functionality. Key WebSphere Portal functionality includes personalization, search, content management, site analysis, enterprise application integration collaboration, and Web services. Portlets can access these services through their portlet container.

Figure 2-5 shows the WebSphere Portal infrastructure of several modular subsystems.

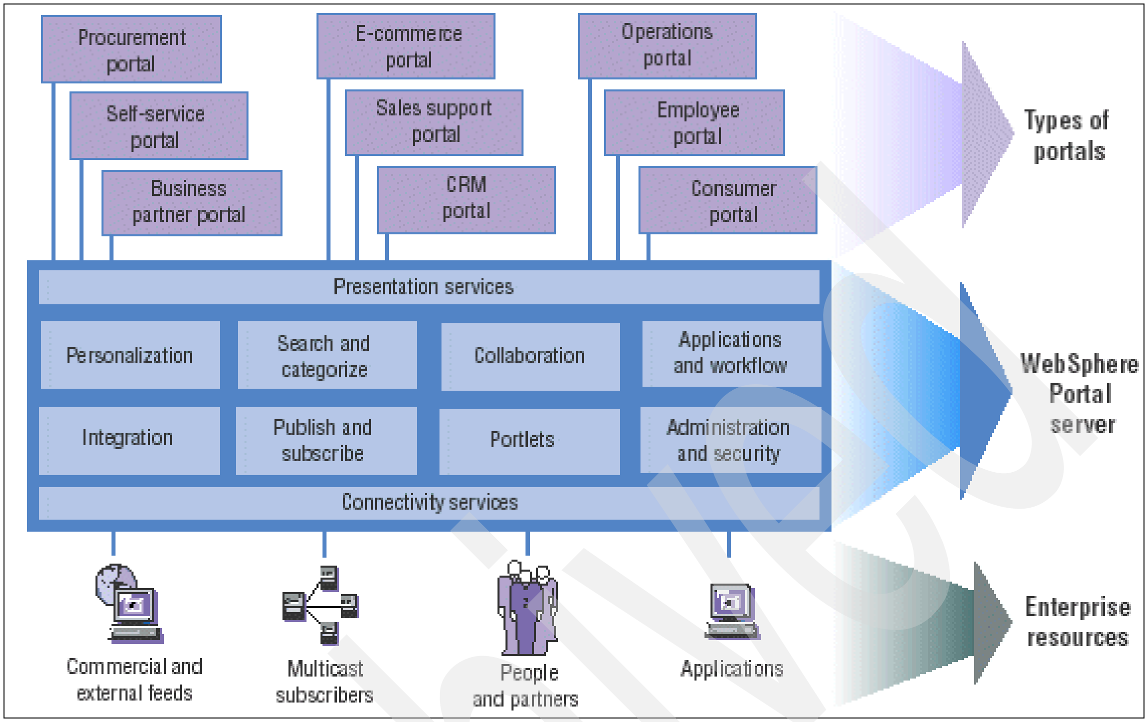


Figure 2-5 WebSphere Portal infrastructure

A brief overview of each of the WebSphere Portal Server components follows:

- **Presentation services** supports a Web user interface plus pervasive device support.
- **Personalization** is the ability to serve dynamic response to the user based on personal profiles.
- **Search and categorize** involves categorizing repositories of content and searching them for relevant information.
- **Collaboration** includes tools that allow e-mail, team rooms, shared places, and so forth to be exchanged.
- **Applications and workflow** involves integration of legacy and new applications.
- **Integration** involves metadata sharing, XML, connectors, standards, and EAI.
- **Publish and subscribe** involves the ability to author new content and publish it to subscribers.

- **Portlets** includes a framework for easily attaching software modules (portlets) and services.
- **Administration and security** supports basic Web site services such as page designers, performance monitors, cluster services, and metadata management.

2.3 WebSphere Portal Family

Figure 2-6 shows the various packaging versions of the WebSphere Portal Family.

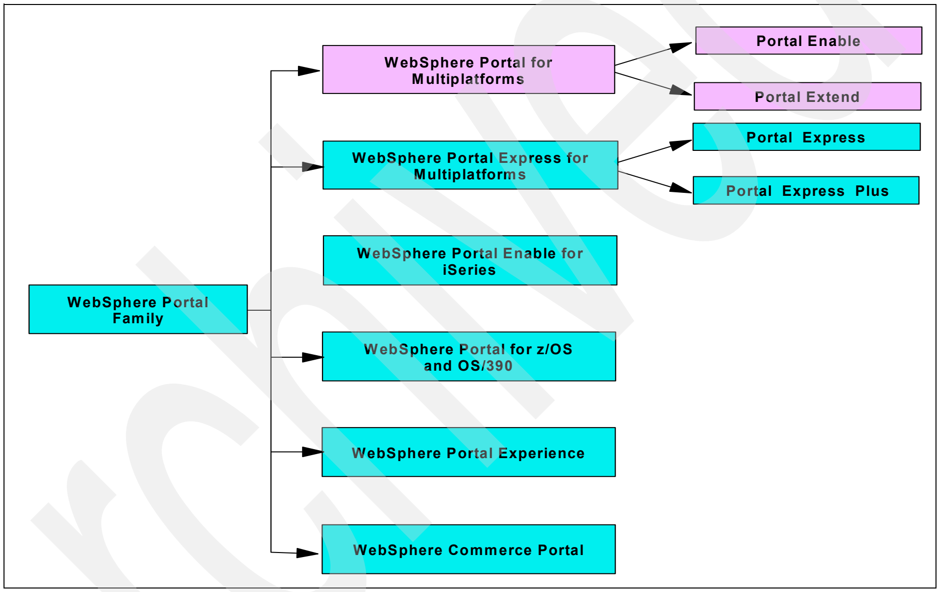


Figure 2-6 WebSphere Portal Family

In this section we list the various components comprising WebSphere Portal for Multiplatforms, which ships in two flavors: IBM WebSphere Portal Enable for Multiplatforms and IBM WebSphere Portal Extend for Multiplatforms.

Figure 2-7 summarizes the various components that are bundled with WebSphere Portal Enable and Extend editions.

Figure 2-7 WebSphere Enable and Extend components

> **Note:** Consult the product license for specific terms of entitlement for the software components that are shipped with your edition of WebSphere Portal. Usage restrictions apply for some software.

2.3.1 IBM WebSphere Portal Enable for Multiplatforms

The IBM WebSphere® Portal Enable for Multiplatforms offering is the basic edition of WebSphere Portal for Multiplatforms, and provides the following capabilities:

- Portal Document Manager provides a way for portal users to share, view, and organize files of all types ranging from documents to spreadsheets within the portal community. This communication enhancer offers category subscription services, simple approval processes for file contribution, automatic dialog boxes for contributing portal search, versioning so that users can track the evolution of a piece of content, and access control for managing viewing privileges of different content items.

- Productivity components allow users to view, create, convert, and edit basic documents, spreadsheets, and presentation files from the portal interface. Therefore, they can execute ad-hoc business processes from the same place they access their applications, search for information, and collaborate with other employees and partners. The productivity components are integrated with the document management feature so that files can be indexed, categorized, and searched by other portal users.
- Portal Application Integrator allows business users to quickly create portlets for interacting with relational databases, Domino® databases, and enterprise applications from Oracle, SAP, Siebel, and PeopleSoft.
- Click-to-action (C2A) technology for portlet-to-portlet communication and action, ensuring the accuracy of information passed and delivering it on demand.
- Integration services that give you access to enterprise data, applications, newsfeeds and Web services.
- Publish local portlets as remote Web services, or subscribe to Web services to make them available to portal users through portlets.
- Presentation services that allow for the customization of the computing desktop to match individual work patterns and roles.
- Browser-based content publishing and personalization technology so that portal users receive a unique experience with the latest information.
- WebSphere Translation Server functionality helps you translate the contents of portlets from English to French, Italian, German, Spanish, Portuguese, Taiwanese, Japanese, simplified Chinese, and traditional Chinese. Or you can translate your portlet content from those languages to English.

2.3.2 IBM WebSphere Portal Extend for Multiplatforms

WebSphere Portal Extend for Multiplatforms, Version 5 includes all the robust features of WebSphere Portal Enable, as well as the following additional capabilities.

- Collaboration Center: A set of ready-to-use collaborative portlets that can be used right out of the box, thereby providing instant value for your portal users. The Collaboration Center integrates portlets for finding, connecting, and working with people inside and outside your organization. It is fully integrated within WebSphere Portal.

- Provides extended search capabilities across relational databases such as DB2® Universal DataBase and Oracle®, Lotus® Notes and Lotus Domino databases, Web search engines, and text or HTML documents.
- Robust Web analysis technology such as Tivoli® Web Site Analyzer provides vital business intelligence about customers using your portal.

3

DB2 Information Integration architecture overview

In this chapter we briefly describe the business needs driving the need for information integration, and IBM's response to this demand with its DB2 Information Integration family of products. We introduce the IBM DB2 Information Integration family of products, and focus on DB2 Information Integrator since this redbook discusses its synergy with WebSphere Portal.

The topics covered are:

- Current business trends
- IBM's DB2 Information Integration overview
- DB2 Information Integrator V8.1

3.1 Introduction

A number of business trends are driving the need for integration of data and processes across employees, customers, business partners, and suppliers. The inherent heterogeneity of hardware and software platforms in intranets and extranets presents unique challenges that must be overcome in order to gain a competitive advantage in the global economy.

In this chapter we discuss the current business trends fueling integration demands, IBM's DB2 Information Integration solution, and IBM's federated DB2 Information Integrator V8.1 offering.

3.2 Current business trends

To keep up with the evolution of e-business computing, companies in every industry are being challenged to act and react on demand. Responding to any customer demand, each market opportunity and every external threat requires integration between people, processes, and information. This integration must extend across the company, and across partners, suppliers, and customers.

Integration, automation, and virtualization are the three key elements of this on-demand operating environment:

- **Integration** is the efficient and flexible combination of data to optimize operations across and beyond the enterprise. It is about people, processes, and information.
- **Automation** is the capability to increasingly automate business processes with the ultimate goal of self-regulation, thereby reducing the complexity of data management to enable better use of assets.
- **Virtualization** provides a single, consolidated view of and easy access to all available resources in a network, no matter where the data resides, or the type of data source.

> IBM defines an on demand business as an enterprise whose business processes integrate end-to-end across the company with key partners, suppliers, and customers in order to respond with speed to any customer demand, market opportunity, or external threat.

IBM has identified five types of integration that are based on an open services infrastructure. You can use these types of integration together or separately to solve business issues. The following five types of integration represent the various integration challenges that face businesses today, with information integration being at the core of these integration types:

- **User interaction**

  A user can work with a single tailored user interface, which is available through virtually any device, with full transactional support. The results of the user's interaction are integrated into multiple business systems.

- **Process integration**

  A business can change how it operates through modeling, automation, and monitoring of processes across people and heterogeneous systems, both inside and outside the enterprise.

- **Application connectivity**

  Applications can connect to one another so that they share and use information for better use at the enterprise level.

- **Build to integrate**

  Users can build and deploy integration-ready applications by using Web services and existing assets. You can integrate new solutions with existing business assets.

- **Information integration**

  Diverse forms of business information can be integrated across the enterprise. Integration enables coherent search, access, replication, transformation, and analysis over a unified view of information assets to meet business needs.

In the following subsections, we describe how the success of an on demand business enterprise is significantly dependent upon a seamless and scalable information integration infrastructure:

- From on demand to grid computing
- From grid computing to data federation
- From data federation to information integration

3.2.1 From on demand to grid computing

Grid computing is distributed computing taken to the next evolutionary level. The grid provides an infrastructure on which to support a large collection of communication resources such as hardware and software.

The standardization of communications between heterogeneous systems created the Internet explosion. The emerging standardization for sharing resources, along with the availability of higher bandwidth, is driving a potentially equally large evolutionary step in grid computing.

One major function of the grid is to better balance resource utilization.

An organization may have occasional unexpected peaks of activity that demand more resources. If the applications are grid enabled, the application workload can be moved to under utilized machines during such peaks. In general, a grid can provide a consistent way to balance workloads on a wider federation of resources.

3.2.2 From grid computing to data federation

An increasing number of grid applications manage very large volumes of geographically distributed data. The complexity of data management on a grid is due to the scale, dynamism, autonomy, and distribution of data sources.

One way of accessing diverse business information from a variety of sources and platform is through data federation.

Data federation is the ability to transparently access diverse business data from a variety of sources and platforms as though it were a single resource. A federated server may access data directly such as accessing a relational database, or access an application that creates and returns data dynamically such as a Web service. Figure 3-1 shows the federated approach to information integration as providing the ability to synchronize distributed data without requiring that it be moved to a central repository.

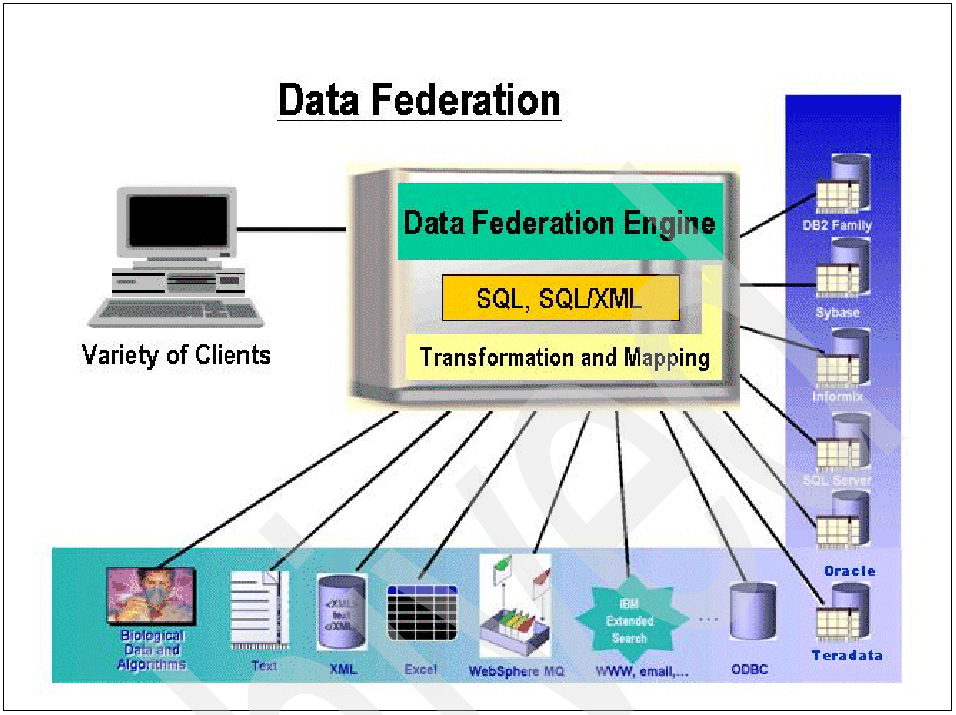


Figure 3-1 Data federation concept

Based on ongoing research investments and proven data management technologies in areas such as relational data, XML, content management, federation, search, and replication, IBM has developed the integrated infrastructure shown in Figure 3-1. Data federation uses SQL as the single language to access all data sources. This enables all data sources to be accessed in a standardized format, whether they are an application program, tool, or program product.

3.2.3 From data federation to information integration

Information integration builds on the solid foundation of existing data management solutions. Information integration provides an end-to-end solution for transparently managing both the volume and diversity of data that exists in enterprises and organizations today.

Increasingly, business IT operations involve the need to integrate diverse and unconnected infrastructures.

The following goals are critical to increasing operations efficiency and gaining a competitive advantage:

- Integrate seamlessly with new businesses and link packaged applications with legacy systems
- Control the accelerating costs of managing disparate systems and integrating across heterogeneous pockets of automation
- Mitigate shortages of people and skills while quickly reaching new markets
- Implement solutions that efficiently access and manage information crosses product and industry boundaries

3.3 IBM’s DB2 Information Integration overview

Today, any but the simplest of business tasks requires the use of information from the variety of data sources that businesses have built over many years. These sources may be local or remote, on the intranet, extranet, or Internet. The data may be stored in any of a variety of formats such as relational or non-relational databases, flat files, and unstructured content stores. The data may be current or point-in-time copies. Often, the users need both read and write access to these sources.

This complex and dynamic environment presents significant challenges to business users and applications, as well as to the IT people who must maintain and manage it.

> **Important:** IBM’s vision of information integration is to significantly reduce or even eliminate these issues.

The underlying principle of information integration is for users to be able to see all of the data they use as if it resided at a single source. Information integration technology shields the requester from all the complexities associated with accessing data in diverse locations including connectivity, semantics, formats, and access methods. Using a standards-based language such as structured query language (SQL), extensible markup language (XML) through SQL/XML, or a standard Web services or content API, information integration middleware enables users, or applications acting on their behalf, to access information transparently without concern for its physical implementation.

The goal of providing an integrated view of information can be achieved in two ways as follows:

- **Data consolidation or placement,** which involves moving the data to a more efficient or accessible location.

  Consolidating data into a single physical store has been the best way to achieve fast, highly available, and integrated access to related information.

Creating a single physical copy lets businesses meet access performance or availability requirements, deliver snapshots that are point-in-time consistent, and provide sophisticated transformation for semantic consistency. Consolidated data stores, which are typically managed to extract, transform, load (ETL) or replication processes, are the standard choice for information integration today.

However, these consolidated stores have some drawbacks as follows:

– They are expensive, racking up significant additional administration, server, and storage costs.

– The latency between the copy and the source of record can be a problem when you need the most current data.

– Rich content such as documents, images or audio is typically not included.

10. **Distributed access,** which involves providing distributed access to data through data access or federation.

Distributed access corresponds to the emerging category of technology called enterprise information integration (EII), which addresses some of the shortfalls of data consolidation or placement. EII represents middleware technology that lets applications access diverse and distributed data as if it were a single source, regardless of location, format, or access language. Access performance will typically be slower than for consolidated stores because the query may have to gather information from distributed locations across the network rather than access a single, local copy of data.

However, the benefits of EII include:

– Reduced implementation and maintenance costs because you do not have the additional hardware (server and storage) and skills and personnel costs.

– Access to current data from the source of record.

– Combining traditional data with mixed format data.

– Access to copy-prohibited data based on data security, licensing restrictions, or industry regulations that restrict data movement; for example, some European countries prohibit commingling a customer's personal data with account data in a single database. But you can materialize a single image of the data by federating them at the time of access.

> **Note:** Distributed sources must be sufficiently consistent to make joining the data both possible and meaningful. There must be a key on which the data can be joined or correlated such as a customer identifier, and the joined data must represent related topics.

Both data consolidation or placement, and distributed access data consolidation serve distinct problem domains and are very complementary — they may be used alone or together to form the heart of what is required to integrate information.

Both approaches require extensive and largely common supporting functionality. Neither distributed access nor data consolidated or placement can exist without mapping and transformation functionality, which ensures data integrity. Furthermore, depending on the business requirement, the same data may need to be consolidated in some cases and federated in others. Therefore, a common set of transformation and mapping functionality is required in both cases to maintain consistency across the data used by the business.

In the following sections, we briefly describe scenarios where data consolidation and distributed access are appropriate, and then provide an overview of DB2 Information Integration products.

3.3.1 Data consolidation or placement

Data consolidation or placement brings together data from a variety of locations into one place, in advance, so that a user query does not always need to be distributed. This approach corresponds to ETL and replication functionality. You can use ETL to build a warehouse, replication to keep it automatically updated on a scheduled basis, and extend it with federation for queries that require data that did not make enough sense to put in the warehouse.

Scenarios where ETL or replication approaches are appropriate include the following:

- Access performance or availability requirements demand centralized or local data.
- Complex transformation is required to achieve semantically consistent data.
- Complex, multidimensional queries are involved.
- Currency requirements demand point-in-time consistency such as at the close of business.

3.3.2 Distributed access (federation)

Very simply, federation takes a query in one location and distributes the appropriate parts of it to act upon the data wherever and in whatever form it resides.

Scenarios where distributed access approaches are appropriate include the following:

- Access performance and load on source systems can be traded for an overall lower implementation cost.
- Data currency requirements demand a fresh copy of the data.
- Widely heterogeneous data
- Rapidly changing data
- Data security
- Licensing restrictions, or industry regulations restrict data movement.
- Unique functions must be accessed at the data source.
- Queries returning small result sets among federated systems.
- Large volume data that are accessed infrequently.

3.3.3 DB2 Information Integration products

IBM's Information Integration solution consists of a number of products and technologies that fall under a solution umbrella called IBM DB2 Information Integration as shown in Figure 3-2.

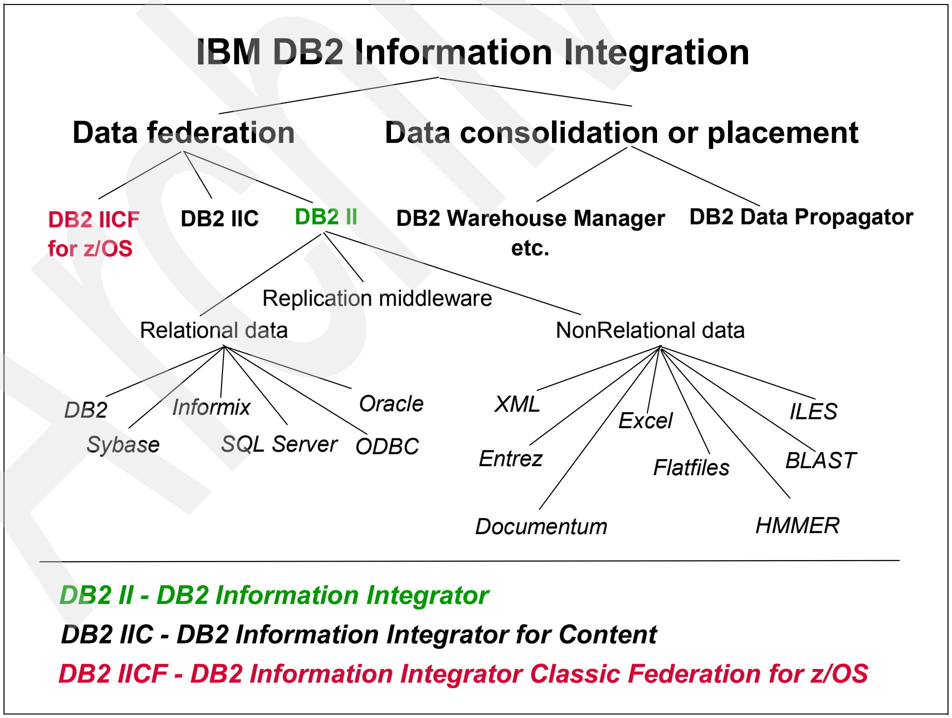


Figure 3-2 Overview of IBM Information products for information integration

There three main products that fall under the federation approach are:

- **DB2 Information Integrator (DB2 II)**

  DB2 II is targeted at the application development community familiar with relational database application development. Applications that use SQL, or tools that generate SQL such as integrated development environments and reporting and analytical tools can now access and manipulate distributed and diverse data through a federated data server.

- **DB2 Information Integrator for Content (DB2 IIC)**

  DB2 IIC is targeted at the content application developer (mainly one who works with DB2 Content Manager) who needs to search for and access text and augment it with other content or relational sources. In addition to federated search, it also offers sophisticated information mining to discover new metadata from text documents and advanced workflow (based on WebSphere MQ Workflow) to facilitate building content-centric processes. DB2 IIC represents a renaming and repositioning of the Enterprise Information Portal (EIP) offering.

- **DB2 Information Integrator Classic Federation for z/OS® (DB2 IICF)**

  DB2 IICF supports read and write access to relational and non-relational mainframe data sources such as IMS™, VSAM, Adabas, CA-IDMS, and CA-Datacom.

> **Note:** This redbook only focuses on DB2 II, hence, the added detail on it found in Figure 3-2.

3.4 DB2 Information Integrator V8.1

In this section we provide an overview of DB2 Information Integrator V8.1, describe its main components, discuss the steps involved in configuring a data source, and review some of the performance considerations.

The topics covered are:

- DB2 II V8.1 overview
- DB2 II components
- Configuring the federated system
- Performance considerations

3.4.1 DB2 II V8.1 overview

DB2 II's federated technology enables customers to abstract a common data model across diverse and distributed data and content sources, and to access and manipulate them as though they were a single source.

As mentioned earlier, with the data federation capability, the federated system acts as a virtual database with remote objects configured similar to local tables as shown in Figure 3-3.

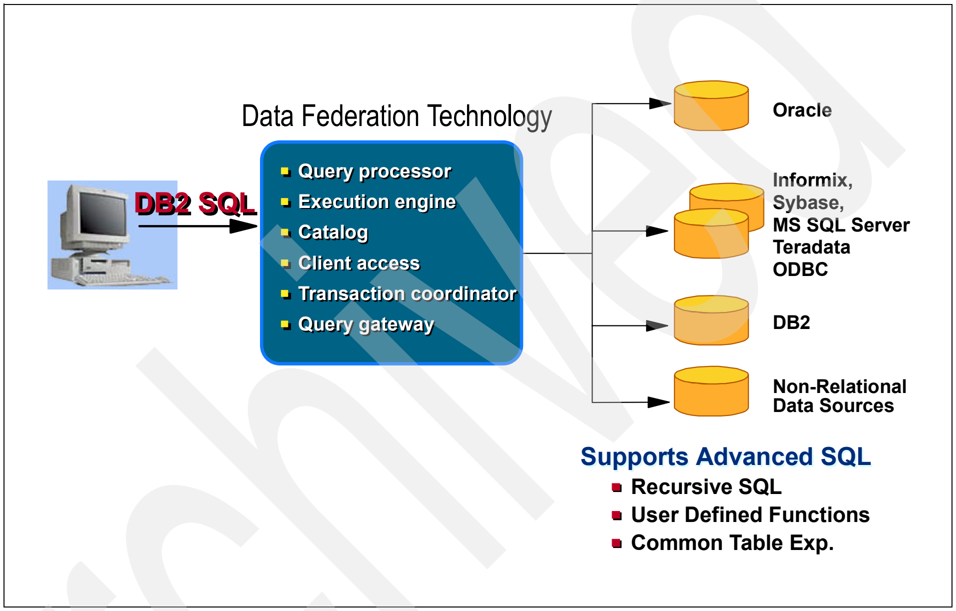


Figure 3-3 Data federation technology

With a federated system, you can send distributed requests to multiple data sources within a single SQL statement; for example, you can join data that is located in a DB2 UDB table, an Informix® table, and an XML tagged file in a single SQL statement.

When an application submits a query to the federated system, the federated DB2 identifies the relevant data sources, and develops a query execution plan for obtaining the requested data. The plan typically breaks the original query into fragments that represent work to be delegated to individual data sources, as well as additional processing to be performed by the federated DB2 to further filter, aggregate, or merge the data. The ability of the federated DB2 to further process data received from sources allows applications to take advantage of the full power of the query language, even if some of the information requested comes from data sources with little or no native query processing capability, such as simple text files. The federated DB2 has a local data store to cache query results for further processing.

A DB2 federated system is a special type of DBMS. A federated system consists of the following:

- A DB2 instance that operates as a federated server.
- A database that acts as the federated database for various relational and non-relational data sources.
- Clients (users and applications) that access the database and data sources. A nickname is the mechanism used by the clients to reference a remote data source object as if it were a local table.

The federated DBMS communicates with the data sources by means of software modules called “wrappers” as shown in Figure 3-4.

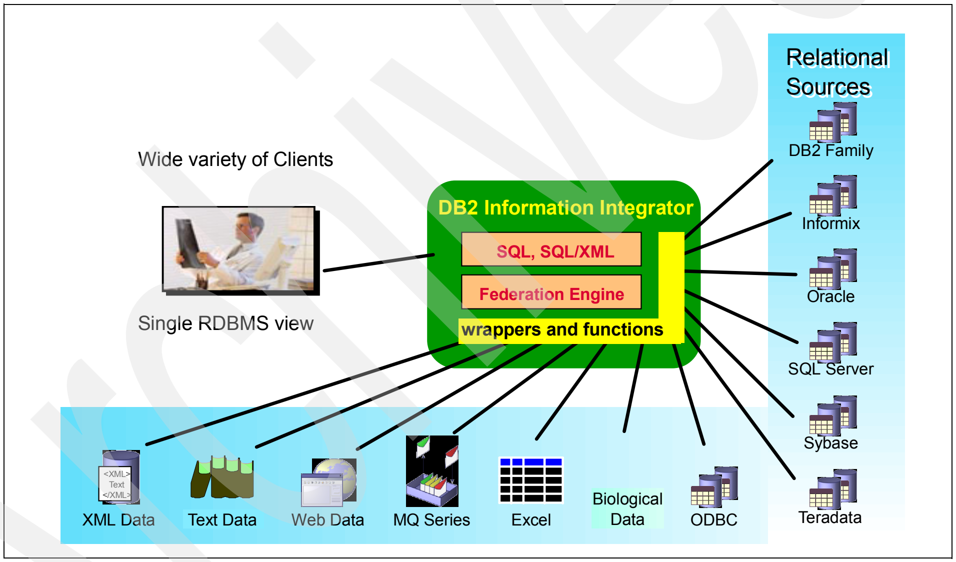


Figure 3-4 DB2 Information Integrator data federation using wrappers

Wrappers are mechanisms by which the federated server interacts with the data sources. The federated server uses routines stored in a library called a “wrapper module” to implement a wrapper. These routines allow the federated server to perform operations such as connecting to a data source and retrieving data from it. The wrapper encapsulates data source information and models data as tables. It is aware of the characteristics of the data source, and it can expose unique functions. A wrapper provides the programming logic to facilitate the following tasks:

- **Federated object registration**

A wrapper encapsulates the data source characteristics from the federated engine. A wrapper knows what information is needed to register each type of data source.

- **Communication with the data source**

  Communication includes establishing and terminating connections with the data source, and maintaining the connection across statements within an application if possible.

- **Services and operations**

  Depending on the capabilities of the type of data sources that a wrapper is meant to access, different operations are supported. The operations can include sending a query to retrieve results, updating remote data, transaction support, large object manipulation, input value binding, compensation,[1] and more.

11. Data modelling

A wrapper is responsible for mapping the data representation of the result of remote queries into the table format as required by the federated engine.

Wrappers are available for each type of data source. For example, if you want to access three DB2 for z/OS database tables, one for DB2 for iSeries™ table, two DB2 UDB for Windows tables, two Informix tables, and one Informix view, you need to define only two wrappers: one for the DB2 data source objects and one for the Informix data source objects. Once these wrappers are registered in the federated database, you can use these wrappers to access other objects from those data sources.

DB2 Information Integrator V8.1 includes the ability to federate, search, cache, transform, and replicate data. As a federated data server, it provides out-of-the box access to DB2 Universal Database™, IBM Informix products, as well as databases from Microsoft, Oracle, Sybase, and Teradata. In addition, it can also access semi-structured data from WebSphere MQ messages, XML documents, Web services, Microsoft Excel, flat files, ODBC, or OLE DB sources, plus a variety of formats unique to the life sciences industry. Integrated support for IBM Lotus Extended Search provides the solution's broad content access to a variety of content repositories, including DB2 Content Manager, as well as e-mail databases, document repositories, third-party Internet search engines, and LDAP directories.

[1] Compensation is the ability by DB2 to process SQL that is not supported by a data source. DB2 compensates for lack of functionality at the data source in two ways — one way is to ask the data source to use one or more operations that are equivalent to the DB2 function stated in the query, and another way is to return the set of data from the data source to the federated server and perform the function locally.

Note: Applications can insert, update, or delete rows in federated relational databases, however, this is limited to single-site updates with only one-phase-commits.

DB2 II V8.1 is supported on the Linux, UNIX®, and Windows platforms.

Table 3-1 lists the data sources supported, their corresponding versions, and the access method used by IBM DB2 Information Integrator V8.1 to access the supported data sources.

Table 3-1 Data sources, supported versions, and access method

Data source	**Supported versions**	**Access methods**
DB2 Universal Database for Linux, UNIX, and Windows	7.1, 7.2, 8.1	DRDA®
DB2 Universal Database for z/OS and OS/390®	6.1, 7.1 with the following APARs applied: PQ62695 PQ55393 PQ56616 PQ54605 PQ46183 PQ62139	DRDA
DB2 Universal Database for iSeries	4.5 (or later) with the following APARs applied: SA95719 SE06003 SE06872 SI05990 SI05991	DRDA
DB2 Server for VM and VSE	7.1 (or later) with fixes for APARs for schema functions applied.	DRDA
Informix	7, 8, 9	Informix Client SDK
ODBC	3.x	ODBC driver for the data source, such as Redbrick ODBC Driver to access Redbrick.
OLE DB		OLE DB 2.0 (or later)

Table 3-1 Data sources, supported versions, and access method

Data source	Supported versions	Access methods
Oracle	7.3.4, 8.x, 9.x	SQLNET or NET8 client software
Microsoft SQL Server	6.5, 7.0, 2000	On Windows, the Microsoft SQL Server Client ODBC 3.0 (or later) driver. On UNIX, the DataDirect Technologies (formerly MERANT) Connect ODBC 3.7 (or later) driver.
Sybase	11.x,12.x	Sybase Open Client
Teradata	V2R3, V2R4	Teradata Call-Level Interface Version 2 (CLIv2) Release 04.06 (or later)
BLAST	2.x	BLAST daemon (supplied with the wrapper)
Documentum	Documentum server: EDMS 98 (also referred to as version 3) and 4i.	Documentum Client API/Library
Entrez	1.0	None
HMMER	2.2g	HMMER daemon (supplied with the wrapper)
IBM Lotus Extended Search	4.0	Extended Search Client Library (supplied with the wrapper)
Microsoft Excel	97, 2000	Excel 97 or 2000 installed on the federated server
Table-structured files		None
XML	1.0 specification	None

Note: For specific details on operating system versions and data sources supported, refer to the *IBM DB2 Information Integrator: Installation Guide*, GC18-7036.

Table 3-2, Table 3-3, and Table 3-4 list some of DB2 II's restrictions as they relate to data types supported and data source writes.

Table 3-2 Unsupported data types on specific data sources

Data	Unsupported data types
DB2 for iSeries	VARG data
Extended Search	DECIMAL
Microsoft SQL Server	SQL_VARIANT
Oracle (NET8 wrapper only)	LONG LONG RAW NCHAR NVARCHAR2 TIMESTAMP (fractional_seconds_precision) WITH TIME ZONE TIMESTAMP (fractional_seconds_precision) WITH LOCAL TIME ZONE
Oracle (SQLNET wrapper only)	BLOB CLOB NCHAR NVARCHAR2 TIMESTAMP (fractional_seconds_precision) WITH TIME ZONE TIMESTAMP (fractional_seconds_precision) WITH LOCAL TIME ZONE
Sybase	unichar univarchar

Table 3-3 Write operation restrictions on data type on specific data sources

Data source	Restriction on data types
Informix	BLOB CLOB TEXT
Microsoft SQL Server	image ntext text SQL_VARIANT
ODBC	SQL_LONGBINARY (length > 255) SQL_LONGVARCHAR (length > 255) SQL_WLONGVARCHAR (length > 255)

Data source	Restriction on data types
Oracle (NET8 wrapper only)	INTERVAL DAY (day_precision) TO SECOND (fractional_seconds_precision) INTERVAL YEAR (year_precision) TO MONTH LONG LONG RAW NCHAR NVARCHAR2 TIMESTAMP (fractional_seconds_precision) WITH TIMEZONE TIMESTAMP (fractional_seconds_precision) WITH LOCAL TIME ZONE
Oracle (SQLNET wrapper only)	BLOB CLOB INTERVAL DAY (day_precision) TO SECOND (fractional_seconds_precision) INTERVAL YEAR (year_precision) TO MONTH NCHAR NVARCHAR2 TIMESTAMP (fractional_seconds_precision) WITH TIME ZONE TIMESTAMP (fractional_seconds_precision) WITH LOCAL TIME ZONE
Sybase (CTLIB wrapper only)	image text unichar univarchar
Sybase (DBLIB wrapper only)	All data types. Write operations are not supported by the DBLIB wrapper.
Teradata	char (length 32673-64000) varchar(length 32673-64000)

Table 3-4 Excel write operations limitations

Type	Read	Insert	Update	Delete
Accessing Excel through ODBC Wrapper	Yes	Yes	Yes	No
Accessing Excel through Excel Wrapper	Yes	No	No	No

> **Note:** A wrapper development kit is included in DB2 II, which allows you to develop your own wrapper modules for non-relational data sources. The wrapper module is a shared library with specific entry points that provide access to a class of data sources. DB2 UDB loads it on demand dynamically. The wrapper module is what you will be developing using the specific classes supplied with DB2. It will contain specific building blocks that allow it to act as a translator between your data source and the federated system.

The power of DB2 II is in its ability to:

- Join data from local tables and remote data sources, as if all the data is stored locally in the federated database.
- Update data in relational data sources, as if the data is stored in the federated database.
- Replicate data to and from relational data sources.
- Take advantage of the data source processing strengths by sending distributed requests to the data sources for processing.
- Compensate for SQL limitations at the data source by processing parts of a distributed request at the federated server.

3.4.2 DB2 II components

DB2 II contains the following components as shown in Figure 3-5:

- **DB2 UDB Enterprise Server Edition (ESE)** for Linux, UNIX, and Microsoft Windows that you can use to create and manage non-partitioned or partitioned database environments.
- The **relational wrappers** are used for non-IBM relational databases. In DB2 UDB Enterprise Server Edition (ESE) V8 for Linux, UNIX, and Windows, relational wrappers are required if you want to access data that is stored in Oracle, Sybase, Microsoft SQL Server, ODBC, and Teradata data sources.

- **Non relational wrappers** are used by the DB2 federated system to integrate non-relational data sources, such as flat files and XML files, and genetic, chemical, biological, and other research data from distributed sources.
- **Global catalog** is the catalog in the federated database that holds information about the entire federated system. The global catalog holds information about the objects (tables, indexes, functions, etc.) in the federated database as well as information about objects (wrappers, remote servers, nicknames and their relationships) at the data sources. The information stored is about local and remote column names, column data types, column default values, and index information.
- **DB2 Net Search Extender** is used to perform SQL-based searches on full-text documents across your enterprise. DB2 Net Search Extender performs searches efficiently by using text indexes, which Net Search Extender updates dynamically, and stores in-memory reducing scans and physical read operations.

DB2 Information Integrator V8.1 *extends* the data federation technology already available in DB2 UDB for Linux, UNIX, and Windows as shown in Figure 3-5.

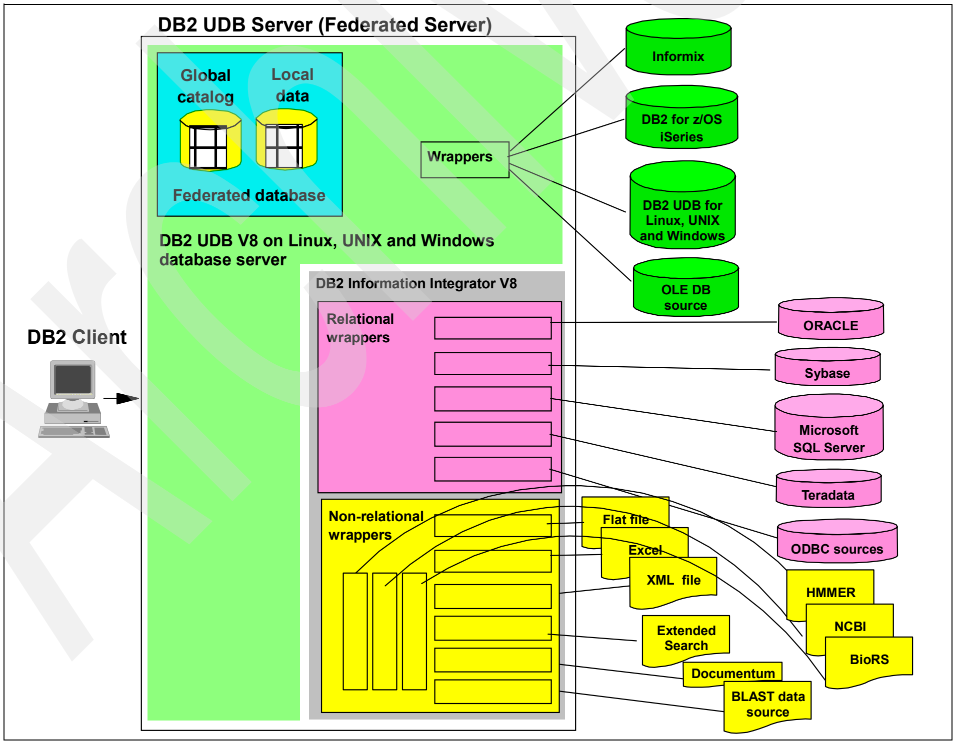


Figure 3-5 DB2 Information Integration V8.1components

The global catalog contains statistical information for nicknames, information on remote indexes for nicknames, and information on some attributes of each remote source as shown in Table 3-5. It also it contains type and function mappings.

Table 3-5 Global catalog contents for remote data sources

Federated objects	Catalog views	Descriptions
Wrappers	SYSCAT.WRAPPERS SYSCAT.WRAPOPTIONS	Registered wrappers and their specific options. (wraptype='R'/'N' for Relational/Non-relational wrapper)
Servers	SYSCAT.SERVERS SYSCAT.SERVEROPTIONS	Registered remote data sources and their specific options
User mappings	SYSCAT.USEROPTIONS	Registered user authentications for specific servers for a DB2 user. The password setting is stored encrypted.
Nicknames	SYSCAT.TABLES SYSSTAT.TABLES SYSCAT.TABOPTIONS SYSCAT.COLUMNS, SYSSTAT.COLUMNS SYSCAT.COLOPTIONS SYSCAT.INDEXES SYSSTAT.INDEXES SYSCAT.INDEXOPTIONS SYSCAT.KEYCOLUSE	Registered nicknames are identified with TYPE='N' in SYSCAT.TABLES. SYSCAT.TABOPTIONS stores specific options about nicknames. SYSCAT.COLOPTIONS stores specific options about nicknames; for instance, the servername, remote schema, and remote table name. SYSCAT.KEYCOLUSE stores information about primary key.
Index specifications	SYSCAT.INDEXES SYSSTAT.INDEXES	Index specifications created for nicknames
Type mappings	SYSCAT.TYPEMAPPINGS	User-defined type mappings used in nickname registration and transparent DDL. Default built-in type mappings are not stored in these catalog views Mapping direction = 'F'/'R'

Federated objects	Catalog views	Descriptions
Function templates	SYSCAT.FUNCTIONS SYSCAT.ROUTINES	Registered user-defined functions. In V8, SYSCAT.ROUTINES supersedes SYSCAT.FUNCTIONS in V8, but SYSCAT.FUNCTIONS still exists, not documented
Function mappings	SYSCAT.FUNCMAPPINGS SYSCAT.FUNCMAPOPTIONS SYSCAT.FUNCMAPPARMOPTIONS	User-defined function mappings to map a local function to a remote function.
Passthru privileges	SYSCAT.PASSTHRUAUTH	Authorization to allow users to query a specific server using PASSTHRU.

This information is collected when the federated system is configured as discussed in 3.4.3, “Configuring the federated system” on page 48. This information can be queried by issuing queries against the catalog.

The DB2 query optimizer uses the information in the global catalog and the data source wrapper to plan the optimal way to process SQL statements. Execution plans for federated queries are chosen by the same DB2 optimizer that optimizes regular queries — the difference is that the federated engine uses the native client interface to each target data source, and sends queries to it in its own dialect.

Figure 3-6 summarizes some of the DB2 II components on a Windows platform.

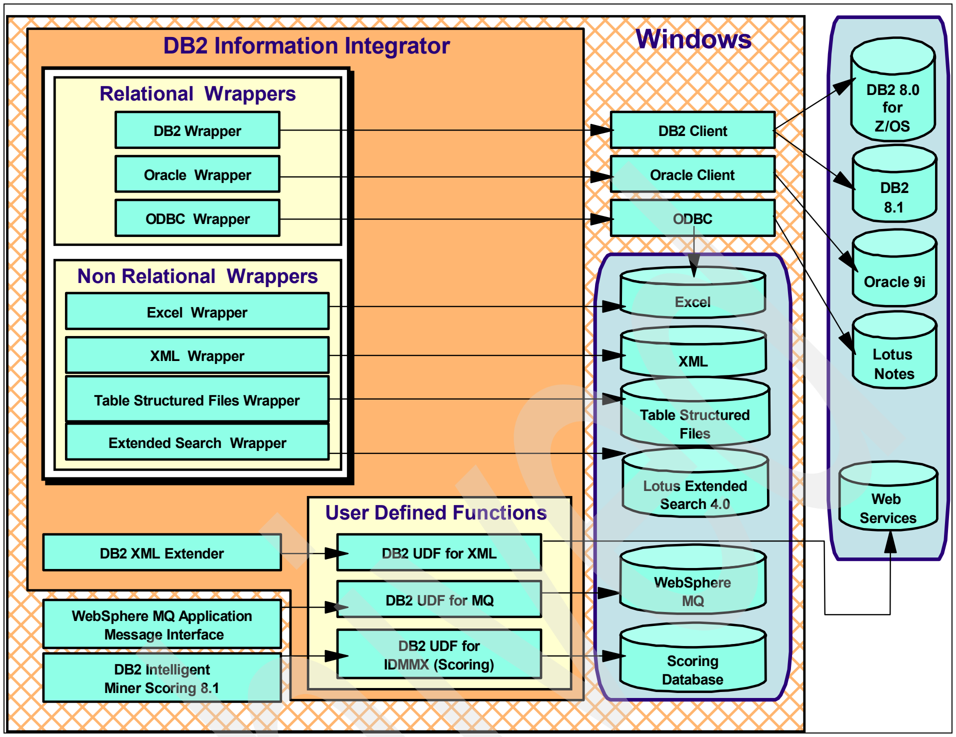


Figure 3-6 DB2 Information Integrator on a Windows platform

3.4.3 Configuring the federated system

The DB2 federated server allows you to access and join data from relational and non-relational data sources. By setting the database manager configuration parameter `FEDERATED` to `YES`, the DB2 instance (without DB2 II) allows federated access to other DB2 sources, Informix, and any OLE DB source as shown in Figure 3-5.

> **Attention:** If you need access to other non-relational or non-IBM relational sources such as Oracle, Sybase, or Microsoft SQL databases as well as generic ODBC access, and Teradata, then you need to install DB2 II.

Figure 3-7 highlights the basic steps involved in configuring the federated system. Some of these steps may be optional depending upon the data source being configured. Most of the steps to configure access to a data source can be accomplished through the DB2 Control Center. Use the DB2 Command Center for the steps that require a command line.

> **Attention:** Before configuring access to a data source, ensure that the federated server has been set up properly. It is especially important to:
>
> - Link DB2 to the client software. This creates the data source wrapper libraries on the federated server.
> - Set up the data source environment variables.
>
> For further details, refer to the *IBM DB2 Information Integrator: Installation Guide,* GC18-7036.

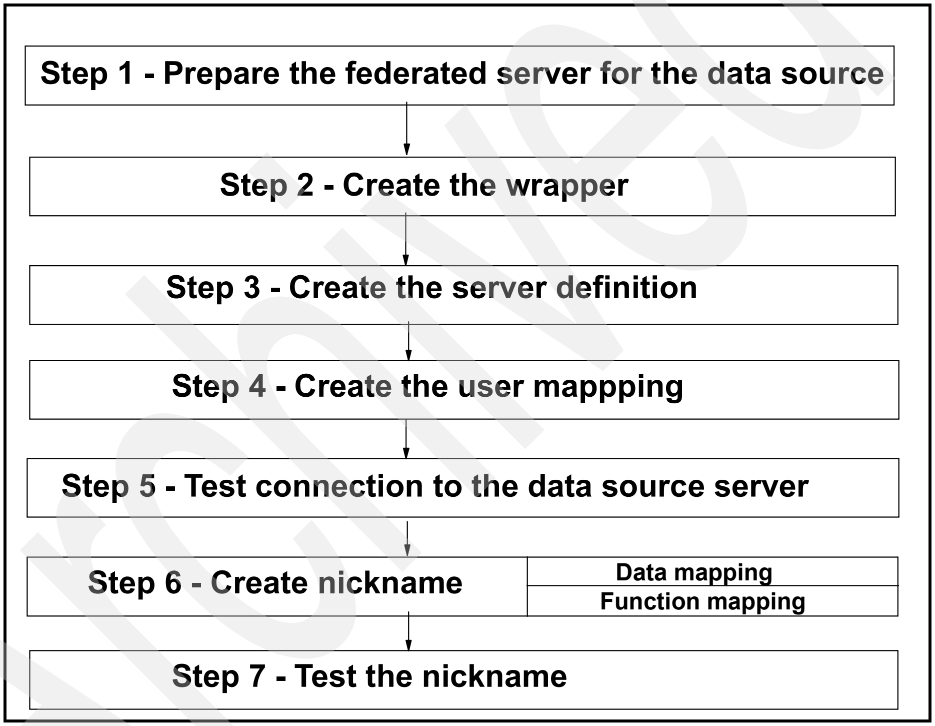


Figure 3-7 Basic steps in configuring a federated system

Each of these steps is described briefly:

1. **Step 1** involves preparing the federated server for the data source. For the DB2 family, this involves cataloging the node and the remote database. For Informix, Sybase, Microsoft SQL Server data sources, it involves setting up and testing the client configuration file.
2. **Step 2** involves creating the wrappers in the federated server. One wrapper is created for each type of data source to be accessed. When a wrapper is

be used to access objects from these data sources.

created, it is registered in the federated database and the wrappers can now be used to access objects from these data sources.

3. **Step 3** involves creating the server definition which defines the data source to be accessed by the federated database. The name of the data source and other information is part of the server definition:

   – For a relational DBMS (RDBMS), it includes the type and version of the RDBMS, the database name for the data source on the RDBMS, and metadata that is specific to the RDBMS. A DB2 data source can have multiple databases and therefore a database name is required to identify it as the target. An Oracle data source can only have a single database and a database name is therefore not included in the federated server definition of an Oracle data source.

   – For non-relational data sources, you must register a server object because the hierarchy of federated objects requires that specific files that you want to access must be associated with a server object.

   During the creation of a server definition of a relational data source, server options can be used to set server attributes that contain information about the data source location, connection security, and some server characteristics that affect performance. These characteristics and restrictions are used by the query compiler in planning the query.

   COLLATING_SEQUENCE, PUSHDOWN and DB2_MAXIMAL_PUSHDOWN are a few of the server attributes discussed here. For more details about server options, refer to *IBM DB2 Information Integrator Federated Systems Guide*, SC18-7364:

   – The COLLATING_SEQUENCE attribute can be used to adjust the collating sequence of a data source.

     Data sources, such as DB2 for z/OS, use a collating sequence based on the EBCDIC encoding scheme. The default setting for the server option COLLATING_SEQUENCE is `N` for such sources because it is not usual for DB2 for z/OS to have a database with similar encoding. DB2 UDB for Linux, UNIX, and Windows uses mostly ASCII encoding, and the default sorting order is dictionary sort. Although some data sources like Oracle use ASCII encoding, their sorting order is different from that of DB2 UDB for Linux, UNIX, and Windows.

     It is possible to create a DB2 II database with a sorting order to match that of sources like Oracle. You will need to specify **`COLLATE USING IDENTITY`** in the **`CREATE DATABASE`** statement at the time the federated database is created. In this case, the server option COLLATING_SEQUENCE can be set to `Y` for sources with ASCII encoding and identity sorting order. This server option allows range comparison such as “`string_col > string_constant`) and “`LIKE`” predicates to be executed remotely.

If the remote data source is case insensitive and the DB2 federated database is set to use case sensitive search collating sequence, the equality comparison operations will also be executed locally.

– Pushdown is a very important aspect of federated query processing. If the PUSHDOWN server option is set to `Y`, the DB2 optimizer will consider generating a plan that "pushes down" certain parts of the query execution to the remote source. The intent of pushdown is to reduce network transport (trips) and exploit the intelligence of the relational remote sources. Pushdown analysis (PDA) is the component of the DB2 optimizer that decides which parts of a query can be pushed down and processed remotely at the data sources. The decision of actually pushing down is cost-based, and influenced by information about the hardware at the remote sources, the network characteristics, and the estimated number of rows processed and returned from the remote sources. Pushdown is discussed in 3.4.4, "Performance considerations" on page 54.

– DB2_MAXIMAL_PUSHDOWN is another server option for relational nicknames that you might want to consider setting to `Y`. For queries that contain nicknames, the federated server identifies which operations in the query can be pushed down to the data sources during the PDA phase. During the subsequent cost optimization phase, this option influences whether the DB2 optimizer determines the execution plan based on cost (default behavior), or favors pushing down the maximum number of operations identified during the PDA phase regardless of cost.

  Setting DB2_MAXIMAL_PUSHDOWN to `Y` directs the query optimizer to favor access plans that tend to allow the remote relational data sources to evaluate as much of the query as possible. Because this setting will change the cost-based decision process in the query optimizer, it is not recommended as the first phase of customization. If your remote data source is as powerful as the federated server, it might make sense to set this option. Setting this option to `Y` is also useful to compare the performance of your queries. Setting this option to `Y` affects *all* queries that reference data from this remote data source.

> **Attention:** Server options are generally set to persist over successive connections to the data source, however, they can be set or overridden for the duration of a single connection.
>
> The federated system provides the SET SERVER OPTION statement for you to use when you want a server option setting to remain in effect while your application is connected to the federated server. When the connection ends, the previous server option setting is reinstated.

4. **Step 4** involves establishing a mapping between the federated server user ID and password, and the user ID and password of the data source. This association is called a user mapping and is required so that the federated server can successfully connect to the target data source. This association must be created for each user ID that will be using the federated system to send distributed requests.

   > **Note:** Each user ID accessing this nickname on DB2 II will need to be mapped to the remote data source user ID.

5. **Step 5** involves checking to see whether the federated system can connect to the target data source. A passthru session allows you to send SQL statements directly to a data source. Ensure proper privileges are granted to those users that can use the passthru session for this new data source. For example, with DB2 UDB for z/OS and OS/390, you can establish a passthru session and issue an SQL SELECT statement on the DB2 system table as follows:

   ```
   SET PASSTHRU servername
   SELECT count(*) FROM sysibm.systables
   SET PASSTHRU RESET
   ```

6. **Step 6** involves creating a nickname, which is an identifier that is used to reference an object located at the data source that you want to access. The objects that nicknames identify are referred to as data source objects. Nicknames are *not* alternative names for data source objects in the same way that aliases are alternative names. They are pointers by which the federated server references these objects. Nicknames are typically defined with the **`CREATE NICKNAME`** statement.

   Additional metadata information can be supplied about the nicknamed object through column options. Data mappings and function mappings may also be required between the target data source and DB2 data types in the federated server, if the default mappings provided in the wrappers are inadequate.

   Determine whether additional data type mappings need to be defined if you are connecting to a relational data source. Specifying additional data type mappings is necessary if you want to change the default mapping between a DB2 data type and a remote data type. If you are accessing a relational data source, determine whether you need to register additional mappings between the DB2 functions and the remote data source functions. These mappings allow data source functions to be used by the federated system:

   – **Data type mappings**

     Data types of remote data sources must correspond to DB2 data types. An appropriate mapping enables the federated server to retrieve data from the data source. These default data mappings are implemented in the wrappers. DB2 Information Integrator supplies a set of default data type mappings such as the following:

- Oracle type FLOAT maps by default to the DB2 type DOUBLE.
- Oracle type DATE maps by default to the DB2 type TIMESTAMP.
- DB2 UDB for z/OS type DATE maps by default to the DB2 type DATE.

If you want to customize the default mapping provided by DB2 II, then you need to create alternative data type mappings.

In order to use an alternative data type mapping for a nickname, you must create this mapping prior to creating the nickname. If you create the nickname first, you may set the appropriate mapping later as follows:

- Altering the nickname
- Changing default mapping types and recreating the nickname

For further details on data mappings, refer to the *IBM DB2 Information Integrator Data Source Configuration Guide Version 8*, available as softcopy from the Web site:
http://www.ibm.com/software/data/integration/solution

– **Function mappings**

DB2 Information Integrator supplies default mappings between existing built-in relational data source functions, and their built-in DB2 counterpart functions. These default function mappings are implemented in the wrappers.

You can create a function mapping if there is no default mapping available. There are several reasons for creating function mappings as follows:

- No DB2 function corresponding to a remote data source function is available.
- A corresponding DB2 function is available, but with a specification that is different from that of its remote counterpart.
- A new built-in function becomes available at the data source.
- A new user defined function becomes available at the data source.

The DB2 catalog view for function mappings is **SYSCAT.FUNCMAPPINGS.**

Function mappings are one of several inputs to the pushdown analysis performed by the query optimizer. If your query includes a function or operation, the optimizer evaluates if this function can be sent to the data source for processing. If the data source has the corresponding function available, then the processing of this function can be pushed down to help improve performance.

A DB2 function template can be used to force the federated server to invoke a data source function. Function templates do not have executable code, but they can be the object of a function mapping. After creating a DB2 function template, you need to create the actual function mapping between the template and the corresponding data source function.

The **CREATE FUNCTION MAPPING** statement gives considerable control over the scope of the mapping. For example, you can:

- Create a function mapping for all data sources of a specific type such as all Informix data sources.
- Create a function mapping for all data sources of a specific type and version, such as all Oracle 9 data sources.
- Create a function mapping for all data source objects located on a specific server.
- Disable a default function mapping. Default function mappings cannot be dropped.

For further details on function mappings, refer to the *IBM DB2 Information Integrator Data Source Configuration Guide Version 8*, available as softcopy from the Web site:
http://www.ibm.com/software/data/integration/solution

For further details on column functions, data mapping and function mapping, refer to *IBM DB2 Information Integrator Federated Systems Guide*, SC18-7364.

7. **Step 7** involves checking to ensure that the nickname has been configured correctly by issuing an SQL statement against it as follows:

```
SELECT count(*) FROM nickname
```

Note: This query may not be appropriate for some data sources; for example, Lotus Extended Search.

Attention: Appendix B, “Configuring data sources in DB2 Information Integrator” on page 225 provides examples of configuring a number of the data sources used in the CFS portal described in 5.3, “The CFS portal” on page 89.

3.4.4 Performance considerations

Probably the most significant concern about federated technology is the issue of acceptable performance. IBM invests heavily in query optimization research and development.

The DB2 Information Integrator optimizer takes into account standard statistics from source data (such as cardinality or indexes), data server capability (such as join features or built-in functions), data server capacity, I/O capacity, and network speed. The following capabilities of the DB2 optimizer have a significant impact on the quality of the access plan generated:

- **Query rewrite** logic rewrites queries for more efficient processing. For example, it can convert a join of unions, which drives a tremendous amount of data traffic into a union of joins that leverages query power at the data server, and thereby minimizes data traffic back to the federated server. The database administrator (DBA) can define materialized query tables (MQTs), which the DB2 optimizer can transparently leverage through query rewrite to satisfy user queries.
- **Pushdown analysis** (PDA) capability identifies which operations can be executed at the data server prior to returning results to the federated server. The DB2 optimizer can perform a nested loop join that queries a small table on one server, and uses the results as query predicates to a large table on another.

This section describes performance factors influencing federated queries as follows:

- Performance factors
- Pushdown concept
- Federated server options for best performance
- Nickname column options for best performance
- Indexes and statistics

Performance factors

Factors that influence federated query performance include:

- The processing power of local and remote machines, as well as the bandwidth of the intervening communication network.
- Quality of the generated query execution plans at the federated server and the remote sources. The query execution plans influence the number of interactions required between the federated server and the remote sources, and the amount of data that is moved.

  > Data movement between the federated server and the remote source is a key performance factor.

  The amount of data moved mainly depends on two factors:

  – The amount of processing and filtering that can be pushed down (see “Pushdown concept” on page 56) to the remote data sources.

If there are some filtering predicates in the WHERE-clause, and the remote source is able to apply those predicates, then the federated server pushes down these predicates to the remote server to reduce the amount of data that needs to be shipped back.

– Data placement among multiple sources

  If you join two tables and they are both on the same data source so that the join can be done at that data source without moving the tables out, then that usually results in better performance than if the two tables resided at two different sources.

  In a join between tables that are not co-located, data from both tables must be moved to the federated server, which will then do the join.

> **Note:** The federated server *never* moves data between remote data sources, only between each remote data source and itself.

DB2 Information Integrator has some efficient techniques for performing this data movement as follows:

– Nested loop join in which the results of SQL sent to one data source are supplied as values for host variables sent in SQL to the second data source.

– Use of hash-joins to obtain a join result from two data sources.

Pushdown concept

Pushdown is an important aspect of federated query processing. As mentioned earlier, if the PUSHDOWN server option is set to Y, the DB2 optimizer will consider generating a plan that "pushes down" certain parts of the query execution to the remote source.

The "pushdown analysis" (PDA) component (described earlier in 3.4.3, "Configuring the federated system" on page 48) decides which parts of a query can be pushed down to the data sources and processed remotely at the data sources.

The decision to pushdown certain parts and operators of a query depends on several factors as follows:

- The availability of required functionality at the remote source

  If the remote source is simply a file system with a flat file, then it is probably not possible to pushdown any filtering predicates.

- The options specified in the server definition of a remote source as discussed in "Federated server options for best performance" on page 57.

For instance, if the collating sequence at the federated server is different from the one at the remote server, then operations on string data like sorting, and some predicates involved in the query have to occur at the federated server, and cannot be pushed down.

- The following issues influence pushdown:
  - You can specify that a remote column always contains numeric strings so that differences in the collating sequence do not matter. Sometimes remote source functionality is dependent on data type issues; for example, a remote function may accept only certain argument types.
  - Attributes within the DB2 Information Integrator wrappers that indicate which operations and functions are supported by the type and version of the data source.
  - Function mappings in the DB2 Information Integrator catalog

    The function mappings in the catalog are created by the DB2 Information Integrator administrator, and are additions and overrides to the default function mappings that are in the wrappers.

For further details, refer to *Using the federated database technology of IBM DB2 Information Integrator*, white paper by Anjali Grover, Eileen Lin, and Ioana Ursu, available from the Web site:
http://www-3.ibm.com/software/data/pubs/papers/#iipapers

Federated server options for best performance

Each remote data source has an entry in the federated server's catalog. Using special DDL, you can add more entries to the catalog that describe attributes of the server, called *server options*.

This section discusses the most common federated server options that can significantly impact the decisions made by the DB2 optimizer, and thereby on query performance:

- COMM_RATE, CPU_RATIO, IO_RATIO

  These attributes describe the communication links to the remote source, and the relative speed of the remote system's CPU and I/O. By default, the federated server assumes that the remote machine is equal in power to the local machine, and that there is a 2 MB/sec link to it. Setting these options to indicate a more powerful remote machine, or a faster link will tend to encourage query pushdown. These knobs are not perfect, but they are a way to indicate to the DB2 optimizer that a remote machine is fast or slow.

- COLLATING_SEQUENCE

  If you set this attribute to `Y`, you are telling the PDA that the remote source sorts characters the same way that DB2 does. This means that the federated server can consider pushing down operations involving sorting, grouping, or inequality comparisons on characters and VARCHAR columns. For instance, setting COLLATING_SEQUENCE to `Y` allows the DB2 optimizer to pushdown ORDER BY clauses that reference character and VARCHAR columns.

Pushdown of these operations on numeric, date, time, and date/time columns is *not* affected by this server option.

> **Attention:** If the remote source's collating sequence does not match DB2's after you set COLLATING_SEQUENCE to `Y`, you can receive incorrect results!

- VARCHAR_NO_TRAILING_BLANKS

  This attribute is used for databases like Oracle that do not pad VARCHAR fields with trailing blanks. The query compiler uses this information while checking any character comparison operations to decide the pushdown strategy to evaluate the operations.

  DB2 uses blank padded comparison semantics while comparing character strings of unequal lengths. The comparison is made by using a copy of the shorter string which is padded on the right with blanks so that its length is equal to that of the longer string. This means that the string "A" is considered equivalent to "A " in DB2 UDB.

  However, this behavior does not apply to all character data types across all data sources, such as the VARCHAR2 data type in Oracle.

  In general, comparison operations on string columns without blank padding comparison semantics need to be evaluated locally unless the query compiler is able to find functions to enforce similar logic remotely. For certain operations such as predicates, the federated system maintains performance by rewriting the predicates to ensure the same semantics when these predicates are sent to an Oracle server. Performance of operations such as DISTINCT, ORDER BY, GROUP BY, UNION, column functions (MIN()/MAX()) evaluation, relational comparison and IN predicates might be affected if this column option is set.

  This attribute is used for databases like Oracle that do not pad VARCHAR fields with trailing blanks. If you are sure that your VARCHAR columns do not contain trailing blanks to begin with, then setting this option at the server level will allow the use of the remote source's non-blank-padded comparison operations that return the same results as DB2.

  Here again, setting VARCHAR_NO_TRAILING_BLANKS to `Y` when trailing blanks do exist at the remote data source can return erroneous results.

Attention: This option can also be set at the nickname column level, which may be a better use of the option.

If it is set at the server level, the user would then need, for example, to ensure that *all* VARCHAR2 columns of all data objects out of this Oracle data source are guaranteed not to contain trailing blanks.

Nickname column options for best performance

The NUMERIC_STRING and VARCHAR_NO_TRAILING_BLANKS and nickname column options impact the decisions made by the DB2 optimizer and thereby query performance:

- NUMERIC_STRING

  This nickname column option applies to character data types and is applicable to those data sources for which the COLLATING_SEQUENCE server option is set to `N`.

  The federated system does *not* push down any operations that can produce different results due to differences in collating sequences between the federated database and the remote data source. Suppose that a data source has a collating sequence that differs from the federated database collating sequence, in this case, the federated server typically does *not* sort any columns containing character data at the data source. It returns the retrieved data to the federated database, and performs the sort locally.

  However, suppose that the column is a character data type (CHAR or VARCHAR) and contains only numeric characters (0 through 9). This fact can be indicated to the DB2 optimizer by setting the NUMERIC_STRING column option to `Y`. This gives the DB2 query optimizer the option of performing the sort at the data source. If the sort is performed remotely, you can avoid the overhead of porting the data to the federated server and performing the sort locally.

- VARCHAR_NO_TRAILING_BLANKS

  As discussed in “VARCHAR_NO_TRAILING_BLANKS” on page 58, this is also a server option. However, specifying this option at the column level provides greater flexibility and granularity if there are multiple tables in a database that do not all have missing trailing blanks.

Indexes and statistics

For the DB2 optimizer to make superior access path decisions, it needs knowledge about available indexes on it, and have accurate statistics about a remote object. The federated server relies on the remote source for its index and statistics information about each remote object. This information is retrieved when a nickname is created and stored in the federated server's global catalog.

> **Attention:** The information in the global catalog is *not* automatically maintained if statistics on the remote object are refreshed or indexes are added or dropped.
>
> It is the federated DBA responsibility to ensure that the statistics and metadata information in the global catalog is kept in sync with the corresponding statistics and index information of the remote data objects.

If you believe that the federated server's information is out of sync with that of the remote data source, drop and re-create the nickname to retrieve fresh information from the remote data source. This may be disruptive, since the action invalidates views and packages that reference the nickname, and also clears any authorizations (GRANTs) on the nickname.

Manually adding statistics or using an "index specification"[2] may be a non-disruptive alternative to dropping and recreating a nickname as discussed in "Indexes" on page 60, and "Statistics" on page 61.

- **Indexes**

  When you create a nickname, DB2 II retrieves information about the indexes defined on the table at the remote source. This information is stored in the federated server's global catalog as an attribute of the nickname, and is used during query optimization.

  Index information for a nickname will *not* be retrieved if:

  – The nickname for a table has no indexes.

  – The nickname is for a remote view, Informix synonym, table structured file, Excel spreadsheet, or XML tagged file.

    Views and Informix synonyms do not have index information in the data source catalog, but the tables referred to by the view or synonym may have indexes.

  – The remote index has a column with more than 255 bytes, or a total index key length with more than 1024 bytes.

  – The remote index is on LOB columns.

  Another possible case in which index information for a nickname will be missing from the catalog is when you create a new index on a remote object *after* creating the nickname for the object. DB2 II is not notified of the change,

[2] This is a set of metadata catalog information about a data source index.

and has no way of knowing that it needs to update its index information in the global catalog to include the new index.

To notify DB2 II of the existence of a missing index for a nickname, you can create an index specification to record information that includes the keys that comprise the index, but does *not* include any statistical information. Thus, creating an index specification does not record as much data about the remote index as would be obtained by dropping and recreating the nickname, but it is less disruptive. If you want to ensure that index information for nicknames includes all available statistical data, you need to follow the steps described in “Statistics” on page 61.

Similarly, when a nickname is created for a remote view, the federated server is unaware of the underlying tables (and their indexes) from which the view was generated. An index specification can be used to tell the DB2 optimizer about indexes on the underlying tables of a remote view, which may help it choose better access paths for queries involving the nickname to the remote view.

In either case, you supply the necessary index information to the global catalog using the `CREATE INDEX... SPECIFICATION ONLY` statement. No physical index is built on behalf of this nickname — only an entry is added to the system catalog to indicate to the query optimizer that such a remote index exists. This helps the query optimizer in generating remote plans for relational nicknames.

An index specification that defines a unique index also conveys the information about the uniqueness of the index columns to the federated system. Just like a regular unique index definition registered during relational nickname registration, such uniqueness information can help the query optimizer to generate a more optimal plan with strategies such as eliminating unnecessary DISTINCT operations.

- **Statistics**

  DB2 stores statistical information on objects stored in the database including tables, table columns, and indexes. These statistics help the DB2 optimizer work out the best access plan for queries. In order to help the DB2 optimizer do its job, it is necessary to keep the statistics for each object in the database up to date. DB2 stores statistical information for nicknames as well. As nicknames are really just local references for remote tables, they look much like local tables to the DB2 optimizer. In fact, statistics for both local tables and nicknames are stored in the same way, and are accessible through DB2 system catalog views in the schema `SYSSTAT`.

  DB2 stores the following types of nickname statistics:

  – Table cardinality (row count) and page counts (`SYSSTAT.TABLES`)

– Column cardinality (number of unique values) and column minima and maxima (**SYSSTAT.COLUMNS**)

– Information on remote indexes for nicknames (**SYSSTAT.INDEXES**)

The amount of statistical information stored for nicknames varies depending on the type of remote data source involved; for example, while table cardinality is available for nicknames on most sources, column minima and maxima are only available for some sources.

As mentioned earlier, nickname statistics and index information are retrieved from available information on the remote data source at the time that the nickname is created. Therefore, nickname statistics can only be as good as available remote statistics at nickname creation time. In particular, if no statistics have been collected on a remote object before a nickname is created, the nickname itself will not have any statistics. Similarly, if statistics are updated for an object on a remote data source, the new information is not automatically propagated to the corresponding DB2 nickname. Again, as discussed earlier, the same principle applies to indexes — DB2 is only aware of remote indexes for an object that are in existence at the time of nickname creation.

To make sure that DB2 nicknames have the best possible statistics and index data:

– Update statistics for objects on remote sources and create remote indexes *before* defining DB2 nicknames to them, so that DB2 can retrieve and store the current statistics information for the nickname.

– If updated statistics are collected for a remote object, or a new remote index is created, the DB2 statistics and index information for the corresponding nickname will be out of date. There is no **runstats** for nicknames. You can drop and re-create the DB2 nickname for the object to re-start the remote statistics and index discovery process, and retrieve the updated information. As mentioned earlier this can be quite disruptive due to side effects such as view and package invalidation, and loss of GRANTs for the nickname.

Some remote data sources store little or no statistical data for their objects, or the information is not available to an external client. In this case, DB2 II is not able to retrieve statistical data at nickname creation time. An alternative in this situation is to:

a. Use a special tool called get_stats that populates DB2 nickname statistics by issuing queries directly against the nickname. This tool is available at:

http://www14.software.ibm.com/webapp/download/preconfig.jsp?id=2003-08-20+20%3A16%3A06.194951C&S_TACT=TrialsAndBetas&S_CMP=&s=

b. Manually update the statistics and create the index information using SQL. Manually supplying the statistics for important columns of frequently-used remote tables may help the DB2 optimizer build better query plans. Basic properties about a column, such as minima and maxima values, and the number of unique values it has can have a significant impact on the quality of the query plan, and thereby query performance.

MQT performance

A materialized query table (MQT) is a table that contains pre-computed aggregate values, or a filtered subset of base tables. An MQT's definition is based on a query that accesses one or more tables. The target of the query may also be a nickname. An MQT is refreshed using a deferred approach or an immediate approach. However, an MQT based on a nickname can only be refreshed through a deferred refresh approach.

When a query is written against base tables and views or nicknames, the DB2 optimizer will automatically rewrite the query to target the MQT instead (if appropriate) in order to satisfy the original query. This can result in significant performance gains.

> **Attention:** MQT functionality is somewhat similar to the role of a DB2 index, which provides an efficient access path that the query user is typically unaware of. However, unlike an index, a user may directly query the MQT, but this is not generally recommended since it detracts from the appeal of an MQT being a black box that a DBA creates and destroys as required in order to deliver superior query performance.

MQT support for nicknames can help improve the performance of distributed queries involving local and remote data. Figure 3-8 shows a possible four-way join across two local tables, and two nicknames referencing a remote data source.

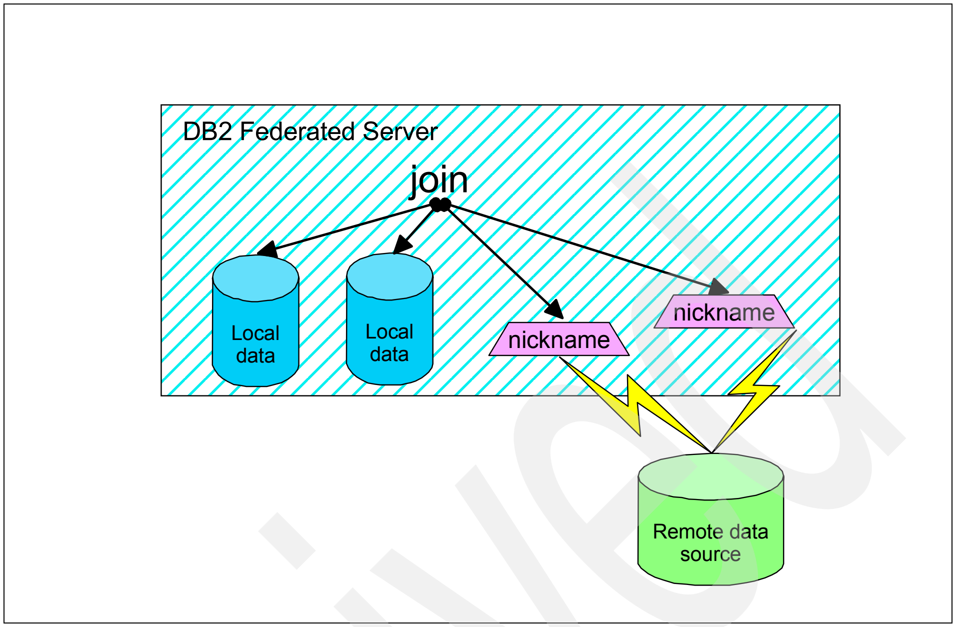


Figure 3-8 MQTs

If the four-way join query is executed frequently and the query can tolerate remote data (accessed through nicknames) being slightly out-of-date, then an MQT created on nicknames provides a local copy of the remote data as shown in Figure 3-9, and provides significant performance benefits. As mentioned earlier, an MQT based on nicknames is not automatically updated, and must be manually refreshed.

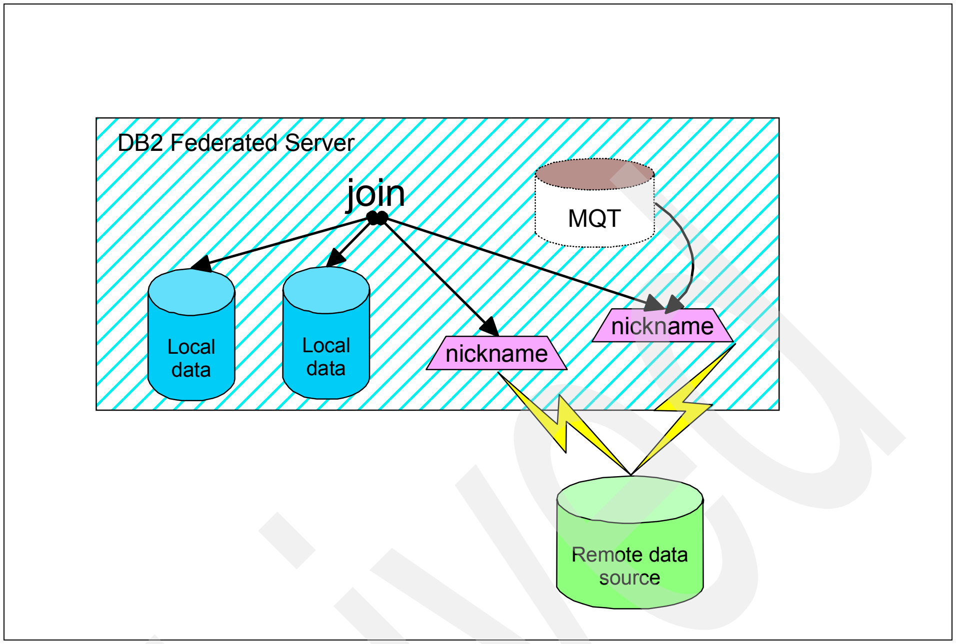


Figure 3-9 MQTs on nicknames

The advantage of using MQTs rather than replication is that the DB2 optimizer may actually choose to use the nickname (and not the local MQT) if the query joins with other remote data on the same data source. But if the nickname is joined with local data, then the DB2 optimizer will choose the MQT instead of the nickname. MQTs are like other tables and must be tuned with indexes and current statistics to ensure that the DB2 optimizer makes the correct decision about choosing the MQT.

4

WebSphere Portal and DB2 Information Integrator

In this chapter we describe the inherent synergy between WebSphere Portal and DB2 Information Integrator (DB2 II), discuss key considerations in integrating DB2 II in different WebSphere Portal topologies, and provide tips for successful deployment.

The topics covered are:

- WebSphere Portal and DB2 II synergy
- WebSphere Portal topologies and DB2 II
- Tips for successful deployment

4.1 WebSphere Portal and DB2 II synergy

Chapter 2, “WebSphere Portal architecture overview” on page 15 describes the strengths of WebSphere Portal in providing a framework for developing portals that integrate structured, semi-structured, and unstructured content from a broad range of geographically distributed data and content repositories, e-mail systems, and the Web. However, portlet developers are responsible for coding the appropriate APIs to access the relevant data sources to retrieve desired information for rendering. If a portlet needs to access multiple data sources with diverse APIs, the portlet developer may require advanced development skills to develop efficient and scalable portlets.

Chapter 3, “DB2 Information Integration architecture overview” on page 27 describes the strengths of DB2 II V8.1 in providing a federated database capability to a wide variety of relational and non-relational data sources. By implementing DB2 Information Integrator, users have the power of access to heterogeneous data sources using the simple SQL API, plus the benefits of transparency, function compensation, autonomy, extensibility, and optimization.

> **Note:** IBM experiments indicate that businesses can net a 40% to 65% reduction in hand-coding projects that have to integrate multiple data sources by implementing DB2 Information Integrator. Details of this experiment are available at:
> `http://www-106.ibm.com/developerworks/db2/library/techarticle/0305saracco1/0305saracco1.html`
>
> and
>
> `http://www-106.ibm.com/developerworks/db2/library/techarticle/0305saracco/0305saracco.html`

The challenges of addressing WebSphere Portal’s need to simply and efficiently integrate geographically distributed diverse data and content repositories, e-mail systems and the Web is clearly met by DB2 II’s federated database capability and sophisticated optimization functionality. Figure 4-1 symbolizes this obvious and inherent synergy between the WebSphere Portal and DB2 II.

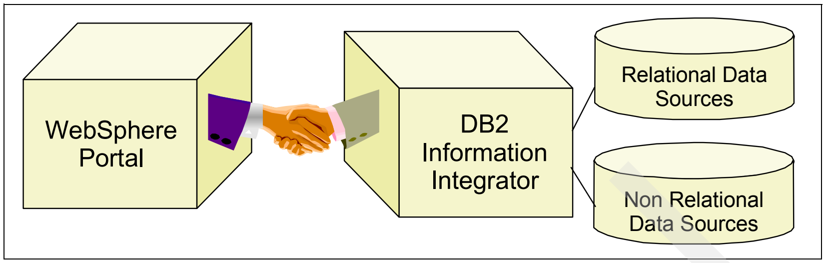


Figure 4-1 WebSphere Portal and DB2 II synergy

DB2 II will enable portal applications to query integrated views across diverse and distributed data sources as if they were a single database. The queries can be expressed using standard SQL statements. SQL expressions can be used to transform the data for business analysis or data exchange. XML documents can be created for flexible presentation. Any Web service can be converted into a function call and used as a transformation. DB2 II can sit transparently behind tools from Business Objects, Cognos, Crystal Decisions, and others, enabling end users to combine historical data with real-time data, develop reports that span departmental datamarts or warehouses, and enrich reports with relevant details from content repositories. DB2 II can therefore be used transparently by off-the-shelf portlets available from IBM as well as third party vendors.

DB2 II also works seamlessly with J2EE objects deployed in WebSphere Application Server. Thus, DB2 II supports more complex portal development scenarios, including those whose architectural requirements demand that integrated data access be encapsulated in other J2EE components, such as entity Enterprise JavaBeans (EJBs), session EJBs, and servlets. Refer to:
`http://www-106.ibm.com/developerworks/db2/library/techarticle/0302saracco/0302saracco.html` and
`http://www7b.software.ibm.com/oldwsdd/http.downloads/1/%3c%3eluzr3x5czalm374vsbnhr3p3/dmdd/db2/0302saracco/0302saracco.pdf`

Note: The My Query Reports portlet displays a list of SQL queries the user previously defined using the portlet Edit mode. When the user clicks on a query, the portlet runs the query. The portlet then displays the results of the query in a table. The user can sort the result columns by clicking on the links in the table headers.

This portlet is available from:
https://www-306.ibm.com/services/cwi/portal/.cmd/ActionDispatcher/_pagr/105/_pa.105/112/.st/X/.piid/135/.ciid/165/PC_135_searchIssued/t/PC_135_NAVCODE/1WP100039/PC_135_searchRadio/t/PC_135_searchTerms/myquery%2b%2breports#165

This portlet is commonly used in the prototyping stage, but is limited to accessing a single database. Therefore, when multiple databases need to be accessed without DB2 II, customers must write their own portlet to integrate data. The alternative is for them to include multiple panes in their portlet application, each of which use this portlet to access a different data source, which would then have to be correlated manually by scrolling the results sets on the screen. With DB2 II, different databases can be configured as a single federated database, thereby allowing the My Query Reports portlet to be used even when multiple physical databases need to be accessed.

Attention: Leveraging DB2 II in a WebSphere Portal environment cannot only greatly enhance the productivity of the portlet developer, but also has the potential to deliver superior query performance especially when multiple heterogeneous data sources are involved. As always, the performance of a query depends upon various factors such as the application workload and the infrastructure involved — your mileage will vary.

Since the federated database of DB2 II appears as just another relational data source to WebSphere Portal, there are no special data source configuration considerations involved in leveraging DB2 Information Integrator.

4.2 WebSphere Portal topologies and DB2 II

Organizations choose WebSphere Application Server (WAS) platforms and topologies based on their application characteristics, scalability, and availability requirements; for example, selecting the AIX platform for WAS and implementing both vertical and horizontal WAS clones to achieve the desired scalability and availability requirements of an online stock trading Web application.

In this section, we describe some typical AIX and Windows configurations and identify possible DB2 II integration scenarios for each topology and key technical considerations in selecting a particular DB2 II configuration.

We discuss the following four “typical” categories:

- WebSphere Portal in a Windows environment
- WebSphere Portal clones in a Windows environment
- WebSphere Portal in an AIX environment
- WebSphere Portal clones in an AIX environment

We assume that the portal accesses disparate data sources on a variety platforms.

Figure 4-2 recaps the relational and non-relational data sources supported by DB2 II.

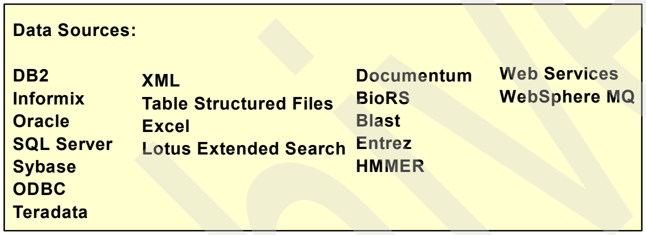


Figure 4-2 Data sources supported by DB2 Information Integrator

Note: Web Services and WebSphere MQ data sources are supported by DB2 II using UDFs.

Support for Lotus Extended Search (LES) extends support to additional data sources such as Lotus Notes® databases, Microsoft Access, Microsoft Index Server, Web search engines, and Lightweight Directory Access Protocol (LDAP) directories.

4.2.1 WebSphere Portal in a Windows environment

Figure 4-3 shows a simple WebSphere Portal configuration in a Windows environment with no cloning involved, which accesses geographically distributed relational and non-relational data sources.

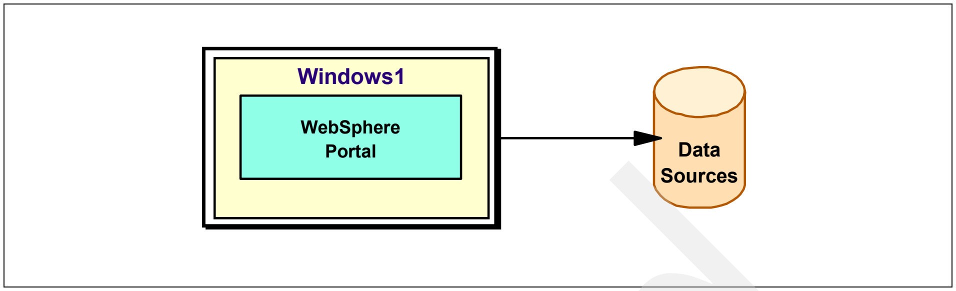


Figure 4-3 WebSphere Portal without clones in a Windows environment

Figure 4-4 shows the recommended configurations for integrating DB2 II into the environment shown in Figure 4-3.

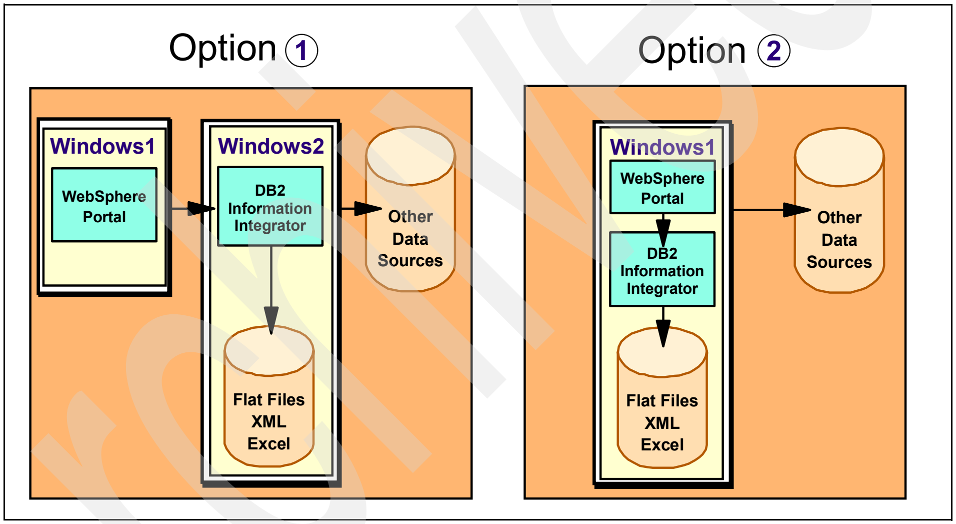


Figure 4-4 WebSphere Portal without clones in Windows and DB2 II scenarios

In Figure 4-4, Option 1 shows the WebSphere Portal on the **Windows1** server and DB2 II on a separate dedicated **Windows2** server, while Option 2 shows the WebSphere Portal and DB2 II on the same **Windows1** server.

Key technical considerations in a Windows environment

The key technical considerations in choosing between Option 1 and Option 2 are:

- **Scalability issues**

  The WebSphere Portal server should have sufficient additional CPU, memory and disk capacity to support the resource requirements of DB2 II, which are both the hardware and software prerequisites and portlet workload needs. For details on DB2 II prerequisites, refer to *IBM DB2 Information Integrator V8.1 Installation Guide Version 8*, SC18-7036.

If there is inadequate capacity on the WebSphere Portal Windows server, then DB2 II must be installed on a separate dedicated server.

- **DB2 UDB Version/fixpak conflicts**

  DB2 II V8.1 requires DB2 UDB ESE V8.1.2 as an installation prerequisite. Since a Windows server can only support a single DB2 UDB version/fixpak level, any requirement to support other DB2 UDB version/fixpak levels will require DB2 II to be installed on a separate Windows server to avoid conflicts.

8. **File access issues**

   DB2 II requires the following prerequisites for certain non-relational data sources:

   – Flat files (table structured files), XML, and Microsoft Excel files *must be* located on a local or network mapped drive of the Windows server where DB2 II is installed.

   > **Attention:** Since it is DB2 that needs to access the files on the local or network drive, you need to ensure that the *user ID under which DB2 is running* (default is db2admin) is authorized to the domain containing the network drives.

   To determine the user ID of DB2, navigate to the Services screen as follows:

   Start --> Settings --> Control Panel --> Administrative Tools --> Services.

   This displays the Figure 4-5. The Log On As field identifies the DB2 user ID (db2admin in Figure 4-5).

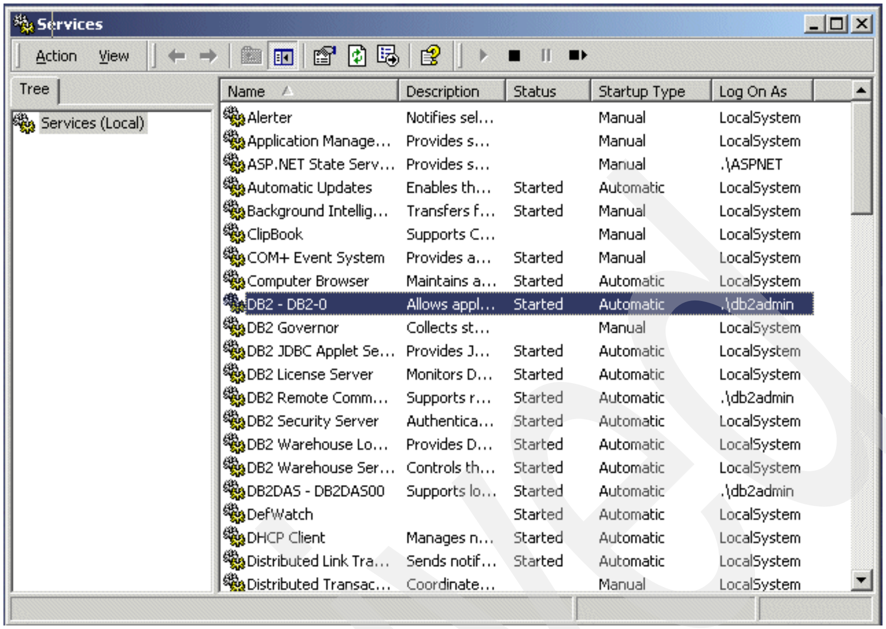


Figure 4-5 Determining user ID of DB2 II

  – Microsoft Excel *must be* installed on the Windows server where DB2 II is installed, if Microsoft Excel files need to be accessed by DB2 II. Appendix B.5.1, “Microsoft Excel data source considerations” on page 268 describes the considerations involved in configuring DB2 II to access Microsoft Excel.

  If the flat files are not located on the WebSphere Portal Windows server, or Microsoft Excel cannot be made available on it, then DB2 II will need to be located on a Windows server where these prerequisites are met.

- **Licensing and administration costs**

  When DB2 II is installed on a separate dedicated Windows server, you incur administration costs of managing an additional Windows server and may also incur (non DB2 II) software licensing costs.

4.2.2 WebSphere Portal clones in a Windows environment

The topologies discussed in this section involve two WebSphere Portal cloning configurations: a vertical cloned WebSphere Portal and a horizontal cloned WebSphere Portal environment. The recommended DB2 II scenarios for each environment are described here.

WebSphere Portal vertical cloned in Windows environments

Figure 4-6 shows the WebSphere Portal vertical cloned environment that accesses geographically distributed relational and non-relational data sources. The Network Deployment Manager of WebSphere connects to the node agent that runs on the **Windows1** Server.

> **Attention:** Vertical cloned systems provide greater scalability in large SMP servers with lots of real memory. For further details on vertical clones, refer to *IBM WebSphere 4.0 Advanced Edition Handbook,* SG24-6176.

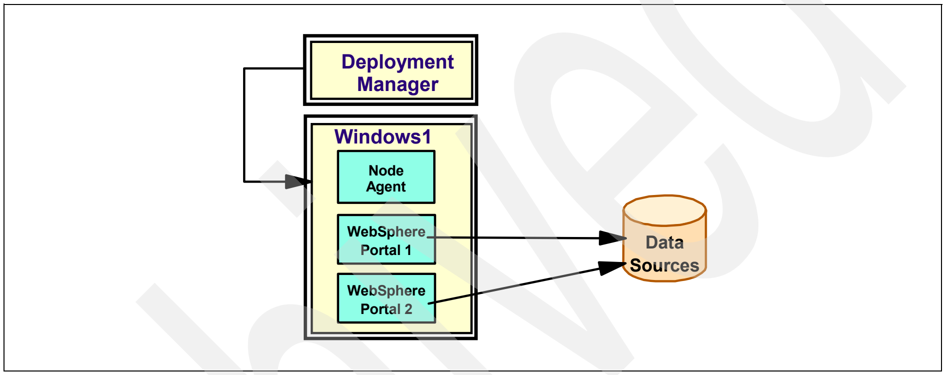


Figure 4-6 WebSphere Portal with vertical clones in a Windows environment

Figure 4-7 shows the recommended configuration for integrating DB2 II into the environment shown in Figure 4-6.

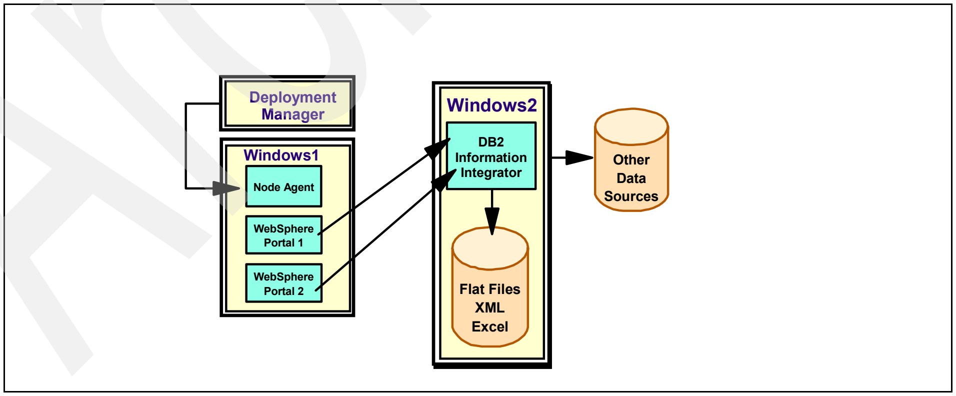


Figure 4-7 WebSphere Portal with vertical clones in Windows and DB2 II scenario

With WebSphere Portal cloned vertically on the Windows1 server, each clone will be accessing the data sources independently, resulting in many processes running concurrently on the Windows1 server.

For good performance, we recommend that DB2 II be installed on a separate dedicated Windows2 server (as shown in Figure 4-7) to handle the excess workload generated by the WebSphere Portal clones shown in Figure 4-6.

Note: We assume that any non-relational data sources on the Windows1 server needed by the portlets are accessible to DB2 II from the separate dedicated Windows2 server as shown in Figure 4-7. Otherwise, this configuration is not viable, and DB2 II will have to be installed on the Windows1 server assuming it has adequate spare capacity to support it.

WebSphere Portal horizontal cloned Windows environments

Figure 4-8 shows the WebSphere Portal horizontal cloned environment that accesses geographically distributed relational and non-relational data sources. The Network Deployment Manager of WebSphere connects to the node agent that runs on the Windows1 server and Windows2 server.

Attention: Horizontal cloned systems not only provide greater scalability but also address high availability and continuous operation requirements. For further details on horizontal clones, refer to *IBM WebSphere 4.0 Advanced Edition Handbook,* SG24-6176.

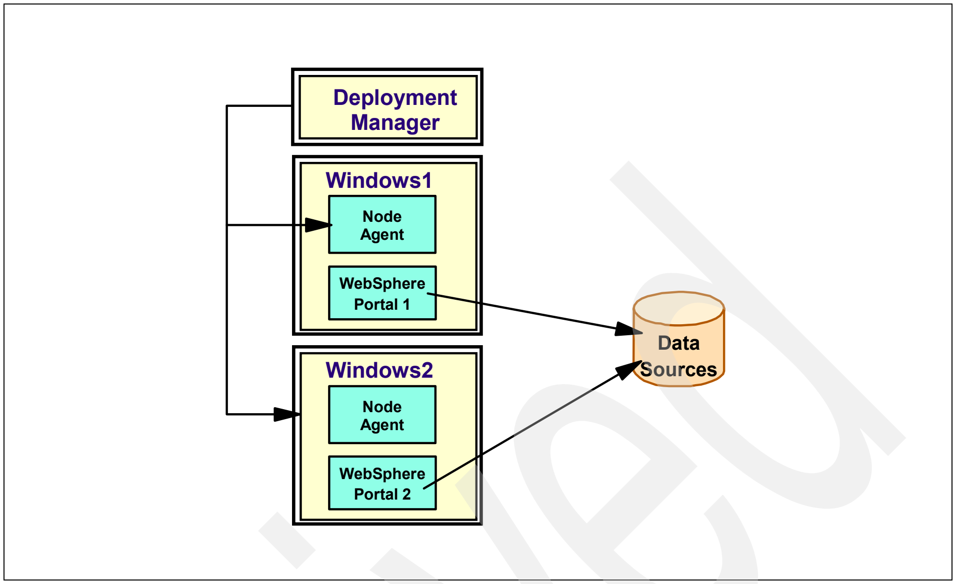


Figure 4-8 WebSphere Portal with horizontal clones in a Windows environment

Figure 4-9 shows the recommended configurations for integrating DB2 II into the environment shown in Figure 4-8.

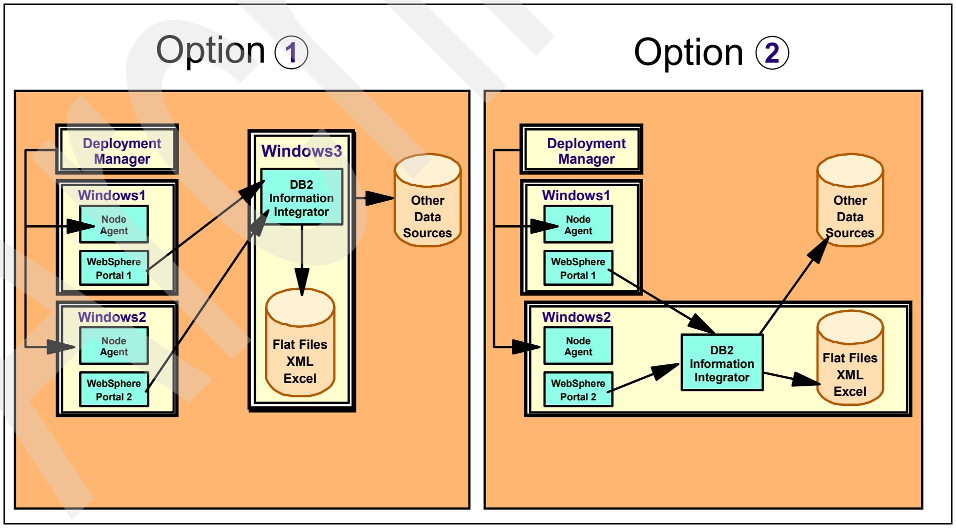


Figure 4-9 WebSphere Portal horizontal clones in Windows and DB2 II scenarios

In Figure 4-9, Option 1 shows the WebSphere Portal on the Windows1 server and Windows2 server and DB2 II on a separate dedicated Windows3 server, while Option 2 shows DB2 II sharing one of the horizontal clones' server (**Windows2)**.

The key technical considerations in choosing between Option 1 and Option 2 are the same as described in "Key technical considerations in a Windows environment" on page 72.

Option 1 assumes that the non-relational data sources (flat files, XML files, and Microsoft Excel files) are accessible by the Windows3 server.

Option 2 may be appropriate when the Windows2 server has flat files required by the portlets, and the Windows2 server has adequate capacity to support the added resources requirements of DB2 II.

> **Attention:** For organizations that choose a horizontally cloned WebSphere Portal configuration for its high availability characteristics, the DB2 II server configurations described in Figure 4-9 constitute a single point of failure. You need to implement a DB2 II high availability cluster environment to overcome this limitation.

4.2.3 WebSphere Portal in an AIX environment

Figure 4-10 shows a simple WebSphere Portal configuration in an AIX environment with no cloning involved, which accesses geographically distributed relational and non-relational data sources.

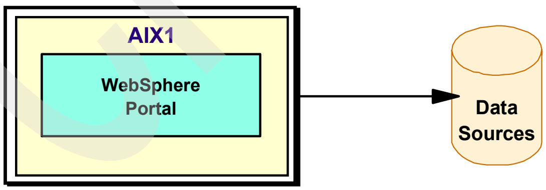


Figure 4-10 WebSphere Portal without clones in an AIX environment

Figure 4-11 shows the recommended configurations for integrating DB2 II into the environment shown in Figure 4-10.

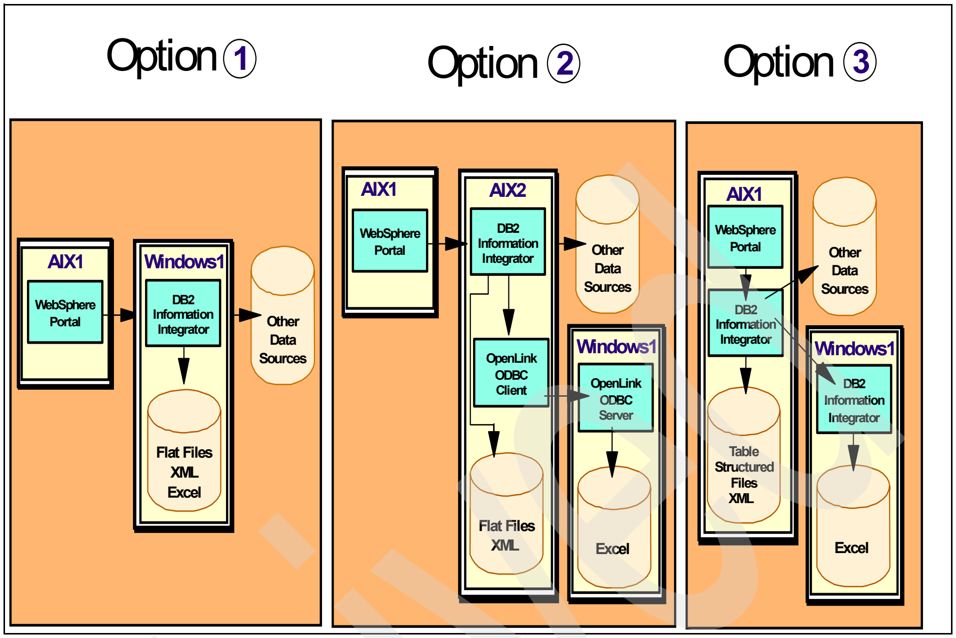


Figure 4-11 WebSphere Portal without clones in AIX and DB2 II scenarios

With WebSphere Portal on an AIX server without clones, DB2 II may be installed on the same AIX server as the WebSphere Portal, or a separate Windows or AIX server. Figure 4-11 describes the following possible options for integrating DB2 II in a WebSphere Portal AIX environment without clones:

- Option 1 shows the WebSphere Portal on the AIX1 server and DB2 II on a separate dedicated Windows1 server, which has access to all the flat files, XML files, and Microsoft Excel files required by the portlets.
- Option 2 shows the WebSphere Portal on the AIX1 server, flat files, and XML files, and DB2 II on the AIX2 server; and Microsoft Excel files on a separate Windows1 server.
- Option 3 shows the WebSphere Portal, flat files, XML files, and DB2 II on the AIX1 server, and another DB2 II and Microsoft Excel files on a separate dedicated Windows1 server.

The choice of a particular option depends upon many of the same considerations described in “Key technical considerations in a Windows environment” on page 72 *except for* the following differences:

- The scalability characteristics of the AIX platform is generally superior to that of Windows, and can support DB2 II installation on the same AIX server as the WebSphere Portal.

- The AIX platform does *not* support Microsoft Excel. If WebSphere Portal requires access to Microsoft Excel files, then it will require special DB2 II configurations such as described in Option 2 and Option 3 in Figure 4-11.
- The AIX platform can support multiple versions/fixpak levels of DB2 UDB, and therefore the DB2 II prerequisite of DB2 UDB 8.1.2 can coexist with other DB2 code levels in the same AIX server.

The following are some of the considerations in the selection of a particular option as described in Figure 4-11:

- Option 1 shows the WebSphere Portal on AIX1 Server and DB2 Information Integrator on Windows1 Server since all the non-relational files (flat files, XML files, and Microsoft Excel files) are only accessible by the Windows1 server.

  Note: Microsoft Excel has to be installed on the server where DB2 II is installed — Windows1 server in this case.

- Option 2 shows the WebSphere Portal on AIX1 server and DB2 II on another AIX2 server. If one of the data sources is Microsoft Excel on a Windows server, then the OpenLink ODBC Client on the AIX2 server can be used to connect to the Microsoft Excel files on the Windows1 server where OpenLink ODBC Server is installed.
- Option 3 shows the WebSphere Portal and DB2 II on the same AIX1 server that has flat files and XML files.

  This option also shows the AIX1 server DB2 II connected to another DB2 II on a Windows1 server which has Microsoft Excel files on it.

  Attention: In real world scenario, it might not be practical to have all the data sources like XML, Excel, and flat files on the same server as DB2 Information Integrator. The option 3 topology shows that DB2 II can actually query another DB2 II by creating nicknames over nicknames. The topology shows DB2 Information Integrator on the AIX1 server querying all the data sources in the disparate systems, as well as Microsoft Excel on the Windows1 server using another DB2 II.

4.2.4 WebSphere Portal clones in an AIX environment

The topologies discussed in this section involve two WebSphere Portal cloning configurations: a vertical cloned WebSphere Portal and a horizontal cloned WebSphere Portal environment. The recommended DB2 II scenarios for each environment are described here.

WebSphere Portal vertical cloned AIX environments

Figure 4-12 shows the WebSphere Portal vertical cloned environment that accesses geographically distributed relational and non-relational data sources. The Network Deployment Manager of WebSphere connects to the node agent that runs on the AIX1 server.

Attention: Vertical cloned systems provide greater scalability in large SMP servers with lots of real memory. For further details on vertical clones, refer to *IBM WebSphere 4.0 Advanced Edition Handbook,* SG24-6176.

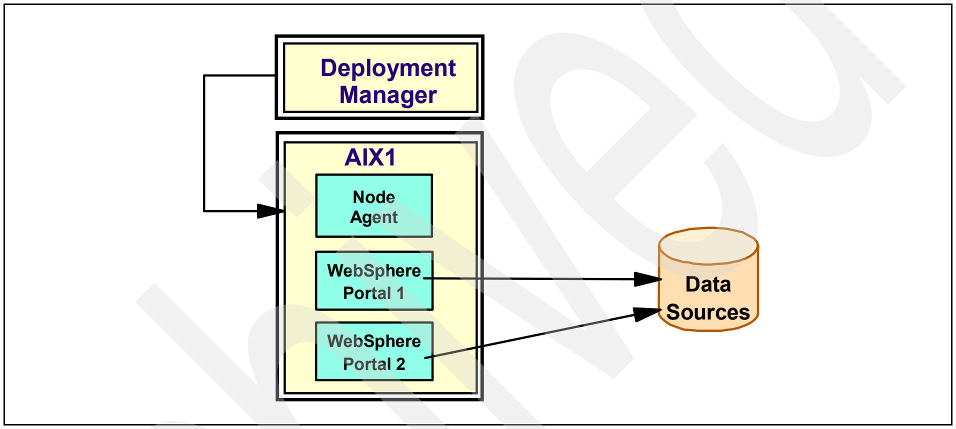


Figure 4-12 WebSphere Portal with vertical clones in AIX environment

Figure 4-13 shows the recommended configurations for integrating DB2 II into the environment shown in Figure 4-12.

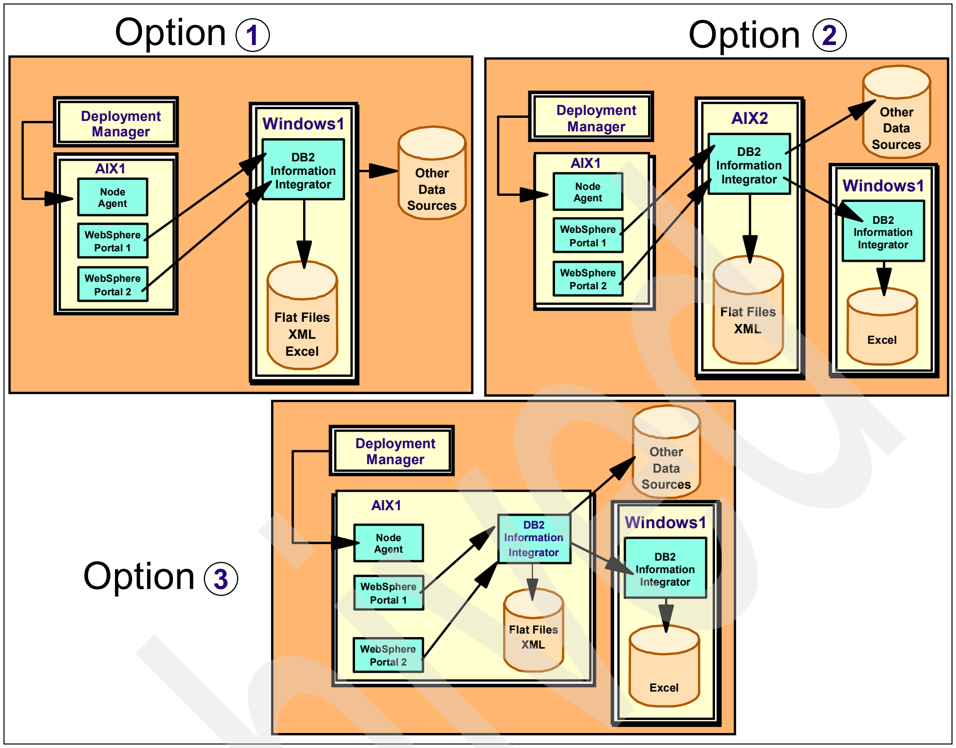


Figure 4-13 WebSphere Portal with vertical clones in AIX and DB2 II scenarios

The considerations for choosing a specific option depend upon the scalability, file access considerations, and DB2 UDB version/fixpak issues described in 4.2.3, “WebSphere Portal in an AIX environment” on page 78.

In the following list are some of the considerations in the selection of a particular option as described in Figure 4-13:

- Option 1 shows the WebSphere Portal on AIX1 Server and DB2 Information Integrator on Windows1 server since all the non-relational files (flat files, XML files and Microsoft Excel files) are only accessible by the Windows1 server.

> **Note:** Microsoft Excel has to be installed on the server where DB2 II is installed — Windows1 server in this case.

- Option 2 shows the WebSphere Portal on AIX1 server and DB2 II on another AIX2 server. If one of your data sources is Microsoft Excel on a Windows server, then the OpenLink ODBC Client on the AIX2 server can be used to connect to the Microsoft Excel files on the Windows1 server where the

OpenLink ODBC server is installed. This option shows the AIX2 server DB2 II connected to another DB2 II on a Windows1 server, which has Microsoft Excel files on it.

- Option 3 shows the WebSphere Portal and DB2 II on the same AIX1 server that has flat files and XML files. This option also shows the AIX2 server DB2 II connected to another DB2 II on a Windows1 server, which has Microsoft Excel files on it.

WebSphere Portal horizontal cloned AIX environments

Figure 4-14 shows the WebSphere Portal horizontal cloned environment that accesses geographically distributed relational and non-relational data sources. The Network Deployment Manager of WebSphere connects to the node agents that run on the AIX1 server and AIX2 server.

> **Attention:** Horizontal cloned systems not only provide greater scalability, but also address high availability and continuous operation requirements. For further details on horizontal clones, refer to *IBM WebSphere 4.0 Advanced Edition Handbook,* SG24-6176.

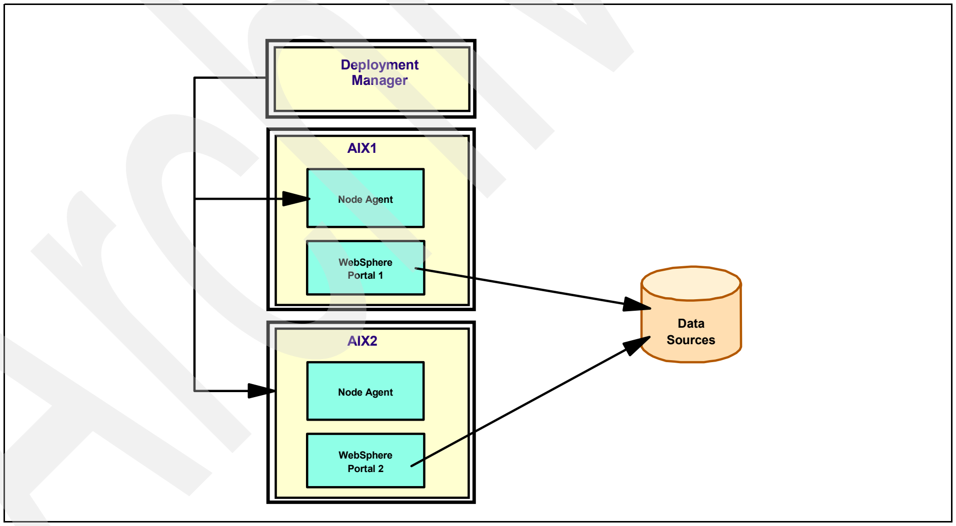


Figure 4-14 WebSphere Portal with horizontal clones in an AIX environment

Figure 4-15 shows the recommended configurations for integrating DB2 II into the environment shown in Figure 4-14.

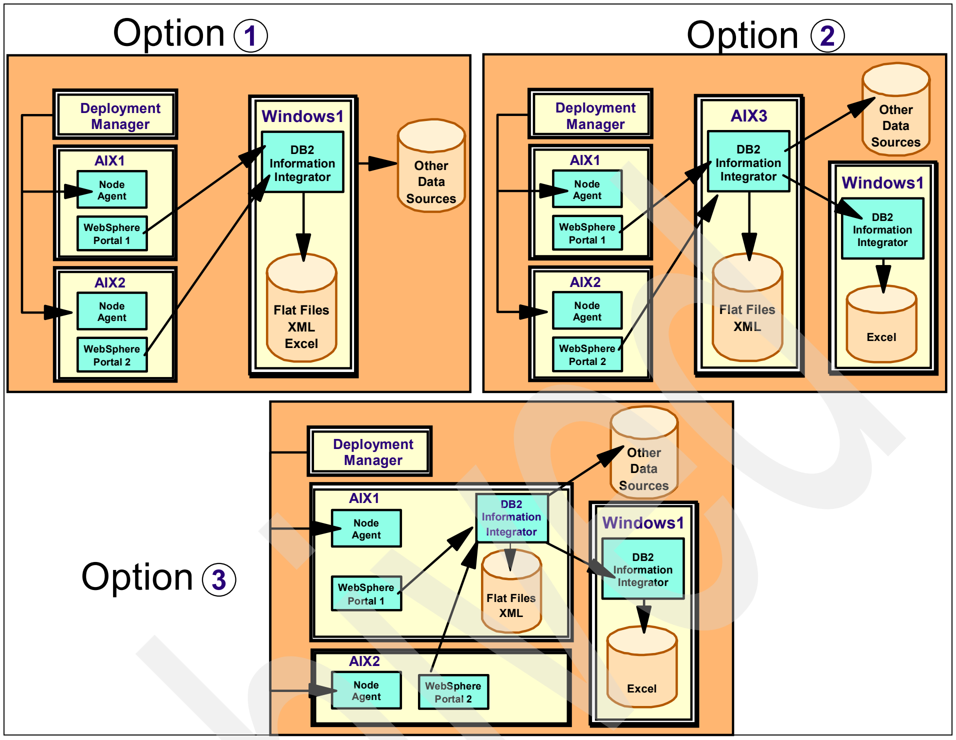


Figure 4-15 WebSphere Portal with horizontal clones in AIX and DB2 II scenarios

Figure 4-15 describes the following possible options for integrating DB2 II in a WebSphere Portal AIX environment with horizontal clones:

- Option 1 shows the WebSphere Portal on the AIX1 and AIX2 servers, and DB2 II on a separate dedicated Windows1 server, which has access to all the flat files, XML files, and Microsoft Excel files required by the portlets.
- Option 2 shows the WebSphere Portal on the AIX1 and AIX2 servers, with flat files and XML files and DB2 II on the AIX2 server, and DB2 II and Microsoft Excel files on a separate dedicated Windows1 server.
- Option 3 shows the WebSphere Portal on the AIX1 and AIX2 servers, with flat files and XML files and DB2 II on the AIX2 server, and DB2 II and Microsoft Excel files on a separate dedicated Windows1 server.

The selection of a particular option depends upon many of the same considerations described in 4.2.3, “WebSphere Portal in an AIX environment” on page 78.

The choice of a particular option shown in Figure 4-15 depends upon the location of the non-relational data sources such as flat files, XML files, and Microsoft

Excel files as discussed in “WebSphere Portal vertical cloned AIX environments” on page 81.

> **Attention:** For organizations that choose a horizontally cloned WebSphere Portal configuration for its high availability characteristics, the DB2 II server configurations described in Figure 4-15 constitute a single point of failure. You need to implement a DB2 II high availability cluster environment to overcome this limitation.

4.2.5 Summary

The recommended options are only meant to provide guidelines for possible DB2 II configurations in typical WebSphere Portal environments. You should tailor the options to your particular environment for maximum effectiveness.

4.3 Tips for successful deployment

We recommend the following steps to ensure a successful deployment of WebSphere Application Server, WebSphere Portal, WebSphere Studio, and DB2 Information Integrator technologies in your environment:

- Ensure that each individual sub component in your architecture is working properly before trying to integrate all these software technologies together. For example, ensure that your federated DBMS environment is properly configured to all necessary back-end data sources before trying to integrate it with WebSphere Studio or WebSphere Application Server. This will aid debugging of any installation-related problems.
- Consider using the IBM DDL transparency feature if you want to create new data objects in OEM data sources to support your work. This can save time and simplify your work.
- Create local tables that match the schema of production remote tables during the testing phase of new EJBs. This will minimize the impact of your development effort on production DBMSs, and potentially facilitate better performance.
- Familiarize yourself with problem diagnosis tools available with the products you are using. In particular, when debugging, make an early attempt to isolate the source of the failure. For example, if a database access activity is failing, determine the exact SQL statement being issued and run this statement from a DBMS command window connected to the federated database to help pinpoint the cause.

- Understand performance monitoring and tuning facilities available with the products you are using. If database-related activities appear to be slow, consider enlisting the support of a database administrator. Both DB2 and DB2 II offer a number of tuning facilities, as do other DBMS products. Activities such as adding appropriate indexes or updating catalog statistics (including information kept by the federated server for global optimization purposes) can have considerable impact on performance.

WebSphere Portal & DB2 Information Integrator scenarios

In this chapter we describe a hypothetical portal scenario involving a financial services company named Cotton-Wood Financial Services (CFS) that leverages DB2 II functionality.

The topics covered are:

- Insurance agent profile
- The CFS portal
- CFS portal custom portlets
- DB2 II versus no DB2 II: A sample portlet
- Configuring the JDBC Business Object Builder

5.1 Introduction

The hypothetical portal scenario involves an insurance agent for a fictitious insurance company named Cotton-Wood Financial Services (CFS). This portal provides a single point of access for an insurance agent of CFS. It aggregates data, content, collaboration, and proprietary systems used in the agent's daily activities.

This objective here is to present a realistic portal scenario using WebSphere Portal, and to demonstrate cases where DB2 II can be leveraged for maximum benefit.

The following subsections discuss the interactions of an insurance agent named John Watson with the CFS portal, the layout of the CFS portal for John, and a detailed review of the portlets exploiting DB2 II. We also review the code of one of the custom portlets with and without the use of DB2 II.

We also provide an example of building a portlet to access a DB2 II nickname using the JDBC Business Object builder off-the-shelf portlet.

5.2 Insurance agent profile

John Watson is an insurance agent employed at CFS with an employee portal that provides access to all necessary systems and information in a single place. The portal is also used by John to collaborate with his colleagues and share documents, as well as pursue personal interests.

John's day-to-day activities can be divided into three categories as follows:

1. **Customer support activities** involve John selling, activating, and processing insurance policies and claims for his clients. The portal provides insurance tools to make intelligent decisions on policy sales, and expediting requests from customers for their insurance needs. Insurance activities include the following items:
   – Submit or track a claim from a customer
   – Evaluate demographic information to determine policy discounts
   – Apply for a new policy
   – View competitive market information
   – Run reports on claims and policies
2. **Employee administrative activities** include John performing administrative tasks by virtue of being an employee of CFS. The portal provides support for the following employee administrative activities:
   – Sending and receiving e-mail

   - Accessing HR forms and documents
   - Viewing corporate news
   - Receiving alerts for time-sensitive information
   - Collaboration with other CFS employees

3. **Personal activities** that include reviewing stock quotes, weather, sports, and breaking news. The portal provides support for a predefined list of portlets for John's personal interests that includes access to stock quotes, national or vertical market news, weather, and Web search portlets.

The combination of portlets provided by the CFS portal creates a complete user experience for John. The CFS portal lets him respond to customer demand, and significantly improves productivity through business intelligence and collaboration tools.

5.3 The CFS portal

The CFS portal supports the three categories of activities mentioned. This requires the portal to provide access to a diverse range of data sources and presents several challenges. In addition to integrating with e-mail and collaboration backend systems (like Lotus Notes), the portlets that make up the CFS portal must access a variety of disparate systems and aggregate federated data sources.

The CFS environment shown in Figure 5-1 reflects a distributed heterogeneous environment with a variety of disparate data sources.

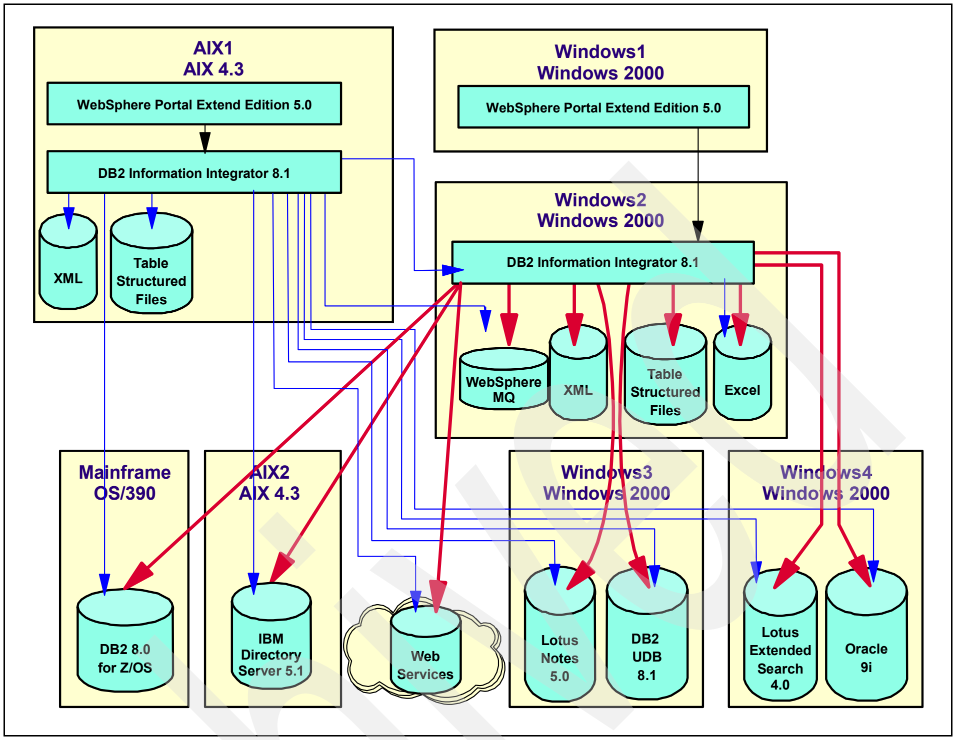


Figure 5-1 CFS environment configuration

WebSphere Application Server Advanced Edition 5.0.2 was used in this environment.

The CFS portal aggregates information from a number of data sources distributed across multiple AIX, Windows and OS/390 servers as shown in Figure 5-1:

- RDBMS (DB2 UDB on Windows, DB2 for z/OS, and Oracle 9i on Windows)
- Web Services
- XML on AIX and Windows
- Table Structured (flat) flat files on AIX and Windows
- Microsoft Excel Spreadsheets on Windows
- Lotus Extended Search (LDAP) on Windows
- ODBC data sources on Windows
- Lotus Notes on Windows
- IBM Directory Server on AIX
- WebSphere MQ on Windows

Figure 5-2 shows the first page of the portal when John Watson logs in.

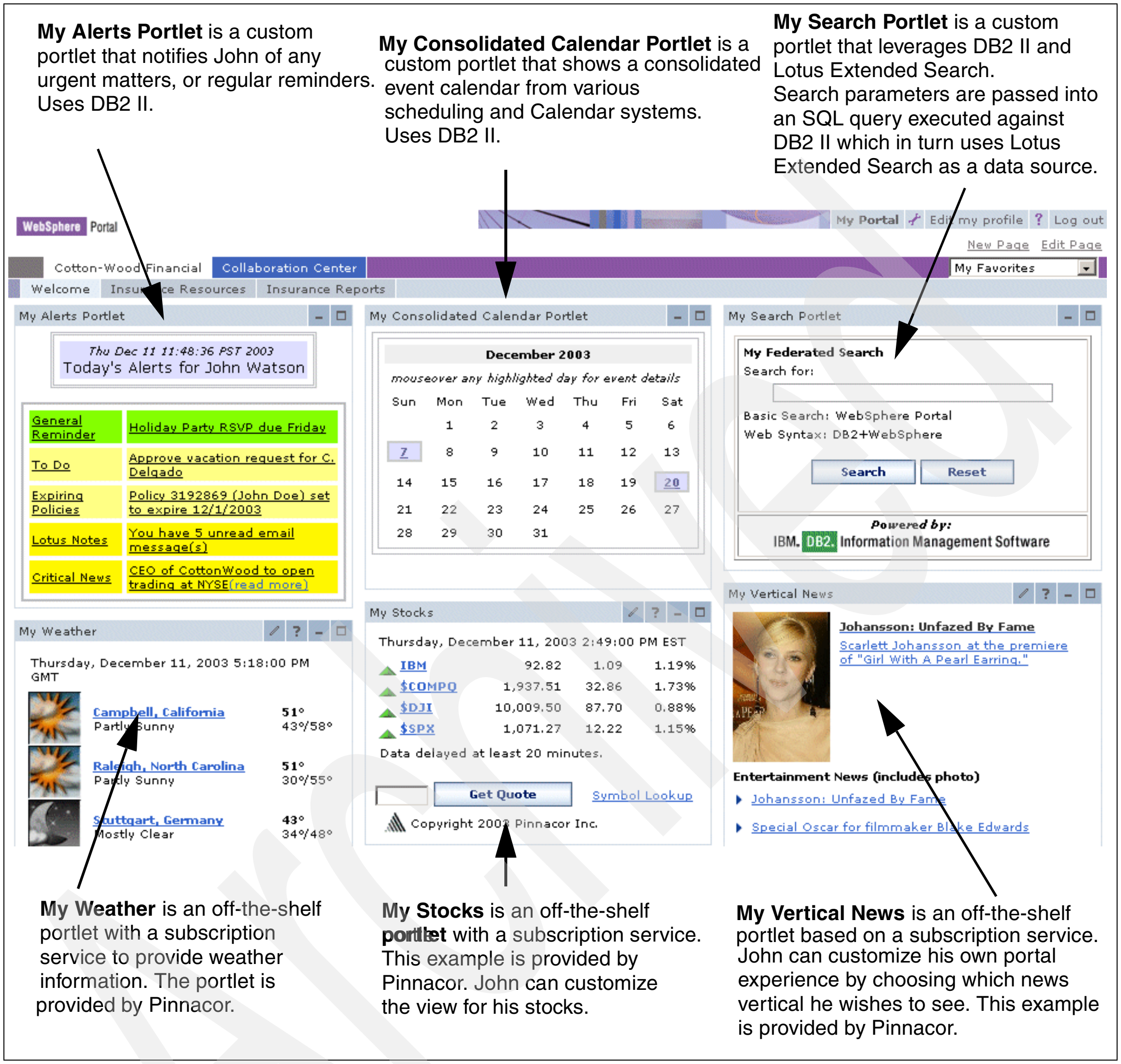


Figure 5-2 CFS portal Welcome page: John sees this page on logging in

Figure 5-2 shows two tabs: Cotton-Wood Financial and Collaboration Center as follows:

- The Cotton-Wood Financial tab is a group of three pages: Welcome, Insurance Resources, and Insurance Reports. Figure 5-2 displays the Welcome page. Figure 5-4 on page 95, and Figure 5-5 on page 97 display the Insurance Resources and Insurance Reports pages respectively.

- The Collaboration Center tab shown in Figure 5-6 on page 98 has only one page.

Each page has one or more portlets, for example, the Welcome page in Figure 5-2 has six portlets: My Alerts Portlet, My Consolidated Calendar Portlet, My Search Portlet, My Weather, My Stocks, and My Vertical News.

The CFS portal has other links such as Edit my profile, New Page, and Edit Page. These are part of the customization services provided by WebSphere Portal. Figure 5-3 shows how John can change some basic profile information or his password by selecting **Edit My Profile** in the WebSphere portal theme.

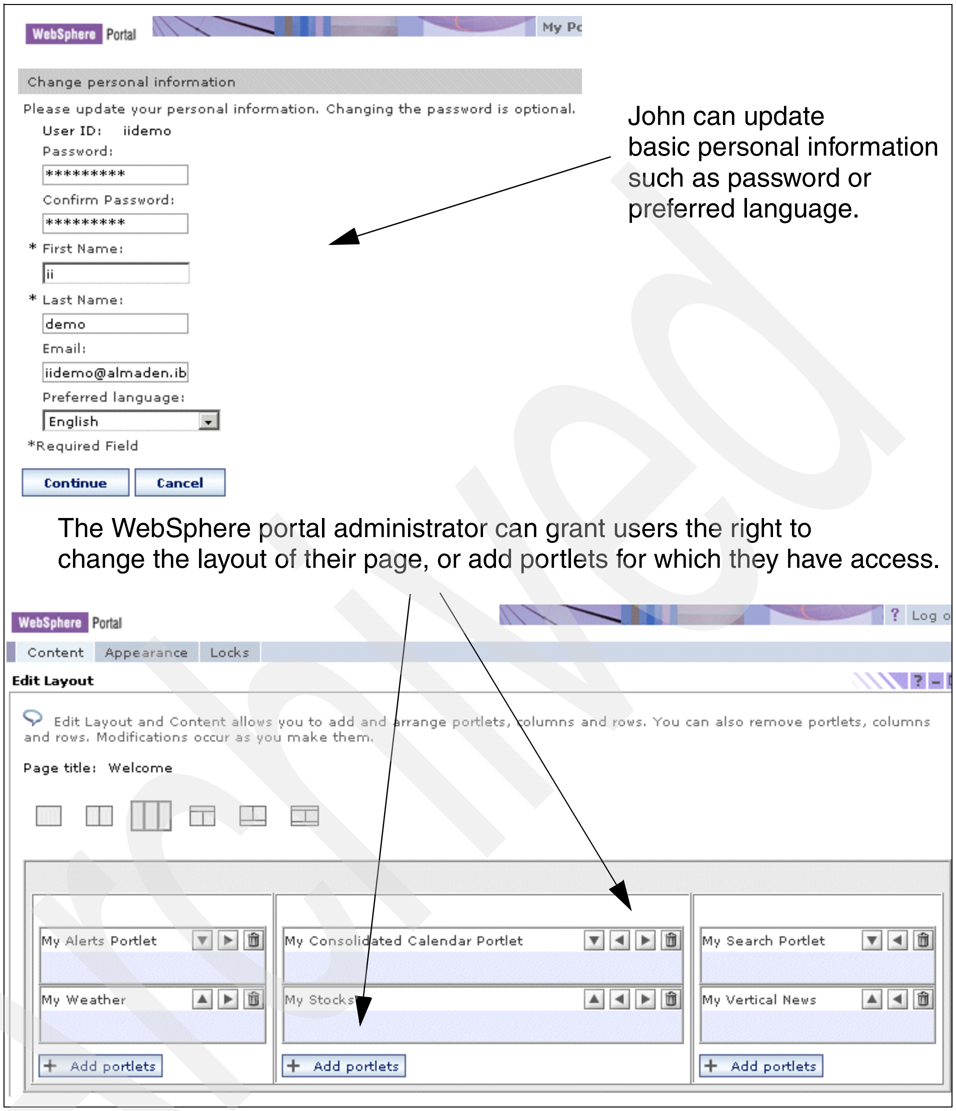


Figure 5-3 Customization and personal information updates in the CFS portal

Important: Portal administrators and developers must decide on pages and groups of pages as part of the layout. The page layout defines the number of content areas within the page and the portlets displayed within each content area. In many cases, the portal administrator defines the page layout. The portal administrator can permit specified users or user groups to change the page layout to reflect individual preferences as well as add available portlets.

5.3.1 Cotton-Wood Financial tab

Upon logging into the portal, John is presented with the Welcome page of the CFS portal. From this page, John can navigate to portal pages such as Insurance Resources and Insurance Reports, which provide tools to perform his daily work that includes processing and handling new claims in addition to selling new policies.

> **Note:** To navigate to the various pages discussed in this chapter, the portal user simply clicks on the tabs or other pages shown in the portal theme.

Welcome page

This page is a combination of portlets that serve the following purpose:

- Alerts John to items that potentially require his immediate attention
- Provides a “bird’s eye” overview of his planned activities through a calendar that consolidates schedules and events from a multitude of sources
- Displays information of personal interest such as stock quotes and weather, which John can configure as appropriate
- Allows John perform federated searches across CFS and the Internet

Table 5-1 lists the portlets on this page, their data sources, and whether they use DB2 II to integrate federated data in our CFS portal.

Table 5-1 Portlets, data sources, and DB2 II usage: Welcome page

Portlet name	Data sources	Uses DB2 II
My Alerts Portlet This is a custom portlet	Lotus Notes, DB2 UDB	Yes
My Consolidated Calendar Portlet This is a custom portlet	DB2 UDB, Oracle 9i	Yes
My Search Portlet This is a custom portlet	Lotus Extended Search	Yes
My Weather This is an off-the-shelf portlet	Subscription (Pinnacor)	No
My Stocks This is an off-the-shelf portlet	Subscription (Pinnacor)	No
My Vertical News This is an off-the-shelf portlet	Subscription (Pinnacor)	No

More details about the custom portlets are provided in 5.4, “CFS portal custom portlets” on page 99.

Insurance Resources page

Figure 5-4 shows the Insurance Resources page.

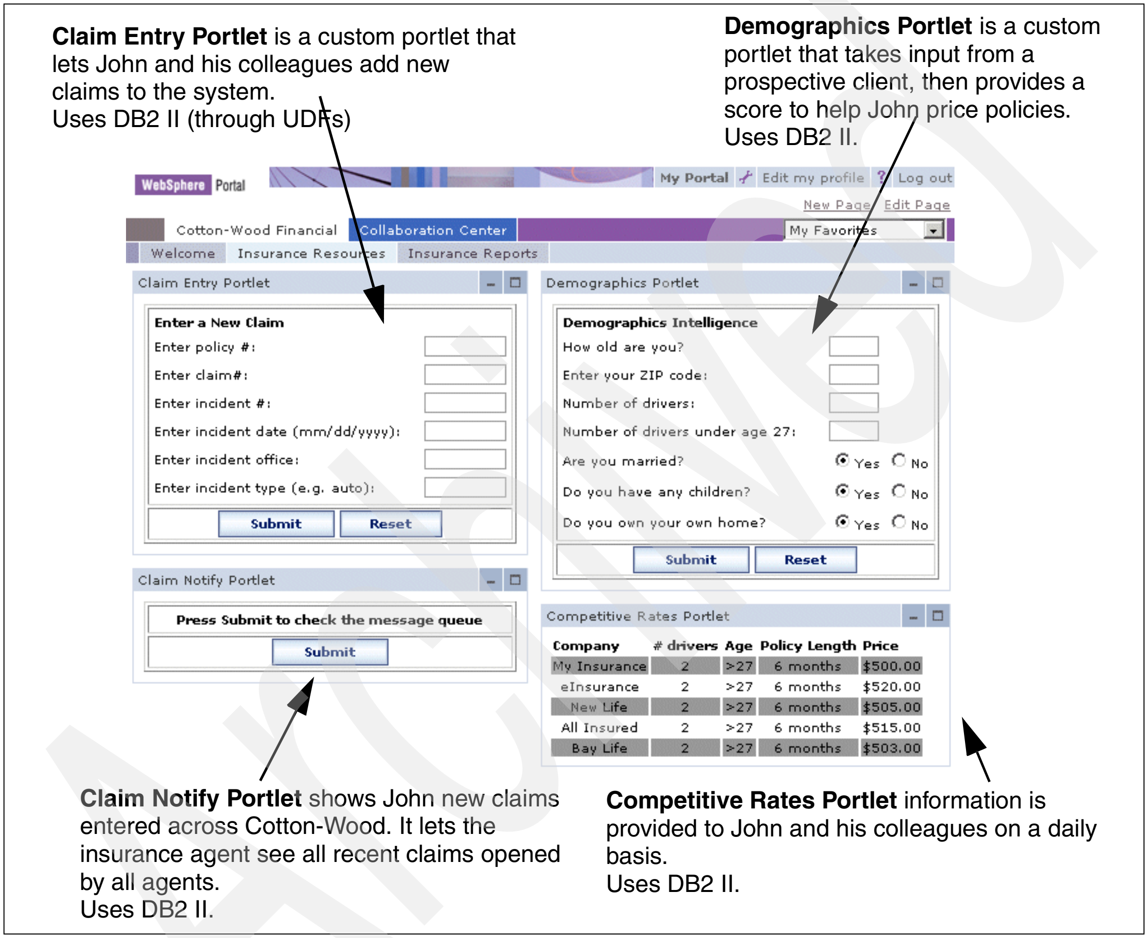


Figure 5-4 Insurance Resources page

This page is a combination of portlets, which provide John the tools to do the following:

- Enter new claims from clients
- Get detailed information on claims
- Get demographic data based on a prospective client
- View competitive information on CFS’ competitor rates

Table 5-2 lists the portlets on this page, their data sources, and whether they use DB2 II to integrate federated data in our CFS portal.

Table 5-2 Portlets, data sources, and DB2 II usage: Insurance Resources page

Portlet name	Data sources	Uses DB2 II
Claim Entry Portlet This is a custom portlet	WebSphere MQ	Yes (through UDFs)
Demographics Portlet This is a custom portlet	Web Services, DB2 on z/OS	Yes
Claim Notify Portlet This is a custom portlet	XML, Lotus Notes, DB2 UDB, WebSphere MQ	Yes
Competitive Rates Portlet This is a custom portlet	Flat Files	Yes

More details of the custom portlets is provided in 5.4, "CFS portal custom portlets" on page 99.

Insurance Reports page

Figure 5-5 shows the insurance reports page of the CFS portal.

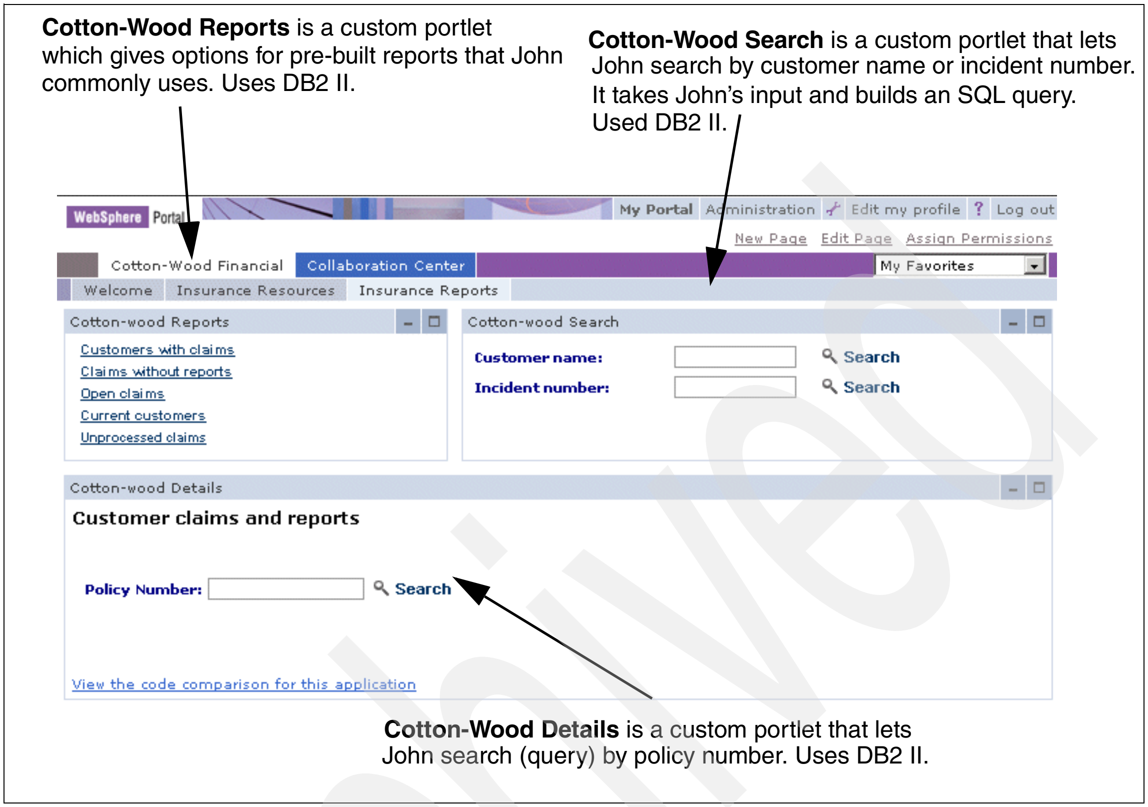


Figure 5-5 Insurance Reports page

This page is a combination of portlets that provide John the tools to do the following:

- Run a variety of reports on customer claims, and open claims based on a number of criteria
- Perform searches for customers or policies
- Execute searches based on policy number

Table 5-3 lists the portlets, data sources, and whether they use DB2 II to integrate federated data.

Table 5-3 Portlets, data sources, and DB2 II usage: Insurance Reports page

Portlet Name	Data Sources	**Uses DB2 II**
Cotton-Wood Reports Portlet This is a custom portlet	DB2 UDB, Lotus Notes, XML	Yes

Portlet Name	Data Sources	**Uses DB2 II**
Cotton-Wood Search Portlet This is a custom portlet	DB2 UDB	Yes
Cotton-Wood Details Portlet This is a custom portlet	DB2 UDB, Lotus Notes, XML	Yes

> **Note:** While the portlets shown here are custom, it is also possible to use off-the-shelf portlets such as My Query Reports, and to add the appropriate SQL that leverages DB2 II for a results set.

More details of the custom portlets is provided in 5.4, “CFS portal custom portlets” on page 99.

5.3.2 Collaboration Center tab

In addition to the customer support and personal portlets described, John can interact with his colleagues, partners, and customers through the (Lotus) Collaboration Center. Figure 5-6 shows the collaborative portlets built into the CFS portal. All the portlets used here are off-the-shelf portlets.

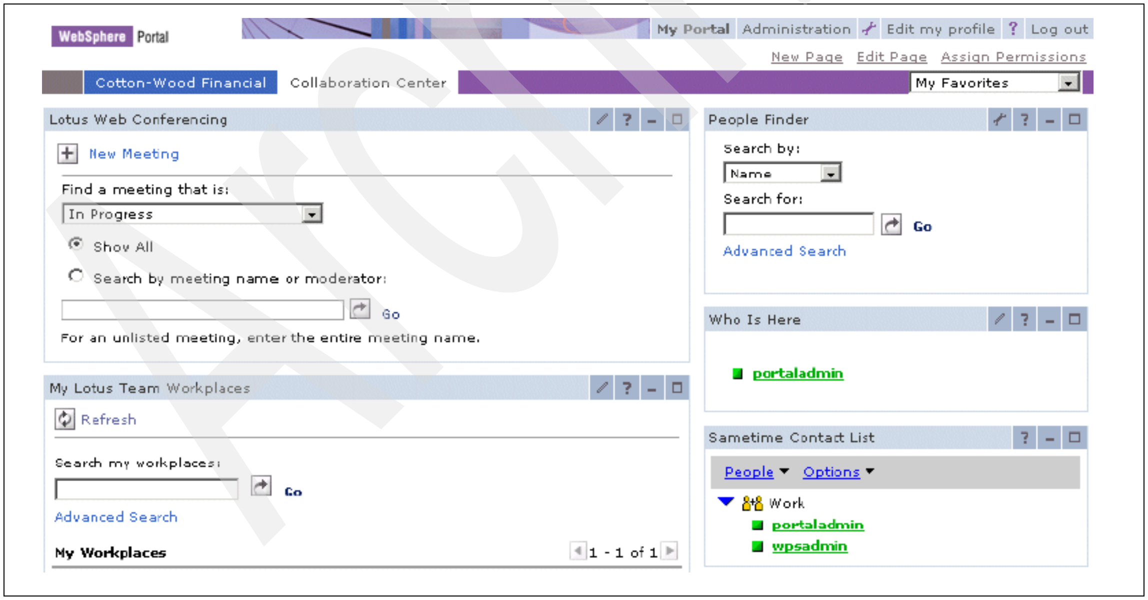


Figure 5-6 Collaboration Center tab

Note: For more information on installing and configuring the (Lotus) Collaboration Center for WebSphere Portal, refer to *IBM WebSphere Portal for Multiplatforms V5,* SG24-6098-00.

5.4 CFS portal custom portlets

All the following custom portlets in the CFS portal leverage DB2 II:

- My Alerts Portlet
- My Consolidated Calender Portlet
- My Search Portlet
- Claim Entry Portlet
- Demographics Portlet
- Claim Notify Portlet
- Competitive Rates Portlet
- Cotton-Wood Reports Portlet
- Cotton-Wood Search Portlet
- Cotton-Wood Details Portlet

For each of these custom portlets, the following subsections provide a more detailed description of the portlet; the data sources; UDFs and nicknames used; a snippet of the portlet code showing the SQL statement leveraging DB2 II nicknames and UDFs; and the effort involved if DB2 II nicknames are not available.

Attention: The custom portlet code used in the CFS portal can be downloaded from:
`ftp://www.redbooks.ibm.com/redbooks/SG246433`

The following nicknames are used in the various portlets: Appendix B, “Configuring data sources in DB2 Information Integrator” on page 225 has examples of the configuration of some of these data sources:

- MAIL_NN on e-mail data source
- POLICY_APP on Oracle 9i data source
- NEWS_NN on DB2 UDB data source
- EVENTS_N1 on DB2 UDB data source
- CALENDAR_EVENTS on Oracle 9i data source
- DEMO_NN on Lotus Extended Search data source
- DEMO_FILE_NN on Lotus Extended Search data source
- DEMOGRAPHICS on DB2 UDB for z/OS data source
- REPORTS_XML on XML data source
- CUSTOMER_DB2 on DB2 UDB data source

- COMPETITIVERATES2 on ASCII file data source
- CLAIMS_DOMINO on Lotus Notes data source

It may be appropriate for an organization to have special naming conventions for nicknames in order to distinguish them tables or views. We chose not to adopt a naming convention for the nicknames used in our federated database.

> **Important:** Without a federated database capability such as DB2 II, the portlet developer must use the native API for each data source, manage connections to the data source, and handle user ID and password mapping as well as data mappings. In addition, to deliver optimal performance for queries that involve joins, aggregates, and sorts, the portlet developer requires advanced programming skills to implement sophisticated optimization algorithms that take into account performance factors such as network bandwidth, differing CPU speeds, and records scanned and returned.
>
> DB2 II replaces the diversity of APIs required to access a wide range of disparate data sources with a single SQL API, and provides a transparent sophisticated optimization mechanism to ensure superior performance when joining heterogeneous data sources. By leveraging DB2 II, the application developer's task is significantly simplified — one only needs to focus on using DB2's the SQL API and associated capabilities such as its mining, Web Services, and WebSphere MQ UDFs. In addition to the simplicity of only having to invoke the simplified DB2 SQL API, the portlet developer also delegates the responsibility of developing sophisticated optimization algorithms for optimal performance to DB2 II.
>
> The simple SQL API advantages of DB2 II to access disparate data sources on distributed heterogeneous platforms are showcased in the custom portlets described in this section.
>
> Refer to "DB2 II versus no DB2 II: A sample portlet" on page 133 for a comparison of portlet code with and without DB2 II.

5.4.1 Custom portlets development environment

The custom portlets were built using WebSphere Studio Application Developer (WSAD) 5.0.1 along with the Portal Toolkit 5.0.

> **Note:** Developers can alternatively use Eclipse or other Java IDEs, however, this will require additional steps to ensure the generation and deployment of correct JAR files and tag libraries necessary for portlet development.

WSAD also assists the developer by helping build the necessary deployment descriptors and packaging for WebSphere Portal.

To render the portlets, we developed presentation within JSPs, and the portlet's rendering methods (for example, doView()) include the JSP. This provides separation between the portlet developer and the JSP designer, and is also easier to manage. For the CFS portal custom portlets, we built the JSPs and portlets using WSAD.

From a best practices point of view, the portlet should be insulated from the specific implementation details of accessing data. This requires discovering the WebSphere data source (that references DB2 Information Integrator) through JNDI, and building a JavaBean whose attributes contain the result set of a query performed against one or more (DB2 Information Integrator federated) data sources. These value objects can then be passed to presentation JSP as part of the PortletRequest object.

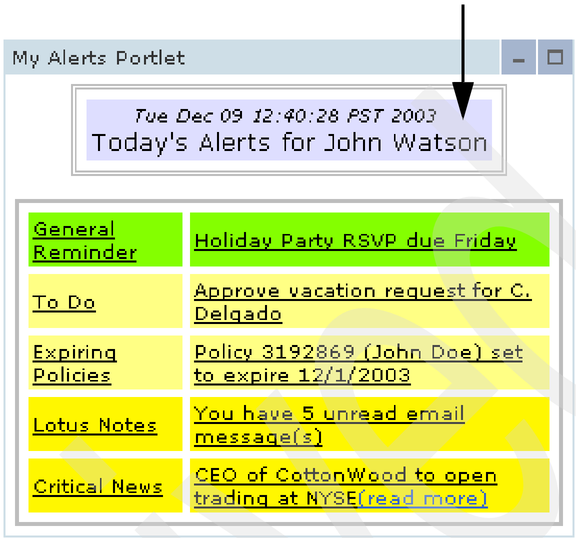
Since the DB2 Information Integrator federated data sources appear as a single DB2 database, we developed queries using the DB2 Control Center and DB2's CLI processor. After we debugged the queries this way, we added them to the data access objects (DAO) constructed by the portlet. With DB2 Information Integrator, the portlet developer can use a single DAO since only a single interface (JDBC) is necessary. This reduces development complexity and achieves best practices per J2EE Design Patterns.

For portlet development best practices, refer to the paper "Portlet Development Best Practices and Coding Guidelines" March 17th 2003 by Marshall Lamb available at:
`ftp://207.25.253.53/1/wsdd/pdf/PortletCodingGuidelines.pdf`

5.4.2 My Alerts Portlet

A common requirement for portals is to alert the user to events that may require immediate action or are time sensitive. The My Alerts Portlet provides this function by alerting John to critical news, or items that may require prompt attention. Figure 5-7 shows the My Alerts Portlet.

Based on the username and password, the portlet retrieves alerts for the user. The user's first and last name can be mapped to LDAP attributes.



Events in the My Alerts Portlet are color coded to help John prioritize his day. John might have alerts from several disparate systems.

Figure 5-7 My Alerts Portlet

> **Note:** In our implementation of this portlet, John needs to refresh the portal to view updated information and events. In the real world, the My Alerts Portlet needs to refresh itself while John is logged in to the portal in order for John to see any new alerts as soon as they occur.

Table 5-4 lists the data sources and the nicknames used in this portlet.

Table 5-4 Data sources used in the My Alerts Portlet

Data sources	Type	DB2 II nickname
E-mail	Lotus Notes	`MAIL_NN`
Expiring policies	Oracle 9i	`POLICY_APP`
Critical news	DB2 UDB	`NEWS_NN`

The e-mail, expiring policies, and critical news data sources are accessed using DB2 II nicknames to identify alert events to be displayed in the portlet.

Example 5-1 shows a snippet of the My Alerts Portlet code that builds the SQL query performing the join using the DB2 II nicknames.

Example 5-1 doView() method in the My Alerts Portlet

```
public void doView(PortletRequest request, PortletResponse response)
   throws PortletException, IOException {

   User u = portReq.getUser();
   String uid = u.getUserID()

   StringBuffer sb = new StringBuffer("select EVENT_DATE, DESCRIPTION, ");
   sb.append("PRIORITY from POLICY_APP, MAIN_NN, NEWS_NN ");
   sb.append("where POLICY_APP.ID = \"" + uid + "\" and ");
   sb.append("MAIN_NN.ID = \"" + uid + "\" and ");
   sb.append("NEWS_NN.ID = \"" + uid + "\" ");
   sb.append("order by PRIORITY");
   ......
   ResultSet rs = st.executeQuery(sb.toString());
   ......
}
```

This portlet captures the user ID that comes as part of the PortletRequest. The user object supplies a method that returns a user ID, which is then used in the query.

5.4.3 Consolidated Calendar Portlet

This portlet gives John Watson a consolidated view of all scheduling and events of interest. Prior to the portal, John had to check several data sources and manually consolidate his calendar. Figure 5-8 shows the Consolidated Calendar Portlet.

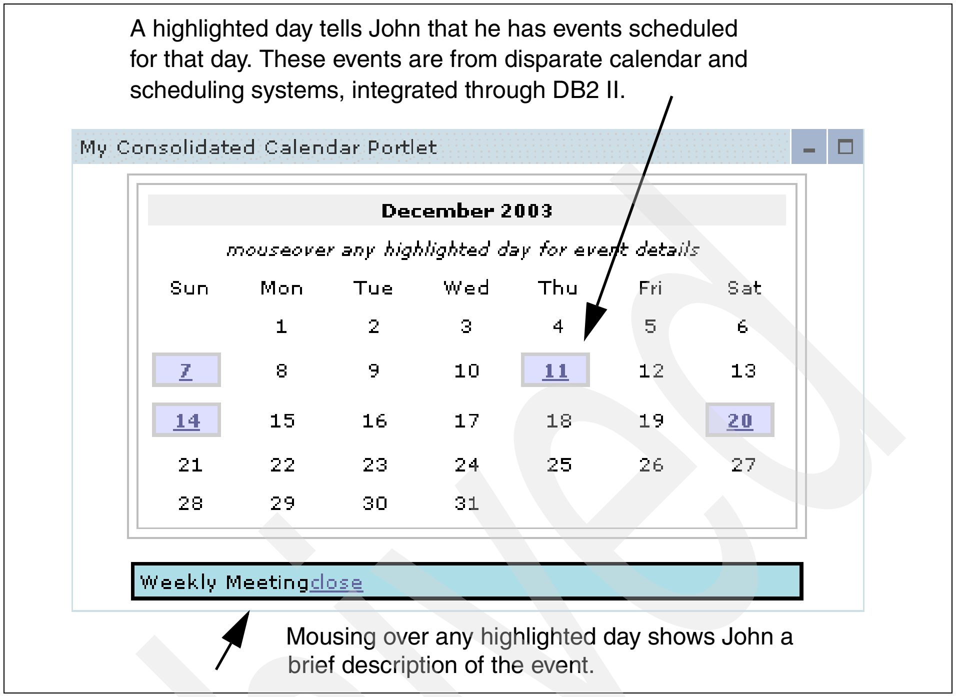


Figure 5-8 My Consolidated Calendar Portlet

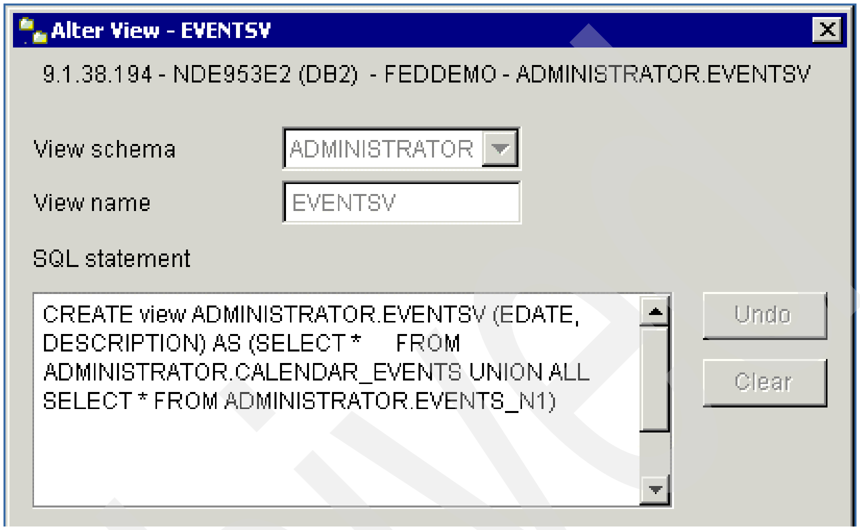
> **Note:** We assume that each of the data sources has a front-end system that allows administrators to add events. For example, a Web application built by Domino Designer® lets a user add new events to the Lotus Notes database. In this portlet, we consolidate and display the event information from the various data sources using DB2 II nicknames.

Table 5-5 details the data sources that are integrated into the portlet using DB2 II.

Table 5-5 Data sources used in the Consolidated Calendar Portlet

Data source	Type	DB2 II nicknames
Proprietary Scheduling System	DB2 UDB database	`EVENTS_N1`
General Events	Oracle 9i database	`CALENDAR_EVENTS`

Nicknames are created for each of the data sources, and then a view **EVENTSV** is built on the nicknames to simplify the SQL used by the Consolidated Calendar Portlet. Figure 5-9 shows the creation of this view in the DB2 Control Center.



This view is created as a UNION ALL between the nicknames. The portlet code can now simply query this view.

Figure 5-9 Create VIEW statement as a UNION ALL of nicknames

Example 5-2 is a snippet of the portlet code with the simple SQL statement to retrieve data from the view.

Example 5-2 doView() method in the Consolidated Calendar Portlet

```
public void doView(PortletRequest request, PortletResponse response)
   throws PortletException, IOException {
      PortletSession s = request.getPortletSession();
      String method = (String)s.getAttribute("MainMethod");
      StringBuffer sb = new StringBuffer("select * from EVENTSV");
      ResultSet rs = st.executeQuery(sb.toString());
}
```

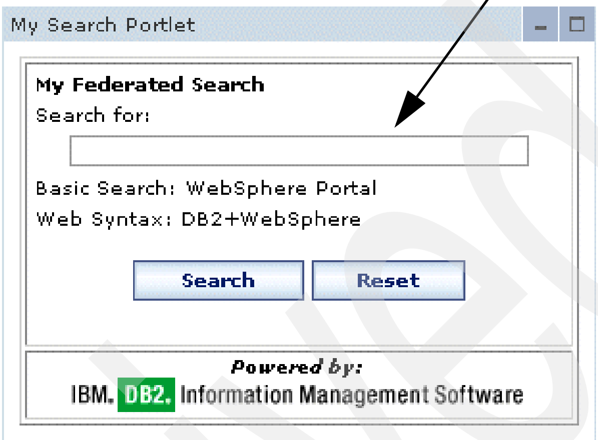
Attention: The portlet developer needs to consider and build the following:

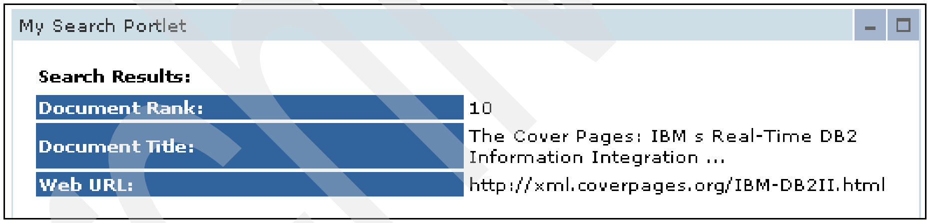
- Create and access JDBC or ODBC data sources to query the disparate data.
- Manage separate connections to each data source.
- Build separate queries or APIs into each system.
- Logically sort or order the event data. Using DB2 II, the developer can build this logic into the single SQL query.

5.4.4 My Search Portlet

A requirement for the CFS portal is the ability to perform federated searches against intranet, extranet, and Internet data sources. The My Search Portlet used Lotus Extended Search v4.0 (LES) as a DB2 II data source, and built SQL queries that executed against this data source. Figure 5-10 shows the My Search Portlet.

John can enter a search string using basic syntax, or Web syntax. The search string entered is built into a query by the portlet. The portlet executes the query against a DB2 II nickname that references Lotus Extended Search as the data source.





A result set, including rank, title, and Web URL are displayed. DB2 II and LES can be configured to return other details about the search.

Figure 5-10 My Search Portlet

Table 5-6 details the data sources used in the My Search Portlet. Note that Lotus Extended Search serves as a data source for DB2 II in our scenario.

Table 5-6 Data sources used in the My Search Portlet

Data source	Type	DB2 II nickname
Lotus Extended Search	Google	`DEMO_NN`
Lotus Extended Search	File system	`DEMO_FILE_NN`

LES administrators can configure LES to expand or limit federated searches by search engine or other criteria.

Figure 5-11 shows the LES administration applet. After installing LES, users can launch:
`https://<LES-server>:9443/lotuskms/ESAdmin`. From this screen, the “demo” LES application search interface can be launched by clicking **Demo JKM** as shown in Figure 5-11.

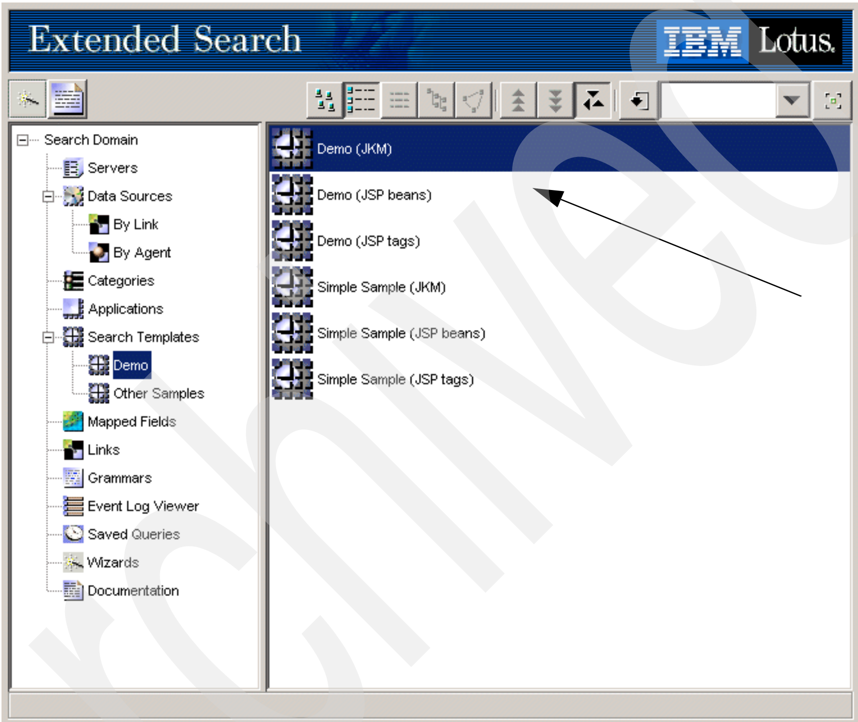


Figure 5-11 Lotus Extended Search administration applet

Figure 5-12 shows the search interface into the “demo” LES application.

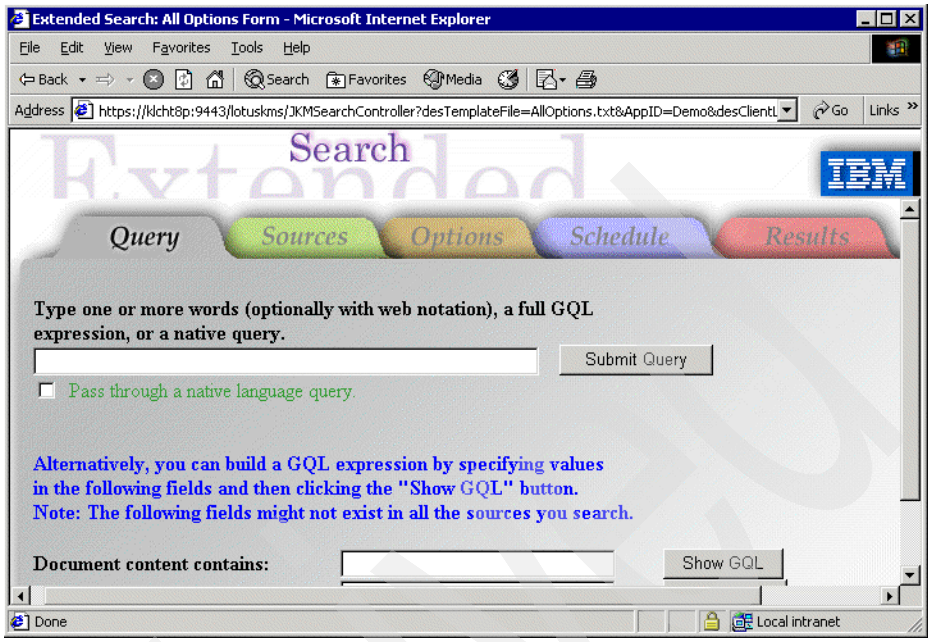


Figure 5-12 Web interface into an LES search application

Click the **Sources** tab to configure the search sources (Google on the Web, and ES Documentation in file systems) for this application as shown in Figure 5-13.

> **Attention:** For more information on Lotus Extended Search v4.0, refer to:
>
> `http://www-12.lotus.com/ldd/doc/uafiles.nsf/docs/les40/$File/readme_es40.html`

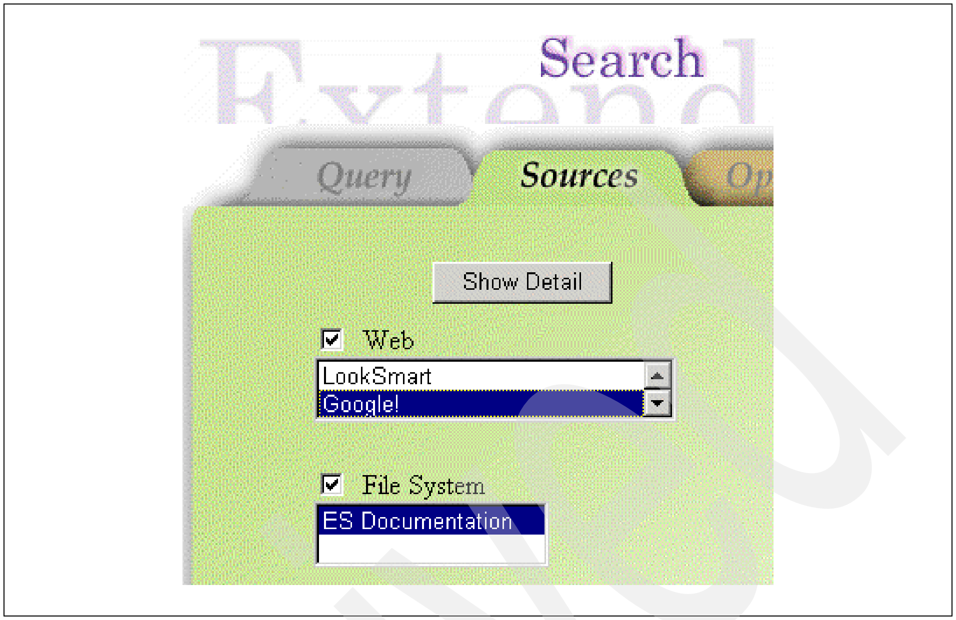


Figure 5-13 LES Administration interface : Data sources selection

The portlet developer can construct a simple SQL query against the nickname (**DEMO_NN**) referencing the LES data source to perform this search, rather than having to build a native API into the LES. Example 5-3 shows the simple SQL query used in the My Search portlet to perform federated searches. **ES_SEARCH** is a UDF that enables more sophisticated search criteria to be applied in an SQL query.

Example 5-3 doView() method in the My Search Portlet

```
if (method.equals("showSearchResults")) {
   String searchParm = (String)s.getAttribute("search_parm");
   StringBuffer sb = new StringBuffer();
   sb.append("select * from DEMO_NN where ");
   sb.append("esserver.es_search(DOC_RANK,'(TOKEN:EXACT \"");
   sb.append(" + searchParm + "\")')=1");
   ....
}
```

The SQL for creating the **DEMO_NN** nickname is shown in Example 5-4.

Example 5-4 Creating the DEMO_NN nickname for Lotus Extended Search

```
CREATE NICKNAME ADMINISTRATOR.DEMO_NN ( WEBURL VARCHAR (100) ,TITLE VARCHAR
(100),WEBDESCRIPTION VARCHAR (100) ,DESCRIPTION VARCHAR (100) ,WEBTITLE VARCHAR
```

```
(100))FOR SERVER "EXTENDEDSEARCHSERVER"  OPTIONS(SORTFIELD 'DOC_RANK' , MAXHIT
'50',SORTORDER 'A',VERTICAL_TABLE 'NO',APPLICATIONID 'Demo',TIMEOUT '30',
TOTALMAXHIT '50' , DATASOURCE 'Google!');
```

Figure 5-14 shows the DB2 Control Center view of the ESSERVER wrapper configuration in DB2 II.

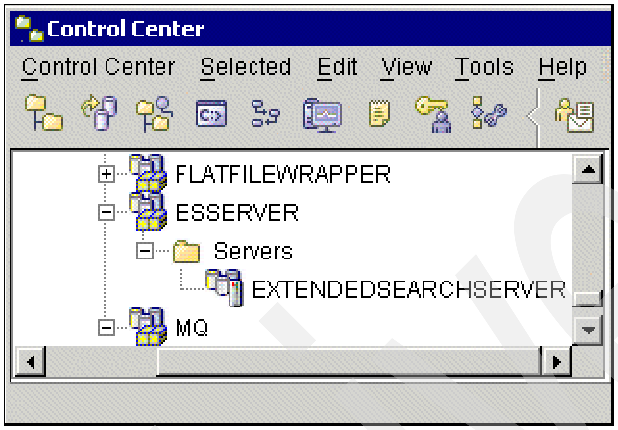


Figure 5-14 Extended Search (ESSERVER) wrapper

Example 5-3 on page 110 shows a simple example of querying a nickname using the Demo application shipped with Lotus Extended Search v4. Any supported LES data source such as e-mail can be queried in the same manner.

Example 5-5 is an example of creating a nickname **EMAIL_NN** against an e-mail data source.

Example 5-5 Creating a nickname EMAIL_NN on an e-mail data source

```
CREATE NICKNAME EMAIL_NN (Owner VARCHAR(80), TO VARCHAR(80), Date DATE, Subject
VARCHAR(80)) FOR SERVER esServer OPTIONS (APPLICATIONID 'db2iidemo', CATEGORY
'Exchange Server;Lotus Notes', VERTICAL_TABLE 'YES',) TIMEOUT '60', MAXHITS
'100', TOTALMAXHITS '1000')
```

Example 5-6 is an example of an SQL query against a nickname referencing an e-mail data source.

Example 5-6 Sample query against a nickname referencing an e-mail data source

```
SELECT * FROM EMAIL_NN WHERE  ESWRAPPER.ES_Search(DOC_RANK,'(("SUBJECT" IN "DB2
II") AND ("DATE" BETWEEN "11/01/2003" AND "12/30/2003"))') = 1
```

Attention: Without the DB2 II nickname for LES, the portlet developer must access LES directly using its native GQL syntax. While the My Search Portlet accessed only the LES as a single data source and could have directly used the GQL interface without having to define a DB2 II nickname for it, the benefits of creating a nickname are as follows:

- Transparency to GQL
- Simplicity of the SQL interface in joining LES data with other data sources not supported by LES

5.4.5 Claim Entry Portlet

The Claim Entry Portlet uses DB2 II (through a UDF), and have included it here since it leverages DB2's UDF capability to access WebSphere MQ message queues. This portlet provides John the ability to create a new claim from data accepted through a JSP form that results in a message being inserted into a WebSphere MQ message queue, which will then be retrieved by a program, validated, and inserted into an appropriate database table or file. Figure 5-15 shows the Claim Entry Portlet.

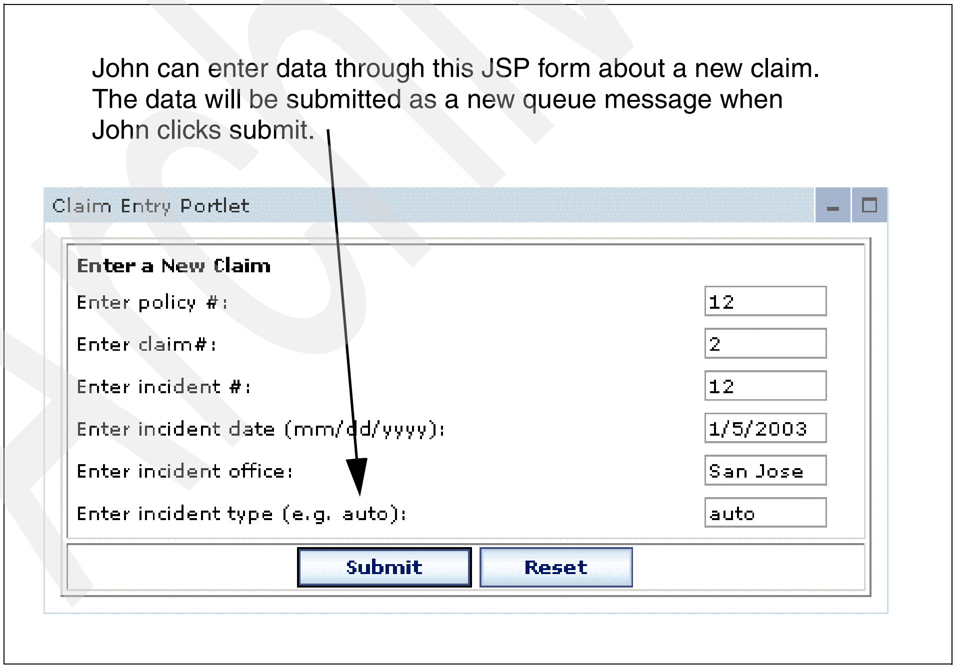


Figure 5-15 Claim Entry Portlet

The Claim Entry Portlet does not display any confirmation data to John. It uses the MQSEND UDF to insert an entry into the WebSphere MQ message queue with a simple SQL statement as shown in Example 5-9.

Example 5-7 SQL that calls the MQSEND UDF to write a message to the queue

```
StringBuffer sb = new StringBuffer();
sb.append("values DB2MQ.MQSEND('DB2.DEFAULT.SERVICE','DB2.DEFAULT.POLICY','");
sb.append(policynum + "," + claimnum + "," + incidentnum + "," + incidentdate);
sb.append("," + office + "," + type + "','NEW_CLAIM')");
```

Attention: The advantage of using the DB2 MQSEND UDF is that the portlet developer can simply connect to the database and execute SQL instead of locating and writing to this queue through another data access bean. In addition, if there is a need to integrate this information with other data sources, then the SQL UDF capability provides superior integration capabilities with these data sources as described in 5.4.6, “Demographics Portlet” on page 113.

5.4.6 Demographics Portlet

The Demographics Portlet lets John enter information about a prospective customer to determine any discounts he might be able to offer on a policy.

In building this portlet, customer supplied information is used to retrieve data from a Web Service that is then input to DB2 Intelligent Miner™ model, which determines a score from which discount eligibility is determined.

All of this is performed in a single SQL query.

Figure 5-16 shows the flow of data through the Demographics Portlet.

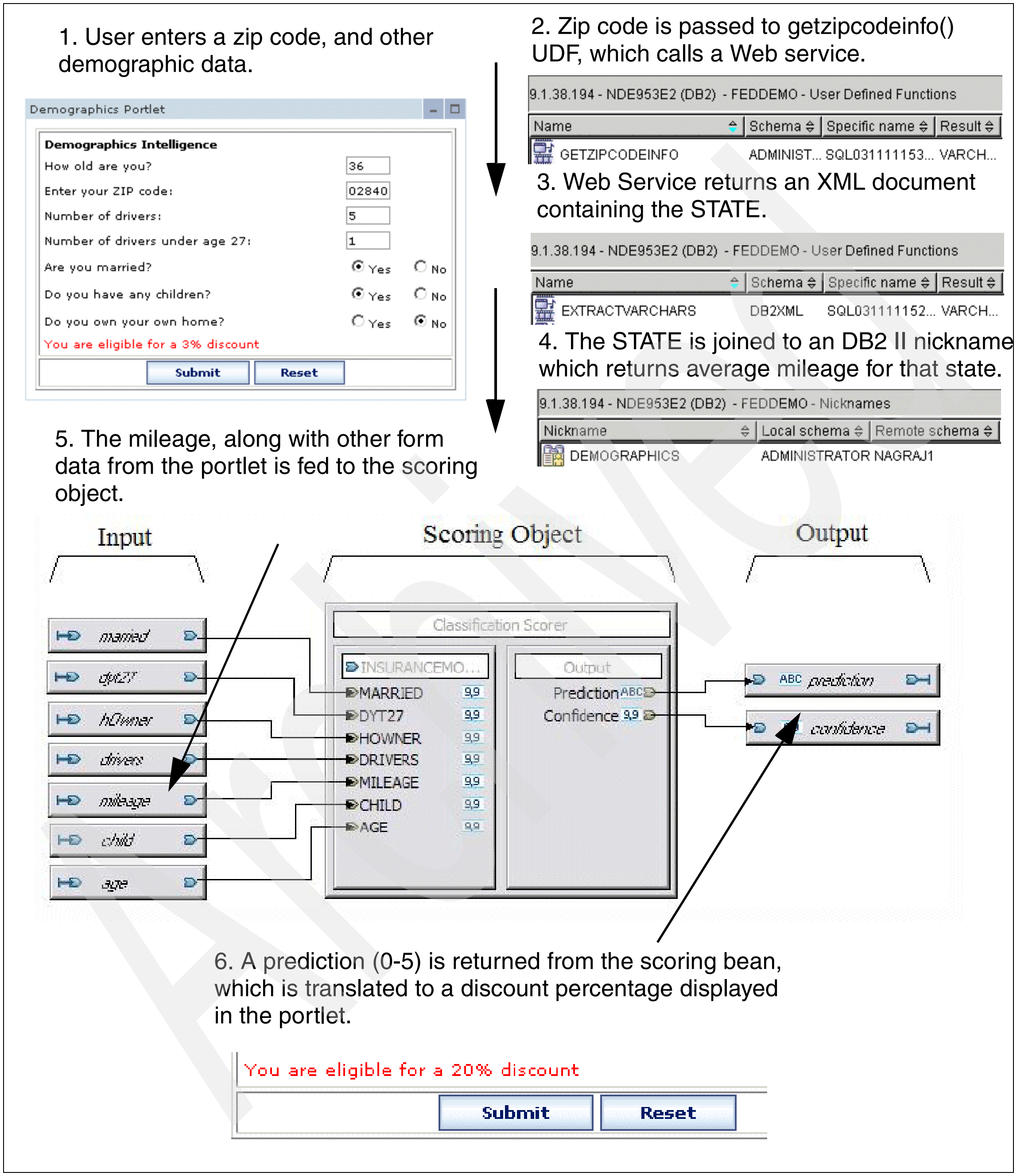


Figure 5-16 Demographics Portlet data flow

Table 5-7 lists the data sources used in the Demographics Portlet.

Table 5-7 Data sources used in the Demographics Portlet

Data source	Type	DB2 II nicknames
Location information	Web Service	No DB2 II — uses UDF
Mileage data	DB2 UDB on OS/390	`DEMOGRAPHICS`
JSP form input	data entered into portlet	No DB2 II
Scoring results	DB2 Intelligent Miner Scoring	No DB2 II — uses DB2 Intelligent Miner Scoring UDF

The Demographics Portlet performs the following:

1. Accepts prospective customer related information through a JSP form — Example 5-8 on page 116 shows the JSP form included in this portlet to collect customer information.
2. Queries a Web Service through DB2's XML Extender UDF, extracts the STATE information for that zip code from the XML result set, and uses the STATE information thus derived to determine the average mileage per driver statistics for that STATE using a DB2 II query. Example 5-9 on page 116 lists the SQL involved in performing this request.
3. This average mileage per driver information is fed to the DB2 Intelligent Miner (called from the portlet using the ScoringBean described in Example 5-7) model along with other customer supplied information (age, marital status, etc.) as shown in Example 5-10 on page 117.

   The result of the ScoringBean is a prediction and confidence level that is based on the design of the DB2 Intelligent Miner model. We used the prediction method **`getPrediction()`** supplied in the ScoringBean to determine discount eligibility.

   > **Attention:** For more information about DB2 Intelligent Miner or the model used in this example, please see:
   > `http://www.ibm.com/software/data/iminer`

4. Example 5-11 on page 118 shows the JSP scriptlet that computes and displays the discount eligibility.

It may seem odd that this application only requests a zip code but not state information from a prospective customer, especially since the application accesses a Web service to determine the state corresponding to the zip code provided. Our objective was to demonstrate the power of DB2 II by using a single query that combined Web Service access (through a UDF), with DB2 data[1] and data mining (also through a UDF), and our emphasis was not necessarily on designing an optimal demographics intelligence portlet.

> **Attention:** Without DB2 II, the data returned from the UDF (state from the Web service) must be manually joined to the mileage data held in DB2 UDB for OS/390. Using DB2 II, the portlet developer can construct an SQL query that joins results from various UDFs with DB2 II nicknames.

Example 5-8 JSP form that takes user input in the Demographics Portlet

```
<%@ page contentType="text/html" %>
<%@ taglib uri='/WEB-INF/tld/portlet.tld' prefix='portletAPI'%>
<jsp:useBean id="returnMyURI" class="java.lang.String" scope="request" />
<jsp:useBean id="imPredicition" class="java.lang.String" scope="request" />
<portletAPI:init/>
......
<FORM method='POST' name="<portletAPI:encodeNamespace value='fForm'/>"
action="<%= returnMyURI %>">
<TD ALIGN="LEFT" width="45%">Enter your ZIP code:</TD>
   <TD ALIGN="left" width="25%">
   <INPUT class="wpsEditField" type="text" name="<portletAPI:encodeNamespace
value='zip'/>" size=5>
   </TD>
</TD>
......
<INPUT type="submit" name="<portletAPI:encodeNamespace value='Submit'/>"
value="Submit" class="wpsButtonText">
<INPUT type="reset" value="Reset" class="wpsButtonText">
</FORM>
```

Example 5-8 highlights the JSP tag library, the useBean tags for the prediction data, and the other portletAPI tags.

Example 5-9 Query combining Web services and DB2 UDB data

```
if (method.equals("getDemoData")) {
   ......
   String zip = (String)s.getAttribute("zip");
   ......
   StringBuffer sb = new StringBuffer();
   sb.append("WITH DEMO_STATE (WEB_STATE) AS (select * from table");
   sb.append("(db2xml.extractVarchars(db2xml.XMLVARCHAR");
   sb.append("(ADMINISTRATOR.getzipcodeinfo('"+zip+"')),");
   sb.append("'/s0:ZipCodesResponse/STATE')) as x )");
   sb.append("select ADMINISTRATOR.DEMOGRAPHICS.MILAGE");
   sb.append("FROM ADMINISTRATOR.DEMOGRAPHICS,");
```

[1] this could well be any other data source supported by DB2 II

```
   sb.append("DEMO_STATE where ADMINISTRATOR.DEMOGRAPHICS.STATE =");
   sb.append("DEMO_STATE.WEB_STATE");
   ......
}
```

Example 5-10 ScoringBean query invoking DB2 Intelligent Miner scoring model

```
public ScoringBean(Connection connection) throws SQLException {
......
StringBuffer sql = new StringBuffer();
sql.append("WITH \n");
sql.append("SCORER_INPUT(MARRIED, DYT27, HOWNER, DRIVERS, MILEAGE, CHILD, AGE)
AS \n");
sql.append("(\n");
sql.append("SELECT DISTINCT \n");
sql.append("CAST(? AS DOUBLE) AS MARRIED, \n");
sql.append("CAST(? AS DOUBLE) AS DYT27, \n");
sql.append("CAST(? AS DOUBLE) AS HOWNER, \n");
sql.append("CAST(? AS DOUBLE) AS DRIVERS, \n");
sql.append("CAST(? AS DOUBLE) AS MILEAGE, \n");
sql.append("CAST(? AS DOUBLE) AS CHILD, \n");
sql.append("CAST(? AS DOUBLE) AS AGE\n");
sql.append("FROM\n");
sql.append("IDMMX.CLASSIFMODELS\n");
sql.append("), \n");
sql.append("SCORER_RESULT(RESULT) AS \n");
sql.append("(\n");
sql.append("SELECT \n");
sql.append("IDMMX.DM_applyClasModel(IDMMX.CLASSIFMODELS.MODEL,
IDMMX.DM_impApplData(rec2xml(1.0, 'COLATTVAL', '', SCORER_INPUT.MARRIED,
SCORER_INPUT.DYT27, SCORER_INPUT.HOWNER, SCORER_INPUT.DRIVERS,
SCORER_INPUT.MILEAGE, SCORER_INPUT.CHILD, SCORER_INPUT.AGE))) AS RESULT\n");
sql.append("FROM\n");
sql.append("SCORER_INPUT, IDMMX.CLASSIFMODELS\n");
sql.append("WHERE\n");
sql.append("IDMMX.CLASSIFMODELS.MODELNAME = 'IMSAMPLE.INSURANCEMODEL' \n");
sql.append("), \n");
sql.append("SCORER_OUTPUT(PREDICTION, CONFIDENCE) AS \n");
sql.append("(\n");
sql.append("SELECT \n");
sql.append("IDMMX.DM_getPredClass(SCORER_RESULT.RESULT) AS PREDICTION, \n");
sql.append("IDMMX.DM_getConfidence(SCORER_RESULT.RESULT) AS CONFIDENCE\n");
sql.append("FROM\n");
sql.append("SCORER_RESULT\n");
sql.append(")\n");
sql.append("SELECT \n");
sql.append("SCORER_OUTPUT.PREDICTION AS prediction, \n");
sql.append("SCORER_OUTPUT.CONFIDENCE AS confidence \n");
sql.append("FROM\n");
```

```
sql.append("SCORER_OUTPUT\n");
stmt = connection.prepareStatement(sql.toString());
......
```

Example 5-11 JSP scriptlet displaying discount eligibility percentage

```
<% if (imPredicition != null || (imPredicition.length() > 0)) { %>
<% if (imPredicition.equals("0")) { %> <font color="#ff0000">You are NOT
eligible for coverage</font> <%}%>
<% if (imPredicition.equals("1")) { %> <font color="#ff0000">You are eligible
for a 3% discount</font> <%}%>
<% if (imPredicition.equals("2")) { %> <font color="#ff0000">You are eligible
for a 8% discount</font> <%}%>
<% if (imPredicition.equals("3")) { %> <font color="#ff0000">You are eligible
for a 13% discount</font> <%}%>
<% if (imPredicition.equals("4")) { %> <font color="#ff0000">You are eligible
for a 20% discount</font> <%}%>
<% if (imPredicition.equals("5")) { %> <font color="#ff0000">You are eligible
for a 30% discount</font> <%}%>
<% } %>
```

5.4.7 Claim Notify Portlet

The Claim Notify Portlet goes hand-in-hand with the Claim Entry Portlet. John can view all claims entered by himself or others, with additional details about the customer and police reports related to the claims. In our simple scenario, the Claim Notify Portlet combines information from the WebSphere MQ message queue, customer data stored in a DB2 UDB table, and police reports in XML files. This portlet does not accept any user input parameters, instead it responds to a request by returning information on all messages in the WebSphere MQ message queue.

> **Note:** In a real world environment, the claims will either still be sitting in the message queue waiting to be processed, or written to a claims table by the message queue processing program. In this case, this portlet needs to UNION the claims in the message queues as well as in the claims table. Another consideration is that claims in the message queue may not have been validated completely, and may therefore fail more stringent validation processing by the message queue processing program.

Figure 5-17 shows the Claim Notify Portlet.

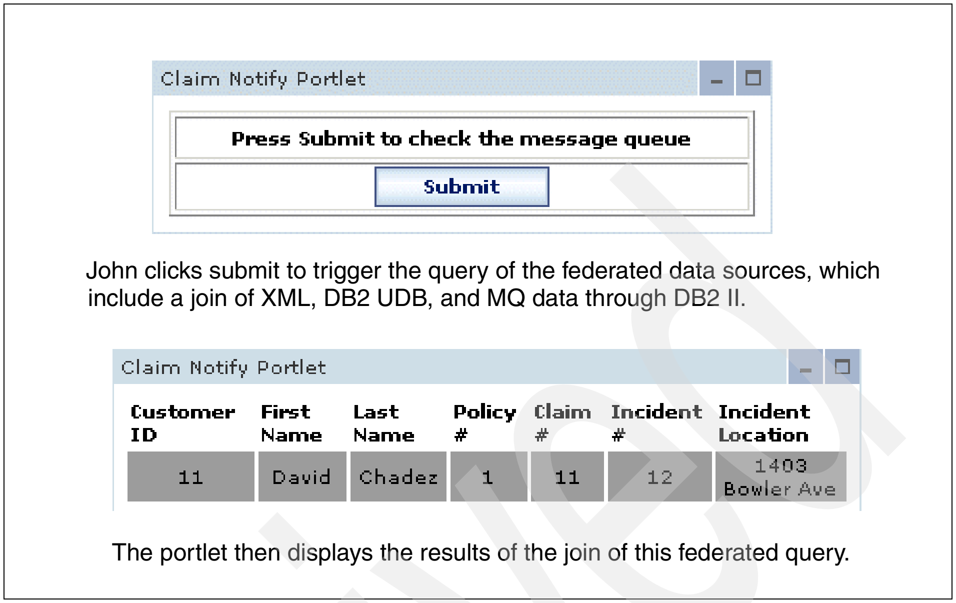


Figure 5-17 Claim Notify Portlet

Table 5-8 lists the data sources used by this portlet.

Table 5-8 Data sources used in the Claim Notify Portlet

Data sources	Type	DB2 II nicknames
Police incident reports	XML files	**REPORTS_XML**
Customer information	DB2 UDB	**CUSTOMER_DB2**
Claim information	MQ message queue	DB2 II (through a UDF)

We created a database view **GET_ALL_CLAIMS** involving the nicknames **REPORTS_XML** and **CUSTOMER_DB2,** and joins and aggregates them as shown in Example 5-12. This considerably simplified the SQL used in the portlet to retrieve the desired information about claims as shown in Example 5-13.

Example 5-12 SQL that builds the GET_ALL_CLAIMS view

```
CREATE view ADMINISTRATOR.GET_ALL_CLAIMS AS
SELECT CUSTOMER_DB2.CUSTOMER_ID, CUSTOMER_DB2.FIRST_NAME,
   CUSTOMER_DB2.LAST_NAME, PULL_NEW_CLAIMS_OFF_QUEUE.POLICY_NUMBER,
   PULL_NEW_CLAIMS_OFF_QUEUE.CLAIM_NUMBER,
   PULL_NEW_CLAIMS_OFF_QUEUE.INCIDENT_NUMBER,
   PULL_NEW_CLAIMS_OFF_QUEUE.INCIDENT_DATE,
   PULL_NEW_CLAIMS_OFF_QUEUE.INCIDENT_OFFICE,
```

```
   PULL_NEW_CLAIMS_OFF_QUEUE.INCIDENT_TYPE,
   PULL_NEW_CLAIMS_OFF_QUEUE.CORRELID,
   REPORTS_XML.INCIDENTID, REPORTS_XML.INCIDENTLOC
FROM ADMINISTRATOR.CUSTOMER_DB2
AS CUSTOMER_DB2
INNER JOIN ADMINISTRATOR.PULL_NEW_CLAIMS_OFF_QUEUE
AS PULL_NEW_CLAIMS_OFF_QUEUE ON CUSTOMER_DB2.CUSTOMER_ID =
PULL_NEW_CLAIMS_OFF_QUEUE.POLICY_NUMBER
LEFT OUTER JOIN ADMINISTRATOR.REPORTS_XML
AS REPORTS_XML ON PULL_NEW_CLAIMS_OFF_QUEUE.INCIDENT_NUMBER =
INTEGER(REPORTS_XML.INCIDENTID)
```

The **GET_ALL_CLAIMS** view itself uses another view called **PULL_NEW_CLAIMS_OFF_QUEUE**, which queries the message queue and filters rows based on the claim identifier. Figure 5-18 shows the SQL for the **PULL_NEW_CLAIMS_OFF_QUEUE** view.

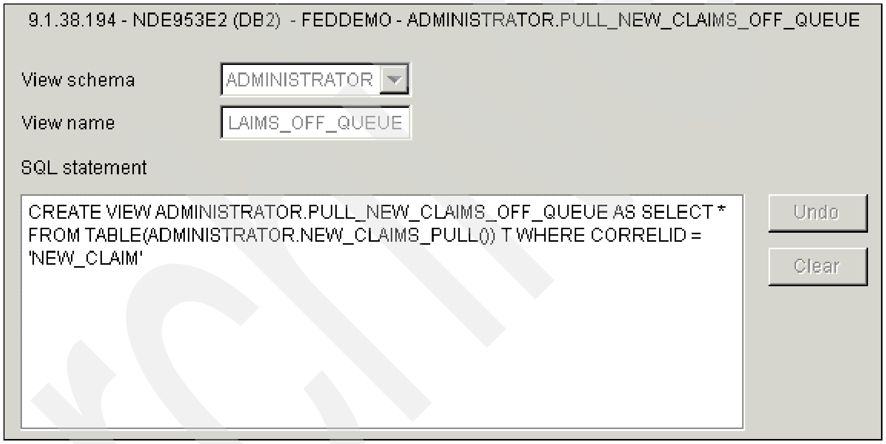


Figure 5-18 DB2 Control Center window with view definition

NEW_CLAIMS_PULL is a UDF that reads claims off the message queue.

Example 5-13 Claim Notify Portlet SQL that queries the GET_ALL_CLAIMS view

```
if (method.equals("getClaim")) {
   StringBuffer sb = new StringBuffer();
   sb.append("select customer_id, first_name, last_name, incident_date,
policy_number,");
   sb.append("claim_number, incidentloc from administrator.GET_ALL_CLAIMS");
   ......
}
```

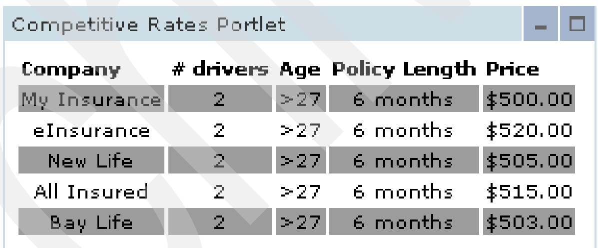
The Claim Notify Portlet is a very good example of the power of DB2 II. By creating the **GET_ALL_CLAIMS** view based on complex joins (including inner and outer joins) of nicknames on XML files and DB2 data, and another view containing a UDF to read data from a message queue, the portlet developer's task is considerably simplified as shown in Example 5-13.

> **Attention:** Without DB2 II nicknames, the portlet developer must access the XML files, DB2 data, and the message queue using native APIs, and perform the mappings and joins and required optimizations — not a simple task by any means.

5.4.8 Competitive Rates Portlet

This portlet provides John with the latest rates offered by CFS' competitors so that he may take them into account when pricing policies. The Competitive Rates Portlet is a read-only portlet that gets current information on the competition's rates from CFS' administrators. The competitive rate information is provided in the form of a delimited ASCII flat file and is updated on a daily basis.

Figure 5-19 shows the Competitive Rates Portlet.



John can view competitor information to assist him in pricing policies. This data comes in the form of a delimited ASCII flat file.

Figure 5-19 Competitive Rates Portlet

Example 5-9 lists the data source for this portlet.

Table 5-9 Data source used in the Competitive Rates Portlet

Data source	Type	Nicknames
Competitive rate information	Delimited ASCII flat file	**COMPETITIVERATES2**

Figure 5-20 shows the definition of this data source in the DB2 Control Center, along with the file configuration.

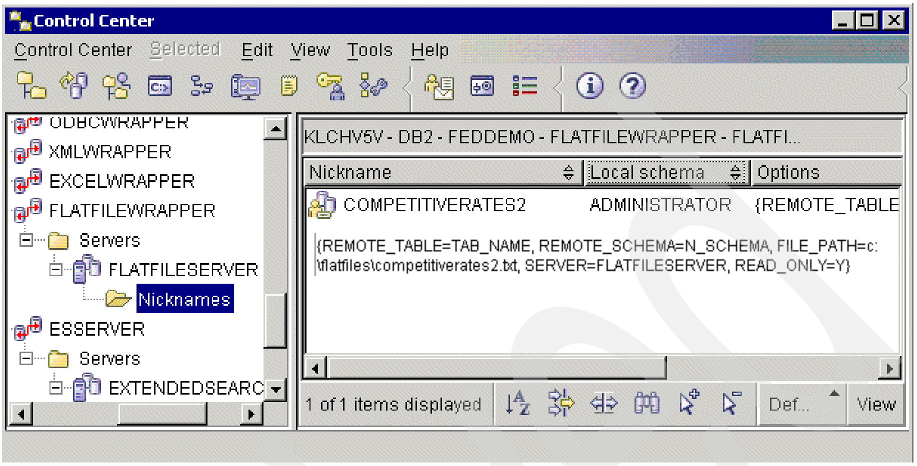


Figure 5-20 DB2 Control Center view of the Flat File wrapper

Figure 5-21 shows a sampling of the contents of this file; the first line in the file is a mapping for the rest of the data.

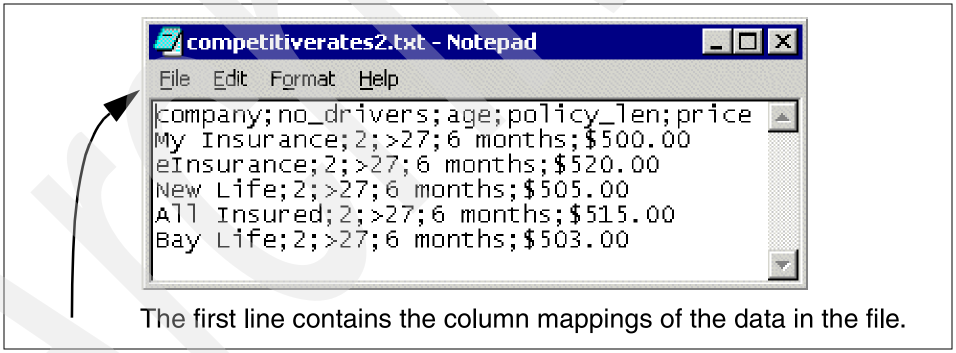


Figure 5-21 Sample contents of the flat file containing daily competitive rates data

Example 5-14 lists the SQL query used in the portlet. In this example, the query is executed directly in the portlet's **doView()** method, and JSP is not included to display the results; they are simply rendered as the query returns rows from the result set.

Example 5-14 doView() method in the Competitive Rates Portlet

```
public void doView(PortletRequest request, PortletResponse response)
   throws PortletException, IOException {

   StringBuffer sb = new StringBuffer();
   sb.append("select * from Administrator.CompetitiveRates");
......
   response.getWriter().println("<html>");
   response.getWriter().println("<body>");
   response.getWriter().println("<table>");
   response.getWriter().println("<tr>");
   response.getWriter().println("<td colspan=2><b>Company</b></td><td
colspan=2><b># drivers</b></td>"+
      "<td colspan=2><b>Age</b></td><td colspan=2><b>Policy Length</b></td><td
colspan=2><b>Price</b></td>");
   response.getWriter().println("</tr>");
......
   ResultSet rs = st.executeQuery(sb.toString());
   while (rs.next()) {
      company = rs.getString(1);
      no_drivers = rs.getString(2);
      age = rs.getString(3);
      policy_len = rs.getString(4);
      price = rs.getString(5);
......
      response.getWriter().println("<tr bgcolor=\""+bgColor+"\">");
      response.getWriter().println("<td colspan=2 align=\"center\">"+company+
         "</td><td colspan=2 align=\"center\">"+no_drivers+
         "</td><td colspan=2 align=\"center\">"+age+
         "</td><td colspan=2 align=\"center\">"+policy_len+
         "</td><td colspan=2 align=\"center\">"+price+"</td>");
      response.getWriter().println("</tr>");
......
```

While the query in Example 5-14 only accesses a single flat file, DB2 II makes it possible to use SQL to join this flat file data with other data sources including unstructured data. With the bulk of the work delegated to DB2 II, the portlet developer is left with a simple query to present John with competitive market information.

> **Attention:** Without DB2 II, the portlet developer has to use file system APIs to access competitive rates information rather than SQL. While this is not necessarily a complex task in this portlet, the portlet developer's task becomes significantly more complex when data from a flat file needs to be joined with other data sources.

5.4.9 Cotton-Wood Reports Portlet

The Cotton-Wood Reports Portlet lets John select from a list of different “canned” reports. These reports organize information about claims, policies, and customers. As in the previous examples, the data needed for these reports is spread across the company on different servers and repositories.

Figure 5-22 shows the Cotton-Wood Reports Portlet.

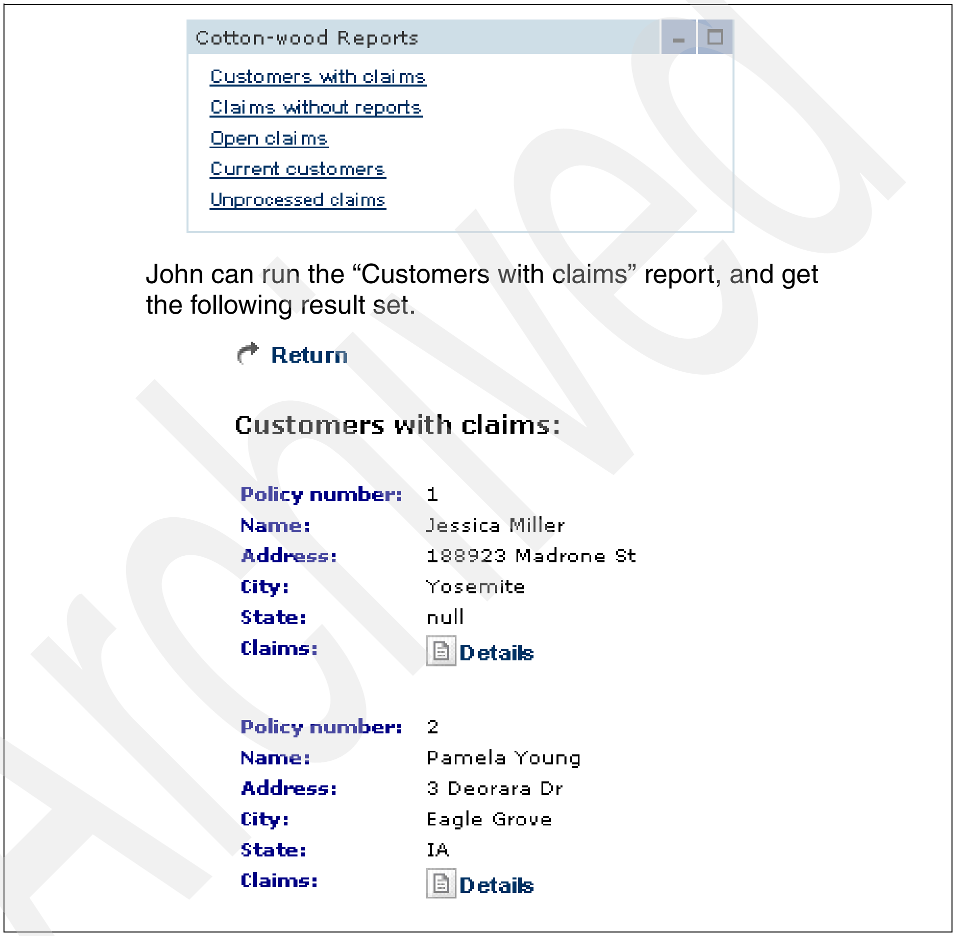


Figure 5-22 Cotton-Wood ReportsPortlet with sample report

Table 5-10 lists the data sources used in the Cotton-Wood Reports portlet.

Table 5-10 Data sources used in the Cotton-Wood Reports Portlet

Data source	Type	DB2 II nickname
Police reports	XML files	**REPORTS_XML**
Customer data	DB2 UDB	**CUSTOMER_DB2**
Incident data	Lotus Notes database	**CLAIMS_DOMINO**

By creating nicknames on the XML, DB2 UDB and the Lotus Notes database, a simple SQL join is used to aggregate necessary information for the reports as shown in Example 5-15, Example 5-16, and Example 5-17.

Example 5-15 Claim Reports: All customer claims query

```
SELECT DEMO.CUSTOMER_DB2.CUSTOMER_ID,
   DEMO.CUSTOMER_DB2.FIRST_NAME || ' ' ||
   DEMO.CUSTOMER_DB2.LAST_NAME,
   DEMO.CUSTOMER_DB2.ADDRESS,
   DEMO.CUSTOMER_DB2.CITY,
   DEMO.CUSTOMER_DB2.STATE,
   DEMO.CUSTOMER_DB2.POLICY_TYPE,
   DEMO.CUSTOMER_DB2.POLICY_PREMIUM,
   DEMO.CLAIMS_DOMINO.ODATE,
   DEMO.CLAIMS_DOMINO.CDATE,
   DEMO.CLAIMS_DOMINO.INCIDENT_ID,
   DEMO.CLAIMS_DOMINO.DOC_ID,
   DEMO.REPORTS_XML.OFFICERNAME,
   DEMO.REPORTS_XML.INCIDENTDESC
FROM DEMO.CLAIMS_DOMINO, DEMO.REPORTS_XML, DEMO.CUSTOMER_DB2
WHERE DEMO.CLAIMS_DOMINO.INCIDENT_ID = DEMO.REPORTS_XML.INCIDENTID
AND DEMO.CUSTOMER_DB2.CUSTOMER_ID = INTEGER(DEMO.CLAIMS_DOMINO.CUSTOMER_ID)
AND (DEMO.CLAIMS_DOMINO.INCIDENT_TYPE = 'Fault'
OR DEMO.CLAIMS_DOMINO.INCIDENT_TYPE = 'No Fault' )
```

Example 5-16 Claim Reports: Claims without police reports query

```
SELECT B.CUSTOMER_ID, B.INCIDENT_ID, A.CITY, A.STATE
FROM DEMO.CUSTOMER_DB2 A, DEMO.CLAIMS_DOMINO B
WHERE A.CUSTOMER_ID=INTEGER(B.CUSTOMER_ID)
AND B.INCIDENT_ID NOT IN
   (SELECT INCIDENTID FROM DEMO.REPORT_XML)
   AND (B.INCIDENT_TYPE='FAULT' OR B.INCIDENT_TYPE='NO FAULT')
```

Example 5-17 Claim Reports: Open claims query

```
SELECT A.CUSTOMER_ID, B.FIRST_NAME || B. LAST_NAME,
   A.INCIDENT_ID, A.INCIDENT_TYPE, A.ODATE
FROM DEMO.CLAIMS_DOMINO A, DEMO.CUSTOMER_DB2 B
```

```
WHERE INTEGER(A.CUSTOMER_ID)=B.CUSTOMER_ID
AND A.CDATE IS NULL
AND (A.INCIDENT_TYPE='FAULT' OR A.INCIDENT_TYPE = 'NO FAULT')
```

The portlet developer may choose to write customized versions or use off-the-shelf portlet such as My Query Reports Portlet.

> **Note:** Off-the-shelf portlets can be downloaded from the IBM portlet catalog at: http://www.ibm.com/websphere/portal/portlet/catalog

With DB2 II enabling joins of federated data, the portlet developer building a custom portlet can place the federated data join query in a data bean, and list the result set through a JSP. Example 5-18 lists the code that constructs the data bean, and shows how the data bean is set as part of the portlet request attribute to be displayed in the JSP.

Example 5-18 Portlet code using a data bean to access DB2 II

```
......
}else if(method.equals("all")){
   Query5 dataBean = new Query5("jdbc/cicdemo");
   request.setAttribute("dataBean",dataBean);
   if(!dataBean.connect()){
      System.out.println("Not able to connect to the database");
      log.error("cotton:JDBCPortlet:Could not connect to the database, please
check user, password and url");
   }
   if(!dataBean.executeQuery()){
      System.out.println("Not able to excecute the query");
      log.error("cotton:SQLPortlet: Could not execute the sql query");
   }
   PortletURI returnBackURI = response.createReturnURI();
   returnBackURI.addAction("default");
   request.setAttribute("returnBackURI",returnBackURI.toString());
   PortletURI sendMessageURI = response.createReturnURI();
   sendMessageURI.addAction("Send_Message");
   request.setAttribute("sendMessageURI",sendMessageURI.toString());
   getPortletConfig().getContext().include("SQL/jsp/allTab.jsp",request,
response);
   dataBean.close();
......
```

Attention: Without DB2 II, the portlet developer must manage connection pools and collect data individually from each of the data sources. “DB2 II versus no DB2 II: A sample portlet” on page 133 gives a side-by-side comparison of this portlet with and without using DB2 II. More than twice as much code is necessary without DB2 II.

5.4.10 Cotton-Wood Search Portlet

The Cotton-Wood Search Portlet lets John search for claims by customer name or incident number and it is a custom portlet.

Figure 5-23 shows the Cotton-Wood Search Portlet.

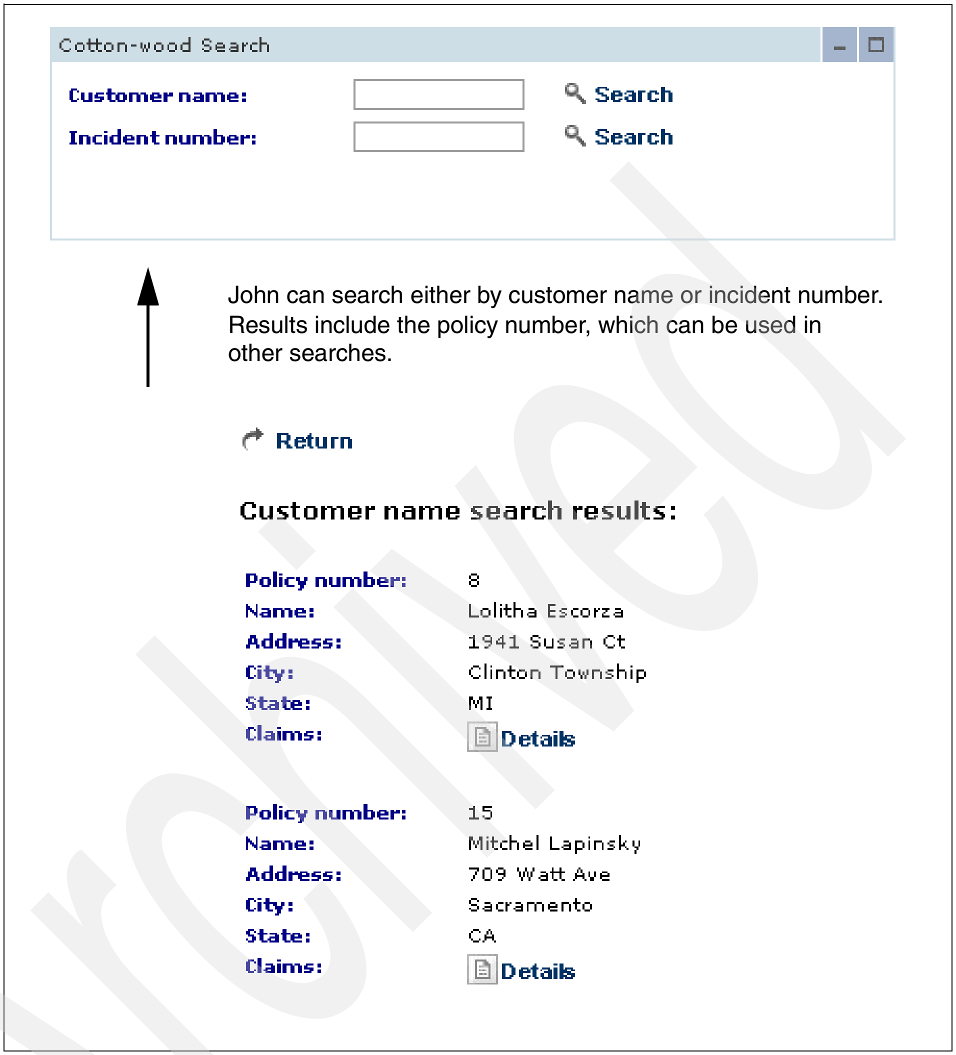


Figure 5-23 The Cotton-Wood Search Portlet with sample result set

Table 5-11 lists the data source used in the Cotton-Wood Search Portlet.

Table 5-11 Data sources used in the Cotton-Wood Search Portlet

Data source	Type	DB2 II nickname
Customer data	DB2 UDB	`CUSTOMER_DB2`

Attention: Without DB2 II, this custom portlet cannot query multiple data sources. While this example is only a single DB2 II nickname of a DB2 UDB data source, this portlet can query a join or union of multiple DB2 II nicknames. From the perspective of the JDBC data source, DB2 II is just another DB2 database.

An alternative way to build this portlet is by using the off-the-shelf JDBC Business Object Builder portlet, which lets you configure a data source and choose a database schema to be used. The JDBC Business Object Builder portlet also allows you to search and update fields. See 5.6, “Configuring the JDBC Business Object Builder” on page 142 for details and step-by-step instructions for using the JDBC Business Object Builder.

5.4.11 Cotton-Wood Details Portlet

The Cotton-Wood Details Portlet is another tool that John uses to search for information of interest by policy number, which is a key field that is included in a number of reports including the output of the Cotton-Wood Reports Portlet.

Figure 5-24 shows the Cotton-Wood Details Portlet with claim information.

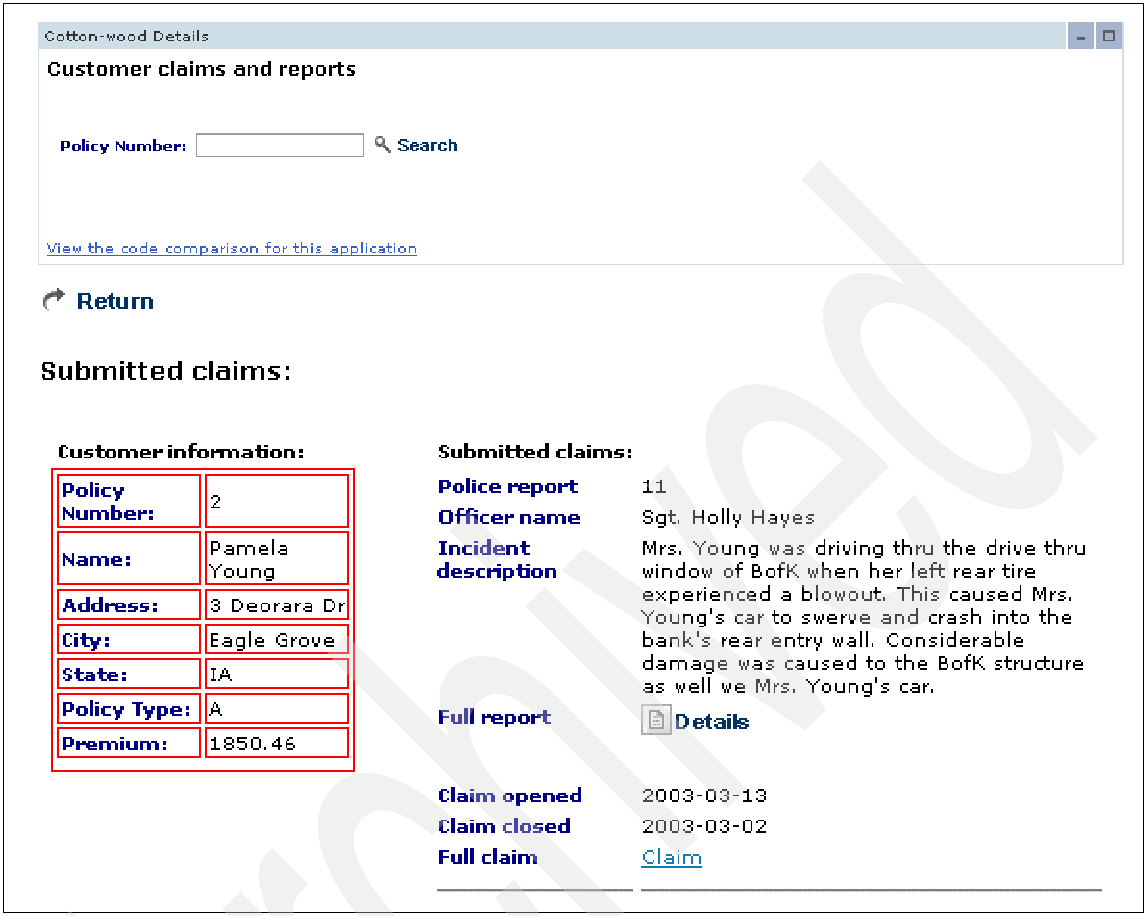


Figure 5-24 Cotton-Wood Details Portlet with sample report data

The Cotton-Wood Details Portlet uses DB2 II nicknames to access claims information contained in the data sources listed in Table 5-12.

Table 5-12 Data sources used in the Cotton-Wood Details Portlet

Data source	Type	DB2 II nickname
Police report	XML file	`REPORTS_XML`
Full claim report	Lotus Notes, DB2, XML	`CLAIMS_DOMINO, CUSTOMER_DB2, REPORTS_XML`

In this portlet, SQL queries are issued against nicknames that refer to XML Lotus Notes and DB2 UDB as shown in Example 5-19 and Example 5-20.

Example 5-19 Three-way join that generates customer policy information

```
SELECT
   DEMO.CUSTOMER_DB2.CUSTOMER_ID,
   DEMO.CUSTOMER_DB2.FIRST_NAME || ' ' || DEMO.CUSTOMER_DB2.LAST_NAME,
   DEMO.CUSTOMER_DB2.ADDRESS,
   DEMO.CUSTOMER_DB2.CITY,
   DEMO.CUSTOMER_DB2.STATE,
   DEMO.CUSTOMER_DB2.POLICY_TYPE,
   DEMO.CUSTOMER_DB2.POLICY_PREMIUM,
   DEMO.CLAIMS_DOMINO.ODATE,
   DEMO.CLAIMS_DOMINO.CDATE,
   DEMO.CLAIMS_DOMINO.INCIDENT_ID,
   DEMO.CLAIMS_DOMINO.DOC_ID,
   DEMO.REPORTS_XML.OFFICERNAME,
   DEMO.REPORTS_XML.INCIDENTDESC
FROM
   DEMO.CLAIMS_DOMINO, DEMO.REPORTS_XML, DEMO.CUSTOMER_DB2
WHERE
   DEMO.CLAIMS_DOMINO.INCIDENT_ID = DEMO.REPORTS_XML.INCIDENTID
AND DEMO.CUSTOMER_DB2.CUSTOMER_ID = INTEGER(DEMO.CLAIMS_DOMINO.CUSTOMER_ID)
AND (DEMO.CLAIMS_DOMINO.INCIDENT_TYPE = 'Fault'
OR DEMO.CLAIMS_DOMINO.INCIDENT_TYPE = 'No Fault')
AND DEMO.CUSTOMER_DB2.CUSTOMER_ID = :cust_id
```

Example 5-20 Query to retrieve police reports from XML documents

```
select vicname,
   incidentloc,
   incidentdesc,
   injury,
   medreq,
   medname,
   medproc,
   medhospitalization,
   medhospital,
   officername,
   propdamage,
   proptype,
   proploc,
   propown,
   propphone
from demo.reports_xml
where incidentid = <incident_id>
```

These queries are found in the data bean classes that are called from Cotton-Wood Details Portlet as shown in Example 5-21 and Example 5-22.

Example 5-21 Portlet code that constructs the data bean in Example 5-19

```
......
} else if (method.equals("showCustomerPolicy")) {
   //Get the customer id from the session
   String cust_id = (String)s.getAttribute("cust_id");

   //Run the query through the JavaBean
   Query1 dataBean = new Query1("jdbc/cicdemo");
   request.setAttribute("dataBean",dataBean);
   if(!dataBean.connect()){
      System.out.println("Not able to connect to the database");
      log.error("cotton:Could not connect to the database, please check user,
password and url");
   }

   dataBean.setCust_id(cust_id);
   if(!dataBean.executeQuery()){
      System.out.println("Not able to execute the query");
      log.error("cotton: Could not execute the sql query");
   }
......
```

Example 5-22 Portlet code that constructs the data bean in Example 5-20

```
......
else if (method.equals("getPoliceReport")){
   String incident_id = (String)s.getAttribute("incident_id");
   Query2 dataBean = new Query2("jdbc/cicdemo");

   //Add the dataBean in the request to access it from the jsp page
   request.setAttribute("dataBean",dataBean);
   //Create the connection to the database
   if(!dataBean.connect()){
      System.out.println("Not able to connect to the database");
      log.error("cotton:Could not connect to the database, please check user,
password and url");
   }
   //Run the sql query
   dataBean.setIncident_id(incident_id);
   if(!dataBean.executeQuery()){
      System.out.println("Not able to excecute the query");
      log.error("cotton: Could not execute the sql query");
   }
......
```

5.5 DB2 II versus no DB2 II: A sample portlet

The Cotton-Wood Reports Portlet is a good example to illustrate application developer productivity gains through reduced complexity when using DB2 II.

The Cotton-Wood Reports Portlet aggregates data from three separate data stores (XML, Lotus Notes, and DB2 UDB).

Building this portlet using native APIs involves the following:

1. Build a join capability into the portlet (Java code). If there had been a request for results in a particular sequence then a sort would be required as well.
2. Implement a DB2 data access class.
3. Implement a Notes/Domino access class.
4. Implement an XML access class.

Using DB2 II, the portlet developer only needs to do the following:

1. Build the data access portion of the portlet.
2. Create a DB2 access class to access the DB2 II federated database.

The following sections illustrate the code that is required to develop the Cotton-Wood Reports portlet using native APIs and DB2 II respectively.

5.5.1 Portlet code using native APIs

Besides the obvious advantages of mapping distributed heterogeneous relational and non-relational data sources into a “local” relational format, DB2 II provides significant performance benefits by handling join strategies and optimization as well.

Example 5-23 shows the complexity of performing just a join when using native APIs.

Example 5-23 Sample code to join multiple data sources using native APIs

```
........
String CUSTOMER_ID="";
String CUSTOMER_NAME="";
String ADDRESS="";
String CITY="";
String STATE="";
String POLICY_TYPE="";
String POLICY_PREMIUM="";

// Create an Hashtable to store the results
Hashtable incidents = new Hashtable();
```

```
// Connect to db2, excecute the query and get the result

// using the db2 access class
DB2Access db2 = new DB2Access("jdbc/cicdemo");
db2.connect();
db2.setCust_id(cust_id);
db2.executeQuery();
db2.getNextResult();
CUSTOMER_ID = db2.getCustID();
CUSTOMER_NAME = db2.getName();
ADDRESS = db2.getAddr();
CITY = db2.getCity();
STATE = db2.getState();
POLICY_TYPE = db2.getType();
POLICY_PREMIUM = db2.getCost();

// Get the results from Notes and store all the matching
// incidents from notes into the hashtable
NotesAccess notes = new NotesAccess("C:\\Notes\\Data\\Claims2.nsf");
notes.access(CUSTOMER_ID,incidents);

//Join the notes data and the xml data in the hashtable
XMLAccess xml = new XMLAccess("C:\\Documents and
Settings\\Administrator\\Desktop\\cotton.war\\jsp\\CIC_Code\\Data\\PoliceReport
s.xml");
xml.access(incidents);

//Put the result in a bean to pass it to the jsp page ...
JoinBean join = new
JoinBean(incidents.values(),CUSTOMER_ID,CUSTOMER_NAME,ADDRESS,CITY,STATE,POLICY
_TYPE,POLICY_PREMIUM);
```

To access the DB2 data source, the portlet developer will most likely use JDBC and build a data access bean to handle access to the DB2 data required by this query. Example 5-24 lists most of the code required to build such a class.

Example 5-24 DB2 data access class needed when using native APIs

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 * This class implement the database connection
 * direct connection or through datasources
 * execute the query
```

```
 * and manage the ResultSet.
 */
public class DB2Access {

   private Connection con;
   private String url;
   private String driver;
   private String user;
   private String pass;
   private int nbRows = 0;
   private PreparedStatement stmt_c = null;
   protected ResultSet rs_c = null;
   private boolean isDS;
   private String cust_id="";
   private javax.sql.DataSource ds;
   private String sql = "SELECT 
DEMO.CUSTOMER_DB2.CUSTOMER_ID,DEMO.CUSTOMER_DB2.FIRST_NAME || ' ' || 
DEMO.CUSTOMER_DB2.LAST_NAME, DEMO.CUSTOMER_DB2.ADDRESS, DEMO.CUSTOMER_DB2.CITY, 
DEMO.CUSTOMER_DB2.STATE, DEMO.CUSTOMER_DB2.POLICY_TYPE, 
DEMO.CUSTOMER_DB2.POLICY_PREMIUM FROM DEMO.CUSTOMER_DB2 WHERE 
DEMO.CUSTOMER_DB2.CUSTOMER_ID =";

   public void setCust_id(String c){
   cust_id=c;
   }
   public DB2Access(String url,String driver,String user,String password){
      this.url = url;
      this.driver = driver;
      this.user = user;
      this.pass = password;
      this.isDS = false;
   }
   public DB2Access(String dsName){
      //Retrieve a datasource through the JNDI name service
      java.util.Properties parms = new java.util.Properties();
      parms.setProperty(javax.naming.Context.INITIAL_CONTEXT_FACTORY,
         "com.ibm.websphere.naming.WsnInitialContextFactory");
         try{
            //Create the initial name context
            javax.naming.Context ctx = new javax.naming.InitialContext(parms);
            //Look through the naming service to restrieve the data
            //source object
            ds = (javax.sql.DataSource) ctx.lookup(dsName);
            ctx.close();
         } catch(Exception e){e.printStackTrace();}
      this.isDS = true;
   }

   public void close(){
```

```
            if (rs_c != null) {
            try{
               rs_c.close();
               stmt_c.close();
               con.close();
            }catch(Exception e){System.out.println(e);}
            }
        }

        public ResultSet executeQuery(){
            try{
               stmt_c = con.prepareStatement(sql + cust_id);
               rs_c = stmt_c.executeQuery();
            }catch(SQLException e){
               System.out.println(e);
               rs_c = null;
               return rs_c;
            }
            return rs_c;
        }

        public boolean connect(){
            if(isDS == false){
               try {
                  Class.forName(driver).newInstance();
                  con = DriverManager.getConnection(url,user,pass);
               }catch(Exception e){
                  System.out.println("Connection error");
                  e.printStackTrace();
                  return false;
               }
               return true;
            }
            else{
               try{
                  con = ds.getConnection();
                  return true;
               }catch(Exception e){
                  return false;
               }
            }
        }
        public boolean getNextResult(){
            if(rs_c != null){
               try{
                  nbRows ++;
                  return rs_c.next();
               }catch(SQLException e){return false;}
            }else return false;
```

```
    }
    public String getCustID(){
        try{
            return rs_c.getString(1);
        }catch(SQLException e){return "";}
    }
    ......
}
```

Similar to the data access class for DB2 shown in Example 5-24, a class is also required for access to the Lotus Notes database. Example 5-25 lists the code required to build such a class.

Example 5-25 Lotus Notes data access class needed when using native APIs

```
import java.util.Hashtable;

// For domino access
import lotus.domino.Database;
import lotus.domino.DateTime;
import lotus.domino.Document;
import lotus.domino.Item;
import lotus.domino.NotesFactory;
import lotus.domino.NotesThread;
import lotus.domino.Session;
import lotus.domino.View;

/**
 *
 * This class implement the domino access
 * It performs a native access to the Domino database
 * through the Notes/Domino API.
 *
 */

public class NotesAccess{

    private String notesurl;

    NotesAccess(String notesurl){
        this.notesurl=notesurl;
    }

    public void access(String cust_id, Hashtable incidents){
        try{
            // Initialize a notes session
            NotesThread.sinitThread();
            Session s = NotesFactory.createSession();
            // Connect to the database
```

```
   Database claims = s.getDatabase(null,notesurl);

// Test if the programm was able to open the database
if (!claims.isOpen()){
    System.out.println(
      "Not able to connect to the server notes database");
      System.exit(0);
   }
   Item it;
   DateTime dt=null;
   // Get a view (List of the documents)
   View v = claims.getView("By Policy");

   // Take all the documents one by one to check the
   // Customer_ID
   Document doc = v.getFirstDocument();
   while(doc!=null){
    if(doc.isValid()){
      // Check if the doc has a matching cust id,
      // get the values needed if it is the case
      if(doc.hasItem("Customer_ID")){
         if(doc.getFirstItem("Customer_ID")
            .getValueString().equals(cust_id)){
            Incident incident = new Incident(
               doc.getFirstItem("Incident_ID")
                  .getValueString());
            if(doc.hasItem("ODate")){
               it = doc.getFirstItem("ODate");
               if(it!=null)
                  dt = it.getDateTimeValue();
                  if(dt instanceof DateTime)
                     incident.setODate(dt.toString());
            }
            if(doc.hasItem("CDate")){
               it=doc.getFirstItem("CDate");
               if(it!=null){
                  dt = it.getDateTimeValue();
                  if(dt instanceof DateTime)
                     incident.setCDate(dt.toString());
               }
            }
            incident.setDoc_id(doc.getNoteID());
            incidents.put(incident.getIncident_id(),incident);
         }
      }

    }
    doc = v.getNextDocument(doc);
   }
```

```
        }
        catch (Exception e){
            System.out.println("Connection issue with the Domino server");
            e.printStackTrace();
        }
        finally {
            // close the notes session
            NotesThread.stermThread();
        }
    }
}
```

Finally, an XML access class is needed parse the police records. Example 5-24 lists the code required to do this.

Example 5-26 XML data access class needed when using the native API

```
import java.util.Hashtable;

// to parse the xml
import org.apache.xerces.parsers.DOMParser;
import org.w3c.dom.Document;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

/**
 *
 * This class implement the xml parsing
 * with the DOM API
 * It will match the incidents and update
 * the values when it is possible.
 */

public class XMLAccess{

    String fileURL = "";
    XMLAccess(String url){
        fileURL=url;
    }
    void access(Hashtable ht){
        // Create a DOM parser
        DOMParser parser= new DOMParser();
        try{
            // Parse the file and get it as a tree
            parser.parse(fileURL);
            Document doc = parser.getDocument();
            // Elements of the tree
            Node reports = doc.getDocumentElement();
            NodeList reportList = reports.getChildNodes();
```

```
         NodeList reportElemList;

         // Go down the tree
         for (int i=0;i<reportList.getLength();i++){
            Node report = reportList.item(i);
            reportElemList = report.getChildNodes();
            // Go trough each report from the tree
            for (int j=0;j<reportElemList.getLength();j++){
               Node elem = reportElemList.item(j);
               // If we have an incidentID TAG
               if(elem.getNodeName().equals("IncidentID")){
                  // Check in the hashtable if we have to add it
                  if(ht.containsKey(elem.getFirstChild().getNodeValue())){
                     Incident inc = (Incident) ht.get(elem.getFirstChild()
                     .getNodeValue());
                     // Then copy the data
                     for (int k=0;k<reportElemList.getLength();k++){
                     Node elem2 = reportElemList.item(k);
                     if(elem2.getNodeName().equals("OfficerName"))
                        inc.setOfficername(elem2.getFirstChild()
                           .getNodeValue());
                     if(elem2.getNodeName().equals("IncidentDesc"))
                        inc.setIncidentdesc(elem2.getFirstChild()
                           .getNodeValue());
                     }
                  }
               }
            }
         }
      }catch(Exception e){
         System.out.println("xml parsing error");
         e.printStackTrace();
         }
   }
}
```

> **Attention:** Implementing and managing these interfaces and connections are time-consuming and complex for a portlet developer. It is also prone to error and maintenance costs. We contrast this effort with the code required when using DB2 II for the exact same query shown in 5.5.2, "Portlet code using DB2 II" on page 140.

5.5.2 Portlet code using DB2 II

With DB2 II, the application developer needs to write far less code since complex join strategy and optimization issues are delegated to DB2 II.

With DB2 II, the steps involved are as follows:

1. Build a data access ability into the portlet: JDBC connection setup and management.
2. Build a DB2 access class or even manage the DB2 access from within the portlet class.

Example 5-27 and Example 5-28 list the portlet code required with DB2 II.

Example 5-27 Data access portion of the portlet when using DB2 II

```
........
dataBean.setCust_id(cust_id);
if(!dataBean.executeQuery()){
   System.out.println("Not able to excecute the query");
   log.error("cotton: Could not execute the sql query");
}
//add the action
PortletURI returnURI = response.createReturnURI();
returnURI.addAction("getPoliceReport");
request.setAttribute("returnURI",returnURI.toString());
//add the action for the back button
PortletURI backURI = response.createReturnURI();
backURI.addAction("default");
request.setAttribute("backURI",backURI.toString());

//Invoke the jsp showCustomerPolicy to print the result
getPortletConfig().getContext().include("Customer/jsp/CICShowCustomerPolicy.jsp
", request, response);
dataBean.close();
......
```

The data access class is similar to the DB2 access class shown in Example 5-24 on page 134 where we only need to execute a single SQL statement. The code should be the same except for the SQL and is listed in Example 5-28.

Example 5-28 DB2 data access class when using DB2 II

```
import java.sql.*;

/**
 *
 * the class Query implements JDBCAbstract
 * adding the sql query and the methods to
 * the data
 */

public class Query1 extends JDBCAbstract{
   String cust_id="";
```

```
   public Query1(String url,String driver,String user,String password){
      super(url,driver,user,password);
   }
   public Query1(String dsName){
      super(dsName);
   }
   public void setCust_id(String c){
   cust_id=c;
   }

   public boolean executeQuery(){
      if(!cust_id.equals(""))
         return super.executeQuery("SELECT DEMO.CUSTOMER_DB2.CUSTOMER_ID,
DEMO.CUSTOMER_DB2.FIRST_NAME || ' ' || DEMO.CUSTOMER_DB2.LAST_NAME,
DEMO.CUSTOMER_DB2.ADDRESS, DEMO.CUSTOMER_DB2.CITY, DEMO.CUSTOMER_DB2.STATE,
DEMO.CUSTOMER_DB2.POLICY_TYPE, DEMO.CUSTOMER_DB2.POLICY_PREMIUM,
DEMO.CLAIMS_DOMINO.ODATE, DEMO.CLAIMS_DOMINO.CDATE,
DEMO.CLAIMS_DOMINO.INCIDENT_ID, DEMO.CLAIMS_DOMINO.DOC_ID,
DEMO.REPORTS_XML.OFFICERNAME, DEMO.REPORTS_XML.INCIDENTDESC FROM
DEMO.CLAIMS_DOMINO, DEMO.REPORTS_XML, DEMO.CUSTOMER_DB2 WHERE
DEMO.CLAIMS_DOMINO.INCIDENT_ID = DEMO.REPORTS_XML.INCIDENTID AND
DEMO.CUSTOMER_DB2.CUSTOMER_ID = INTEGER(DEMO.CLAIMS_DOMINO.CUSTOMER_ID) AND
(DEMO.CLAIMS_DOMINO.INCIDENT_TYPE = 'Fault' OR DEMO.CLAIMS_DOMINO.INCIDENT_TYPE
= 'No Fault')  AND DEMO.CUSTOMER_DB2.CUSTOMER_ID=" + cust_id);
      else return false;
   }
      public String getCustID(){
         try{
            return rs_c.getString(1);
         }catch(SQLException e){return "";}
      }
......
}
```

Important: This example clearly demonstrates the advantages of using DB2 II — it enhances the productivity of developers and administrators building portals by significantly reducing the complexity of coding and accessing data sources in a distributed heterogeneous environment.

5.6 Configuring the JDBC Business Object Builder

The Cotton-Wood Search Portlet described in 5.4.10, “Cotton-Wood Search Portlet” on page 127 was custom built. This section describes an alternative method to build the Cotton-Wood Search Portlet using the JDBC Business Object Builder off-the-shelf portlet.

The steps involved are:

1. Add the JDBC Business Object Builder to an existing page in the portal. This is most likely to be part of an application developer portal rather than the user portal of John Watson.

   Note: This step is not shown here.

2. Generate the Cotton-Wood Search Portlet.

5.6.1 Generate the Cotton-Wood Search Portlet

Figure 5-25 shows the JDBC Business Object Builder portal page.

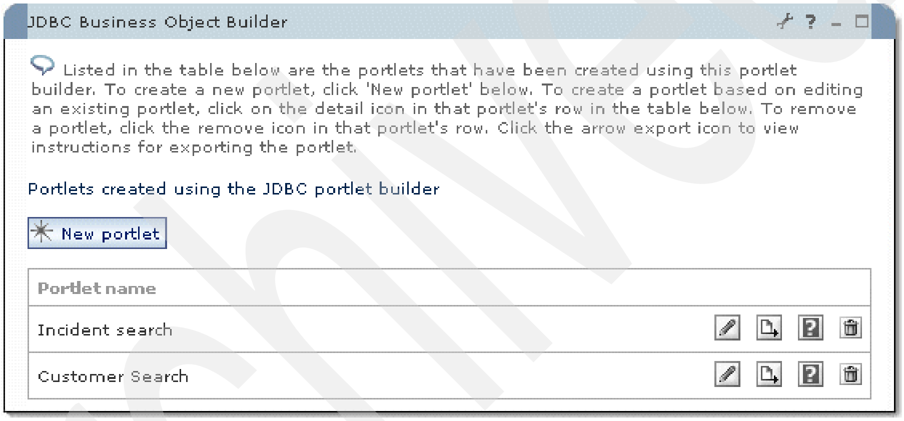


Figure 5-25 The off-the-shelf JDBC Business Object Builder portlet

The following steps will generate the Cotton-Wood Search Portlet:

1. Click ***New Portlet** to view Figure 5-26.

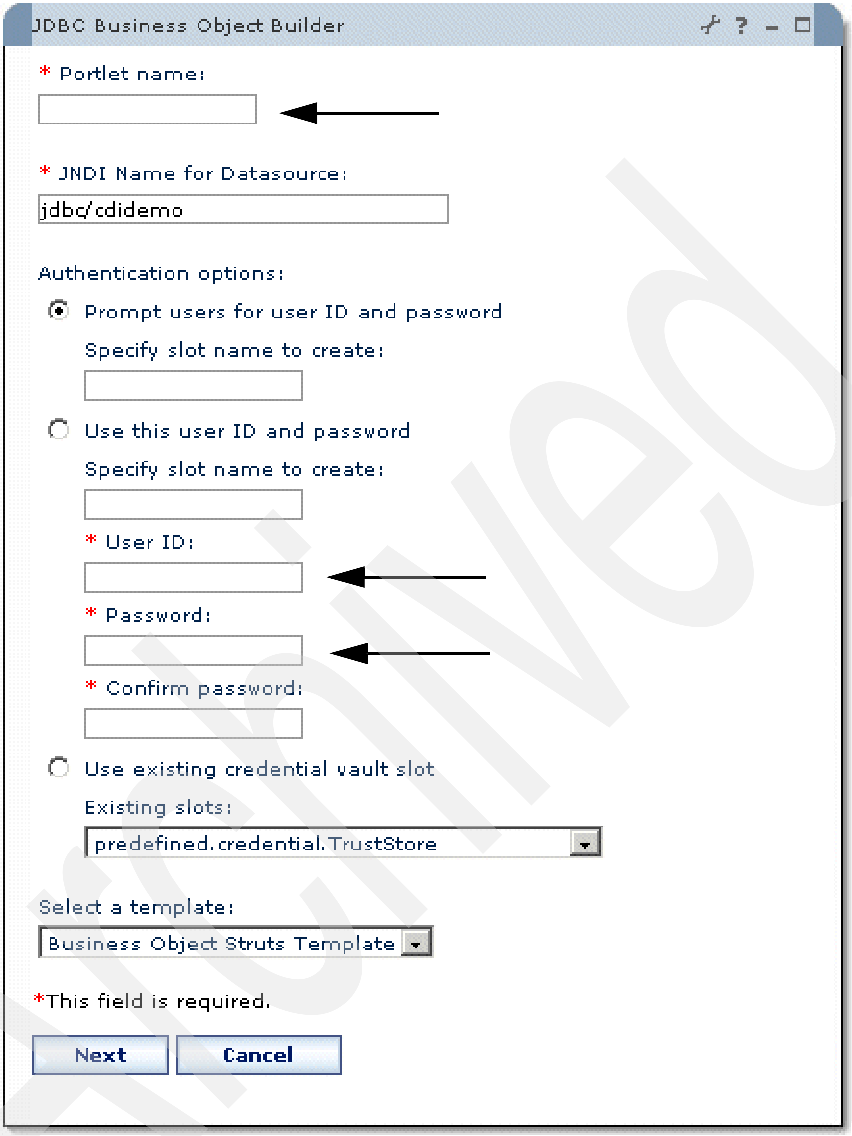


Figure 5-26 Set DB2 II information in the JDBC Business Object Builder portlet

2. Enter the information listed in Table 5-13 into the appropriate fields and click **Next** to view Figure 5-27.

Table 5-13 Update these fields with the information for your DB2 II database

Field	**Update value**
Portlet name	Customer1 Search
Use this user ID and password	<check this checkbox>
User ID	DB2 user id for the DB2 II database
Password	DB2 password for the DB2 II database

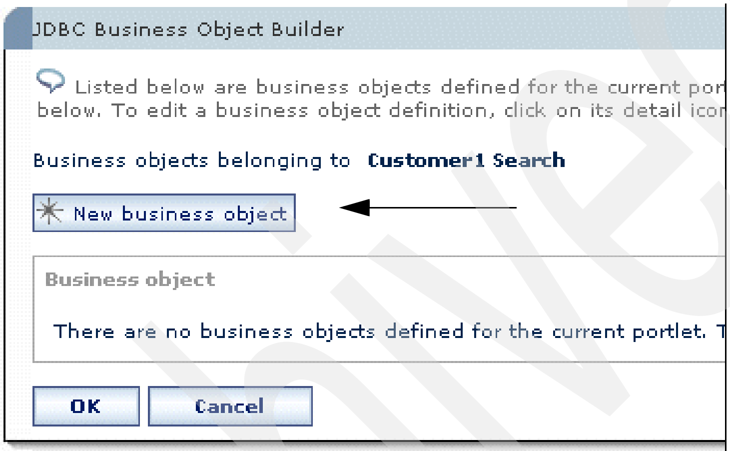


Figure 5-27 Creating a new business object

3. Click ***New business object** to begin working with the schema as shown in Figure 5-28.

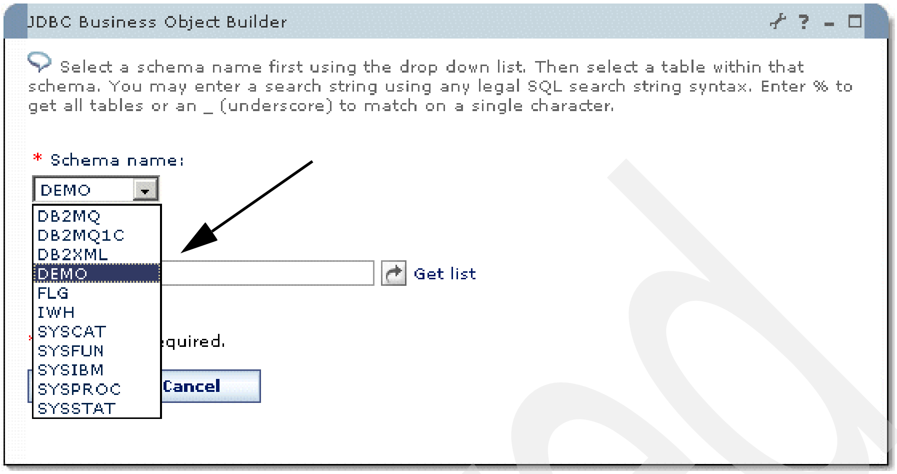


Figure 5-28 Choosing the DB2 II database

4. Type or select the name of the schema that you want to display all of the DB2 objects for (which is **DEMO** in our case).

> **Attention:** This portlet sees the DB2 II federated database as any other DB2 database.

5. Click **Get List** to list all the DEMO.* objects in the **DEMO** database as shown in Figure 5-29.

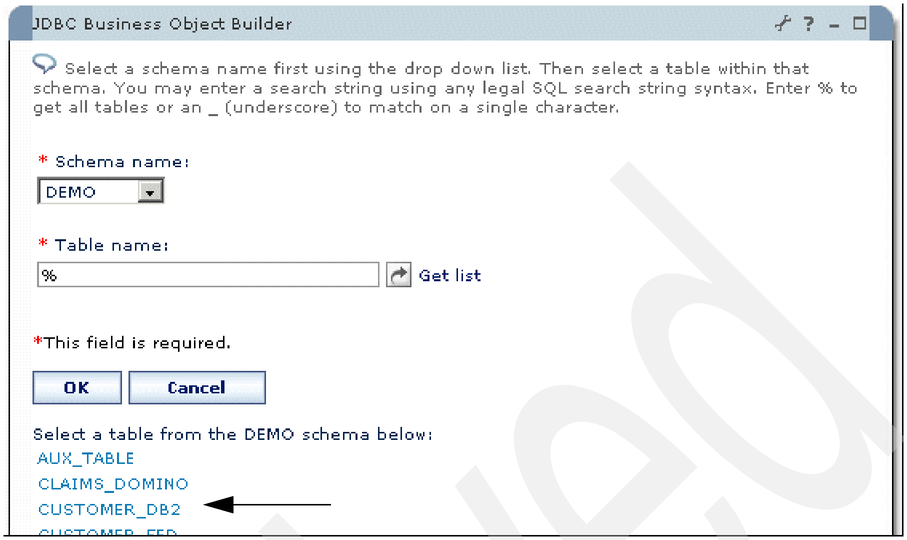


Figure 5-29 CUSTOMER_DB2 is a DB2 II nickname

6. Select **CUSTOMER_DB2** and click **OK** on Figure 5-29 to create a business object as shown in Figure 5-30.

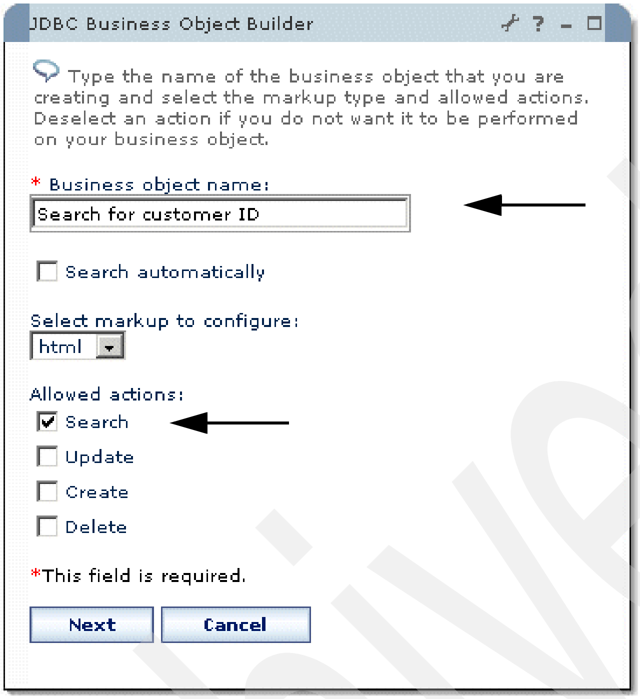


Figure 5-30 Select the allowed actions for this business object

7. Select **Search in Allowed actions** in Figure 5-30, and click **Next** to see all the columns in the table as shown in Figure 5-31.

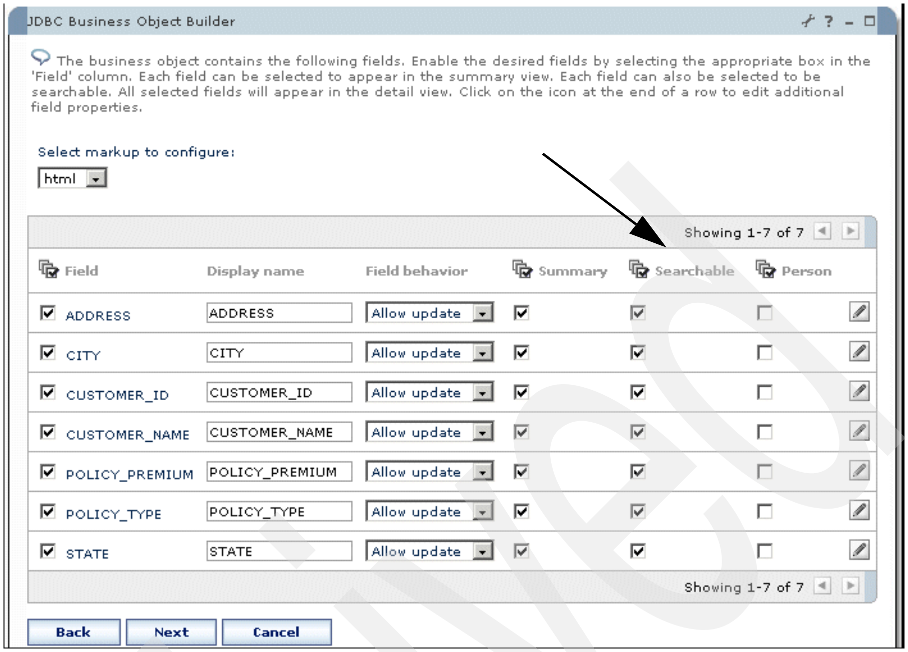


Figure 5-31 Verify that the fields you selected are searchable

8. Verify that the columns you want to search are checked as **Searchable**, and after making the appropriate changes, click **Next** to view Figure 5-32.

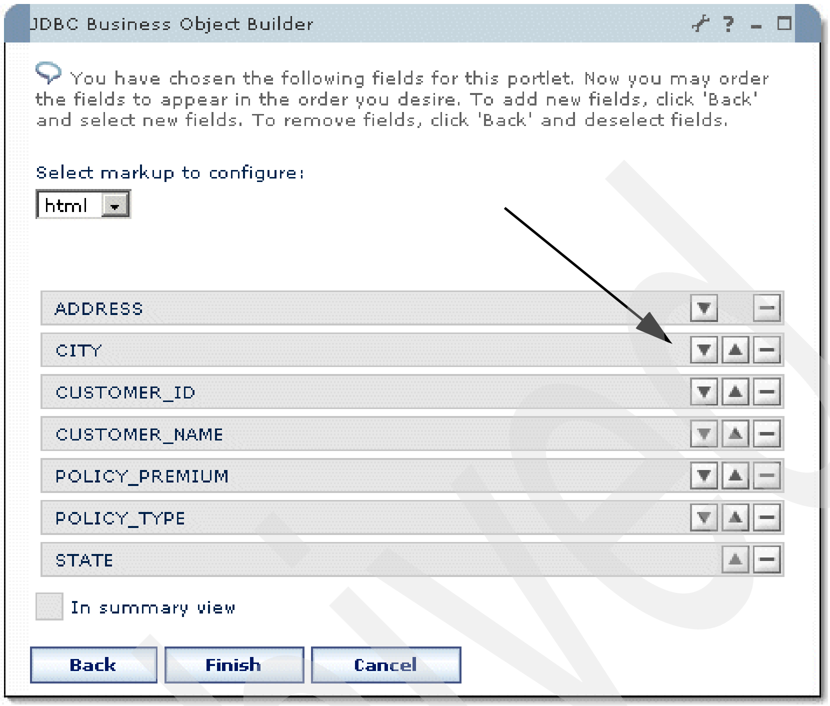


Figure 5-32 You can re-position the fields and select the appropriate markup

9. Rearrange the fields as desired using the arrow controls shown in Figure 5-32. Click **Finish** to create this new business object, which should be shown in the portlet as seen in Figure 5-33.

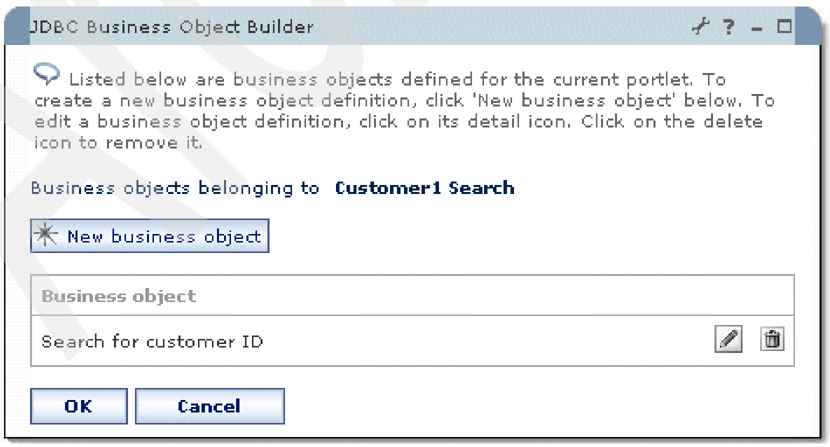


Figure 5-33 The completed JDBC business object

The Customer Search Portlet is now ready to be deployed in your portal. A complete list of portlets is shown in Figure 5-34.

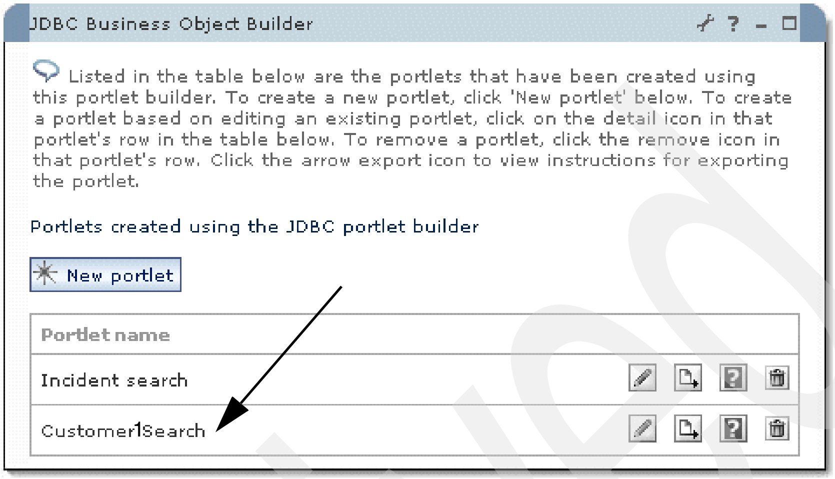


Figure 5-34 List of available portlets built using the JDBC object builder

A

DB2 Information Integrator installation

In this appendix we describe the installation of DB2 Information Integrator Version 8.1 on IBM AIX and Microsoft Windows 2000 platforms.

The topics covered are:

- General prerequisites
- Installing DB2 Information Integrator Version 8.1
- Post-installation steps

A.1 Introduction

This appendix describes the general prerequisites for DB2 Information Integrator (DB2 II) Version 8.1, and covers a typical installation of DB2 Information Integrator Version 8.1 on IBM AIX and Microsoft Windows 2000. Also covered are post-installation tasks such as verifying the installation and creation of a federated database.

A.2 General prerequisites

For a complete list of prerequisites, consult the *IBM DB2 Information Integrator: Installation Guide*, GC18-7036, and the appropriate *IBM DB2 Information Integrator: Release Notes* of the fixpak being installed.

However, the following list of prerequisites is meant to familiarize you with the main products required to successfully install DB2 Information Integrator Version 8.1:

- DB2 Enterprise Server Edition Version V8.1.2 minimum
- Oracle Client. To integrate Oracle data sources
- For relational data sources, you need to install the data source client on the same server as DB2 Information Integrator as follows:
  - Oracle: Install Oracle Client, and create a new entry for the Oracle server in *tnsnames.ora*
  - Sybase: Install Sybase Open Client, and create a new entry for the Sybase server in the *interfaces* file (on UNIX), or *sql.ini* file (on Windows)
  - Informix: Install Informix Client SDK, and create a new entry for the Informix server in the *sqlhosts* file (on UNIX), or SQLHOSTS Registry (on Windows)
  - Microsoft SQL Server:
    - DB2 Information Integrator on Windows: SQL Server ODBC driver on Windows. An entry for the SQL Server needs to be added to the system DSNs in ODBC data source administration.
    - DB2 Information Integrator on UNIX/Linux: Install DataDirect Connect ODBC and create an entry for the SQL Server in the *odbc.ini file.*
  - Teradata
    - DB2 Information Integrator on Windows: Install Teradata Windows CLI Client and create an entry for the Teradata server in the *c:\winnt\system32\drivers\etc\hosts* file.

   - DB2 Information Integrator on AIX: Install Teradata CLIV2 (PIOM and BTEQ are also recommended for testing) and create the entry for the Teradata server in */etc/hosts*.

 – ODBC data sources

   Install an open database connectivity 3.0 driver, and create the entry for the data source in *.odbc.ini* file (UNIX/Linux), or create a System DSN entry for the data source in the ODBC data source administration (on Windows).

- Non relational data sources:
  - Excel 97 or Excel 2000 files
  - Flat file
  - XML

> **Important:** The DB2 II server must have read access to these files. More accurately, the user ID under which DB2 II executes must be able to read these files.

We strongly recommend that you install, configure, and test the connectivity to your federated data sources *prior* to the installation of DB2 II. If the connectivity software is added *after* you install DB2 II, you need to:

- Manually update the environment, with: *db2dj.ini* file (UNIX, Windows).
- Set the environment variables.
- Link the wrapper to the data source client software (UNIX only).

The following requirements need to be met in order to ensure a successful DB2 II installation:

- DB2 II, its components, and the data source client software such as Oracle Client and DataDirect Connect ODBC *must be* installed on the *same server*.
- DB2 II installation procedure needs GUI support.
- The installation procedure needs root authority in Linux and UNIX environments.

The operating systems supported for relational wrappers and non relational wrappers vary, depending on the wrappers installed. Table A-1 lists the current operating systems supported by each IBM DB2 Information Integrator Version 8.1.2 wrapper.

Table A-1 Wrappers and supported operating systems

Wrapper	Windows	AIX	HP-UX	Linux	Solaris
DRDA	32-bit, 64-bit	32-bit, 64-bit	32-bit, 64-bit	32-bit, 64-bit	32-bit, 64-bit
Informix	32-bit	32-bit, 64-bit	32-bit, 64-bit	32-bit	32-bit, 64-bit
Microsoft SQL Server	32-bit	32-bit	32-bit	32-bit	32-bit
ODBC	32-bit	32-bit	32-bit	32-bit	32-bit
OLE DB	32-bit				
Oracle NET8	32-bit	32-bit, 64-bit	32-bit, 64-bit	32-bit	32-bit, 64-bit
Oracle SQLNET	32-bit	32-bit			
Sybase CTLIB	32-bit	32-bit, 64-bit	32-bit, 64-bit	32-bit	32-bit, 64-bit
Sybase DBLIB	32-bit	32-bit	32-bit	32-bit	32-bit
Teradata	32-bit	32-bit			
BLAST	32-bit	32-bit		32-bit	32-bit
BioRS	32-bit	32-bit			
Documentum	32-bit	32-bit			32-bit
Entrez	32-bit	32-bit			
HMMER	32-bit	32-bit			
IBM Lotus Extended Search	32-bit	32-bit			
Microsoft Excel	32-bit				
Table-structured files	32-bit	32-bit, 64-bit	32-bit, 64-bit	32-bit	32-bit, 64-bit
XML	32-bit	32-bit	32-bit	32-bit	32-bit

There are restrictions related to the possible combinations of 32-bit and 64-bit DB2 ESE 8.1, DB2 II wrappers, and the data source client software. Refer to the *IBM DB2 Information Integrator: Installation Guide*, GC18-7036, and the appropriate *IBM DB2 Information Integrator: Release Notes* of the fixpak being installed for further details.

A.3 Installing DB2 Information Integrator Version 8.1

This section describes the following DB2 II Version 8.1 installation on the following platforms:

- Installing on AIX
- Installing on Windows

> **Note:** We created a user called `ADMINISTRATOR` on Windows for the DB2 system, and used it for the DB2 II installation and configuration.

DB2 II installs DB2 Enterprise Server Edition[1], relational wrappers and non-relational wrappers. After the wrappers are installed, you need to set up a federated server and database to configure and access the data sources.

If you add DB2 II relational and non-relational wrappers to an existing DB2 ESE that is at fixpak 3 or higher, then you need to re-apply the DB2 fixpak *after* the DB2 II wrappers are installed in order to apply the updates of the DB2 fixpak to the wrappers:

- On UNIX (AIX, Solaris, HP-UX, and Linux), if you install the relational wrappers for Oracle, Sybase, Microsoft SQL Server, or Teradata, and apply the DB2 fixpak to update the wrappers, you must also run the appropriate **`djxlink`** scripts to update wrapper libraries to the DB2 fixpak level.

  On UNIX, the wrappers are linked to the data source client software — the DB2 fixpak provides an input library for the link, but does not perform the link itself. The **`djxlink`** scripts links the new input library provided by the DB2 fixpak to create a wrapper library that is linked to the data source client at the DB2 fixpak level.

> **Attention:** If **`djxlink`** is not executed after applying the DB2 fixpak, then DB2 II configuration and access to data sources may be unpredictable.

A.3.1 Installing on AIX

The following installation was done on an AIX model P270 with 1 GB memory, 36 GB of disk space and operating system Version 5.1.

[1] If DB2 Enterprise Server Edition or DB2 Connect™ Enterprise Edition is already installed at the correct level, reinstall is not required. The installation wizard within DB2 Information Integrator detects previous installations.

> **Attention:** Ensure that all the prerequisites are in place prior to commencing the installation.
>
> We strongly recommend that you consult the DB2 II release notes, and other installation documentation for a successful DB2 II installation.

The following steps install DB2 II on AIX:

1. Open an AIX terminal window with root authority.
2. **`cd /cdrom`**

   Set the DISPLAY environment variable as follows:

   ```
   Export DISPLAY=<HOSTNAME>:0.0
   ```

   where <HOSTNAME> is the hostname of the machine where you want the screens to be displayed.
3. Execute the following command:

   ```
   ./iiSetup.bin
   ```

   This displays the dialog shown in Figure A-1.

Figure A-1 DB2 Information Integrator Installation - Launch screen

Click **Install Products** to launch the Software License Agreement dialog box shown in Figure A-2.

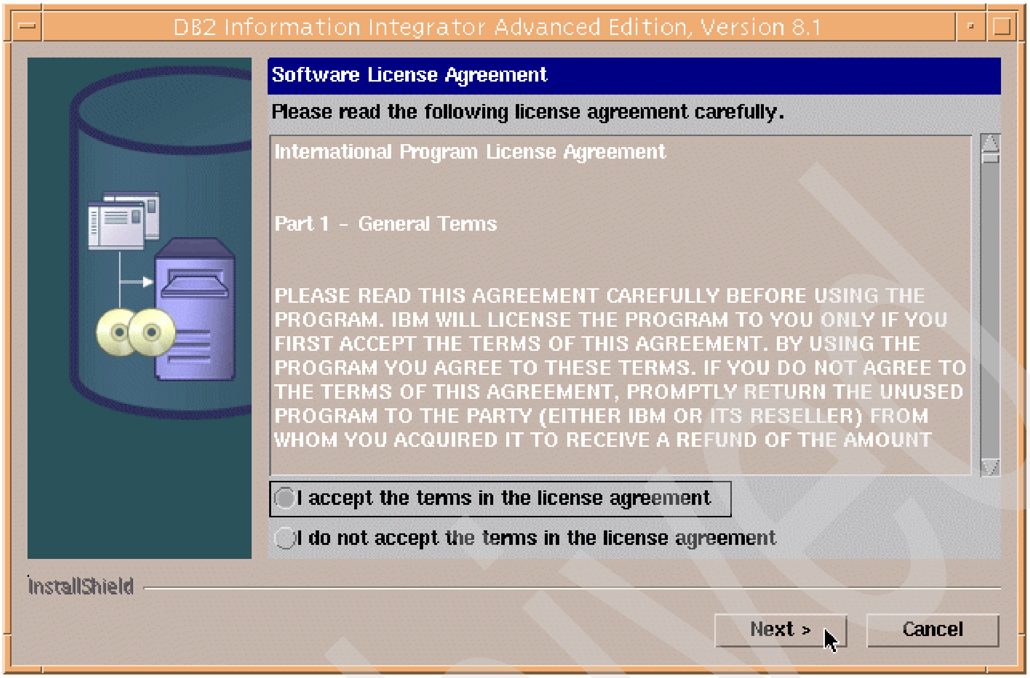


Figure A-2 SW license agreement

Please read the license agreement before you proceed.

Select **I accept the terms in the license agreement** and click **Next** to view the screen shown in Figure A-3.

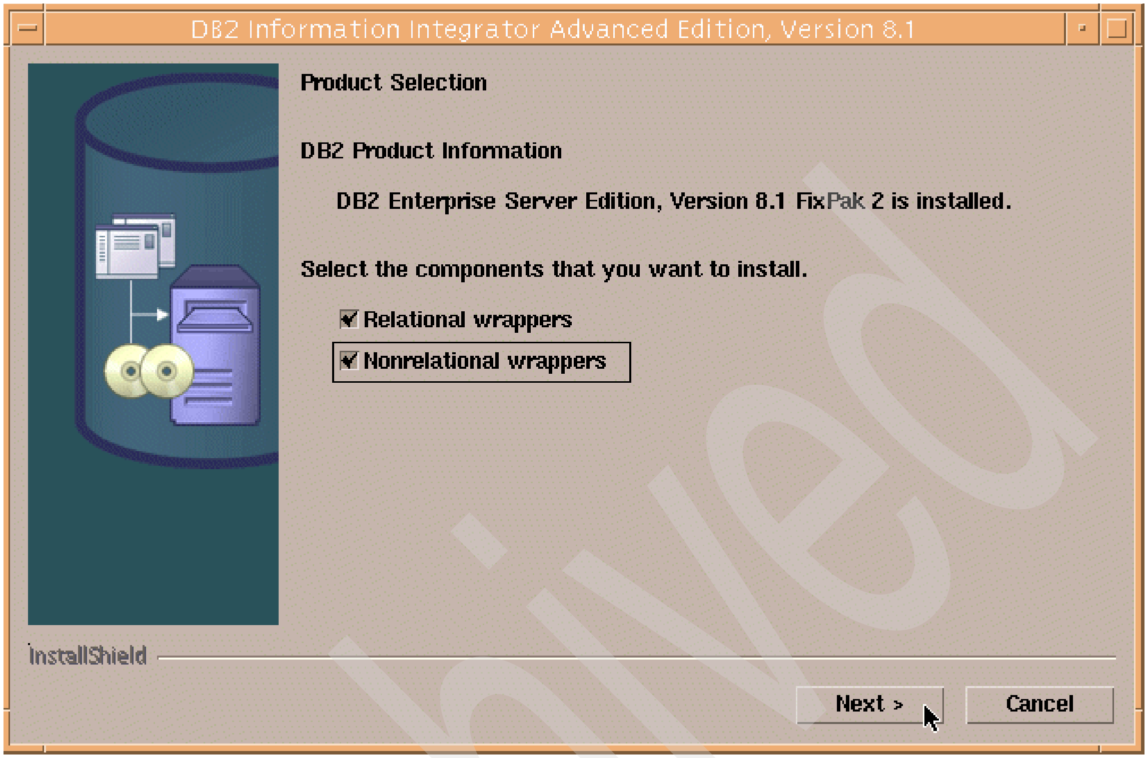


Figure A-3 Product selection - Relational, non relational wrappers

Figure A-3 provides the choice of components to install as follows:

- Relational wrappers include libraries to access relational data sources like Teradata, Sybase, Microsoft SQL Server, and Oracle.
- Non relational wrappers include libraries to access structured files (Excel, Table-structured files, XML), application data sources (BioRS, Documentum, Entrez, IBM Lotus Extended Search) and scientific related data (BLAST, HMMER).

Select both types of wrappers, and click **Next** view the screen shown in Figure A-4.

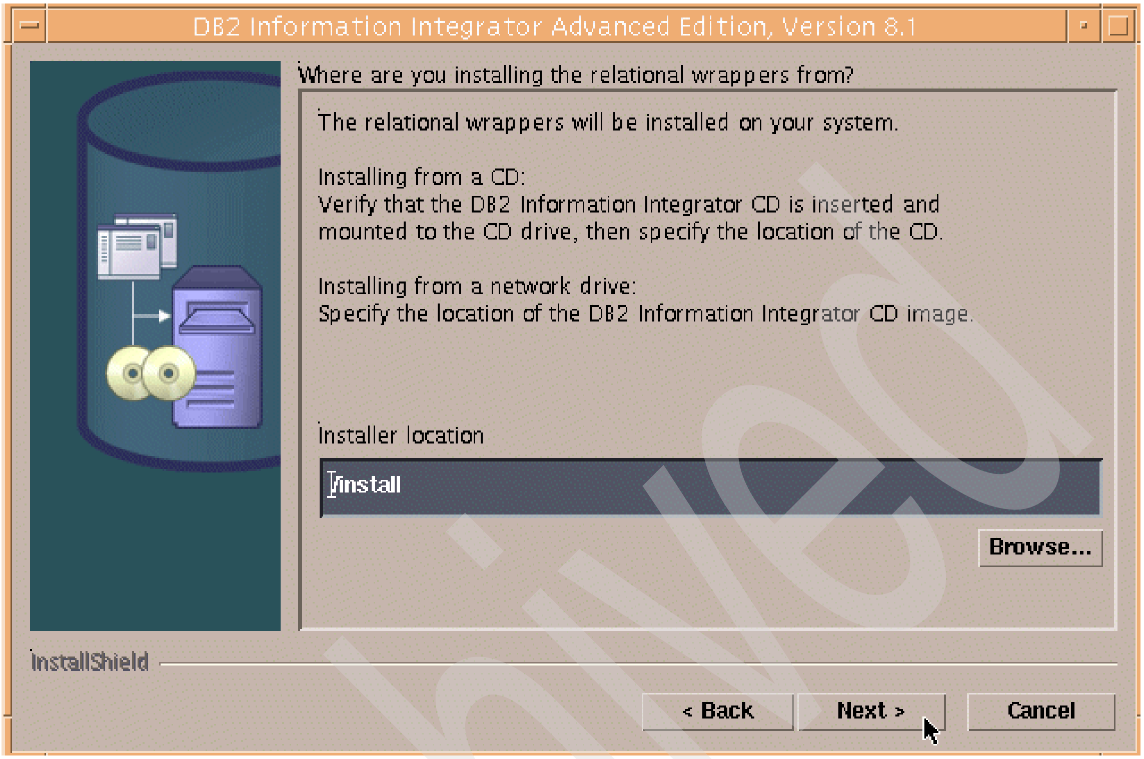


Figure A-4 Source of relational wrapper installation

Figure A-4 requests the location of the relational wrapper code.

Browse the CD, or any other file system to select the path for the installation files of the relational wrappers as shown in Figure A-4. Click **Next** to start the relational wrappers installer as shown in Figure A-5.

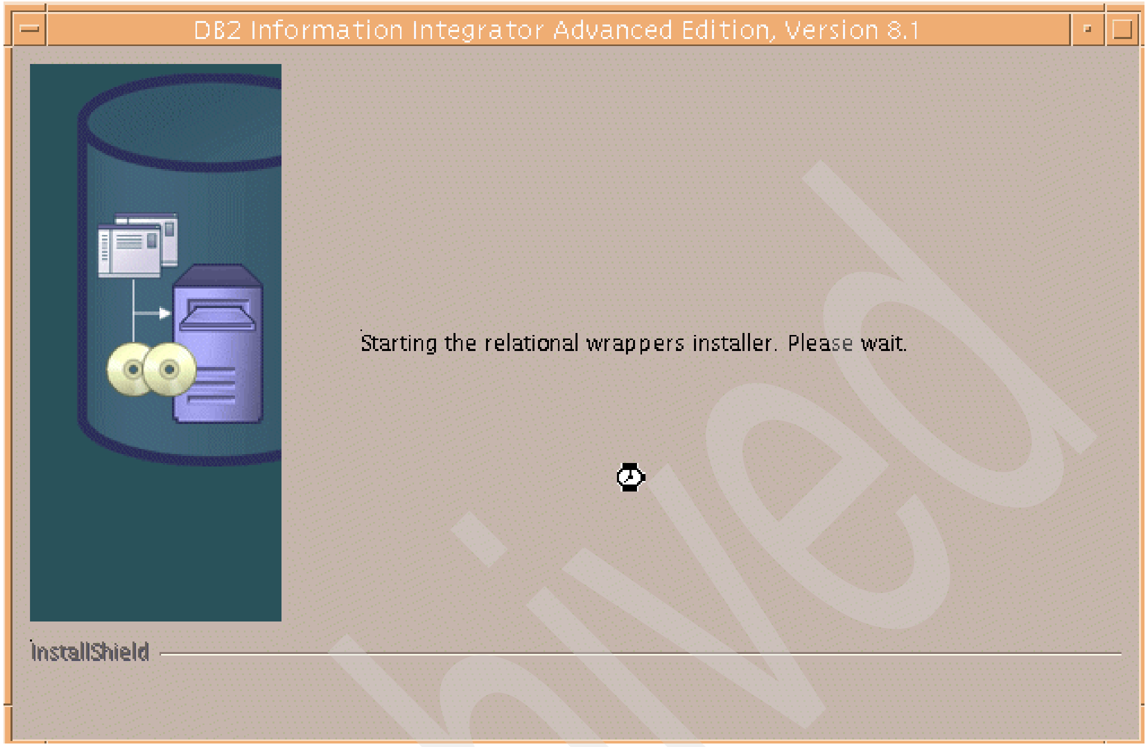


Figure A-5 Start the relational wrappers installer

After the relational wrapper installer finishes loading, it opens the welcome screen for the relational wrapper installer as shown in Figure A-6.

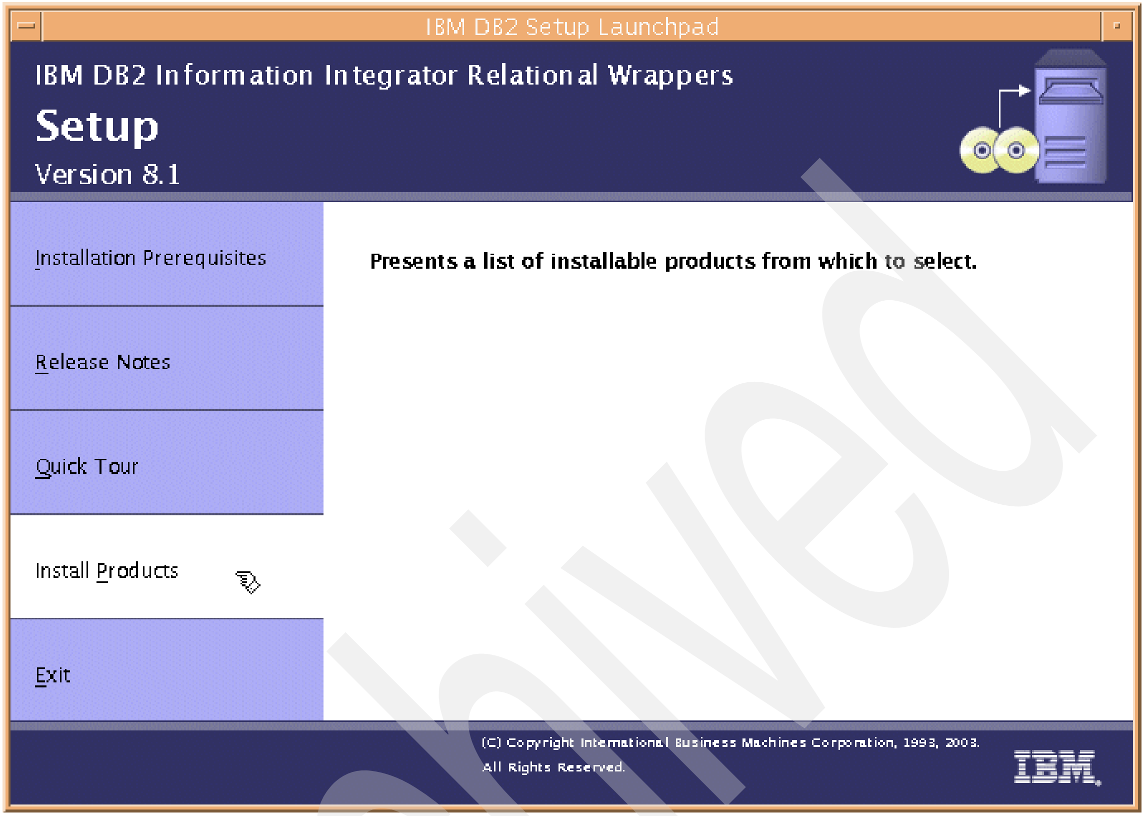


Figure A-6 Relational wrappers setup

Figure A-6 provides options for reviewing the most current information on installation prerequisites and release notes for the relational wrappers installation.

> **Note:** We recommend that you browse through these documents to ensure that all prerequisites are in place.

Click **Install Products** to select the product to install as shown in Figure A-7.

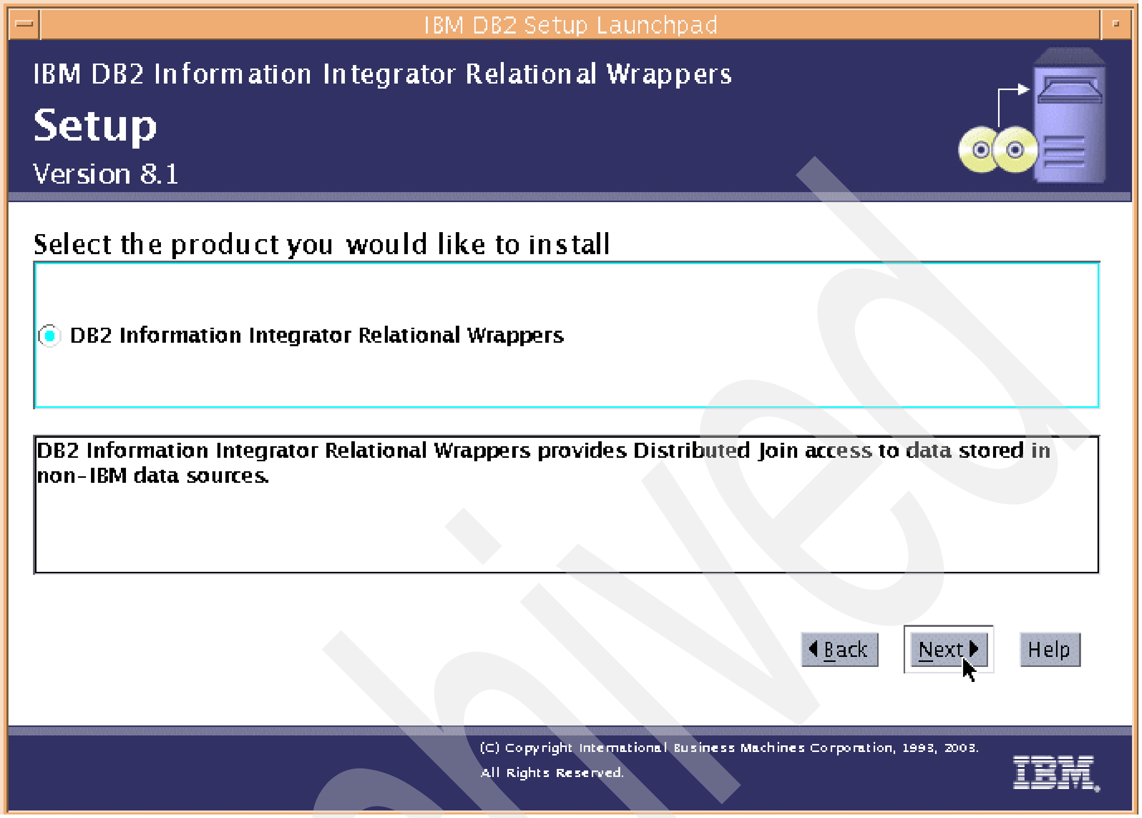


Figure A-7 Relational wrappers setup - Select products

Select the only option: **DB2 Information Integrator Relational Wrappers** and click **Next** to continue.

> **Note:** The screen might flash a few times and then a DB2 installation Splash screen is displayed as shown in Figure A-8.

Figure A-8 Relational wrappers setup - Introduction

Click **Next** to display the Software License Agreement page shown in Figure A-9.

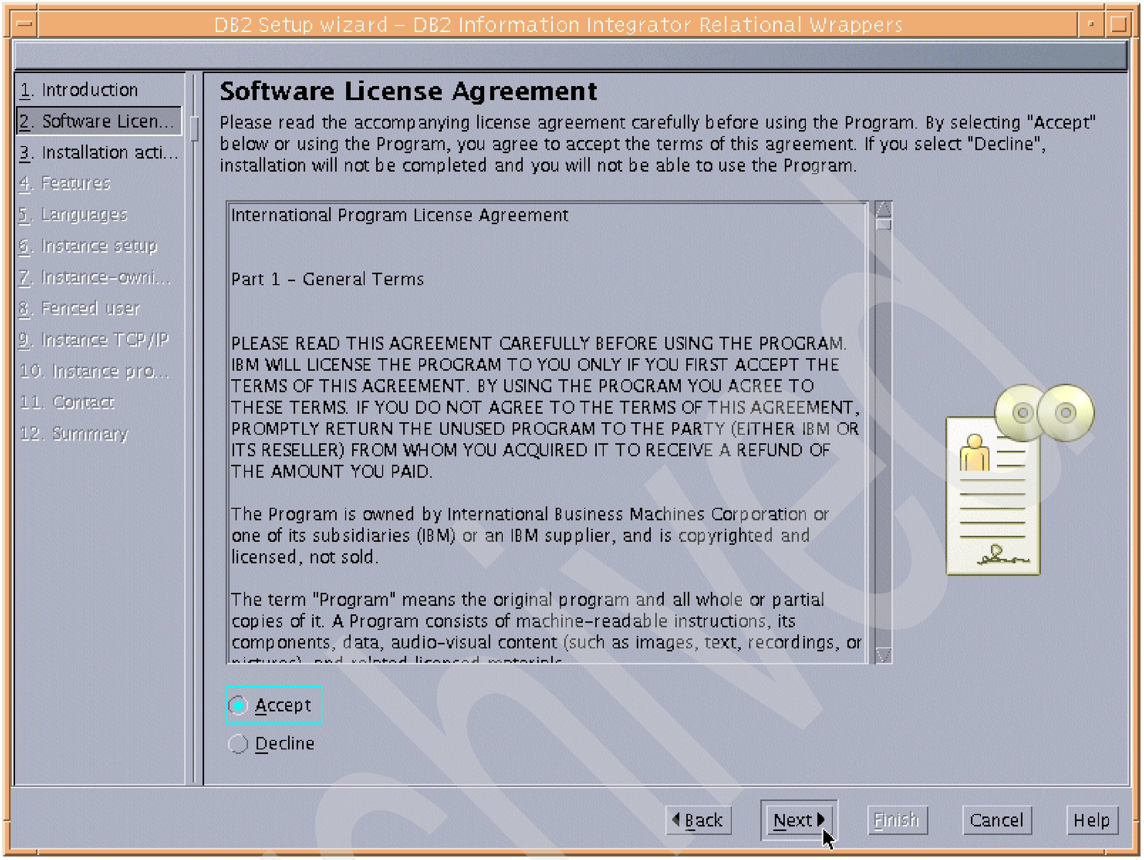


Figure A-9 Relational wrappers setup - SW License Agreement

Please read the license agreement before you proceed.

Select **Accept** and click **Next** to view the screen shown in Figure A-10.

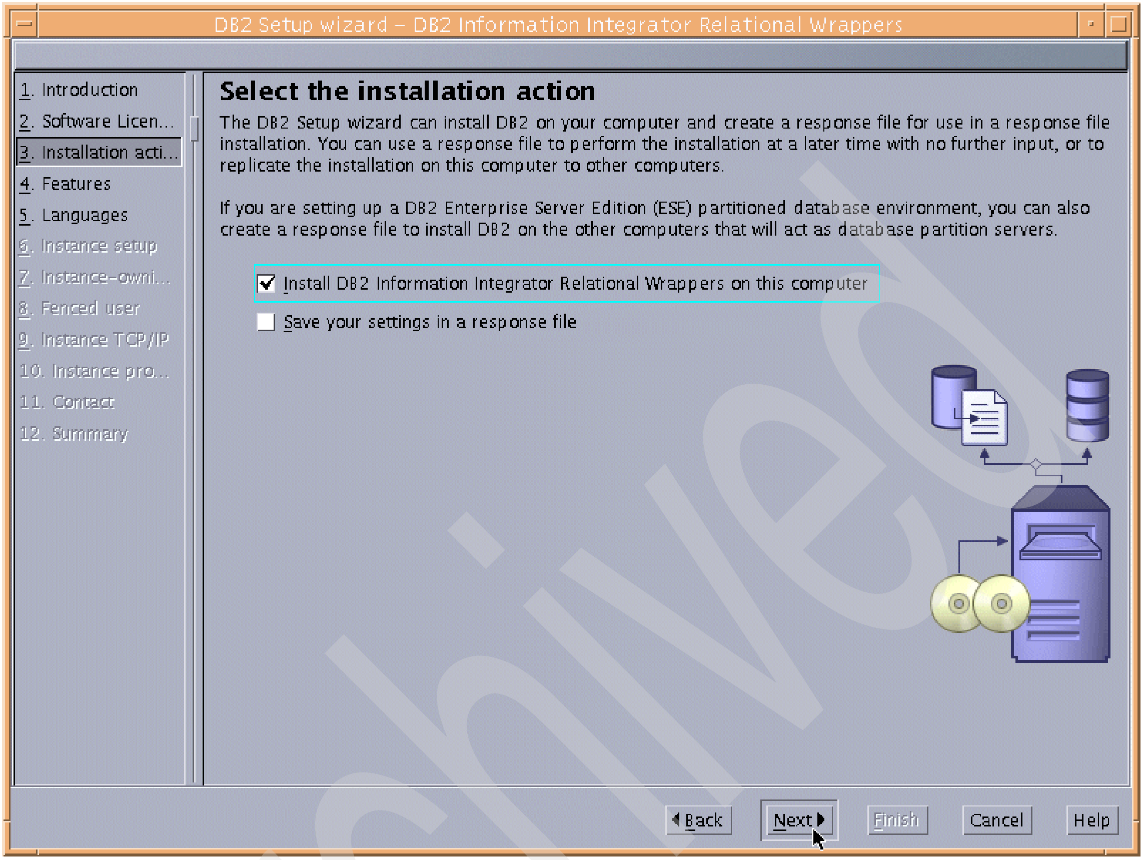


Figure A-10 Relational wrappers setup - Select installation action

Select **Install DB2 Information Integrator Relational Wrappers on this computer**. We chose not to select the **Save your settings in a response file** option, which provides for recording the interactions with the installation process in a text file.

> **Tip:** A silent DB2 II installation can be performed by supplying a response file as an argument for `iisetup.bin`.

Click **Next** to view the screen shown in Figure A-11.

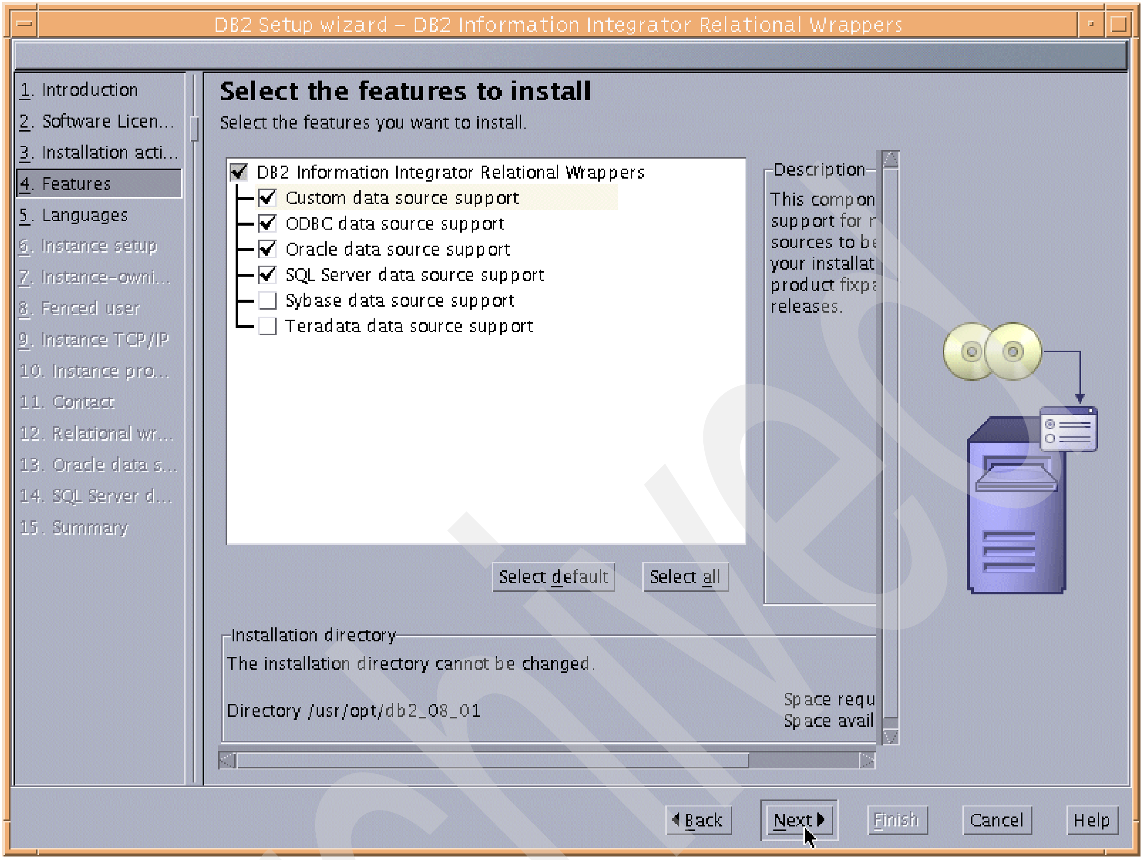


Figure A-11 Relational wrappers setup - Features to install

Figure A-11 provides the option of selecting the specific types of wrappers you want installed.

Select all the data sources you want to connect to from DB2 II. If you are not sure, just install all the wrappers to avoid having to install one later.

> **Attention:** You need to choose **custom data source support** to enable the possibility of adding new wrappers included in future fixpak releases.

Click **Next** to view the screen shown in Figure A-12.

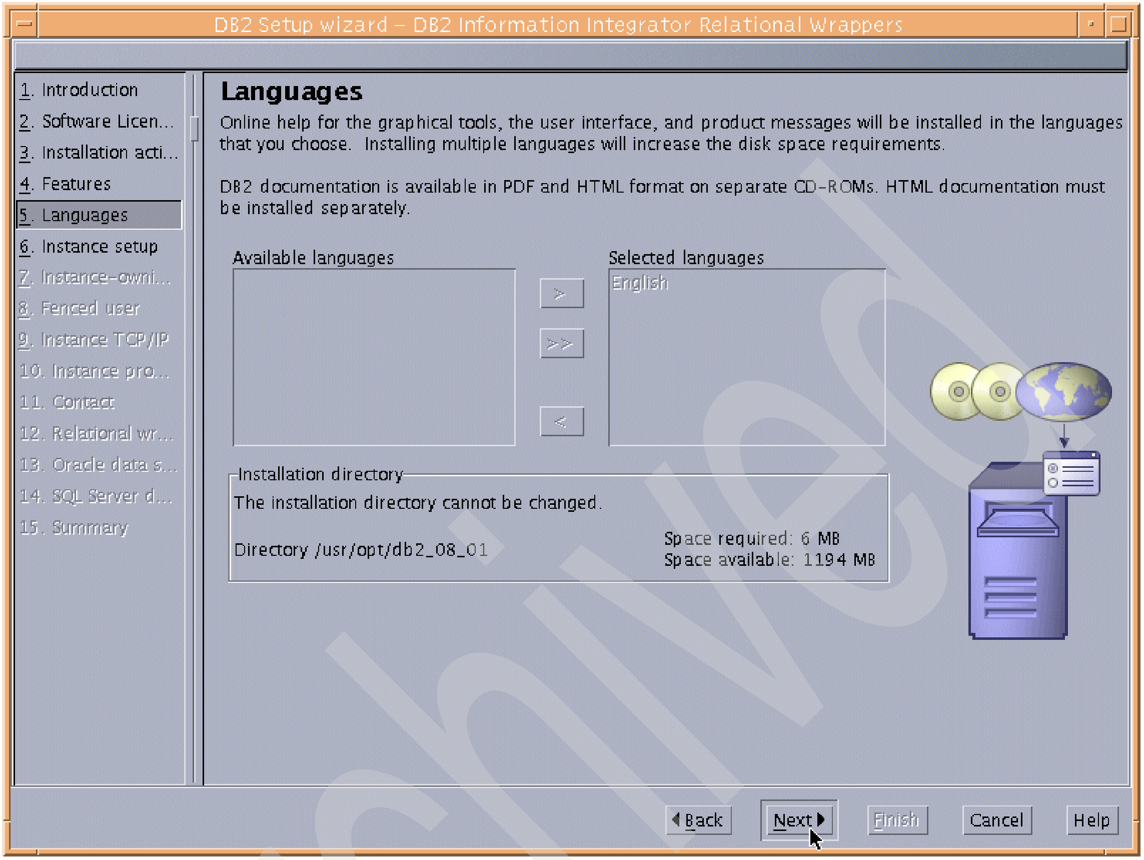


Figure A-12 Relational wrappers setup - Languages

Figure A-12 provides the option of selecting the languages to be supported by DB2 II.

Add your desired languages from the **Available languages** list and then click **Next** to view the screen shown in Figure A-13.

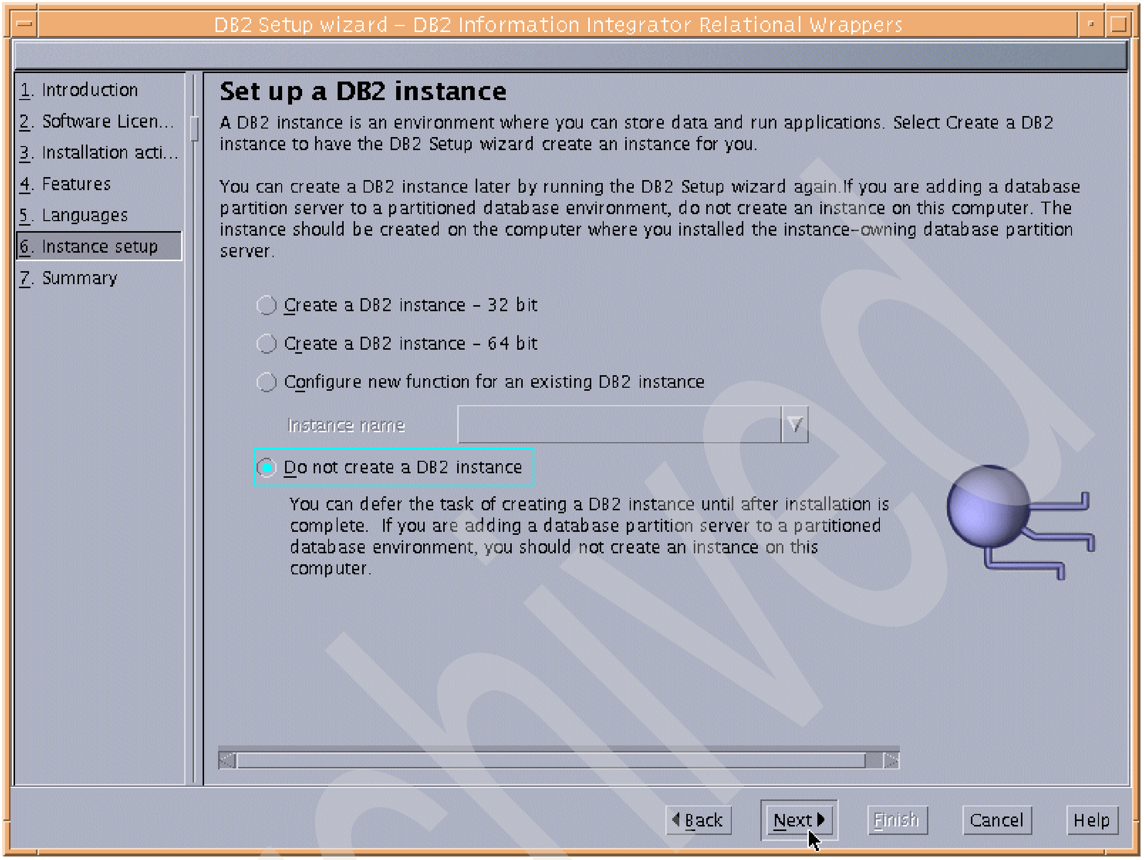


Figure A-13 Relational wrappers setup - Setup DB2 instance

Figure A-13 provides the option of having the DB2 Setup wizard create a DB2 instance.

Select **Do not create a DB2 instance** and then click **Next** to display the screen shown in Figure A-14.

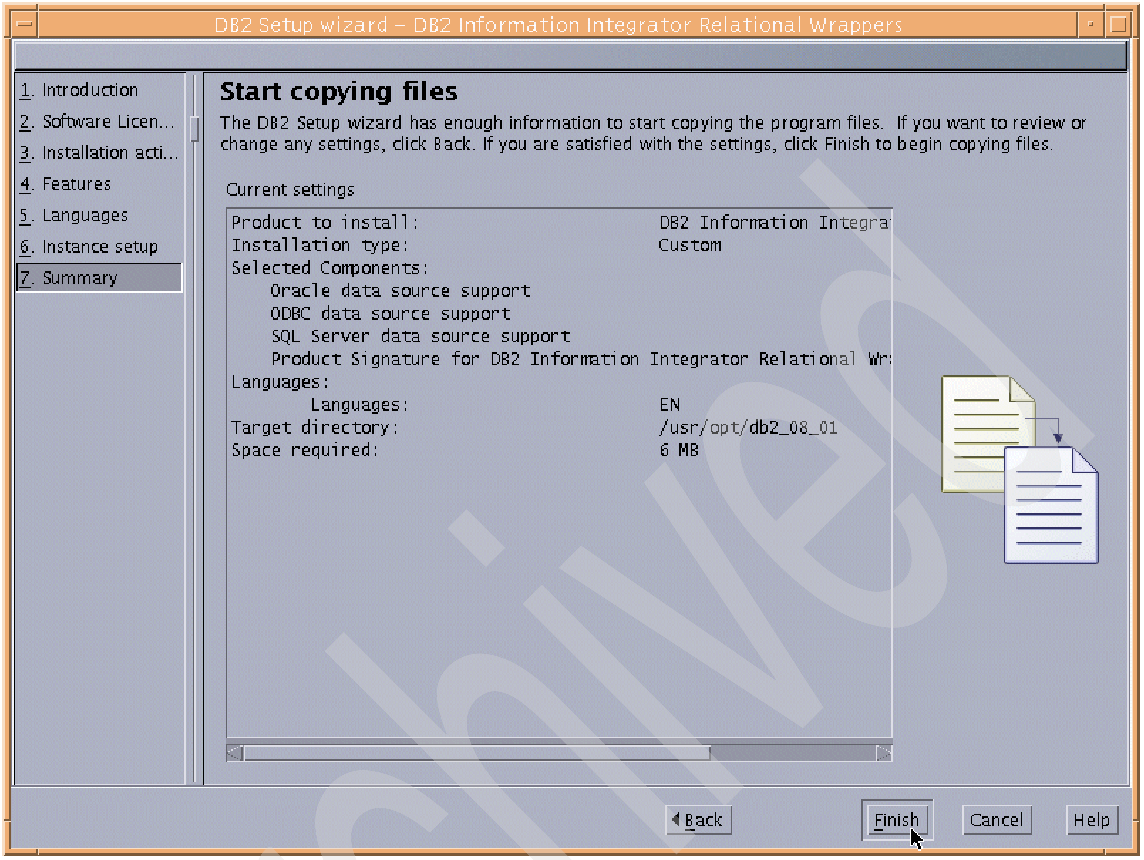


Figure A-14 Relational wrappers setup - Summary

Figure A-14 summarizes the selections made. You can click the **Back** button to change any of the selections.

Click **Finish** to proceed with the installation as shown in Figure A-15.

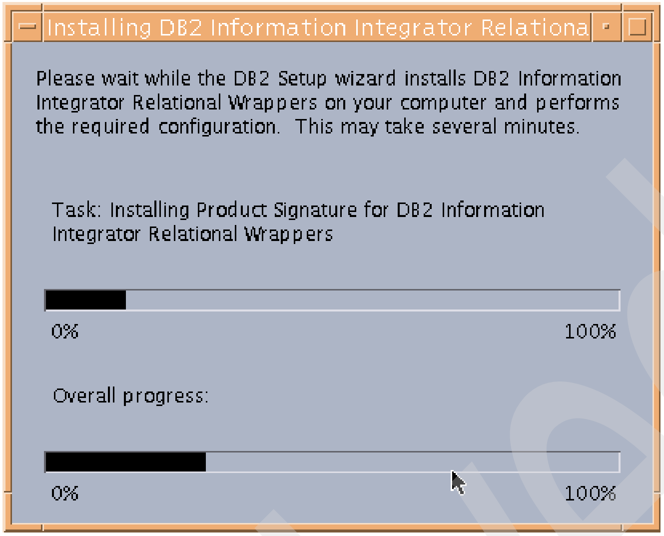


Figure A-15 Relational wrappers setup - Installation progress

After a few minutes, the status of the install is displayed on **the Setup is complete** screen as shown in Figure A-16.

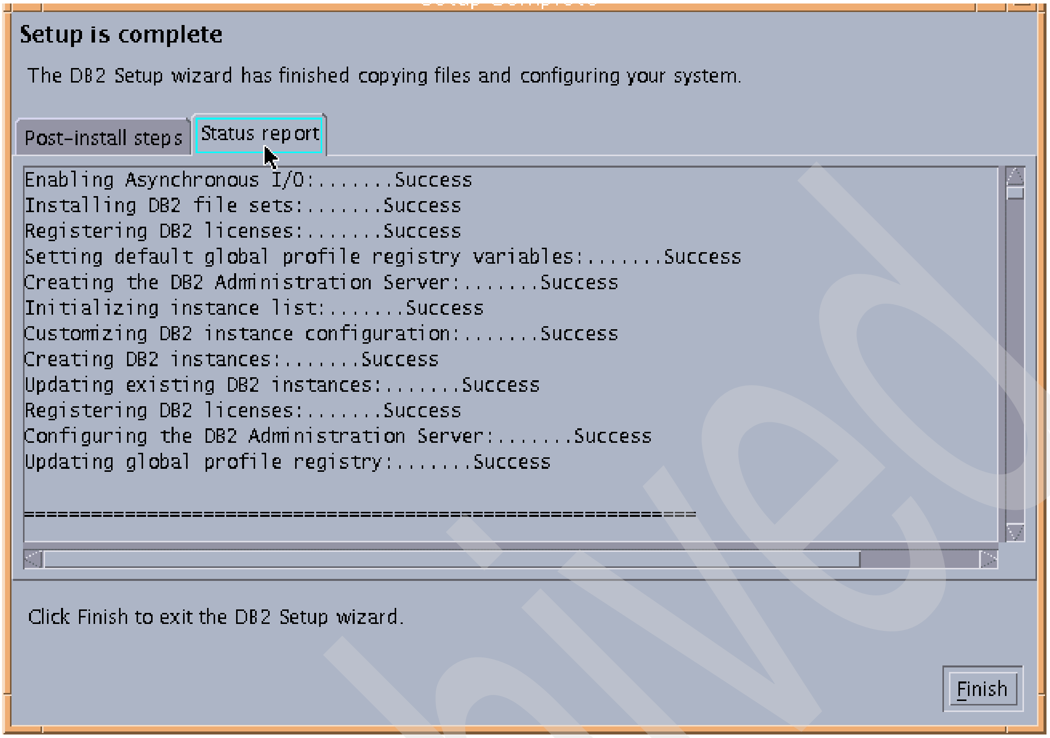


Figure A-16 Relational wrappers setup - Status report tab

Click **Finish** to proceed to the relational wrappers status screen shown in Figure A-17. i

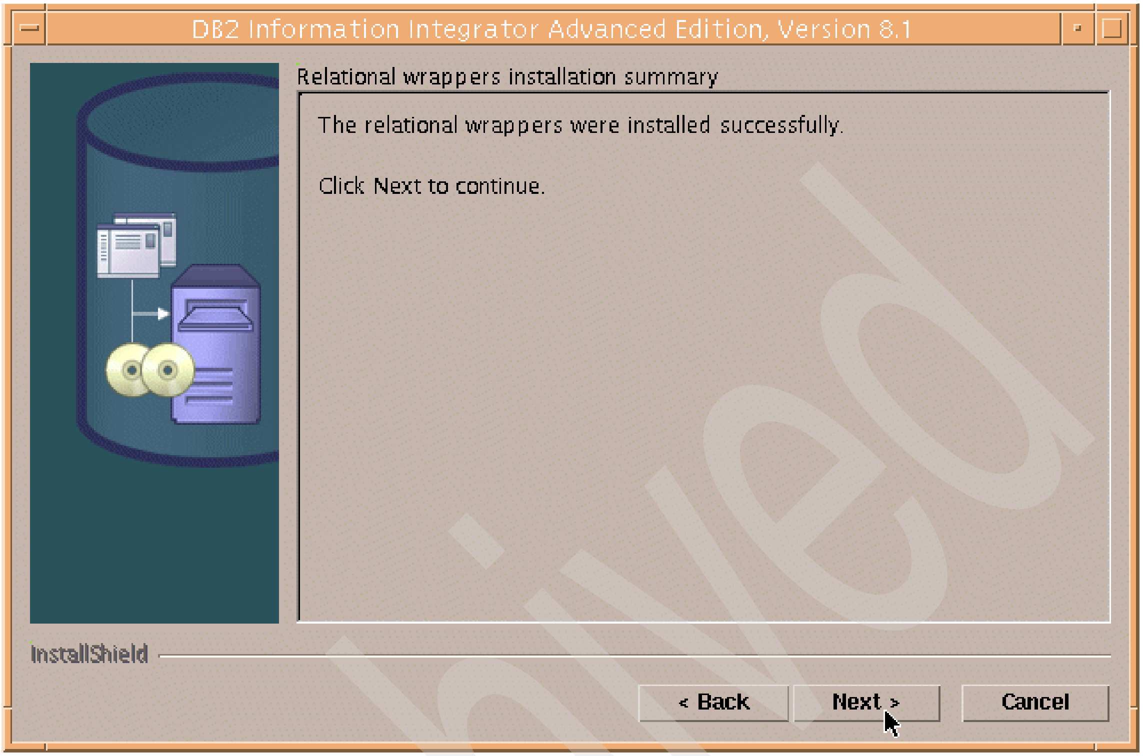


Figure A-17 Relational wrappers setup- Status

Figure A-17 indicates that the relational wrappers were installed successfully.

Click **Next** to continue with the installation of the nonrelational wrappers as shown in Figure A-18.

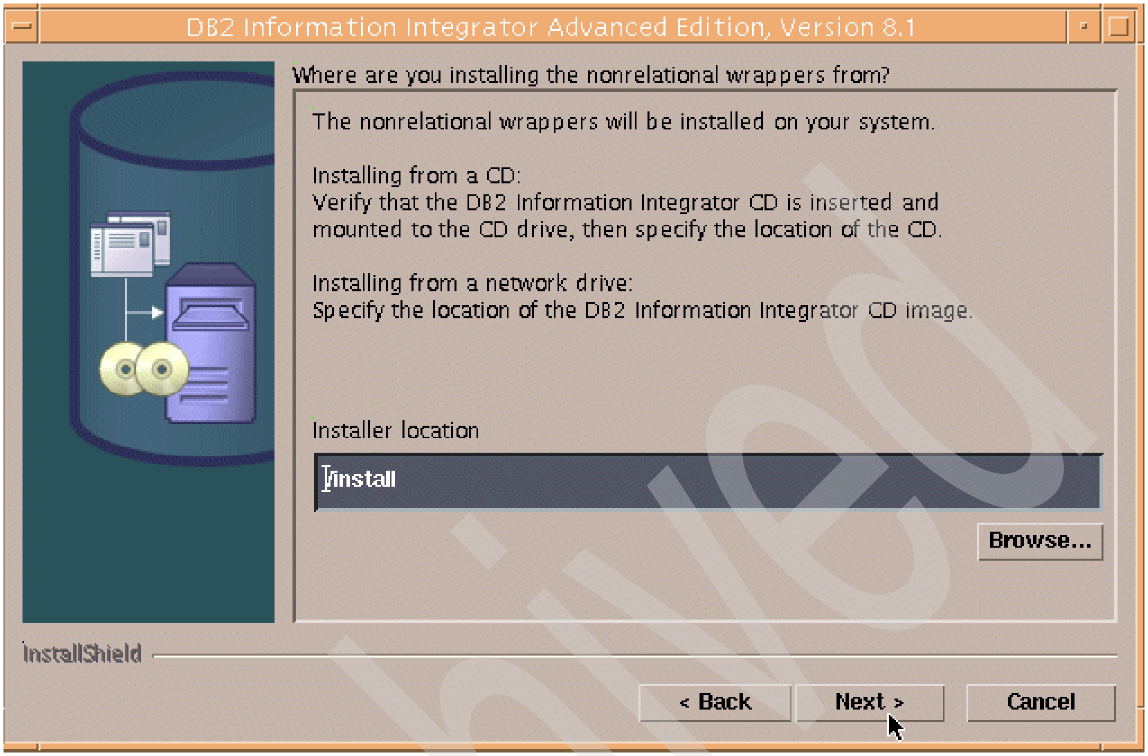


Figure A-18 Source of non relational wrapper installation

Figure A-18 requests the location of the nonrelational wrapper code.

Browse the CD or any other file system to select the path for the nonrelational wrappers installation source, and click **Next** to begin loading the nonrelational wrappers installer as shown in Figure A-19.

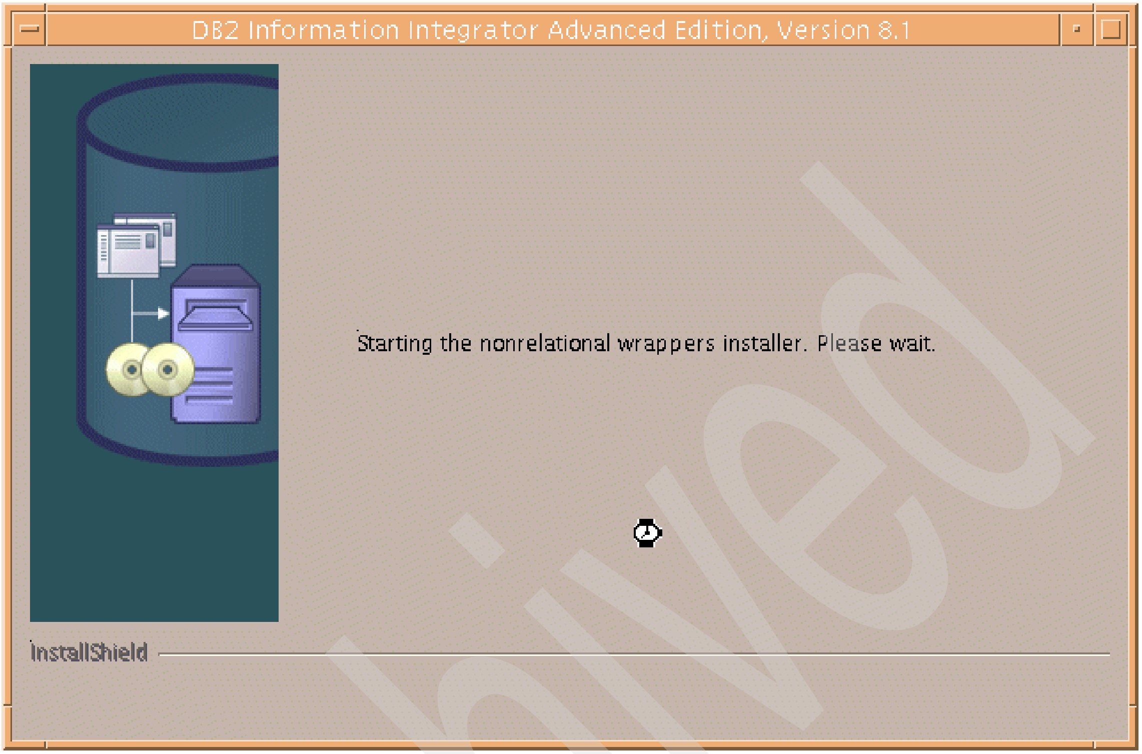


Figure A-19   Start the non relational wrappers installer

After the nonrelational wrappers installer finishes loading it launches the **IBM DB2 Setup Launchpad** screen for the nonrelational wrappers as shown in Figure A-20.

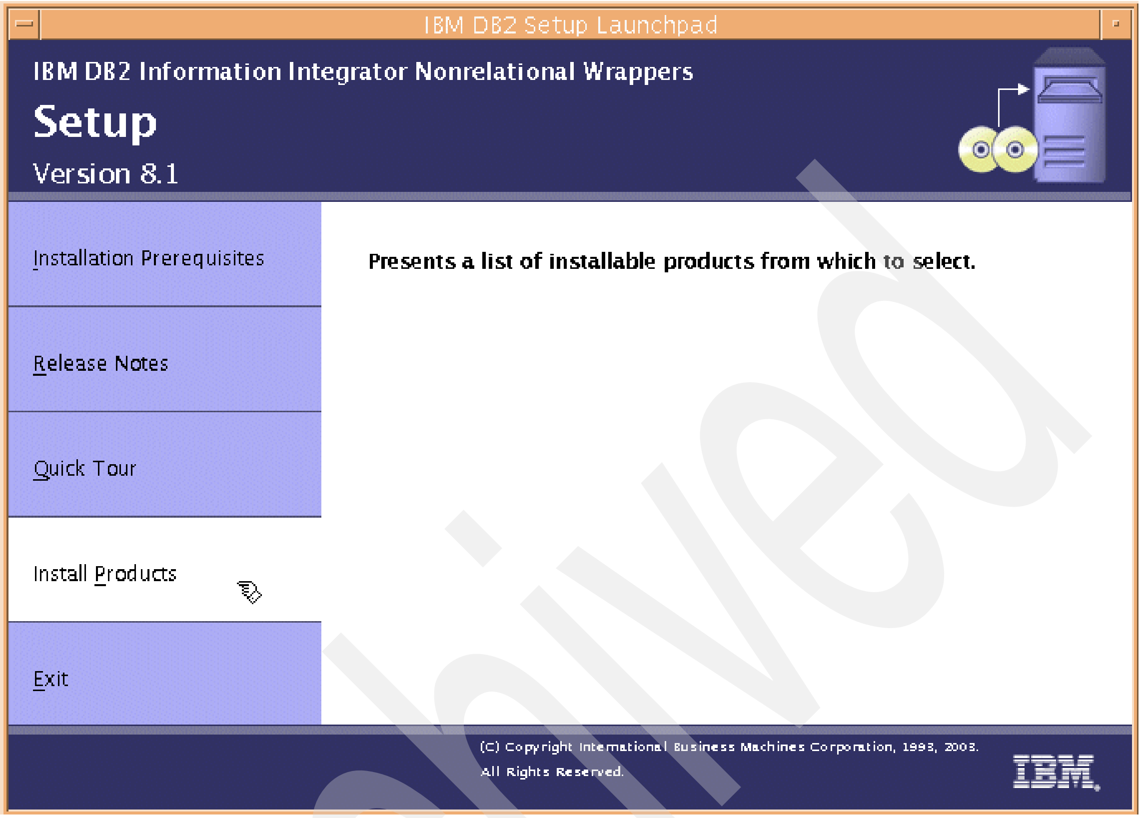


Figure A-20 Non relational wrappers setup

Figure A-20 provides options for reviewing the most current information on installation prerequisites and release notes for the nonrelational wrappers installation.

> **Note:** We recommend that you browse through these documents to ensure that all prerequisites are in place.

Click **Install Products** to select the product to install as shown in Figure A-21.

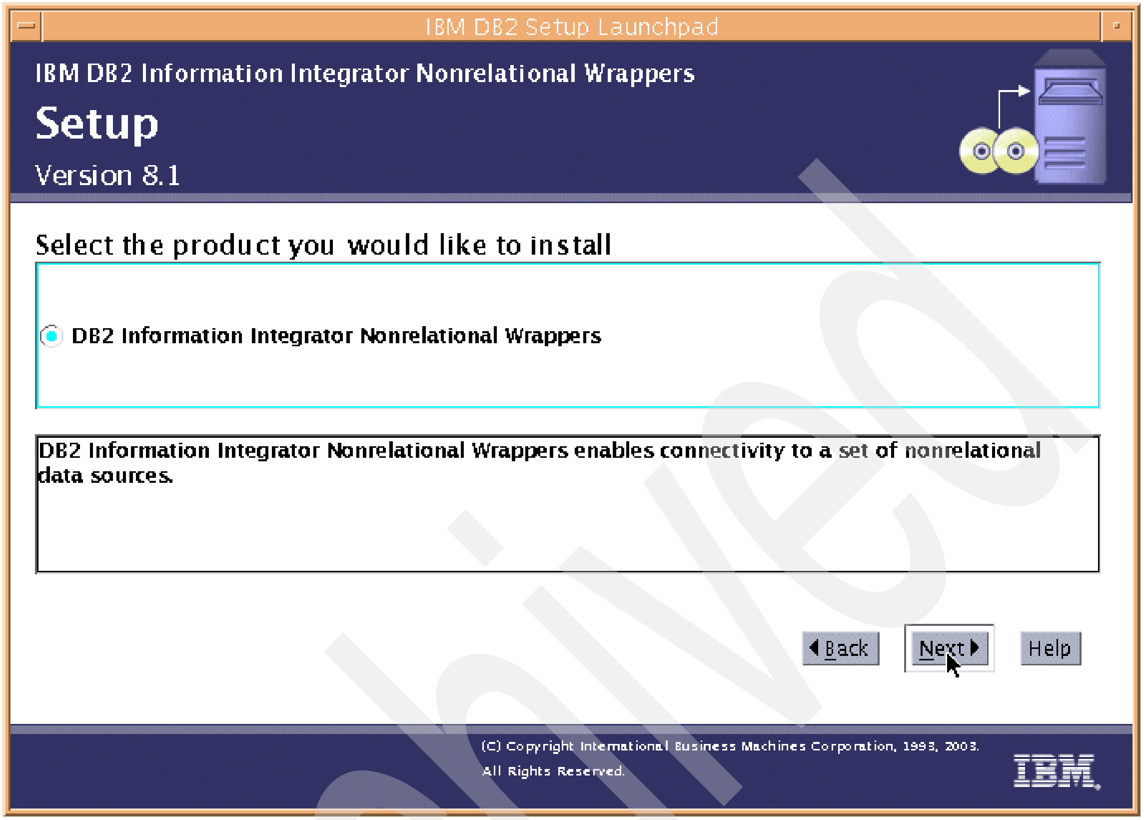


Figure A-21 Non relational wrappers setup - Select products

Select the only option: **DB2 Information Integrator Nonrelational Wrappers** and click **Next** to continue.

> **Note:** The screen might flash a few times and then a DB2 installation Splash screen is displayed as shown in Figure A-22.

Figure A-22 Non relational wrappers setup - Introduction

Click **Next** to display the **Software License Agreement** page as shown in Figure A-23.

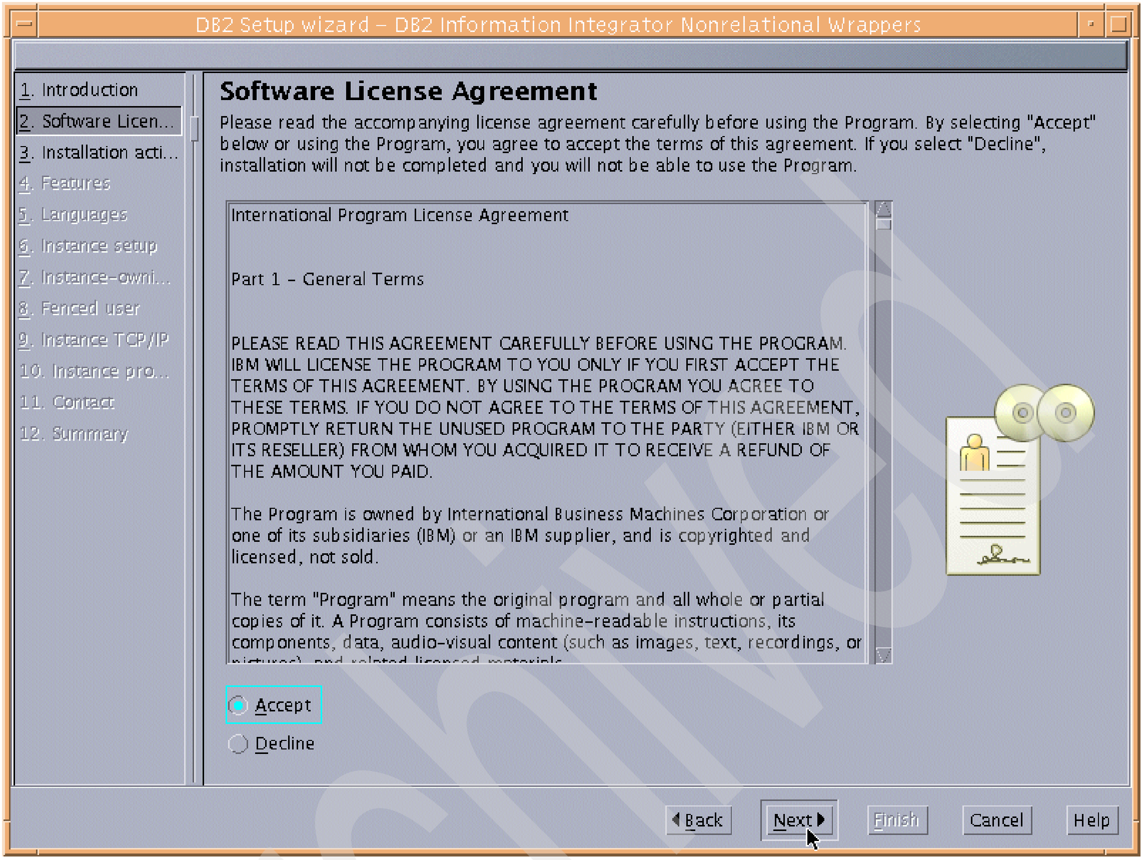


Figure A-23 Non relational wrappers setup - SW License Agreement

Please read the license agreement before you proceed.

Select **Accept** and click **Next** to view the screen shown in Figure A-24.

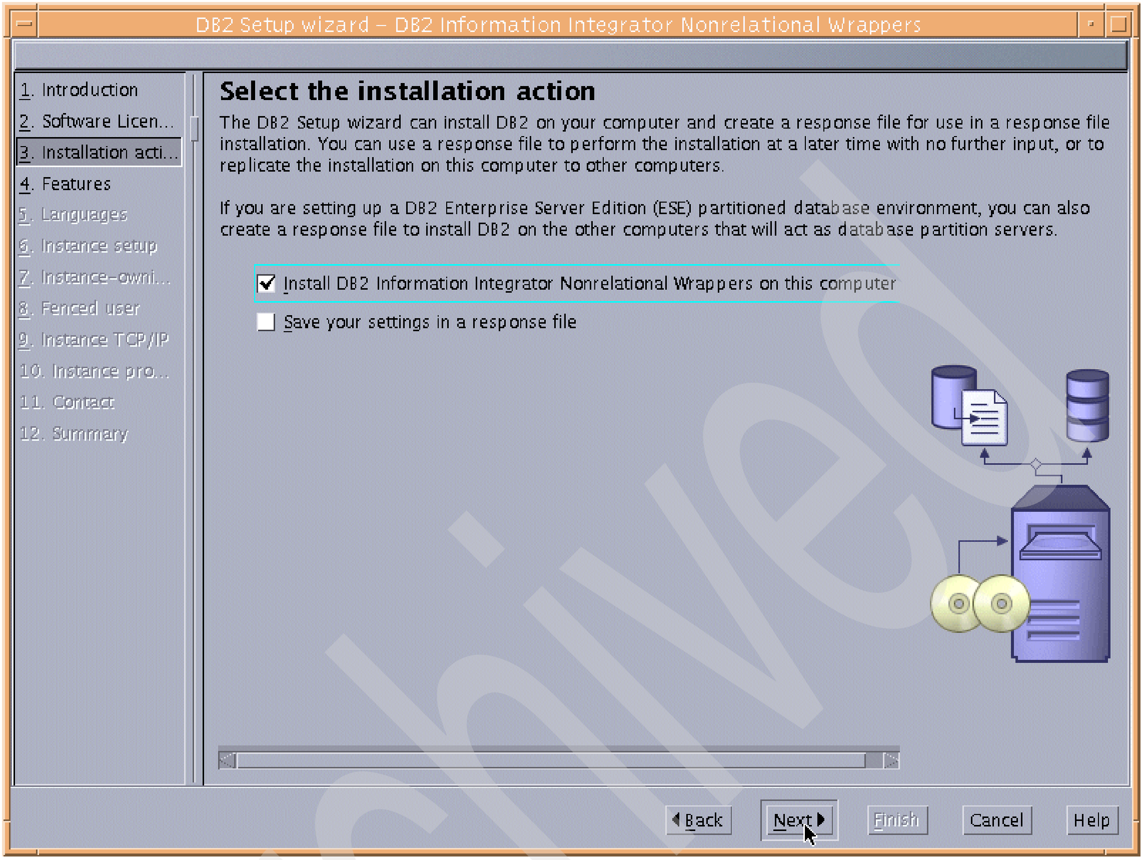


Figure A-24 Non relational wrappers setup - Select install actions

Select **Install DB2 Information Integrator Nonrelational Wrappers on this computer**. We chose not to select the **Save your settings in a response file** option, which provides for recording the interactions with the installation process in a text file.

> **Tip:** A silent DB2 II installation can be performed by supplying a response file as an argument for `iisetup.bin`.

Click **Next** to view the screen shown in Figure A-25.

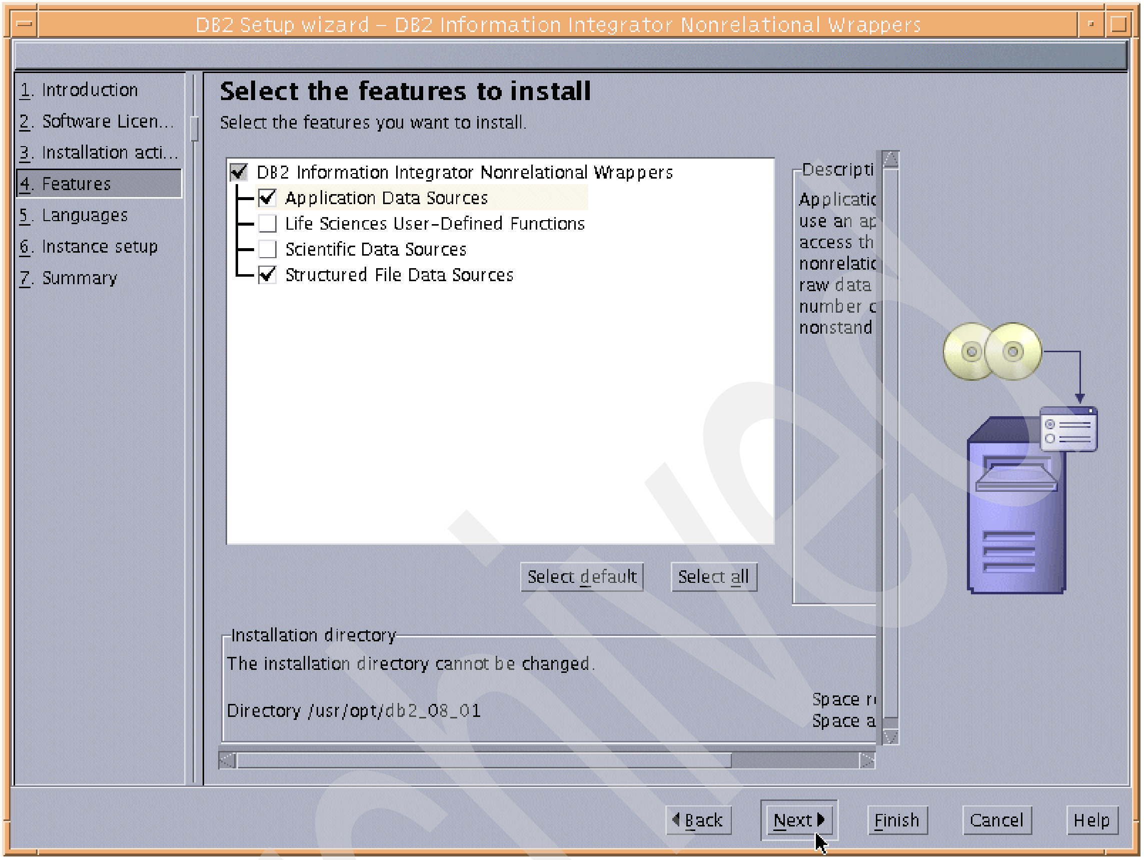


Figure A-25 Non relational wrappers setup - Features to install

Figure A-11 provides the option of selecting the specific data sources that DB2 II should connect to.

Select all the data sources you want to connect to from DB2 II — if you are not sure, just install all the wrappers to avoid having to install one later.

> **Attention:** You need to choose **custom data source support** to enable the possibility of adding new wrappers included in future fixpak releases.

Click **Next** to view the screen shown in Figure A-26.

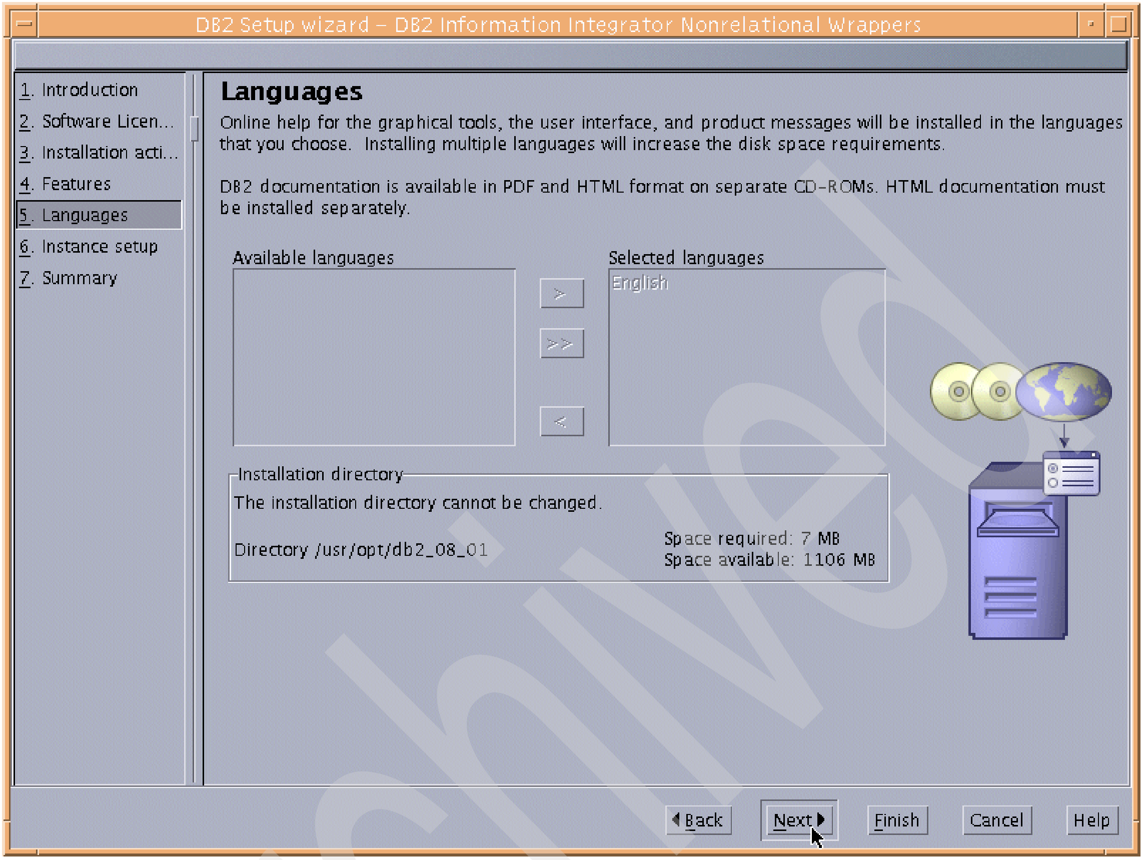


Figure A-26 Non relational wrappers setup - Languages

Select the desired languages from the **Available languages** list, and click **Next** to view the screen shown in Figure A-27.

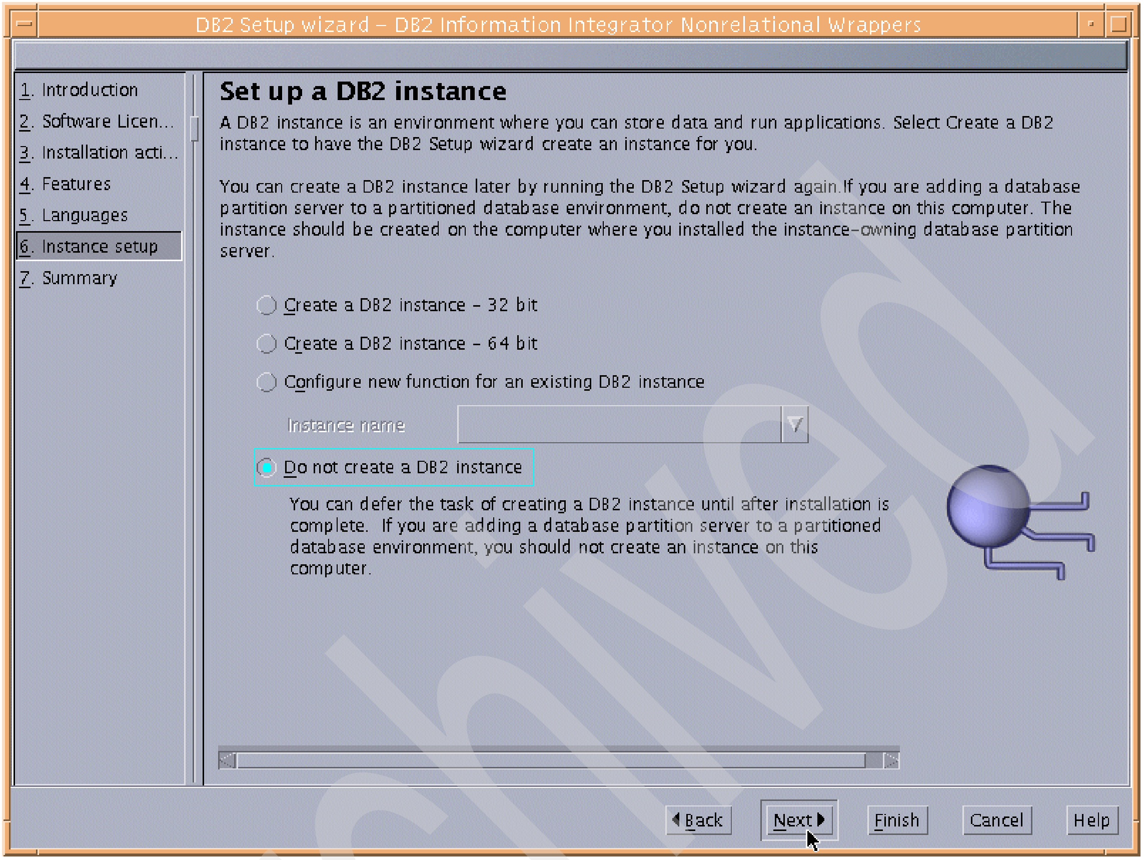


Figure A-27 Non relational wrappers setup - Setup DB2 instance

Figure A-27 provides the option of having the DB2 Setup wizard create a DB2 instance.

Select **Do not create a DB2 instance** and then click **Next** to display the screen shown in Figure A-28.

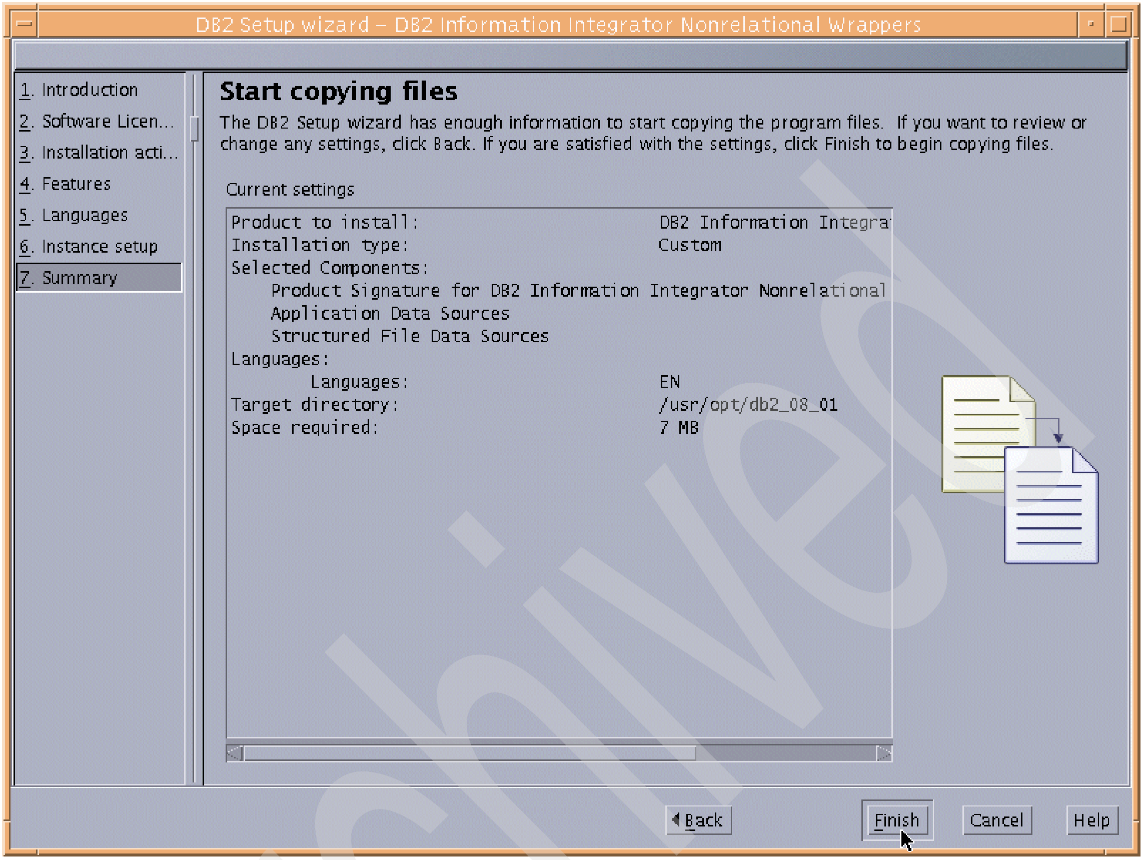


Figure A-28 Non relational wrappers setup - Summary

Figure A-28 summarizes the selections made. You can click the **Back** button to change any of the selections.

Click **Finish** to proceed with the installation as shown in Figure A-29.

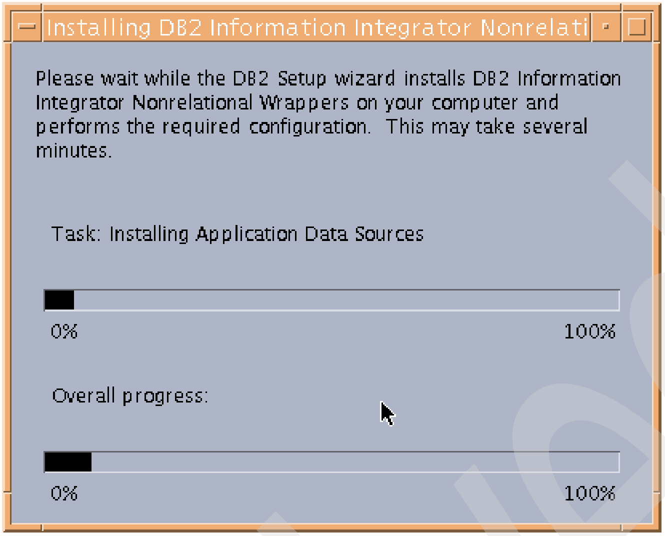


Figure A-29 Non relational wrappers setup - Installation progress

After a few minutes, the status of the install is displayed on **the Setup is complete** screen as shown in Figure A-30.

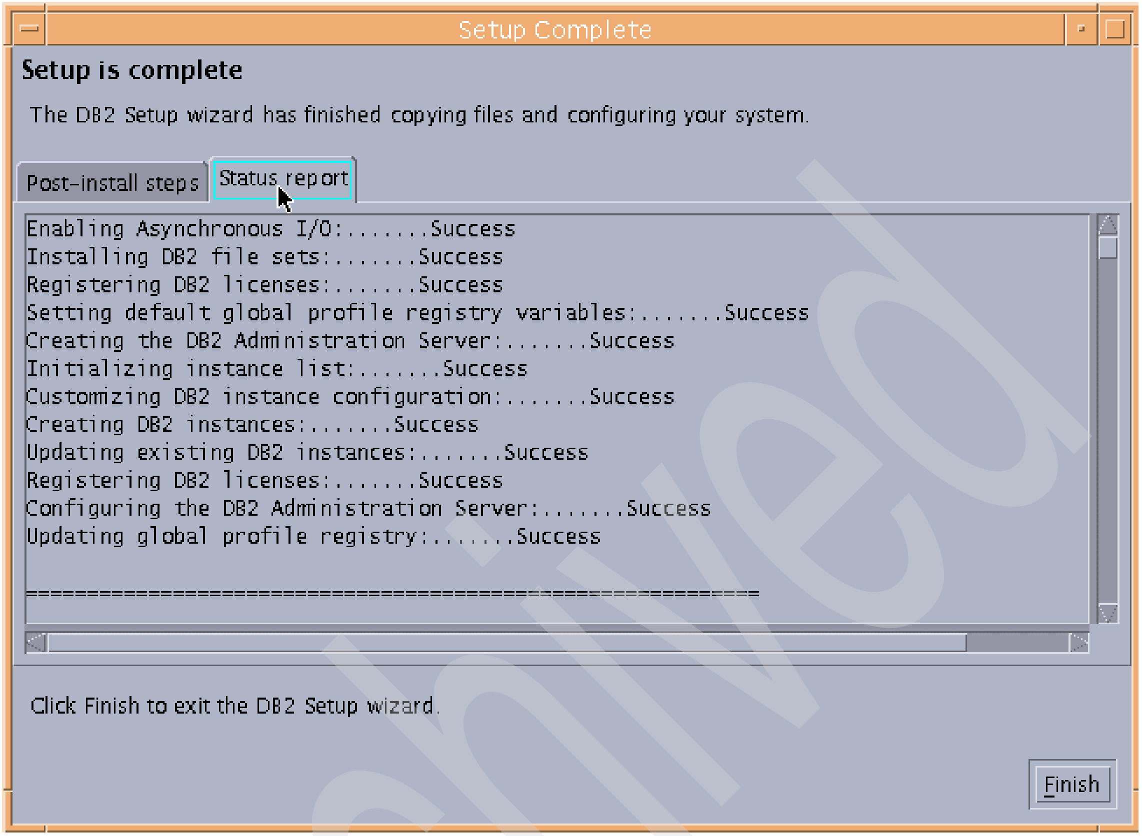


Figure A-30 Non relational wrappers setup - Status

Check the success of the installation by clicking the **Status report** tab.

Click **Finish** to proceed to the screen shown in Figure A-31.

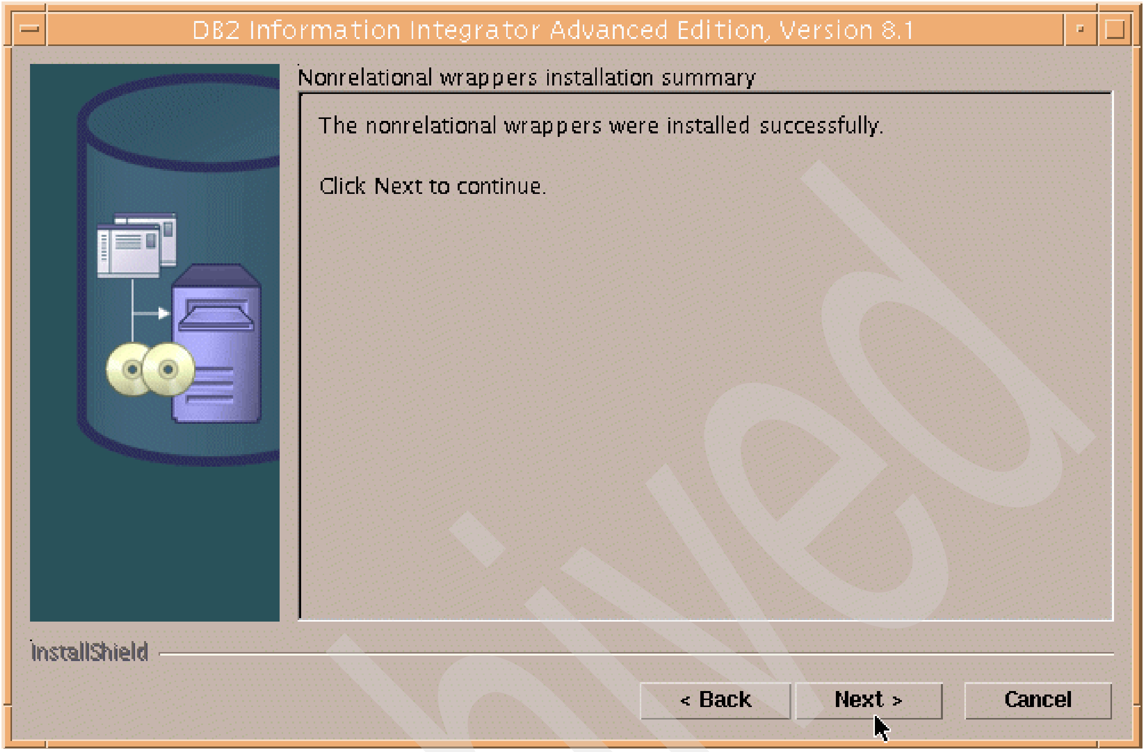


Figure A-31   Non relational wrappers setup - Complete

Figure A-31 displays the status with a message saying that the nonrelational wrappers were installed successfully.

Click **Next** to view the screen shown in Figure A-32.

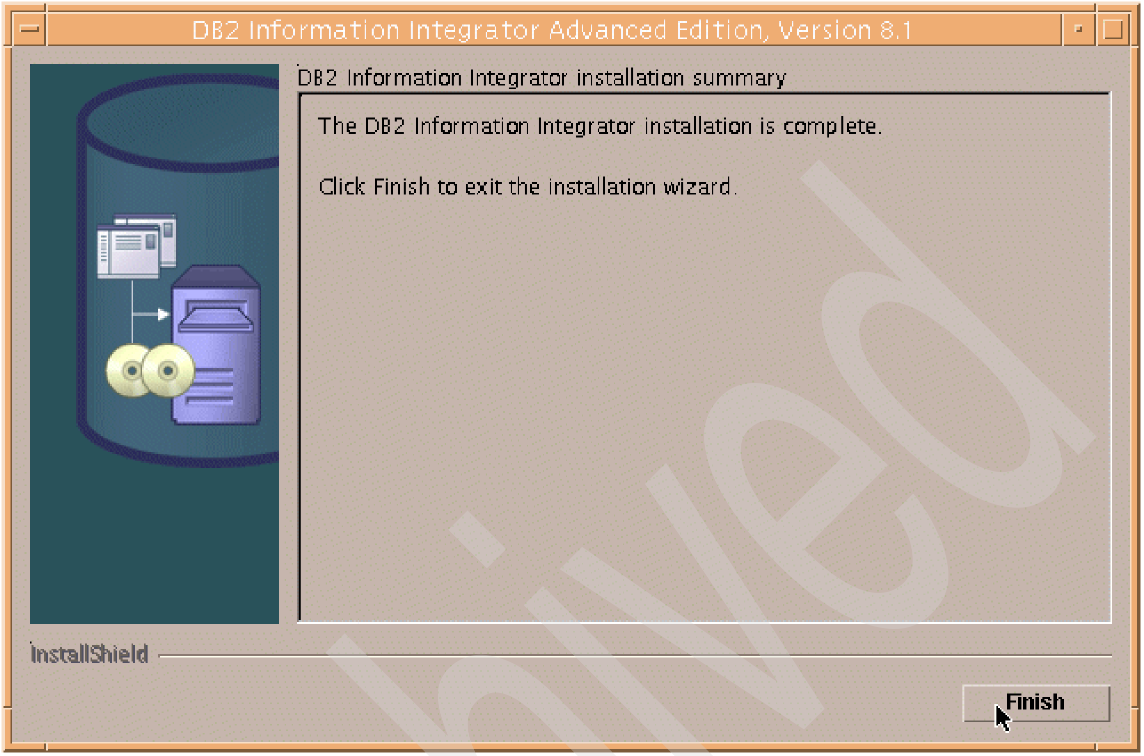


Figure A-32 DB2 Information Integrator installation - Complete

Figure A-32 is the final screen in the DB2 II Version 8.1 installation. Click **Finish** to complete the installation.

A.3.2 Installing on Windows

This following installation was done on a Windows 2000 NetVista™ Model 8305 2.4 GHz with 1 GB memory, and 40 GB of disk space.

> **Attention:** Ensure that all the prerequisites are in place prior to commencing the installation.
>
> We strongly recommend that you consult the DB2 II release notes, and other installation documentation for a successful DB2 II installation.

To install DB2 II on Windows, double click **iiSetup.exe** from the root directory of your CD, or the directory having the DB2 II image. This displays the **DB2 Information Integrator Installation Launchpad** screen shown in Figure A-33.

Figure A-33 DB2 Information Integrator Installation - Launch screen

Figure A-33 provides options for reviewing the most current information on installation prerequisites and release notes for the DB2 Information Integrator, Version 8.1 installation.

> **Note:** We recommend that you browse through these documents to ensure that all prerequisites are in place.

Click **Install Products** to display the **Software License Agreement** page shown in Figure A-34.

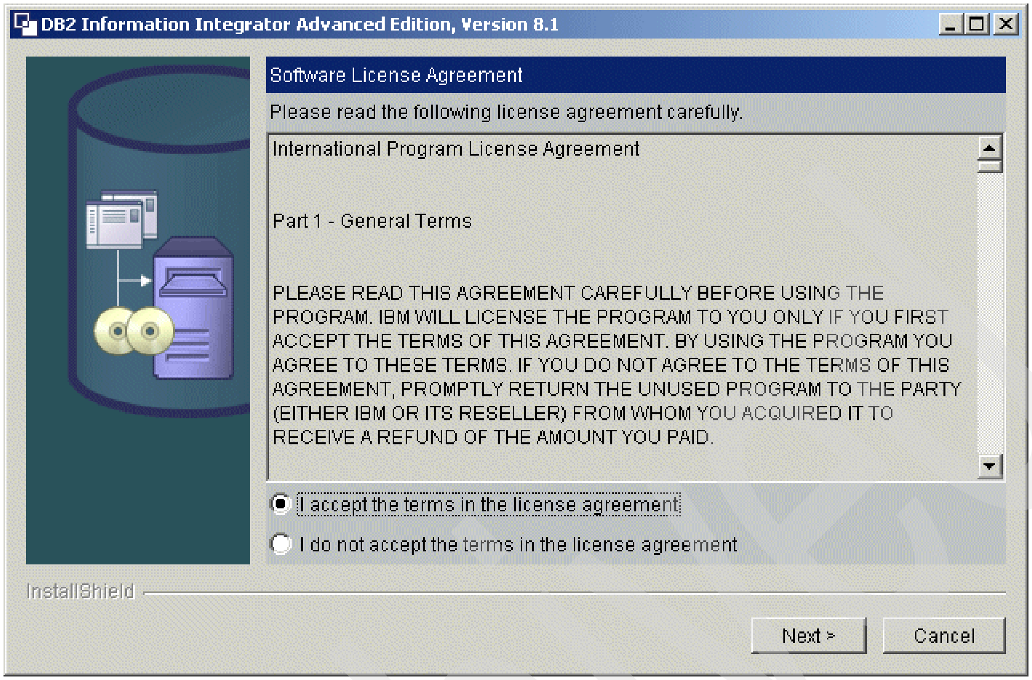


Figure A-34 SW license agreement

Please read the license agreement before you proceed.

Select **I accept the terms in the license agreement** and click **Next** to view the screen shown in Figure A-35.

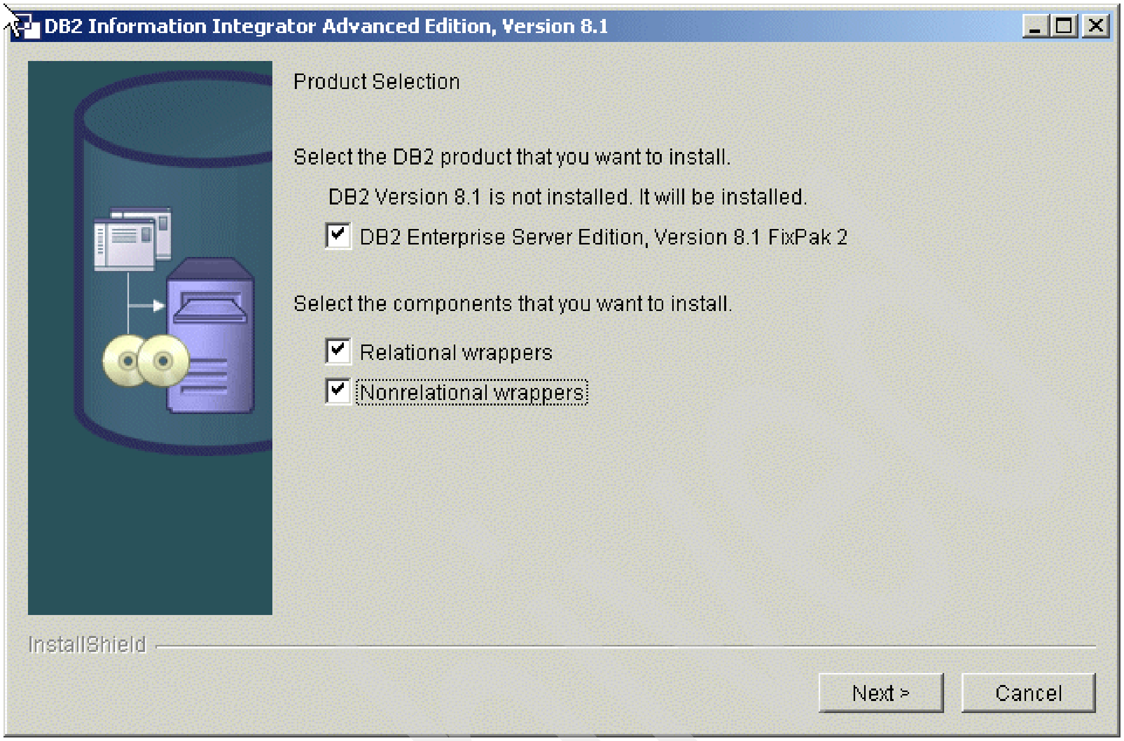


Figure A-35 Product selection - Relational, non relational wrappers

Figure A-35 provides a choice of the products to install. The two product choices are as follows:

1. **Relational wrappers** that include libraries to access relational data sources like DB2, Microsoft SQL Server, Oracle, Sybase, and Teradata.
2. **Nonrelational wrappers** that include libraries to access structured files (Excel, Table-structured files, XML), application data sources (BioRS, Documentum, Entrez, IBM Lotus Extended Search) and scientific related data (BLAST, HMMER).

Select both products as shown in Figure A-35.

> **Note:** Since we did not have DB2 UDB ESE installed, we chose to install it first. The following screens do *not* include details of the installation of DB2 UDB.

Click **Next** to display screen Figure A-36 begin the installation of the relational wrappers.

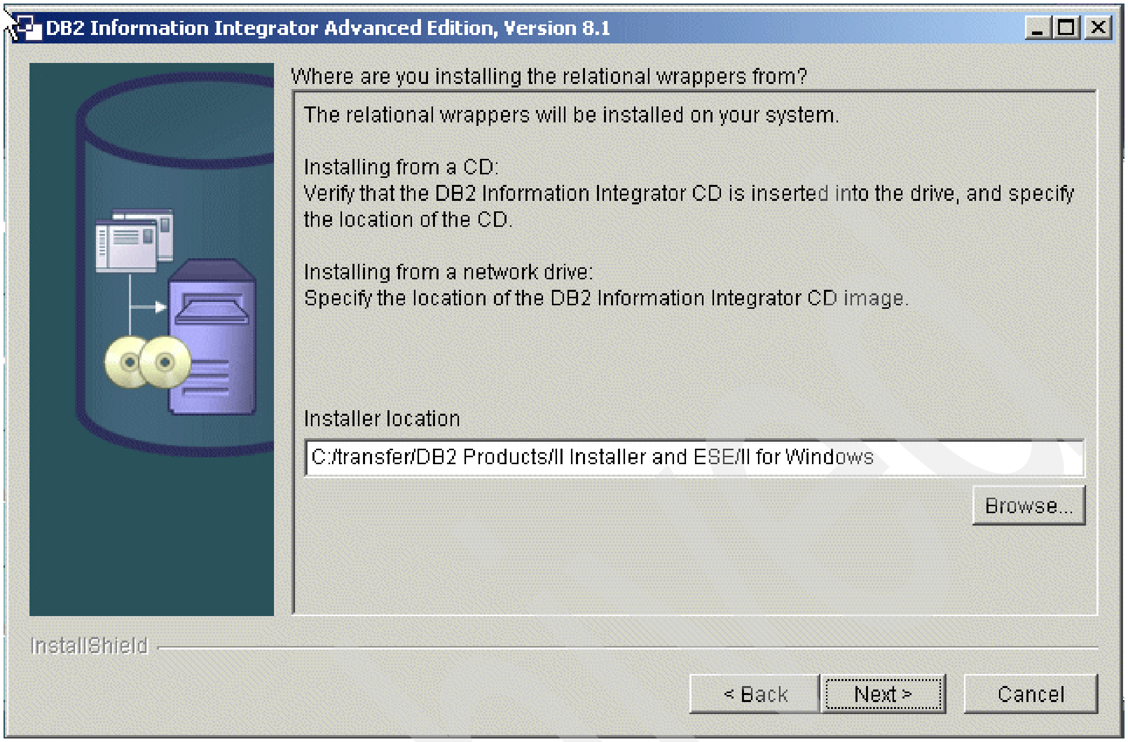


Figure A-36 Source of relational wrapper installation

Figure A-36 requests the location of the relational wrapper code.

Browse the CD, or any other file system to select the path for the installation files of the relation wrappers as shown in Figure A-36. Click **Next** to start the relational wrappers installer as shown in Figure A-37.

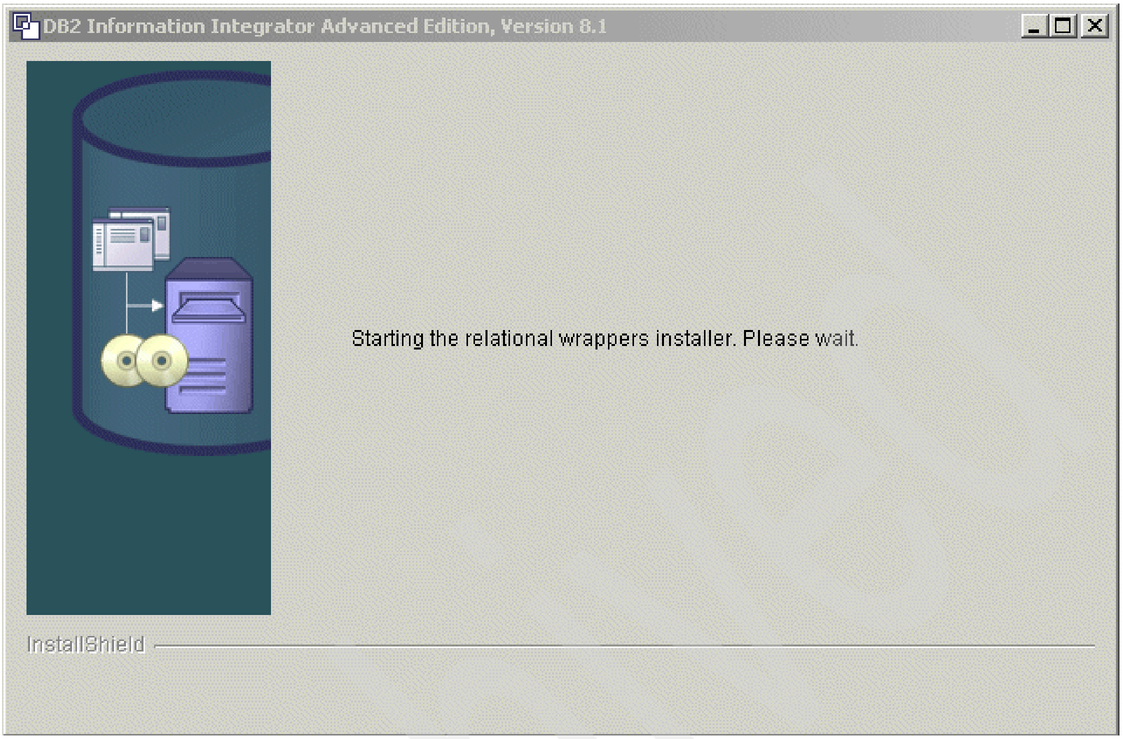


Figure A-37 Start the relational wrappers installer

After the relational wrapper installer finishes loading it opens the welcome screen for the relational wrappers installer as shown in Figure A-38.

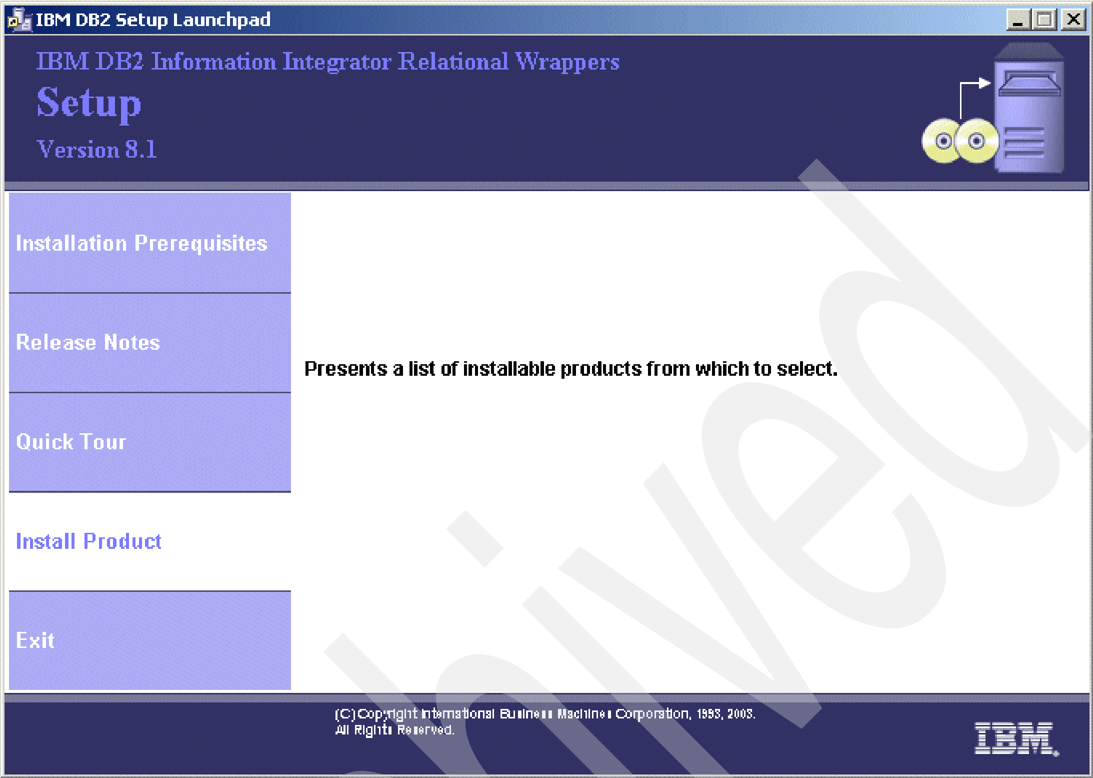


Figure A-38 Relational wrappers setup

Figure A-38 provides options for reviewing the most current information on installation prerequisites and release notes for the relational wrappers installation.

> **Note:** We recommend that you browse through these documents to ensure that all prerequisites are in place.

Click **Install Products** to select the products to install as shown in Figure A-39.

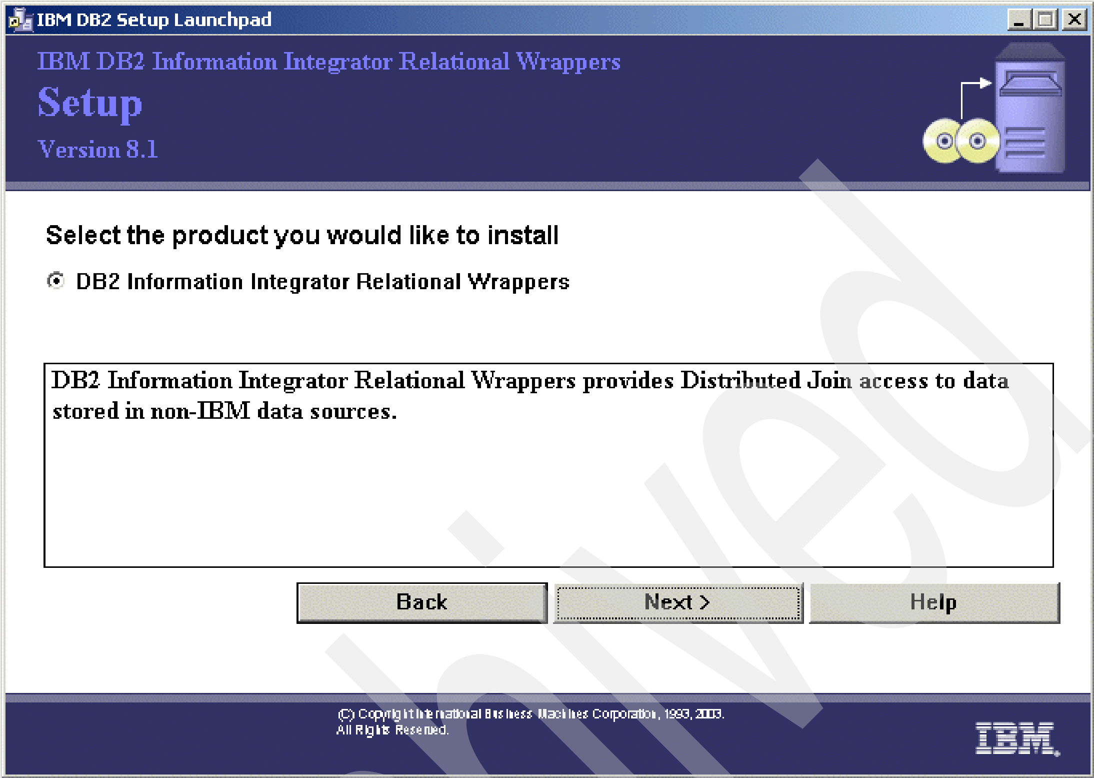


Figure A-39 Relational wrappers setup - Select products

Select the only option: **DB2 Information Integrator Relational Wrappers** and click **Next** to continue.

> **Note:** The screen might flash a few times and then a DB2 installation Splash screen is displayed as shown in Figure A-40.

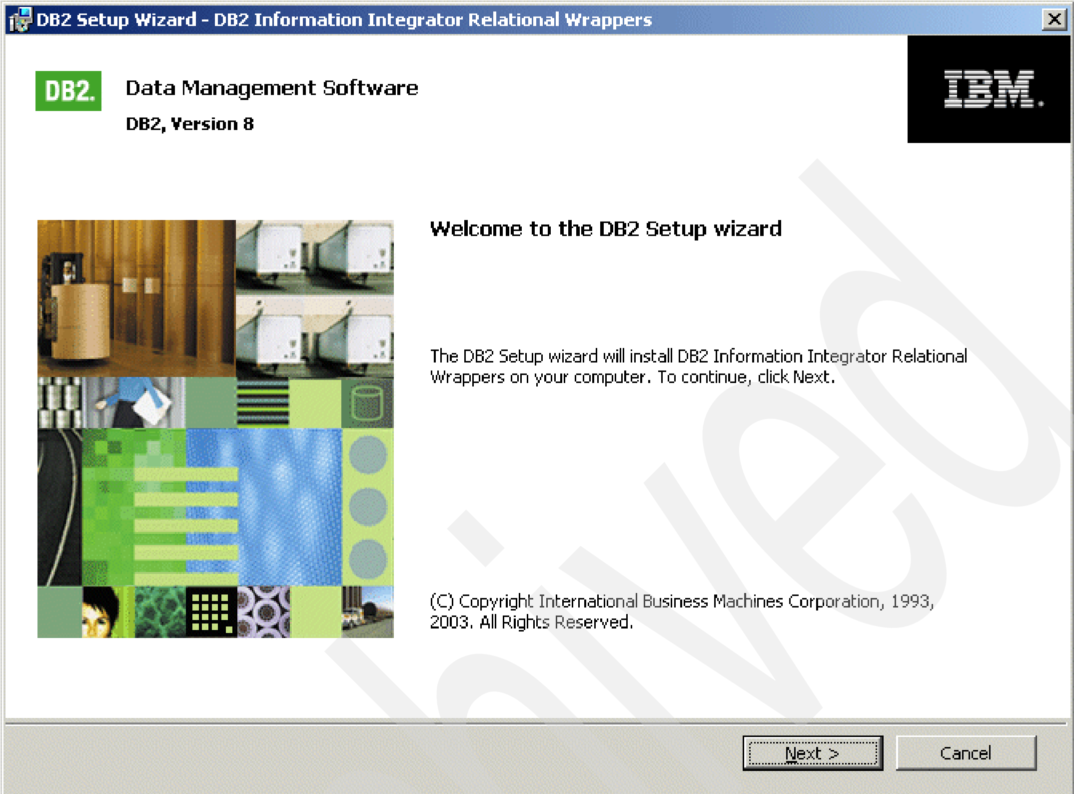


Figure A-40 Relational wrappers setup - Introduction

Click **Next** to display the **Software License Agreement** page as shown in Figure A-41.

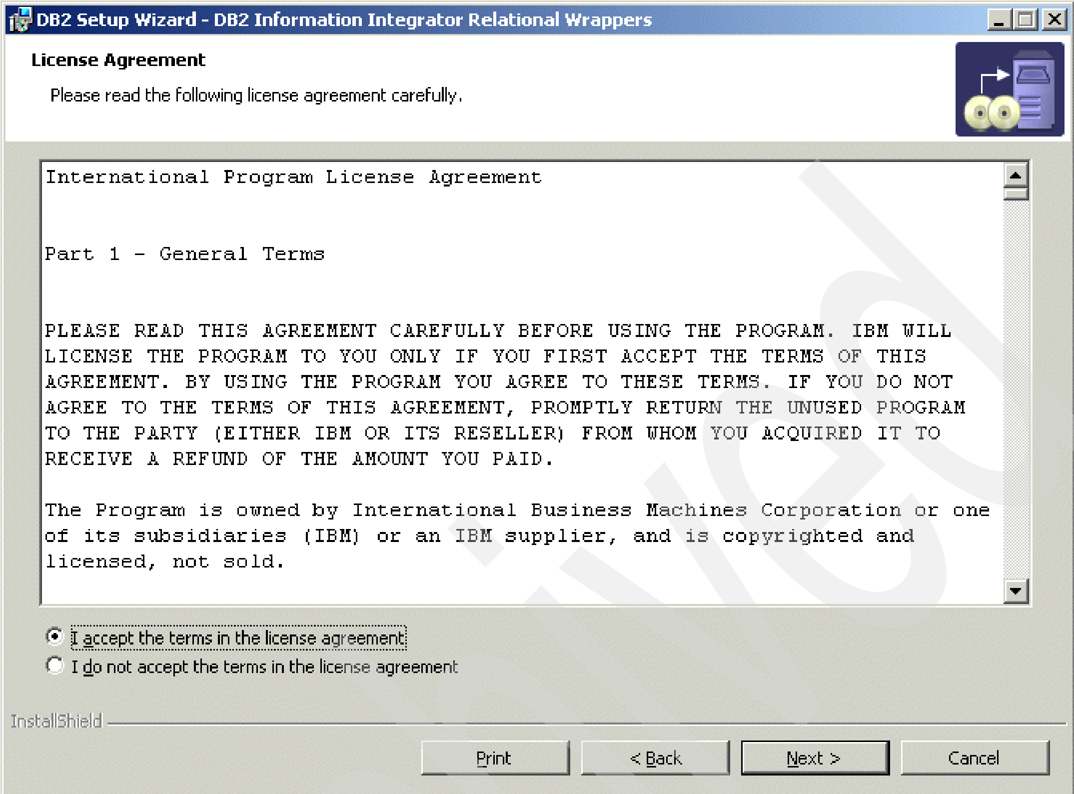


Figure A-41 Relational wrappers setup - SW License Agreement

Please read the license agreement before you proceed.

Select **I accept the terms in the license agreement** and click **Next** to view the screen shown in Figure A-42.

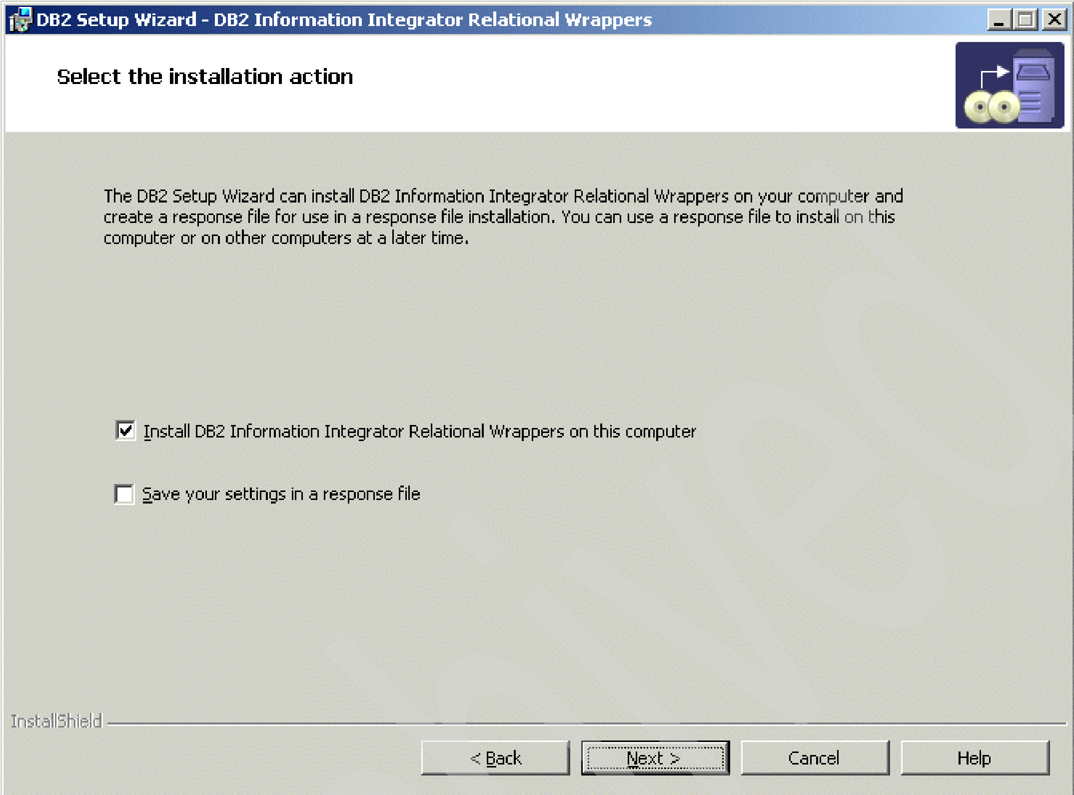


Figure A-42 Relational wrappers setup - Select installation action

Select **Install DB2 Information Integrator Relational Wrappers on this computer**. We chose not to select the **Save your settings in a response file** option which provides for recording the interactions with the installation process in a text file.

> **Tip:** A silent DB2 II installation can be performed by supplying a response file as an argument for iisetup.bin.

Click **Next** to view the screen shown in Figure A-43.

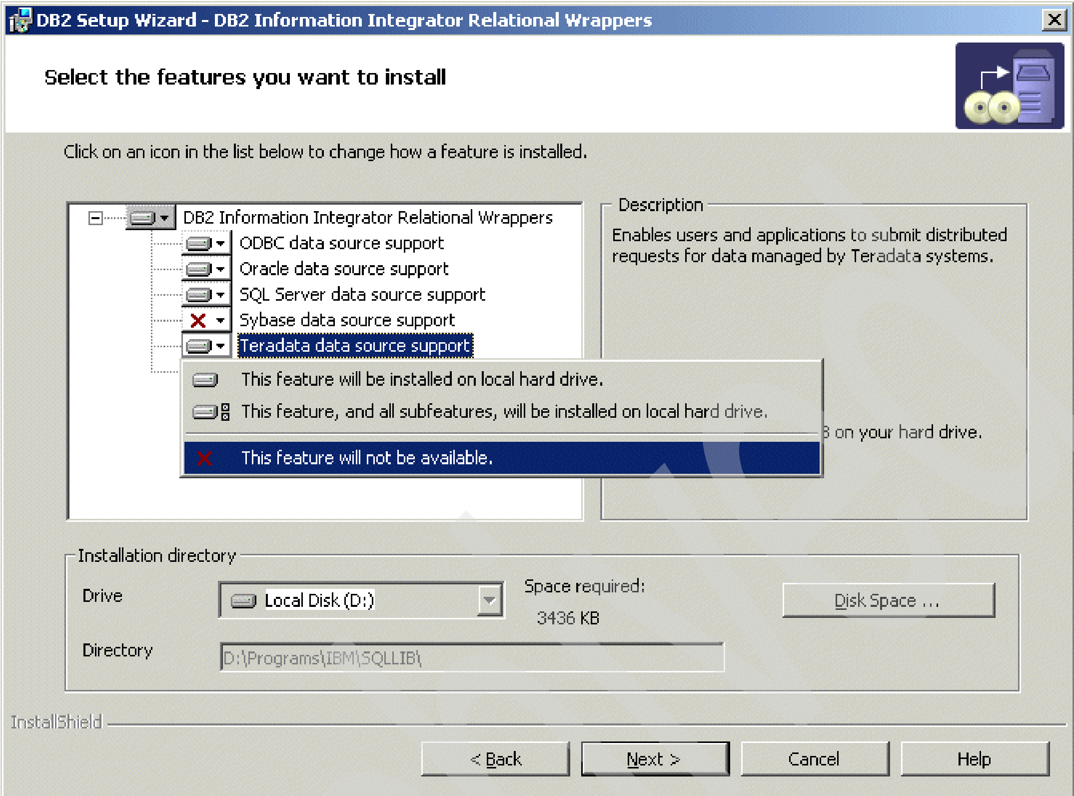


Figure A-43 Relational wrappers setup - Features to install

Figure A-43 provides the option of selecting the specific types of wrappers you want installed.

Select all the data sources you want to connect to from DB2 II. If you are not sure, just install all the wrappers to avoid having to install it later on.

> **Attention:** You need to choose **custom data source support** to enable the possibility of adding new wrappers included in future fixpak releases.

Click **Next** to view the screen shown in Figure A-44.

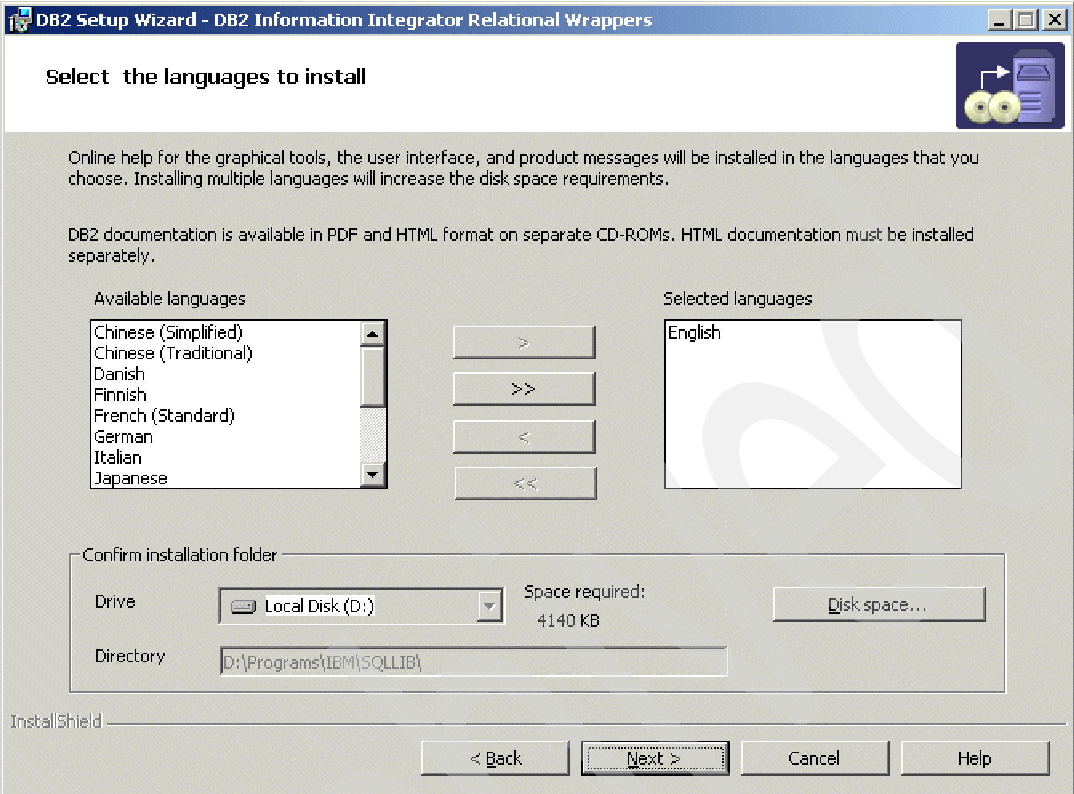


Figure A-44 Relational wrappers setup - Languages

Figure A-44 provides the option of selecting the languages to be supported by DB2 II.

Add your desired languages from the **Available languages** list and click **Next** to view the screen shown in Figure A-45.

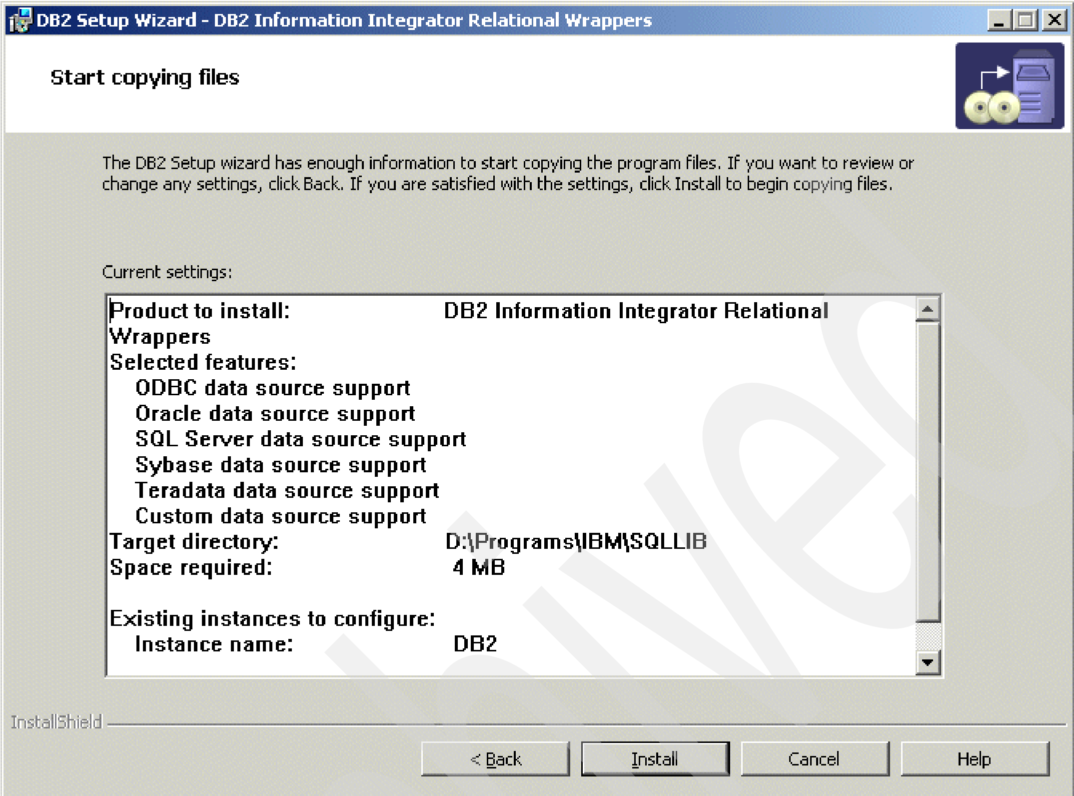


Figure A-45 Relational wrappers setup - Summary

Figure A-45 summarizes the selections made. You can click the **Back** button to change any of the selections.

Click **Install** to proceed with the installation of the relational wrappers as shown in Figure A-46.

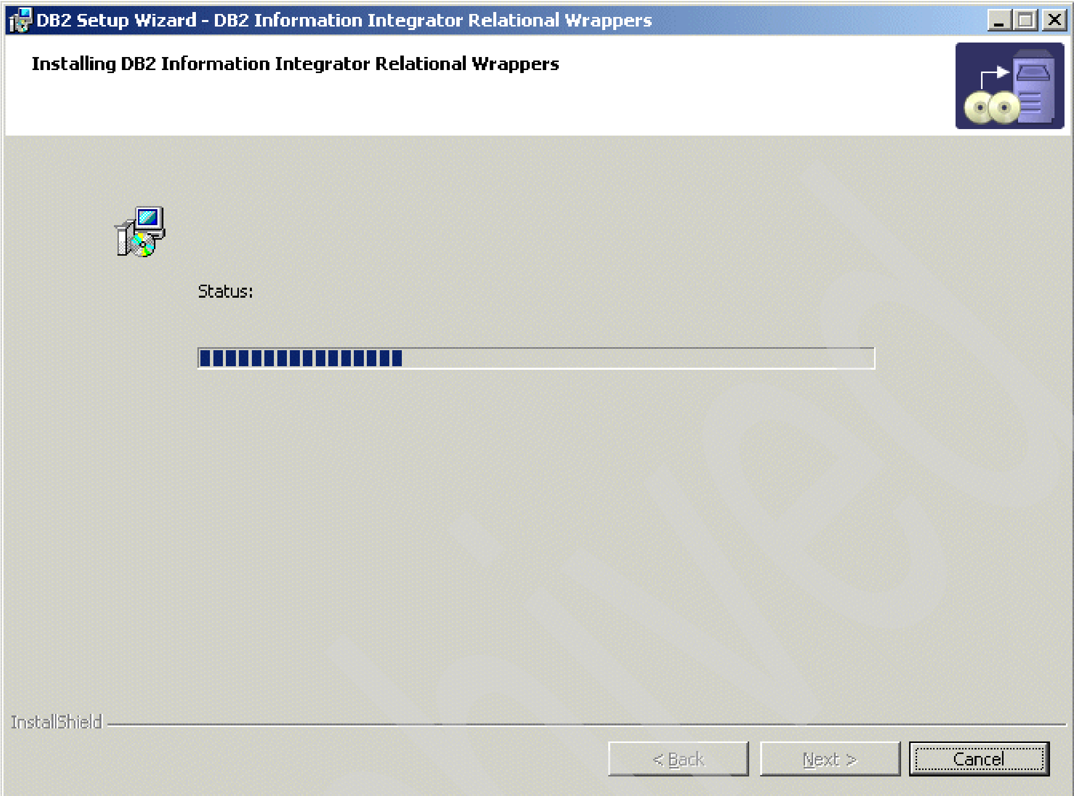


Figure A-46 Relational wrappers setup - Installation progress

After a few minutes, the status of the install is displayed on the **Setup is complete** screen as shown in Figure A-47.

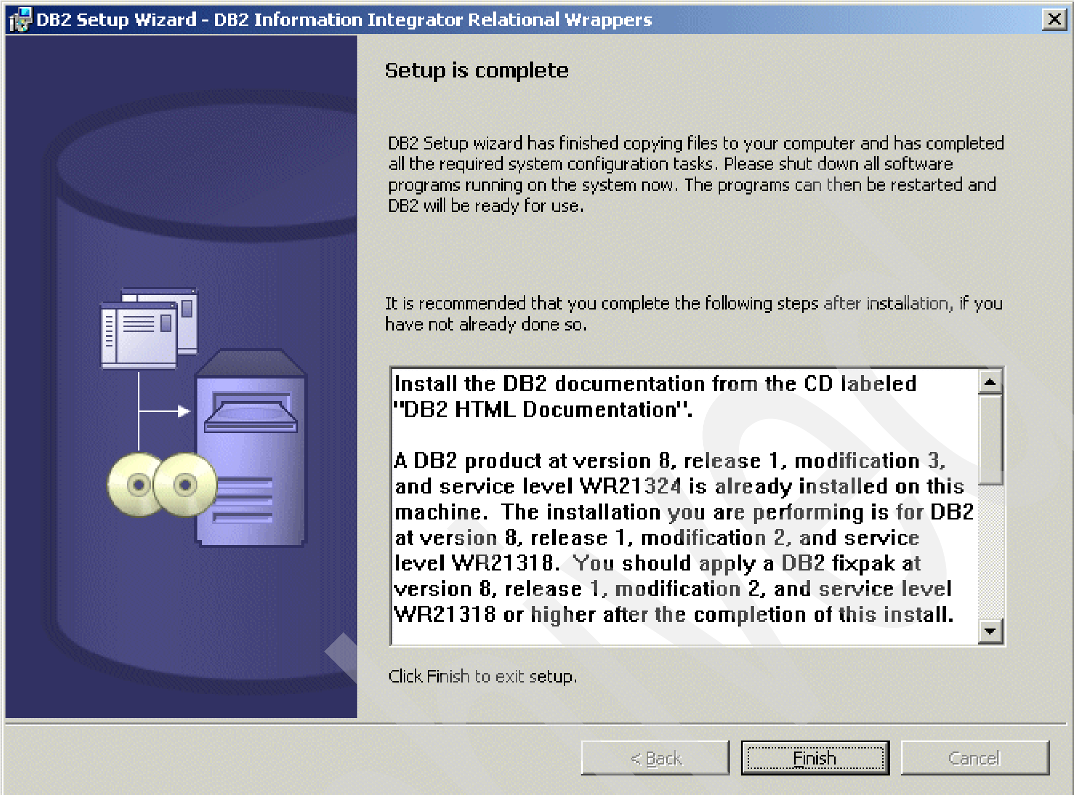


Figure A-47 Relational wrappers setup - Complete

Click **Finish** to proceed to the relational wrappers status screen shown in Figure A-48.

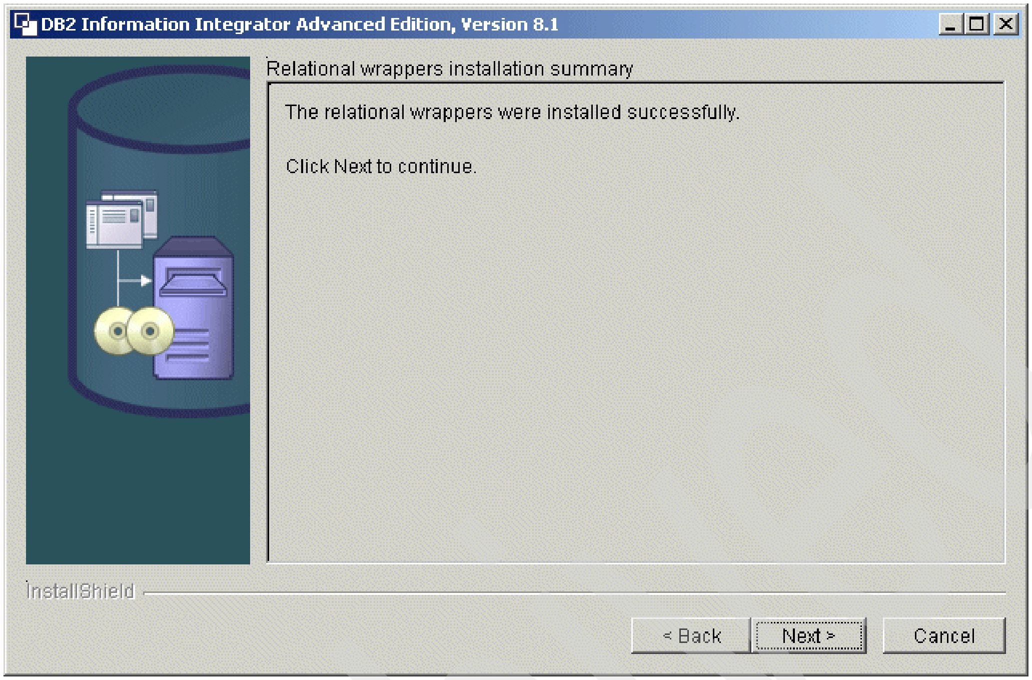


Figure A-48 Relational wrappers setup - Status

Figure A-48 indicates that the relational wrappers were installed successfully.

Click **Next** to continue with the installation of the nonrelational wrappers as shown in Figure A-49.

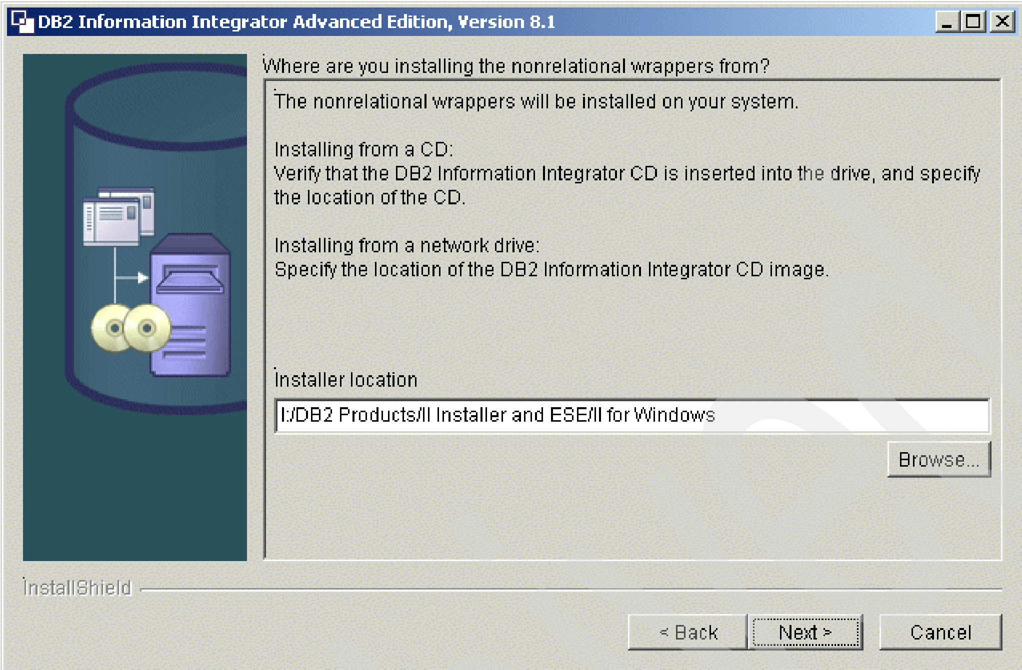


Figure A-49 Source of non relational wrapper installation

Figure A-49 requests the location of the nonrelational wrapper code.

Browse the CD or any other file system to select the path for the nonrelational wrappers installation source, and click **Next** to begin loading the nonrelational wrappers installer as shown in Figure A-50.

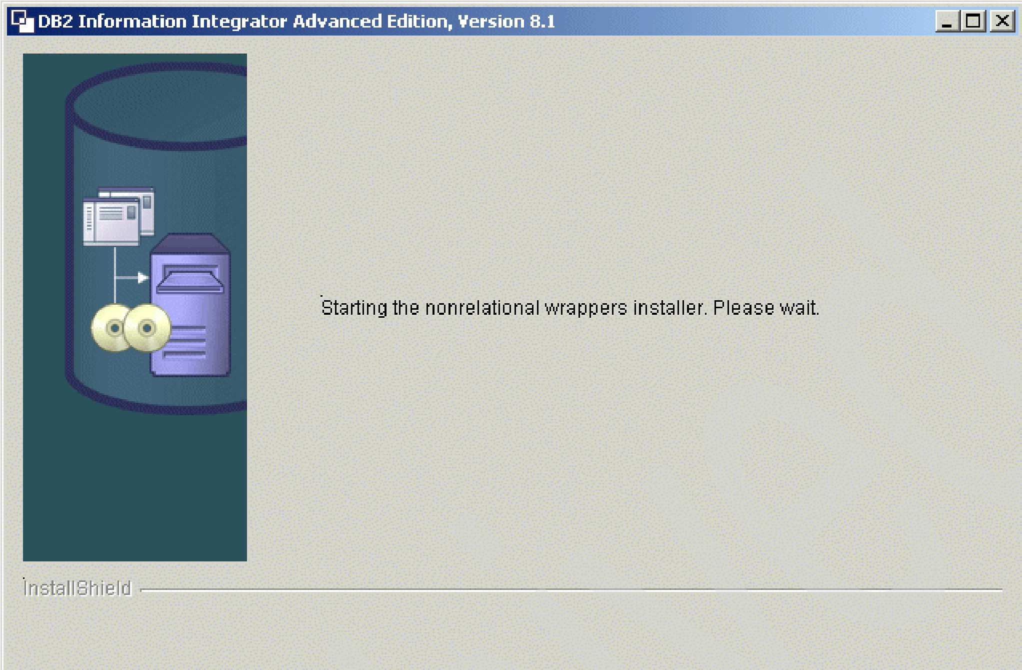


Figure A-50 Start the non relational wrappers installer

After the nonrelational wrappers installer finishes loading, it launches the **IBM DB2 Setup Launchpad** screen for the nonrelational wrappers as shown in Figure A-51.

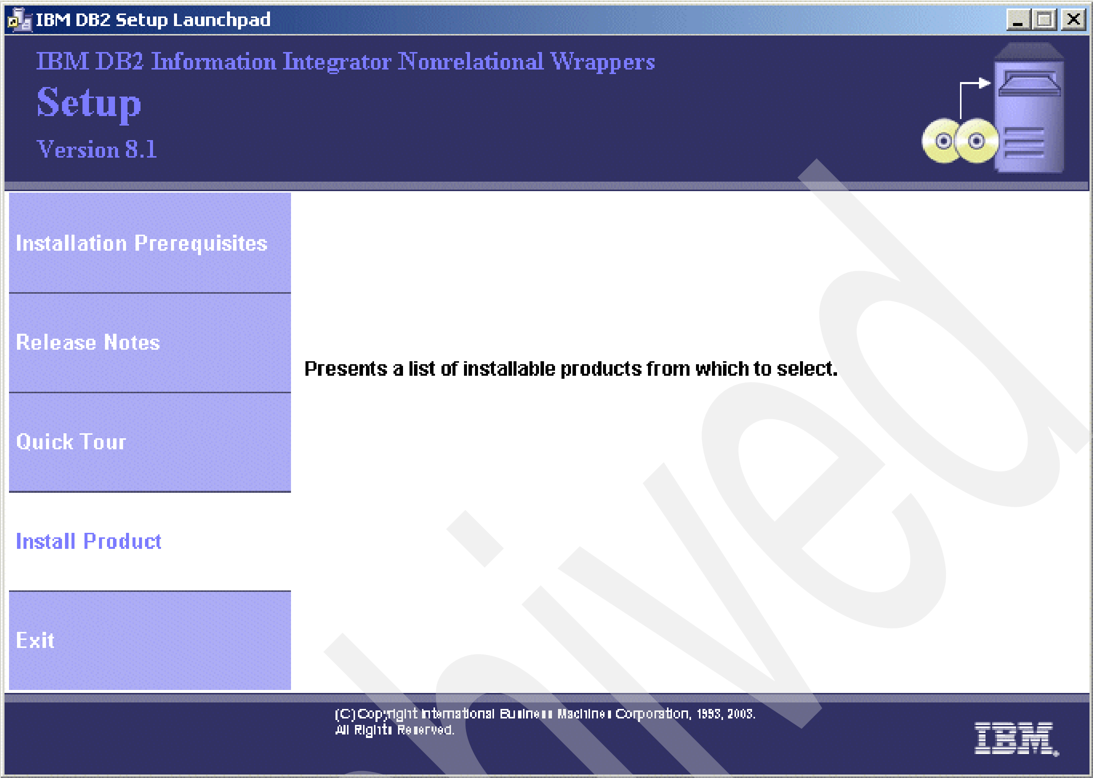


Figure A-51 Non relational wrappers setup

Figure A-51 provides options for reviewing the most current information on installation prerequisites and release notes for the nonrelational wrappers installation.

Note: We recommend that you browse through these documents to ensure that all prerequisites are in place.

Click **Install Products** to select the product to install as shown in Figure A-52.

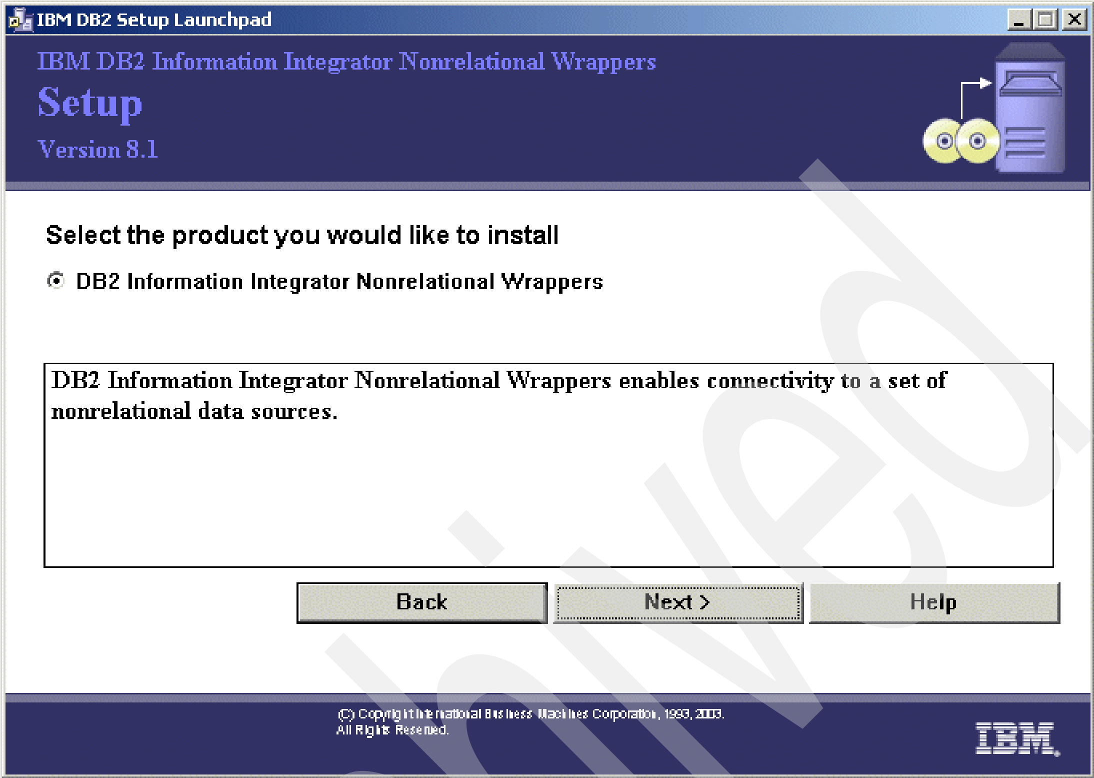


Figure A-52 Non relational wrappers setup - Select products

Select the only option: **DB2 Information Integrator Nonrelational Wrappers** and click **Next** to continue.

> **Note:** The screen might flash a few times and then a DB2 installation Splash screen is displayed as shown in Figure A-53.

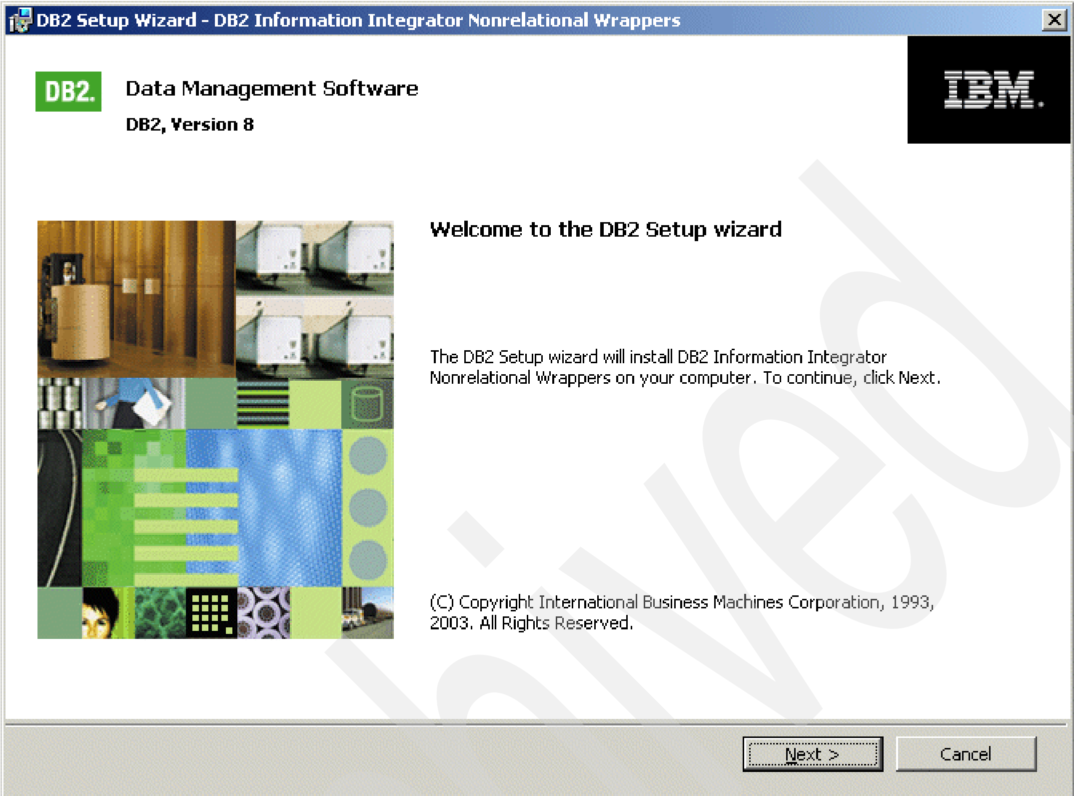


Figure A-53 Non relational wrappers setup - Introduction

Click **Next** to display the **Software License Agreement** page as shown in Figure A-54.

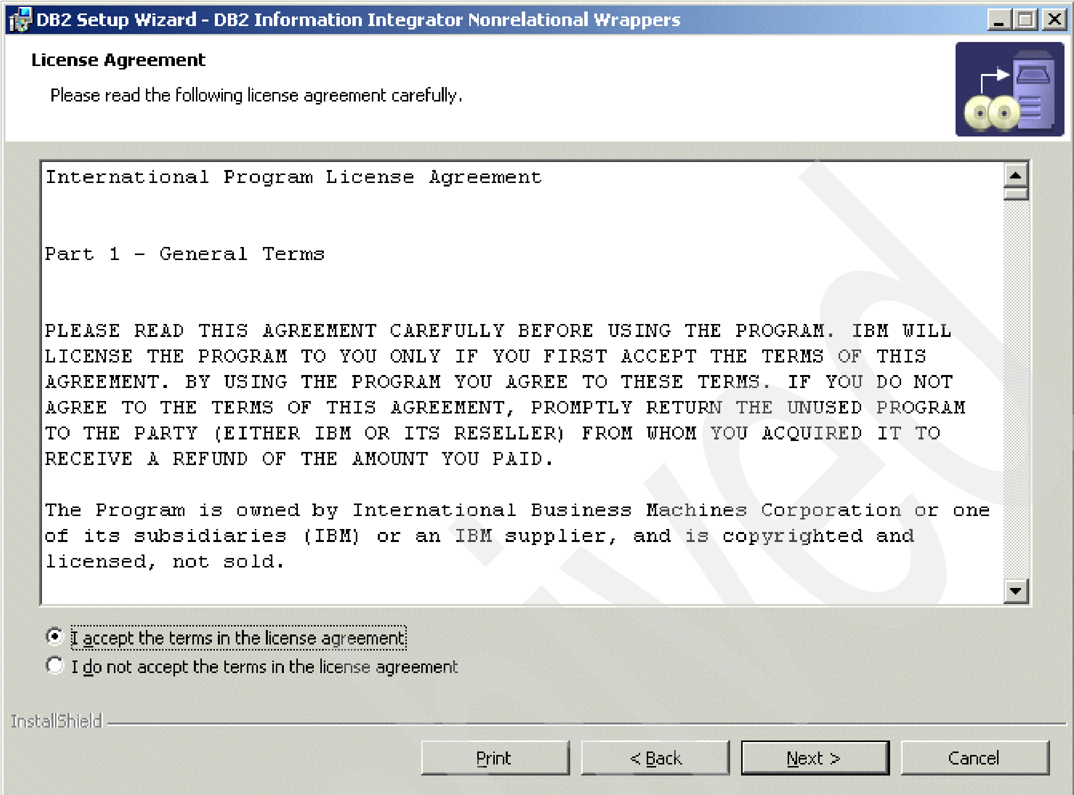


Figure A-54 Non relational wrappers setup - SW License Agreement

Please read the license agreement before you proceed.

Select **I accept the terms in the license agreement** and click **Next** to view the screen shown in Figure A-55.

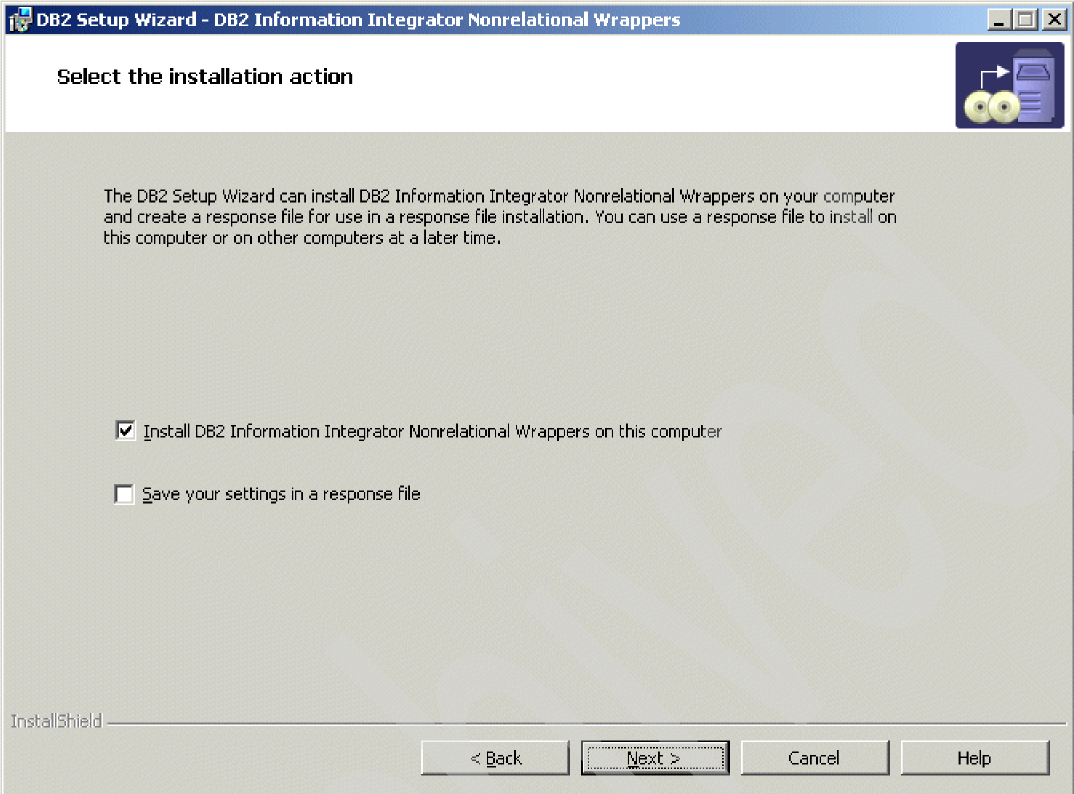


Figure A-55 Non relational wrappers setup - Select install actions

Select **Install DB2 Information Integrator Nonrelational Wrappers on this computer.** We chose not to select **Save your settings in a response file** option, which provides for recording the interactions with the installation process in a text file.

> **Tip:** A silent DB2 installation can be performed by supplying a response file as an argument for iisetup.bin.

Click **Next** to view the screen shown in Figure A-56.

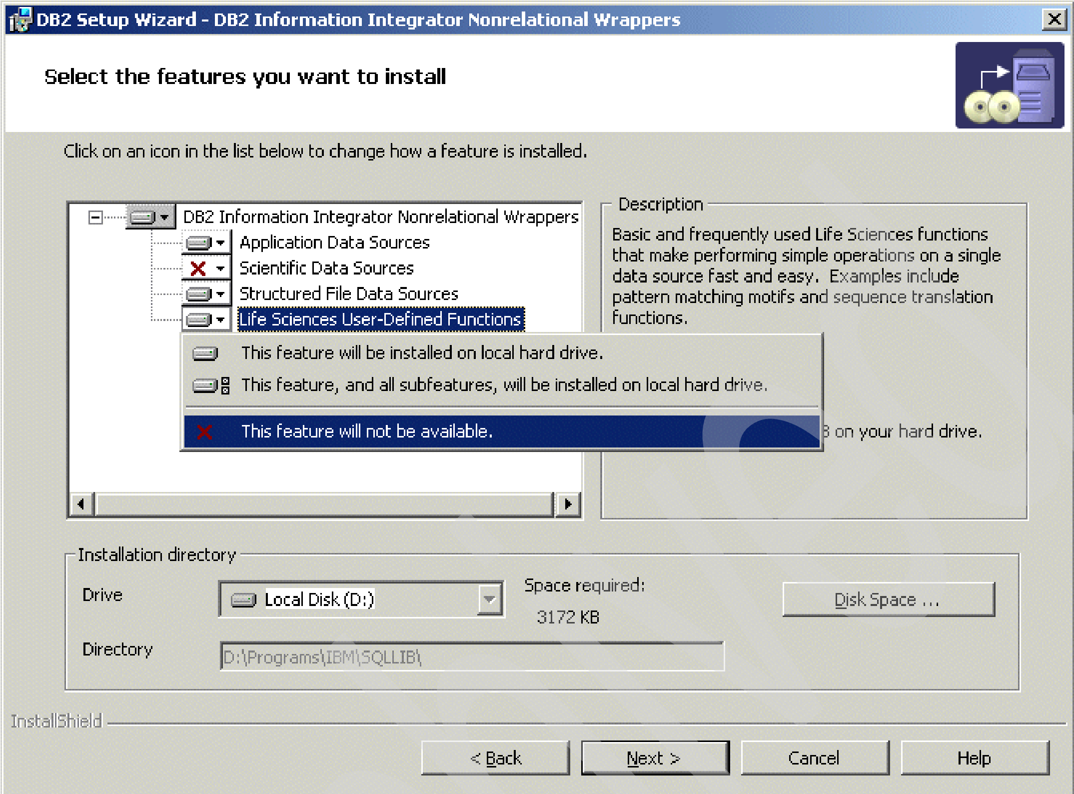


Figure A-56 Non relational wrappers setup - Features to install

Figure A-56 provides the option of selecting the specific data sources that DB2 II should connect to.

Select all the data sources you want to connect to from DB2 II.If you are not sure, just install all the wrappers to avoid having to install one later.

> **Attention:** You need to choose **custom data source support** to enable the possibility of adding new wrappers included in future fixpak releases.

Click **Next** to view the screen shown in Figure A-57.

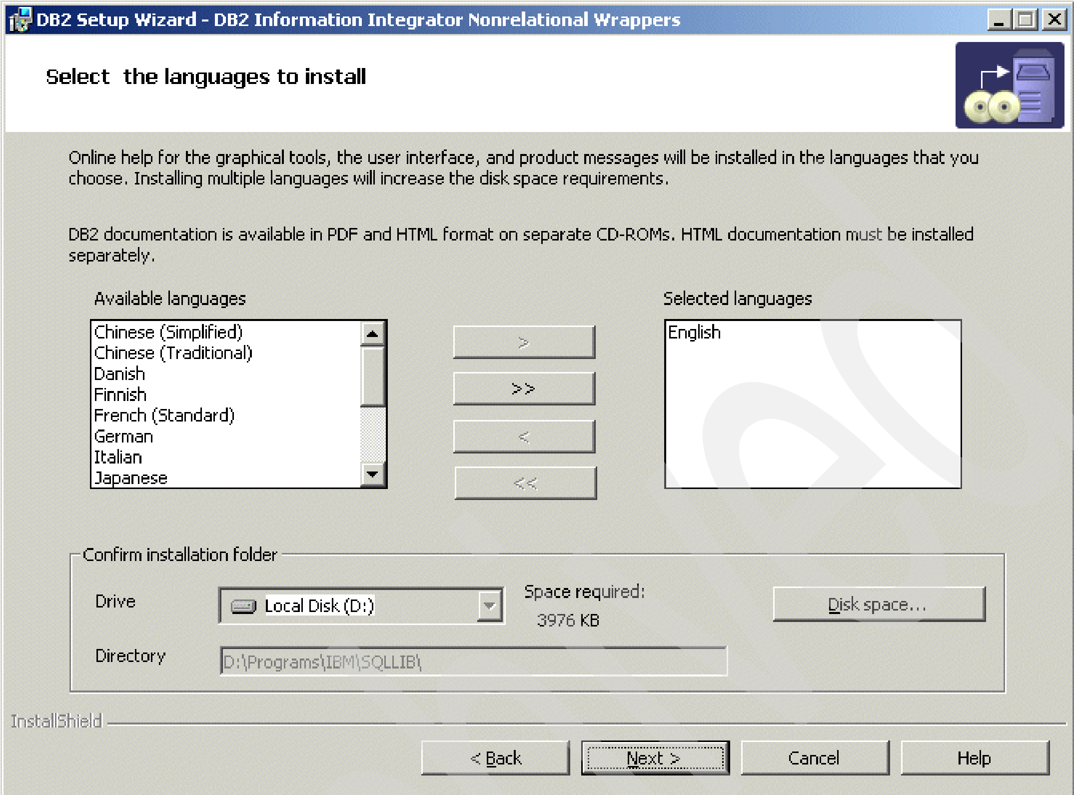


Figure A-57 Non relational wrappers setup - Languages

Figure A-57 provides the option of selecting the languages to be supported by DB2 II.

Add your desired languages from the **Available languages** list and click **Next** to continue to the screen shown in Figure A-58.

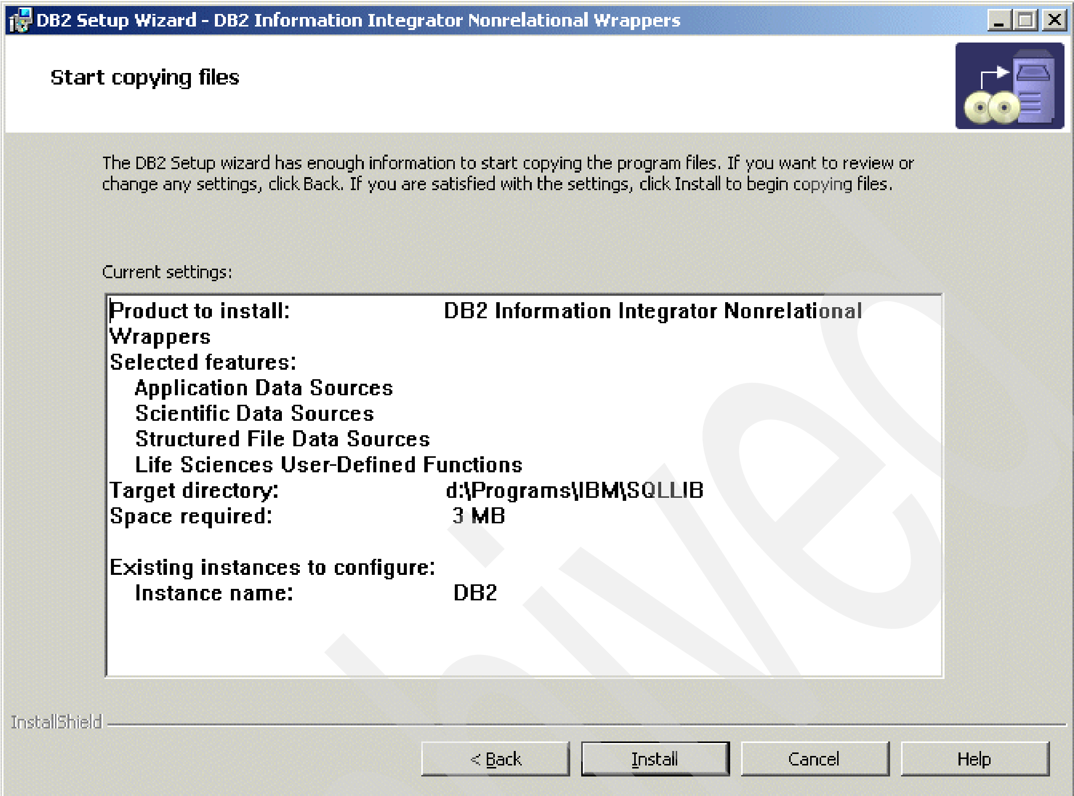


Figure A-58 Non relational wrappers setup - Summary

Figure A-58 summarizes the selections made. You can click the **Back** button to change any of the selections.

Click **Install** to proceed with the installation as shown in Figure A-59.

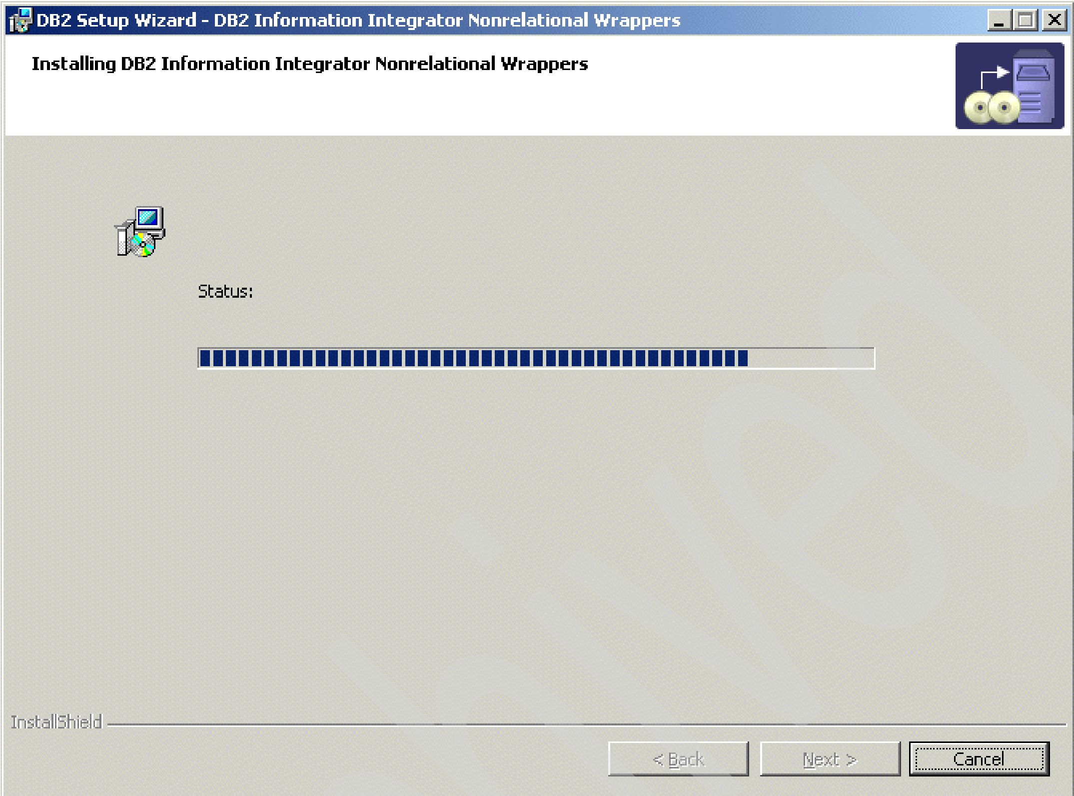


Figure A-59 Non relational wrappers setup - Installation progress

After a few minutes, the status of the install is displayed on the **Setup is complete** screen as shown in Figure A-60.

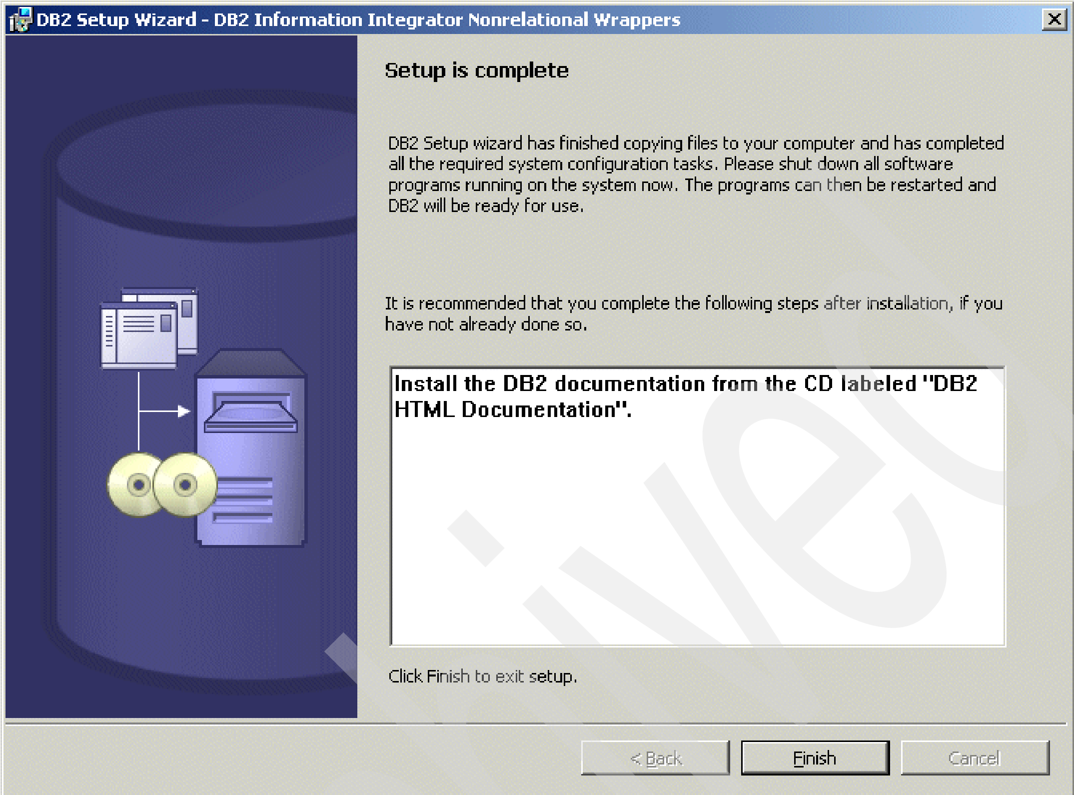


Figure A-60   Non relational wrappers setup - Complete

Click **Finish** to proceed to the nonrelational wrappers status screen shown in Figure A-61.

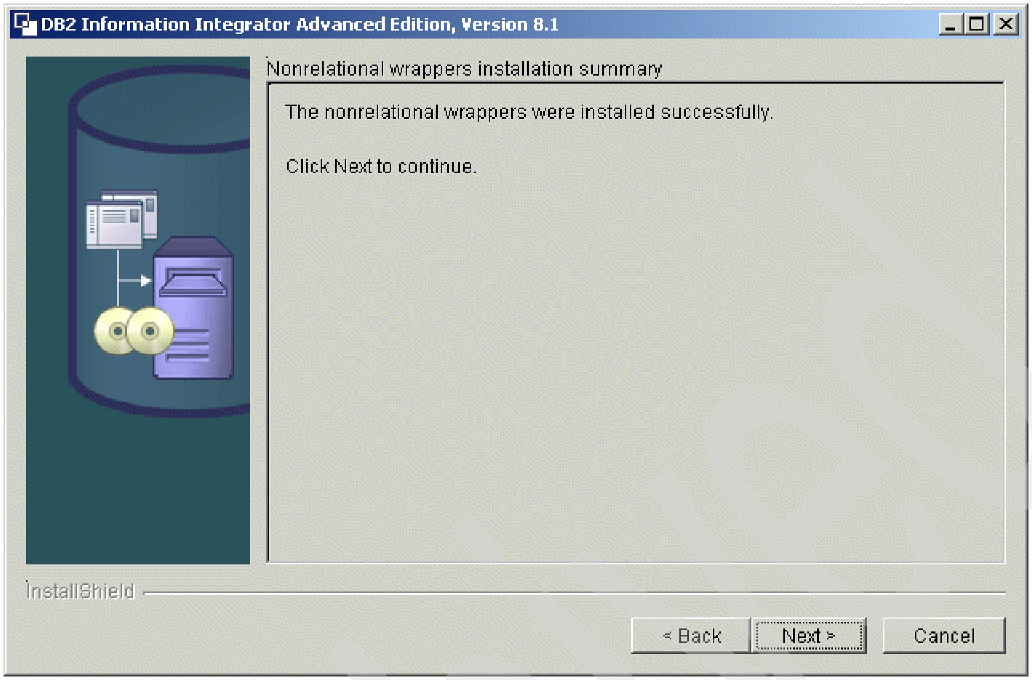


Figure A-61 Non relational wrappers setup - Status

Figure A-61 indicates that the nonrelational wrappers were installed successfully.

Click **Next** to continue to Figure A-62, which is the last screen in the installation process of DB2 Information Integrator.

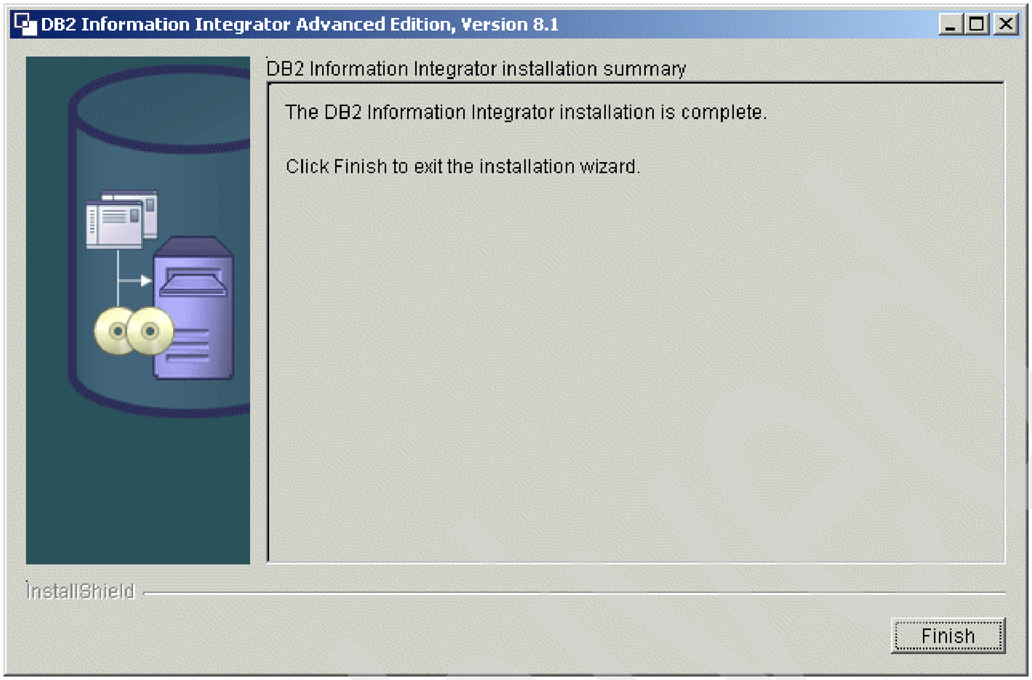


Figure A-62 DB2 Information Integrator installation - Complete

Click **Finish** to end the installation.

A.4 Post-installation steps

After the federated server (DB2 II and wrappers) has been installed, the following steps need to be performed before DB2 II can be used in SQL applications:

1. Verify relevant data source environment variables are set.
2. Verify dbm **`FEDERATED`** configuration parameter is set to `YES`.
3. Create the federated database.

A.4.1 Verify relevant data source environment variables are set

When you set up the federated server, the installation process attempts to set the environment variables for the Documentum, Informix, Oracle, Microsoft SQL Server, Sybase, and Teradata data sources:

- On UNIX you need to check the environment variables for Microsoft SQL Server, Sybase, and Teradata data sources.

- For Oracle, Informix, and Documentum data sources, you need to check the environment variables on both UNIX and Windows operating systems.

Verify that the environment variables for the data sources you want to access are set in the sqllib\cfg\db2dj.ini file — the sqllib\cfg\ directory is located in the directory where you installed DB2.

The valid environment variables for each data source are listed in Table A-2.

Table A-2 Valid data source environment variables

Data source	**Valid environment variables**
Informix	INFORMIXDIR INFORMIXSERVER INFORMIXSQLHOSTS (optional) CLIENT_LOCALE (optional) DB_LOCALE (optional) DBNLS (optional)
Oracle	ORACLE_HOME ORACLE_BASE ORA_NLS (optional) TNS_ADMIN (optional)
Microsoft SQL Server	DJXODBCTRACE DJX_ODBC_LIBRARY_PATH ODBCINI DB2LIBPATH DB2ENVLIST
Sybase	SYBASE SYBASE_OCS
Teradata	COPERR COPLIB

The data source environment variables will *not* be set in the sqllib\cfg\db2dj.ini file if you:

- Install the data source client software *after* the DB2 federated server is set up.
- Have *not* installed the data source client software.

> **Note:** If you change any settings in the *db2dj.ini* file, restart the DB2 instance to ensure that your changes are accepted.

A.4.2 Verify dbm FEDERATED configuration parameter is set to YES

The **FEDERATED** database manager configuration parameter must be set to **YES** to enable access to the data sources. It is possible that this parameter was set when you created the DB2 instance. However, it is important to make certain that the **FEDERATED** parameter set to **YES**.

To check the **FEDERATED** parameter setting, issue the following DB2 command:

```
get dbm cfg
```

This will display all the database manager configuration parameters as shown in Example A-1.

Example: A-1 Partial list of database manager configuration parameters

```
db2 => get dbm cfg

          Database Manager Configuration

     Node type = Enterprise Server Edition with local and remote clients

 Database manager configuration release level            = 0x0a00

 Maximum total of files open               (MAXTOTFILOP) = 16000
 CPU speed (millisec/instruction)             (CPUSPEED) = 6.140476e-007
 Communications bandwidth (MB/sec)      (COMM_BANDWIDTH) = 1.000000e+000

 Max number of concurrently active databases     (NUMDB) = 8
 Data Links support                          (DATALINKS) = NO
 Federated Database System Support           (FEDERATED) = NO
 Transaction processor monitor name        (TP_MON_NAME) =
......
```

Verify that the Federated Database System Support (**FEDERATED**) parameter is set to **YES**.

If the **FEDERATED** parameter is set to **NO**, issue the following command to set it to **YES**:

```
update dbm cfg using federated yes
```

A.4.3 Create the federated database

Create a database to be the federated database through the Control Center or the command line.

To create the federated database through the Control Center, launch Control Center and expand the node where you installed DB2 II. Right-click the **Databases** folder, select **Create,** and then select **Database Using Wizard** as shown in Figure A-63.

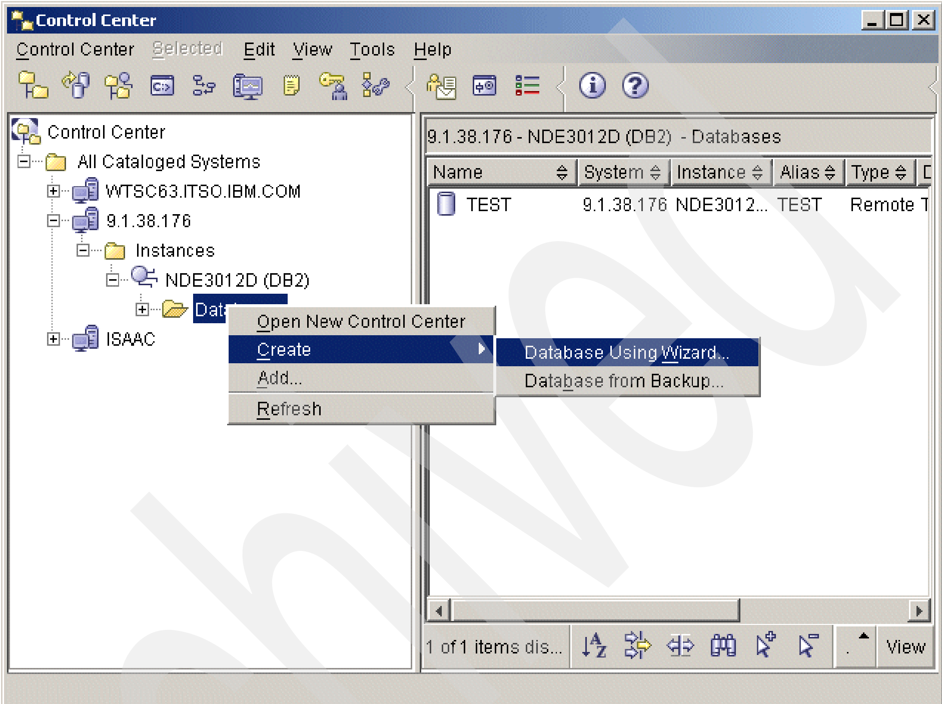


Figure A-63 Post installation steps - Create federated database

This action displays the screen shown in Figure A-64.

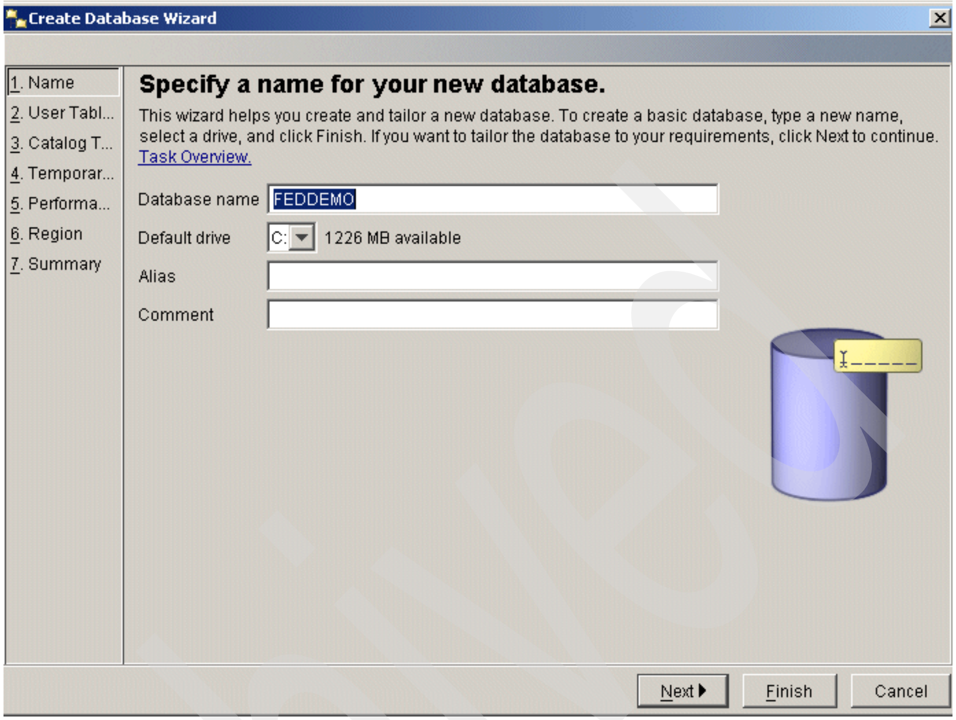


Figure A-64 Post installation steps - Create database - enter name

Figure A-64 provides the capabilities to define the name of the federated database, the default drive on which to create it, etc. We chose **FEDDEMO** as the database name, and clicked **Finish** to create the database.

> **Attention:** The federated database can be tailored by clicking **Next** and supplying custom values.

To create a database with default options through the command line, issue the following command:

```
create database FEDDEMO
```

B

Configuring data sources in DB2 Information Integrator

In this appendix we describe the DB2 Information Integrator Version 8.1 configuration of data sources used in the CFS portal discussed in Chapter 5, “WebSphere Portal & DB2 Information Integrator scenarios” on page 87.

The topics covered are:

- Oracle 9i data source
- XML data source
- Table-structured files data source
- Microsoft Excel data source
- ODBC data source
- WebSphere MQ data source
- DB2 UDB for z/OS data source
- Lotus Extended Search data source

B.1 Introduction

DB2 II Version 8.1 supports a wide variety of data sources with many options to customize each data source configuration. Full complete details on all the supported data sources and the options available are documented in *IBM DB2 Information Integrator Federated Systems Guide*, SC18-7364, and refer to *IBM DB2 Information Integrator Data Source Configuration Guide Version 8*, available as softcopy from the Web site:

http://www.ibm.com/software/data/integration/solution

In this appendix, we focus only on the configuration of data sources used in the CFS portal discussed in Chapter 5, “WebSphere Portal & DB2 Information Integrator scenarios” on page 87.

Refer to Figure 3-7 on page 49, which highlights the basic steps involved in configuring a data source.

In the following subsections, we will briefly describe the steps involved in configuring the following data sources used in the CFS portal:

- Oracle 9i data source
- XML data source
- Table-structured files data source
- Microsoft Excel data source
- ODBC data source
- WebSphere MQ data source
- DB2 UDB for z/OS data source
- Lotus Extended Search data source

> **Note:** The following configurations assume a DB2 II Windows install.

B.2 Oracle 9i data source

This section describes all the steps involved in configuring an Oracle 9i data source, and creating a nickname for a database object on it.

The basic steps for configuring a data source are as shown in Figure 3-7 on page 49, and the following steps describe the configuration of an Oracle 9i data source:

1. Install and test the Oracle Client connection.
2. Create the Oracle wrapper.
3. Create the Oracle server definition.
4. Create the Oracle user mappings.
5. Create the Oracle nickname.

6. Test the Oracle nickname.

We used the DB2 Control Center on the Windows platform to configure the Oracle 9i data source.

B.2.1 Install and test the Oracle Client connection

Verify that the following actions have been completed successfully:

1. Oracle Client is installed on the federated server, and it has been successfully configured and tested to connect to the Oracle server using Oracle Client utilities like SQLPlus or the Oracle Enterprise Management Console.

   For more information on using the Oracle Client, please refer to the documentation that comes with the Oracle Client.

2. Ensure that there is an entry in the Oracle tsnames.ora[1] defining the parameters of the target Oracle Server that the federated server will be accessing. Example B-1 shows the contents of our tnsnames.ora file.

Example: B-1 The tnsnames.ora file

```
ITSO1 =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 9.1.38.196)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = itso1.ibm)
    )
  )
```

The ITSO1 entry at the beginning of the tnsnames.ora file in Example B-1 is called the *Oracle Network Service Name*. This is the value that will be used as the NODE in our server definition in B.2.3, "Create the Oracle server definition" on page 229.

Test the connection to Oracle as follows:

```
c:\>SQLPLUS user/password@ITSO1
```

where the user ID password is valid for the Oracle system.

[1] This is a configuration file that contains the information needed by the Oracle Client to connect to the Oracle server. This file is usually located in the \network\admin sub-directory of the installed directory of the Oracle Client on both Windows and UNIX.

B.2.2 Create the Oracle wrapper

Navigate to the **FEDDEMO** federated database that was created during the DB2 II install as described in A.4.3, “Create the federated database” on page 222, and right-click the **Federated Database Objects** and click **Create Wrapper** as shown in Figure B-1.

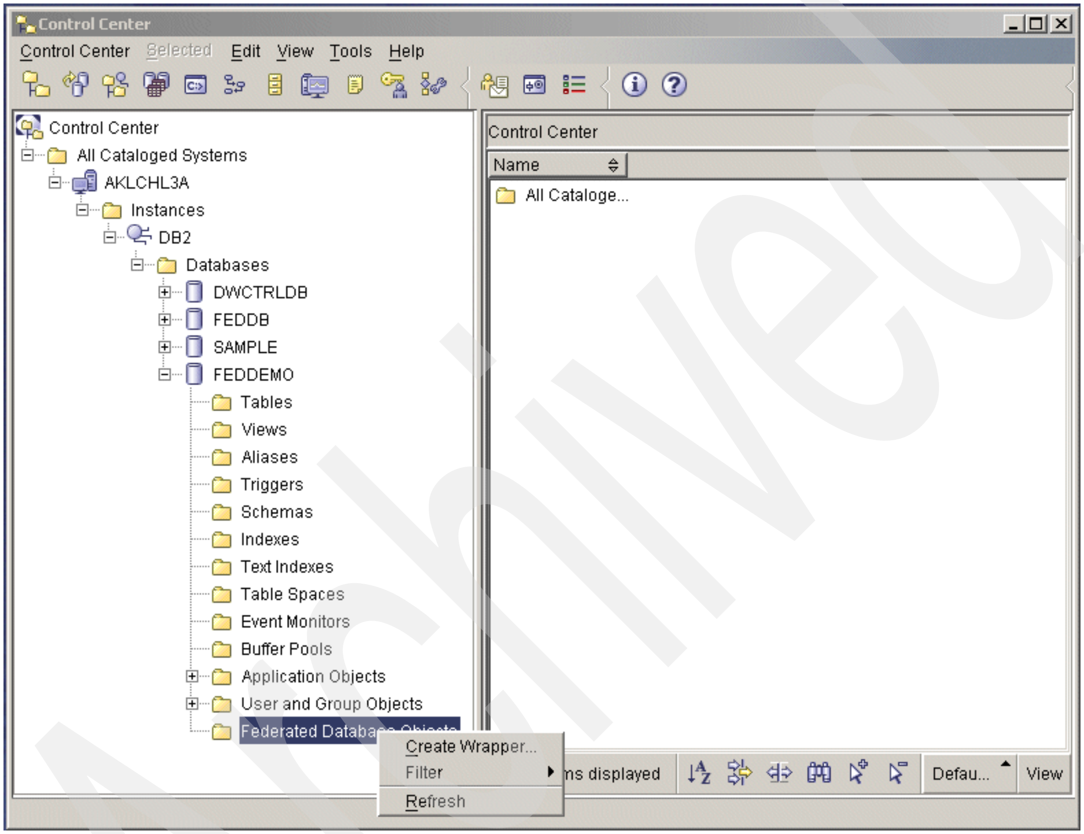


Figure B-1 DB2 Control Center navigation to Federated Database Objects

This action displays Figure B-2.

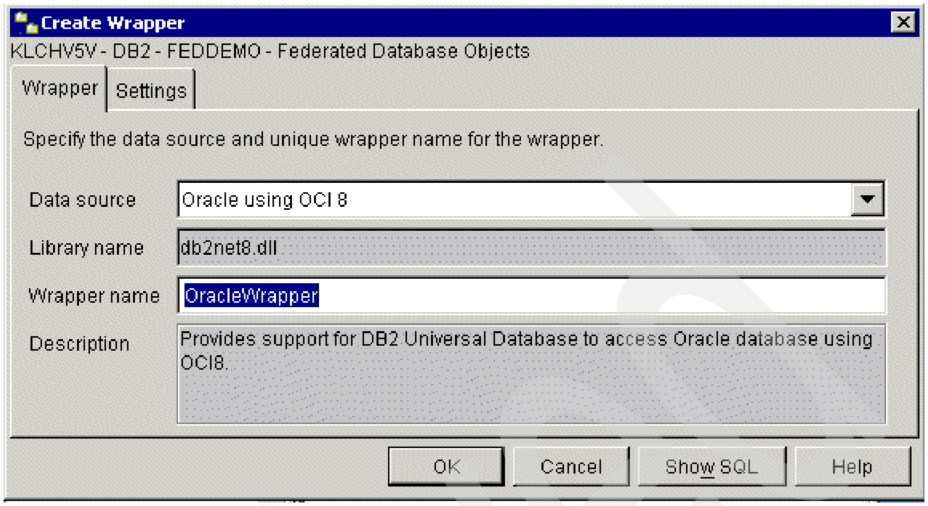


Figure B-2 Oracle - Create Wrapper

If your target Oracle Server is Version 8i or higher, select **Oracle using OCI 8**, otherwise choose **Oracle using OCI 7**. Since our Oracle Server was Oracle 9i, we selected **Oracle using OCI 8**. For Wrapper name, we chose **OracleWrapper** as shown in Figure B-2.

Click **OK** to create the Oracle wrapper.

Example B-2 shows the command line version for creating the OracleWrapper wrapper.

Example: B-2 Create wrapper statement for Oracle

```
CONNECT TO FEDDEMO;
CREATE WRAPPER "ORACLEWRAPPER" LIBRARY 'db2net8.dll';
CONNECT RESET;
```

B.2.3 Create the Oracle server definition

After creating the OracleWrapper that just specifies the type and version of the Oracle server, create the server definition for OracleWrapper as follows:

Select and expand the **ORACLEWRAPPER,** right-click the **Servers** folder and click **Create** as shown in Figure B-3.

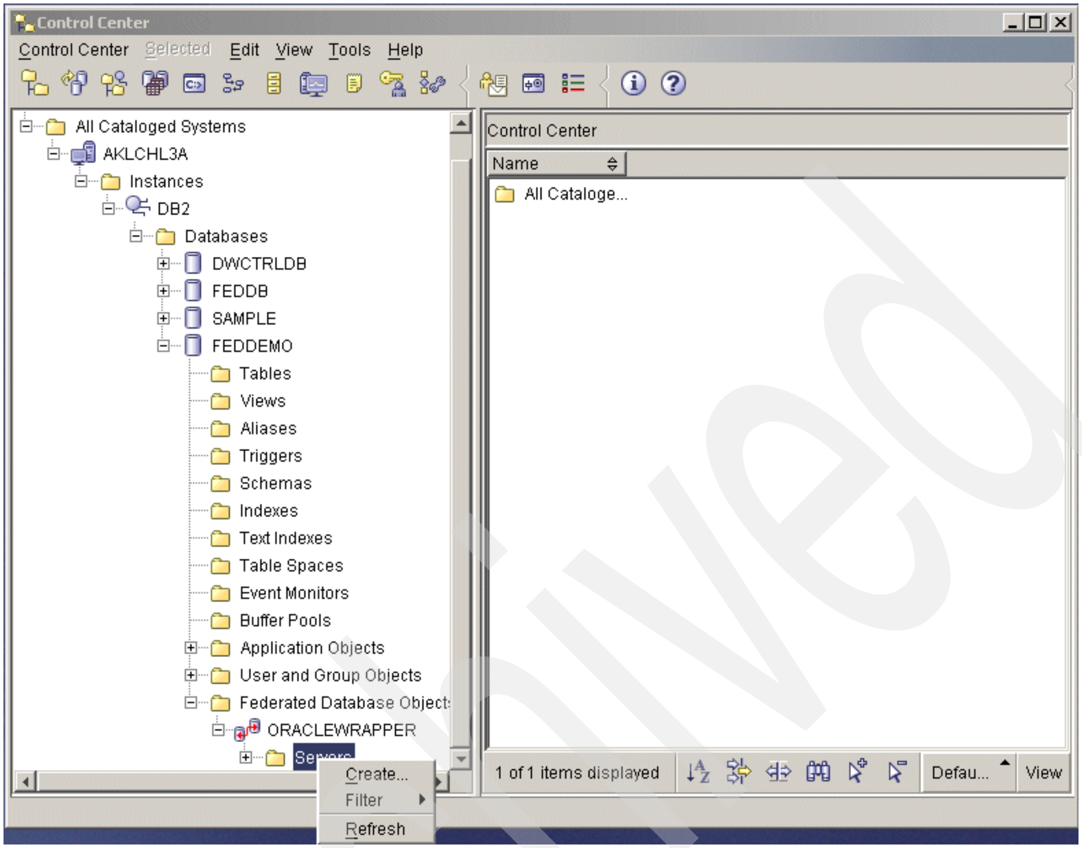


Figure B-3   Server definition for ORACLEWRAPPER

This action will display Figure B-5.

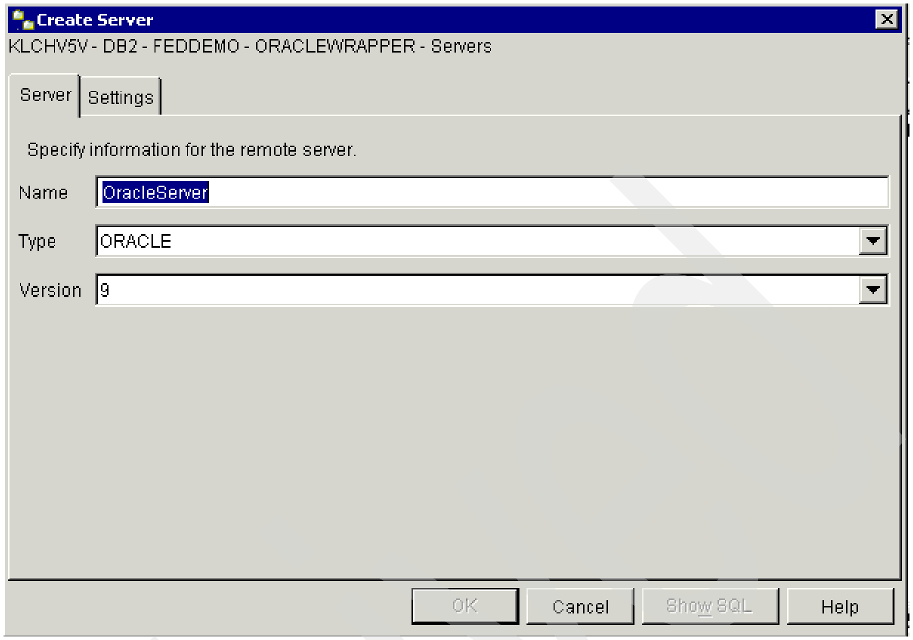


Figure B-4   Oracle - Create Server dialog

The server definition requires the following:

- **Name**: The name of the server definition must be a unique within the federated database.

  In our case we entered `ORACLESERVER`.

- **Type**: Select **ORACLE.**
- **Version**: Select **8** or **9.** If you defined an Oracle wrapper using OCI 7, select the right version of your Oracle data source server.

  In our case we selected Version **9** as shown in Figure B-4.

Click the **Settings** tab to complete the server definition as shown in Figure B-5.

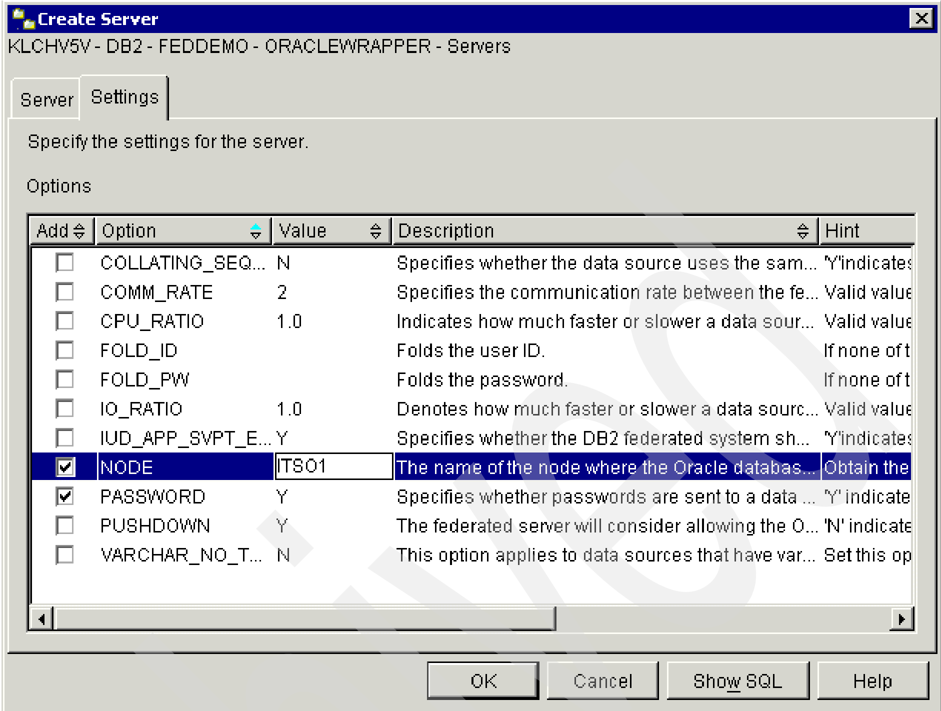


Figure B-5 Oracle - Create Server - Settings

In Figure B-5 the only required options for the server settings are those for **Node** and **Password**.

- **Node** was set to `ITS01`: The value from our *tnsnames.ora* file is in Example B-1 on page 227.
- Set the **Password** parameter to `Y`.

A number of server options are available to describe a data source server; the set of options applicable to Oracle servers are listed in Table B-1. These server options may be set at creation time or modified later as described in "Altering the Oracle nicknames" on page 243.

Note: There can be multiple server definitions for each wrapper.

Example B-3 shows the command line version of creating the ORACLESERVER server definition.

Example: B-3 Oracle - Create server statement

```
CONNECT TO FEDDEMO;
CREATE SERVER ORACLESERVER TYPE ORACLE VERSION '9' WRAPPER "ORACLEWRAPPER"
OPTIONS( ADD NODE 'ITSO', PASSWORD 'Y');
```

Table B-1 Oracle server options

Connectivity	Node	It identifies the entry name (Network Service Name) in Oracle *tnsnames.ora* file, and is case sensitive.
	Password	**Default=Y**: Include the password on connections to Oracle
	Fold_ID / Fold_PW	**Default** is wrapper dependent: **Y**: Connect four times with user ID/password with all combinations of lower and uppercase. **N** (recommended): Connect once with user ID/password exactly as specified **U**: Connect once with user ID/password in uppercase **L**: Connect once with user ID/password in lower case
Pushdown	Collating_Sequence	It specifies whether the data source uses the same default collating sequence as the federated database. This affects the pushdown of operations on character columns. **Default=N** **Y**: Both use same collating sequence. ORDER BY can be pushed down without compromising integrity of result **N**: Both use different collation sequence. ORDER BY cannot be pushed down **I**: case-insensitive. ORDER BY, DISTINCT, WHERE= cannot be pushed down
	Pushdown	**Default: Y**: The SQL operations are pushed down to data sources based on the decision of the pushdown analyzer and optimizer.

Optimization	CPU_RATIO	**Default: 1.0**: Specifies the ratio of the DB2 Information Integrator server CPU capacity against the data source CPU capacity
	IO_RATIO	**Default: 1.0**: Specifies the ratio of the DB2 Information Integrator server I/O rate against the data source I/O rate
	COMM_RATE	**Default: 2**: Specifies the effective data rate of network to data source in MB per second.
Other	IUD_APP_SVPT_ENFORCE	**Default=Y**: Should the DB2 federated system use save-points in multi-update transactions?
	VARCHAR_NO_TRAILING_BLANKS	**Default=N**: This option applies to variable character data types that do not pad the length with trailing blanks. Set this option to **Y**, when none of the columns contains trailing blanks. If only some of the VARCHAR columns contain trailing blanks, you can set an option with the ALTER NICKNAME statement.

B.2.3.1 Altering the Oracle server definitions and options

You may modify a server definition when you:

- Upgrade a data source to a new version, for example:

  ```
  alter server ORACLESERVER version 9
  ```

- Want to modify the server options

  Server options are set to values that persist over successive connections to the data source. The values are stored in the federated system catalog. The following example activates (add), sets and deactivates (drop) a server option:

  ```
  alter server ORACLESERVER options (add IO_RATIO ‘2.0)
  alter server ORACLESERVER option (set IO_RATIO ‘3.0’)
  alter server ORACLESERVER option (drop IO_RATIO)
  ```

To temporarily set a server option for the duration of a single connection to the federated database, use the **SET SERVER OPTION** statement as follows:

```
set server option COMM_RATE TO ‘3’ for server ORACLESERVER
```

The settings are not stored in the federated system catalog.

> **Attention:** The server options shown in Table B-2 are *not* available through the DB2 Control Center, they must be set through the command line.

Table B-2 Oracle additional server options

DB2_MAXIMAL_PUSHDOWN	It specifies the primary criteria for the optimizer in choosing the access plan. The optimizer can choose between cost optimization and the user requirement to perform query processing by the remote data source as much as possible. ‘**Y**’: Choose the plan with most query operations to be pushed down to the data sources. ‘**N**’: Choose the plan with minimum cost.
PLAN_HINTS	Hints are statement fragments that provide extra information for the Oracle optimizer. ‘**Y**’: Enabled ‘**N**’: Disabled

B.2.4 Create the Oracle user mappings

As mentioned earlier, the user mapping defines an association between a user ID on the federated server and a user ID on the Oracle server. This user mapping is used by the federated database server whenever it connects to the Oracle server on behalf of the calling federated database user. An association must be created for each user that would be using the federated system.

In our case, we define a single user mapping for our **ADMINISTRATOR** user since that is the only user ID used in our CFS portal.

Select and expand **ORACLESERVER**, right-click the **User Mappings** folder, and click **Create** as shown in Figure B-6.

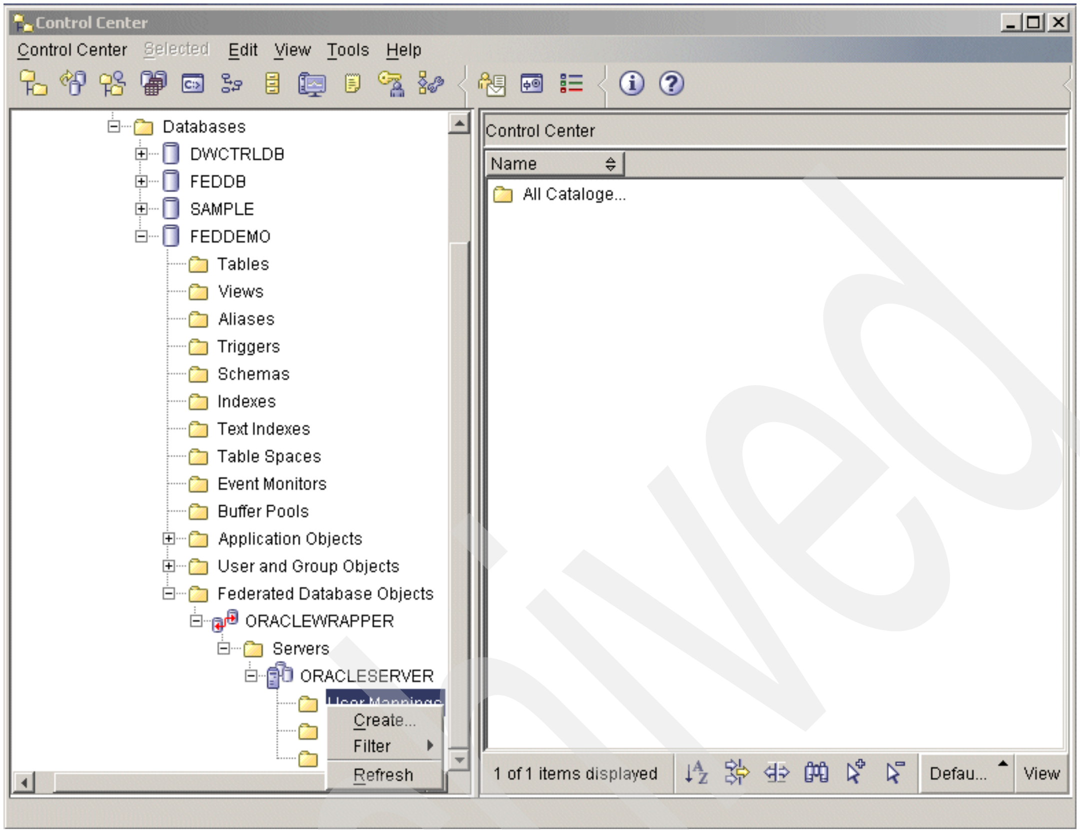


Figure B-6 User mapping for ORACLESERVER

This action displays the view shown in Figure B-7.

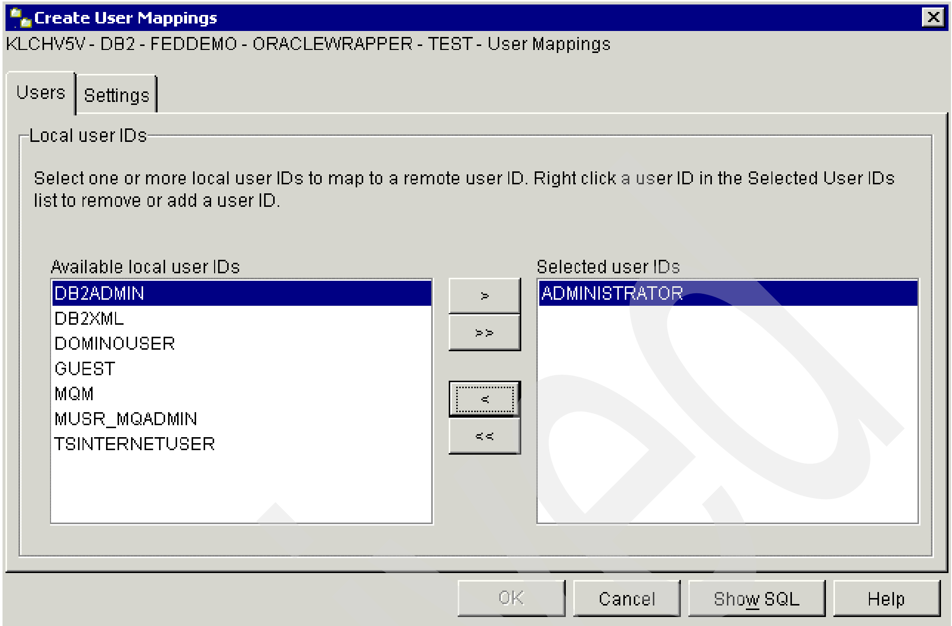


Figure B-7 Oracle - Create User Mappings

Figure B-7 lists all the user IDs available on our federated system. Select the user that sends the federated requests to the Oracle data source.

We selected the **Administrator** user, and clicked the **Settings** tab to view Figure B-8.

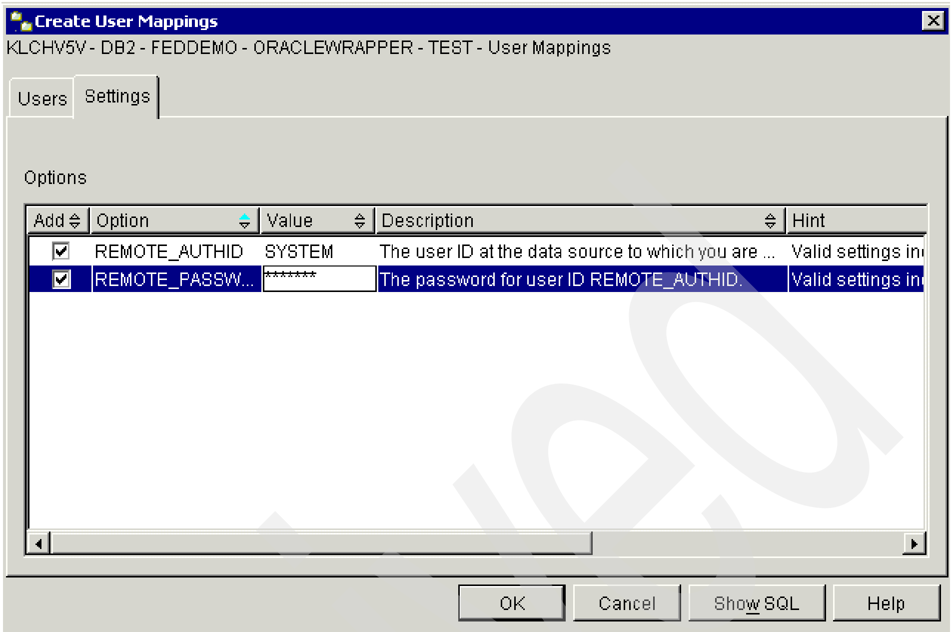


Figure B-8 Oracle - Create User Mappings - Settings

In Figure B-8 you need to enter a valid user ID and password to connect to the Oracle data source to complete the user mapping. Click **OK** to complete the user mapping.

User mappings may be modified after creation time as described in “Altering the Oracle user mappings” on page 238.

Example B-4 shows the command line version of creating our user mapping for our Oracle instance.

Example: B-4 Oracle - Create user mapping statement

```
CONNECT TO FEDDEMO;
CREATE USER MAPPING FOR "ADMINISTRATOR" SERVER "ORACLE9" OPTIONS ( ADD
REMOTE_AUTHID 'Administrator', ADD REMOTE_PASSWORD '********');
CONNECT RESET;
```

B.2.4.1 Altering the Oracle user mappings

You can use the **ALTER USER MAPPING** statement to modify a user mapping, and it is used to change the authorization ID or password, which is used at the Oracle data source as follows:

```
alter user mapping for Administrator SERVER ORACLESERVER options
```

```
(set remote_authid 'Administrator')
alter user mapping for Administrator server ORACLESERVER options
(set remote_password 'newpass')
```

> **Note:** The REMOTE_AUTHID and REMOTE_PASSWORD user options are both case sensitive unless you set the FOLD_ID and FOLD_PW server options to `U` or `L` in the **`CREATE SERVER`** statement.

User mappings may also be altered through the DB2 Control Center.

B.2.5 Create the Oracle nickname

Once the Oracle wrapper, the server definition, and the user mapping are complete, one needs to test the connection to the data source.

After successfully testing the connection to the data source (not shown here), one may create nicknames for the database objects on the data source.

One may choose to define data mappings and function mappings during the creation of a nickname. Data mappings are described in "Data type mappings" on page 52, while function mappings are described in "Function mappings" on page 53.

> **Note:** We did not define any data mappings or function mappings for our nicknames.

When you create a nickname for an Oracle table, catalog data from the remote server is retrieved and stored in the federated global catalog.

Select and expand **ORACLESERVER,** right-click **Nicknames,** and click **Create** as shown in Figure B-9.

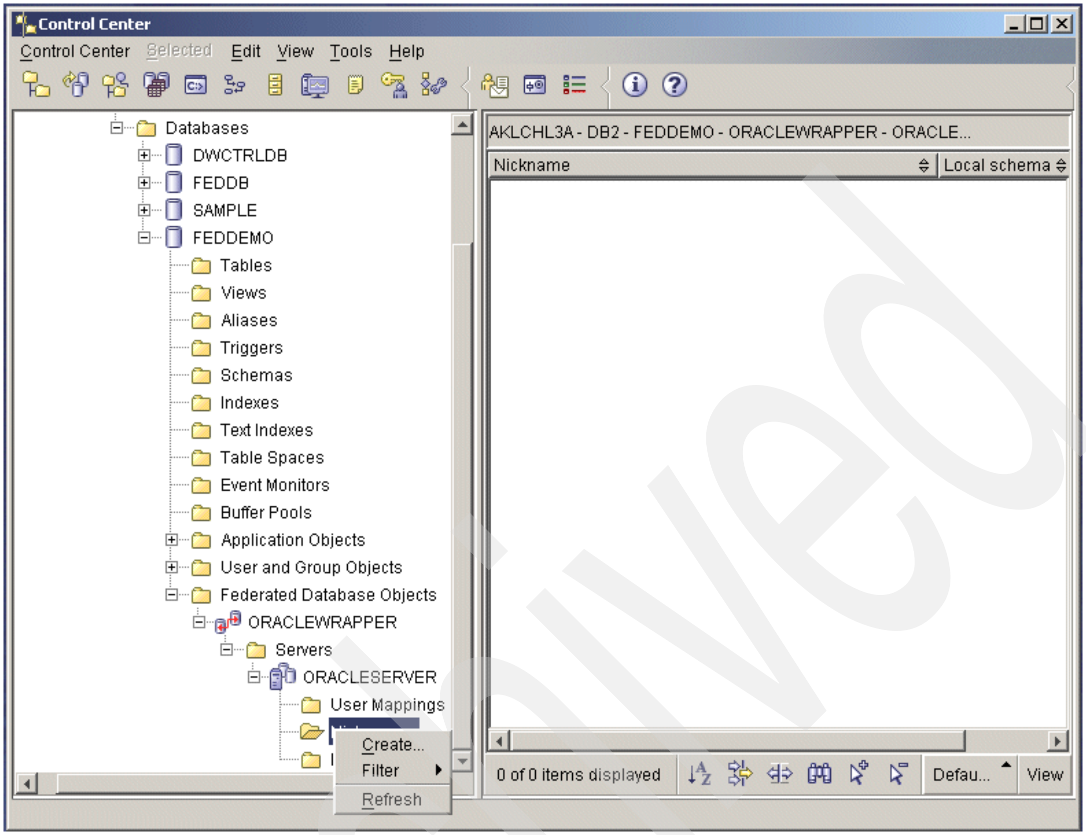


Figure B-9 Nickname creation for ORACLESERVER

This action opens up a dialog window as displayed in Figure B-10.

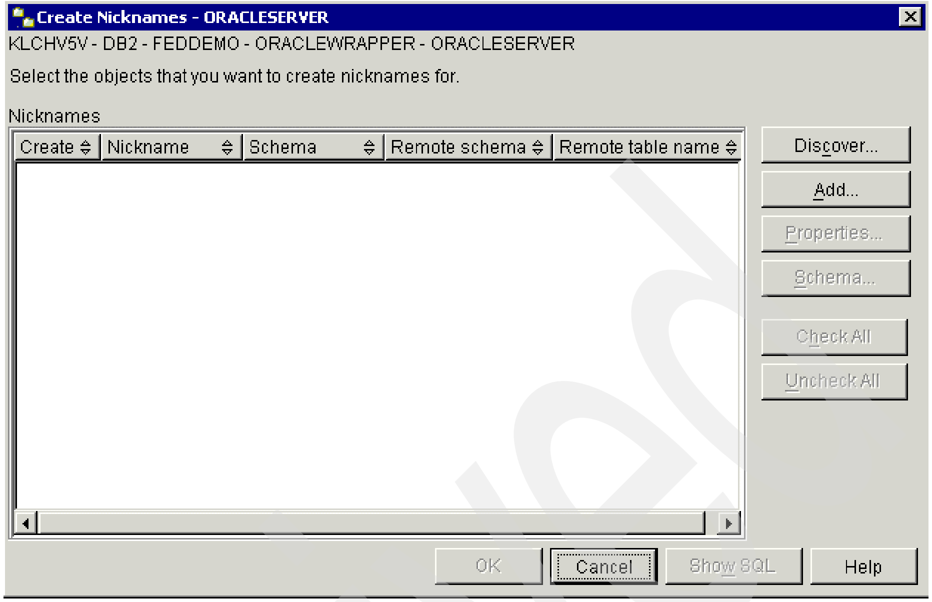


Figure B-10 Oracle - Create Nicknames

Figure B-10 shows two options for adding a nickname:

- Manually add a nickname (click **Add**) by specifying local and remote schema and table identification.
- Use the discover functionality (click **Discover)** to be prompted for providing all the required information as follows:

  Click **Discover** to open up a dialog window to filter out the results as shown in Figure B-11.

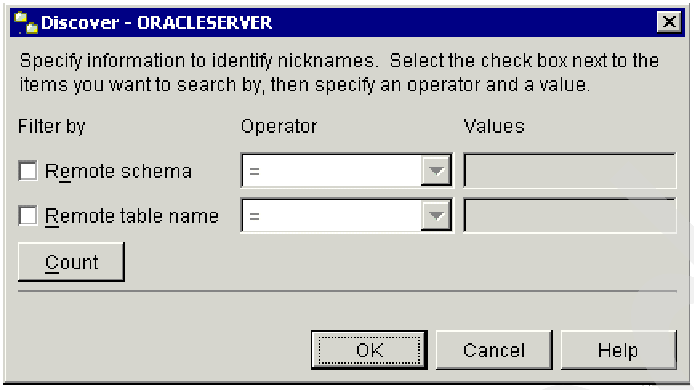


Figure B-11 Oracle - Create Nicknames - Discover with filter

Choose a **filter** method, an **operator**, and the comparing **value,** and click **Count.** Upon successful access to the Oracle database, the number of objects that meet the filter criteria will be displayed. Adjust your filter criteria if you find the number is too few or many.

For our scenario we had a table called **POLICY_APP** in our Oracle Server, so our filter and matching criteria was adjusted (not shown here) to list only this table. Click **OK** to list the actual objects as a list in the dialog as shown in Figure B-12.

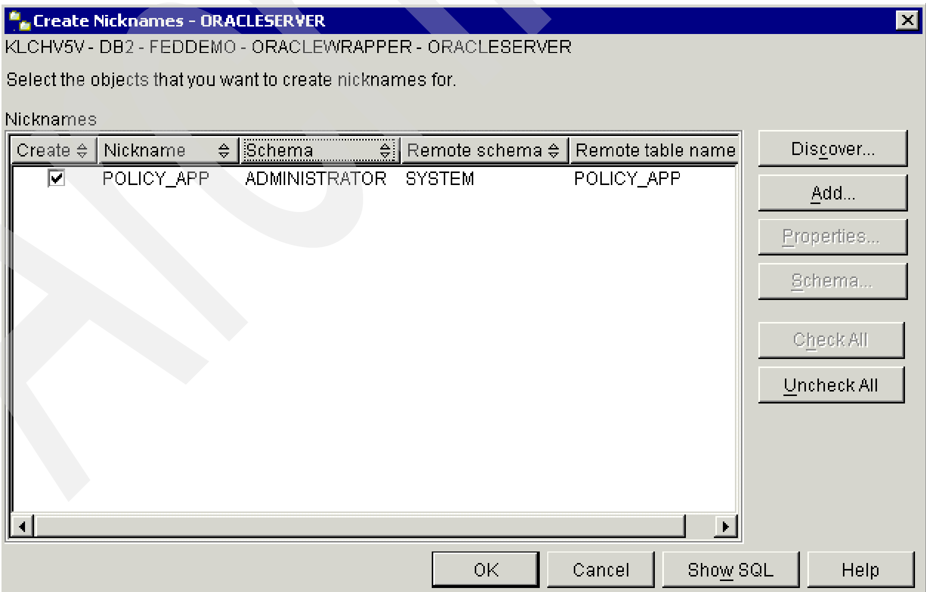


Figure B-12 Create nickname - list selected table

With the **Discover** filter, the default schema for the entries added to the **Create Nickname** window is the user ID that is creating the nicknames. In our case it is **ADMINISTRATOR**.

To change any of the properties of the nickname, click **Properties** and perform the appropriate modifications (not shown here since we did not make any changes).

Click **OK** to create the nickname.

> **Attention:** Once a nickname is created, its name and schema cannot be modified without dropping and recreating it. However, some other changes can be made as described in "Altering the Oracle nicknames" on page 243.

Example B-5 shows the command line version of creating the **POLICY_APP** nickname for ORACLESERVER.

Example: B-5 Oracle - Create nickname statements

```
CONNECT TO FEDDB;
CREATE NICKNAME ADMINISTRATOR.POLICY_APP FOR ORACLESERVER.SYSTEM.POLICY_APP;
CONNECT RESET;
```

B.2.5.1 Altering the Oracle nicknames

The ALTER NICKNAME statement can be used to modify the federated database representation of a data source as follows:

- Change the local data type of a column:

```
ALTER NICKNAME "ADMINISTRATOR"."POLICY_APP" ALTER COLUMN LNAME LOCAL
TYPE VARCHAR (50);
```

- Change the local column name:

```
ALTER NICKNAME "ADMINISTRATOR"."POLICY_APP" ALTER COLUMN LNAME LOCAL
NAME LASTNAME;
```

B.2.6 Test the Oracle nickname

To test the nickname right-click it and click **Sample Contents** as shown in Figure B-13.

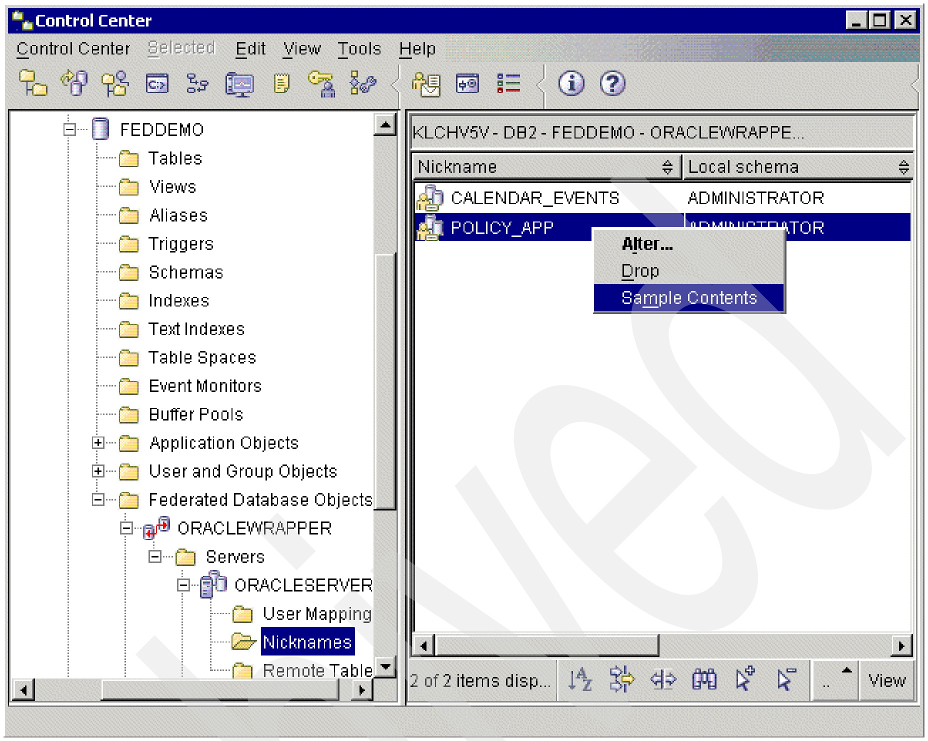


Figure B-13 Oracle - sample contents of nickname

This action should display the contents of the database object referenced by the nickname if nickname creation was successful. We have not displayed the contents here.

B.3 XML data source

This section describes all the steps involved in configuring an XML data source, and creating a nickname on it.

The basic steps for configuring a data source are as shown in Figure 3-7 on page 49, and the following steps describe the configuration of an XML data source:

1. Create the XML wrapper.
2. Create the XML server definition.
3. Create the XML nickname.
4. Test the XML nickname.

We used the DB2 Control Center on the Windows platform to configure the XML data source.

B.3.1 Create the XML wrapper

The XML wrapper can be used in the following cases:

- The user wants to join XML data with other nicknames or other relational data.
- The user wants to keep the original XML intact (to avoid replicating data or data that may change often, or is composed on the fly).

The XML document used in a nickname may come from any of the following sources:

- A text file: for example, c:\myfiles\mydata.xml
- A directory of text files: for example, c:\myfiles
- An XML document stored in a column of a table in a database
- From a URI: for example: `http://www.mysite.com/mydata.xml`

The XML wrapper supports the following data types: INTEGER, SMALLINT, FLOAT, REAL, DECIMAL, CHAR, VARCHAR, and DATE.

> **Note:** The XML wrapper does not support the INSERT, UPDATE, DELETE functions.

Ensure that the following prerequisites are in place before deciding to use the XML wrapper:

- Data files must be accessible to the DB2 Information Integrator server. They must reside on the same machine or on a network accessible shares and mounts.
- The user ID that runs DB2 (in our case, `ADMINISTRATOR`) must be able to open and read the XML file for which the nickname is being created. Refer to "Key technical considerations in a Windows environment" on page 72 for a brief discussion on accessing files on network drives.

Navigate to the **FEDDEMO** federated database that was created during the DB2 II install as described in A.4.3, "Create the federated database" on page 222, and right-click the **Federated Database Objects** and click **Create Wrapper** as shown in Figure B-1.

This action displays Figure B-14.

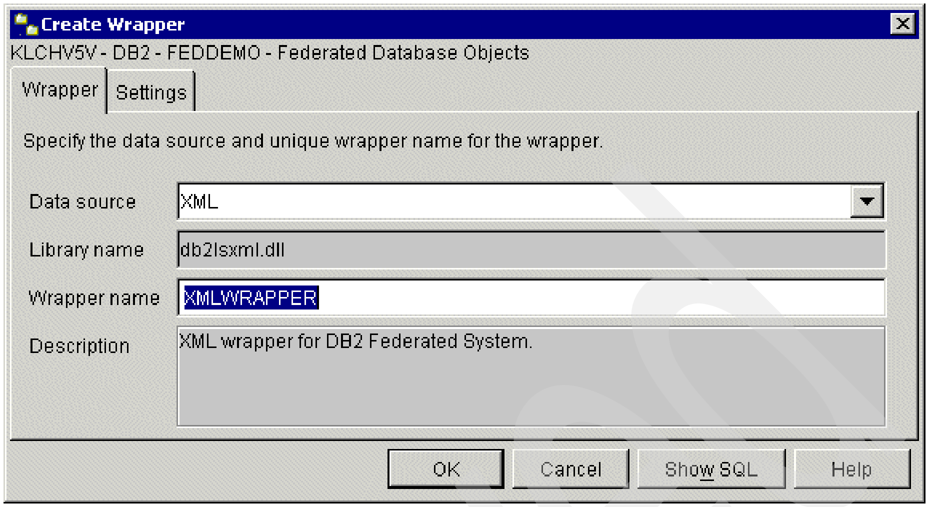


Figure B-14 XML - Create Wrapper

Select **XML** in the **Data source** field, and type `XMLWRAPPER` in the `Wrapper name` field in Figure B-14.

Click the **Settings** tab to view Figure B-15.

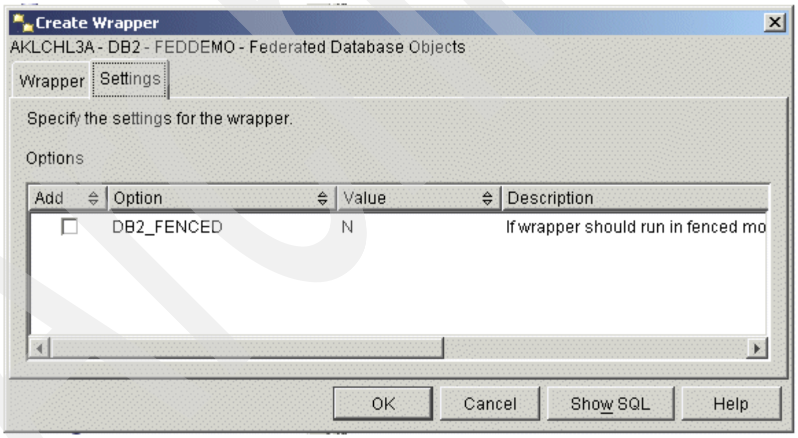


Figure B-15 XMLWRAPPER Settings option

Click **OK** to create the **XMLWRAPPER** wrapper.

Example B-6 shows the command line version of creating the XMLWRAPPER wrapper.

Example: B-6 XML - Create wrapper statement

```
CONNECT TO FEDDEMO;
CREATE WRAPPER "XMLWRAPPER" LIBRARY 'db2lsxml.dll';
CONNECT RESET;
```

B.3.2 Create the XML server definition

For XML, the DB2 Information Integrator server definition is required even though it does not require information such as the version and connection information needed in the server definition for relational data sources.

Select and expand the **XMLWrapper**, right-click the **Servers** folder, and click **Create** similar to the actions taken for the **ORACLEWRAPPER** in Figure B-3.

This action will display Figure B-16.

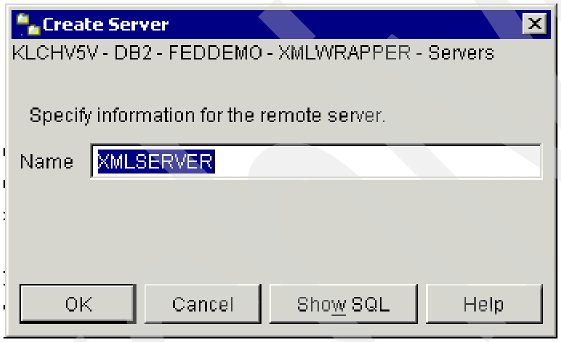


Figure B-16 XML - Create Server

The server definition requires the name of the server definition to be unique within the federated database. In our case we entered `XMLSERVER` as shown in Figure B-16.

Click **OK** to complete the XMLSERVER server definition.

For the XML server, the server options cannot be updated, added, or dropped.

> **Note:** There can be multiple server definitions for each defined XML wrapper.

Example B-7 shows the command line version of creating the XMLSERVER server definition.

Example: B-7 XML - Create server statement

```
CONNECT TO FEDDEMO;
CREATE SERVER XMLSERVER WRAPPER "XMLWRAPPER";
CONNECT RESET;
```

B.3.3 Create the XML nickname

After setting up the XML wrapper and the XML server definition, we can create the actual link to an XML file, a directory containing the XML file, or a URI.

> **Attention:** Unlike some of the other data sources, there are no user mappings associated with XML data sources since the federated server user ID must have access to the XML file, or directory.

Example B-8 shows a subset of the PoliceReports.xml file used in the CFS portal. This file resides in the c:\flatfiles directory.

Example: B-8 Police Reports XML file

```
<Reports>
   <Report>
      <VicName>Mitch Lapinsky</VicName>
      <IncidentLoc>Q street at 17th</IncidentLoc>
      <IncidentDesc>
         Mr. Lapinsky's car was stopped at stop sign awaiting clearing to
continue forward. Vehicle B traveling northbound lost control of their car and
proceeded thru the stop sign and struck
         Mr. Lapinsky's car from the front.
      </IncidentDesc>
      <IncidentID>1</IncidentID>
      <Injury>Minor</Injury>
      <MedReq>Yes</MedReq>
      <MedName>Samantha Moore</MedName>
      <MedProc>Oxegen administered</MedProc>
      <MedHospitalization>No</MedHospitalization>
      <MedHospital>N/a</MedHospital>
      <OfficerName>Sgt. Mary A. Cooper</OfficerName>
      <PropDamage>Yes</PropDamage>
      <PropType>Auto</PropType>
      <PropLoc>N/A</PropLoc>
      <PropOwn>Mitch Lapinsky</PropOwn>
      <PropPhone>555-1212</PropPhone>
   </Report>
   <Report>
      <VicName>Mitchel Lapinsky</VicName>
      <IncidentLoc>1403 Bowler Ave</IncidentLoc>
```

```
      <IncidentDesc>
         Sgt. Lapinski was in pursuit of the Christianson residence thief when
the police cruiser he was driving lost its right wheel and spun out of control.
The cruiser slammed into a tree located on Bowler Ave.
      </IncidentDesc>
      <IncidentID>2</IncidentID>
      <Injury>Major</Injury>
      <MedReq>Yes</MedReq>
      <MedName>Bob Symptoms</MedName>
      <MedProc>Traction and hospital transportation</MedProc>
      <MedHospitalization>Yes</MedHospitalization>
      <MedHospital>Camina Med Center</MedHospital>
      <OfficerName>Mark Ravenswood</OfficerName>
      <PropDamage>Yes</PropDamage>
      <PropType> Foliage and Auto</PropType>
      <PropLoc>1403 Bowler Ave, Sacramento </PropLoc>
      <PropOwn>Sven Bailet</PropOwn>
      <PropPhone>555-1213</PropPhone>
   </Report>
</Reports>
```

Select and expand **XMLSERVER,** right-click **Nicknames,** and click **Create** similar to the actions taken for the ORACLESERVER in Figure B-9 on page 240.

This action opens up a dialog window as displayed in Figure B-17.

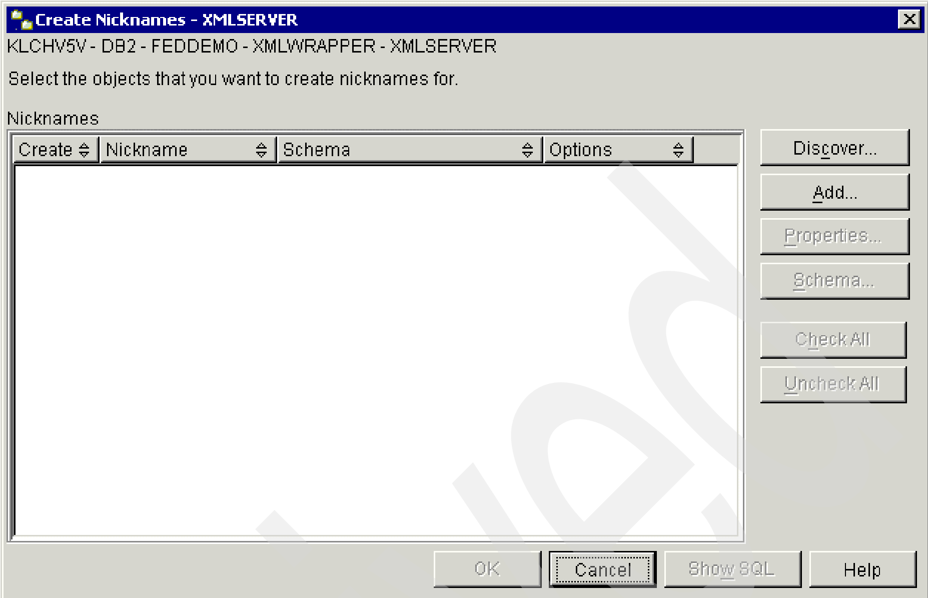


Figure B-17 XML - Create Nicknames

Figure B-17 is similar to the Figure B-10 on page 241, and provides two options for adding a nickname:

- Manually add a nickname (click **Add**) by specifying local schema.
- Use the discover functionality (click **Discover)** to be prompted for providing all the required information as follows:

  Click **Discover** to open up a dialog window shown in Figure B-18.

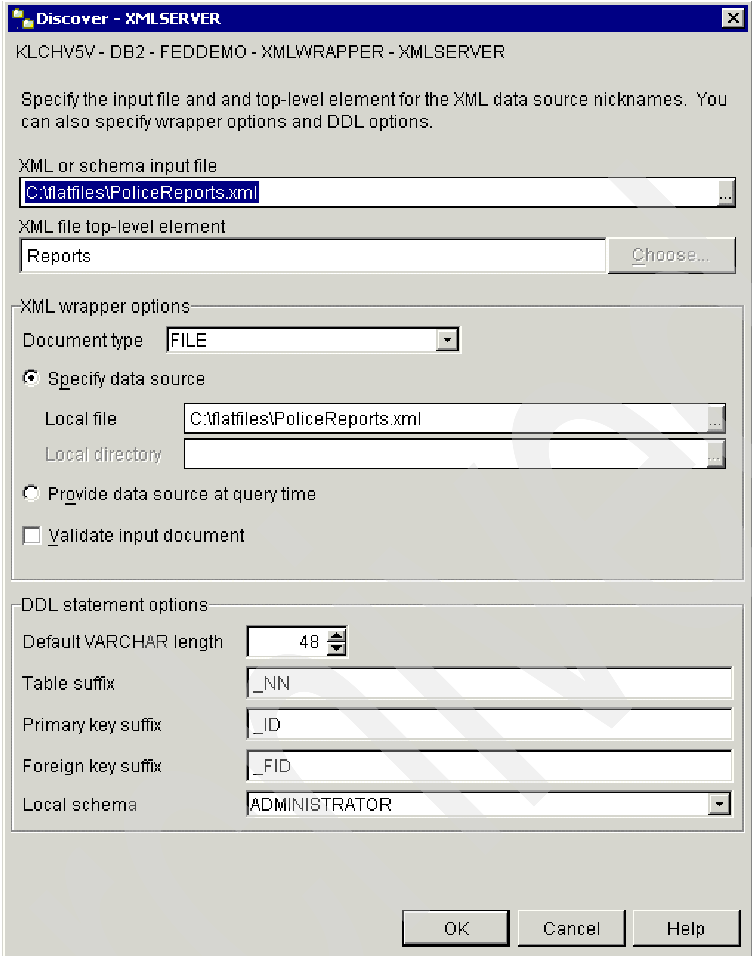


Figure B-18 XML - Create nickname Discover screen

When you click **Discover:**

– An XML schema file or an XML document file can be specified in the XML or schema input file field of Figure B-18.

  If an XML schema file is specified in this field, then specify the XML document file for which the nickname is to be created in the Local file field in the middle of Figure B-18. The schema file specified in the top field must be appropriate for the document file specified in the Document type field.

– The Discover filter can declare the columns of the nicknames to be created. It reads the XML schema file or the XML document file and creates an appropriate nickname definition with elements and attributes in the XML schema, or a document mapped to columns of the nickname. The Discover filter has a default behavior for declaring the data types of the nickname columns:

- If an XML schema is specified in the top field, then the **Discover** filter will use information in the XML schema to find out if any elements and attributes should be mapped to DB2 numeric or date data types; and the remaining “string” columns will be mapped to DB2 VARCHAR with the default length specified in the `Default VARCHAR length` field of the Discover window.
- If an XML file is specified in the top field of the Discover filter, then all elements and attributes of the document are mapped to DB2 VARCHAR data type, and the length is based on the value in the `Default VARCHAR length` field in the Discover window.

In our case, we selected the **PoliceReports.xml** file in the XML or schema input file field. This results in the rest of the fields being filled automatically for you. Click **OK** to populate the Create Nicknames window as shown in Figure B-19.

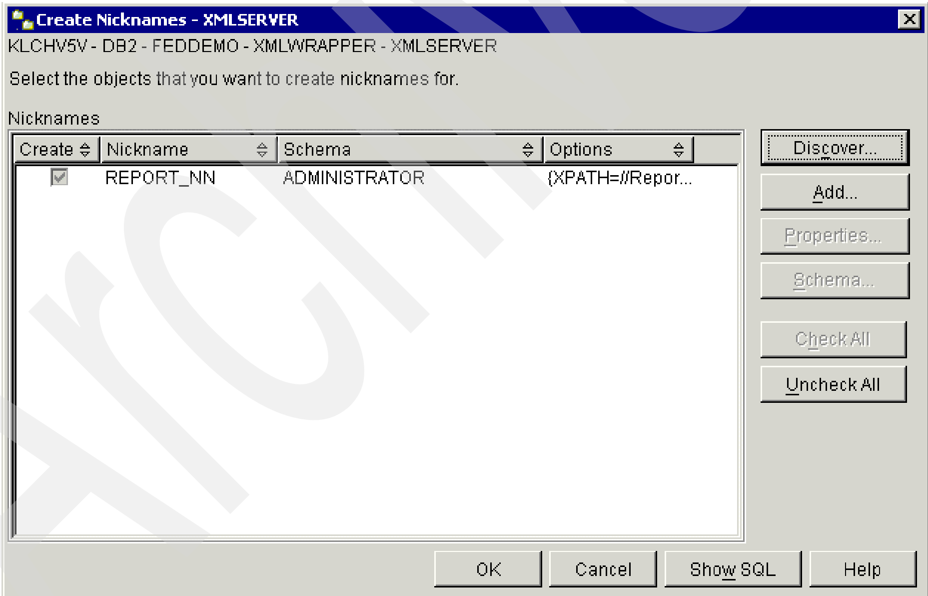


Figure B-19 XML - Create nickname window after discovery

Select the nickname and click **Properties** to view Figure B-20**.**

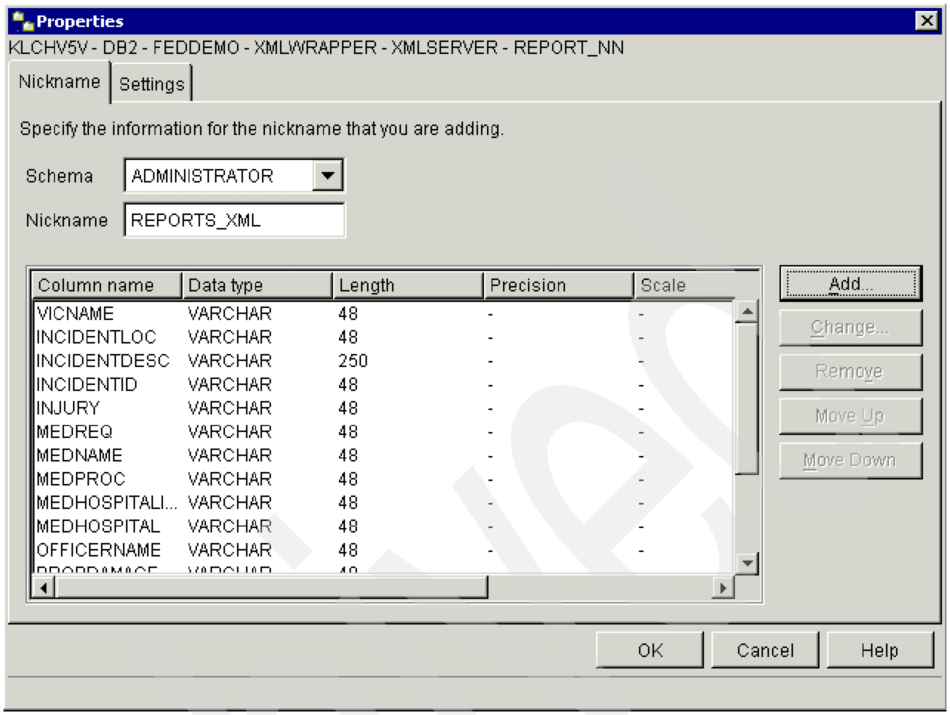


Figure B-20 XML - Create nickname - modify properties

Change the **Nickname** field from **REPORT_NN** to **REPORTS_XML** and also change the length of INCIDENTDESC column from 48 to 250 as shown in Figure B-20. Click **OK** to complete the definition of the nickname.

> **Attention:** Once a nickname is created, its name and schema cannot be modified without dropping and recreating it. However, some other changes can be made as described in Appendix B.3.3.1, "Altering the XML nicknames" on page 254.

Example B-9 shows the command line version of nickname definitions for three of the eight tables derived from the PoliceReports.xml file.

Example: B-9 XML - Create nickname statements

```
CONNECT TO FEDDEMO
CREATE NICKNAME ADMINISTRATOR.REPORT_XML (
      VICNAME VARCHAR (48) OPTIONS(XPATH './VicName/text()'),
      INCIDENTLOC VARCHAR (48) OPTIONS(XPATH'./IncidentLoc/text()'),
      INCIDENTDESC VARCHAR (250) OPTIONS(XPATH './IncidentDesc/text()'),
```

```
INCIDENTID VARCHAR (48) OPTIONS(XPATH './IncidentID/text()'),INJURY
VARCHAR (48) OPTIONS(XPATH './Injury/text()'),
MEDREQ VARCHAR (48) OPTIONS(XPATH './MedReq/text()'),
MEDNAME VARCHAR (48) OPTIONS(XPATH './MedName/text()'),
MEDPROC VARCHAR (48) OPTIONS(XPATH './MedProc/text()'),
MEDHOSPITALIZATION VARCHAR (48) OPTIONS(XPATH
'./MedHospitalization/text()'),
MEDHOSPITAL VARCHAR (48) OPTIONS(XPATH './MedHospital/text()'),
OFFICERNAME VARCHAR (48) OPTIONS(XPATH './OfficerName/text()'),
PROPDAMAGE VARCHAR (48) OPTIONS(XPATH './PropDamage/text()'),PROPTYPE
VARCHAR (48) OPTIONS(XPATH './PropType/text()'),
PROPLOC VARCHAR (48) OPTIONS(XPATH './PropLoc/text()'),
PROPOWN VARCHAR (48) OPTIONS(XPATH './PropOwn/text()'),
PROPPHONE VARCHAR (48) OPTIONS(XPATH './PropPhone/text()'))
FOR SERVER "XMLSERVER"
OPTIONS(XPATH '//Report' , FILE_PATH 'C:\flatfiles\PoliceReports.xml');
CONNECT RESET
```

The **FILE_PATH** option is only used in the top-level nickname, and it defines where the data comes from.

B.3.3.1 Altering the XML nicknames

You can use the ALTER NICKNAME statement to modify the federated database representation of a data source. This statement can be used to:

- Change local data type of a column as follows:

```
ALTER NICKNAME "ADMINISTRATOR"."REPORTS_XML" ALTER COLUMN VICNAME LOCAL
TYPE VARCHAR (50) ;
```

- Change local column name as follows:

```
ALTER NICKNAME "ADMINISTRATOR"."REPORTS_XML" ALTER COLUMN VICNAME LOCAL
NAME VICTIMNAME ;
```

- Add the streaming option (DB2 V8.1 fixpak 3 or later) as follows:

```
ALTER NICKNAME "ADMINISTRATOR"."REPORTS_XML" OPTIONS (ADD STREAMING
'YES')
```

- Drop the streaming option as follows:

```
ALTER NICKNAME "ADMINISTRATOR"."REPORTS_XML" OPTIONS (DROP STREAMING)
```

B.3.4 Test the XML nickname

To test the REPORTS_XML nickname, right-click it and click **Sample Contents** as shown in Figure B-21.

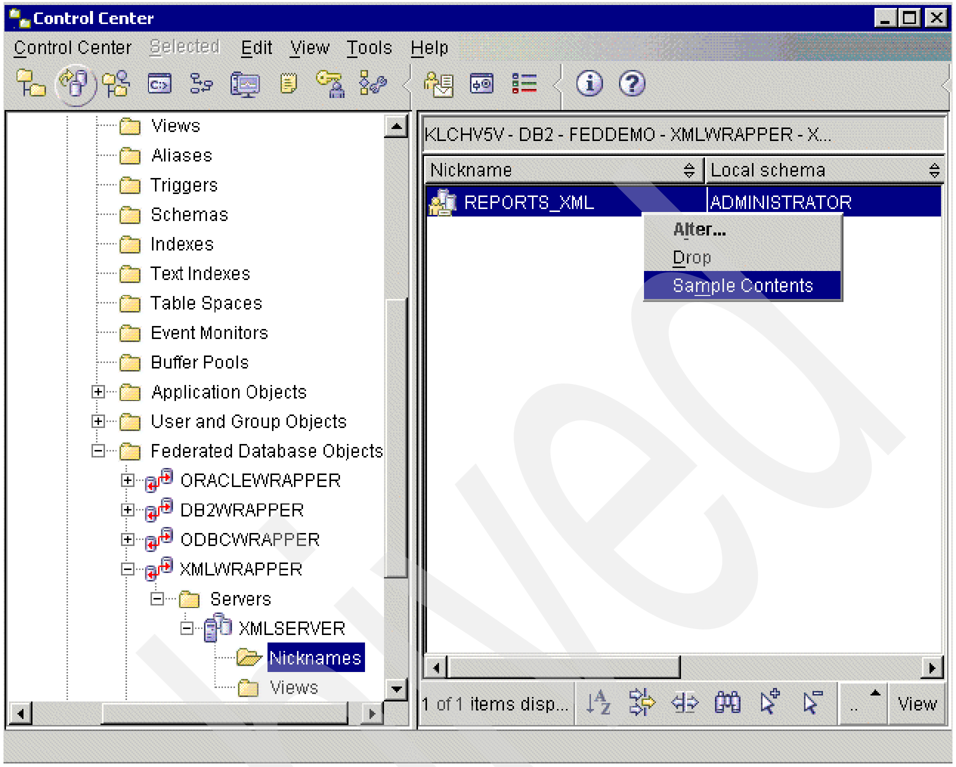


Figure B-21 XML - Create nickname - sample contents

This action should display the contents of the database object referenced by the nickname if nickname creation was successful; we have not displayed the contents here.

B.4 Table-structured files data source

This section describes all the steps involved in configuring a table-structured files data, and creating a nickname on it.

The basic steps for configuring a data source are as shown in Figure 3-7 on page 49, and the following steps describe the configuration of a table-structured files data source:

1. Create the table-structured files wrapper.
2. Create the table-structured files server definition.
3. Create the table-structured files nickname.
4. Test the table-structured files nickname.

We used the DB2 Control Center on the Windows platform to configure the table-structured files data source.

B.4.1 Create the table-structured file wrapper

The table-structured (or flat file) wrapper maps table-structured files to a relational table without having to unload/load data.

A table-structured file has a regular structure consisting of a series of records, where each record contains the same number of fields, separated by an arbitrary delimiter. Null values are represented by two delimiters next to each other.

Table-structured files may be sorted or unsorted, but we recommend the use of sorted files.

The table-structured files wrapper can be used in the following cases:

- The user wants to join table-structured files data with other nicknames or other relational data.
- The user wants to keep the original table-structured file intact (to avoid replicating data or data that may change often, or is composed on the fly).

DB2 Information Integrator can process SQL statements that query data in a table-structured file as if it were contained in a relational table or view. This enables data in a table-structured file to be joined with relational data, nicknames or data in other table-structured files.

The wrapper supports the following data types: CHARACTER, VARCHAR, INTEGER, SMALLINT, DOUBLE, REAL, and DECIMAL:

- CHARACTER and VARCHAR can be declared as the data type for any column of a nickname for a flat file.
- INTEGER, SMALLINT, DOUBLE, REAL, and DECIMAL can be declared as the data type for a column of nickname for a flat file, if the value in all occurrences of that field in every record of the file meet the criteria for the values of this data type.

> **Note:** The table-structured files wrapper does not support the INSERT, UPDATE, DELETE functions.

Ensure that the following prerequisites are in place before deciding to use the table-structured files wrapper:

- Data files must be accessible to the DB2 Information Integrator server. They must reside on the same machine or on a network accessible shares/mounts.

- The user ID that runs DB2 (in our case, ADMINISTRATOR) must be able to open and read the file for which the nickname is being created. Refer to "Key technical considerations in a Windows environment" on page 72 for a brief discussion on accessing files on network drives.
- The data schema for the data file (number of column and columns data types) must be consistent throughout the file.
- The data can only be sorted on one column in ascending order.

Navigate to the FEDDEMO federated database that was created during the DB2 II install as described in A.4.3, "Create the federated database" on page 222, and right-click the **Federated Database Objects** and click **Create Wrapper** as shown in Figure B-1.

This action displays Figure B-22.

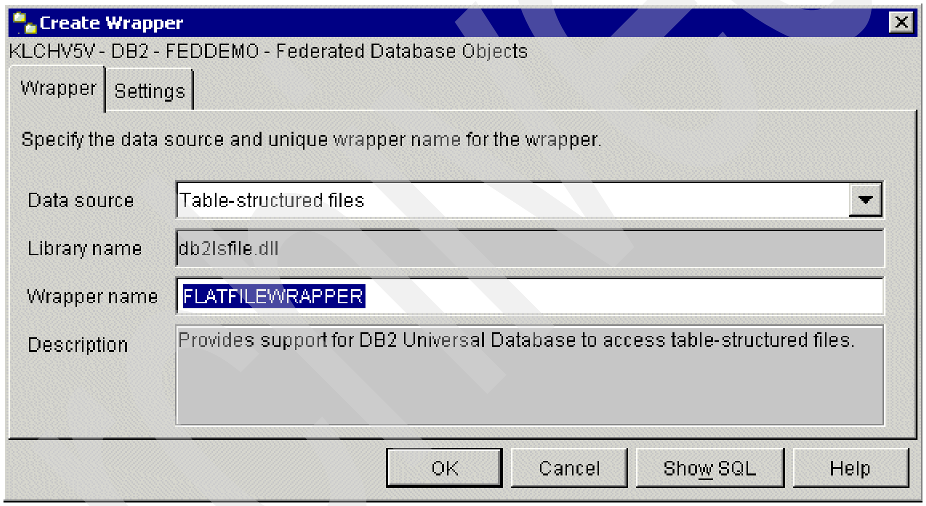


Figure B-22 Table-structured file - Create Wrapper

Select **Table-structured files** in the Data source field, and enter `FLATFILEWRAPPER` for the Wrapper name.

The **Settings** tab displays the same screen shown in Figure B-15 on page 246. Click **OK** to create the FLATFILEWRAPPER wrapper.

Example B-10 shows the command line version of creating the FLATFILEWRAPPER wrapper.

Example: B-10 Table-structured file - Create wrapper SQL statement

```
CONNECT TO FEDDEMO;
CREATE WRAPPER "FLATFILEWRAPPER" LIBRARY 'db2lsfile.dll';
CONNECT RESET;
```

B.4.2 Create the table-structured file server definition

For table-structured files, the DB2 Information Integrator server definition is required even though it does not require information similar to the version and connection information needed in the server definition for relational data sources.

Select and expand the **FLATFILEWRAPPER**, right-click the **Servers** folder and click **Create** similar to the actions taken for the ORACLEWRAPPER in Figure B-3 on page 230.

This action will display Figure B-23.

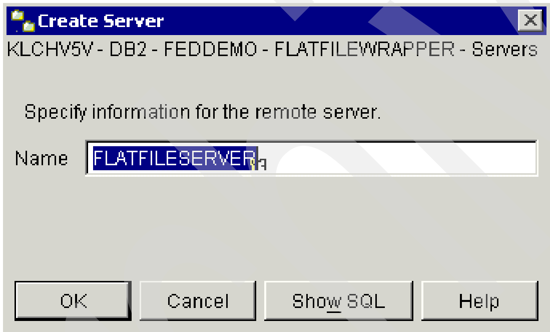


Figure B-23 Table-structured file - Create Server

The server definition requires the name of the server definition to be unique within the federated database. In our case we entered `FLATFILESERVER` as shown in Figure B-23.

Click **OK** to complete the server definition.

For the table-structured file server, the server options cannot be updated, added, or dropped.

> **Note:** There can be multiple server definitions for each table-structured file wrapper.

Example B-11 shows the command line version of creating the FLATFILESERVER server definition.

Example: B-11 Table-structured file - Create server statement

```
CONNECT TO FEDDEMO;
CREATE SERVER FLATFILESERVER WRAPPER "FLATFILEWRAPPER";
CONNECT RESET;
```

Note: For a table-structured file server, we cannot update, add, or drop server options.

B.4.3 Create the table-structured file nickname

After setting up the table-structured file wrapper and the table-structured file server definition, we can create the actual link to a table-structured file, or a directory containing the table-structured file.

Attention: Unlike some of the other data sources, there are no user mappings associated with table-structured file data sources since the federated server user ID must have access to the table-structured file, or directory.

Example B-12 shows a subset of the competitiverates.txt file containing competitive insurance rates used in the CFS portal. This file resides in the c:\flatfiles directory.

Example: B-12 Contents of file competitiverates.txt

```
My Insurance,2,27,6 months,$500.00
eInsurance,2,27,6 months,$520.00
New Life,2,27,6 months,$505.00
All Insured,2,27,6 months,$515.00
Bay Life,2,27,6 months,$503.00
```

Example B-12 shows the file containing five fields; we assumed that they were all character fields. The field delimiter is the comma character (`,`) and each record ends with a line feed including the last record.

Select and expand **FLATFILESERVER,** right-click **Nicknames,** and click **Create**, which is similar to the to the actions taken for the ORACLESERVER in Figure B-9 on page 240.

This action opens up a dialog window as displayed in Figure B-24.

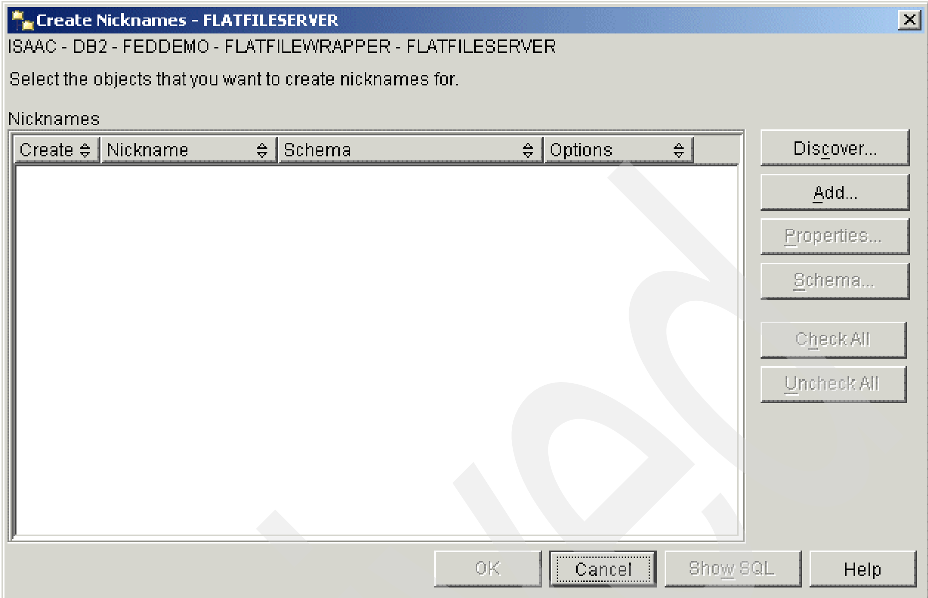


Figure B-24 Table-structured file - Create Nickname

Figure B-24 is similar to the Figure B-10 on page 241, and provides two options for adding a nickname:

- Manually add a nickname (click **Add**) by specifying local schema.
- Use the discover functionality (click **Discover)** to be prompted for providing all the required information as follows:

  Click **Discover** to open up a dialog window shown in Figure B-25.

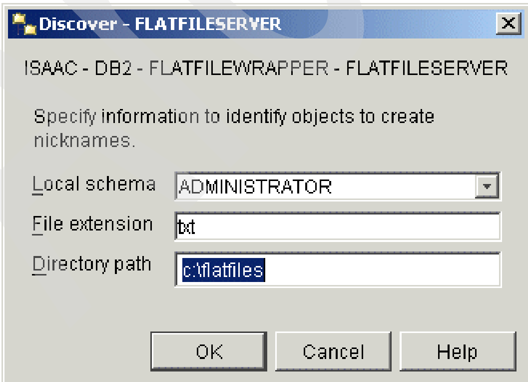


Figure B-25 Table-structured file - Create nicknames Discover window

Set the following values in the Discover Nickname window of Figure B-25:

– ADMINISTRATOR for the Local schema option
– `txt` for the File extension option
– c:\flatfiles for the Directory path option

Click **OK** to open a list of nicknames corresponding to all files in that directory with file extension .txts as shown in Figure B-26.

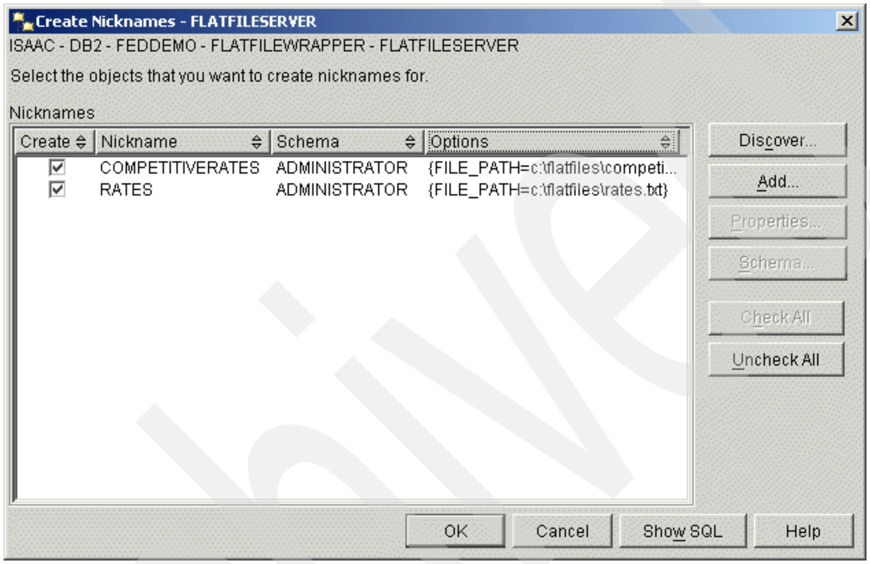


Figure B-26 Table structured file - Create nickname - list nicknames

Since we want to create a nickname only for the competitiverates.txt file, we unchecked everything except that one.

Important: For nicknames of table structured files data sources to work, you need to define the column and data types for the fields in the file.

Select the **COMPETITIVERATES** nickname, and click the **Properties...** button to add the columns and data types. Click **Add...** to view Figure B-27.

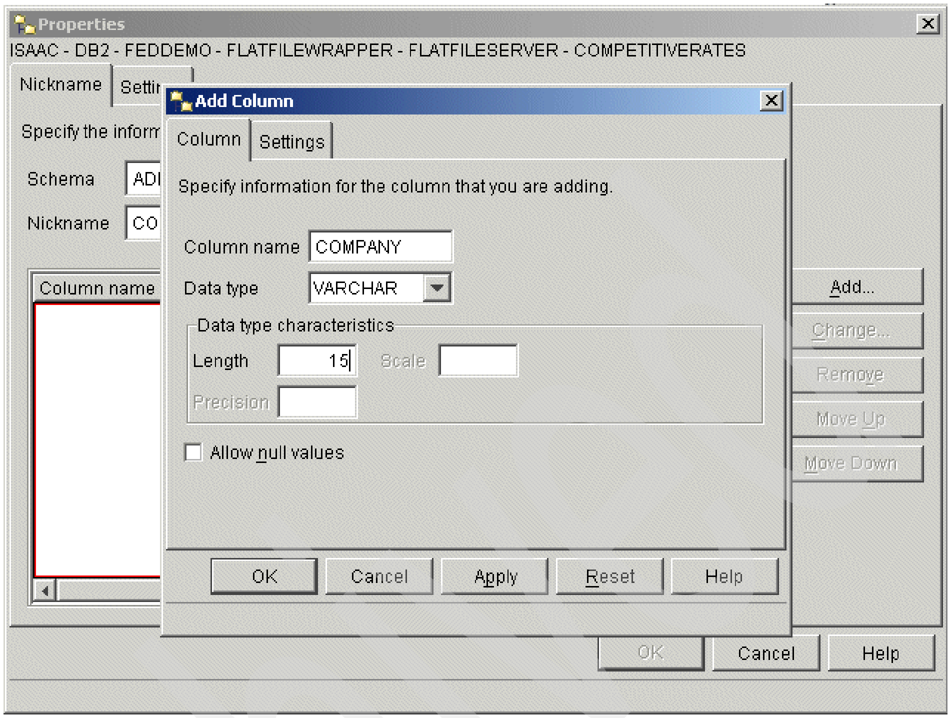


Figure B-27 Table-structured file - Add column

Enter the following information in Figure B-27:

- COMPANY for Column name
- VARCHAR for Data type
- 15 for length
- Deselect Allow null values as shown in Figure B-27

Click **Apply** to add the column definition.

Repeat the process for the rest of the columns: DRIVERS, AGE, POLICY_LEN and PRICE.

After the last column has been entered, click the **Cancel** button. You should have window displaying your columns as shown in Figure B-28.

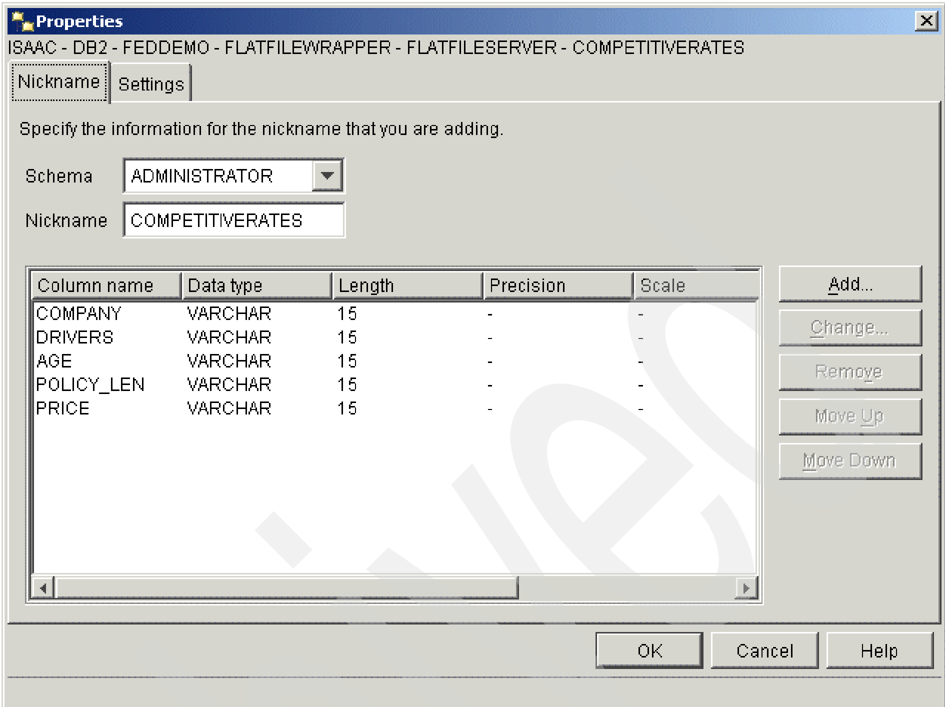


Figure B-28 Table structured file - Create nickname - list all columns

Click the **Settings** page in Figure B-28 to view Figure B-29.

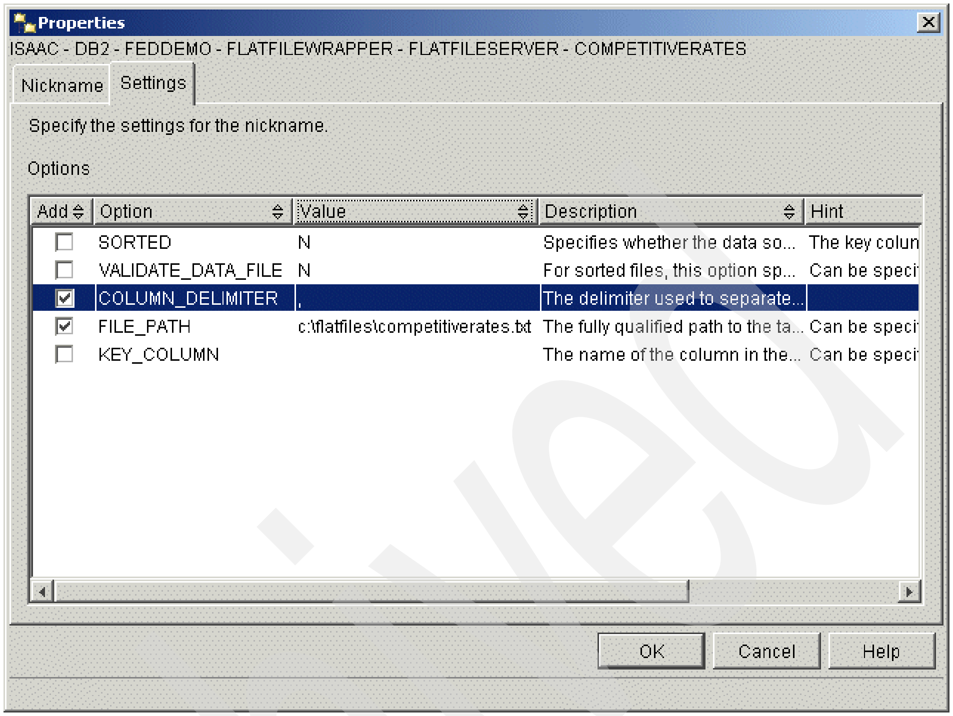


Figure B-29   Table structured file - Create nickname - setting page

Ensure the **Column Delimiter** is checked and set to (**,**) as shown in Figure B-29. Click **OK** to finish modifying the nickname properties, click **OK** to finish creating the nickname.

> **Attention:** Once a nickname is created, its name and schema cannot be modified without dropping and recreating it. However, some other changes can be made as described in B.4.3.1, “Altering the table-structured file nicknames” on page 265.

Example B-13 displays the command line version of the COMPETITIVERATES nickname.

Example: B-13   Table-structured file - Create nickname statement

```
CONNECT TO FEDDB
CREATE NICKNAME ADMINISTRATOR.COMPETITIVERATES (
   COMPANY VARCHAR (15) NOT NULL ,
   DRIVERS VARCHAR (15) NOT NULL ,
   AGE VARCHAR (15) NOT NULL ,
   POLICY_LEN VARCHAR (15) NOT NULL ,
```

```
   PRICE VARCHAR (15) NOT NULL )
   FOR SERVER "FLATFILESERVER"
   OPTIONS(COLUMN_DELIMITER ',' ,
   FILE_PATH 'c:\flatfiles\competitiverates.txt');
CONNECT RESET;
```

B.4.3.1 Altering the table-structured file nicknames

You can use the **ALTER NICKNAME** statement to modify the federated database representation of a data source. This statement can be used to:

- Change the local data type of a column as follows:

```
alter nickname ADMINISTRATOR.COMPETITIVERATES alter column COMPANY local
type varchar (25)
```

- Change local column name as follows:

```
alter nickname ADMINISTRATOR.COMPETITIVERATES alter column COMPANY local
name "INSURANCE_NAME"
```

B.4.4 Test the table-structured file nickname

To test the COMPETITIVERATES nickname, right-click it and click **Sample Contents** as shown in Figure B-30.

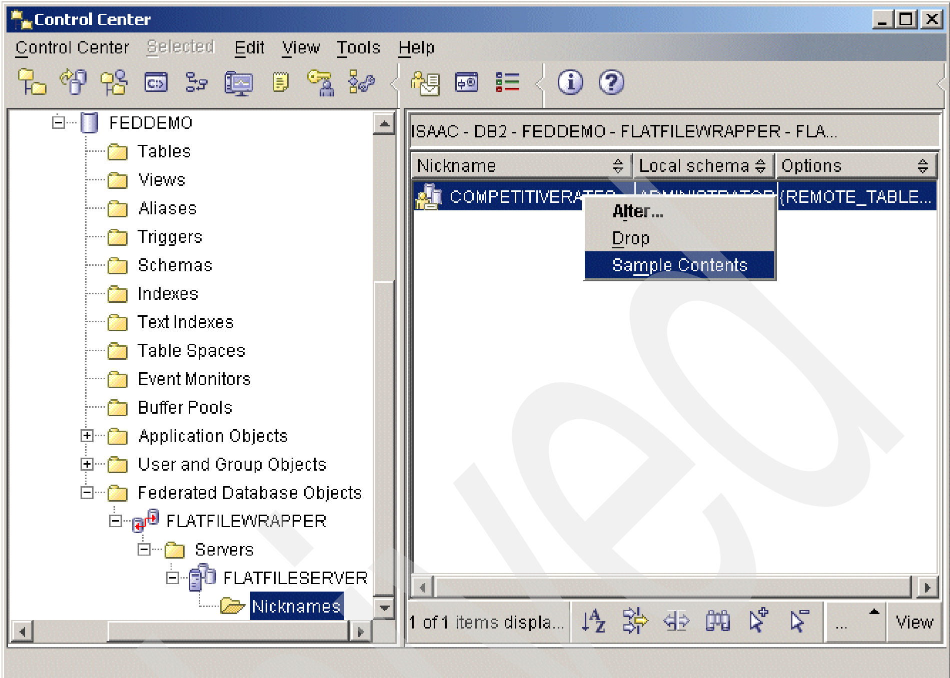


Figure B-30 Table structured file - Create nickname - sample contents

The results of the sample contents request are displayed in Figure B-31, showing that the nickname creation was successful.

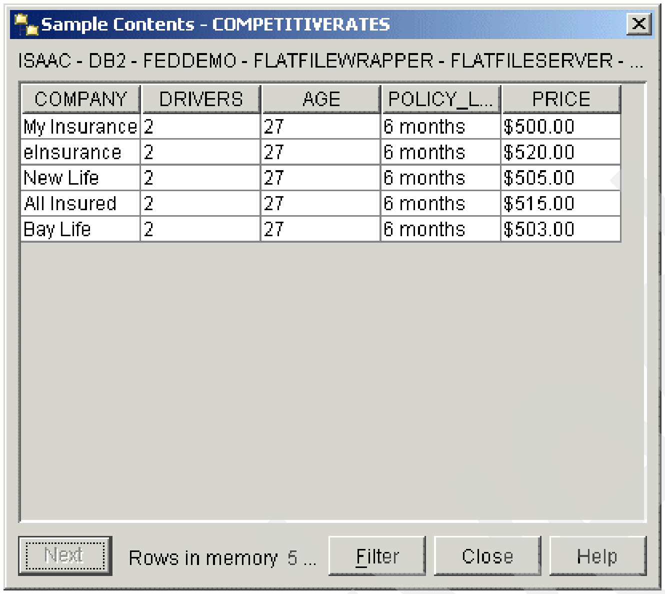


Figure B-31 Table structured file - Create nickname - sample contents results

B.5 Microsoft Excel data source

This section describes the steps involved in configuring a Microsoft Excel file, and creating a nickname on it.

The basic steps for configuring a data source are as shown in Figure 3-7 on page 49.

In the following subsections, we describe the:

- Considerations in choosing between two options for configuring Microsoft Excel data sources
- A Microsoft Excel data source configuration using ODBC. These include the following steps:

  c. Set up Excel ODBC data source names on Windows.
  d. Create the ODBC wrapper.
  e. Create the ODBC server definition.
  f. Create the ODBC nickname.
  g. Test the ODBC nickname.

We used the DB2 Control Center on the Windows platform to configure the XML data source.

B.5.1 Microsoft Excel data source considerations

DB2 Information Integrator provides wrappers that access enterprise data and integrates it with data from other sources. One popular source of data is Microsoft Excel worksheets. Excel worksheets are used in many different types of businesses and contain a wide variety of data.

DB2 Information Integrator provides two wrappers that can be used to access data in Excel worksheets as follows:

- Excel wrapper
- ODBC wrapper

Both these wrappers enable access Excel worksheet data, but each wrapper has different capabilities. The following are some of the considerations in choosing the appropriate wrapper for your installation:

- Software licensing/installation considerations:
  - For the ODBC wrapper, you will not have to install any application software to access the worksheets since the Microsoft Excel driver is shipped with the Windows operating system. Additionally, you have to consider the Microsoft Excel application license requirements.
  - If you use the Excel wrapper, the Microsoft Excel application *must* be installed on the federated server. You should review the Microsoft Excel license terms and conditions for specific information for your environment.
- Data type considerations:
  - When you use the ODBC wrapper, the data types are determined by the Microsoft Excel driver. The Microsoft Excel driver maps the Microsoft Excel data types to ODBC data types. Then DB2 Information Integrator ODBC wrapper maps the ODBC data types (ODBC 3.0 only) to DB2 data types. The DB2 data types for each column are stored in the federated database catalog table. Table B-3 shows the data type mappings.

Table B-3 Excel data type mapping

Excel data type	ODBC data type	DB2 data type
CURRENCY	SQL_NUMERIC	DECIMAL/DOUBLE
DATETIME	SQL_TIMESTAMP	TIMESTAMP
LOGICAL	SQL_BIT	SMALLINT
NUMBER	SQL_DOUBLE	DOUBLE
TEXT	SQL_VARCHAR	VARCHAR

Because the ODBC wrapper supports many different data types, you can manually alter the data type mappings to other DB2 data types through the **ALTER NICKNAME** statement to change the local data types. The list of data types that are available through the ODBC wrapper is comprehensive, and includes data types such as LOBs and other double-byte data types. However, there is the potential of running into a data type mismatch when you alter the local data type to something other than the original mapping. Additionally, issuing these **ALTER NICKNAME** statements can be time consuming when you have a large number of columns or nicknames. If you alter the local type of a nickname column, be sure that the new type will work for all the values for the corresponding column of the Excel spreadsheet.

– The Excel wrapper supports four DB2 data types: DATE, DOUBLE, INTEGER, and VARCHAR. When you issue the **CREATE NICKNAME** statement, you specify one of these data types for each of the columns in the worksheet that you are accessing.

- **Update considerations**
  - Using the ODBC wrapper, you can perform insert and update operations on the worksheet. The Microsoft Excel driver does not support delete operations. To delete data from the worksheet, you must open the worksheet directly to make the changes.
  - The Excel wrapper is a read-only wrapper. To insert, update, or delete data from the worksheet, you must open the worksheet directly using Excel to make the changes.
- Access considerations
  - ODBC wrapper
    - The ODBC wrapper cannot access a worksheet when the workbook is opened by a user or an Excel application in exclusive (read and write) mode. However, if the ODBC wrapper opens the workbook before applications and users open the workbook, the Excel applications and users can open the workbook in read-only mode.
    - With the ODBC wrapper, you can access data from any of the worksheets within a workbook. The Microsoft Excel driver interprets the workbook as a database, and each worksheet within the workbook as a table.
    - When you use the ODBC wrapper to access Excel data, you are limited by what the Microsoft Excel driver supports. The Microsoft Excel driver is strict about the worksheet format. The driver assumes that the first non-blank row will always contain the column labels. If the first non-blank row contains data instead, the data in the first row is treated

    as the column labels. This results in losing the data in the first row whenever you access the data.

    You can overcome this limitation by modifying your worksheet and inserting a row of column labels into the worksheet. If your worksheet has several rows of titles or column labels, you must use a named range to explicitly designate the location of the data within the worksheet. The ODBC wrapper recognizes only one row of labels, the first row in the range. No blank rows are allowed between the labels and the data. The named range must include only one row of column labels.

  - Through the Microsoft Excel driver, the ODBC wrapper allows predicates and aggregate functions to be pushed down to the data source for processing. The driver can also handle joins of worksheets. Pushdown processing can improve performance. All of the data source rows are not returned back to DB2 when the query references a worksheet. Predicates and aggregate functions can be processed by the data source.

- Excel wrapper

  - Since the Excel wrapper supports read-only operations, the wrapper can access the worksheet when other applications and users already have the workbook open in exclusive mode.
  - Because the Excel wrapper interprets the workbook as a table, you can access only the first worksheet in the workbook.
  - The Excel wrapper expects no column labels or titles in the worksheet. However, if the worksheet does include column labels, you can use the `RANGE` option in the `CREATE NICKNAME` statement, and exclude all titles and column labels from the range.
  - The Excel wrapper does not allow any functions or predicates to be pushed down to the data source. The predicates and functions must be processed by the federated server when the rows are returned. Processing predicates and functions at the federated server instead of pushing down the process to the data source can have a negative impact on performance.

  > **Note:** We recommend using the ODBC wrapper for a Microsoft Excel data source to leverage DB2 Information Integrator features and performance optimization capabilities.

If DB2 Information Integrator is installed on AIX where there is no Excel ODBC driver, nor Microsoft Excel, you can still access Microsoft Excel worksheets by installing a multi-tier ODBC provider such as the OpenLink Software solution shown in Figure B-32. For more information on OpenLink Software products, and specifically MultiTier ODBC Client 5.0, go to the following Web site:

http://www.openlink.co.uk

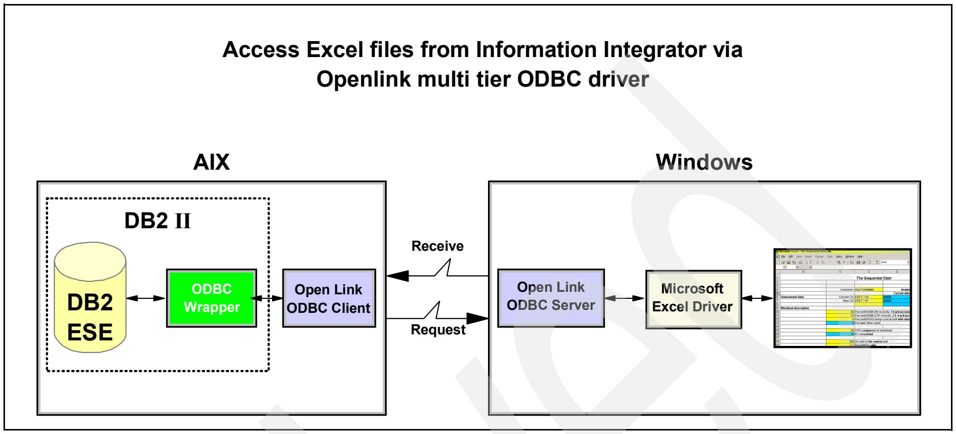


Figure B-32 Microsoft Excel - files through OpenLink

> **Attention:** Another option is for DB2 II on AIX to connect another DB2 II on Windows as a data source that has access to the Microsoft Excel file. In this case, you would create a nickname on the AIX federated server that referenced another nickname on the Windows federated server that referenced a Microsoft Excel file.

B.5.2 Set up Excel ODBC data source name on Windows

Figure B-33 shows the contents of the Microsoft Excel file that will be the target of the nickname creation. Its contents are the same as the competitive rates information used in the table-structured file example of B.4, “Table-structured files data source” on page 255.

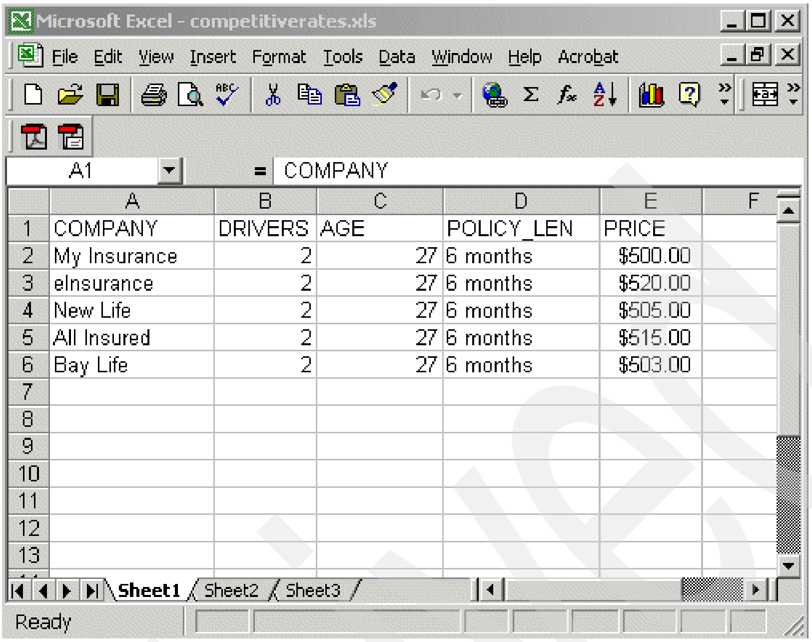


Figure B-33 Excel wrapper - sample data

This Microsoft Excel file competitiverates.xls is stored in the c:\flatfiles directory.

Perform the following steps to set up the Microsoft Excel ODBC Data Source Name on Windows 2000:

1. Select **Start -> Settings -> Control Panel -> Administrative Tools -> Data Sources (ODBC).**

   This opens the ODBC Data Source Administrator screen as shown in Figure B-34.

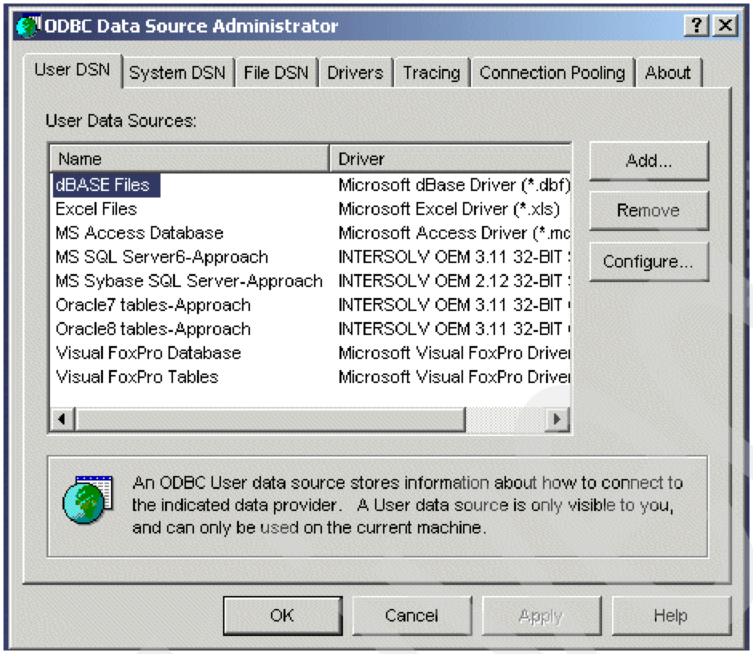


Figure B-34 ODBC Data Source Administrator

2. Click the **System DSN** tab.
3. Click **Add** to add a new DSN.
4. Select the driver **Microsoft Excel Driver(*.xls)** as shown in Figure B-35.

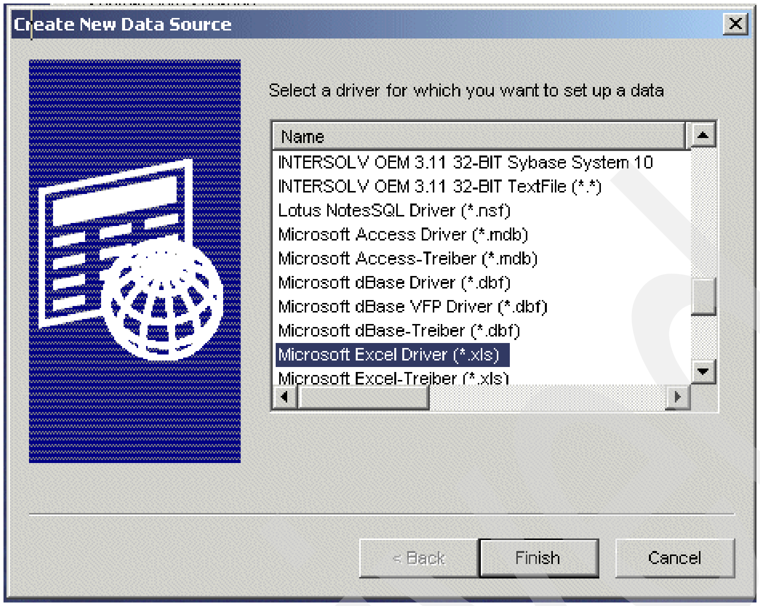


Figure B-35 Excel ODBC create new data source

5. Click the **Finish** button to view the ODBC Microsoft Excel Setup screen shown in Figure B-36.

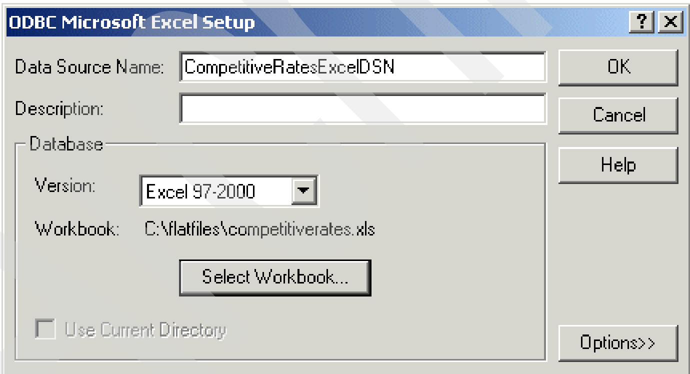


Figure B-36 Excel wrapper - ODBC DNS configuration

6. Fill in the **Data Source Name** field as **CompetitiveRatesExcelDSN.**
7. Check Excel **Version** as **Excel 97-2000.**
8. Click the **Select Workbook** button to select the file c:\flatfiles\competitiverates.xls as shown in Figure B-37.

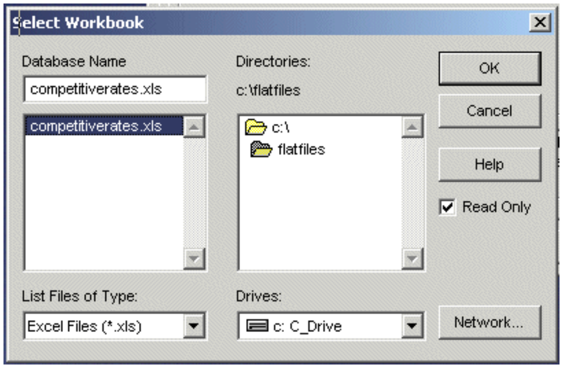


Figure B-37 Select Workbook

9. Click **OK** to take you back to the ODBC Microsoft Excel Setup screen as shown in Figure B-36.
10. Click the **OK** button to create the CompetitiveRatesExcelDSN data source name.

Repeat these steps for each workbook that you plan to access.

B.5.3 Create the ODBC wrapper

Navigate to the FEDDEMO federated database that was created during the DB2 II install as described in A.4.3, “Create the federated database” on page 222, and right-click the **Federated Database Objects** and click **Create Wrapper** as shown in Figure B-1.

This action displays Figure B-38.

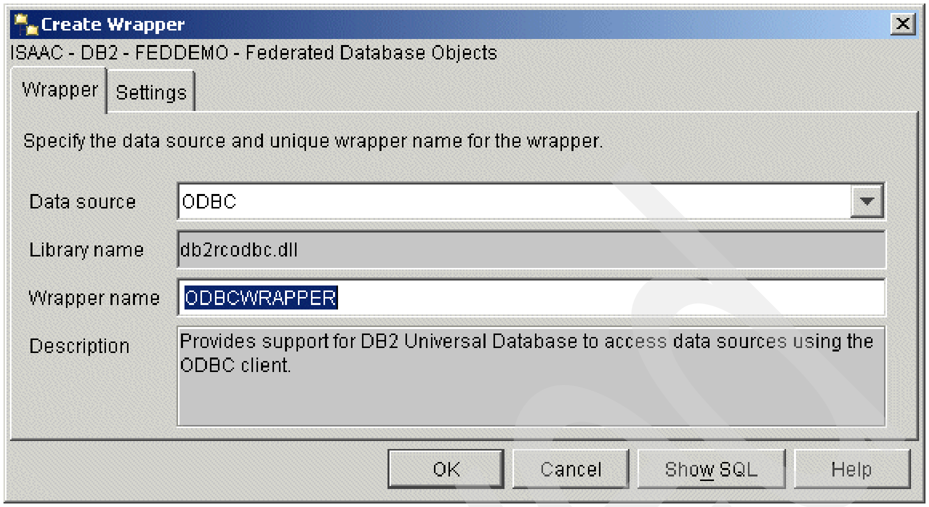


Figure B-38 Excel wrapper - Create wrapper

Select **ODBC** in the **Data source** field, and ODBCWRAPPER in the Wrapper name field as shown in Figure B-38.

Click the **Settings** tab in Figure B-39.

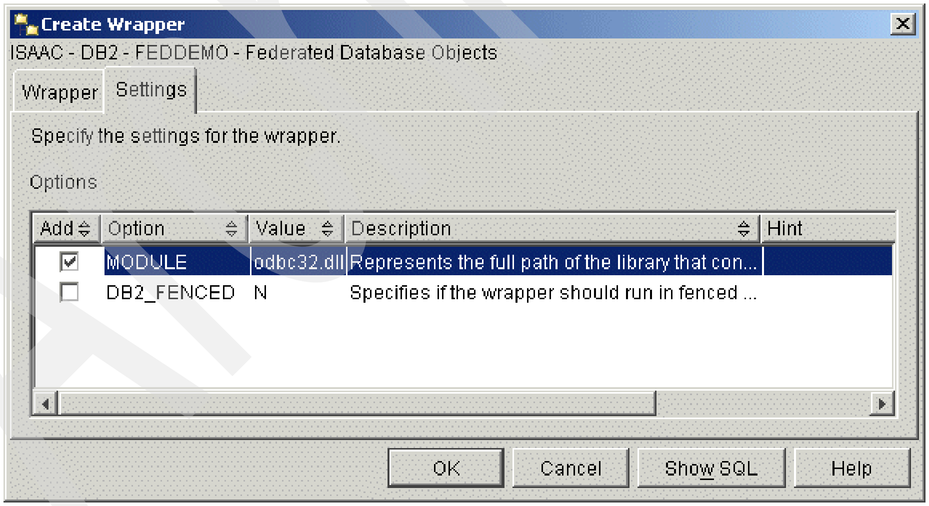


Figure B-39 Excel wrapper - Create wrapper settings

Ensure that the DB2_FENCED option is set to `N` (the default value), and set the MODULE option with the full library path for the ODBC driver as shown in Figure B-39.

Click **OK** to create the ODBCWRAPPER wrapper.

Example B-14 shows the command line version of creating the ODBCWRAPPER wrapper.

Example: B-14 Excel wrapper - Create wrapper statement

```
CONNECT TO FEDDEMO
CREATE WRAPPER "ODBCWRAPPER" LIBRARY 'db2rcodbc.dll'
      OPTIONS( ADD MODULE 'odbc32.dll');
CONNECT RESET
```

B.5.4 Create the ODBC server definition

This step needs to be repeated for each workbook that you plan to access.

Select and expand the **ODBCWRAPPER**, right-click the **Servers** folder and click **Create,** similar to the actions taken for the ORACLEWRAPPER in Figure B-3 on page 230.

This action will display Figure B-40.

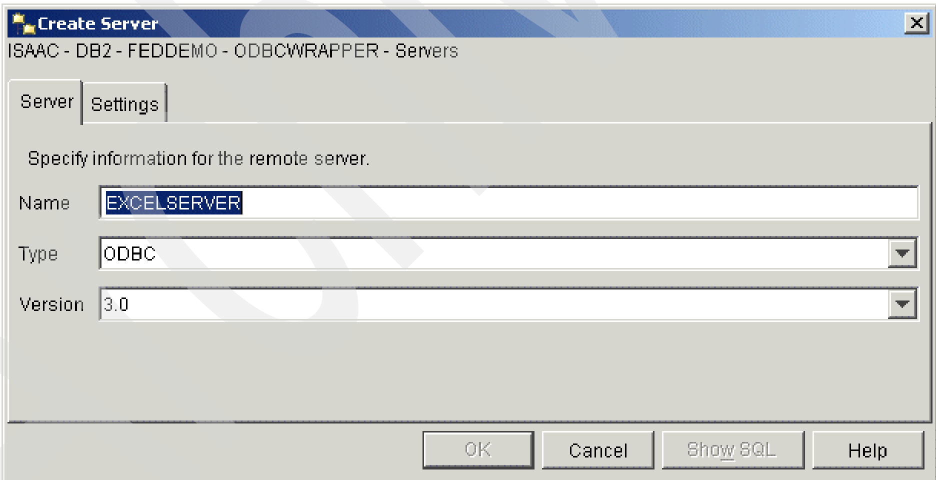


Figure B-40 Excel wrapper - Create server

The server definition requires the following in Figure B-40:

- Name of the server definition to be unique within the federated database. In our case we entered `EXCELSERVER` in the Name field**.**
- Select **ODBC** in the Type field.
- Select **3.0** in the Version field.

Click the **Settings** tab to open the dialog window shown in Figure B-41.

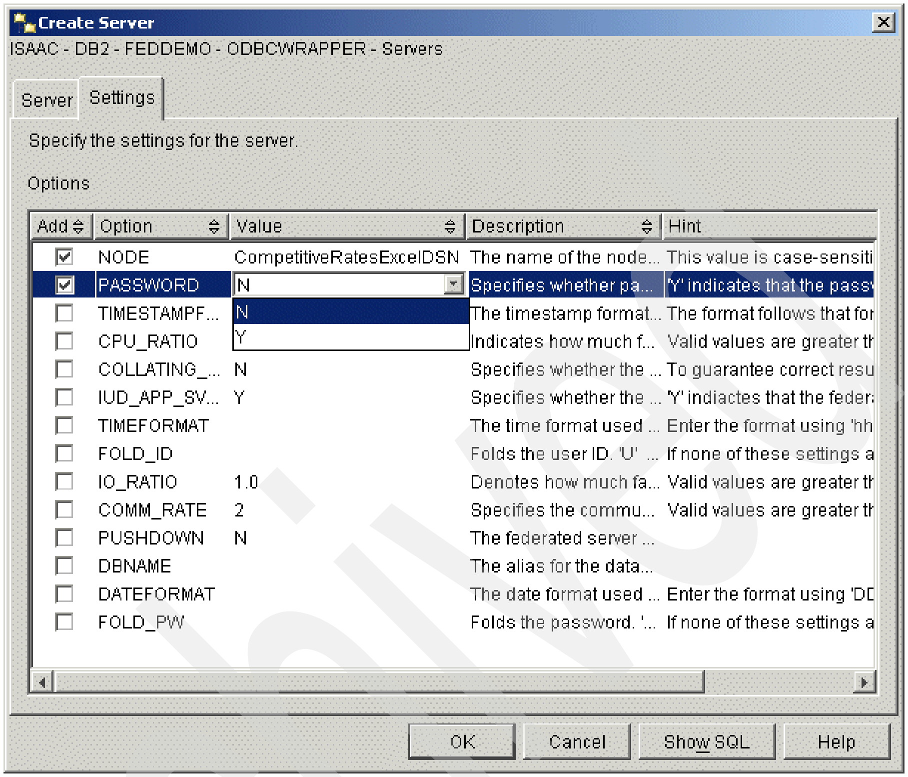


Figure B-41 Excel wrapper - Create server settings

Set the following server options in Figure B-41:

1. For the Node option, type `CompetitiveRatesExcelDSN` in the Value column. This is the name of the ODBC data source name that we configured earlier.
2. For the **Password** option, select **N** in the Value column.

Click **OK** to create the server definition.

Example B-15 shows the command line version of the EXCELSERVER server definition.

Example: B-15 Excel wrapper - Create server statement

```
CONNECT TO FEDDEMO;
CREATE SERVER EXCELSERVER TYPE ODBC VERSION '3.0' WRAPPER "ODBCWRAPPER"
        OPTIONS( ADD NODE 'CompetitiveRatesExcelDSN', PASSWORD 'N');
CONNECT RESET;
```

The following options may be set for increased performance using the **CREATE SERVER** statement, or by using the **ALTER SERVER** statement:

- PUSHDOWN 'Y'
- DB2_BASIC_PRED 'Y'
- DB2_ORDER_BY 'Y'
- DB2_GROUP_BY 'Y'
- DB2_COLFUNC 'Y'
- DB2_SELECT_DISTINCT 'Y'

B.5.4.1 Altering the ODBC server definition

You may modify a server definition when you:

- Want to modify server options

  Server options are set to values that persist over successive connections to the data source. The values are stored in the federated system catalog. The following example activates (add), sets and deactivates (drop) a server option:

  ```
  alter server EXCELSERVER options (add PUSHDOWN 'Y')
  alter server EXCELSERVER options (set NODE 'NEWXLS')
  alter server EXCELSERVER options (drop DB2_ORDER_BY)
  ```

To temporarily set a server option for the duration of a single connection to the federated database, use the **SET SERVER OPTION** statement as follows:

```
set server option DB2_GROUP_BY TO 'N' for server EXCELSERVER
```

The settings are not stored in the federated system catalog.

> **Attention:** Not all the ODBC server options are available through the DB2 Control Center. They must be set through the command line.

Additional server options for ODBC are listed in Table B-4.

Table B-4 ODBC additional server options

Server options	Description
DB2_GROUP_BY	GROUP BY is supported
DB2_ORDER_BY	ORDER BY is supported
DB2_BASIC_PRED	It allows ‘=’, ‘<‘, ‘>’ predicates
DB2_COLFUNC	It allows column functions
DB2_SELECT_DISTINCT	SELECT DISTINCT is supported

Important: These server options' settings *override the default settings* for attributes within the DB2 Information Integrator ODBC wrapper. The wrapper needs to work with *any* ODBC data sources (even those with little SQL functionality) to avoid the occurrences of errors that arise from pushing down SQL operations and functions not supported by an ODBC data source.

Multiple ODBC server definitions can be defined for a single wrapper with each server definition having different options.

The settings for these attributes in the ODBC wrapper are conservative.

Since Excel supports some basic SQL functionality, we can set these server options to override the default settings in the ODBC wrapper. DB2 Information Integrator can then be able to pushdown simple WHERE clauses, ORDER BY, and GROUP BY clauses to Excel, which should improve overall performance.

B.5.5 Create the ODBC nickname

After setting up the ODBC wrapper and the ODBC server definition, we can create the actual link to the Excel file.

Attention: Unlike some of the other data sources, there are no user mappings associated with Microsoft Excel data sources since the federated server user ID must have access to the Excel file.

Select and expand **XMLSERVER,** right-click **Nicknames,** and click **Create** similar to the actions taken for the **ORACLESERVER** in Figure B-9 on page 240.

This action opens up a dialog window as displayed in Figure B-42.

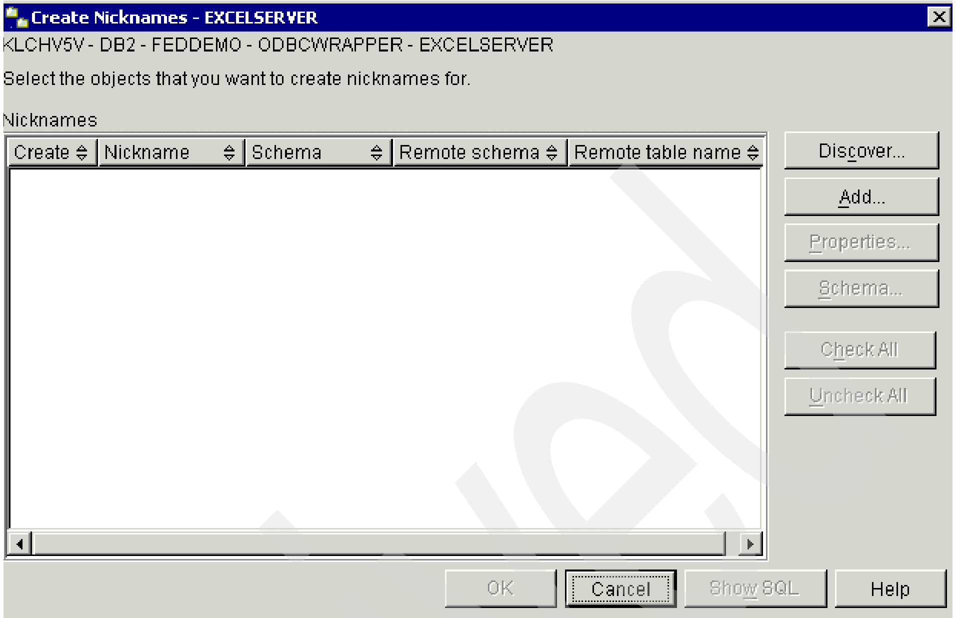


Figure B-42 Excel wrapper - Create nickname - main window

Figure B-42 is similar to the Figure B-10 on page 241, and provides two options for adding a nickname:

1. Manually add a nickname (click **Add**) by specifying local and remote schema and table identification.
2. Use the discover functionality (click **Discover)** to be prompted for providing all the required information as follows:

   Click **Discover** to open up a dialog window shown in Figure B-43.

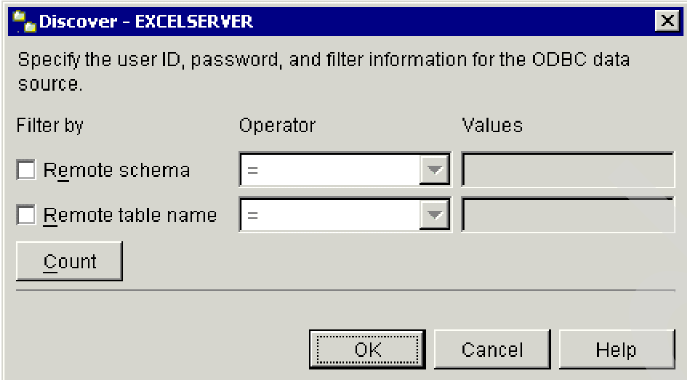


Figure B-43 Excel wrapper - Create Nickname - Discover with filter

Choose a **filter** method, an **operator**, and the comparing **value,** and click **Count.** Upon successful access to the Microsoft Excel data source, the number of objects that meet the filter criteria will be displayed. Adjust your filter criteria if you find the number is too few or many. When you are satisfied with the **Count** result, click **OK** to list the actual objects as a list in the dialog, as shown in Figure B-44.

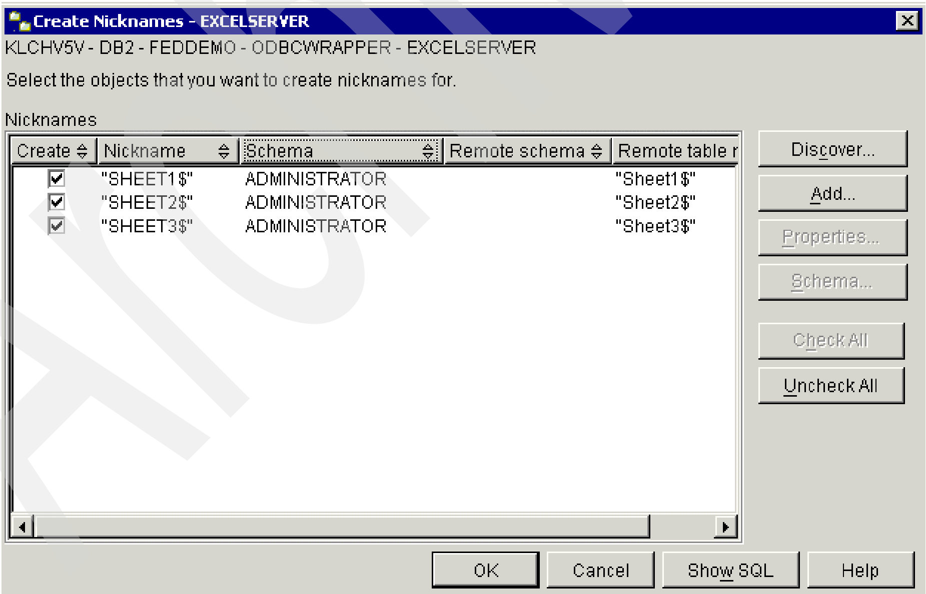


Figure B-44 Excel wrapper - Create nickname -Discover list

Figure B-44 lists the number of sheets within the workbook, and each of these sheets can be configured as independent nicknames.

Note: When you use the Discover filter, the default schema for the entries added to the Create Nickname window is the user ID that is creating the nicknames. In our case, it is ADMINISTRATOR.

If you want to change the properties of the nickname, click the **Properties...** button and make appropriate changes to the nickname.

Since we only have our data in the first sheet, deselect the SHEET2$ and SHEET3$, and rename the nickname `SHEET1$` to `CompetitiveRatesExcelDSN` by clicking **Properties...** and changing the Nickname column value. This results in a nickname list as shown in Figure B-45.

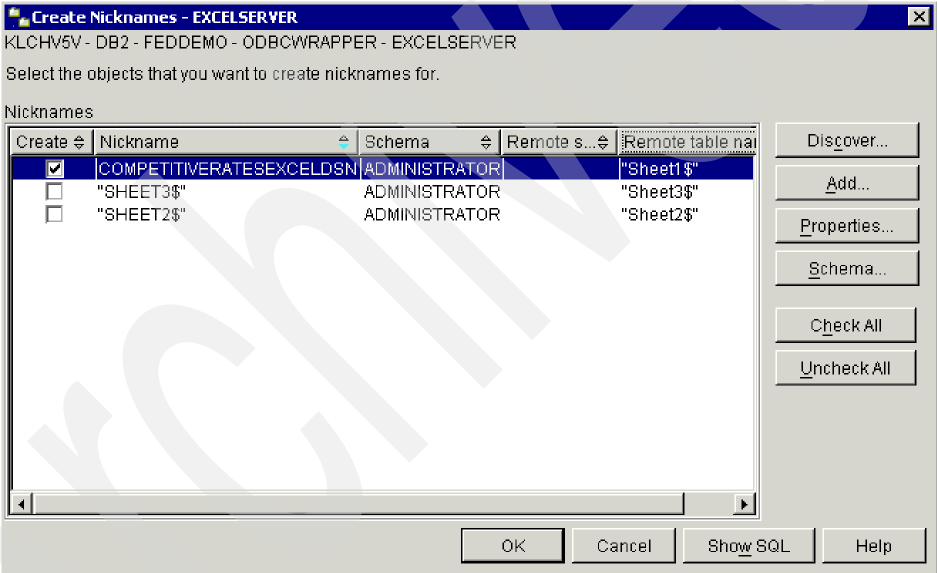


Figure B-45 Excel wrapper - Create nickname - modified list

Click **OK** to create the nickname.

Note: Once a nickname is created, its name and schema cannot be modified without dropping and recreating it. However, some other changes can be made as described in "Altering the ODBC nicknames" on page 284.

If you want to change the properties of the nickname, select the nickname and click **Properties...**

Example B-16 shows the command line version of creating the COMPETITIVERATESEXCELDSN nickname.

Example: B-16 Excel wrapper - Create nickname statements

```
CONNECT TO FEDDEMO;
CREATE NICKNAME ADMINISTRATOR.COMPETITIVERATESEXCELDSN FOR
EXCELSERVER."Sheet1$";
CONNECT RESET;
```

B.5.5.1 Altering the ODBC nicknames

You can use the **ALTER NICKNAME** statement to modify the federated database representation of a data source. This statement can be used to:

- Change local data type of a column as follows:

```
alter nickname ADMINISTRATOR.COMPETITIVERATESEXCELDSN alter column
company local type varchar (25)
```

- Change local column name as follows:

```
alter nickname ADMINISTRATOR.COMPETITIVERATESEXCELDSN alter column
company local name "NEW_COMPANY"
```

B.5.6 Test the ODBC nickname

To test your COMPETITIVERATESEXCELDSN nickname, right-click it and click **Sample Contents** as shown in Figure B-46.

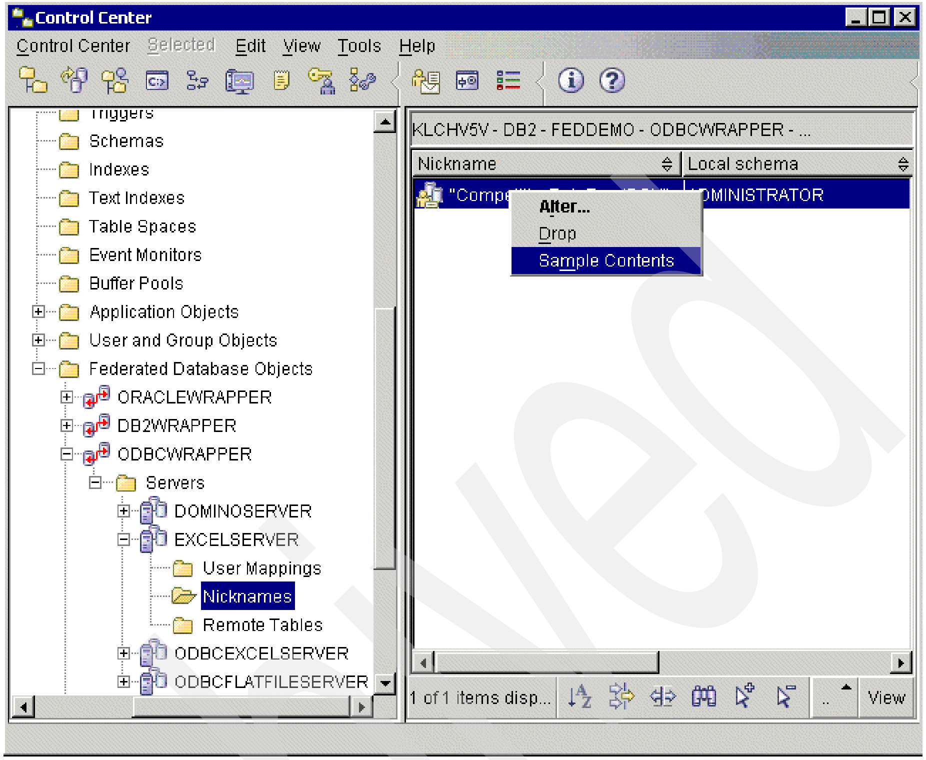


Figure B-46 Excel wrapper - Create nickname - sample contents

The results of the sample contents request is displayed in Figure B-47, showing that the nickname creation was successful.

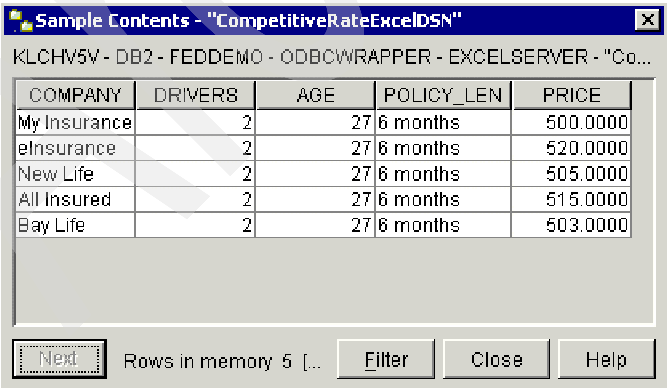


Figure B-47 Excel wrapper - Create nickname - sample contents results

B.6 WebSphere MQ data source

This section describes all the steps involved in configuring a WebSphere MQ data source. The steps involved are shown in Figure B-48.

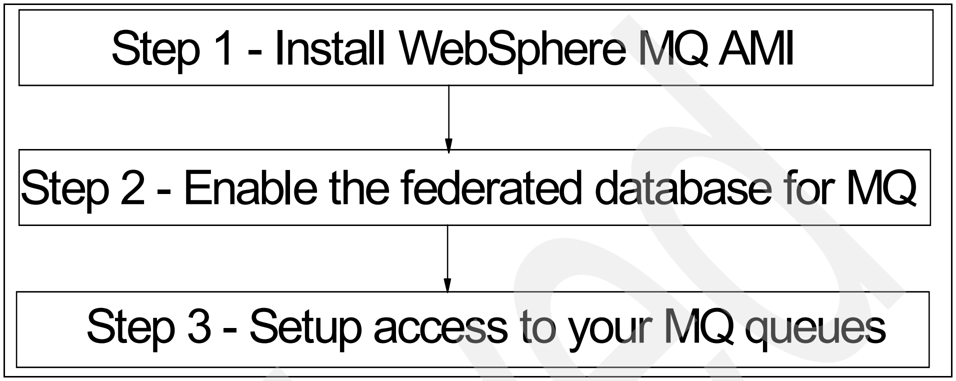


Figure B-48 WebSphere MQ - processes for enabling federated access

It is assumed that WebSphere MQ is installed and configured.

> **Attention:** DB2 Information Integrator integration with WebSphere MQ is done using DB2 MQ UDFs.The MQ UDFs for DB2 are enabled by installing IBM WebSphere Application Messaging Interface (AMI), and then enabling the federated database for MQ UDF functionality.

B.6.1 Install WebSphere MQ AMI

The following actions will install the WebSphere MQ Application Messaging Interface (AMI):

1. Using Windows Explorer, navigate to the DB2 UDB installation directory. In our installation this was c:\Program Files\IBM\SQLLIB as seen in Figure B-49.

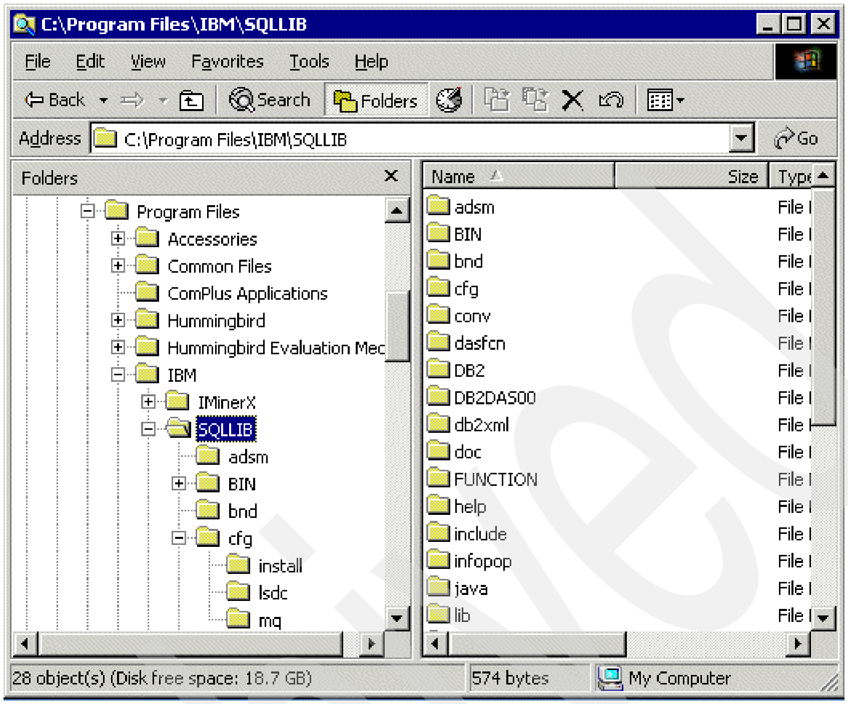


Figure B-49 DB2 UDB install directory

2. Within this folder, navigate to the subfolder cfg\mq as shown in Figure B-50, and open the zip file **ma0f_nt.zip** (WebSphere MQ AMI setup file) with a zip extraction utility.

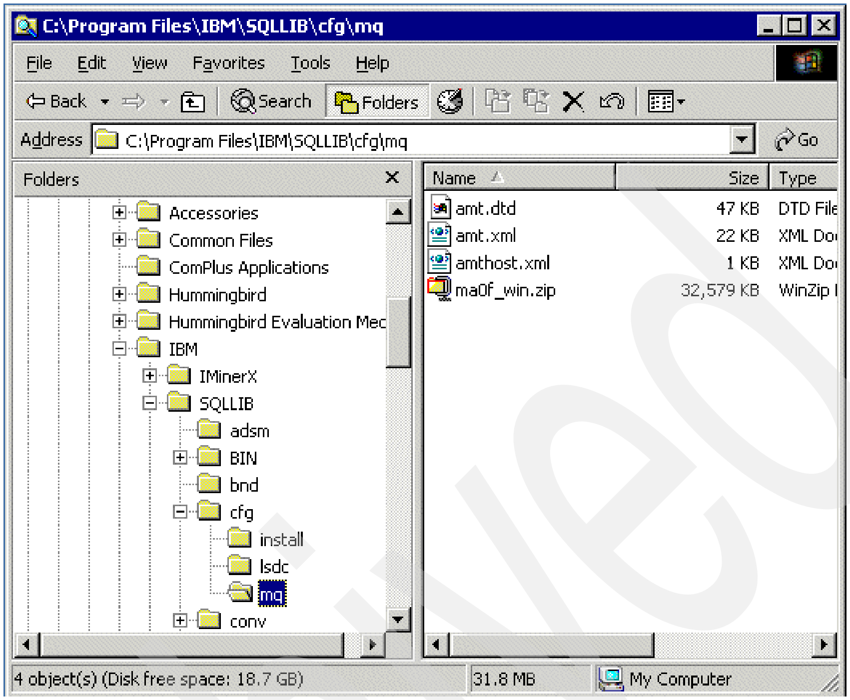


Figure B-50 WebSphere MQ AMI file

3. Extract the zip file to a temporary directory, and launch **`setup.exe`** from that directory. This opens a welcome screen as shown in Figure B-51.

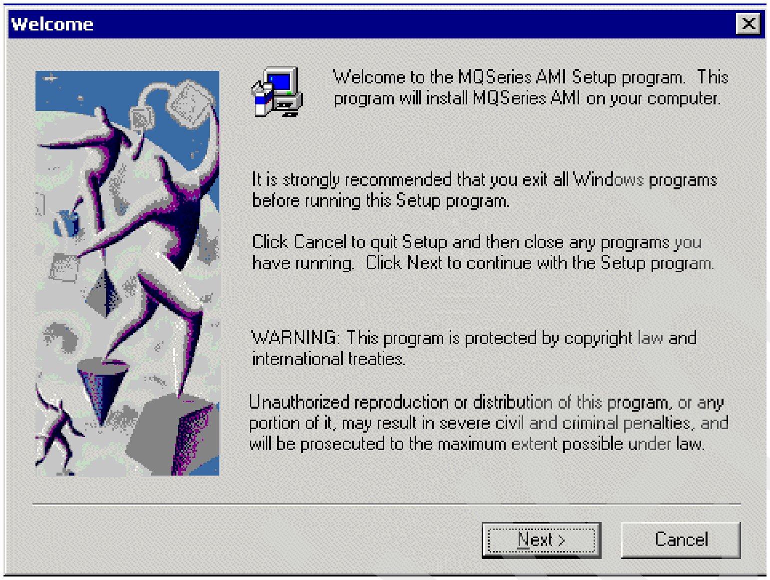


Figure B-51 WebSphere MQ AMI setup welcome screen

4. Click **Next** to open the dialog screen shown in Figure B-52 for you to select the destination location. The default value is the WebSphere MQ installed directory.

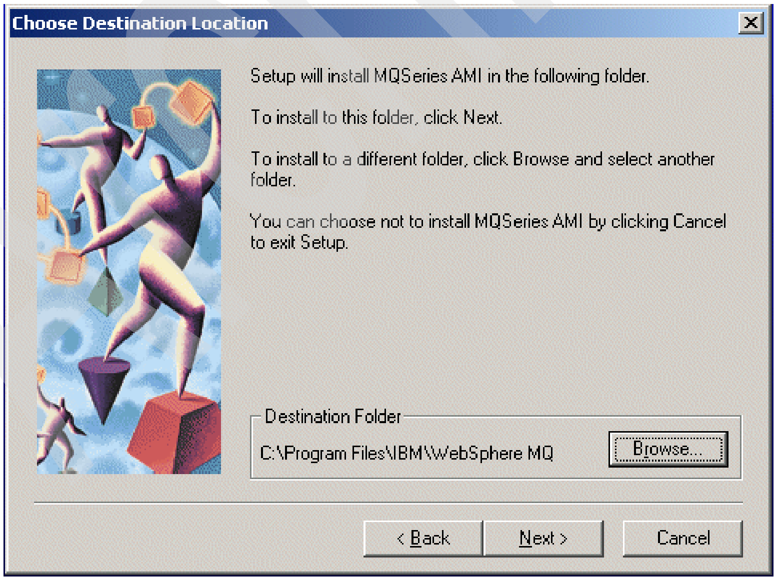


Figure B-52 WebSphere MQ AMI install destination location

5. Select the default installation folder or specify another one (through the **Browse...** button) and click **Next** to open the dialog screen (shown in Figure B-53) for you to select the components you want to install.

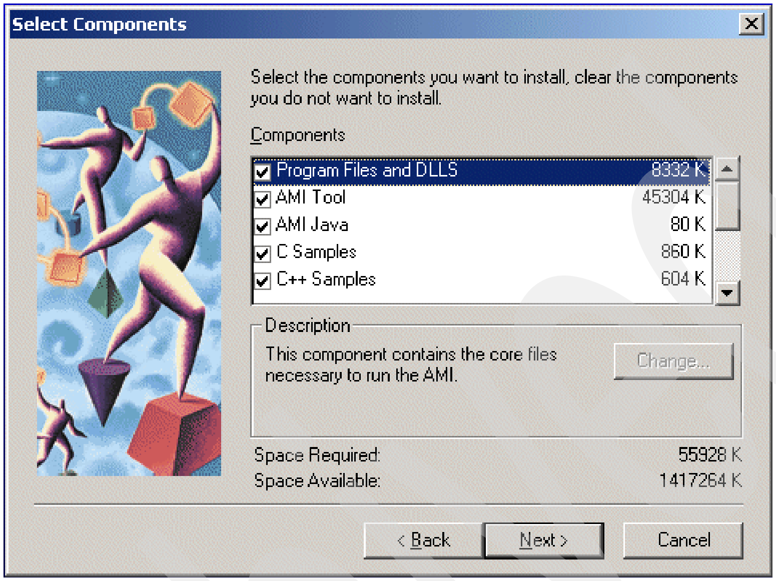


Figure B-53 WebSphere MQ AMI install components

6. Select all components, and click **Next** to open the dialog screen (shown in Figure B-54) for you to select a program folder for the installation.

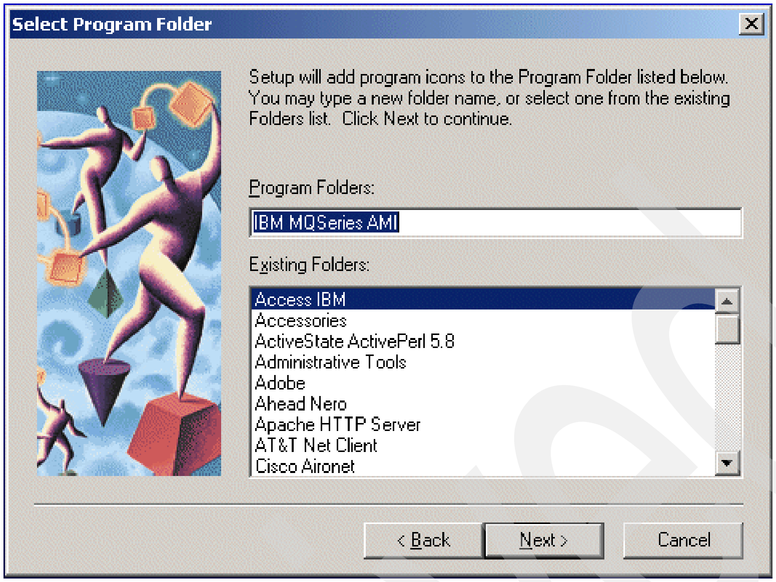


Figure B-54 WebSphere MQ AMI install program folder

7. Leave the default program folder (or modify to suite) and click **Next**.

   This begins the installation of WebSphere MQ AMI. After the installation is done, a pop-up message as shown in Figure B-55 asks you if you want to run a script that defines some WebSphere MQ Objects needed by WebSphere MQ AMI.

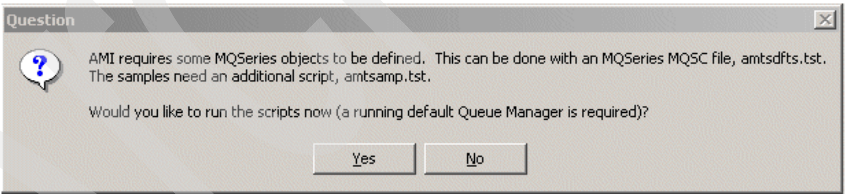


Figure B-55 WebSphere MQ AMI install run scripts

8. Make sure that the WebSphere MQ default Queue Manager is running, and then Click **Yes** to run the script to define WebSphere MQ objects.
9. Click **OK** on the following screen, then close all running programs, and select **Yes, I want to restart my computer.**

> **Attention:** You must restart your machine now.

B.6.2 Enable the federated database for WebSphere MQ UDFs

The following commands must be executed from a DOS command window or AIX terminal to enable the federated database FEDDEMO for MQ functions:

```
enable_MQFunctions -n FEDDEMO -u <userid> -p <password> -force
```

where <userid> is your DB2 userid, and <password> is your DB2 password.

B.6.3 Setup access to your WebSphere MQ queues

When the federated database is enabled for WebSphere MQ, several UDFs were installed that know how to access the WebSphere MQ queue, format the message, and return it back to the application. These UDFs are part of the standard integration between DB2 Information Integrator and WebSphere MQ.

This section describes the creation of a table function over WebSphere MQ queues to hide the use of these UDFs, and gives the appearance of accessing a local DB2 table, rather than a WebSphere MQ queue through the MQ UDFs.

The objective is to create DB2 Information Integrator objects to access the WebSphere MQ queue as follows:

1. Create a table function that provides the “relational” view of the WebSphere MQ queue.

   The table function invokes the WebSphere MQ integration UDFs (schema is db2mq.*) that reads the message from the WebSphere MQ queue, and also formats the output into the correct data types and lengths. The table function will then return the data in a tabular format.

2. Once the table function has been created, create a view that invokes this table function.

   The view insulates the MQ UDFs from the application developers using the view in their applications.

> **Note:** We assume that the following example uses default queues, services, and policies that were created when DB2 was enabled for MQ.

Issue the following commands from the DB2 command line to create the view:

1. Connect to the DB2 II federated database server:

   ```
   CONNECT TO FEDDEMO;
   ```

2. Create the table function:

   ```
   CREATE FUNCTION ADMINISTRATOR.NEW_CLAIMS_READ()
      RETURNS TABLE
   ```

```
    (POLICY_NUMBER INTEGER,
    CLAIM_NUMBER INTEGER,
    INCIDENT_NUMBER INTEGER,
    INCIDENT_DATE VARCHAR(50),
    INCIDENT_OFFICE VARCHAR(50),
    INCIDENT_TYPE VARCHAR(6),
    CORRELID VARCHAR(80)
    )
  LANGUAGE SQL
  NOT DETERMINISTIC
  EXTERNAL ACTION
  READS SQL DATA
  RETURN
    SELECT
      BIGINT(DB2MQ.GETCOL(T.MSG,',',1)),
      BIGINT(DB2MQ.GETCOL(T.MSG,',',2)),
      BIGINT(DB2MQ.GETCOL(T.MSG,',',3)),
      VARCHAR(DB2MQ.GETCOL(T.MSG,',',4),50),
      VARCHAR(DB2MQ.GETCOL(T.MSG,',',5),50),
      VARCHAR(DB2MQ.GETCOL(T.MSG,',',6),6),
      CORRELID
    FROM
      TABLE
      (DB2MQ.MQREADALL
        ('DB2.DEFAULT.SERVICE',
        'DB2.DEFAULT.POLICY'
        )
      )
    AS T;
```

3. Create a view on top of the table function created in the previous step, which then results in the WebSphere MQ queue appearing to be a local relational table:

```
CREATE VIEW ADMINISTRATOR.NEW_CLAIMS_FROM_QUEUE AS
  SELECT *
  FROM TABLE(TUTOR.NEW_CLAIMS_READ()) T
  WHERE CORRELID = 'NEW_CLAIM';
```

4. Insert some test data into the MQ queue using the following commands:

```
VALUES DB2MQ.MQSEND
(
'DB2.DEFAULT.SERVICE',
'DB2.DEFAULT.POLICY',
'1,1,1,"Sept 30 2003","Las Vegas Police Dept", Auto',
'NEW_CLAIM'
);
VALUES DB2MQ.MQSEND
(
'DB2.DEFAULT.SERVICE',
```

```
'DB2.DEFAULT.POLICY',
'2,2,3,"Oct 30 2003","Sunnyvale Police Dept", Home',
'OLD_CLAIM'
);
```

This inserts two messages into the WebSphere MQ queue — one with a correlation id of “NEW_CLAIM“, and the other “OLD_CLAIM“.

5. Issue the following commands to retrieve the messages from the WebSphere MQ queue:

```
Connect to FEDDEMO;
select * from ADMINISTRATOR.NEW_CLAIMS_FROM_QUEUE;
connect reset;
```

> **Attention:** The **`NEW_CLAIMS_FROM_QUEUE`** view only returns rows from the queue where the correlation ID is set to `NEW_CLAIM`, and ignores everything else. Therefore, if you insert messages with a correlation ID set to `OLD_CLAIM`, the select will exclude these messages from its result set. Of the two messages inserted, one is an `OLD_CLAIM` and the other is a `NEW_CLAIM`. The select will therefore only return one row in the result.

B.7 DB2 UDB for z/OS data source

This section describes the steps involved in configuring a DB2 UDB for z/OS data source, and creating a nickname for a database object on it.

The basic steps for configuring a data source are as shown in Figure 3-7 on page 49, and the following steps describe the configuration of an DB2 UDB for z/OS data source:

1. Catalog DB2 UDB for z/OS.
2. Create the DB2 UDB for z/OS wrapper.
3. Create the DB2 UDB for z/OS server definition.
4. Create the DB2 UDB for z/OS user mappings.
5. Create the DB2 UDB for z/OS nickname.
6. Test the DB2 UDB for z/OS nickname.

We used the DB2 Control Center on the Windows platform to configure the DB2 UDB for z/OS data source.

B.7.1 Catalog DB2 UDB for z/OS

Table B-5 lists the information needed to configure a DB2 UDB for z/OS data source.

Table B-5 The DB2 UDB for z/OS system

Parameter	Value
Host name	9.12.6.8
TCP Port	33378
User	<user>
Password	<password>
Location	DB2G
Creator	NAGRAJ1

The following steps catalog a database from the DB2 command line:

1. Log in with user `administrator` or `db2admin` to the Windows system.
2. Catalog the DB2 UDB UDB for z/OS node as follows:

   ```
   db2 catalog tcpip node DB2ZSRV remote 9.12.6.8 server 33378
   ```

3. Store information about the remote host in the Database Connection Services (DCS) directory as follows:

   ```
   db2 catalog dcs database DCSDB2G as DB2G with “Comment on DB2 for z/OS”
   ```

4. Catalog the database as follows:

   ```
   db2 catalog database DCSDB2G at node DB2ZSRV authentication dcs
   ```

5. Test the connection to the database as follows:

   ```
   db2 connect to DCSDB2G user <user> using <password>
   ```

   Note: DCSDB2G is the database name that will be used in the DBNAME server option in our federated server definition for this DB2 UDB for z/OS data source.

B.7.2 Create the DB2 UDB for z/OS wrapper

Note: If there is an existing wrapper for DB2, it can be reused for the DB2 UDB for z/OS subsystem, and a new wrapper is not necessary.

Navigate to the FEDDEMO federated database that was created during the DB2 II install as described in A.4.3, “Create the federated database” on page 222, and right-click the **Federated Database Objects** and click **Create Wrapper** as shown in Figure B-1.

This action displays Figure B-56.

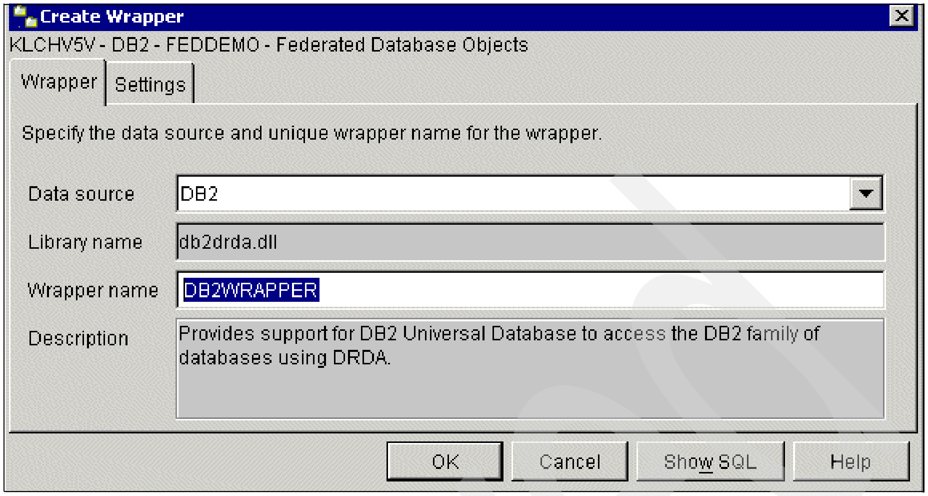


Figure B-56 DB2 UDB for z/OS - Create wrapper

Select **DB2** in the **Data Source** field, and enter the unique name `DB2WRAPPER` as the Wrapper name. The **Settings** tab option is allowed to default. Click **OK** to create the DB2 wrapper.

Example B-17 shows the command line version for creating the DB2WRAPPER wrapper for the DB2 UDB for z/OS instance.

Example: B-17 DB2 UDB for z/OS - Create wrapper statements

```
CONNECT TO FEDDEMO;
CREATE WRAPPER "DB2WRAPPER" LIBRARY 'libdb2drda.a';
CONNECT RESET;
```

B.7.3 Create the DB2 UDB for z/OS server definition

A server definition identifies a data source to the federated database. A server definition consists of a local name and other information about that data source server.

Select and expand the **DB2WRAPPER**, right-click the **Servers** folder, and click **Create** similar to the actions taken for the ORACLEWRAPPER in Figure B-3 on page 230.

This action will display Figure B-57.

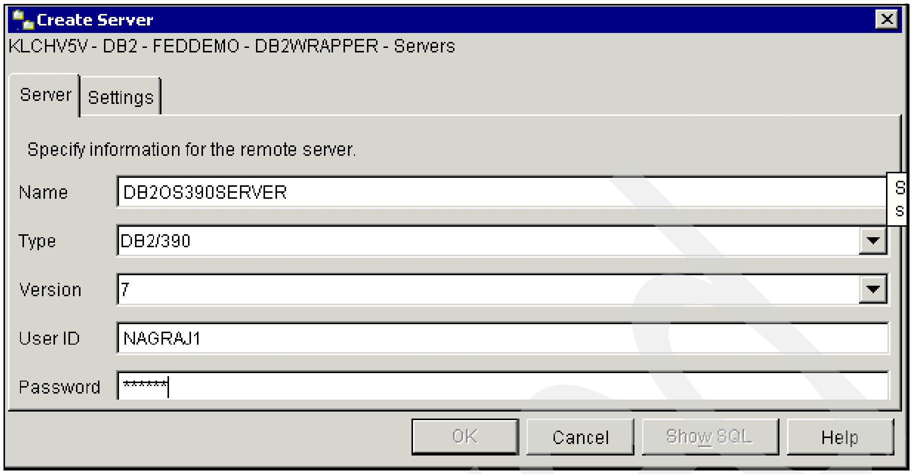


Figure B-57 DB2 UDB for z/OS - Create server

The server definition requires the following:

- **Name**: The name of the server must be unique within the federated database. In our case we chose **DB2OS390SERVER**.
- **Type**: Select **DB2/390.**
- **Version**: Select **6** or **7.**

  In our case we chose **7.**
- Supply the **User ID** (NAGRAJ1) and **Password** values to connect to this server.

Click the **Settings** tab to complete the server definition as shown in Figure B-58.

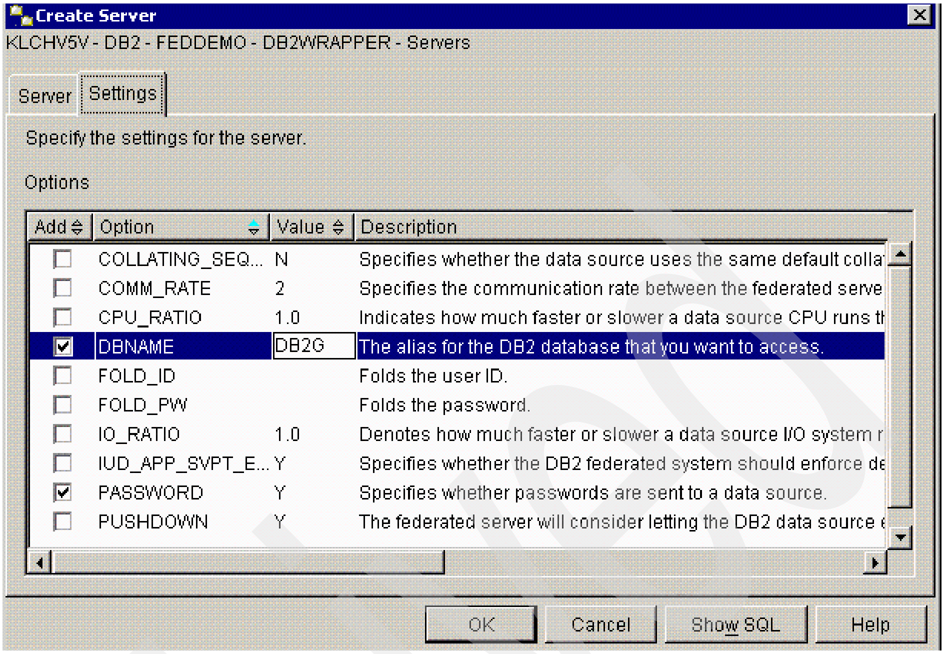


Figure B-58 DB2 UDB or z/OS - Create server settings

In Figure B-58, only the DBNAME and PASSWORD fields are required values — the rest are optional. Server options are used to describe a data source server. The DB2 UDB for z/OS server has a number of options as listed in Table B-6. These options may be set at server creation time, or modified later as described in “Altering DB2 UDB for z/OS server definition & options” on page 299.

Table B-6 DB2 UDB for z/OS server options

Connectivity	DBName	**DB Alias in DB Directory on DB2 Information Integrator server**
	Password	**Default=Y,** Include the password on connections to DB2 UDB for z/OS
	Fold_ID / Fold_PW	Option to fold or not for the DRDA wrapper. **Default=N,** Connect once with user ID/password exactly as specified (recommended) **Y** - connect up to four times with user ID/password with all combinations of lower and uppercase **U** - connect once with user ID/password in uppercase **L** - connect once with user ID/password in lower case

Pushdown	Collating_Sequence	It specifies whether the data source uses the same default collating sequence as the federated database. This affects the pushdown of operations on character columns. **Default=N** **Y**: both use same collating sequence. ORDER BY can be pushed down without compromising integrity of result **N**: both use different collation sequence. ORDER BY cannot be pushed down **I**: case-insensitive. ORDER BY, DISTINCT, WHERE= cannot be pushed down
	Pushdown	**Default: Y**: the SQL operations are pushed down to data sources based on decision of pushdown analysis and optimizer.
Optimization	CPU_RATIO	**Default: 1.0**: specifies the ratio of the DB2 Information Integrator server CPU capacity against the data source CPU capacity
	IO_RATIO	**Default: 1.0**: specifies the ratio of the DB2 Information Integrator server IO rate against the data source IO rate
	COMM_RATE	**Default: 2**: specifies the effective data rate of network to data source in MB per second.
Other	IUD_APP_SVPT_ENFORCE	**Default=Y**: should the DB2 federated system use save-points in multi-update transactions?

Note: There can be multiple server definitions for each wrapper.

Example B-18 shows the command line version for creating the DB2OS390SERVER server definition.

Example: B-18 DB2 UDB for z/OS - Create server statements

```
CONNECT TO FEDDEMO;
CREATE SERVER DB2OS390SERVER TYPE DB2/390 VERSION '7' WRAPPER "DB2WRAPPER"
AUTHID "NAGRAJ1" PASSWORD "*****" OPTIONS( ADD DBNAME 'DB2G', PASSWORD 'Y');
```

B.7.3.1 Altering DB2 UDB for z/OS server definition & options

You may modify a server definition when you:

- Upgrade DB2 UDB for z/OS to a new version:

  ```
  ALTER SERVER DB2OS390SERVER VERSION 7
  ```

- Want to modify the server options

  Server options are set to values that persist over successive connections to the data source. The values are stored in the federated system catalog. The following example activates (add), sets and deactivates (drop) a server option:

```
ALTER SERVER DB2OS390SERVER OPTIONS (ADD IO_RATIO '2.0')
ALTER SERVER DB2OS390SERVER OPTIONS (SET IO_RATIO '3.0')
ALTER SERVER DB2OS390SERVER OPTIONS (DROP IO_RATIO)
```

To temporarily set a server option for the duration of a single connection to the federated database, use the **SET SERVER OPTION** statement as follows:

```
SET SERVER OPTION COMM_RATE TO '3' FOR SERVER DB2OS390SERVER
```

The settings are not stored in the federated system catalog.

> **Tip:** The server option shown in Table B-7 is *not* available through the Db2 Control Center. It must be set through the command line.

Table B-7 DB2 UDB for z/OS additional server options

DB2_MAXIMAL_PUSHDOWN	Default: 'N' The optimizer can choose between cost optimization and the user requirement to perform as much as possible query processing by the remote data source. '**Y**': choose the plan with most query operations to be pushed down to the data sources. '**N**': choose the plan with minimum cost.

B.7.4 Create the DB2 UDB for z/OS user mappings

As mentioned earlier, the user mapping defines an association between a user ID on the federated server and a user ID on the DB2 UDB for z/OS server. This user mapping is used by the federated database server whenever it connects to the DB2 UDB for z/OS server on behalf of the calling federated database user. An association must be created for each user that would be using the federated system.

In our case, we define a single user mapping for our ADMINISTRATOR user since that is the only user ID used in our CFS portal.

Select and expand **DB2OS390SERVER,** right-click the **User Mappings** folder and click **Create,** similar to what is shown for ORACLESERVER in Figure B-6 on page 236.

This action displays the screen shown in Figure B-59.

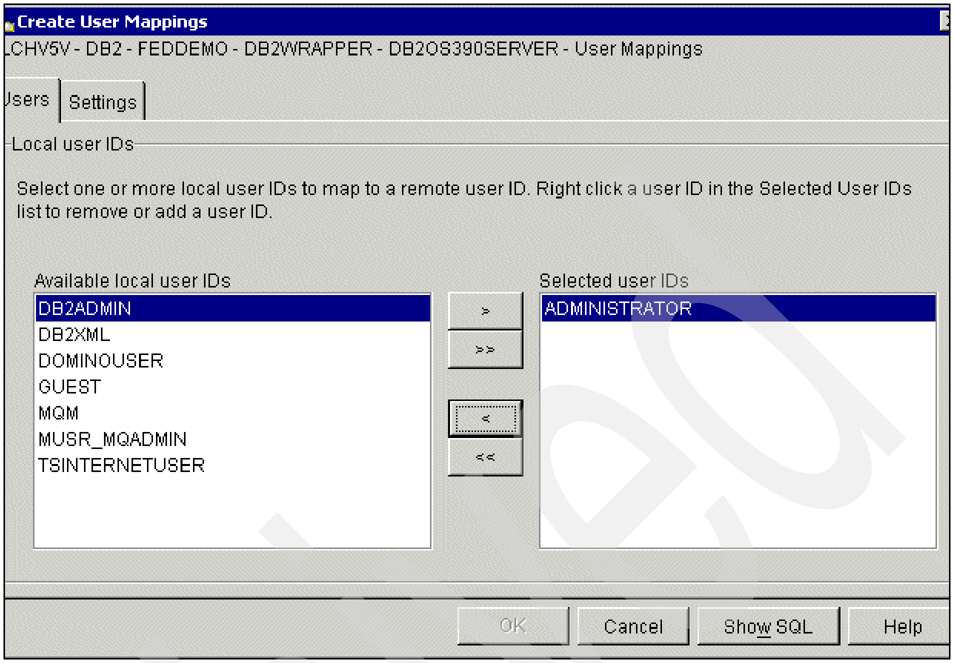


Figure B-59 DB2 UDB for z/OS - Create user mapping dialog

Figure B-59 lists all the user IDs available on the federated system. Select the user that sends the federated requests to the DB2 UDB for z/OS data source. We selected the **ADMINISTRATOR** user, and switched to the **Settings** menu as shown in Figure B-60 to complete the user mapping.

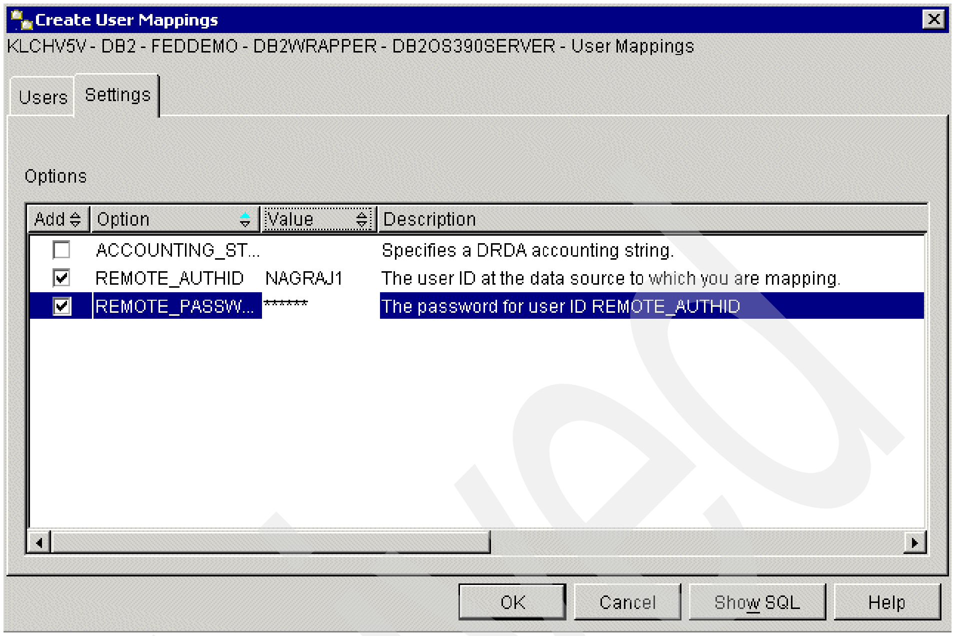


Figure B-60 DB2 UDB for z/OS - Create user mapping settings

In Figure B-60, you need to enter a valid user ID and password to enable the federated system to connect to our DB2 UDB for z/OS data source.

User mappings may be modified after creation time as described in B.7.4.1, “Altering the DB2 UDB for z/OS user mappings” on page 303.

Example B-19 shows the command line version of creating the user mapping for our DB2 UDB for z/OS instance.

Example: B-19 DB2 UDB for z/OS - Create user mapping statements

```
CONNECT TO FEDDEMO;
CREATE USER MAPPING FOR "ADMINISTRATOR" SERVER "DB2OS390SERVER" OPTIONS ( ADD
REMOTE_AUTHID 'NAGRAJ1', ADD  REMOTE_PASSWORD '*****') ;
```

Attention: You might also consider adding the user mapping option ACCOUNTING_STRING. DB2 UDB for z/OS is the only data source that uses it.

B.7.4.1 Altering the DB2 UDB for z/OS user mappings

You can use the **ALTER USER MAPPING** statement to modify a user mapping. It is used to change the authorization ID or password that is used at the DB2 UDB for z/OS data source as follows:

```
ALTER USER MAPPING FOR ADMINISTRATOR SERVER DB2OS390SERVER OPTIONS
   (SET REMOTE_AUTHID 'NAGRAJ1')
ALTER USER MAPPING FOR ADMINISTRATOR SERVER DB2OS390SERVER OPTIONS
   (set REMOTE_PASSWORD 'newpass')
```

Note: The REMOTE_AUTHID and REMOTE_PASSWORD user options are both *case sensitive* unless you set the FOLD_ID and FOLD_PW server options to U or L in the **CREATE SERVER** statement.

User mappings may also be altered through the DB2 Control Center.

B.7.5 Create the DB2 UDB for z/OS nickname

Once the DB2 wrapper, server definition, and user mapping is complete, one needs to test the connection to the data source.

After successfully testing the connection to the data source (not shown here), we can create the actual link to a table located on our remote database as a nickname.

One may choose to define data mappings and function mappings during the creation of a nickname. Data mappings are described in "Data type mappings" on page 52, while function mappings are described in "Function mappings" on page 53.

Note: We did not define any data mappings or function mappings for our nicknames.

When you create a nickname for a DB2 UDB for z/OS table, catalog data from the remote server is retrieved and stored in the federated global catalog.

Select and expand **DB2OS390SERVER,** right-click **Nicknames,** and click **Create** similar to the actions taken for **ORACLESERVER** in Figure B-9 on page 240.

This action opens up the dialog window similar to the one shown in Figure B-10 on page 241, which shows two options for adding a nickname as follows:

1. Manually add a nickname (click **Add**) by specifying local and remote schema and table identification.

2. Use the discover functionality (click **Discover)** to be prompted for providing all the required information as follows:

   Click **Discover** to open up a dialog window to filter out the results as shown in Figure B-61.

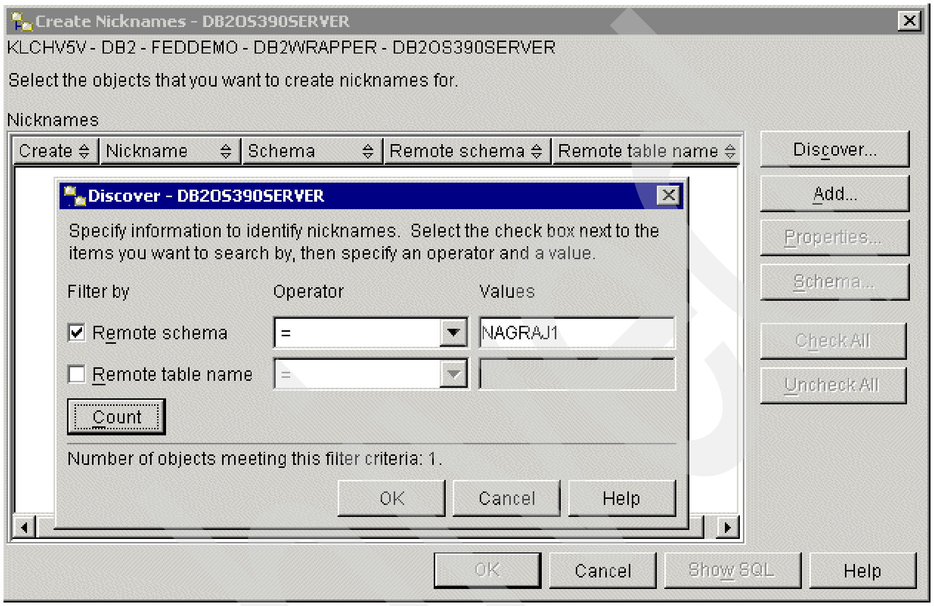


Figure B-61 DB2 UDB for z/OS - Nickname created

   Discover the remote DB2 UDB for z/OS data source by specifying either a **remote schema** or a specific **remote table name** that you want to create a nickname for.

   Choose a **filter** method, an **operator** and the comparing **value**, and click **Count.** Upon successful access to your DB2 UDB for z/OS database, the number of objects that meet the filter criteria will be displayed. Adjust your filter criteria if you find the number is too few or too high. Click **OK** to list the actual objects for remote schema name of NAGRAJ1 as shown in Figure B-62.

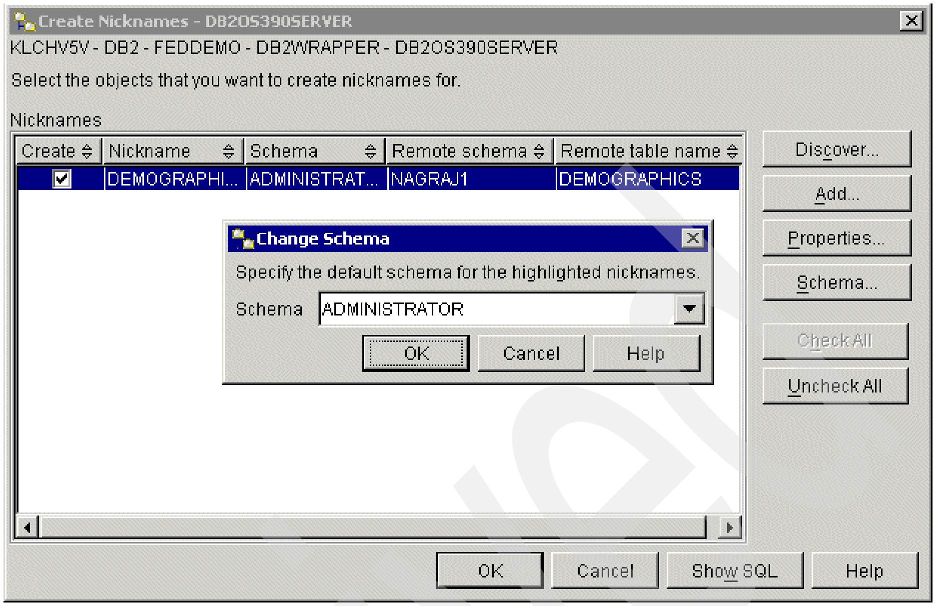


Figure B-62 DB2 UDB for z/OS - Change schema

Note that when the Create Nickname is populated with records from the Discover Filter:

- All the entries are checked so that nicknames will be created for all if the **OK** button is clicked. Click the button **Uncheck All** if you do not want to create all these nicknames at once, and if you want to pick which nicknames you will create from the list.
- The default schema for the nicknames is the user ID that is creating it. In our case that is ADMINISTRATOR. We immediately want to change this so that the schema of our new nicknames will adhere to our naming convention.

Click the **Schema** button to change the local schema for your DB2 UDB for z/OS nicknames.

> **Tip:** We recommend using the same schema name for all DB2 UDB for z/OS nicknames.

Example B-20 shows the command line version of creating the DEMOGRAPHICS nickname for DB2OS390SERVER.

Example: B-20 DB2 UDB for z/OS - Create nickname statements

```
CONNECT TO FEDDEMO;
CREATE NICKNAME ADMINISTRATOR.DEMOGRAPHICS FOR
DB2OS390SERVER.NAGRAJ1.DEMOGRAPHICS;
```

B.7.5.1 Altering the DB2 UDB for z/OS nicknames

The **ALTER NICKNAME** statement can be used to modify the federated database representation of a data source as follows:

- Change the local data type of a column:

  ```
  ALTER NICKNAME "ADMINISTRATOR"."DEMOGRAPHICS" ALTER COLUMN MILAGE LOCAL
  TYPE VARCHAR (10) ;
  ```

- Change the local column name:

  ```
  ALTER NICKNAME "ADMINISTRATOR"."DEMOGRAPHICS" ALTER COLUMN MILAGE LOCAL
  NAME MILAGENEW ;
  ```

B.7.6 Test the DB2 UDB for z/OS nickname

To test the DEMOGRAPHICS nickname, right-click it and click **Sample Contents** as shown in Figure B-63.

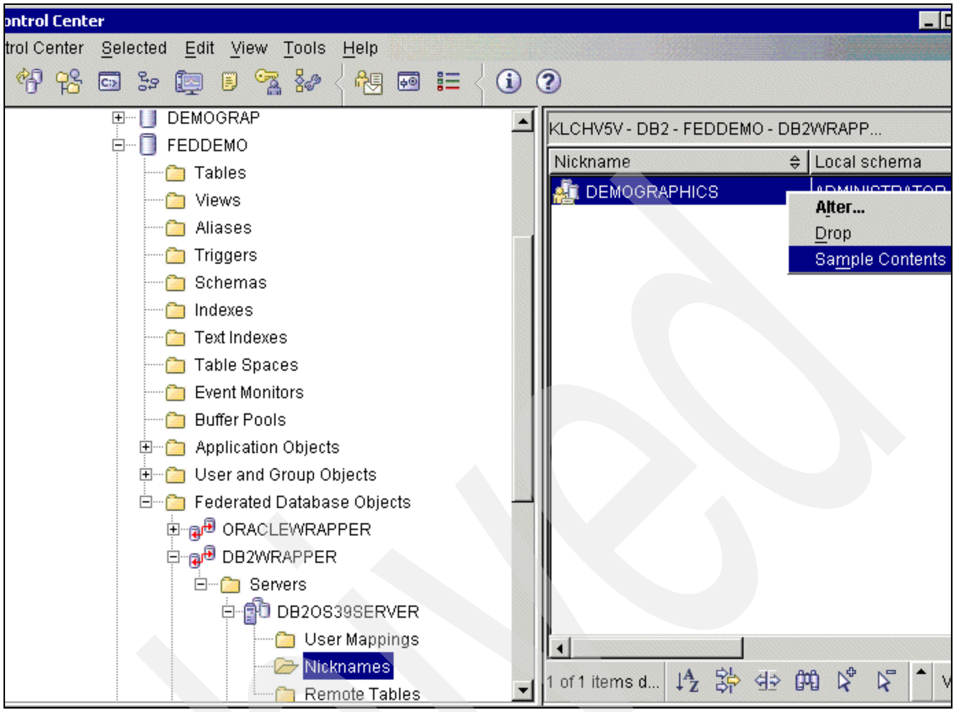


Figure B-63 DB2 UDB for z/OS - Sample contents of nickname

The results of the sample contents request is displayed in Figure B-64, which shows that the nickname creation was successful.

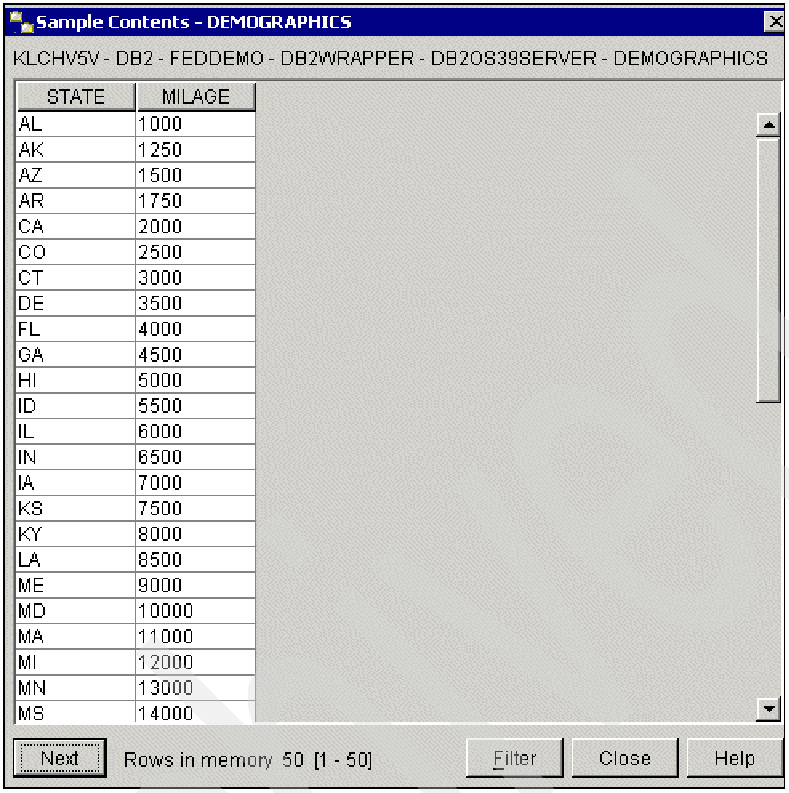


Figure B-64 DB2 UDB for z/OS - Sample contents results of nickname

B.8 Lotus Extended Search data source

The Lotus Extended Search (LES) product is a multi-tiered client/server system that provides extensive search and retrieval capabilities. With LES, you can enter a single request and search potentially thousands of data repositories and the Internet at the same time.

With LES, you can search the following types of data sources:

- Many popular Web search sites and news sites. If you need to search your intranet's search site, or other internal or external search sites, you can easily add support for doing so.
- Mail systems, such as those that you manage with Lotus Notes and Microsoft Exchange Server.
- Document management systems, such as DB2 Information Integrator for Content databases.

- Relational databases, such as IBM DB2, Oracle, Microsoft SQL Server, Microsoft Access, and other databases that comply with Open Database Connectivity (ODBC) standards.
- Full text indexes, such as those that you create with IBM WebSphere Portal, Domino Domain Index, Microsoft Index Server, and Microsoft Site Server.
- Lotus repositories, including Notes databases, Domino.Doc® libraries and cabinets, Lotus QuickPlace® places, and Lotus Discovery Server™ knowledge maps (K-maps).
- Instant messaging systems, such as Lotus Sametime®. This feature enables you to direct queries to knowledgeable persons, not just searchable data repositories.
- Lightweight Directory Access Protocol (LDAP) directories, such as those that you manage with IBM SecureWay®, Domino LDAP Server, and Exchange LDAP Server.
- File systems. You can search text files that are stored locally or on network drives. You cannot search compressed or encrypted files.
- Extended Search supports distributed, heterogeneous searching of structured and unstructured data through a single point of access.

This chapter describes the steps involved in configuring an LES data source, and creating a nickname for it.

The basic steps for configuring a data source are as shown in Figure 3-7 on page 49, and the following of these steps describe the configuration of an LES data source:

1. Create the LES wrapper.
2. Create the LES server definition.
3. Create the LES user mappings.
4. Create the LES nickname.
5. Test the LES nickname.
6. Registering the Extended Search custom function.

We used the DB2 Control Center on the Windows platform to configure the LES data source.

B.8.1 Create the LES wrapper

Navigate to the **FEDDEMO** federated database that was created during the DB2 II install as described in A.4.3, “Create the federated database” on page 222, and right-click the **Federated Database Objects,** and click **Create Wrapper** as shown in Figure B-1.

This action displays Figure B-65.

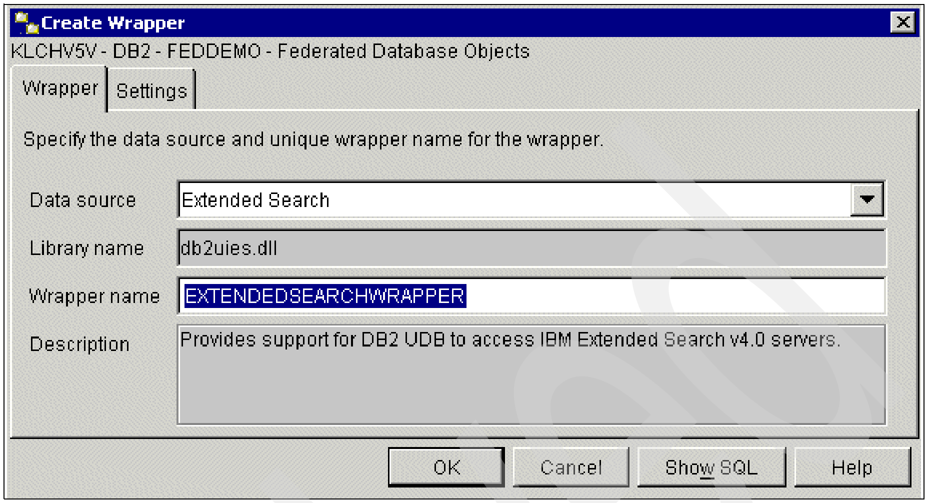


Figure B-65 Extended Search - Create wrapper

Select **Extended Search** in the Data source field and **EXTENDEDSEARCHWRAPPER** in the Wrapper name field in Figure B-65. Let the Settings tab default and click **OK** to create the EXTENDEDSEARCHWRAPPER wrapper.

Example B-21 shows the command line version of creating the EXTENDEDSEARCHWRAPPER wrapper.

Example: B-21 DB2 UDB for z/OS - Create wrapper statements

```
CONNECT TO FEDDEMO;
CREATE WRAPPER "EXTENDEDSEARCHWRAPPER" LIBRARY 'db2uies.dll';
CONNECT RESET;
```

B.8.2 Create the LES server definition

Select and expand the **EXTENDEDSEARCHWRAPPER**, right-click the **Servers** folder, and click **Create** similar to the actions taken for the ORACLEWRAPPER in Figure B-3 on page 230.

This action will display Figure B-66.

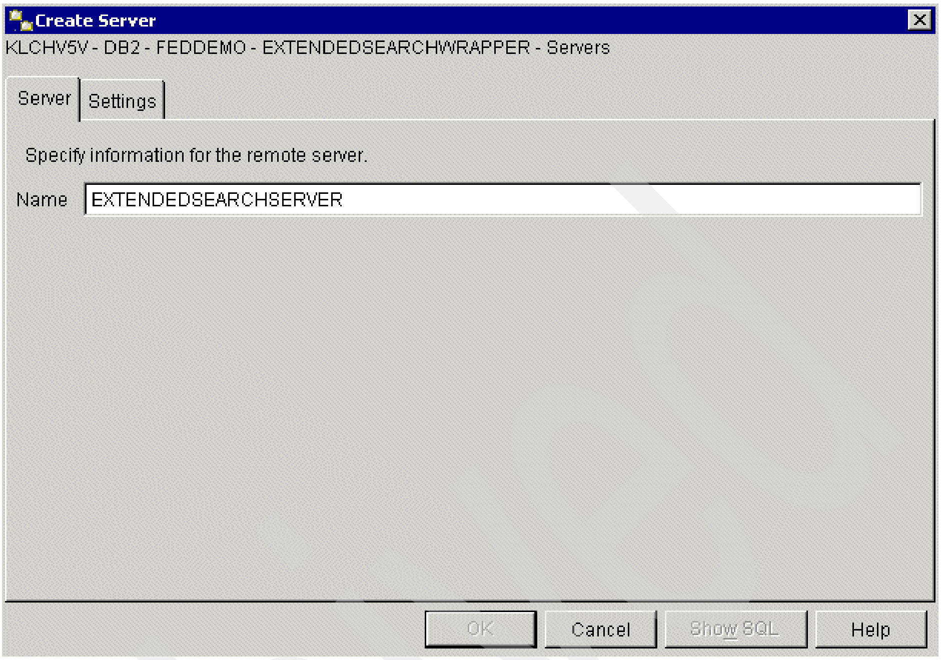


Figure B-66 Extended Search - Create server

The server definition requires the name of the server definition to be unique in the federated database. In our case we entered `EXTENDEDSEARCHSERVER` as shown in Figure B-66.

Switch to the **Settings** menu to complete the server definition as shown in Figure B-67.

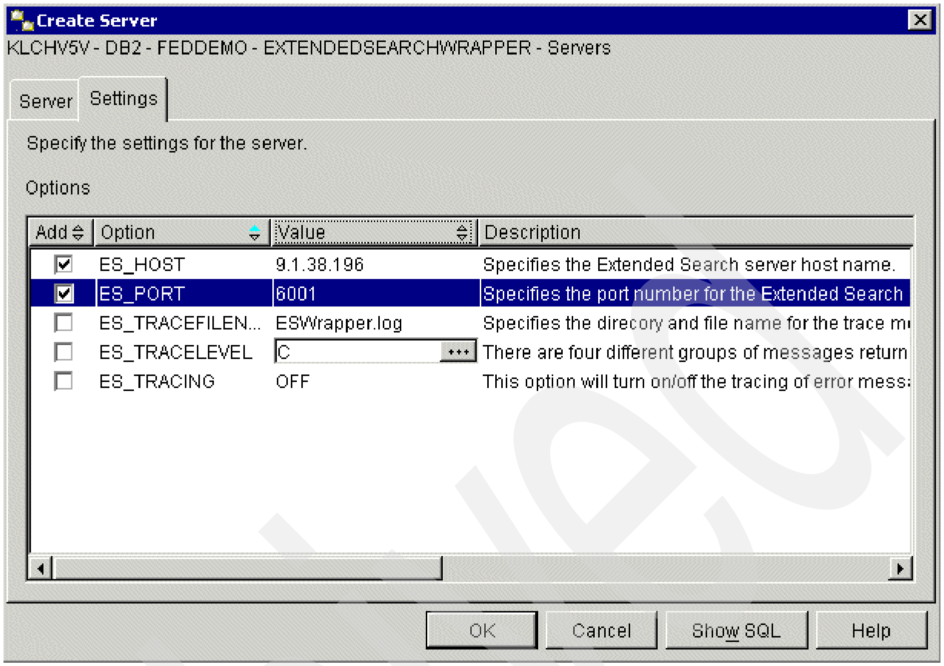


Figure B-67 Extended Search - Create Server Settings

For server options, only the ES_HOST and ES_PORT are required, the rest are optional.

Server options are used to describe a data source server. The LES server has a number of options as listed in Table B-8. These options may be set at server creation, or modified later as described in B.8.2.1, “Altering LES server definition and options” on page 314.

Table B-8 Lotus Extended Search server options

Server option	Description
ES_HOST	Specifies the fully qualified host name or IP address of the Extended Search server that you want to search. This option is required.
ES_PORT	Specifies the port number where this Extended Search server listens for requests. If you omit this option, the default value is 6001.

Server option	Description
ES_TRACEFILENAME	Specifies whether tracing should be enabled for error messages, warning messages, and informational messages that are produced by the remote Extended Search server. The default value, OFF, means that no trace messages will be logged.
ES_TRACELEVEL	If tracing is enabled, this option specifies the types of messages that will be written to the log file. The default value, C, logs only critical messages. You can enable and disable the following trace levels independently: C -- Critical error messages N -- Noncritical messages W -- Warning messages I -- Informational messages For example: ES_TRACELEVEL 'W' ES_TRACELEVEL 'CN' ES_TRACELEVEL 'CNWI'
ES_TRACING	If tracing is enabled, this option specifies the name of a directory and file where messages will be written. If you omit this option, the default value is the ESWrapper.log file in your DB2 temp directory (%DB2TEMPDIR%\ESWrapper.log or %DB2TEMPDIR%/ESWrapper.log).

Example B-18 shows the command line version for creating the EXTENDEDSEARCHSERVER server for your LES data source.

Example: B-22 Extended Search - Create server statements

```
CONNECT TO FEDDEMO;
CREATE SERVER EXTENDEDSEARCHSERVER WRAPPER "ESSERVER" OPTIONS( ADD ES_HOST
'9.1.38.196', ES_PORT '6001');
CONNECT RESET;
```

B.8.2.1 Altering LES server definition and options

You may modify the server options by selecting the **EXTENDEDSEARCHWRAPPER,** right-clicking **EXTENDEDSEARCHSERVER,** clicking **ALTER,** and switching to the **Settings** tab to view Figure B-68**.**

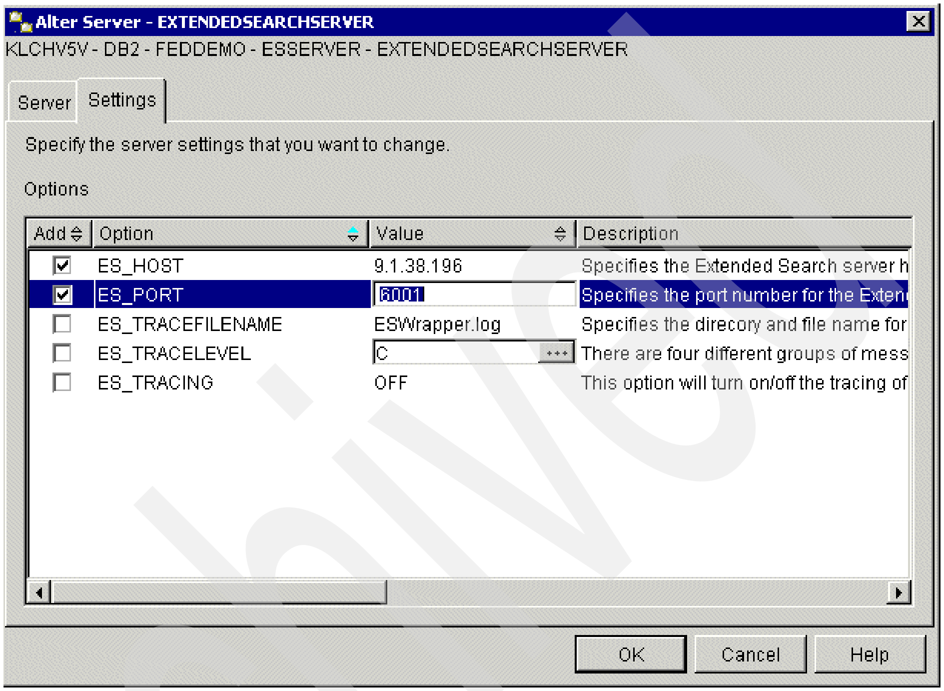


Figure B-68 Extended Search - Alter server options

Modify any of the server options and click **OK**.

B.8.3 Create the LES user mappings

Select and expand EXTENDEDSEARCHSERVER, right-click the **User Mappings** folder and click **Create**, which is similar to what is shown for **ORACLESERVER** in Figure B-6 on page 236.

This action displays the screen shown in Figure B-69.

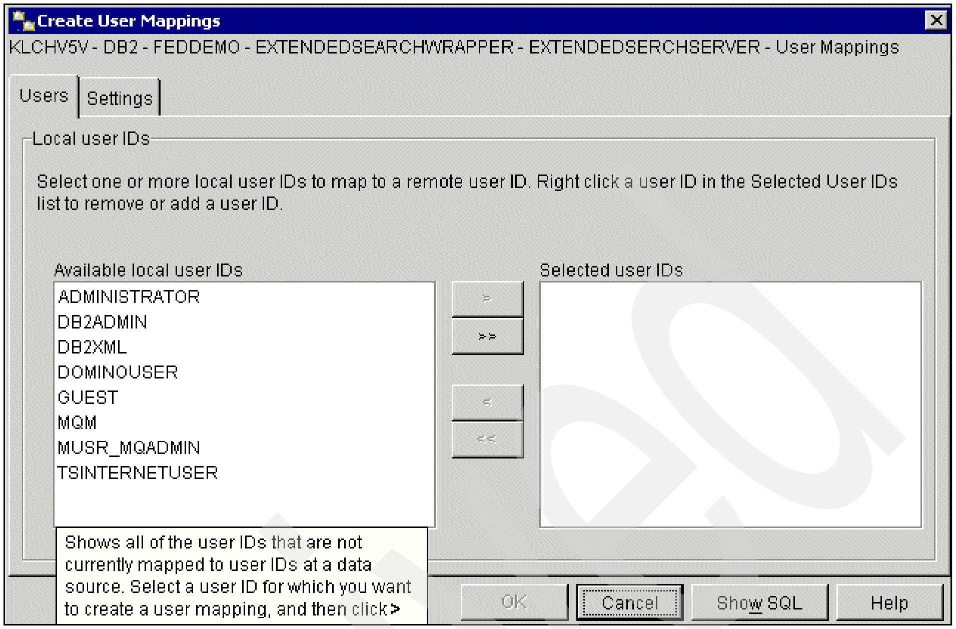


Figure B-69 Extended Search - Create user mapping dialog

Figure B-69 lists all user IDs available on the federated system. Select the user that sends the federated requests to the LES data source. We selected the ADMINISTRATOR user, and switched to the **Settings** menu to complete the user mapping as shown in Figure B-70.

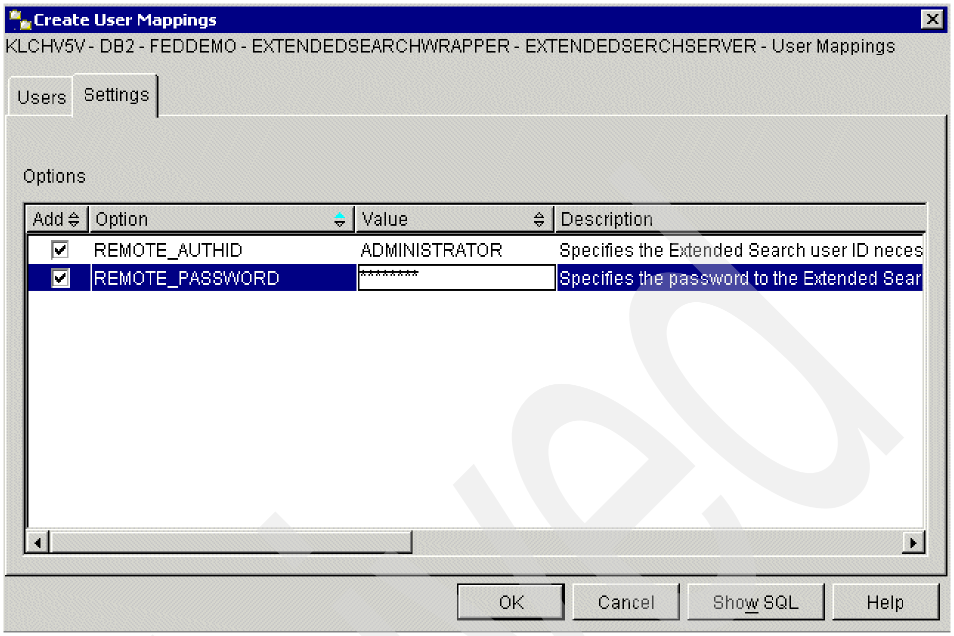


Figure B-70 Extended Search - Create user mapping settings

In Figure B-70, you need provide a valid user ID and password for the system where LES is running to enable the federated system to connect to the LES data source.

User mappings may be modified after creation time as described in B.8.3.1, “Altering the LES user mappings” on page 316.

Example B-23 shows the command line version of creating the user mapping for the LES data source.

Example: B-23 Extended Search - Create user mapping statements

```
CONNECT TO FEDDEMO;
CREATE USER MAPPING FOR "ADMINISTRATOR" SERVER "EXTENDEDSEARCHSERVER" OPTIONS (
ADD REMOTE_AUTHID 'ADMINISTRATOR', ADD  REMOTE_PASSWORD '*****');
CONNECT RESET;
```

B.8.3.1 Altering the LES user mappings

You can use the **ALTER USER MAPPING** statement to modify a user mapping, and is used to change the authorization ID or password that is used at the LES data source as follows:

```
ALTER USER MAPPING FOR "ADMINISTRATOR" SERVER "EXTENDEDSEARCHSERVER"
OPTIONS (SET REMOTE_AUTHID 'ADMINISTRATOR1' ) ;
ALTER USER MAPPING FOR "ADMINISTRATOR" SERVER "EXTENDEDSEARCHSERVER"
OPTIONS (SET REMOTE_AUTHID 'ADMINISTRATOR1' , SET REMOTE_PASSWORD '*****' )
;
```

User mappings may also be altered through the DB2 Control Center.

B.8.4 Create the LES nickname

Once the LES wrapper, server definition, and user mapping is complete, one needs to test the connection to the data source.

Select and expand **EXTENDEDSEARCHSERVER,** right-click **Nicknames** and click **Create** similar to the actions taken for ORACLESERVER in Figure B-9 on page 240.

This action opens up the dialog window shown in Figure B-71.

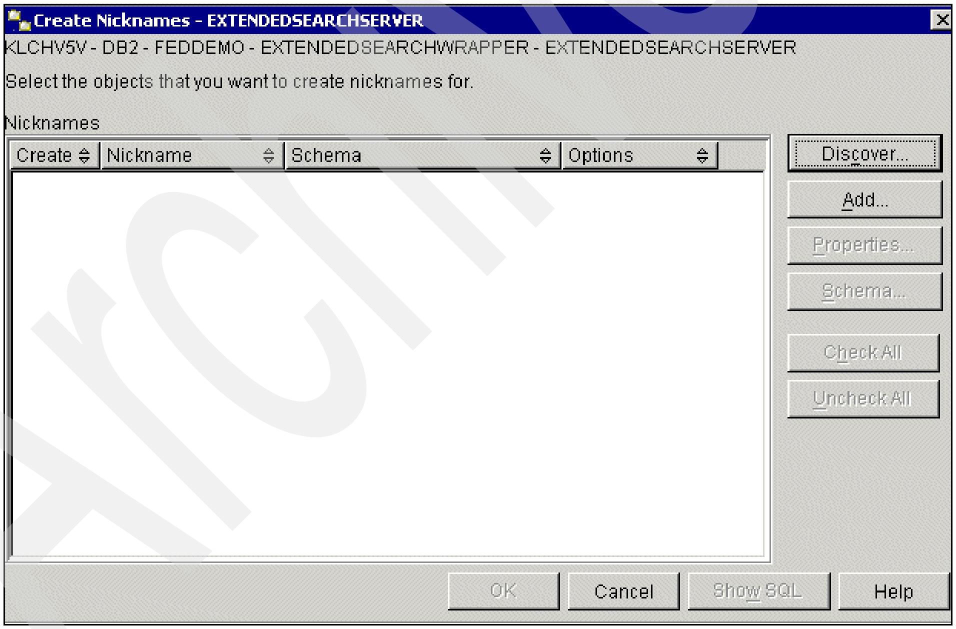


Figure B-71 Extended Search - Create nickname

Figure B-71 shows two options for adding a nickname as follows:

- Manually add a nickname (click **Add**) by specifying local schema.

- Use the discover functionality (click **Discover)** to be prompted for providing all the required information as follows:

  Click **Discover** to open up a dialog window to discover the LES sources by selecting the fields that you want to create a nickname for (as shown in Figure B-72).

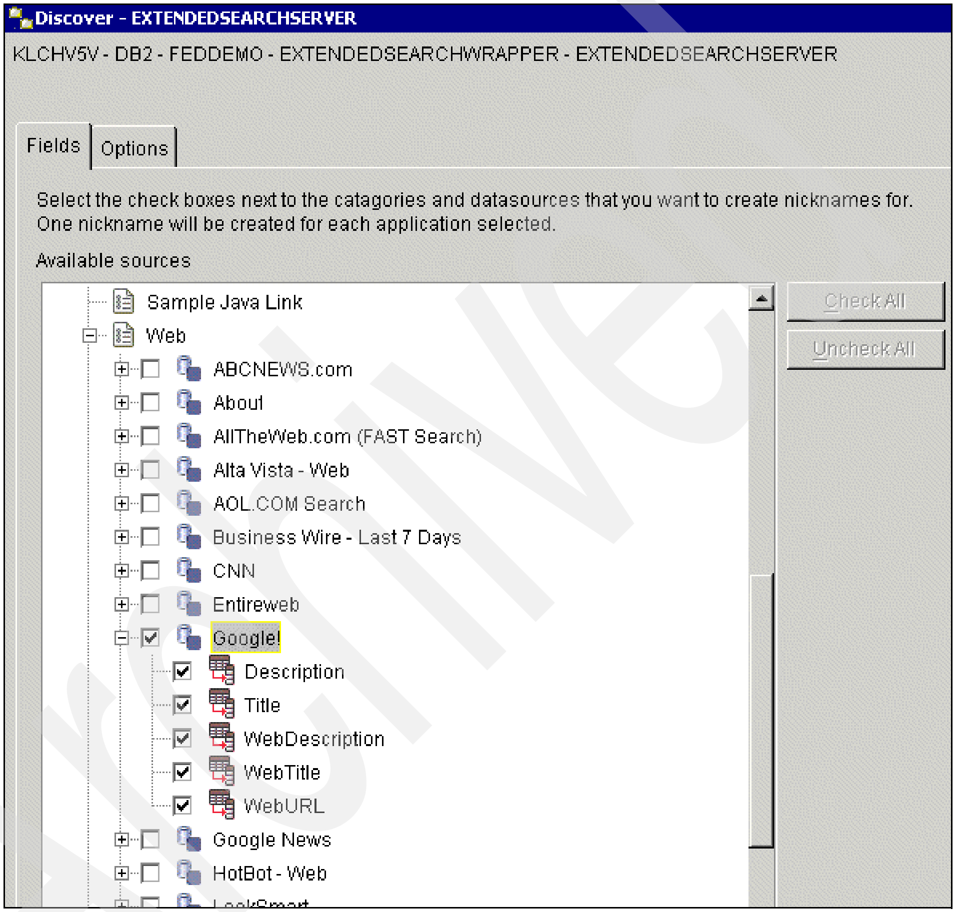


Figure B-72 Extended Search - Nickname discover fields

Select **Google** as the Web data source, and we let the Options to default by clicking **OK** to view Figure B-73.

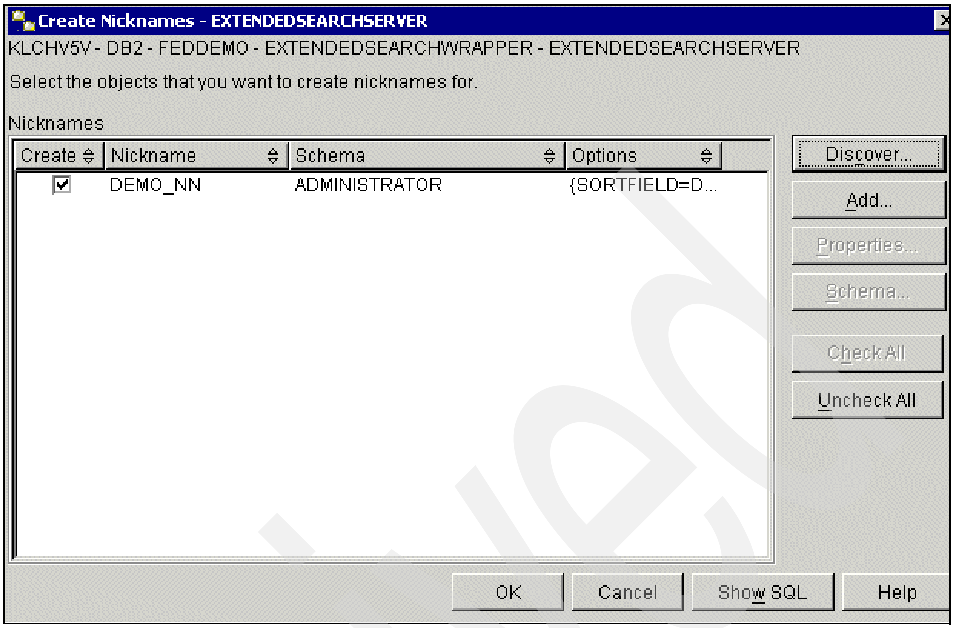


Figure B-73 Extended Search - Nickname created

Click **OK** to create the default DEMO_NN nickname as shown in Figure B-73.

Although a single nickname table can contain information about all the sources that are configured in LES, creating several nickname tables might be more useful. To use the full power of DB2 II, create a separate nickname for each type of data source that you plan to search with the LES wrapper.

For example, you might have one nickname for Web sources, one for Notes databases, one for file systems, and so on. By having separate nickname tables, you are better able to perform joins on the data that is returned to the wrapper, relate diverse sources based on field values, and integrate the result data with other data in your federated system.

Example B-24 shows the command line version of creating the DEMO_NN nickname for the EXTENDEDSEARCHSERVER.

Example: B-24 Extended Search - Create nickname statements

```
CONNECT TO FEDDEMO;
CREATE NICKNAME ADMINISTRATOR.DEMO_NN ( WEBURL VARCHAR (100) ,TITLE VARCHAR
(100) ,WEBDESCRIPTION VARCHAR (100) ,DESCRIPTION VARCHAR (100) ,WEBTITLE
VARCHAR (100) )  FOR SERVER "EXTENDEDSEARCHSERVER"  OPTIONS(SORTFIELD
'DOC_RANK' , MAXHIT '50' , SORTORDER 'A' , VERTICAL_TABLE 'NO' , APPLICATIONID
'Demo' , TIMEOUT '30' , TOTALMAXHIT '50' , DATASOURCE 'Google!');
```

```
CONNECT RESET;
```

B.8.4.1 Altering the LES nicknames

Figure B-74 shows the Alter Nickname dialog box for LES. You can modify the nickname as well as the nickname settings as shown in Figure B-75.

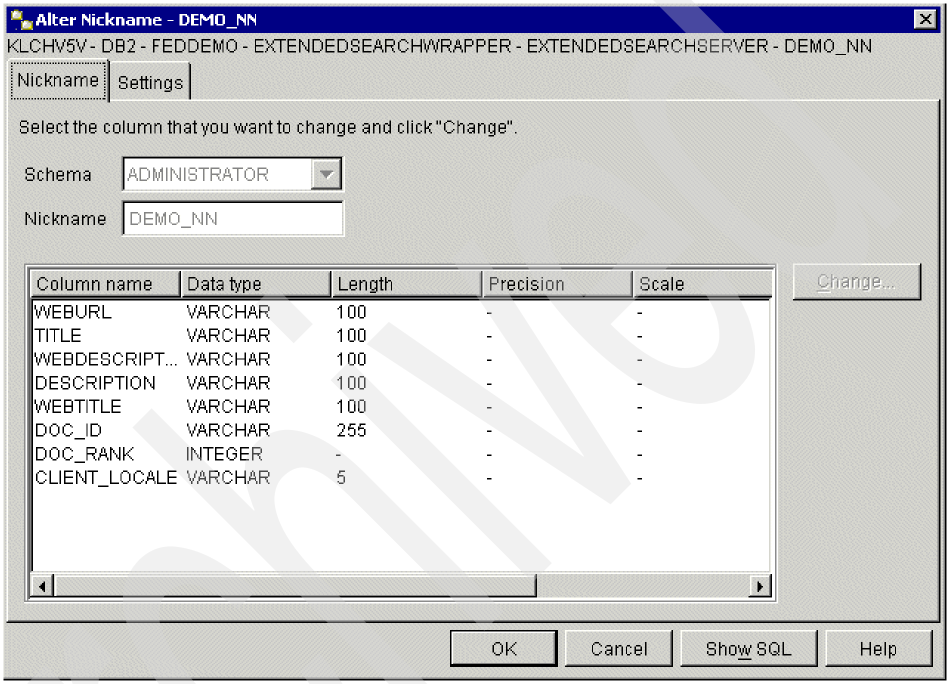


Figure B-74 Extended Search - Alter nickname

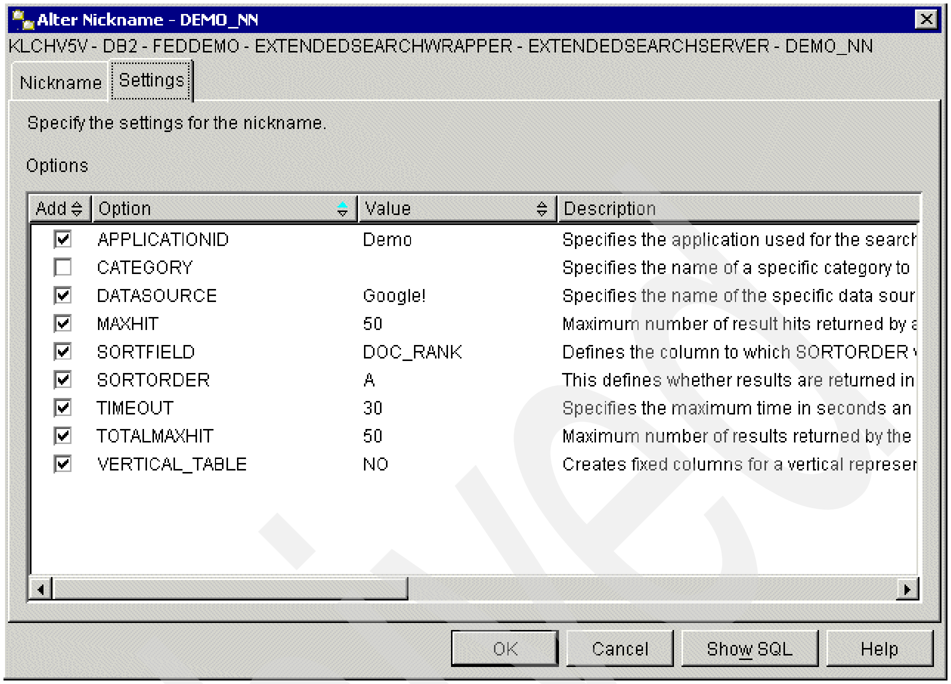


Figure B-75 Extended Search - Alter nickname settings

B.8.5 Test the LES nickname

To test the DEMO_NN nickname, issue the commands shown in Example B-25. This assumes that the ES_SEARCH custom function has been registered as described in B.8.6, “Register the Extended Search custom function” on page 322.

Example: B-25 Test LES nickname

```
connect to feddemo;
select * from demo_nn where eswrapper.es_search(doc_rank, ‘”ibm”’) =1;
```

Since we need to provide LES a search predicate, we cannot use the Sample Contents function of the DB2 Control Center, which we used in the other examples.

The results of this command are shown in Figure B-76 indicating a successful nickname creation.

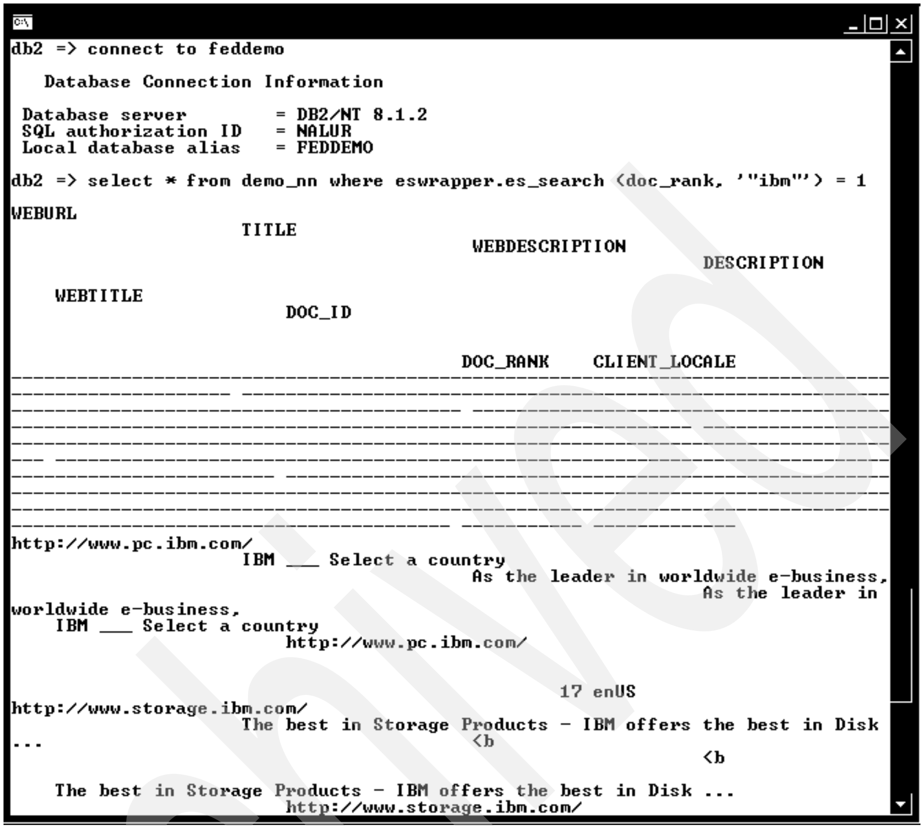


Figure B-76 Test DEMO_NN nickname results

B.8.6 Register the Extended Search custom function

This is an optional step in the main task for adding Extended Search data sources to a federated system.

Custom functions contain no executable code. After you register a function, you can refer to it in queries to alter default search behavior. The custom function for the Extended Search wrapper is ES_SEARCH, which enables you to specify precise search expressions and search content that is not defined as a column in the nickname table.

The restrictions with the use of ES_SEARCH are as follows:

- You can call the ES_SEARCH function only with a WHERE clause.

- The WHERE clause must contain at least one predicate that serves as a search predicate, either the ES_SEARCH function, or a predicate of type column-name operator constant.
- The ES_SEARCH function is a scalar function template. It must use the EQUAL (=) operator and the comparison value must be one (1).
- The first parameter in the ES_SEARCH function serves as an anchor value for identifying the nickname to which the function should be applied, such as the document's rank (DOC_RANK) in the search results. You must specify an INTEGER field for this parameter. This parameter, which does not get evaluated, is particularly important if the SQL query contains more than one nickname or a combination of nicknames and tables. For example:

```
SELECT * FROM ES_N1, ES_N2
WHERE ESWRAPPER.ES_SEARCH (ES_N1.DOC_RANK, '"IBM"') = 1 AND
ESWRAPPER.ES_SEARCH (ES_N2.DOC_RANK, '"IBM"') = 1
```

  This statement queries nicknames ES_N1 and ES_N2 looking for entries with "IBM" that occur in both data sources.

To register the Extended Search custom function, issue the following **CREATE FUNCTION** statement from the DB2 Command Line Processor:

```
CREATE FUNCTION ESWRAPPER.ES_SEARCH(INTEGER, VARCHAR(1024)) RETURNS INTEGER
AS TEMPLATE
```

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 327. Note that some of the documents referenced here may be available in softcopy only.

- *IBM WebSphere Portal V5 Handbook,* SG24-6098-00
- *Data Federation with IBM DB2 Information Integrator,* SG24-7052
- *Fundamentals of Grid Computing,* REDP-3613-00
- *A Practical Guide to DB2 Data Replication V8,* SG24-6828
- *Getting Started in Integrating Your Information,* SG24-6892
- *XML for DB2 Information Integration,* SG24-6994
- *IBM Life Sciences Solutions: Turning Data into Discovery with DiscoveyLink,* SG24-6290
- *IBM Informix: Integration Through Data Federation,* SG24-7032
- *Moving Data Across the DB2 Family,* SG24-6905
- *Architecting Portal Solutions,* SG24-7011
- *The IBM Enterprise Information Portal: A Practical 5749*
- *The IBM Enterprise Information Portal: A Cookbook,* SG24-6125

Other publications

These publications are also relevant as further information sources:

- *IBM Systems Journal Vol. 41, No. 4, 2002, Information Integration,* G321-01473
- *IBM DB2 Information Integrator Developer’s Guide Version 8,* SC18-7359
- *IBM DB2 Information Integrator Federated Systems Guide Version 8,* SC18-7364

- *IBM DB2 Information Integrator Installation Guide Version 8,* SC18-7036
- *IBM DB2 Information Integrator Migration Guide Version 8,* SC18-7360
- *IBM DB2 Information Integrator Solutions Guide Version 8,* SC18-7037
- *IBM DB2 Information Integrator Data Source Configuration Guide Version 8,* available as softcopy only from the Web site*:*

  http://www.ibm.com/software/data/integration/solution
- *IBM DB2 Information Integrator Installation Release Notes Version 8,* available as softcopy only from the Web site*:*

  http://www.ibm.com/software/data/integration/solution
- *IBM DB2 Universal Database Administration Guide: Planning Version 8,* SC09-4822
- *IBM DB2 Universal Database Command Reference, Version 8,* SC09-4828
- *IBM DB2 Universal Database Message Reference Volume 1 Version 8, G*C09-4840
- *IBM DB2 Universal Database Replication Guide and Reference Version 8, Release 1,* SC27-1121
- *IBM DB2 Universal Database SQL Reference Volume 1 Version 8,* SC09-4844
- *IBM DB2 Universal Database SQL Reference Volume 2 Version 8,* SC09-4845
- *IBM DB2 Universal Database System Monitor Guide and Reference Version 8,* SC09-4847
- *Information On Demand* DB2 Magazine article by Holly Hayes and Nelson Mattos, Quarter 3, 2003, Volume 8, Issue 3 available at: http://www.db2mag.com/story/showArticle.jhtml?articleID=12803103
- *Portlet Development Best Practices and Coding Guidelines* white paper dated March 17th 2003 by Marshall Lamb, available from the Web site: ftp://207.25.253.53/1/wsdd/pdf/PortletCodingGuidelines.pdf
- *Using the federated database technology of IBM DB2 Information Integrator*, white paper by Anjali Grover, Eileen Lin and Ioana Ursu, available from the Web site:

  http://www.ibm.com/software/data/pubs/papers/#iipapers

Online resources

These Web sites and URLs are also relevant as further information sources:

- DB2 Information Integration products:

  http://www.ibm.com/software/data/integration/solution
- DB2 offerings information:

  http://www.ibm.com/common/ssi/
- DB2 Information Integration support site:

  http://www.ibm.com/software/data/integration/db2ii/support.html
- Systems Journal on Information Integration:

  http://www.research.ibm.com/journal/sj41-4.html
- IBM DiscoveryLink offering:

  http://www.ibm.com/solutions/lifesciences/solutions/discoverylink.html
- DataDirect ODBC driver for UNIX 4.2:

  http://www.datadirect.com
- OpenLink MultiTier ODBC Client 5.0:

  http://www.openlink.co.uk
- Samba Server 2.2:

  http://www.samba.org
- IBM developerWorks:

  http://www.ibm.com/developerworks/db2

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

Y

IBM

Redbooks

WebSphere Portal and DB2 Information Integrator: A Synergistic Solution

(0.5” spine)
0.475”<->0.875”
250 <-> 459 pages

WebSphere Portal and DB2 Information Integrator: A Synergistic Solution

WebSphere Portal architecture overview

DB2 Information Integration architecture overview

WebSphere Portal and DB2 II synergy scenarios

Portals provide users with a single point of access to multiple types of information and applications through a Web interface. IBM WebSphere Portal for Multiplatforms is one of the industries' most comprehensive portal offerings. It contains a wide range of portal technologies that help you develop and maintain B2C, B2B, and B2E portals. WebSphere Portal provides considerable flexibility in addressing dynamic changes in marketplace conditions through advanced integration, Web services, and collaboration technology.

DB2 Information Integrator provides federated data access to a diverse range of heterogeneous data sources including relational data sources such as the DB2 family, Oracle, SQL Server, and Teradata, and semi-structured data from WebSphere MQ messages, XML documents, Web services, Microsoft Excel, flat files, ODBC or OLE DB sources, plus a variety of formats unique to the life sciences industry. It also provides integrated support for Lotus Extended Search, which enables broad content access to a variety of content repositories including DB2 Content Manager, as well as e-mail databases, document repositories, third-party Internet search engines, and LDAP directories.

This IBM Redbook provides examples of the synergy between WebSphere Portal Server and DB2 Information Integrator in delivering portal solutions, and will include sample portlets exploiting SQL, UDFs, JDBC, and Web Services.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6433-00 ISBN 0738498432