IBM

Siebel 7.8 with IBM DB2 UDB V8.2 Handbook

- Step-by-step installation and configuration procedures
- Database administration, monitoring, and tuning methods
- High availability options and setup steps

Whei-Jen Chen
Priti Desai
Ujwala Devi
Sreeman Kancherla
Venkatram Menakuru
Raghu Ramaswamy
Scott Saunders
Arjun Sirohi

Redbooks

ibm.com/redbooks

International Technical Support Organization

Siebel 7.8 with IBM DB2 UDB V8.2 Handbook

November 2005

SG24-6766-00

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

First Edition (November 2005)

This edition applies to DB2 UDB V8.2, Siebel 7.8, AIX 5.2, Windows 2000 Server, and Windows Server 200.

Contents

Figures

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:

@server®	C Set ++®	IBM®
@server®	DB2 Connect™	MQSeries®
Redbooks (logo) ™	DB2 Universal Database™	POWER™
pSeries®	DB2®	Redbooks™
z/OS®	DRDA®	Summit®
zSeries®	Enterprise Storage Server®	SysBack™
AIX 5L™	HACMP™	Tivoli®
AIX®	Informix®	WebSphere®

The following terms are trademarks of other companies:

Siebel and the Siebel Logo are trademarks of Siebel Systems, Inc., and may be registered in certain jurisdictions.

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

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

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

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

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

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

Preface

This IBM Redbook gives a broad understanding of the integration of Siebel 7.8 and IBM DB2® Universal Database (UDB) V8.2. It covers planing, installation, administration, monitoring, performance tuning, and high availability of DB2 UDB V8.2 in a Siebel environment. This book includes the following chapters:

- **Chapter 1** outlines the partnership between Siebel Systems and IBM® and the benefits of using DB2 UDB V8.2 to support the Siebel Enterprise.
- **Chapter 2** introduces the components of the Siebel Enterprise and the DB2 UDB V8.2 architecture. The most commonly used Siebel components are described and the Siebel Object Architecture is explained. Detailed information on the DB2 UDB V8.2 architecture, including memory, process, and storage concepts, are explained.
- **Chapter 3** provides a brief overview for installation and planning considerations of DB2 UDB V8.2 in a Siebel environment. The Siebel release matrix, database storage layout, and installation check list are discussed.
- **Chapter 4** describes the installation and configuration procedures for DB2 UDB on AIX® and Windows® environment. It also provides Siebel upgrade steps.
- **Chapter 5** covers the Siebel object-oriented architecture. It describes information on methods to populate and maintain data in Siebel tables. Information on ensuring data integrity and data quality is included along with data archival techniques and products.
- **Chapter 6** covers database administration information including backup, recovery, statistics collection, table reorganization, and query control.
- **Chapter 7** covers the description of the tools, how to use the tools, and how to interpret the output generated by the tools for potential performance problems. Learn how to monitor the system and how to use the tools provided by DB2 UDB and Operating Systems for tuning and maintaining your system.
- **Chapter 8** introduces performance tuning for DB2 UDB in a Siebel environment. The tuning best practices and scenarios are provided.
- **Chapter 9** discuss the high availability options, the installation, and configuration procedures for HACMP™, MSCS, and DB2 HADR.
- **Chapter 10** provides information about the components of Siebel Analytics and where these components fit in the overall scheme with Siebel Enterprise. Specifically, the Analytics database objects created and maintained on the OLTP database are discussed along with the impact these may have on the

Siebel CRM users. Useful practical advice is also provided to achieve optimum performance on the OLTP database during the Analytics Extract, Transform, Load (ETL) process.

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.

Figure 1 Left to right: Scott, Raghu, Arjun, Sreeman, Venkat, Whei-Jen, Priti, and Ujwala

Whei-Jen Chen is a Project Leader at the International Technical Support Organization, San Jose Center. She has extensive experience in application development, database design and modeling, and DB2 system administration. Whei-Jen is an IBM Certified Solutions Expert in Database Administration and Application Development as well as an IBM Certified IT Specialist.

Priti Desai is a DB2 Technical Consultant with the Information Management Partner Enablement organization at IBM Silicon Valley Lab, where her main focus

is assisting IBM Business Partners with high-water benchmarks, performance tuning, application migration, and training.

Ujwala Devi is a project lead with IBM Global Services India. She holds a Bachelor of Engineering in Computer Science and a Master of Business Administration in Finance. She has over five years of experience in Siebel. She is Siebel 2000 and Siebel 7 certified. She has implemented six Siebel projects in the United States, Germany, Jordan, and India.

Sreeman Kancherla is a DB2-Certified I/T Architect working for IBM Global Services, Charlotte, NC. He has over 12 years of experience as a DBA on different DBMS. He has over five years of experience in implementing Siebel on DB2 UDB. He is the Lead DBA and DB Architect, working on internal Siebel implementation at IBM. He has a Masters Degree from UNC-Charlotte in Computer Science.

Venkatram Menakuru is a Siebel Certified Consultant currently working as a Siebel Operations Architect for IBM Global Services. He has over four years experience deploying and supporting the Siebel CRM application. He also has nine years of experience in the I/T industry providing UNIX and Web Server Administration and SAP Basis support. He has an Engineering Degree from Bangalore University and a Masters Degree in Computer Science from the University of Missouri-Kansas City. He is also a certified SAP Basis Consultant and a certified IBM DB2 Associate.

Raghu Ramaswamy is a DB2 Certified DBA working for IBM Global Services, Bangalore, India. He has over eight and a half years of experience in the PLM and CRM domain, primarily focusing on database and system administration. He is currently working for the Siebel Technology Center as lead DBA. He has a Bachelors Degree from Bangalore University in Mechanical Engineering.

Scott Saunders is a Principal Architecture Specialist with Siebel Expert Services located in St. Petersburg, Florida. With over 20 years of professional experience, he is skilled as a technical architect, troubleshooter, and performance analyst for Siebel Applications and has been a member of the Siebel Systems team for more than five years. He has provided guidance on performance enhancements, capacity planning, design and architecture best practices for many of Siebel's largest deployments. He has written numerous Tech Notes and FAQ on DB2 topics on DB2 implementation issues. He has been a presenter at International DB2 Users Group conferences and has been previously published in the IDUG Journal.

Arjun Sirohi is an Architecture Specialist with Siebel Expert Services located in Bellevue, Washington, USA. He is a Siebel Certified Consultant as well as Siebel Certified Master Analytics Consultant. He has worked with DB2 UDB for over seven years and is an IBM Certified Solution Designer - DB2 Business

Intelligence and also an IBM Certified Database Administrator - DB2 UDB V8.1. With over 16 years of professional IT experience, he is skilled as a technical architect, troubleshooter, and performance analyst for DB2 UDB and Siebel Applications both for transactional databases as well as data warehouses. Before joining Expert Services, he was a member of the Siebel Systems Engineering team in San Mateo, California, for over four years during which time he provided guidance on performance, scalability and reliability enhancements, design and architecture best practices for many of Siebel's software releases on DB2 UDB both for CRM as well as Business Analytics.

Acknowledgement

The authors express their deep gratitude to the following people for their support and contributions to this project:

Deb Jenson
Paul Dopp
Sunil Sabat
Hareendranath Kattana
Dilip Kikla
IBM Software Group, Information management, DB2, USA

Noureddine Brahim
IBM Software Group, Toronto Laboratory, Canada

Svetlana Sicular
Lei Ni
Jannie Chan
Sandeep Deshmukh
Eric Aronson
Donald McInturff
Sreenivasan Muneswara
Vijayakumar Ranganathan
Siebel Systems, Inc.

Kazuchika Okamura
IBM Global Services, Japan

Ramesh Chitor
Jubal Kohlmeier
Mark Greenberg
IBM System & Technology Group, USA

Emma Jacobs
Sangam Racherla
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 e-mail 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

This chapter discusses:

- Partnership between IBM and Siebel
- Benefits of using DB2 UDB with the Siebel Enterprise
- Introduction to new features in DB2 UDB V8.2

1.1 Partnership between IBM and Siebel Systems

IBM and Siebel Systems deliver proven customer relationship management (CRM) solutions that have set global standards for delivering more business value and a higher return on investment (ROI). Today, the IBM Siebel alliance is enabling companies to realize the benefits of transformational change in their marketing, sales, and service organizations.

Since partnering in 1999, IBM and Siebel Systems have collaborated on more than 2,000 CRM engagements addressing all aspects of the application life cycle - from strategic consulting, to ongoing transformation, management, hosting, outsourcing, and specialized systems-management services.

1.1.1 Ongoing activities

IBM and Siebel collaborate on many levels to provide customers with development expertise and support.

- Joint development and engineering
  - Technical enablement resources on-site
  - Siebel at DB2 Laboratories
  - Integration centers for software certifications and special projects
  - Joint product design reviews
  - Escalated PMR processes
  - Benchmarks, Proof-of-concepts, and stress tests at IBM labs
  - Closed-loop processes for implementing Siebel requirements in each DB2 release
- Integrated support
  - Direct hand off of customer problems
  - Linked teams for early Siebel release and DB2 adopters
  - Review and feedback on new Siebel releases at IBM
  - Joint Siebel-IBM education for Siebel and IBM experts
  - Publication of IBM Redbooks and Redpieces
- IBM international competency center
  - Server sizing and guidance
  - Pre-Sales support and customer briefings
  - Demo development and deployment
  - Publication of technical white papers

1.1.2 Recent developments

Within the past year, IBM and Siebel have entered into several strategic agreements including:

- Siebel support for DB2 UDB on the Linux® operating system.
- Signing a five year agreement to create a client-focused delivery center to coordinate world-wide delivery capabilities.
- After five years of collaboration in public sector implementations, IBM and Siebel have agreed to jointly deliver case management solutions to the Social Services and Social Security market.
- At the 2005 IBM Executive Service-Oriented Architecture (SOA) Summit®, IBM and Siebel announced a joint plan to deliver SOA-based composite applications leveraging the IBM WebSphere® Portal.

1.2 Benefits of DB2 UDB V8.2 with the Siebel Enterprise

DB2 Enterprise Server Edition is a multiuser version of DB2 UDB that provides the ability to create and manage non-partitioned or partitioned database environments. IBM database systems can manage high volumes of data and provide benefits such as high scalability, increased performance, high availability, and failover support.

Customers who choose DB2 UDB join a DB2 UDB user community of over 60 million users consisting of more than 80 percent of the Fortune 500 and 100 percent of the Fortune 100. All Siebel solutions are developed and tested using DB2 UDB.

Following are a few of the DB2 UDB strengths:

Reliability

DB2 UDB provides best in class reliability to protect valuable corporate information assets. Sophisticated recovery, backup and restore features provide recoverability from system failures. High-availability features provide proven support for systems that cannot tolerate system outages. Online system support features allow systems administrators the ability to apply needed maintenance while the system remains available.

Scalability

DB2 UDB is the only DBMS that is used across all platforms supported by Siebel Business Applications. DB2 UDB is scalable from small business implementations to massive enterprise platforms. There are implementations in

production supporting both small businesses and large enterprises with tens of thousands of concurrent Siebel users and systems of every scale in between.

Performance

The DB2 UDB optimizer is a proven technology for providing optimal performance for dynamic and static SQL. (Siebel uses dynamic SQL only.) Robust tools are provided to assist administrators in providing peak throughput to maximize implementation ROI.

1.2.1 DB2 UDB V8.2 enhancements

With the release of DB2 UDB Version 8.2, administrators gain access to numerous enhancements to DB2's already robust scalability, performance, availability, integrated tools, and functions.

Features that can be important to administrators of Siebel applications include:

- The ability to use DB2 `LOAD` utility to populate individual Enterprise Integration Manager (EIM) tables. This eliminates the need to separate EIM tables into different table spaces to be able to load them concurrently. The data in the table being loaded now also can be queried while the load is in progress.
- Dual logging has been extended to all supported platforms. Database administrators now have control over the placement of the mirrored log. Maximum log size has been increased from 32 GB to 256 GB to support much larger loads of long-running activity.
- Siebel applications have specific code page requirements. In the event that an implementation is accidentally built on the wrong code page, DB2 UDB now provides the ability to restore to a different code page, greatly facilitating recovery should an incorrect code page be specified.
- The enhanced DB2 Design Advisor includes Materialized Query Table advisor, Multi-Dimensional Clustering advisor, and Partitioning advisor, which can be used individually or in combination to design the indexes to achieve the optimal performance goal.
- Two new management by exception features will help database administrators proactively respond to system problems before outages can occur:
  - The Health Monitor can be preconfigured to alert administrators or take specified actions when an exception is detected.
  - The Health Center is the graphical user interface to the Health Monitor. Health Beacons can be accessed by Web browser or personal digital assistant (PDA) to provide instant administrator response to issues as they arise.

- Type-2 indexes will improve concurrency by eliminating next-key locking. Indexes can now exceed 255 bytes, providing greater flexibility in index design.
- Of interest to administrators will be the ability to use SQL to perform detailed analysis of event monitor information. This can reduce the time required to find exceptions in large systems.
- Online buffer pool management may be of interest to administrators of applications with varying workloads. Buffer pools can be expanded, contracted, created, and dropped dynamically. Resources can be shifted from one area to another as needed. For example, in an environment with a large EIM requirement during the evening hours, online buffer pool management could shift memory resources to dedicated buffer pools or other memory structures such as `SORTHEAP` for the duration of the EIM activity to reduce elapsed time.
- Rename index is a potentially time-saving feature. In testing indexes performance, database administrators often prefer to build the index directly in a test environment to determine the effectiveness of the index design prior to adding the index to Siebel Tools. For cases where the index is proven to be useful, the index could be added to Siebel Tools and `DBCHCK` (A siebel tool to check that physical database schema matches the Siebel Repository) used to verify that the index name and index design were identical. If the index names do not match, the index in the physical schema can now be easily renamed to match the name in Siebel Tools, eliminating the need to drop and rebuild the index.
- DMS containers can be dropped, expanded, and contracted as needed. Commonly after a initial EIM data load, EIM storage will exceed ongoing requirements. With DB2 UDB V8.2, it is now easy to reclaim these resources.
- The ability to dynamically manage database and database manager configuration parameters increases resource flexibility. Over fifty parameters, including important features such as `LOCKLIST` and `SORTHEAP` can be changed as needed.

Numerous new and improved features of DB2 UDB V8.2 will make support of Siebel implementations easier while enhancing reliability, scalability, and performance.

2

Architecture

This chapter discusses the following topics:

- Architecture of the Siebel enterprise
- DB2 UDB architecture
  - Process concepts
  - Storage concepts
  - Memory concepts

2.1 Architecture layout of Siebel Enterprise

Figure 2-1 shows the major components of the Siebel 7.8 deployment architecture. Great flexibility is available to allow customers to adapt their servers to suit business needs for performance, scalability, and availability.

Figure 2-1 shows a typical deployment architecture. It is not required to spread components over a large number of devices unless a given implementation requires this level of hardware support. For some environments, components can be placed on the same device or within the same partition of a server.

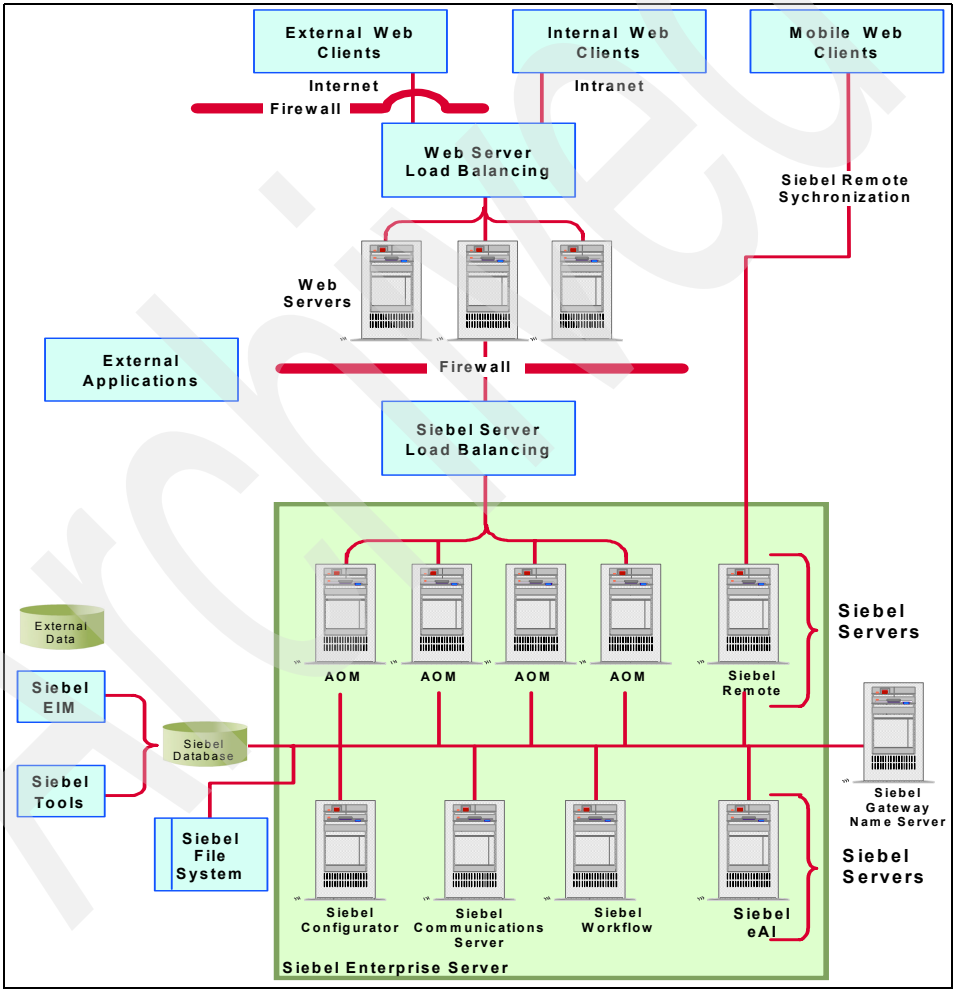

Figure 2-1 Siebel 7.8 architecture[1]

[1] Siebel Systems, Reprinted by Permission.

A brief overview of the most commonly used components of Siebel 7.8 follows, beginning with Siebel Clients, Siebel Web Server Extension, Siebel Enterprise Server, Siebel Server, Siebel Gateway Name Server, Siebel File System, Siebel Database, and finishing with a discussion of the Siebel object architecture.

2.1.1 Siebel Clients

There are five types of Siebel clients. The Siebel dedicated client is used only in some specific circumstances, usually by developers and administrators. The probability of a dedicated client being used by a user is low.

There are four Siebel Web Clients specifically designed to provide Web access to Siebel data:

Siebel Web Client

The Siebel Web Client uses the Web server to connect to the Siebel Enterprise server. Database connections are created and maintained by the servers comprising the Siebel enterprise.

These are deployed in one of two modes:

- **Standard interactivity**

  The standard interactivity client is often used for interaction with customers, partners, and other users who are not employees of the customer. The user experience will be similar to a traditional Web application. Most interactions involve a page refresh each time a new record is accessed. No software is required on the Web client aside from a standard browser.

- **High interactivity**

  For some employee facing implementations, such as users of Siebel Call Center, standard interactivity may not meet usability requirements. The high interactivity client supports fewer page refreshes and implicit commit to improve performance. High-interactivity clients require ActiveX® controls and JavaScript™ routines which are downloaded automatically.

Siebel Mobile Web Client

The Mobile Web Client also uses a standard browser. Additional software is installed which contains Siebel software needed to run the required services on the client. This includes a local database and local file system.

The Mobile Web Client connects periodically to the database through the Remote Server. Data that has been changed on the Siebel Database is sent to the Mobile Web Client. Changed data from the Mobile Web Client is routed to the

Siebel Database. Data is synchronized on both the Mobile Web Client and the Siebel database asynchronously.

Siebel Wireless Client

The Siebel Wireless Client is an Internet-enabled device running a specialized Siebel Web Client. Several device types are supported including personal digital assistants and Internet-enabled mobile phones. See the *Siebel Wireless Administration Guide* for a list of Siebel Business Applications that can be enabled for wireless access. A list of supported browsers can be found on Siebel Support Web in the *System Requirements and Supported Platforms* documentation.

Siebel Handheld Client

Used only by field technicians, the Siebel Handheld Client is a version of the Siebel Mobile Web Client. The Siebel handheld client has much of the functionality of the Mobile Web Client using the same configuration and data relationships. Siebel Handheld Client runs on devices using the Windows CE operating system.

2.1.2 Siebel Web Server Extension

The Siebel Web Server Extension (SWSE) is a plug-in for third-party Web servers that flags requests for Siebel information for routing to the Siebel servers. The SWSE provides round-robin load balancing for the Siebel servers.

If the implementation is multi-lingual, language packs installed on the Siebel servers must also be installed on the Web servers. Implementors may be able to avoid installing all language packs on every Web server if a mechanism is in place to route the requests to the correct Web server.

For information about supported hardware, operating system platforms' Web browsers, and Web servers, please refer to *Siebel System Requirements and Supported Platforms* found in Siebel SupportWeb site at:

http://supportweb.siebel.com

2.1.3 Siebel Enterprise Server

The Siebel Enterprise Server is a logical group of Siebel servers. All the Siebel servers in the enterprise will be configured, managed, and monitored as a single logical group. The Siebel Administrator can start, stop, and monitor all the Siebel servers within the enterprise as well as set server parameters.

The Siebel servers function as application servers. The server components that run on a given Siebel server define the function or functions of that Siebel server.

2.1.4 Siebel Server

Server components or server component groups installed on a Siebel Server determine the services and applications provided by that server. Server components can run in one of three modes:

- **Interactive mode**: Interactive components start tasks automatically in response to user requests. The tasks persist for the duration of the user session. Application Object Managers (OM) and Synchronization Manager are examples of interactive mode tasks.
- **Background mode**: Background tasks are usually called by interactive mode tasks and run until explicitly shut down. Examples include Transaction Router and Workflow Monitor Agent.
- **Batch mode**: Batch components handle asynchronous work requests. The component exits when the task is completed. Database Extract and Enterprise Integration Manager (EIM) are examples of batch mode tasks.

A specialized server component is the Siebel Connection Broker (SCBroker). SCBroker provides load balancing for multiple Application Object Managers (OM).

The Siebel server runs as a system service under Windows and as a process on Unix operating systems. This system service or process monitors and controls the state of all server components on that Siebel server. Each Siebel server is one instantiation of the Siebel server system service or process within the Siebel enterprise server.

Interactive and batch mode components can be configured to run as multiple processes or as multi-threaded processes. Background mode components can only be run as multiple processes.

For more information about administration of Siebel server services or processes, please consult the *Siebel System Administration Guide* in Siebel Bookshelf.

Application Object Manager

Application Object Manager (OM) is one of the most important and most commonly used types of server components. OM runs on interactive mode and processes user requests. OM components are application-specific or server-specific. For example, the Siebel Employee Relationship Manager component group contains the Employee Relationship Object Manager which provides the session environment in which the application runs. OM components also contain a data manager and the Siebel Web Engine.

An OM component is implemented as a multi-threaded process on the Siebel server. One or more AOMs are started by a parent process according to the OM configuration. The terms multi-threaded server, MT server, and multi-threaded process are used interchangeably.

OM threads reside in a pool and are assigned to user sessions or can be created as needed. A few OM threads are dedicated to housekeeping tasks.

2.1.5 Siebel Gateway Name Server

In a production environment, there can be only one Siebel Gateway Name Server installed. The Siebel Gateway Name Server is usually deployed for high availability using a cluster configuration.

The Siebel Gateway Name Server stores important information about the Siebel servers using both non-persistent and persistent methods.

Non-persistent information

The Siebel servers use the Siebel Gateway Name Server as the dynamic address registry. When a Siebel Server is started within the Siebel Enterprise, its network address is stored in the non-persistent address registry of the Siebel gateway name server.

Server components search the Siebel gateway name server address registry to find the address and availability of Siebel servers. At shutdown, a Siebel server removes its information from the address registry.

Persistent information

The Siebel Gateway Name Server contains a file called *siebns.dat*. This file contains Siebel Server configuration information such as connectivity information, definitions and assignments of components and component groups, as well as operational parameters. As Siebel Server information changes, the `siebns.dat` file is updated with the new configuration details.

2.1.6 Siebel File System

The Siebel File System stores document files, Siebel configurator models, Web template definitions, and other files that are not suitable for storage in a database management system. Siebel user preferences are also stored in the Siebel File System.

2.1.7 Siebel Database

The Siebel Database contains application data contained within the physical schema including Repository objects. The database is outside of the Siebel architecture. Communications to the database are handled by the Siebel Object Layer.

2.1.8 Siebel Object architecture

The Siebel eBusiness architecture consists of four layers of objects plus the relational database. Objects depend upon the objects that are defined in the layers below. They are insulated from each other in that changes to objects in one layer require little or no changes to objects at lower layers.

The Siebel Object-Based Architecture is depicted in Figure 2-2.

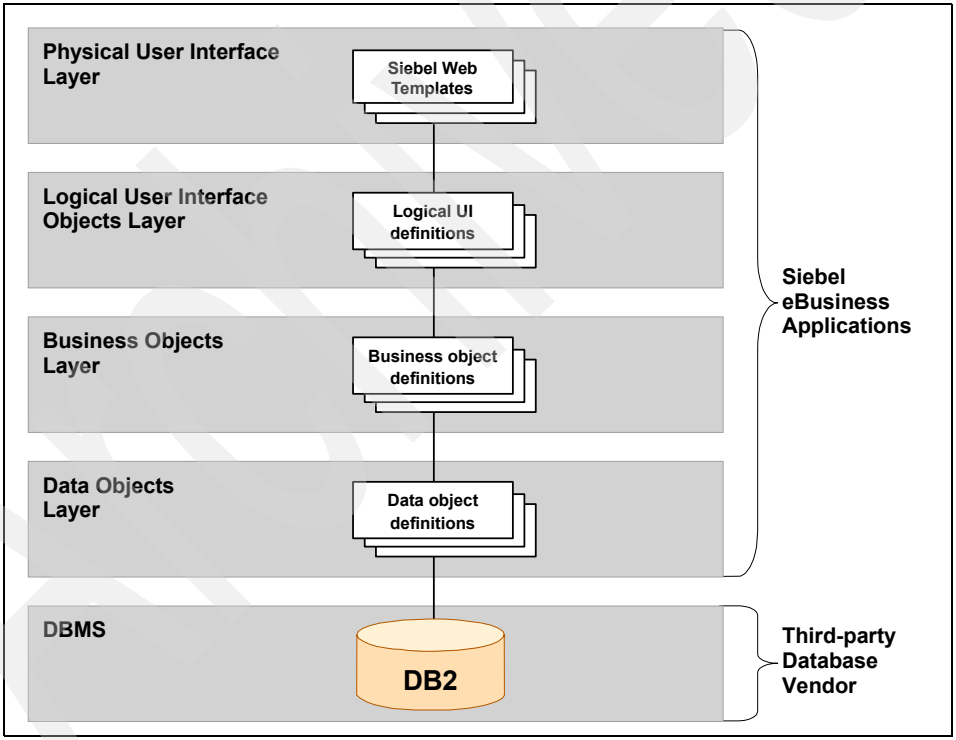

Figure 2-2 Siebel Object Architecture[2]

The definitions of the layers are as follows:

[2] Siebel Systems, Reprinted by Permission.

Physical user interface layer

This layer contains the Siebel Web template files that control the physical style and structure of the user interface. The user interface is interactive and customizable. The physical user interface layer includes template files that control the style and structure of the user interface. Siebel tags are embedded in the HTLML template files to serve as placeholders for objects defined in the Repository such as controls and applets.

Logical user interface

This layer contains the object definitions which are the visual representation of the objects in the Business objects layer. These define the interface presented to the user and allow the user to manipulate data.

Business objects layer

Individual business entities are defined in this layer. Business entities include Accounts, Contacts, Activities, and so forth. These objects are based on data object definitions.

Data objects layer

These are the logical representation of underlying physical databases. Data objects such as tables, columns, and indexes describe the physical database while remaining independent of the relational database management system (RDBMS) being used.

Database management system

The database management system (DBMS) manages the Data objects layer. The DBMS is not part of the Siebel eBusiness Application.

For further information about the Siebel Object Architecture, refer to *Configuring Siebel eBusiness Applications* and *Developing and Deploying Siebel eBusiness Applications* which can be found in Siebel Bookshelf.

2.2 DB2 UDB architecture

This section provides a high level overview of DB2 UDB V8.2 data storage, processes, memory structures, and concepts.

2.2.1 Architecture overview

Figure 2-3 illustrates the processes and memory architecture within DB2 UDB and their relationships to one another. We discuss the memory components in the next section.

Local and remote clients are linked to the DB2 UDB client library. In a Siebel implementation, most connections to DB2 UDB are through the various Siebel servers via interactive Application Object Manager (OM) sessions. TCP/IP and APPC are included in the supported communications protocols.

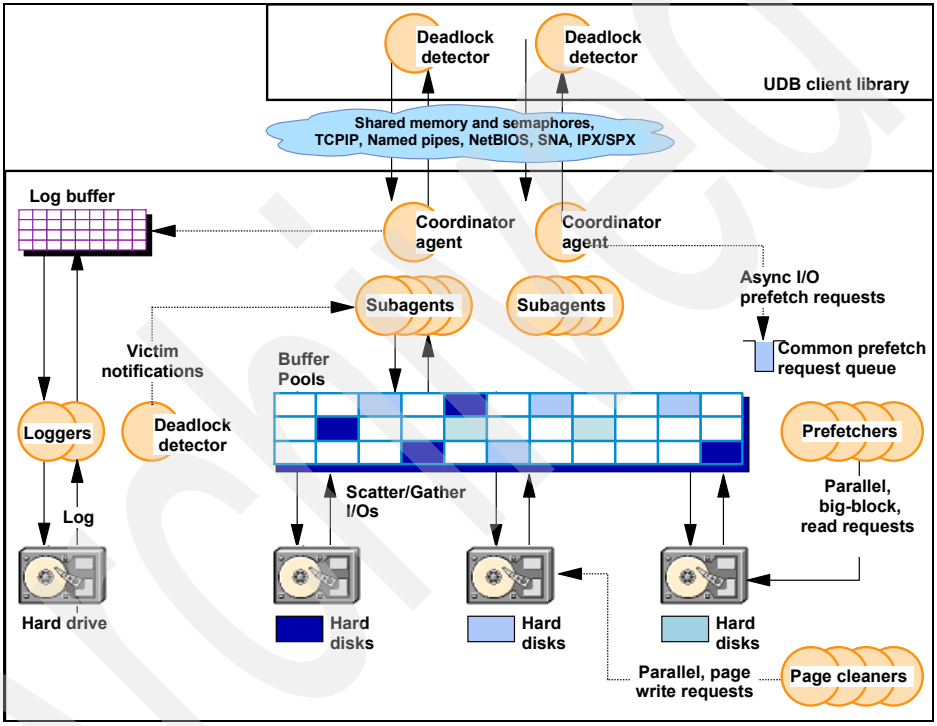

Figure 2-3 DB2 UDB process model

Client application

Client applications may connect remotely or locally. Client applications initially connect to DB2 UDB using one of three listeners:

- *db2tcpcm* is the listener used for client connections using TCP/IP.
- *db2snacm* is the listener used for client connections using APPC.
- *db2ipccm* is the listener used for local client connections.

The DB2 UDB client software is installed on the Siebel servers and is not required on the Siebel Web clients.

Engine dispatchable units

The circles or sets of circles on the DB2 UDB server diagram shown in Figure 2-3 on page 15 represent engine dispatchable units or EDUs. On Windows platforms, EDUs are threads within a user process. On UNIX® platforms, EDUs are processes. The listeners that the client application uses to connect to DB2 UDB are specialized EDU processes.

db2agents are by far the most common type of EDU. db2agents perform most of the SQL work required by applications. Other EDU types include loggers and deadlock detectors.

The connection with the application is managed by the Coordinator agent. Intra-parallelism is disabled by default and should be disabled for Siebel OLTP processing. There are no subagents when intra-parallelism is disabled.

2.2.2 Process concepts

DB2 UDB uses a variety of processes to manage tasks that must be used to manage and support database tasks efficiently while providing recovery from system problems. Figure 2-4 on page 17 illustrates the DB2 UDB process model.

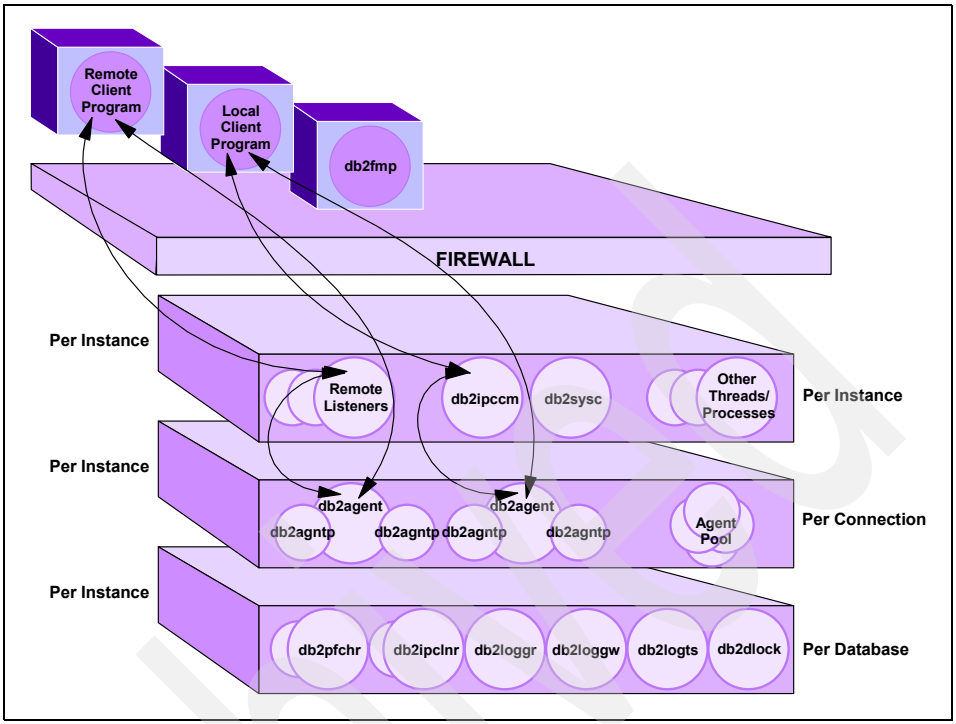

Figure 2-4 DB2 UDB process model

When an application connects to the database, a coordinator agent is assigned to the application. The coordinator agent handles communications with other agents on behalf of the application.

UNIX environments are architected based on system processes. Intel® based servers use a thread-based architecture to maximize performance. This section uses the term “process” to apply to both UNIX processes and Intel threads.

The major components of the process model are as follows:

DB2 UDB firewall

The DB2 UDB firewall has nothing to do with firewalls that prevent customers from external attacks. The DB2 UDB firewall protects internal database buffers and files from being corrupted or overwritten by user defined functions or stored procedures. The firewall also prevents application errors from crashing the database manager. Behind the firewall, the fenced mode process, called *db2fmp*, executes the fenced stored procedures and user defined functions in a separate address space behind the firewall.

Listener processes

Client application programs run remotely or locally and make their first contact to the database via the listener processes. There are four listener processes, one for each configured communication protocol.

Agents

All connection requests are assigned a coordinator agent (db2agent). When the connection occurs, an agent may be created or an idle agent may be assigned from the agent pool. The coordinator agent performs all database requests on behalf of the client application.

For intra-parallel enabled environments running OLAP type applications, subagents, *db2agntp*, will be assigned portions of the work effort by the coordinator agent.

Database server threads and processes

At the instance level, there are several processes. The system-controlled process, *db2sysc*, runs on all platforms. `db2sysc` process must exist for the database server to function.

Aside from `db2sysc`, there are three server processes common to all servers:

- *db2resyn* which scans the global resync list.
- *db2cart* is responsible for archiving log files when `USEREXIT` has been enabled.
- *db2fmtlg* formats log files and is used when `LOGRETAIN` is enabled but `USEREXIT` is not enabled.

There are two database server processes that are specific to UNIX-based systems:

- *db2gds* is responsible for spawning new processes.
- *db2wdog* handles abnormal terminations.

Partitioned database environments have a specific set of processes:

- db2fcmdm is the fast communications manager daemon for inter-partition communication.
- *db2pdbc* is the parallel system controller used to handle parallel requests from remote database partitions.
- *db2panic* handles urgent requests after agent limits have been reached on a database partition.

Database threads and processes

Each database within an instance has several threads and processes. The most important ones are listed here:

- *db2pfchr* is the buffer pool prefetch process. This handles bringing groups of sequential pages into memory when prefetch occurs.
- *db2plcnr* is the buffer pool page cleaner. This key process writes dirty pages to disk asynchronously to prevent delays in bringing data pages into memory.
- *db2loggr* manipulates log files for transaction processing and recovery from failures.
- *db2loggw* writes records to log files. This process maintains integrity in an important way. If a hardware failure occurs and the system goes down, all updates in memory would be lost; except, db2loggw keeps the logs up to date and available for crash recovery to bring the system back to a consistent state.
- *db2logts* collects historical information about which logs are active when a table space is modified. It is used to speed up table space rollforward recovery. This type of recovery would be unusual in a Siebel implementation due to the needs of programmatic referential integrity. In general, all table spaces containing transactional data must consistent.

db2lock is used for deadlock detection. In partitioned database environments, *db2glock* is used to coordinate the `db2lock` process running on each database partition.

2.2.3 Memory concepts

The amount of memory used by applications varies by the settings for database manager configuration parameters, database configuration parameters, size of memory structures, and the number of concurrent connections to the system. It is important that the memory allocated does not exceed the real memory available on the database server. Paging or swapping to virtual memory is a very expensive process that will greatly reduce the scalability and performance of your implementation.

Figure 2-5 on page 20 illustrates the memory structure of DB2 UDB V8.2. Some of these memory structures are defined as ceilings and can grow incrementally. For example, `LOCKLIST` can expand and contract depending upon processing requirements. This characteristic can create the impression that there is no risk in setting these to very high levels. This is not the case. If at maximum size, the sum memory allocation of these heaps plus private memory and other memory overhead reaches the limit of real memory, paging or swapping to virtual memory will occur.

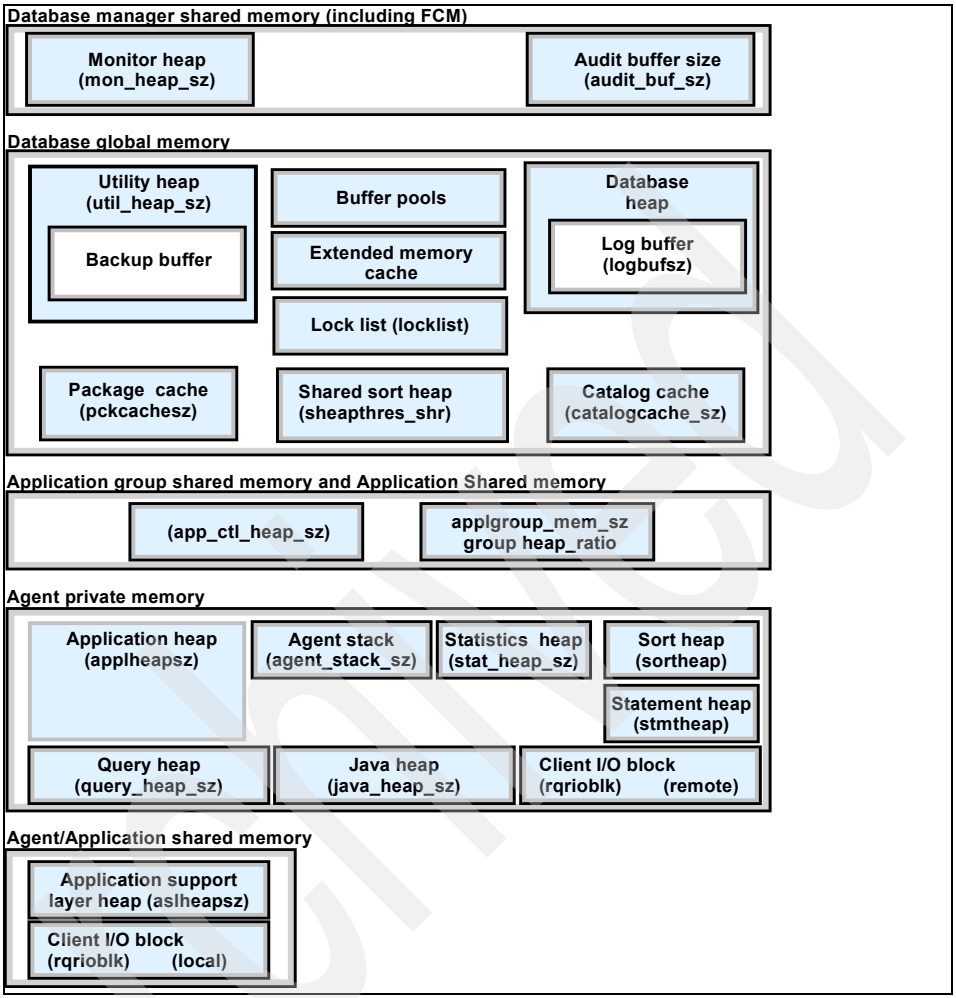

Figure 2-5 DB2 UDB memory model

Figure 2-5 shows memory being allocated at five different levels. Each level of memory is allocated at different points in time as follows:

- *Database manager shared memory* is associated with the DB2 instance. It is allocated when the `db2start` command is issued and deallocated when the `db2stop` command is executed.
- *Database global memory* is associated with a given database within an instance. This memory is allocated either when the database is activated or when the first connection is made to the database.

- *Application group and application shared memory* are only allocated when either intra-parallel is enabled or the DB2 connection concentrator is used.
- *Agent private memory* is allocated to an agent when work is assigned to the agent.
- *Agent/application shared* memory is used for local applications.

For 64-bit implementations, the amount of memory that DB2 UDB can address is dramatically increased. Physical limitations must be kept in mind when deploying very large buffer pools in 64-bit environments. Note that `DBHEAP` may have to be increased as well. There is a 100-byte descriptor added to the database heap for each page of buffer pool memory. A 10 GB buffer pool would require 250 MB of space in the database heap.

Memory structure definition

In this section, we discuss several of the more important memory structures for Siebel applications in a Siebel/DB2 UDB environment.

Buffer pools

These are the memory structures into which data is retrieved, entered, or manipulated. Sizing the buffer pools appropriately to your environment is critical to the success of any implementation. If the buffer pools are sized too small, additional I/O occurs because data pages which could have been reused will be flushed to disk to make room for pages containing modified or new data.

In DB2 UDB V8.2, buffer pools can be created or altered dynamically. This provides great flexibility in adjusting to fluctuating or unexpected workloads.

Database heap

This heap reserves a minimum amount of space for event monitor buffers, utility temporary memory, the log buffers, and overflow from package or catalog caches.

Utility heap

The utility heap defines the maximum amount of memory available for utilities such as backup.

Locklist

This important memory structure stores all the locks held at any given moment by connected applications. If this memory structure becomes full or if one connection consumes more than the MAXLOCKS percent of LOCKLIST, a lock escalation will be attempted by the next application requesting a lock. In DB2 UDB, the next level of locking is a lock on the entire table. Under normal OLTP circumstances, lock escalations are to be avoided. There can be exceptions for

some batch processing because it can be more efficient to hold one table lock than to manage numerous row locks.

Catalog cache

This is a soft limit on the amount of memory allocated to cache the contents of the DB2 UDB system catalog. The catalog cache can overflow to other memory structures such as `DBHEAP`. In practice, database administrators should try to detect and prevent such overflows to avoid contention for memory resources.

Package cache

The package cache contains the descriptors for dynamic and static SQL. This is also a soft limit and can overflow, which should be avoided. Siebel SQL is all dynamically generated.

Shared sort memory

This is a hard limit on the amount of memory that any one thread can use. This should only be used when DB2 connection concentrator is enabled or `INTRA_PARALLEL` is set. `INTRA_PARALLEL` should not be used for Siebel OLTP environments. Siebel recommends using Siebel connection concentration, if this this feature is required.

Siebel applications use private sorts for OLTP implementations because `INTRA_PARALLEL` is not enabled. Siebel data warehousing implementations on partitioned databases could enable `INTRA_PARALLEL` and thereby have shared sorts. If a shared sort exceeds the limit set in the `SHEAPTHRES` parameter, an SQL0955C error message will be issued to applications requiring sort resources until the total sort memory in use falls to less than `SHEAPTHRES`.

SORTHEAP

`SORTHEAP` defines the amount of memory pages available for private or shared sorts. For private sorts, if this limit is exceeded, the database manager snapshot will record post-threshold sorts.

For private sorts, sort space will be allocated to the connection. When defining memory structures, it is very important to leave memory for user connections to prevent swapping or paging from occurring.

Statement heap

This memory space is used for the compilation of SQL. Errors can occur if this area is sized too small.

2.2.4 Storage concepts

A high level model of the DB2 UDB logical storage model is shown in Figure 2-6. All table spaces, tables, indexes, catalogs, objects, and database partition groups form an entity called a *database*. Each database has its own physical control, log, and data files.

Each database resides within a DB2 instance. An instance can contain more than one database. Some memory structures are defined at the instance level.

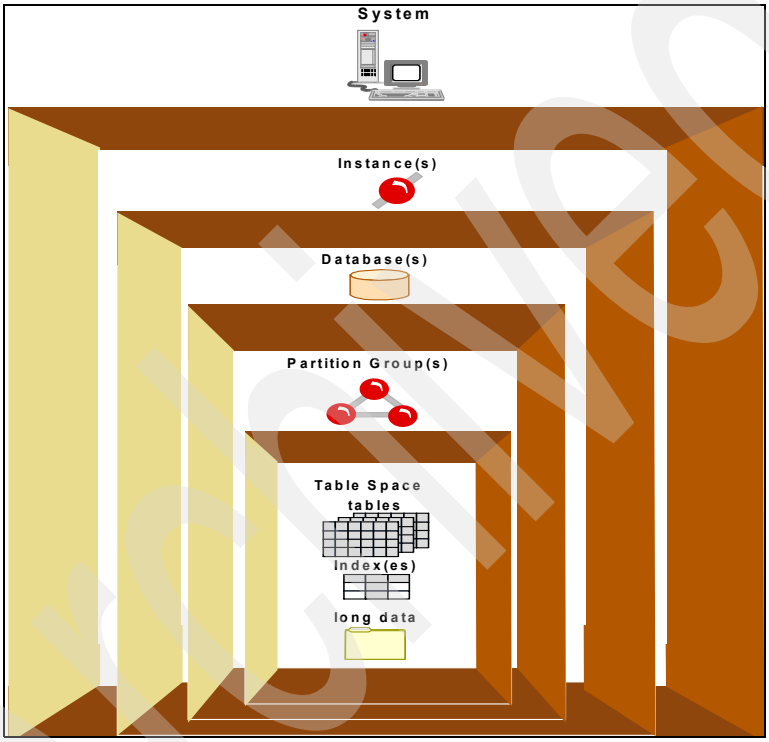

Figure 2-6 DB2 UDB logical storage model

The table space is the logical storage construct for table and index data. There are two types of table spaces:

- System managed space (SMS): Data is stored in system directories and managed by the operating system.
- Database managed space (DMS): Data is stored in file system files or on raw devices and managed by the database. DMS storage is a pre-defined size and increased or decreased as necessary.

For Siebel applications, Siebel recommends DMS table spaces for all user data out of the box. Siebel separates indexes and large objects into separate table spaces to enhance I/O performance. Typically, SMS table spaces are used for the DB2 UDB system catalog and temporary storage.

Physical disk space is allocated to a table space in *containers*. For DMS table spaces, containers can be added as needed. Existing containers can be expanded, contracted, or removed if necessary. For SMS table spaces, containers must be created when the table space is created. SMS container size can be modified by increasing or decreasing the space available in the file system or file systems to which the SMS container was mapped.

For Siebel Data Warehousing products, database partition groups can be used. These were called *nodegroups* in earlier releases of DB2 UDB. A database partition group is a set of DB2 partitions. Multiple database partitions can be defined on one or more physical servers. Database partitions are not supported for Siebel OLTP.

3

Planning considerations

This chapter gives you a brief overview of installation and planning considerations.

This chapter discusses the following items:

- Siebel release matrix for DB2 UDB
- Siebel sizing considerations
- Hardware and storage requirements
- Database storage and layout

This chapter also describes the users and groups required for Siebel installation, and, in addition, it provides a Siebel installation checklist.

3.1 Siebel release matrix for DB2 UDB

This section provides the software and hardware requirements for DB2 UDB in a Siebel environment.

3.1.1 IBM DB2 Universal Database™ V8.2 ESE

The IBM DB2 Universal Database (UDB) software required for DB2 UDB database server are:

- DB2 UDB database uses Unicode and UTF-8 code set.
- Refer to Siebel system requirements and supported platforms in Siebel Bookshelf for the supported DB2 UDB FixPak. The FixPak software can be found at:

  `ftp://ftp.software.ibm.com/ps/products/db2/fixes/english-us/siebel/siebel7/DB2V8/`
- DB2 UDB V8.2 ESE is supported as a single partition instance (one node) only.
- The DB2 UDB V8.2 ESE supports both 32-bit and 64-bit. At the present time, Siebel Server can work only with 32-bit DB2 UDB clients. You can have databases under a 64-bit instance. The connection will be via a 32-bit instance. This is demonstrated in the lab example.

3.1.2 Operating systems and hardware for DB2 UDB

DB2 UDB V8.2 requirements for DB2 UDB database server are given below:

- DB2 UDB V8.2 on AIX pSeries® systems based on POWER™ 3, POWER 4, and POWER 5 processors are supported. AIX 5.2 (32 bit and 64 bit) ML 6 with C++ runtime level is `XlC.rte 6.0.0.0` and `xlC.aix50.rte.6.0.0.5.`
- Minimum hardware for DB2 UDB V8.2 for Windows x86 requires Intel Pentium® 4 or higher, Intel Xeon™, or AMD Athlon based systems; x86-64 requires Intel EM64T or AMD64 based systems; IA64 requires Intel Itanium® based systems. Windows Server 2003 Standard Edition (32-bit and 64-bit), Enterprise Edition (32- bit and 64-bit), and Datacenter Edition (32-bit and 64-bit) ares supported. Windows 2000 Standard, Advanced, and Datacenter Editions with Service Pack 4 or later are also supported.
- Minimum hardware for DB2 UDB V8.2 for HP-UX requires either PA-RISC CPU or Itanium CPU. HP-UX 11i RISC (32-bit and 64-bit) GOLDBASE11i bundle is required for PA-RISC CPU. HP-UX 11i I86 with patches. `PHKL_30065`, `PHSS_30230`, and `PHSS_31086` are required for Itanium CPU.

- Sun™ Solaris™ 9 (32-bit and 64-bit) is required. For Sun Solaris, CPUs prior to UltraSparc are not supported.

3.1.3 Siebel Developer Web Client

Siebel Developer Web Client is required for developers and System Administrators. DB2 UDB Runtime client is required for Developer Web Client along with Siebel client software. The DB2 UDB Runtime client requirements are:

- IBM DB2 Runtime Client V8.1, minimum FixPak 8s
- Siebel specific FixPaks are found at:

  ftp://ftp.software.ibm.com/ps/products/db2/fixes/english-us/siebel/siebel7/DB2V8/

3.2 Siebel sizing considerations

Use sizing reviews to estimate the resource requirements of a given implementation. The sizing review provides guidelines for resources required for all tiers of the implementation. Siebel Expert Services conducts sizing reviews in close cooperation with the hardware vendors with whom the customer chooses to work.

In this section, we discuss the following:

- Information requirements
- Related optional reviews

3.2.1 Information requirements

The customer's Siebel Technical Account Manager will provide a Sizing Questionnaire to guide information gathering. Use this to consolidate information for delivery to Siebel Expert Services. To estimate hardware requirements as accurately as possible, gather detailed information in the following areas:

- User characteristics
  - Number of initial users.
  - Rate at which users are expected to be added, if applicable. Some projects are rolled out in phases.
  - Estimated think time of user activity. Scalability testing is conducted assuming a 30-second think time. If think times are expected to be higher or lower, the reviewer must be informed in order to adjust the resource estimates accordingly.

  - Numbers of users for each application object manager. For example, the customer's given implementation may plan on 2,000 Call Center users with 500 concurrent and 200 Remote users with 20 synchronizing at one time. The sizings for OM and Remote servers would reflect the user estimates for each server.
- Data storage
  - Identify which Siebel objects will be used, such as Accounts, Opportunities, and Contacts.
  - Estimate initial data load volumes, if any. Often existing data is converted into the Siebel architecture.
  - Optionally estimate row lengths of key tables. Siebel Expert Services has row length averages from previous customer implementations that you can use if estimates are not available.
  - Estimate ongoing insert, update, and delete activity from the user interface and from externally integrated systems.
  - If archival activity is planned, resulting reductions in storage requirements will be taken into account.
  - The sizing review does not include storage requirements for database backups, temporary storage, or log space requirements. These vary considerably between implementations and cannot be accurately estimated in advance.
- Component configuration
  - Identify the components that will be deployed: Siebel Remote, Enterprise Application Integration, Enterprise Integration Manager, Actuate Reporting, Object Managers, and so on.
  - Estimate the level of activity for each component, such as the number of users, rates of activity, and so forth.
  - If High Availability (HA) is required, the hardware requirements will be included in the estimates.
  - The sample topology will show which components can be placed together on which servers. This is especially important for HA environments since some components have specific HA support requirements.

3.2.2 Sizing deliverables

Sizing reviews will provide estimates for memory, CPU, and disk requirements for all tiers. The reviews provide initial and future disk estimates for OLTP and/or Analytic databases. Hardware requirements for all servers will be estimated, and

information on backup and recovery is provided along with network requirements.

3.2.3 Related reviews

There are two optional reviews related to the sizing review that can be performed prior to the sizing review or another time. Both optional reviews provide information on options available for deployment to help guide the customer to the most appropriate architectural design.

Architecture design workshop

The architecture design workshop is complementary to other Expert Services reviews, including the sizing review. The workshop discusses high level hardware configuration, backup and recovery, database and server performance optimization, interface strategies and design, assignment manager design, workflow manager design, deployment strategies, and roll-out options.

Customized design workshops can be delivered to accommodate specialized implementation considerations.

High-availability workshop

This technical workshop outlines available high-availability solutions and how they can be applied to high-availability requirements for an implementation. This workshop is a complement to the sizing review.

This workshop can be customized for requirements of the planned implementation.

3.3 Hardware and storage requirements

This section gives the hardware and storage requirements for Siebel Servers and DB2 UDB database server.

3.3.1 Siebel hardware and storage requirements

The minimum hardware and storage requirements for Siebel Servers are given below:

Minimum hardware requirements

The minimum hardware requirements for Siebel Enterprise is given below:

AIX 5.2 operating system

Table 3-1 shows AIX 5.2 minimum hardware requirements for Siebel Servers.

Table 3-1 Siebel Enterprise hardware requirements

Siebel Server environment	Hardware requirement
Siebel Gateway Server	IBM pSeries Server @ 500 MHz with 256 MB memory
Siebel Server	IBM pSeries Server @ 500 MHz with 1 GB memory
Siebel Web Server	IBM pSeries Server @ 500 MHz with 512 MB memory
Database Server	Data and System Architects working with Siebel Expert Services will provide it.

Windows operating system

Table 3-2 shows the minimum hardware requirements for Siebel Servers on Windows operating systems.

Table 3-2 Siebel Enterprise hardware requirements for Windows OS

Siebel Server environment	Hardware requirements
Siebel Gateway Name Server	PIII XEON @ 500 MHz with 256 MB memory
Siebel Server	PIII XEON with two processors @ 500 MHZ with 1 GB memory
Siebel Web Server	PIII XEON @ 500 MHZ with 512 MB Memory
Database Server	Data and System Architects working with Siebel Expert Services will provide it.

Siebel Developer Web Client

Table 3-3 shows the minimum hardware requirements for Siebel Developer Web Client.

Table 3-3 Hardware requirements for Siebel Developer Web Client

Hardware components	Hardware requirements
CPU	CPU – PII 500 MHz or Celeron® 800 MHz
Storage	650 MB

Memory	Windows 2000 Professional with Service Pack 4 or above with 512 MB Windows XP Professional with Service Pack 1 or above with 512 MB

Siebel storage requirements

Siebel installation code is either downloaded from:

`ftp://ftp.siebel.com`

or code is provided in DVD disks. If the code is downloaded from the FTP Web site, the download requires a large amount of space. In our lab implementation, we downloaded the following for Siebel Installation on AIX, and it required about 1.5 GB of space.

- AIX_ImageCreator
- SIA_7.8.2.0_Base_AIX_Siebel_Enterprise_Server.jar
- SIA_7.8.2.0_Base_AIX_Siebel_Web_Server_Extension.jar
- SIA_7.8.2.0_enu_AIX_Siebel_Enterprise_Server.jar
- SIA_7.8.2.0_enu_AIX_Siebel_Web_Server_Extension.jar
- imagecreator.jar
- media.inf
- siebel.ini

AIX Storage Requirements

The default installation location of Siebel on AIX is `/siebel`. We recommend that the installation of Siebel in a production environment should be done on a separate file system.

The size of the typical base install of a Siebel Gateway Name server is about 40 MB. If the gateway is a standalone server, then we recommend a minimum of 2 GB of space for production environment.

The size of the typical base install of a Siebel Server is about 1 GB. A production environment requires a minimum of 6 GB. If the Siebel database server is also installed on the Siebel application server, then a typical base install requires about 2.2 GB. In a production environment, we recommend a minimum of 8 GB of space for Siebel Server with Siebel database server installed on it.

The size of the typical Siebel Web Server Extension (SWSE) is about 130 MB. We recommend a minimum of 1 GB for SWSE in a production environment.

Windows storage requirements

The default installation path of Siebel is `c:\sea78`.

- The size of a typical Siebel Gateway Name server installation on Windows is about 30 MB. The size of a typical Siebel Application Server installation on Windows is about 550 MB. The size of a typical Siebel database server installation on Windows is about 1.6 GB. We recommend a minimum of 8 GB of space for Siebel implementation on Windows for a production enterprise.

3.3.2 Considerations for hardware planning

Ordering the proper hardware is key to a successful implementation. Consider the following input when you plan your hardware purchase. Order hardware that is scalable and has proven high performance.

- Consider buying a higher quantity of smaller size drives instead of a few larger size drives in order to spread your database across a large number of disks. For example, If you need 600 GB of storage, we recommend that you order nine 70 GB drives instead of two 300 GB drives.
- Recommend having four to eight drives per CPU. This way you can spread the table spaces across a number of drives and achieve parallel I/O.
- Advise you have multiple I/O adapters to the disk drives whether they are internal (SSA) or external (enterprise storage). Recommend about one adapter per every four CPUs (Minimum of two adapters in case you have a two or four CPU machine).
- Recommend using 70 - 80% of the memory in the machine for OLTP database and about 50 - 60% for OLAP database.
- Suggest you have 4-8 GB of memory per CPU.
- Separate the communication network among the database server, application servers, and general communications (public network). Also, we suggest keeping the ADSM/backup manager communications separate.

3.3.3 DB2 UDB

This section describes the storage requirements of DB2 UDB database.

- DB2 UDB on AIX:

  A typical DB2 UDB install on AIX requires about 1.3 GB in `/usr`. DB2 UDB is installed in `/usr/opt` directory. We recommend the DB2 instance home directory have at least 2 GB of storage. If `/home/ucsdb2/sqllib/db2dump` is moved to another filesystem, then 1GB is enough for the DB2 instance home directory. The amount of space required for an empty database is about 35 MB. A minimum of 12 GB of space is required to load the Siebel seed and repository data. The sizing from the Siebel Expert Service will determine the final database size.

- DB2 UDB on Windows:

  A typical DB2 UDB install on Windows requires about 350-560 MB of space. We recommend a minimum of 2 GB space for db2dump. A minimum of 12 GB is required to load the Siebel seed and repository data. The sizing from the Siebel Expert Service will determine the final database size.

  The amount of space required for an empty database is 35 MB. The sizing from the Siebel Expert Service will determine the final database size.

- DB2 UDB logs:

  We recommend a database log file size of 32000 4K pages, 20 primary log files, and 100 secondary log files.

3.4 Database layout and recommendations

In this section, we discuss the recommended database layout for a Siebel CRM (OLTP) database. The layout here also will be used in Chapter 8, “Performance tuning” on page 199 and carried through Chapter 9, “High availability and disaster recovery” on page 245.

3.4.1 Decision time

Before starting database layout planning, gather background information for your goals. The following list will drive you toward what will be a good database layout.

- Should the catalog table space be SMS or DMS?
  - What kind of storage do you have?

    Having a fast storage (where multiple I/O adapters and fast drives are available) can mean that you can go with SMS.
  - Will you use flashcopy (function on the IBM Enterprise Storage Server® that can create a point-in-time copy of data while an application is running) or hardware with similar capabilities?

    To do a flashcopy with DB2 UDB, your catalog table space has to be DMS.

  In a Siebel environment, we recommend DMS catalog table space and higher `DBHEAP` so you can cache the whole database catalog in the memory. You can use `DB_HEAP_TOP` to find out what the maximum memory used is.

- What should the split of table and indexes be?

  Siebel out-of-the-box will require 4K, 16K, and 32K page size table spaces for data and an index table space with 4K page size for all indexes. This works fairly well for small to medium size databases.

  For better performance, we recommend splitting out the indexes also.

Table 3-4 shows a simple modification to the database layout. In this table, we created table spaces with names such as `siebel_4k`, `siebel_4ki`, `siebel_16k`, `siebel_16ki`, `siebel_32k`, and `siebel_32ki`.

Table 3-4 Modification to out-of-the-box Siebel database layout

Type	4K	16K	32K
Data	`siebel_4k`	`siebel_16k`	siebel_32k
Index	siebel_4ki	siebel_16ki	siebel_32ki

Note: The index table space page size is always 4k regardless of name.

- Separate table space for long column data.

  Having a separate table space for long column data helps in a couple of ways:

  – Reduces the size of the data table space.
  – Separation of regular data from long data, so during a table scan, you will not pull up the long column data.

- Number of containers for the temporary table space `TEMPSPACE`.

  This should depend on couple of factors:

  – Number of I/O adapters available on the system.
  – Is the space spread across multiple adapters in the storage area?

  Having `TEMPSPACE` across multiple adapters or file systems helps divide the I/O when it hits the storage area.

- Will you use dual logging for DB2 UDB database?

  This is totally dependent on what your high availability and recovery strategy is. For development environment systems, you may not need it. For production systems, we highly recommend dual logging.

Different file systems for DB2 UDB for lab environment

We are using separate file systems so that we can spread out the I/O. A similar approach will also help you in case you need to move the file systems around at a later time to reduce the hot spots.

We used the following layout to test.

- File systems for catalog, temporary, and large (long) table spaces

  `/db2/fs1`, `/db2/fs2`, `/db2/fs3`, and `/db2/fs4` - one file system per adapter

- File system for dual logging

  `/db2/fs5`, `/db2/fs6` - one on external storage area and one on internal drives

- File system for DB2 UDB log archiving

  `/db2logarch` - which is on a separate DASD/RAID from all the DB2 UDB containers (raw) and file systems

- Database backup file system

  `/dbbkup` - which is on a separate DASD/RAID from all the other devices

  Table 3-5 is the list of logical volume names and file systems we planned for the lab environment.

Table 3-5 Logical volume names and file systems used in the lab environment

lvname	Size	Mount point	Usage
ucsdb2lv	2 GB	/home/ucsdb2	ucsdb2 home directory
sebldb2lv	2 GB	/home/sebldb2	sebldb2 home directory
fencdb2	9 MB	/home/fencdb2	fenced id home directory
db2fs1lv	20 GB	/db2/fs1	DB2 file system
db2fs2lv	20 GB	/db2/fs2	DB2 file system
db2fs3lv	20 GB	/db2/fs3	DB2 file system
db2fs4lv	20 GB	/db2/fs4	DB2 file system
db2fs5lv	20 GB	/db2/fs5	DB2 dual logging file system
db2fs6lv	20 GB	/db2/fs6	DB2 dual logging file system
dbbkuplv	400 GB	/dbbkup	File system for Database backups and work area
siebellv	10 GB	/siebel	file system for Siebel software installation
siebelfslv	10 GB	/siebelfs	file system for Siebel attachments
vpath0cont001lv to vpath0cont120lv	2 GB	-	raw devices for DMS table space
aixdbalv	4 GB	/home/aixdba	aixdba home directory

This is the process we followed to create the volume group, logical volumes, and file systems.

- Creating volume group

In our lab, we created the volume group `datavg` on the shared storage area hdisk3 using the below command.

```
mkvg -f -y'datavg' hdisk3
```

- Creating logical volumes

  Create the logical volumes using **`mklv`** command.

  In our lab, we created the following logical volumes in volume group `datavg`:

```
# filesystem dbbkuplv for database backup
mklv -y'dbbkuplv' -t'jfs2' -w'n' datavg 400 hdisk3

# db2 sms filesystem fs1 to fs4
mklv -y'db2fs1lv' -t'jfs' -w'n' datavg 20 hdisk3 db2fs1lv
mklv -y'db2fs2lv' -t'jfs' -w'n' datavg 20 hdisk3 db2fs2lv
mklv -y'db2fs3lv' -t'jfs' -w'n' datavg 20 hdisk3 db2fs3lv
mklv -y'db2fs4lv' -t'jfs' -w'n' datavg 20 hdisk3 db2fs4lv

# db2 dual logging filesystem fs5 and fs6
mklv -y'db2fs5lv' -t'jfs' -w'n' datavg 20 hdisk3 db2fs5lv
mklv -y'db2fs6lv' -t'jfs' -w'n' datavg 20 hdisk3 db2fs6lv

# filesystem ucsdb2lv for ucsdb2 home directory
mklv -y'ucsdb2lv' -t'jfs' -w'n' datavg 2 hdisk3 ucsdb2lv

# filesystem sebldb2lv for sebldb2 home directory
mklv -y'sebldb2lv' -t'jfs' -w'n' datavg 2 hdisk3 sebldb2lv

# filesystem fencdb2lv for fencdb2 home directory
mklv -y'fencdb2lv' -t'jfs' -w'n' datavg 9 hdisk3 fencdb2lv

# filesystem aixdbalv for aixdba home directory
mklv -y'aixdbalv' -t'jfs' -w'n' datavg 4 hdisk3 aixdbalv

# filesystem siebellv for Siebel software installation
mklv -y'siebellv' -t'jfs' -w'n' datavg 10 hdisk3 siebellv

# filesystem siebelfslv for Siebel attachments
mklv -y'siebelfslv' -t'jfs' -w'n' datavg 10 hdisk3 siebelfslv

# raw devices for DMS table spaces vpath0cont001lv to vpath0cont120lv of 2
gb each
mklv -y'vpath0cont001lv' -t'jfs' -w'n' datavg 2 hdisk3 vpath0cont001
```

We had one disk and all of the logical volumes were created on the same disk. We have created logical volumes for database, Siebel install, and backup separately, so we can move them around if necessary. This also helps reduce the I/O contention, when the logical volumes are spread across

several disks. We highly recommend having multiple disks and spreading logical volumes across these disks.

- Creating file systems

  Create the file systems using the **crfs** command. Mount the file systems. Give proper permissions to the DB2 maintenance ID. In our lab environment, the maintenance ID owns all the file systems and raw devices.

  In our lab, we created the file systems using the following script:

```
/usr/sbin/crfs -v jfs -a bf=true -d'db2fs3lv' -m'/db2/fs3' -A''`locale
yesstr | awk -F: '{print $1}'`'' -p'rw' -a options='rbrw' -t''`locale nostr
| awk -F: '{print $1}'`'' -a nbpi='4096' -a ag='64'
```

  Before and after mounting the file systems, the permissions were set to 775 and with owner.group name set to aixdba.aixdbag.

  aixdba is the maintenance ID and aixdbag is the primary group for the maintenance.

```
chmod -R 775 /db2
chown -R aixdba.aixdbag /db2

mount /db2/fs1
mount /db2/fs2
mount /db2/fs3
mount /db2/fs4
mount /db2/fs5
mount /db2/fs6

chmod -R 775 /db2
chown -R aixdba.aixdbag /db2
```

3.4.2 Table grouping by Siebel components used

Tables can get very big as the number of users and the amount of incoming data increase over a period of time. This can also be true if you are moving from a different CRM application.

Sample list of tables that grow over time:

```
S_PARTY, S_CONTACT, S_ORG_EXT, and S_EVT_ACT
```

We recommend that tables that grow quickly over time need to be in a separate table space away from your general data. There are two schools of thought:

- If there are only a handful of tables that grow, are volatile, or are heavily used, create one table space per table for each of these big tables. Always keep the index in a separate table space.

- Group tables by their usage: Create some extra table spaces and move the tables, so that there are more than one large table in these extra table spaces, but their uses are different.

  For example, you can put `S_CONTACT` and `S_ORG_EXT_XM` or `S_ADDR_ORG` in one table space and `S_ORG_EXT` and `S_CONTACT_XM` in a different one.

Always keep `S_PARTY` by itself in yet a different (third) table space.

Tables space for lab environment

On the AIX box, these are the types of table spaces we used.

Table 3-6 provides you with the minimum sizes you need for a blank Siebel database (including Siebel seed data). These numbers are just a guide.

Table 3-6 Minimum database table space sizes with just seed data

Table space	Type (SMS/DMS)	Type (fs,raw)	Size	Page size
CATALOG	DMS	file	100000	4096
TEMP4K	SMS	file system		4096
TEMP16K	SMS	fs(path)		16384
TEMP32K	SMS	fs(path)		32768
SIEBEL_4K	DMS	raw	1000000	4096
SIEBEL_16K	DMS	raw	200000	16384
SIEBEL_32K	DMS	raw	100000	32768
SIEBEL_4KI	DMS	raw	1000000	4096
SIEBEL_16KI	DMS	raw	400000	16384
SIEBEL_32KI	DMS	raw	200000	32768
SIEBEL_4KL	DMS	file	200000	4096

3.5 Using Siebel with DB2 UDB 32-bit versus 64-bit

Siebel supports DB2 UDB environments running 32-bit and 64-bit.

If you have fewer than 4 GB of RAM on the machine, you may not see any difference between DB2 UDB 32-bit and 64-bit. A 32-bit DB2 UDB instance can only address up to 4 GB of memory. However, a 64-bit DB2 UDB instance does not have a limitation. If you have more than 4 GB of memory and are using a 64-bit DB2 UDB instance, be wary of over-allocating memory. A good rule of

thumb is to allocate approximately 60-70% of the total memory on the system. Allocating more than that has shown signs of excessive paging.

The Siebel application itself is currently still in 32-bit. If you plan to have DB2 UDB database on 64-bit instance and Siebel Server on the same machine, you must:

- Run **`export EXTSHM=ON`** before starting a Siebel Server.
- Create a DB2 UDB 32-bit instance on the same server and catalog the database on the 32-bit instance; then, have Siebel initialize and connect via the 32-bit instance.

Depending on the 32-bit or 64-bit DB2 UDB you choose, the Siebel stored procedure `siebproc` is different. Copy `siebproc` (32-bit Siebel stored procedure) or `siebproc64` (64-bit Siebel stored procedure) as `siebproc` from `$SIEBEL_ROOT/dbsrvr/db2udb/siebproc/<OS>` to the `sqllib/function` directory under the instance home directory (AIX) or the DB2 UDB installed directory (Windows). `siebproc` needs to be copied regardless of the operating system.

3.6 Users and groups

In our lab installation, we created two DB2 instances to show different implementations of DB2 UDB and Siebel. We recommend keeping the DB2 UDB instance owner ID and maintenance user IDs separated. We suggest having a fenced user ID, so that stored procedures can run in a separate memory space.

We separated the Siebel software installation user ID from the user ID (*siebsupp*) that connects to the database. *siebsupp* owns all the Siebel files in the Siebel Enterprise Server.

Both DB2 UDB and Siebel user IDs and groups can be created using the UNIX administrative tool or using commands **`mkuser`** and **`mkgroup`**.

User IDs and groups for the lab environment

In our lab, we created four user accounts for DB2 UDB on AIX: Two user accounts: One for 32-bit instance `sebldb2` and one for 64-bit instance `ucsdb2`, `aixdba` for maintenance, and `fencdb2` is the fenced ID for `ucsdb2` instance.

We created four userids for Siebel: `siebel`, `sadmin`, `siebsupp`, and `guest1`.

Refer to Table 3-7 on page 40 and Table 3-8 on page 40 for the groups and users we created on lab systems `KANAGA` and `ATLANTIC`.

To maintain consistency and also to support cluster failover, all the user's uids and group's gids should be identical across servers.

Table 3-7 List of Groups on server KANAGA and ATLANTIC

Group name	ID	Usage
aixdbag	201	Administrator group
ucsdb2g	202	64-bit database owner group
sebldb2g	203	32-bit database owner group
fencdb2g	204	Fenced ID group
sse_role	205	Siebel group
sse_admn	206	Siebel administration/installer group
sse_user	207	Siebel user group

Table 3-8 List of User IDs on servers KANAGA and ATLANTIC

User ID	ID	Primary group	Groups	Home directory	Usage
aixdba	201	aixdbag	aixdbag, ucsdb2g, sebldb2g	/home/aixdba	DB2 maintenance ID
ucsdb2	202	ucsdb2g	ucsdb2g, aixdbag	/home/ucsdb2	64-bit database instance owner
sebldb2	203	sebldb2g	sebldb2g, aixdbag	/home/sebldb2	32-bit database instance owner
fencdb2	204	fencdb2g	fencdb2g	/home/fencdb2	db2 fenced ID for ucsdb2
siebel	205	sse_role	sse_role, sse_admn	/home/siebel	siebel ID
sadmin	206	sse_role	sse_role, sse_admn	/home/sadmin	Siebel administrator ID
siebsupp	207	sse_admn	sse_role, sse_admn	/home/siebsupp	Siebel software install ID
guest1	208	sse_user	sse_user	/home/guest1	Siebel anonymous user

In our lab, we created AIX groups and users using the commands shown in Example 3-1:

Example 3-1 Userid and group creation in AIX

```
#Create group script
mkgroup -'A' aixdbag

#Create user script

mkuser pgrp='aixdbag' groups='aixdbag,ucsdb2g,sebldb2g,' home='/home/aixdba'
shell='/usr/bin/ksh' gecos='aixdba id' registry='files' fsize='-1' data='-1'
aixdba
```

For Windows, create the groups and users using **Administrative Tools** → **Computer Management** → **Local Users and Groups.**

More information about user account, refer to Siebel documentation:

- *Siebel Installation Guide for UNIX: Servers, Mobile Web Clients, Tools*
- *Siebel Installation Guide for Microsoft Windows: Servers, Mobile Web Clients, Tools for more information on Siebel Service Owner Accounts*

3.7 Checklist

The following checklists form the basis for a standard software environment of Siebel 7.8.2 on DB2 UDB V8.2.

Refer to *System Requirements and Supported Platforms* on Siebel support Web site for more information.

- **Operating system**

  Make sure you have adequate disk space and the correct version of the operating system, including the patches/service pack.

  – To find the version and maintenance level in AIX, type the command:

  ```
  oslevel -r
  ```

  ```
  An example of the above command is as follows:
  # oslevel -r
  5200-04.
  ```

  – To check the file set C++ version, execute command:

  ```
  lslpp -l | grep xlC
  ```

  An example of the above command is as follows:

```
xlC.aix50.rte            6.0.0.13  COMMITTED  C Set ++ Runtime for AIX
5.0
```

– To check the version of JDK™ installed, execute command,

`lslpp -l | grep Java™`

An example of the above command is as follows:

```
Java14_64.sdk             1.4.1.3  COMMITTED  Java SDK 64-bit
```

For Windows, right click on the **My computer** icon and select **properties**.

The Windows version and service pack details are displayed.

- **Web server**

  Make sure you have the correct version of Web server installed. In AIX, change directory to `/usr/IBMIHS` and type command:

  `cat version.signature`

  An example of the output is as follows:

  ```
  IBM HTTP Server 2.0.47.
  ```

  In Windows 2003, Internet Information Services (IIS) will be installed by default.

  To check if the Web server is running, in AIX, type the command:

  `ps -ef | grep httpd`

  In Windows, launch the Web site:

  `http://<myserver:port no.>`

  Specify the port number if the server is not running on the default port.

- **Network connectivity**

  Verify the network connectivity among all the machines, using the **`ping`** utility.

- **Siebel File System**

  Create a shared directory that is network accessible to all Siebel servers in the enterprise.

  In AIX, we created a Siebel File System `/siebelfs` of 2 GB on server `KANAGA and` mounted it on client (`ATLANTIC`) using the following command:

  `Mount KANAGA:/siebelfs /siebelfs`

  In Windows, we created a directory `C:\siebelfs` and shared the folder.

- **Storage**

  For AIX, make sure there is a minimum of 200 MB space in `/tmp` directory.

- **Users and Groups**

  Make sure that both the DB2 user IDs and Siebel Enterprise owner user IDs are created.

- **Deployment checklist**

  Make sure the details of the followings are written down before the installation:

  – Server names and installation paths
  – Port and ODBC information
  – Database information

Deployment plan for lab environment

In our lab, we have AIX and Windows environments.

Deployment plan for AIX

In our lab, we had the below deployment plan for Siebel DB2 on AIX:

- Server names and installation path

  In our lab, the Siebel Enterprise Server name is `siebaix`.

  Table 3-9 shows the server names on `ATLANTIC`.

  Table 3-9 Server names on ATLANTIC

Component name	Name	Installation directory
Siebel Gateway Name Server	atlantic	/siebel/gtwysrvr
Siebel Server	atlantic	/siebel/siebsrvr
Siebel File System	siebelfs	/siebelfs

  Table 3-10 shows the server names on `KANAGA`.

  Table 3-10 Server names on KANAGA

Component name	Name	Installation directory
Siebel Database Server	kanaga	/siebel/dbsrvr
Siebel Server	kanaga	/siebel/siebsrvr
Siebel File System	siebelfs	/siebelfs

- Port and ODBC information

Table 3-11 shows the Gateway Port, Siebel Connection Broker port (SCB) and ODBC data source information. In our lab, we used default values for gateway port and SCB port.

Table 3-11 List the Port and ODBC information

Item	Value
Gateway port	2320
Siebel Connection Broker (SCB) port	2321
ODBC data source	siebsrvr_siebaix

- Database information

  Table 3-12 shows the database information.

Table 3-12 Lists database information

Item	Value
Database Server Host Name	kanaga
Database Name	vol78rba
Table owner/password	siebel/******
Database owner/password	sadmin/******
Siebel Index table space	SIEBEL_4KI
Siebel 4-KB table space	SIEBEL_4K
Siebel 16-KB table space	SIEBEL_16k
Siebek 32-KB table space	SIEBEL_32K

Deployment plan for Windows

Deployment plan for Siebel/DB2 on Windows is as follows:

- Server names and installation path

  In our lab, the Siebel Enterprise Server name is `siebwin`. Table 3-13 shows the server names on `PUGET`.

Table 3-13 Server Names on `PUGET`

Component name	Name	Installation directory
Siebel Server	puget	C:\sea782\siebel\siebsrvr
Siebel Database Server	puget	C:\sea782\siebel\dbsrvr

- Port and ODBC information

  Table 3-14 shows the Siebel Gateway Name Server Port, Siebel Connection Broker port and ODBC data source information. In our lab, we used default values for Siebel Gateway Name Server and SCB port.

  Table 3-14 List the Port and ODBC information

Item	Value
Gateway Port	2320
Siebel Connection Broker (SCB) port	2321
ODBC data source	siebsrvr_siebwin

- Database information

  Table 3-15 shows the database information.

  Table 3-15 Lists database information

Item	Value
Database Server Host Name	puget
Database name	siebeldb
Table owner/password	siebel
Database owner/password	sadmin
Siebel Index table space	SIEBEL_4KI
Siebel 4-KB table space	SIEBEL_4K
Siebel 16-KB table space	SIEBEL_16k
Siebek 32-KB table space	SIEBEL_32K

4

Installation and configuration

In this chapter, we explain:

- Installation and configuration of DB2 UDB on AIX, Linux, and Windows
- Installation of Siebel Gateway Name Server on AIX and Windows
- Installation of Siebel Application Server on AIX and Windows
- Installation of Siebel Web Server Extension on AIX and Windows
- Migration of DB2 UDB Database from V7 to V8

4.1 Installation flow chart

The sequence of steps for installation of Siebel Business Application is shown in Figure 4-1.

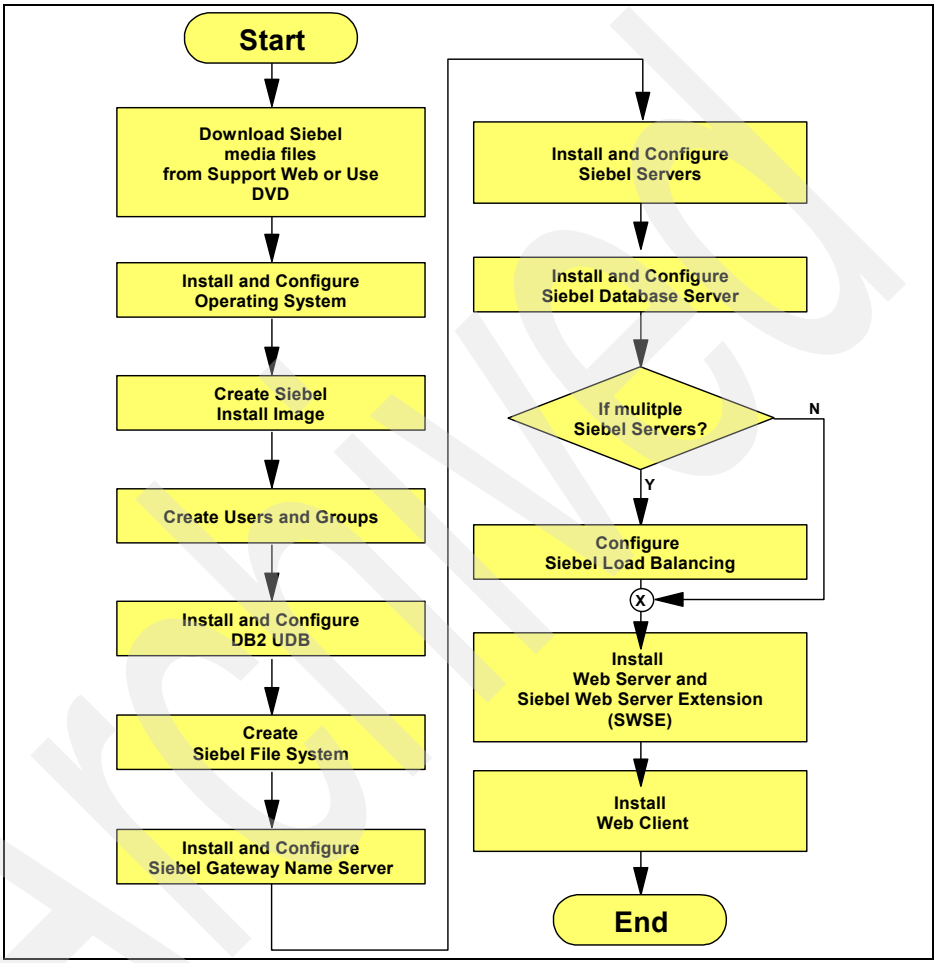

Figure 4-1 Installation sequence for Siebel Business Application

It is imperative that before installing Siebel application components and DB2 UDB, you considered and plan the operating system and DB2 UDB specific configuration parameters based on your environment. These considerations can play an important role in the performance and scalability of the application.

Please refer to Chapter 8, “Performance tuning” on page 199, for details about configuration considerations.

4.2 Lab environment

To demonstrate the installation, configuration, and failover, we have set up two Siebel environments.

AIX environment

Figure 4-2 illustrates the AIX environment.

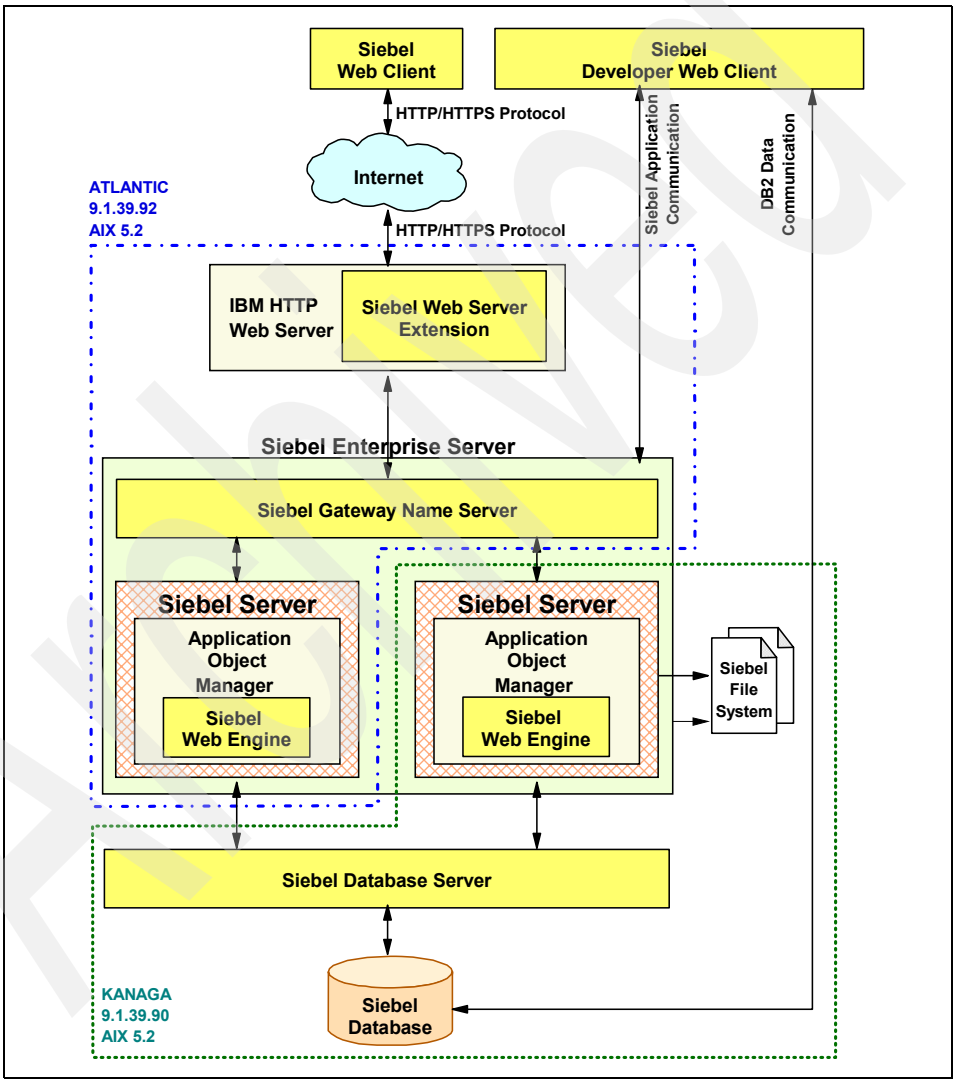

Figure 4-2 Siebel 7.8.2 installation and configuration on AIX environment

Table 4-1 lists the operating system type, level, and hardware that we used.

Table 4-1 Operating system specification (lab)

Operating system type	Installed software and hardware
AIX operating system	AIX 5L™ v5.2 ML 04 with APAR IY65761. C++ Runtime version 6.0.0.13
	pSeries 630 with four processors @ 1200 MHz with 4 GB of memory.
Windows operating system	Windows 2000 with Service Pack 4
	2-Way 2GHz Xeon processor with 3 GB of memory.
Linux operating system	Red Hat Enterprise Linux (RHEL) 4.0 with base kernel 2.6.9
	2-Way 2GHz Xeon processor with 3 GB of memory

Table 4-2 lists the AIX environment details.

Table 4-2 AIX environment (lab)

AIX hostname	Installed component
ATLANTIC	Siebel Gateway Name Server
	IHS Web Server
	Siebel Web Server Extension
	Siebel Server
KANAGA	Siebel Server
	Siebel Database Server (dbsrvr)
	Siebel File System
	DB2 Universal Database

Windows environments

Figure 4-3 on page 51 illustrates the Siebel system in the Windows environment.

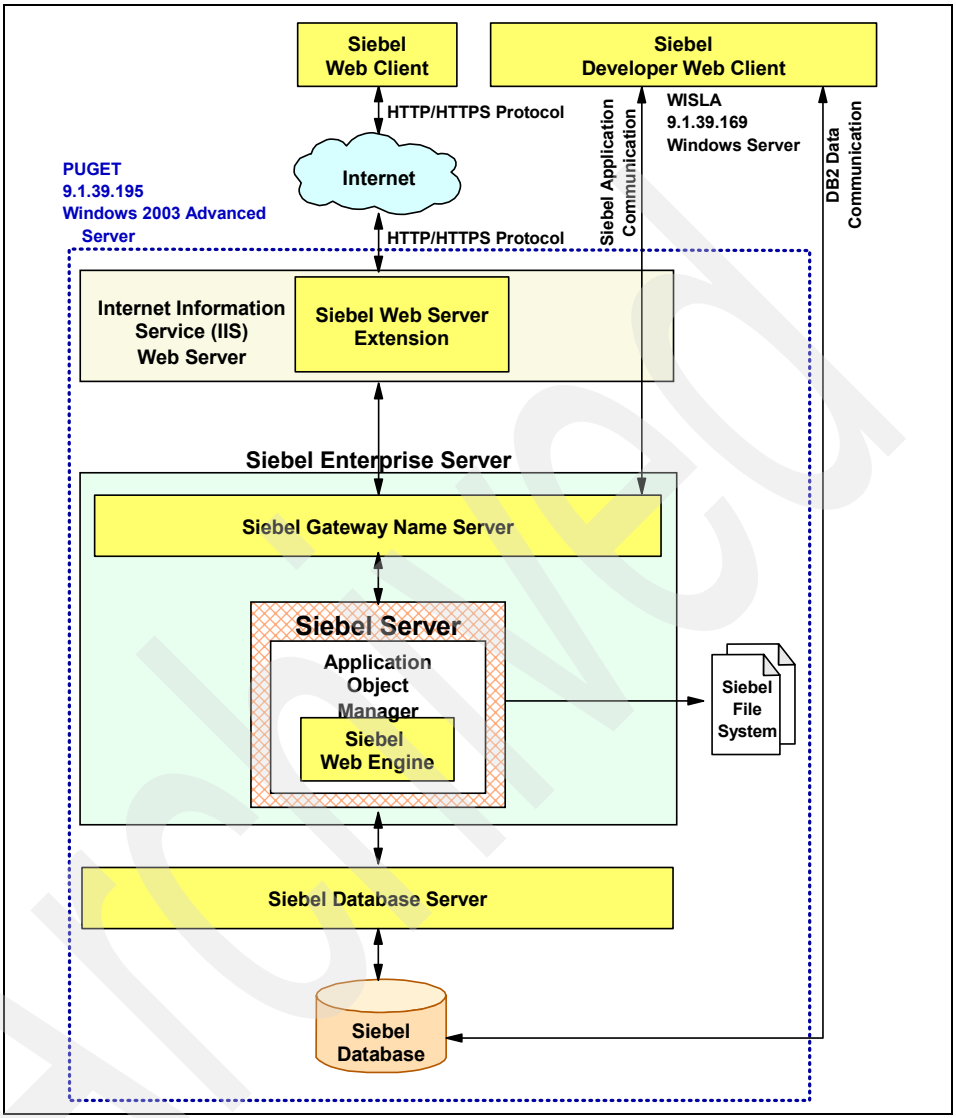

Figure 4-3 Siebel 7.8.2 installation and configuration on Windows environment

Table 4-3 on page 52 lists the Window environment details.

Table 4-3 Windows environment (lab)

Windows host name	Installed components
PUGET	Siebel Gateway Name Server
	Siebel Server
	Siebel Database Server
	Siebel Web Server Extension
	Siebel File System
	DB2 Universal Database

Database server on Linux

We also set up a database server on Linux. Table 4-4 lists the Linux environment details.

Table 4-4 Linux environment (lab)

Linux Hostname	Installed component
KAZAN	DB2 Universal Database

4.3 Installing DB2 UDB and Siebel on AIX

In this section, we describe the installation process for DB2 UDB and Siebel on AIX.

For supported RDBMS and system requirements, refer to *System Requirements and Supported Platforms* on the Siebel support Web site:

`http://supportweb.siebel.com`

This Web site requires a login account from Siebel Technical Support.

For more information on supported DB2 UDB version and FixPaks, refer to "Siebel release matrix for DB2 UDB" on page 26.

4.3.1 Installation and configuration of DB2 UDB server and client on AIX

In this section, we briefly describe the DB2 UDB server installation process. The IBM DB2 UDB documentation, *Quick Beginnings for DB2 Servers V8*, GC09-4836-01, provides detailed installation instructions.

You can create the DB2 UDB instance at the time of DB2 UDB installation. In our lab, we installed the DB2 UDB software first, and then created and configured the instances.

Installation of DB2 UDB server

To invoke DB2 UDB install, use the product CD or go to the directory where DB2 UDB software is located and run the DB2 install command. In our lab, DB2 UDB server V8.1 ESE with FixPak 8s was installed on KANAGA using the following commands:

```
./db2install -p DB2.ESE
Usage: db2_install [-p <product_name>] [-d <directory>] [-z <device>]
the different options for -p
 DB2.ESE        DB2 Enterprise Server Edition for AIX
 DB2.ADMCL      DB2 Administration Client for AIX
 DB2.ADCL       DB2 Application Development Client for AIX
```

Once the installation is complete, the following message is displayed. xxxxxx represents the log number:

```
The installation logfile can be found in /tmp/db2_install_log.xxxxxx
db2_install program completed successfully.
```

Check the log file db2_install_log.xxxxxx for any errors by running the grep command on the log file for SUCCESS and FAILED. For example:

```
cat db2_install_log.xxxxxx | grep SUCCESS
cat db2_install_log.xxxxxx | grep FAILED
```

After the successful base installation, apply the Siebel specific FixPak fp8s. Untar the file siebel_13948_fp5L64.tar and run command:

```
./installFixPak -a -y
```

Run the command, **lslpp -l|grep db2**, to see the DB2 files installed.

Configuration of DB2 UDB server

This section explains DB2 UDB configuration for Siebel Business Applications running on AIX. The scripts we used in our lab are listed in Appendix A.1, “Database scripts for AIX” on page 340. The scripts can serve as reference documentation. The parameters chosen would need to be reviewed for relevance to your physical environment and usage. Refer to Chapter 8, “Performance tuning” on page 199 for recommended DB2 UDB configuration parameter settings for Siebel/DB2 environment.

- Create a DB2 UDB instance using the command **db2icrt**

We created two DB2 instances. A 64-bit instance named `ucsdb2` and a 32-bit instance named `sebldb2`. To create a DB2 instance, use the `db2icrt` command in the directory `/usr/opt/db2_08_01/instance`.

We did not have a separate fenced id for `sebldb2`, since it does not host local databases.

```
./db2icrt -a SERVER_ENCRYPT -p 8100 -s ese -u fencdb2 -w 64 ucsdb2
./db2icrt -a SERVER_ENCRYPT -p 8200 -s ese -u sebldb2 -w 32 sebldb2
```

After the instances are created, stop and start the DB2 UDB instances using the **db2stop** and **db2start** commands.

To ensure that you have the correct DB2 UDB version and FixPaks installed, execute the following command from the DB2 instance ID:

```
db2level
DB21085I  Instance "ucsdb2" uses "64" bits and DB2 code release "SQL08021"
with level identifier "03020106".
Informational tokens are "DB2 v8.1.1.81", "siebel_13948", "U800400_13948",
and FixPak "8".
Product is installed at "/usr/opt/db2_08_01".
```

- Create directories for database

  Create the directory structure that holds the database using **mkdir** command.

  In our lab, we created the directories using the script `Create.dir`:

  ```
  sh create.dir ucsdb2 vol78rba
  ```

- Update the DB2 UDB registry variables and database manager configuration parameters

  – Set DB2 registry variables using **db2set** command.

    In our lab, we used script `db2set.all.sh` to set the registry variables:

    ```
    sh db2set.all.sh ucsdb2
    ```

  – Updating database manager configuration parameters.

    The database manager configuration parameters can be set using the `UPDATE DATABASE MANAGER CONFIGURATION` command on the DB2 command line processor or by using the DB2 Control Center.

    In our lab, we used the script `dbmcfg.sh` to set all the database manager configuration parameters:

    ```
    sh dbmcfg.sh ucsdb2
    ```

    After the DB2 registry variables and database manager configuration parameters are updated, stop and start the DB2 UDB instance using the **db2stop** and **db2start** commands.

- Apply the DB2 UDB license key

Depending on the product installed, you may need to apply the license key. If you need to update the license key, a warning message will be shown in the `db2diag.log`. Locate the license file and update the license key using the **db2licm** command:

```
db2licm -a db2ese.lic
```

After updating the license, stop and start the DB2 UDB instance using the **db2stop** and **db2start** commands.

- Create DB2 UDB unicode database

  Create the database using the `CREATE DATABASE` command.

  In our lab, we created a unicode database with DMS catalog table space with the script `create_unicode.sh:`

  ```
  sh create_unicode.sh ucsdb2 vol78rba
  ```

  To verify that the database was created successfully, try to connect to the database, using:

  ```
  db2 connect to vol78rba
  ```

- Set database configuration parameters

  You set the database configuration parameters using the `UPDATE DATABASE CONFIGURATION` command on the DB2 command line processor or by using the DB2 Control Center.

  In our lab, we used the script `dbcfg.sh` to update the database manager configuration parameters:

  ```
  sh dbcfg.sh ucsdb2 vol78rba
  ```

  After changing the database manager parameters, stop and start the DB2 UDB instance using the **db2stop** and **db2start** commands.

- Create buffer pools

  You need to create one buffer pool for each table space page size that you plan to create. Siebel recommends that you have three different table spaces of sizes 4K, 16K, and 32K. You must create three buffer pools with three different page sizes.

  In our lab, we used the script `create_bp.sh` to create three buffer pools: one of 4K page size, one of 16K page size, and one of 32K page size:

  ```
  sh create_bp.sh ucsdb2 vol78rba
  ```

- Create temporary table spaces

  For each table space page size you plan to have, you also need to create a temporary table space of the same page size.

In our lab, we created three temporary table spaces of 4K, 16K, and 32K using the DDL file `create_temptbs.ddl`:

```
db2 -tf create_temptbs.ddl
```

- Create table spaces

  We created DMS table spaces for data. We highly recommend you use DMS table spaces for medium and large size databases.

  In our lab, we created two regular table spaces of 4K page size and a 32K page size using the following script:

  ```
  db2 -tf create_tbs.ddl > create_tbs.out
  ```

  Check `create_tbs.out` for any errors.

- Increase the number of ODBC statement handles

  For a Siebel DB2 database, it is recommended that you set the number of ODBC client statement handles to 6.

  In our lab, we ran the following commands to set ODBC client statement handles to 6:

  ```
  db2 connect to vol78rba
  cd /home/ucsdb2/sqllib/bnd
  db2 bind @db2cli.lst blocking all grant public clipkg 6
  db2 terminate
  ```

Installation of DB2 UDB client

Refer to IBM DB2 UDB manual, *Quick Beginnings for DB2 Clients V8*, GC09-4832-01, for detailed installation instructions.

In our lab, we installed DB2 UDB client version V8.1 with FixPak 8s on `ATLANTIC`.

To install DB2 UDB client, use the product CD or go to the directory where the client code is available and run the following command:

```
./db2install -p DB2.ADMCL
```

Once the installation is complete, the following message displays:

```
The installation logfile can be found in /tmp/db2_install_log.xxxxxx
db2_install program completed successfully.
```

Check the log file `db2_install_log.xxxxxx` for any errors, by running `grep` command on the log file for `SUCCESS` and `FAILED`.

After the successful DB2 UDB installation, apply the Siebel specific FixPak fp8s. Untar the file `siebel_13948_fp5L64.tar` and run command:

```
./installFixPak -a -y
```

Run the command, **`lslpp -l|grep db2`**, to see the DB2 files installed.

Configuration of DB2 UDB client

After the installation of the DB2 client, configure the client to access the server database.

Complete the following steps:

1. Create a DB2 UDB client instance.

   Create a DB2 UDB client instance on the client to access the server database.

   In our lab, we created an instance `sebldb2` on client `ATLANTIC` using the following command:

   ```
   ./db2icrt -a SERVER_ENCRYPT -s client sebldb2
   ```

   To ensure you have the correct DB2 UDB version and FixPaks installed, execute the following command on the DB2 instance:

   ```
   db2level
   DB21085I  Instance "sebldb2" uses "32" bits and DB2 code release "SQL08021"
   with level identifier "03020106".
   Informational tokens are "DB2 v8.1.1.81", "siebel_13948", "U800400_13948",
   and FixPak "8".
   Product is installed at "/usr/opt/db2_08_01".
   ```

2. Catalog the node and database.

   Catalog the Siebel database on the client using the following commands:

   ```
   db2 catalog tcpip node <nodename> remote <hostname> server <portno>
   ```

   ```
   db2 catalog db <database name> at node <nodename> authentication
   SERVER_ENCYPRT
   ```

   For example:

   ```
   db2 catalog tcpip node KANAGA remote 9.1.39.90 server 8100
   db2 catalog db vol78rba at node KANAGA authentication SERVER_ENCYPRT
   ```

4.3.2 Creation of the Siebel installation image

You install Siebel Business Applications from a network image. The image is created from the Siebel media files using the Siebel Image Creator utility.

To create the Siebel install image, download the Siebel media files (JAR files of both base and language) and other required files (`AIX_ImageCreator`, `imagecreator.jar`, `media.inf`, and `siebel.ini`) from the FTP site on Siebel Support Web or copy the file from the DVD shipped by Siebel Systems into a single directory on a system with network access. Ensure that you have sufficient disk space.

Ensure the user creating the network image has write access permissions to the Siebel image directory.

In our lab, we downloaded the following files to implement Siebel on AIX with ENU language.

- AIX_ImageCreator
- SIA_7.8.2.0_Base_AIX_Siebel_Enterprise_Server.jar
- SIA_7.8.2.0_enu_AIX_Siebel_Enterprise_Server.jar
- SIA_7.8.2.0_Base_AIX_Siebel_Web_Server_Extension.jar
- SIA_7.8.2.0_enu_AIX_Siebel_Web_Server_Extension.jar
- imagecreator.jar
- media.inf
- siebel.ini

Run the Siebel image creator

`YOu can run AIX_ImageCreator` in both GUI and console mode. We chose console mode for AIX.

1. Go to the directory where the Siebel media files exist. In our lab, files were located at: `/installcode/siebel/sia_7.8.2.0`
2. Run the image creator utility:

   ```
   ./AIX_ImageCreator -is:javaconsole -console
   ```

   The message: `Welcome to the InstallShield Wizard for the Siebel Image Creator Utility` displays.
3. Specify whether to create a new image (or add products) to an existing image, or add languages to an existing image. We selected:

   ```
   [X] 1 - Create a new image or add product(s) to an existing image
   ```
4. Specify the directory in which to create the product install image. We entered `/Siebel_Install_Image/SIA_7.8.2.`
5. Select a Siebel version for which to create network installation images. We selected

   ```
   [X] 1 - 7.8.2.0
   ```
6. Specify the application type: either Siebel Enterprise Application (SEA) or Siebel Industry Application (SIA). We selected

```
[X] 2 - Industry (SIA)
```

7. Select a platform. We selected

```
[X] 3 - AIX
```

8. Select products for image creation. We selected

```
[X] 13 - Siebel Enterprise Server
[X] 14 - Siebel Web Server Extension
```

9. Specify the languages you wish to include with the SIA image. We selected

```
[X] 1  - ENU - English (American)
```

10. Specify the location of the indicated Siebel media file. If the media file is not available, download it and place it in the specified directory before clicking **Next**. If all the files are found in the current directory, the Siebel image will be created. We entered

```
/installcode/siebel/sia_7.8.2.0/
```

11. Image Creator indicates that the Siebel image has been created successfully.

    Review the log file `log.txt` for any errors.

4.3.3 Install and configure the Siebel Application in console mode

In this section, we provide the installation and configuration procedures for Siebel Server, Siebel Gateway Name Server, and Database server by demonstrating the setup process in our lab.

For Siebel Load Balancing, the Siebel Enterprise Server consists of two Siebel Servers, `KANAGA` and `ATLANTIC`. The Siebel Gateway Name Server is located at `ATLANTIC`, and the Siebel Database Server was installed and configured on `KANAGA`.

Installing Siebel Gateway Name Server and Siebel Server

The Siebel Gateway Name Server and Siebel Server were installed on `ATLANTIC`. Before installing Siebel components, ensure you complete the following tasks:

- Set umask to 027 on the installation directory.
- Create `/var/adm/siebel` directory and make sure the user ID performing the installation has write access permission to it.
- Modify the `siebel.ini` located in: `/installcode/siebel/Siebel_Install_Image/7.8.2.0/AIX/Server/Siebel_Enterprise_Server/`. Locate the [RunAfter.AIX] section and change the values for the following parameters:

```
ConfigGtwysrvr = no
ConfigSiebsrvr = no
```

Installing Siebel Server in console mode

To install the Siebel Server in console mode, change to the Siebel image location directory. In our lab, it is located in:

```
/installcode/siebel/Siebel_Install_Image/7.8.2.0/AIX/Server/Siebel_Enterpri
se_Server
```

then execute the following command:

```
./setupaix -is:javaconsole -console -args
SS_SETUP_INI=/installcode/siebel/Siebel_Install_Image/7.8.2.0/AIX/Server/Si
ebel_Enterprise_Server/siebel.ini
```

The InstallShield Wizard begins:

1. Enter Siebel Enterprise Server Install Location. We entered

   ```
   /siebel
   ```

2. Select the products you wish to install. We selected Gateway Name Server and Siebel Server to install both products at same time.

   ```
   [X] 1 - Gateway Name Server
   [X] 2 - Siebel Server
   ```

3. Choose the setup type that best suits your needs. We selected

   ```
   [X] 1 - Typical
   ```

4. Select the features for “Siebel Enterprise Server” you would like to install:

   To select/deselect a feature or to view its children, type its number:

   ```
   1.  [X] Gateway Name Server
   2. +[X] Siebel Server
   Other options: 0. Continue installing
   ```

   We selected **0** to continue installing

5. Select the languages you wish to install. We selected

   ```
   [X] 1 - enu - English (American)
   ```

The installer displays the following message:

```
Siebel Enterprise Server will be installed in the following location: /siebel
with the following features:
Gateway Name Server
Siebel Server
   Siebel Server Executables
   Siebel Server Core Components
   Siebel Server Remote Components
   Siebel Server Object Manager
   Siebel Field Service Components
   Data Quality Connector
for a total size: 951.6 MB
```

```
The InstallShield Wizard has successfully installed Siebel Enterprise Server.
Choose Finish to exit the wizard.
```

For any errors, check the log file in `/siebel/log.txt`.

Configuring the Gateway Name Server and Siebel Server

After you install the Gateway and Siebel Servers, configure the Servers.

Configuring the Siebel Gateway Name Server

To configure the Siebel Gateway Name Server, change to the Siebel image location directory. In our lab:

```
/installcode/siebel/Siebel_Install_Image/7.8.2.0/AIX/Server/Siebel_Enterpri
se_Server/gtwysrvrcfg
```

then execute the command:

```
$ ./setupaix -is:javaconsole -is:log /tmp/gtwysrvr_config.log -console
-args StringTable=/siebel/_uninst/ses/table.txt
```

The InstallShield Wizard begins:

1. Specify Gateway Port. We selected

   ```
   2320 (default)
   ```

2. Should the Siebel Gateway Server have autostart enabled? We selected

   ```
   [X] 1 - Yes
   ```

This completes the Siebel Gateway Name Server configuration. Check `/tmp/gtwysrvr_config.log` to ensure it is successful.

To check if the gateway is up, run the following command:

```
$ ps -ef | grep siebsvc | grep -v grep
```

Output of the command shows that the Gateway Name Server process is running:

```
siebsupp 438498      1   0 12:29:26  pts/0  0:00 siebsvc -s gtwyns -a /f
/siebel/gtwysrvr/sys/siebns.dat /t 2320
```

Configuring the Siebel Server

To configure the Siebel Server, change to the siebel image location directory. In our lab:

```
/installcode/siebel/Siebel_Install_Image/7.8.2.0/AIX/Server/Siebel_Enterpri
se_Server/siebsrvrcfg
```

then execute the command:

```
$ ./setupaix -is:javaconsole -is:log /tmp/siebsrvr_config.log -console
-args StringTable=/siebel/_uninst/ses/table.txt
```

The InstallShield Wizard begins.

1. Enter the address of the Siebel Gateway Server. We entered

   `Atlantic.almaden.ibm.com`

2. Enter the port of the Siebel Gateway Server.

   `2320`

3. The indicated Component Groups are selected to be enabled at startup of the Siebel Server. We selected

   `[X] 21 - Siebel Call Center`

4. Enter a name for the new Siebel Enterprise. We entered

   `siebaix`

5. Enter the Siebel File System path. We entered

   `/siebelfs`

6. Provide a brief description of this Enterprise. We entered

   `Siebel Enterprise`

7. Specify the port to be used by the Synchronization Manager. We selected

   `40400`

8. Select the software your Siebel installation will use for data matching. We selected

   `[X] 3 - None`

9. Select the server database used by your organization. We selected

   `[X] 2 - IBM DB2 UDB for Windows and UNIX`

10. Enter the name of the database. We entered

    `vol78rba`

11. Enter the database tableowner. We entered

    `siebel`

12. Enter the database user name to be used by the Siebel Server for version checking, automatically starting of server components, and operation of the Synchronization Manager. We entered `sadmin` for administrator ID.

13. Specify the Chart Server host. We left it blank.

14. Chart Image Format. We selected

    `[X] 1 - png`

15. Encryption Type. We selected

```
[X] 1 - NONE
```

16. Enter a name for the new Siebel Server. We entered

```
atlantic
```

17. Enter a description for this Siebel Server. We entered

```
Siebel Server atlantic
```

18. Enter the name of the directory in which the DB2 client is installed. We specified

```
/home/sebldb2/sqllib
```

19. SynchMgr static port override. We selected

```
[X] 2 - No
```

20. Do you want to configure a Search Server? We selected

```
[X] 3 - Skip
```

21. Should this Siebel Server have autostart enabled? We selected

```
[X] 1 - Yes
```

22. Do you want to start the server? We selected

```
[X] 2 - No
```

23. Deploy Secure Sockets Layer (SSL) in the Enterprise. We selected

```
[X] 2 - No
```

24. Click **Next** to apply settings.

```
Primary Language = enu
Language for Application Object Manager = enu
Gateway Address = Atlantic.almaden.ibm.com
Gateway Port = 2320
Siebel Enterprise = siebaix
FileSystem = /siebelfs
Description of the Enterprise = Siebel Enterprise
Database Username = sadmin
Database Password = ********
Database Platform = IBM DB2 UDB for Windows and UNIX
Database Name = vol78rba
Table Owner = siebel
Synch Manager Port = 40400
Data Matching = None
Chart Server Host =
Chart Image Format = png
Encryption Type = NONE
Server Name = atlantic
```

Press Enter to read the text.

The installer displays the parameters you have chosen, for review:

```
Description of this Siebel Server = Siebel Server atlantic
DB2 Directory = /home/sebldb2/sqllib
Synch Manager port override = N
Configure Search Server = Skip
Autostart = Y
Start Server = N
Deploy Secure Sockets Layer (SSL) in the Enterprise = false
```

To correct any values, Press **2** for Previous menu window and fix the value. When verified, Press **1** for the next window for the wizard to apply the settings.

This completes the Siebel Server configuration and the settings will be applied now.

Check the log file `/tmp/siebsrvr_config.log` to make sure it was successful.

Installation and configuration of Siebel Database Server

We use the console mode to install both Siebel Server and Database Server at `KANAGA`. The installation procedure is the same as installing the Siebel Gateway Name Server described in “Installing Siebel Gateway Name Server and Siebel Server” on page 59 except for Step 2 and Step 4 where we selected Siebel Server and Database Server.

2. Select the products you wish to install. We selected

```
[X] 2 - Siebel Server
[X] 3 - Database Server
```

4. Select the features for “Siebel Enterprise Server” you would like to install on Siebel Enterprise Server. We selected

```
1. +[x] Siebel Server
2. +[x] Database Server
```

Configuring the additional Siebel Server

The second Siebel Server on `KANAGA` was configured with the following settings.

```
Gateway Address = atlantic.almaden.ibm.com
Gateway Port = 2320
Component Groups = Siebel Call Center
Enterprise of Siebel Server = siebaix
Server Name = kanaga
Description of this Siebel Server = Siebel Server kanaga
Synch Manager port override = N
Configure Search Server = Skip
Autostart = Y
Start Server = N
```

```
Deploy Secure Sockets Layer (SSL) in the Enterprise = false
```

Refer to “Configuring the Gateway Name Server and Siebel Server” on page 61 for Siebel Server configuration steps.

Copying the Siebel stored procedure file

After installing the Siebel Database Server, the DB2 stored procedure file, `siebproc` (`siebproc64` for 64-bit database servers), needs to be copied from the Siebel database server install location to the DB2 instance as *siebproc*. Make sure that permission on the file is set to 555.

Copy the file from `$DBSRVR_ROOT/db2udb/siebproc/DBSRVR_OS` to the appropriate `$INST_HOME/sqllib/function` directory

where `$INST_HOME` is the directory where the instance is located.

In the lab, we copied the file `siebproc64` from `/siebel/dbsrvr/db2udb/siebproc/aix` to */home/ucsdb2/sqllib/function/* as `siebproc`.

Configuring the DB2 UDB database for Siebel

Before configuring the Siebel Database, execute the `grantusr.sql` script located in the `$DBSRVR_ROOT/db2udb` subdirectory.

In our lab, our file was located at `/siebel/dbsrvr/db2udb/grantusr.sql`

The `grantusr` script performs the following functions:

- Grants database administrator access permissions to user `siebel` (table owner account) that will own all the data objects.

  ```
  GRANT DBADM ON DATABASE TO USER SIEBEL
  ```

- Grants database privileges to group `sse_role`.

  ```
  GRANT CONNECT ON DATABASE TO GROUP sse_role
  ```

Installing Siebel Database components

Before running the database server configuration utility, make sure to navigate to `$SIEBEL_ROOT` and source Siebel environmental variables (`siebenv.sh`) to set the Siebel environment variables.

`$SIEBEL_ROOT` is the location of the Siebel Server root directory, for example, `/siebel/siebsrvr`.

The Siebel database component installation procedure creates a Siebel schema, imports Siebel seed data, and sets the system preferences.

Launch the database server configuration utility from `<$SIEBEL_ROOT>/bin` and run:

```
$ ./dbsrvr_config.ksh
```

The script launches the installer.

1. Validate the `$SIEBEL_ROOT` directory and language environment variables

   ```
   SIEBEL_ROOT =  /siebel/siebsrvr
   LANGUAGE =  ENU
   Are these correct?  (Y/N).
   ```

   We selected

   ```
   Y
   ```

2. Specify the Siebel Server root directory. We entered

   ```
   /siebel/siebsrvr
   ```

3. Specify the Database Server root directory. We entered

   ```
   /siebel/dbsrvr
   ```

4. Select the RDBMS Platform. We selected

   ```
   1- IBM DB2 UDB for Unix and Windows
   ```

5. Select the Siebel Database Operation. We selected

   ```
   1- Install Database
   ```

6. Has GRANTUSR.SQL been run by the DBA to create Siebel users/roles? We selected

   ```
   1- Yes
   ```

7. Select one of the following options. We selected

   ```
   1- Install Siebel Database
   ```

8. Specify the Database Encoding. We selected

   ```
   1- UNICODE
   ```

9. Choose one of the following languages. We selected

   ```
   1- English
   ```

10. Specify the ODBC Data Source Name for the database. We entered

    ```
    siebsrvr_siebaix
    ```

11. Specify the Database User Name. We entered

    ```
    sadmin
    ```

12. Specify the Database Password. Enter your password.

13. Specify the Table Owner for the database. We entered

    `siebel`

14. Specify the Table Owner Password for the database. Enter your password.
15. Select the Database Server operation system. We selected

    `3- AIX`

16. Specify the Index Table Space name. We entered

    `SIEBEL_4KI`

17. Specify the 4K Table Space name. We entered

    `SIEBEL_4K`

18. Specify the 16K Table Space name. We entered

    `SIEBEL_16K`

19. Specify the 32K Table Space name. We entered

    `SIEBEL_32K`

20. Siebel Log Process.

    Press Enter for default.

21. Review the parameters.

```
ConfigAction = install
InstallType = install
DatabasePlatform = db2udb
SiebelServerRoot = /siebel/siebsrvr
DbsrvrRoot = /siebel/dbsrvr
ODBCDataSource = siebsrvr_siebaix
UserName = sadmin
Password = ********
TableOwner = siebel
TablePassword = ********
Language = ENU
IndexSpace = SIEBEL_4KI
TableSpace = SIEBEL_4K
TableSpace16K = SIEBEL_16K
TableSpace32K = SIEBEL_32K
DatabaseServer = aix
DatabaseOwner = siebel
SiebelLogDir = /siebel/siebsrvr/log/install/output
SiebelLogProcess = install
SiebelLogEvents = 3
OracleParallelIndex = N
```

22. Would you like to run `srvrupgwiz`? We selected

    `2- No`

We ran the Upgrade wizard later in the `nohup` mode.

Run the following command from `<$SIEBEL_ROOT>/bin`.

```
nohup srvrupgwiz -m master_install.ucf  >> master_install.log &
```

To see the data being appended to the `master_install.log`, type the following command:

```
tail -f master_install.log (and press Enter)
```

> **Note:** The Siebel Upgrade Wizard is restartable at most stages within the process. If there are any errors, verify, resolve the failures, and the process may be restarted. The install will continue from the last step that completed successfully.

To restart the upgrade, execute the command:

```
srvrupgwiz -m master_install.ucf
```

Configuring Database Import

The final step is to populate the repository tables in the Siebel Database.

Importing repository

To import the data to the repository, execute `dbsrvr_config.ksh` located at Use `<$SIEBEL_ROOT>/bin`. The script launches the installer.

1. Validate the `$SIEBEL_ROOT` directory and language environment variables

   ```
   SIEBEL_ROOT =  /siebel/siebsrvr
   LANGUAGE =  ENU
   Are these correct?  (Y/N).
   ```

   We selected **Y**.

2. Specify the Siebel Server root directory. We selected

   ```
   /siebel/siebsrvr
   ```

3. Specify the Database Server root directory. We selected

   ```
   /siebel/dbsrvr
   ```

4. Select the RDBMS Platform. We selected

   ```
   1- IBM DB2 UDB for Unix and Windows
   ```

5. *Select the Siebel Database Operation.* We selected

   ```
   3- Import/Export Repository
   ```

6. Select one of the following options. We selected

   ```
   1- Import Repository
   ```

7. Specify your import repository option. We selected

```
1- Import Siebel Repository
```

8. Choose one of the following languages. We selected

```
1- English
```

9. Specify the ODBC Data Source Name for the database. We entered

```
siebsrvr_siebaix
```

10. Specify the Database User Name. We entered

```
sadmin
```

11. Specify the Database Password. Enter your password.
12. Specify the Table Owner for the database. We entered

```
siebel
```

13. Specify the Table Owner Password for the database. Enter your password.
14. Specify the import repository name. We entered

```
Siebel Repository
```

15. Specify the repository file name. We entered

```
/siebel/dbsrvr/common/mstrep.dat
```

16. Siebel Log Process. The default is `imprep`.

```
Press Enter for default
```

17. Review the parameters. The parameters we entered were:

```
ConfigAction = imprep
ImprepType = imprep
DatabasePlatform = db2udb
SiebelServerRoot = /siebel/siebsrvr
DbsrvrRoot = /siebel/dbsrvr
ODBCDataSource = siebsrvr_siebaix
UserName = sadmin
Password = ********
TableOwner = siebel
TablePassword = ********
RepositoryName = Siebel Repository
Language = ENU
DatabaseOwner = siebel
SiebelLogDir = /siebel/siebsrvr/log/imprep/output
SiebelLogProcess = imprep
SiebelLogEvents = 3
OracleParallelIndex = N
RepositoryFileName = /siebel/dbsrvr/common/mstrep.dat
```

18. Would you like to run `srvrupgwiz`? We selected

```
2- No
```

To execute the srvrupgwiz in nohup mode, execute the following command from <$SIEBEL_ROOT>/bin.

```
nohup srvrupgwiz -m master_install.ucf  >> master_install.log &
```

To see the data being appended to the master_imprep.log, type the following command:

```
tail -f master_imprep.log
```

If there are any errors, verify and resolve the failure. The process can be restarted. The imprep will start from the last step that failed.

To restart the upgrade, execute the command:

```
srvrupgwiz -m master_install.ucf
```

Populating the Siebel File System

The Siebel Database Server installer creates a subdirectory *files* under $DBSRVR_ROOT. This directory will contain the file attachments for the Siebel seed data and must be copied into the Siebel File System.

Load Balancing

Siebel Load Balancer resides in the Siebel Web Server Extension (SWSE) on the Web Server. It allows each instance of SWSE to distribute connection requests to multiple application servers in a round-robin fashion. Application server information is stored in a configuration file.

Generating the Load Balancing configuration file

If Siebel load balancing is used, the load balancing configuration file (lbconfig.txt) must be generated before installing the SWSE. The Siebel Servers that will be part of load balancer need to be up and available. The Application Object Manager needs to be enabled before generating lbconfig.txt.

The procedure to generate the lbconfig.txt is as follows.

- Start the server manager program at enterprise level (without /s option) using the following parameters.

  ```
  srvrmgr -g gateway -e enterprise -u sadmin -p ******
  ```

  From the srvrmgr prompt, run the command:

  ```
  srvrmgr> generate lbconfig
  ```

  This generates the lbconfig.txt file. The file is stored in the admin subdirectory of the Siebel Server installation directory.

- Copy the `lbconfig.txt` file to a shared file system location. During the Siebel Web Server Extension installation, you will be prompted for the location of this file.

  In our lab, we copied the file into `/tmp on ATLANTIC`.

For more details, refer to *Siebel System Administration Guide* and *Siebel Installation Guide for UNIX: Servers, Mobile Web Clients, Tools* in Siebel Bookshelf.

Install Siebel Web Server Extension in console mode

Before installing SWSE in console mode, you must execute the following steps.

1. Modify the `siebel.ini` of Siebel Web Server Extension installer located in `/installcode/siebel/Siebel_Install_Image/7.8.2.0/AIX/Server/Siebel_Web_Server_Extension`.

   Locate the `[RunAfter.Unix]` section and change the values for the following parameter:

   ```
   Config.SWSE.Unix = no
   ```

2. Make sure that the Web Server administrator has permissions on SWSE install directories and Web Server directories.

   In our lab, we gave write access permissions to files under SWSE file system `/siebel/swse` and for all files under Web Server directory `/usr/IBMIHS`.

Note: If the Web Server is installed using root privileges, SWSE must be installed from an user ID with root privileges.

Installing SWSE

Follow these steps to install SWSE:

1. Change to the Siebel image location directory. In our lab,

   ```
   /installcode/siebel/Siebel_Install_Image/7.8.2.0/AIX/Server/Siebel_Web_Server_Extension
   ```

2. Execute command.

   ```
   $ ./setupaix -is:javaconsole -console -args
   SS_SETUP_INI=/installcode/siebel/Siebel_Install_Image/7.8.2.0/AIX/Server/Siebel_Web_Server_Extension/enu/siebel.ini
   ```

   The InstallShield Wizard for Siebel Web Server installation appears.

3. Specify a directory or press Enter to accept the default directory. We entered

   ```
   /siebel/swse
   ```

4. Select the languages you wish to install. We selected

   ```
   [X] 1 - enu - English (American)
   ```

   The installer displays the installation location of SWSE. It also displays the disk space required for the SWSE.

   ```
   Siebel Web Server will be installed in the following location: /siebel/swse
   with the following features:
   Siebel Web Server Extensions for a total size:  129.2 MB
   ```

Configure Siebel Web Server Extension

Before you configure Siebel Web Server Extension, ensure you have already executed the following tasks:

- If another Siebel Enterprise installation exists on the same server, do not source the siebel environment variables.
- Copy the `lbconfig.txt` file generated from the shared directory to `<$SWSE_ROOT>/ADMIN/`

In our lab, we copied `lbconfig.txt` from `/tmp` to `/siebel/swse/admin/`

To configure SWSE:

1. Change to the location of siebel image directory. In our lab, the siebel image files are located in:

   ```
   /installcode/siebel/Siebel_Install_Image/7.8.2.0/AIX/Server/Siebel_Web_Serv
   er_Extension
   ```

2. Execute the command:

   ```
   ./setupaix -is:javaconsole -is:log /tmp/swse_config.txt -console -args
   StringTable=/siebel/swse/_uninst/ses/table.txt
   ```

   The InstallShield Wizard for Siebel Web Server configuration appears.

3. Specify the root directory for the Apache Web Server Instance on which you wish to install the Siebel Web Server Extension Plugin. We entered:

   ```
   /usr/IBMIHS
   ```

4. Specify the Load Balancing Configuration. We selected

   ```
   [X] 3 - Use the Built in Load Balancing Capability of the Siebel Web Server
   Extension
   ```

   If option 3 is selected, it prompts you to enter the full path of the load balancing configuration file (`lbconfig.txt`).

5. Specify Virtual Host File Name. We entered

   ```
   /siebel/swse/admin/lbconfig.txt
   ```

6. Specify the enterprise name. We entered

   `siebaix`

7. Select compression type. We selected

   `[X] 1 - none`

8. Select encryption type. We selected

   `[X] 1 - none`

9. Specify the Web Server Http Port. We selected

   `80 (default)`

10. Specify the Web Server Https Port. We selected

    `443 (default)`

11. Specify a Web Update Protection Key. This key is used to update image files on Web servers from a Siebel Server using a Siebel Web Engine command. We entered

    `itso`

12. Specify anonymous login EMPLOYEE name. We entered

    `guest1`

13. Specify anonymous login CONTACT name. We entered

    `guest1`

14. To see your changes, the Web Server needs to be restarted. Restart Web Server? We selected

    `[X] 1- Yes`

    Click **Next** to apply settings. The installer displays the following settings:

```
Primary Language = enu
Web Server Directory = /usr/IBMIHS
Load Balancing Selection = Use the Built in Load Balancing Capability of
the
Siebel Web Server Extension
Virtual Hosts File Name = /siebel/swse/admin/lbconfig.txt
Siebel Enterprise = siebaix
Compression Type = none
Encryption Type = none
Http Port = 80
Https Port = 443
Web Update Key = fgS6wDKlAWkBAAD/EA==
Employee Name = guest1
Employee Password = ********************
Contact Name = guest1
Contact Password = ********************
Restart Web Server = Y
```

Post installation task for SWSE

After the SWSE installation, modify the Web Server configuration for use with SWSE.

The configuration file `httpd.conf` is located under `$IHS_ROOT/conf.` The `$IHS_ROOT` is the IBM HTTP Server installation directory.

In our lab, we set the following values in the `worker.c` module:

```
<IfModule worker.c>
ThreadLimit         30
ServerLimit          1
StartServers         1
MaxClients          30
MinSpareThreads      1
MaxSpareThreads     30
ThreadsPerChild     30
MaxRequestsPerChild  0
</IfModule>
```

In our lab, we assumed the number of concurrent users to be 15. The value of 30 `in <IfModule worker.c> was` twice the expected number of concurrent users.

The other recommended changes to `httpd.conf` are:

- Set `KeepAliveTimeout` to 15 seconds.
- Set `MaxKeepAliveRequests=0` for maximum performance.

Refer to *Siebel Installation Guide for UNIX: Servers, Mobile Web Clients, Tools* for detailed information.

Siebel Web Client

After the SWSE installation, to bring up the Siebel Web Client:

1. Stop IBM IHS Web Server by executing **`stopapa`** command.
2. Start Siebel Gateway and Siebel Servers.
3. Start IBM IHS Web Server by executing **`startapa`** command.
4. Open the Web browser.
5. Go to the Web site for your Siebel application. In our example, it is: `http://9.1.39.92/callcenter_enu`.
6. Log in using a valid user ID and password.

4.4 Installing Siebel on Windows

In our lab, for Siebel Enterprise on Windows, we used two database environments:

- DB2 UDB on Windows
- DB2 UDB on Linux

4.4.1 Installation and configuration of DB2 UDB server on Windows

Refer to the IBM DB2 UDB manual, *Quick Beginnings for DB2 Servers V8*, GC09-4836-01, for the detailed installation instructions.

You must install FixPak fp8s for DB2 UDB installations of both the server and the client prior to the installation of Siebel code.

Before installing DB2 UDB client, make sure you create the DB2 Administration server user account (`db2admin`) with install privileges.

In our lab, we installed DB2 UDB ESE V8.2 with Siebel FixPak fp8s on `PUGET`.

Installation of DB2 UDB server on Windows

Use DB2 UDB setup wizard to install DB2 UDB.

1. Go to the DB2 install directory or place the DB2 Server (Enterprise Server Edition) Install CD in the drive and double-click **setup.exe**. The *IBM DB2 Setup Launchpad* dialog appears (Figure 4-4 on page 76). Click the **Install Products** tab.

Figure 4-4 DB2 Setup: Install Products window

2. On Setup dialog (Figure 4-5 on page 77), make sure that the DB2 UDB Enterprise Server Edition is selected and click **Next**.

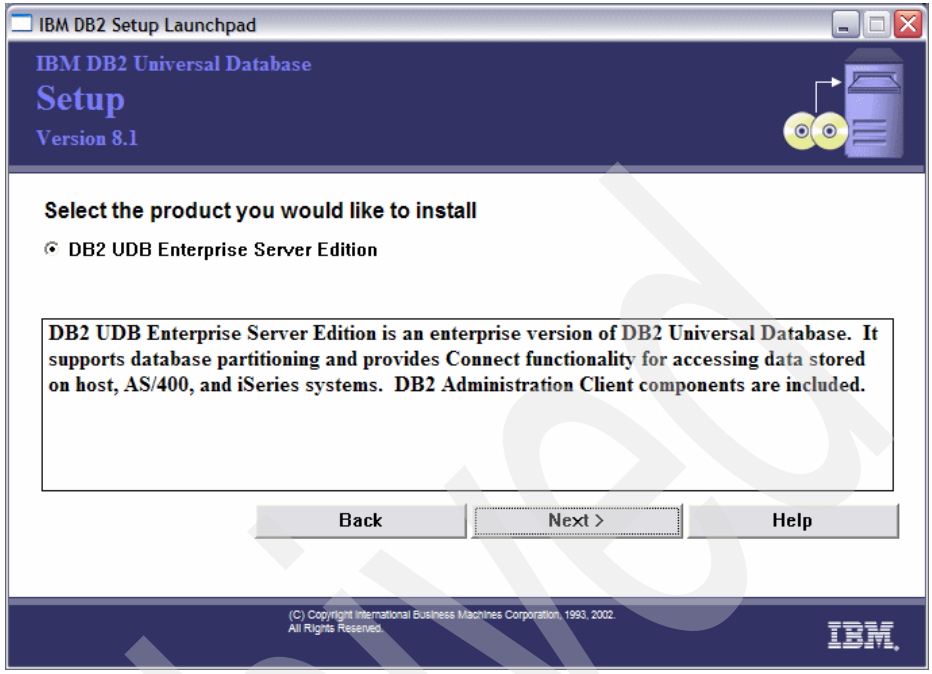

Figure 4-5 DB2 Setup: Select DB2 UDB ESE window

3. On *Welcome to the DB2 Setup Wizard*, click **Next**.
4. On *License Agreement* dialog, choose **I accept the terms of the license agreement** and click **Next**.
5. In the *Select the installation type* dialog (Figure 4-6 on page 78). Select **Typical**.

   Depending on the product you are installing, the following options may be available:

   – Data warehousing
   – Satellite administration capability

   In our lab, we had the above two options unchecked. If needed later, rerun DB2 Setup wizard and select them to install. Click **Next**.

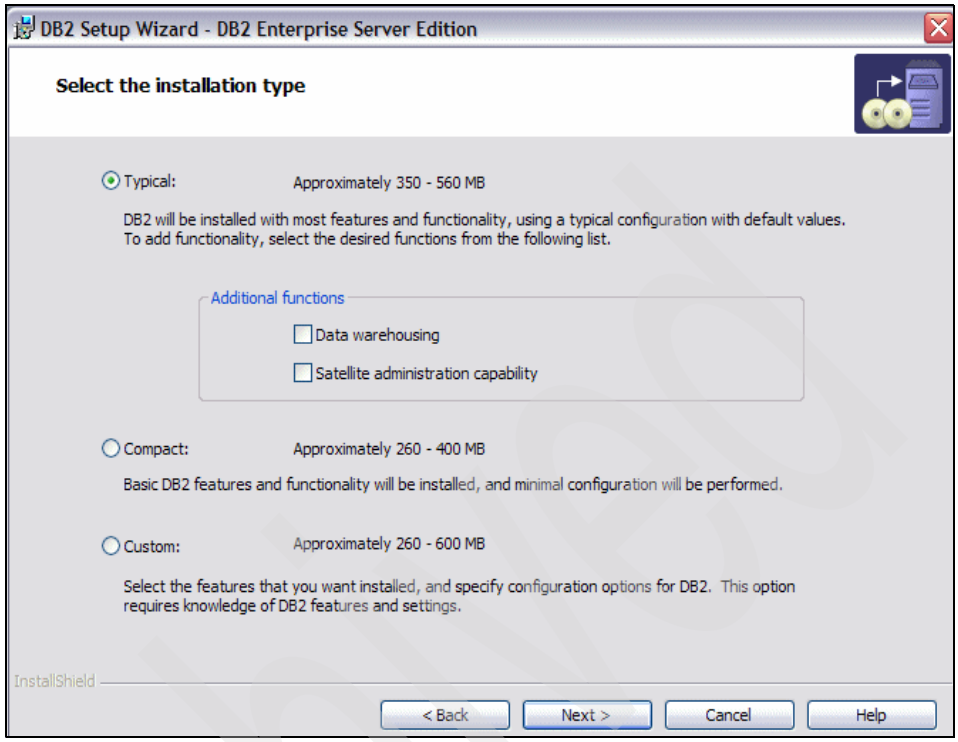

Figure 4-6 DB2 Setup: Installation Type window

6. A warning dialog about APPC support appears. We do not use APPC, so we continue by clicking **OK**.
7. In the *Select the installation action* dialog, select **Install DB2 Enterprise Server edition on this computer**. Click **Next**.
8. The *Select installation folder* dialog (Figure 4-7 on page 79) is shown. Select the drive to install all DB2 files. Specify the path in the drive to where DB2 UDB will be installed. Click **Next**.

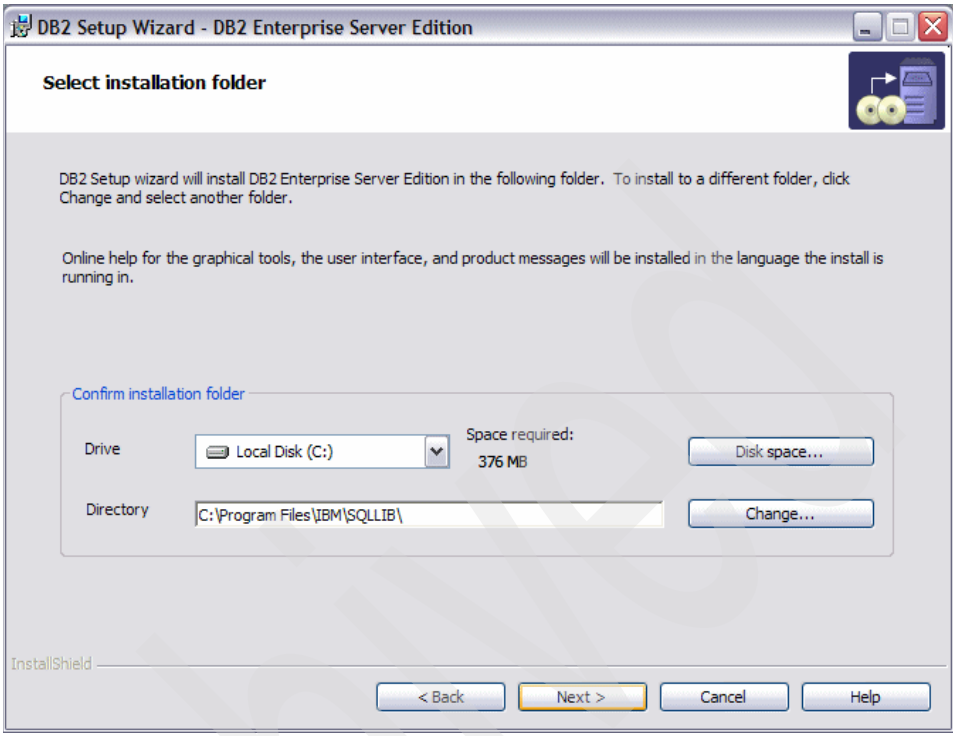

Figure 4-7 DB2 Setup: Installation folder window

9. In *Set user information for the DB2 Administration Server* dialog (Figure 4-8 on page 80). Enter the user account that will start the DB2 administration service (DAS). Enter the domain, user name, and password. Select **Use the same user and password for the remaining DB2 services**. Click **Next**.

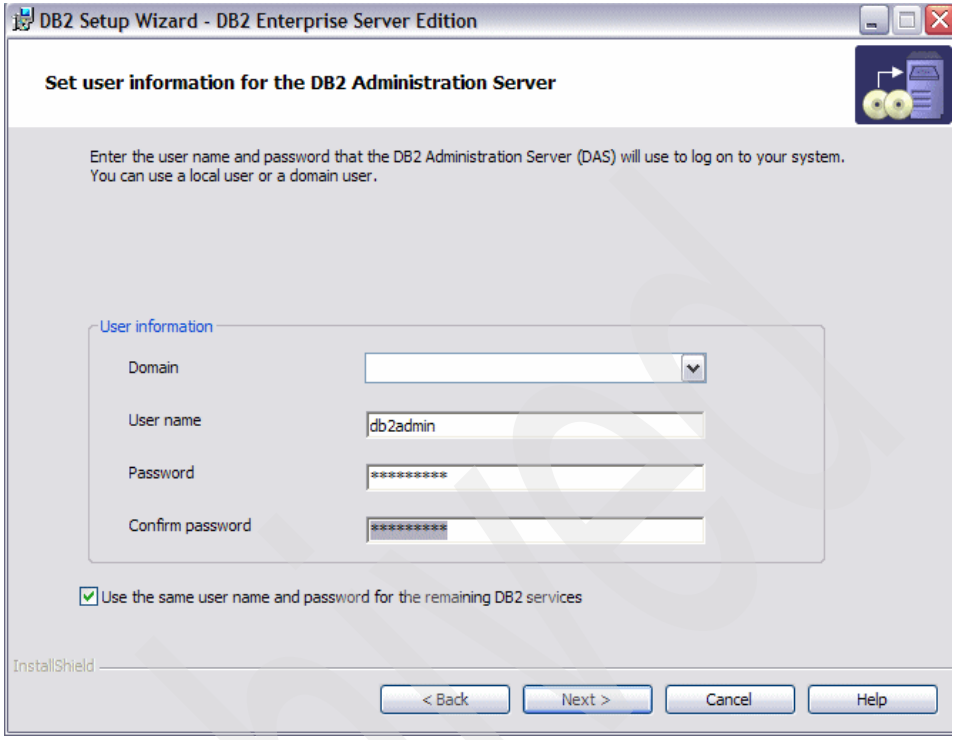

Figure 4-8 DB2 Setup: Set user information window

10. In the *Set up administration contact list* dialog, select **Local - Create a contact list on this system**. Click **Next**.
11. A warning displays advising that there are no SMTP servers specified to which to send notifications. Click **OK**.
12. In the *Configure DB2 instances* (Figure 4-9 on page 81), select the default value **DB2** for DB2 instance. *Protocols...* and *Setup...* buttons allow you to specify network protocols such as port number and so on. We use the default. Click **Next**.

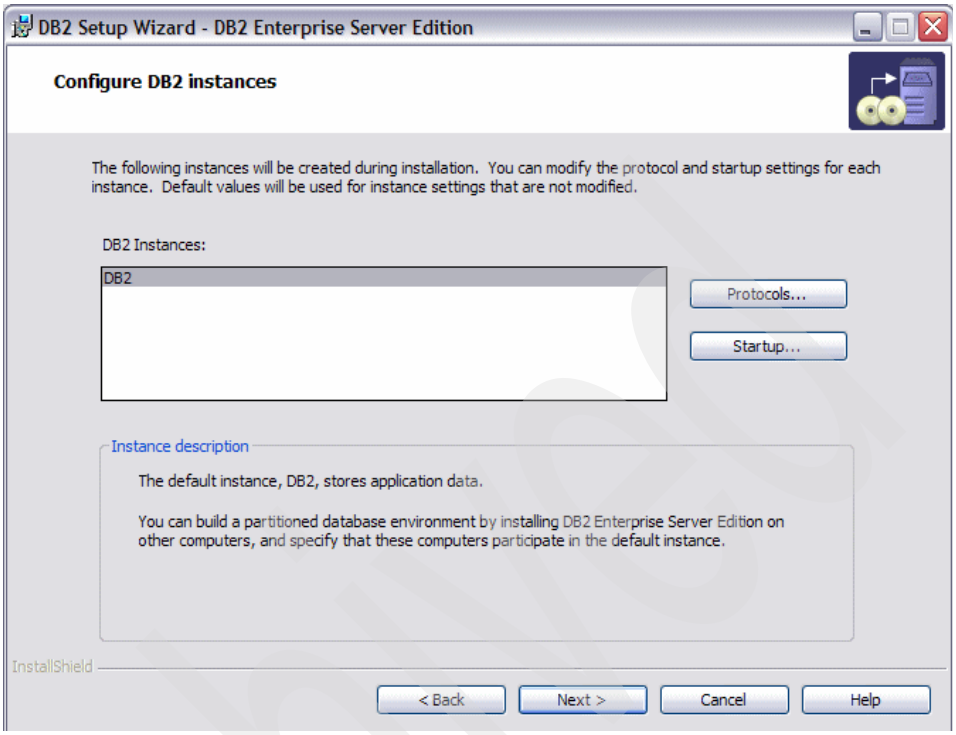

Figure 4-9 DB2 Setup: Configure DB2 instances window

13. In the *Prepare the DB2 tools catalog* dialog, select **Prepare the DB2 tools catalog in a local database**. Click **Next**.

14. In the *Specify a local database to store the DB2 tools catalog* dialog (Figure 4-10 on page 82), specify the local instance in which you want the Setup Wizard to create the tools catalog. The DB2 tools catalog will be stored in a local database. If it does not exist, it will be created. Click **Next**.

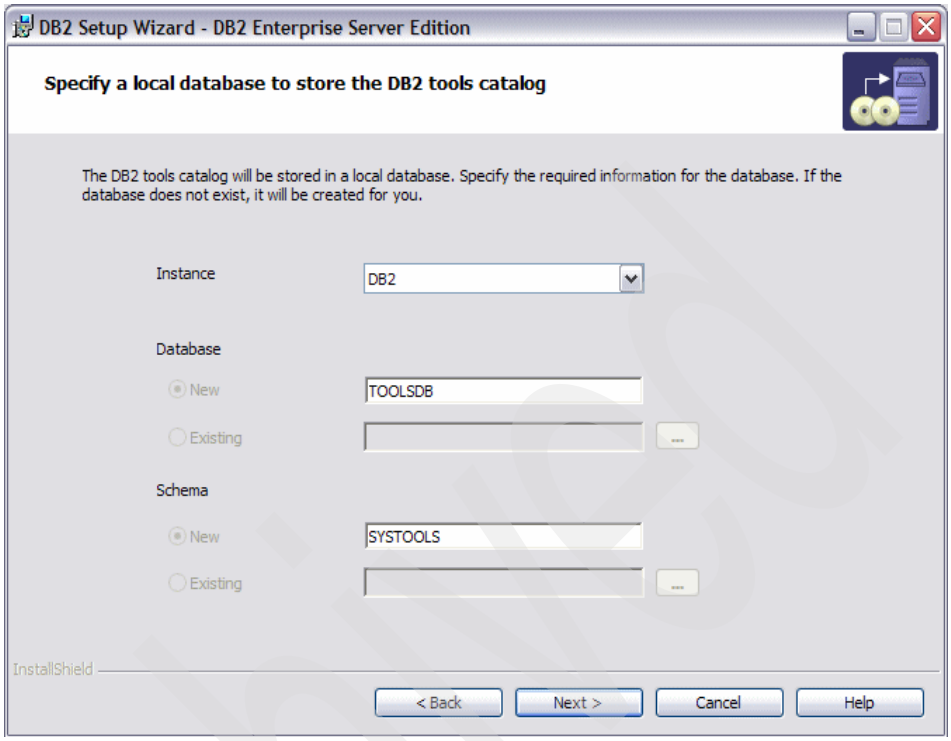

Figure 4-10 DB2 Setup: Local database creation window

15. In the *Specify a contact for health monitor notification* dialog, you can enter the administrator's e-mail address. We select **Defer the task until after installation is complete**. Click **Next**.
16. The *Start copying files* dialog displays the list of files that will be copied. To review or change any settings, click **Back** to make the changes and come back to the same dialog. If the settings are OK, click **Install** to start copying files.
17. Once the installation is complete, the *Setup is complete* dialog (Figure 4-11 on page 83) displays. Click **Finish** to complete the installation process.

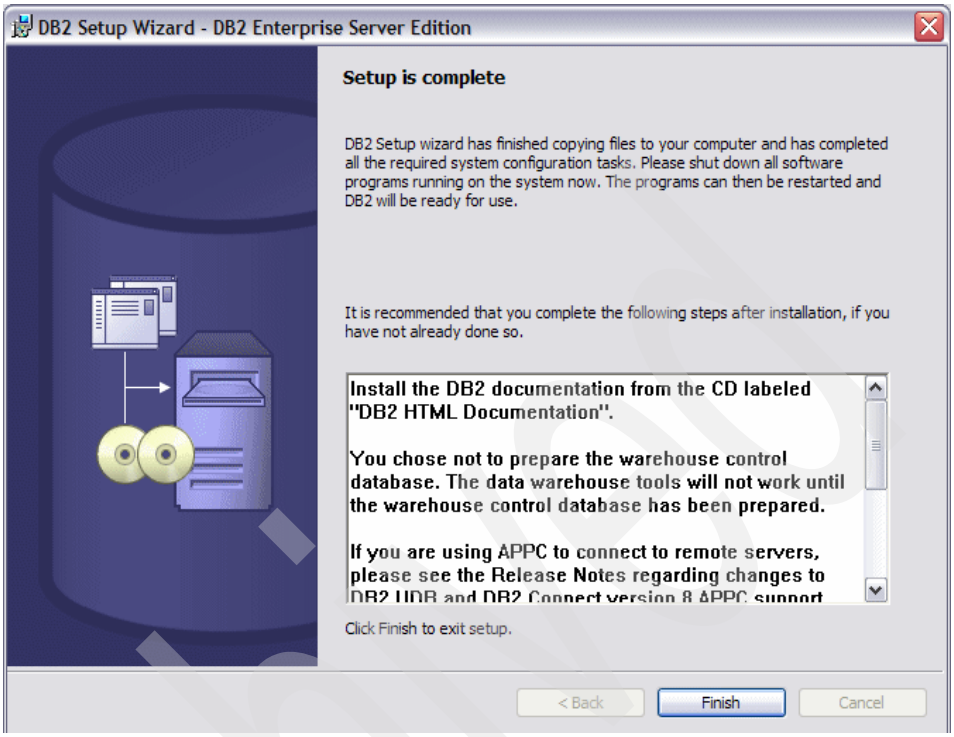

Figure 4-11 DB2 Setup: Setup is complete window

18. The last dialog is *First Steps* which allows you to create a sample database, go to a tutorial, or launch to a quick tool. To exit, click **Exit First Steps**.

After the base installation, download and apply FixPak fp8s. The FixPak is available on FTP site:

`ftp://ftp.software.ibm.com/ps/products/db2/fixes/english-us/siebel/siebel7/DB2V8`

For any errors, check the `db2.log` file located in `\MyDocuments\DB2LOG` directory.

Configuration of DB2 UDB server on Windows

This section explains DB2 UDB configuration for Siebel Business Applications on Windows. The scripts used in our lab are listed in Appendix A.2, “Database scripts for Windows” on page 347. The scripts serve as your reference documentation. The values chosen need to be reviewed for relevance to your physical environment and usage.

1. Create DB2 UDB instance using the command **`db2icrt`**.

In Windows, a default instance DB2 is created during DB2 UDB installation. Because we wanted to use a different name for the instance, we dropped the instance and recreated a new instance named `sebldb2`.

Use the following command to drop the instance.

```
C:\> db2idrop DB2
```

We created a 32-bit instance named `sebldb2` using `db2icrt` command from DOS prompt.

```
C:\> db2icrt sebldb2 -s ese -u db2admin,db2admin -r:8300,8301
```

After the instance is created, stop and start the DB2 UDB instance using the **db2stop** and **db2start** commands.

To ensure the correct DB2 UDB version and FixPak are installed, execute **db2level** from command line processor window. You should have the output as following:

```
DB21085I  Instance "SEBLDB2" uses "32" bits and DB2 code release "SQL08021"
with level identifier "03020106". Informational tokens are "DB2
v8.1.8.897", "siebel_13937", "WR21348_13937", and FixPak "8".
Product is installed at "C:\PROGRA~1\IBM\SQLLIB".
```

2. Create directory for database.

   The directory structure that holds the database is created using **mkdir** command.

   ```
   mkdir C:\db2\siebeldb\sebldb2
   ```

3. Update the DB2 UDB registry variables and database manager configuration parameters.

   – Set DB2 registry variables using **db2set** command.

     In our lab, we used the script `db2set.all.bat` to set the registry variables.

     ```
     C:\> db2set.all.bat
     ```

   – Update database manager configuration parameters.

     You set the database manager configuration parameters by using the `UPDATE DATABASE MANAGER CONFIGURATION` command on the DB2 command line processor or by using the DB2 Control Center.

     In our lab, we used the script `dbmcfg.bat` to set all the database manager configuration parameters.

     ```
     C:\> dbmcfg.bat
     ```

     After the DB2 registry variables and database manager configuration parameters are updated, stop and start the DB2 UDB instance using the **db2stop** and **db2start** commands.

4. Apply the DB2 UDB license key.

Depending on the product installed, the license key may need to be applied. If the license key needs to be updated, a warning message will be shown in the `db2diag.log` located `in C:\Program Files\IBM\SQLLIB\SEBLDB2`.

Locate the license file and update the license key using the **db2licm** command:

```
db2licm -a db2ese.lic
```

After updating the license, stop and start the DB2 UDB instance using the **db2stop** and **db2start** commands.

5. Create DB2 UDB unicode database.

   Create the database using `CREATE DATABASE` command.

   In our lab, a unicode database with DMS catalog table space was created with script `create_unicode.bat`.

   ```
   C:\> create_unicode_db.bat siebeldb
   ```

   This should give you a message back, saying:

   ```
   DB20000I  The CREATE DATABASE command completed successfully.
   ```

   To verify that the database was created successfully, try to connect to the database, using:

   ```
   db2 connect to siebeldb
   ```

6. Set database configuration parameters.

   You set the database configuration parameters using the `UPDATE DATABASE CONFIGURATION` command on the DB2 command line processor or by using the DB2 Control Center.

   In our lab, we used the script `dbcfg.bat` to update the database manager configuration parameters:

   ```
   C:\>dbcfg.bat siebeldb
   ```

   After changing the database manager parameters, stop and start the DB2 UDB instance using the **db2stop** and **db2start** commands.

7. Create buffer pools.

   You need to create one buffer pool for each table space page size that you plan to create. Siebel recommends you have three different table spaces of sizes 4K, 16K, and 32K. You need to create three buffer pools with different page sizes.

   In our lab, we used the script `create_bp.bat` to create three buffer pools: one of 4K page size, one of 16K page size, and one of 32K page size.

   ```
   C:\>create_bp.bat siebeldb
   ```

8. Creating temporary table spaces.

   For each table space page size you plan to have, you also need to create a temporary table space of the same page size.

   In our lab, we created three temporary table spaces of 4K, 16K, and 32K using the DDL file `create_temptbs.ddl`:

   ```
   db2 -tf create_temptbs.ddl
   ```

9. Create table spaces.

   We created DMS table spaces for data. DMS table spaces are highly recommended for medium and large size databases.

   In our lab, we created two regular table spaces of 4K page size and a 32K page size using the following script:

   ```
   db2 -tf create_tbs.ddl > create_tbs.out
   ```

   Check `create_tbs.out` for any errors.

10. Increase the number of ODBC statement handles.

    For a Siebel/DB2 database, it is recommended to set the number of ODBC client statement handles to 6.

    In our lab, we ran the following commands to set ODBC client statement handles to 6:

    ```
    db2 connect to vol78rba
    cd /home/ucsdb2/sqllib/bnd
    db2 bind @db2cli.lst blocking all grant public clipkg 6
    db2 terminate
    ```

4.4.2 Creating the Siebel install image

Siebel Business Applications are installed from a network image. The image is created from the Siebel media files using the Siebel Image Creator utility.

In our lab, we downloaded the following files for to implement Siebel on Windows with the ENU language:

- Windows_ImageCreator
- SIA_7.8.2.0_Base_Windows_Siebel_Enterprise_Server.jar
- SIA_7.8.2.0_enu_Windows_Siebel_Enterprise_Server.jar
- SIA_7.8.2.0_Base_Windows_Siebel_Web_Server_Extension.jar
- SIA_7.8.2.0_enu_Windows_Siebel_Web_Server_Extension.jar
- SIA_7.8.2.0_Base_Windows_Siebel_Web_Client.jar
- SIA_7.8.2.0_enu_Windows_Siebel_Web_Client.jar
- imagecreator.jar
- media.inf
- siebel.ini

Running Siebel Image Creator

Use the following steps to create the Siebel image using Siebel Image Creator:

1. Go to the directory where the Siebel media files and related files are downloaded or copied. In our lab, these files were located at:

   `C:\Siebel_Install_Image\7.8.2`

2. Run the Image Creator utility **`Windows_ImageCreator.exe`**

   The utility displays the *Welcome to the InstallShield Wizard for the Siebel Image Creator utility* dialog. Click **Next**.

3. The next dialog has two selections:

   – Create a new image or add products to an existing image.
   – Add languages to an existing image.

   We selected the first one, then clicked **Next**.

4. In the diaglog to specify the directory in which the product install image is to be created, take the default `C:\Siebel_Install_Image`. Click **Next**.

5. Specify the application type, either Siebel Enterprise Application (SEA) or Siebel Industry Application (SIA), shown in Figure 4-12 on page 88.

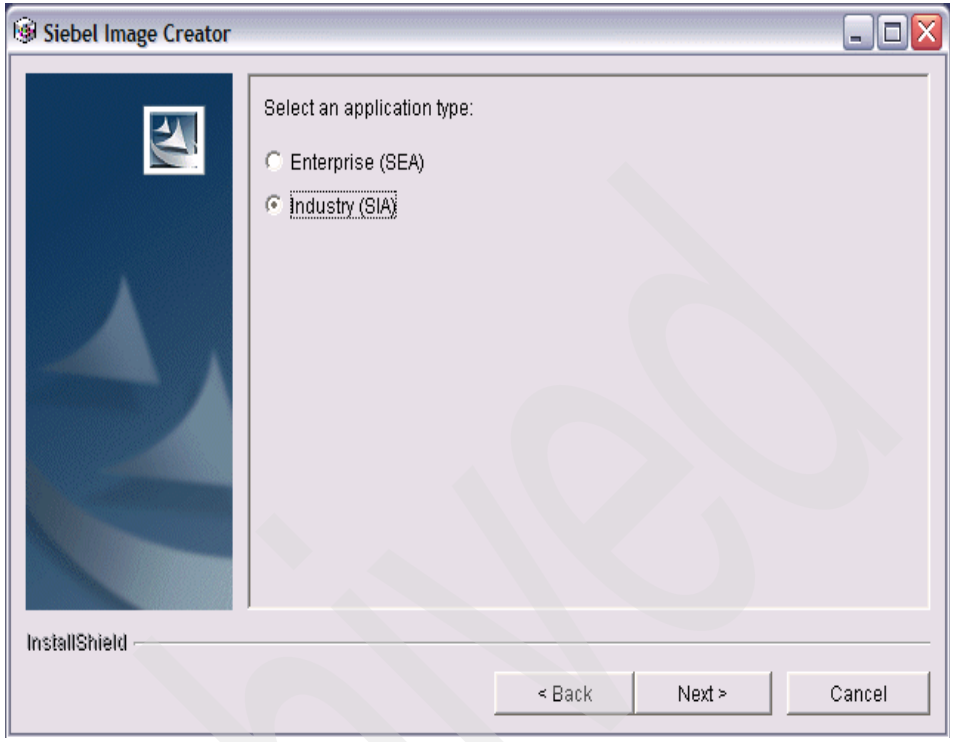

Figure 4-12 Siebel Image Creator application type window

6. In the *Select a platform* dialog, we chose **Windows**. Specify the products to include in the Siebel Image (Figure 4-13 on page 89). Click **Next**.

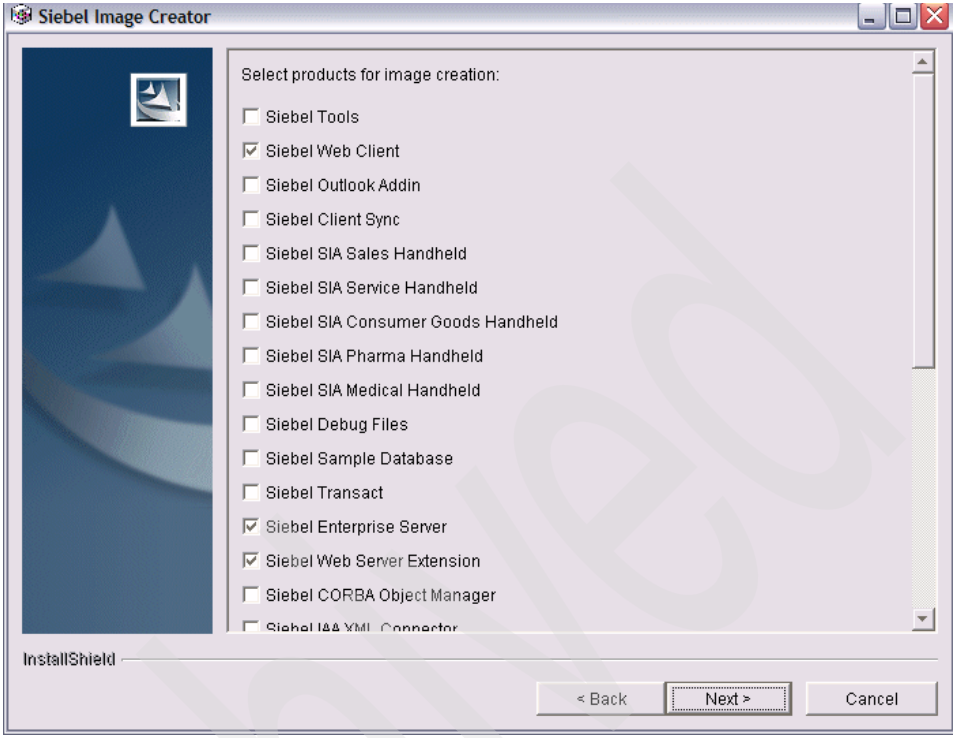

Figure 4-13 Siebel Image Creator product selection window

7. In the specify languages dialog, we chose **ENU-English (American)**. Click **Next**.
8. In this step, you enter the media directory containing SIA_7.8.1.0_Base_Windows_Siebel_Web_Client.jar. If the media file is not available, download it and place it in the specified directory, before clicking **Next**. If all the files are found in the current directory, the Siebel image is created.
9. Once finished, Image Creator indicates that the Siebel image has been created successfully. Click **Finish**.

4.4.3 Installation and configuration of Siebel Application on Windows

This section provides the steps to install and configure the Siebel Application on the Windows environment. Siebel Gateway Name Server, Siebel Server, and Siebel Database Server are installed first then configured separately.

Installing Siebel Gateway Name Server, Siebel Server, and Siebel Database Server on Windows

The following procedure describes the steps for installing and configuring Siebel Gateway Name Server, Siebel Server, and Siebel Database Server.

1. In Windows explorer, navigate to the Siebel image location to execute the `setup.exe`. In our lab, it is located in:

   ```
   C:\Siebel_Install_Image_7.8.2_Win\7.8.2.0\Windows\Server\Siebel_Enterprise_
   Server\
   ```

2. The *Welcome to InstallShield Wizard for Siebel Enterprise Server* window appears. Click **Next**.
3. Confirm the default directory or specify the directory in which to install Siebel Enterprise Server. We entered **C:\sea782**.
4. Select the products you wish to install. We selected:

   ```
   Gateway Name Server
   Siebel Server
   Database Server
   ```

5. Choose the type of installation. We selected **Typical.**
6. Select the language to install. We selected **ENU-English (American)**.
7. Confirm the program folder for shortcuts. We entered **Siebel Enterprise Server Configuration 7.8.**
8. The installer displays the installation location of Siebel Enterprise Server. It also displays the components that would be installed and the disk space required for the SES.

   Siebel Enterprise Server will be installed in `C:\sea78` with the following features:

   ```
   Gateway Name Server
   Siebel Server
   Object Manager Component
   Handheld Synchronization
   Data Quality Connector
   Remote Search Support
   Java Integrator
   Database Server
   Sample Database Support
   Microsoft SQL Server 2000
   Oracle Database Enterprise Edition
   IBM DB2 UDB for Unix and Windows
   IBM DB2 UDB for Z/OS

   for a total size: 2,1,47.3 MB
   ```

9. Clustering Configuration for Siebel Gateway Name Server window appears. We selected **No**.
10. In the *Windows User Account* window, provide the domain name\user account name. If the account is a local account, specify machine name. We entered `\siebsupp` for Windows User Account.
11. Specify the Gateway Name Server Listening Port. We entered **2320** (default).
12. Configure System Service for Automatic Start. We selected **Yes**.
13. Start the Gateway Name Server now. We selected **Yes**.
14. Review the configuration parameters. Following are the parameters for our system:

    ```
    Clustering Configuration for Siebel Gateway Name Server: false
    Windows User Account: .\sadmin
    Windows User Account Password: XXXXXXXX
    Gateway Name Server Listening Port: 2320
    Configure System Service for Automatic start: true
    Start the Gateway Name Server now: true
    ```

    Press Enter to continue.

    The following message appears:

    ```
    The configuration changes were applied successfully.
    ```

    Press Enter to continue.
15. Clustering Configuration for Siebel Servers window appears. We selected **No**.
16. In Co-located Siebel Gateway Name Server window. We selected **Yes**.
17. In the Gateway name Server Listening Port window. We entered:

    ```
    Gateway Name Server Listening Port: 2320(default)
    Enterprise for this server: Siebwin
    ```

18. In the Select the software your Siebel installation will use for data matching, we selected **None.**
19. In the Siebel Database Platform window, we selected **IBM DB2 UDB for Windows and UNIX**.
20. In the Database User Account window, we provided **SADMIN** for Database User Account.
21. In Configure Database Options: Schema Qualifier/Table Owner window, we entered:

    ```
    Schema Qualifier/Table Owner: SIEBEL
    Database Alias:  siebeldb
    ```

22. In Configure Database Options: Register External Oracle DB ODBC Driver window, we selected **No**.

23. In the Setup the Siebel Environment window, we provided `C:\siebelfs` for Siebel File System field.
24. Specify Chart Image Format, we selected **PNG** (default).
25. Specify the encryption type used for communication between the Siebel Server and the Web Server, we selected **None.**
26. The Remote Search Server window prompts you to enter the host name and the port number for the Remote Search Server. We selected Default values.
27. Enable Siebel Components window was shown for selecting component groups to be automatically enabled at Siebel Server startup. The components also can be enabled later after the installation, using server manager. We selected **Siebel Call Center**.
28. Specify the Siebel Server name and description, we entered:

    ```
    Siebel Server Name: PUGET
    Siebel Server Description: Siebel Server PUGET
    ```

29. Specify the port number assigned to Siebel SCBroker component, we selected **2321** (default).
30. Indicate whether or not to override synchronization manager port, we selected **No**.
31. In the Windows User Account window, provide the *<domain name>\user account name*. If the account is a local account, specify machine name. We entered **`\siebsupp`** for Windows User Account.
32. Set the Siebel service to start automatically window appears. We selected **Yes**.
33. Start the Siebel service now window appears. We selected **No**.
34. In Deploy Secure Sockets layer (SSL) in the Enterprise window, we selected **No**.
35. Review the configuration parameters. For any changes, use *Previous* and *Next* buttons to make changes and come back to the same window and click **Next**. The following message appears:

    ```
    'The configuration changes were applied successfully'. Click OK.
    ```

36. To complete the installation, click **Finish**.

Review the log files generated at `<SIEBEL_ROOT>\siebsrvr\LOG` (for example: `C:\sea782\siebsrvr\log`) directory for any errors.

Configuring the Siebel DB2 UDB Database

This section describes the procedure for configuring Siebel DB2 UDB databases for Windows.

Copying the Siebel stored procedure file

After installing the Siebel Database Server and the DB2 stored procedure file, `siebproc` needs to be copied from the Siebel database server install location to DB2 instance as `siebproc`.

Copy the `file siebproc.dll` from Siebel database server installation directory `DBSRVR_ROOT\DB2UDB\SIEBPROC\DBSRVR_OS` to the `FUNCTION` subdirectory within the DB2 UDB instance directory. In the lab, we copied the files from `C:\sea782\dbsrvr\DB2UDB\SIEBPROC\WIN32\` to `C:\Program Files\IBM\SQLLIB\FUNCTION\`

Configuring the Siebel Database on DB2 UDB for Windows

Before configuring the Siebel database, execute the `grantusr.sql` script located in the `C:\sea782\dbsrvr\db2udb` subdirectory to grant database permissions. Ensure the user executing the script has DBA privileges.

Run the commands:

```
db2 connect to DB2 database_alias user instance_owner_username using
password
db2 -vf SIEBEL_ROOT\DBSRVR\DB2UDB\grantusr.sql
```

where `SIEBEL_ROOT` = The full path to the Siebel root directory.

The `grantusr` script:

- Grants permissions to user `SIEBEL` (table owner account) that will own all the data objects.

  ```
  GRANT DBADM ON DATABASE TO USER SIEBEL
  ```

- Grants database privileges to group `sse_role`.

  ```
  GRANT CONNECT ON DATABASE TO GROUP sse_role
  ```

Installing Siebel database components

The database server configuration utility can be launched through Windows desktop or the DOS command prompt. In our lab, we used Windows desktop, **Start → Programs → Siebel Enterprise Server Configuration 7.8.2 → Database Server Configuration:**

1. The Siebel Enterprise Parameters window appears:

   ```
   Gateway Name Server Address: puget
   Enterprise Server Name: Siebwin
   ```

2. Specify the Siebel Server directory. We selected **c:\sea782\siebsrvr.**
3. Specify the Siebel Database Server directory. We selected **c:\sea782\dbsrvr**.

4. Select the RDBMS Platform. We selected **IBM DB2 UDB for Windows and UNIX.**
5. Select the Siebel Database Operation. We selected **Install Database**.
6. In the Siebel User/Roles creation? window, we selected

   GRANTUSR.SQL has been run by the DBA to create Siebel users and roles.
7. Select Installation options. We selected **Install Siebel Database**.
8. Specify Database Encoding. We selected **UNICODE Database**.
9. Specify the ODBC Data Source Name for the database. We entered **siebsrvr_siebwin**.
10. On the Database User Name window, we provided **sadmin** for Database User Name field.
11. On the Database Table Owner window, we provided **siebel** for Database Table Owner field.
12. Select the Database Server operation system. We selected **AIX**.
13. Specify the Index Table Space name. We entered **SIEBEL_4KI**.
14. Specify the 4K Table Space name. We entered **SIEBEL_4K**.
15. Specify the 16K Table Space name. We entered **SIEBEL_16K**.
16. Specify the 32K Table Space name. We entered **SIEBEL_32K**.
17. Specify Log output directory. We selected **install** (default).
18. A message dialog appears with the prompt:

    ```
    To apply configuration now, press OK.
    To apply configuration now, press Cancel.
    The command line to apply the configuration later is
    C:\sea782\siebsrvr\bin\siebupg.exe /m master_install.ucf
    ```

    We clicked **Cancel**.
19. The Configuration parameters window appears for review.

    Review the parameters. For any changes, use *Previous* and *Next* buttons. After verifying the parameters, click **Finish**.
20. A message box appears with the prompt:

    ```
    The configuration changes were applied successfully.
    ```

The Siebel Upgrade Wizard appears, press **OK** to begin install.

The log files can be found under `SIEBEL_ROOT\siebsrvr\LOG` subdirectory.

The Siebel Upgrade Wizard is restartable at most stages within the process. After verifying for errors and resolving any failures, the process may be restarted.

The install will continue from the last step that completed successfully. To invoke Siebel Upgrade Wizard, go to `SIEBEL_ROOT\siebsrvr\bin` and run command:

```
siebupg.exe /m master_install.ucf >> master_install.log
```

Configuring Database import

The Siebel Repository Import populates the Siebel database with Siebel application objects. In our lab, we used the GUI tool provided by Siebel to import database objects.

To launch the tool, **Start** → **Programs** → **Siebel Enterprise Server Configuration 7.8.2** → **Database Server configuration**:

1. The Siebel Enterprise Parameters window appears:

   ```
   Gateway Name Server Address: puget
   Enterprise Server Name: Siebwin
   ```

2. Specify the Siebel Server root directory. We selected **c:\sea78\siebsrvr.**
3. Specify the Database Server root directory. We selected

   ```
   c:\sea78\dbsrvr
   ```

4. Select the RDBMS Platform. We selected

   ```
   IBM DB2 UDB for Windows and UNIX
   ```

5. Select the Siebel Database Operation. We selected

   ```
   Import/Export Repository
   ```

6. Select one of the following options. We selected

   ```
   Import Repository
   ```

7. Specify your import repository option. We selected

   ```
   Import Standard Siebel Repository
   ```

8. Choose one of the following languages. We selected

   ```
   1- English
   ```

9. Specify the ODBC Data Source Name for the database. We entered

   ```
   siebsrvr_siebwin
   ```

10. On the Database User Name window, we provided the following values:

    ```
    Database User Name: sadmin
    Database Password:
    Database Password (confirm):
    ```

11. On the Database Table Owner window, we provided the following values:

    ```
    Database Table Owner: siebel
    Database Table Owner Password.
    Database Table Owner Password (confirm).
    ```

12. On the Import Repository Name window, we entered:

```
Repository Name: Siebel Repository
Repository file name: C:\sea782\dbsrvr\common\mstrep.dat
```

13. Siebel Log output directory. We selected

```
imprep (default)
```

14. A message box appears with the prompt:

```
To apply configuration now, press "OK".

To apply configuration now, press "Cancel".

The command line to apply the configuration later is
C:\sea782\siebsrvr\bin\siebupg.exe /m master_imprep.ucf
```

    We clicked **Cancel.**

15. The configuration parameters window appears for review.

    Review the parameters for any changes, then use *Previous* and *Next*. After verifying the parameters, click **Finish.**

16. A message box appears:

```
The configuration changes were applied successfully.
```

To continue later, go to `SIEBEL_ROOT\siebsrvr\bin` and run command:

```
siebupg.exe /m master_imprep.ucf >> master_imprep.log
```

The Siebel Upgrade Wizard is restartable at most stages within the process. After verifying for errors and resolving any failure, the process may be restarted. The install will continue from the last step that completed successfully.

The log files can be found under `<SIEBEL_ROOT>\siebesrvr\LOG` subdirectory.

Populating the Siebel File System

The Siebel Database Server install creates a subdirectory *files* under `DBSRVR_ROOT`. This directory contains the file attachments for the Siebel seed data, and you must copy it into the Siebel File System.

In our lab, we copied all the files under `C:\sea782\dbsrvr\FILES\` to Siebel File System `C:\siebelfs\`

Installing the Siebel Web Server Extension on Windows

Before installing the SWSE, you must install, configure, and start the Web Server. In our environment, we used Internet Information Services (IIS) Web Server.

Installing SWSE

Before installing SWSE, ensure that the user running the SWSE has system administrator privileges.

1. Go to the directory where the installer is located. In our lab, it is located in:

   `C:\Siebel_Install_Image_7.8.2_Win\7.8.2.0\Windows\Server\Siebel_Web_Server_Extension\`

2. Double-click **`setup.exe`**
3. The Welcome to the InstallShield wizard for Siebel Web Server window appears. Click **Next**.
4. In the Directory Name window, confirm the default directory or specify the directory to install Siebel Web Server. We entered

   `C:\sea782\SWEApp`

5. Select the Language pack for the SWSE. We selected

   `enu - English (American)`

6. The Program Folder for shortcuts window appears. Select the program folder name for the setup to add shortcuts or select the default. We entered

   `Siebel Enterprise Server Configuration 7.8`

7. The SWSE Install Information window appears. The installer displays the location into which it will install the SWSE. It also displays the disk space required for the SWSE.

   ```
   Siebel Web Server will be installed in the following location:
      C:\sea782\SWEApp
   with the following features:
      Siebel Web Server Extensions
   for a total size:
         122.3 MB
   ```

8. In the Load Balancing Configuration window, select the load-balancing method based on your requirement. In our lab, we selected:

   `Single Siebel Server in the Enterprise`

9. In the Server Information window, we entered

   Specify the hostname for the Siebel Server: **PUGET**
   Specify the port number assigned to SiebelSCBroker component: **2321** (default)

10. In the Enterprise Server Name window, we entered

    siebwin

11. In the Encryption Type window, select the encryption type that the Siebel Web Clients should use to communicate with the Application Object Manager (OM). We selected

    `none`

12. In the Compression Type window, select the type of compression to use for communications between the SWSE and Siebel Application Servers. We selected

    `none`

13. In the Web Server Port window, we entered

    ```
    Please specify the Web Server HTTP Port: 80 (default)
    Please specify the Web Server HTTPS Port: 443 (default)
    ```

14. In the Anonymous Employee Login window, we entered **guest1** `for` anonymous login EMPLOYEE name.

15. In the Anonymous Contact Login window, we entered **guest1** for anonymous login CONTACT name.

16. Enter the Web Update Protection Key that the administrator will use for updating files on the SWSE, and click **Next**.

17. In the Deploy Secure Sockets Layer (SSL) in the Enterprise?, we selected

    `No (Unchecked)`

18. The installer displays the configuration values. Review the values. If you need to change any values, use the *Previous* and *Next* buttons and make the changes and come back to this window.

    Click **Next**.

19. The message box appears with the following text:

    `The configuration changes were applied successfully.`

20. To complete the installation, click **Finish**.

Siebel Web Client

After the SWSE installation, to bring up the Siebel Web Client:

1. Stop the Web Server service.
2. Start the Siebel Gateway and Siebel Server services.
3. Start the Web Server service.
4. Open the Web browser.
5. Go to the Web site for your Siebel application.
6. Log in using a valid user ID and password.

4.5 Installation and configuration of DB2 UDB server on Linux

At the time we wrote this redbook (November 2005), Siebel did not support the Linux environment. However, you can install DB2 UDB on Linux and have the Siebel Gateway Name Server, Siebel Server, and Siebel Database Server located on Windows. Figure 4-14 on page 100 illustrates this type of architecture.

The procedure to install DB2 UDB on Linux is similar to the DB2 UDB installation on AIX:

1. Install DB2 UDB ESE server code.
2. Configure the DB2 UDB Server, including:
   - Create DB2 instance
   - Set up DB2 registry variables
   - Configure DB2 database manager and database parameters

   Refer to 4.3.1, “Installation and configuration of DB2 UDB server and client on AIX” on page 52 for details and steps.
3. Copy the Siebel stored procedure

   The DB2 stored procedure file, `siebproc`, needs to be copied from the Siebel Database Server install location to the DB2 instance as `siebproc`:

   Copy the file `siebproc` from `$DBSRVR_ROOT/db2udb/siebproc/DBSRVR_OS` to `$INST_HOME/sqllib/function` directory

   where `$INST_HOME` = The directory where the instance is located.

   In the lab, we copied the file siebproc from `C:\sea782\dbsrvr\DB2UDB\SIEBPROC\LINUX` to `/home/sebldb2/sqllib/function/` on `KAZAN`.

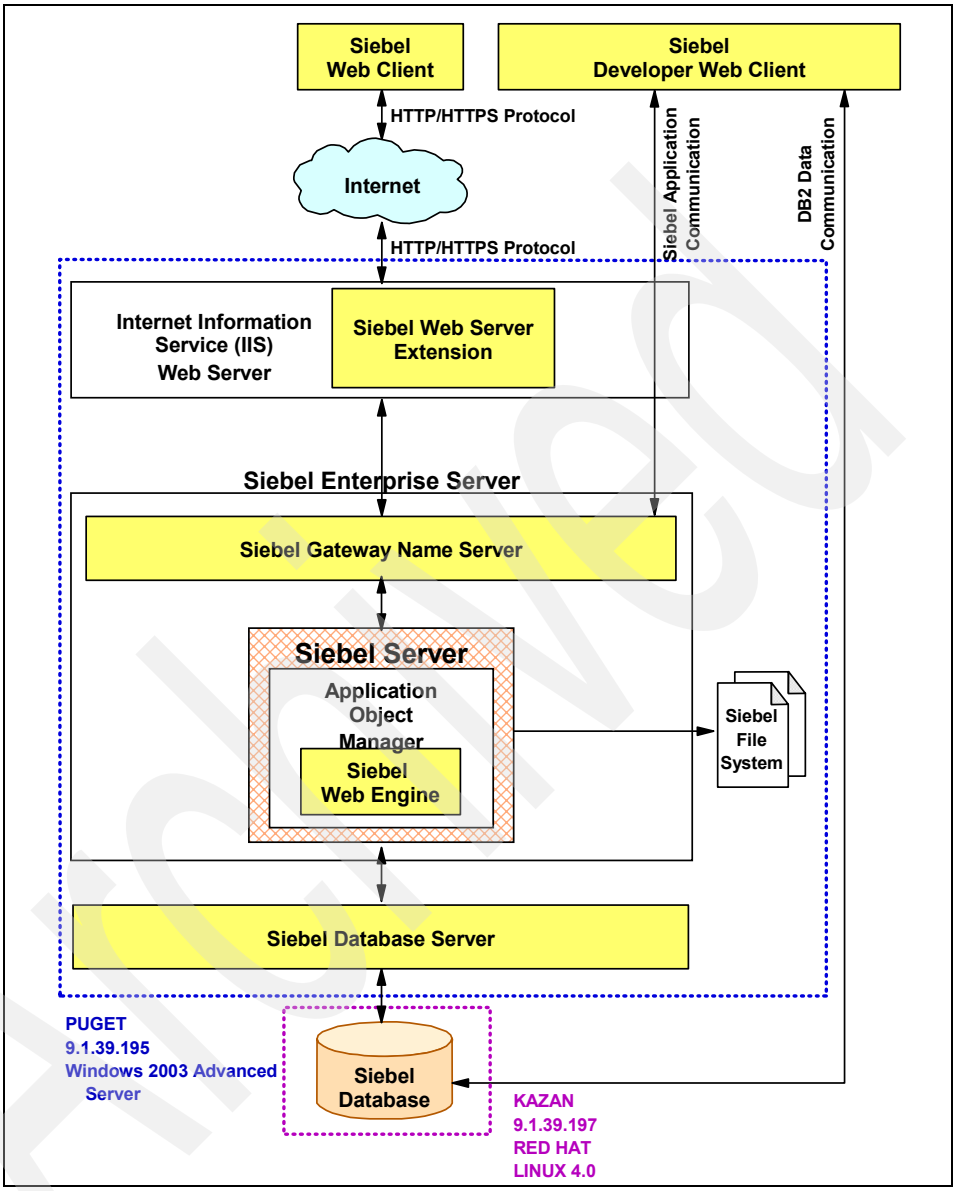

Figure 4-14 Siebel 7.8.2 installation on Windows with DB2 on Linux

4.6 Siebel Developer Web Client

If the Database Server and Siebel Developer Web Client are installed on different servers, then on the Siebel Developer Web Client server, install DB2 UDB client to connect to the database server.

Installation of DB2 UDB Client on Windows

Refer to the IBM DB2 UDB manual, *Quick Beginnings for DB2 Clients V8*, GC09-4832-01, for the actual installation instructions.

DB2 UDB installations of the client must have FixPak 8s installed prior to the installation of Siebel code.

Before installing DB2 client, make sure the DB2 Administration server user account (db2admin) is created with install privileges.

In our lab, we installed DB2 UDB Client V8.2 with Siebel specific FixPak fp8s on `WISLA:`

1. To invoke the DB2 Setup Wizard, execute **`setup.exe`** from the DB2 install directory or the DB2 Client Install CD.
2. On *Welcome to the DB2 Setup Wizard for DB2 Run-Time Client* dialog, click **Next**.
3. Read the License Agreement dialog, choose *I accept the terms of the license agreement* and click **Next**.
4. In *Select the Installation type* dialog (Figure 4-15 on page 102), Select **Typical** and **Next**.

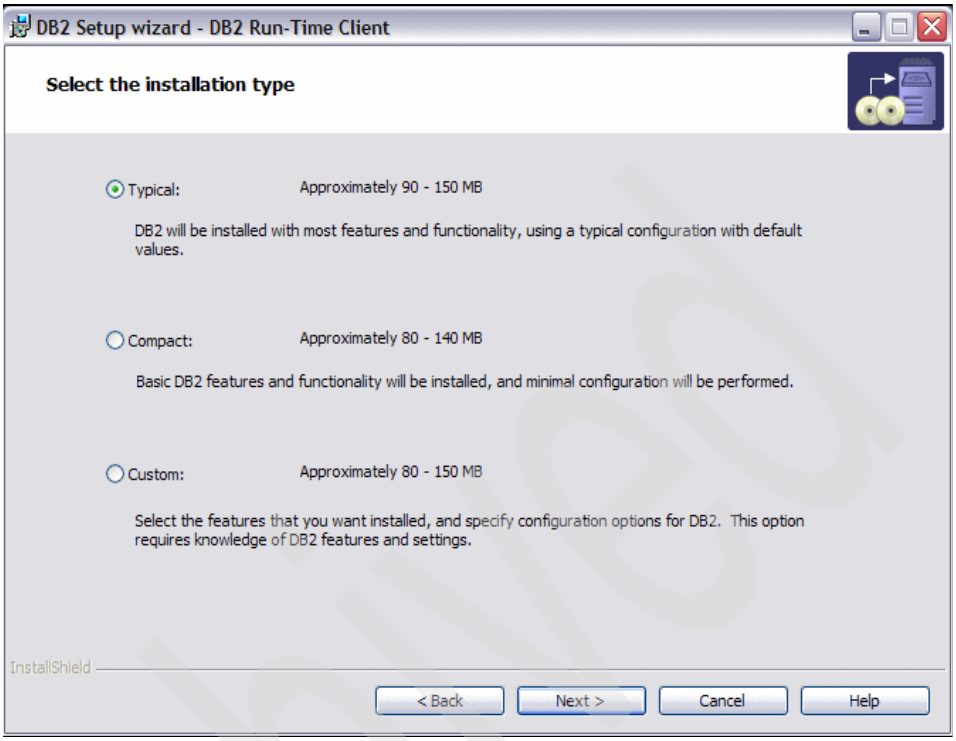

Figure 4-15 Select DB2 Run-Time Client installation type window

5. In *Select installation folder* dialog (Figure 4-16 on page 103). Select the drive to install. Click **Next**.

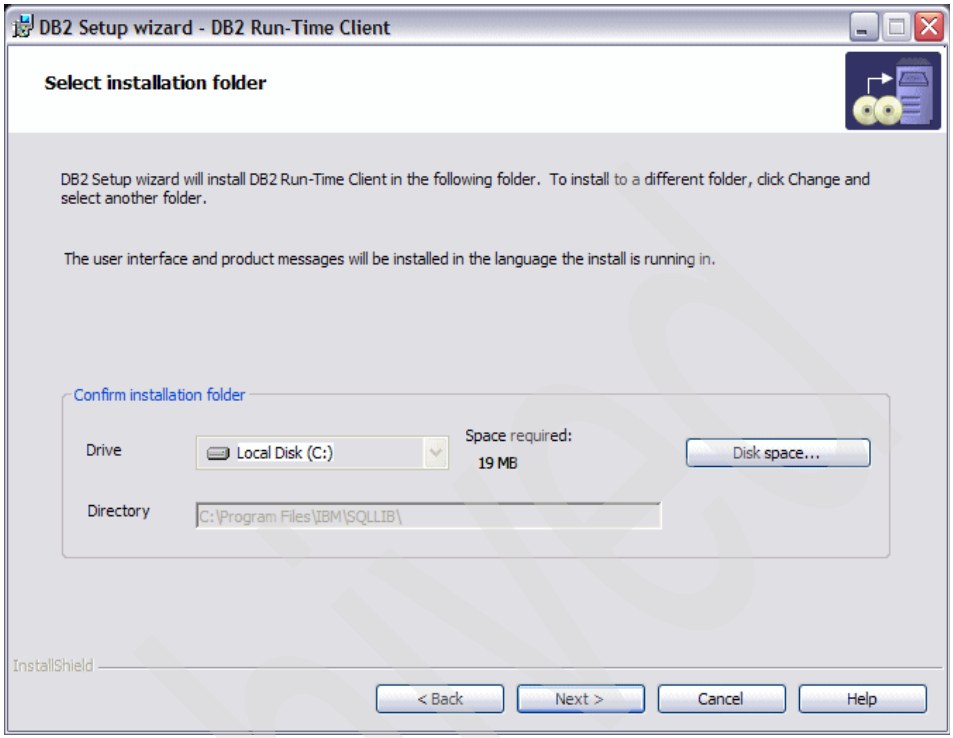

Figure 4-16 Select DB2 Run-Time Client installation folder window

6. The *Start Copying files* dialog displays the list of files that will be copied. To review or change any settings, click **Back** button make the changes and come back to the same dialog. If the settings are OK, click **Install** to start copying files.
7. In *Setup is complete* dialog (Figure 4-17 on page 104), click **Finish** to complete the installation process.

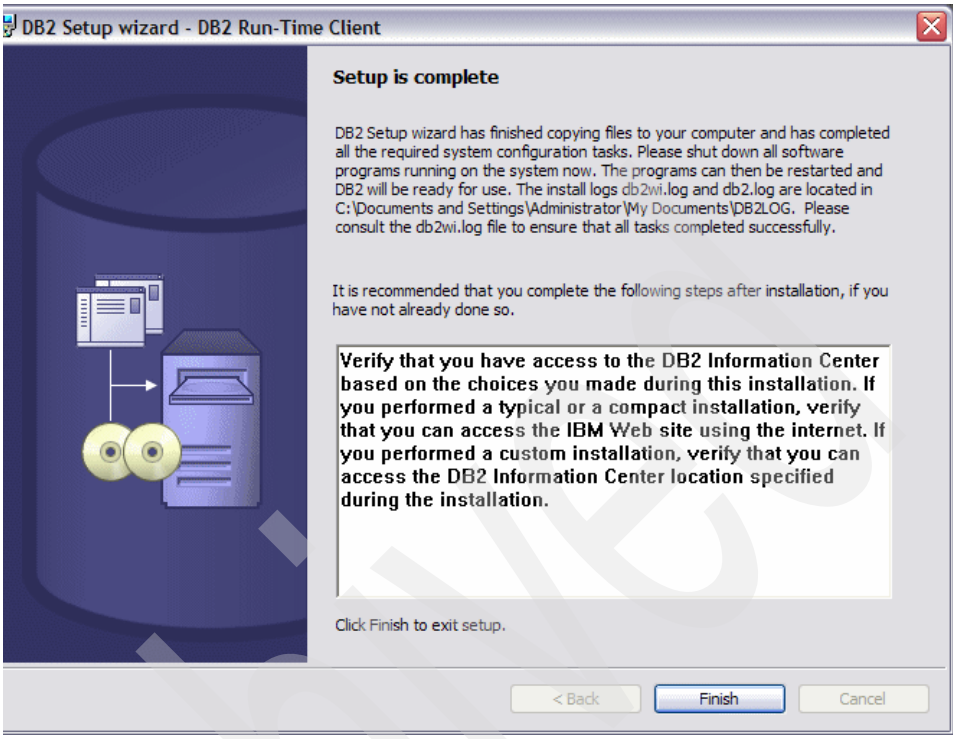

Figure 4-17 DB2 Run-Time Client Setup is complete window

Installing Siebel Developer Web Client

Follow these steps to install Siebel Developer Web Client.

1. Go to the directory where the installer is located. In our lab,

   `C:\Siebel_Install_Image_7.8.2_Win\7.8.2.0\Windows\Client\Siebel_Web_Client\`

   Double-click on **`setup.exe`**.

2. The Choose Setup Language window (select the language for the installation) appears. We selected **English**.

3. The Welcome to Siebel Business Applications Client setup window appears. Click **Next**.

4. Choose the installation setup type and specify the directory for the installation. We selected **Typical** (default).

5. On Select Languages to install window, we selected **English (American)** (the default).

   This window also provides the information about the disk space requirement for installation.

6. Select the type of client to install, we selected **Developer Web Client**.
7. Select the server database:

   ```
   IBM DB2 UDB for Windows and UNIX
   ```

8. Specify the File System Directory Path:

   ```
   C:\siebelfs
   ```

9. The Siebel Remote Server window appears, click **Next**.
10. Specify the database identification:

    ```
    Database Alias: QA78RBW
    Table Owner: SIEBEL
    ```

11. Specify the Gateway Name Server address and the Enterprise Server Name for connectivity to server components:

    ```
    Gateway Name Server Address: PUGET
    Enterprise Server Name: Siebwin
    ```

12. Specify the name of the Siebel Server on which the Server Request Broker (SRB) component is running:

    ```
    Request Server Name: PUGET
    ```

13. Specify the parameters for the server on which Siebel search is operating. We accepted default values.
14. Specify the program folder which will contain the Siebel shortcuts. We selected **Siebel Web Client 7.8**.
15. In Predeploying ActiveX Controls dialog, the following message was shown:

    ```
    After the installation, the file predeploy.htm is loaded in a browser
    window. This file downloads ActiveX controls used by the Siebel client.
    The file is located in the directory SIEBEL_CLIENT_ROOT\public\language

    where SIEBEL_CLIENT_ROOT= the directory where the client is installed.

    When the page displays the following message, indicating that ActiveX
    downloading is finished, close the browser window:

    The download is complete, you may close the window.
    ```

    For more information about the ActiveX controls used by the Siebel client, related browser settings, and the predeploy.htm file, refer to the *Siebel System Administration Guide* in Siebel Bookshelf.
16. The Event Log window appears, we select **Review the details,** and click **Next**.
17. The Registry Log window appears, we select **Review the registry details,** and click **Next**.

18. In the Installation setup complete window, click **Finish**.

This completes the install of the Siebel developer client. Siebel shortcuts are created in the program folder `siebel782`. Review the log file to ensure that all the components are installed successfully.

4.7 Installation troubleshooting

Both Siebel and DB2 UDB provide logs which record installation progress and results. The installation logs are a good starting point for installation troubleshooting. In addition to log files, you can use Siebel Environment Verification Tool (EVT) to check the Siebel systems after installation.

4.7.1 Environment Verification Tool

- After installing your Siebel gateway, application servers, and database servers, run Environment Verification Tool (EVT). Siebel includes this tool, EVT, as a way to help system administrators verify the configuration of the Siebel Business Applications environment. For more details, see *Siebel Installation Guide for Microsoft Windows: Servers, Mobile Web Clients, Tools,* section “Verifying Your Server Environment” in the Siebel support Web site:

  http://supportweb.siebel.com

You can find the parameters for EVT by typing "**`evt -h`**". When you pass in the "`-d EXPLAIN`" parameter, you will get a short description of the parameter check's purpose, so you can evaluate whether or not if you need to have that value set.

Before running EVT, make sure the `siebel server/bin` directory is in your path. EVT is case sensitive, even on Windows, so type all commands carefully. Finally, note the change needed to the *evt.ini* file for the database check.

Here are sample commands that you can run on Windows for each Siebel architectural level:

To verify the gateway:

```
evt -f evt.ini -o HTML -t GTWYNS -d EXPLAIN > gtwyns_output.htm
```

To verify the Siebel Web plugins:

```
evt -f evt.ini -o HTML -t SWSE -d EXPLAIN > swse_output.htm
```

To verify the Siebel server:

```
evt -f evt.ini -o HTML -t SIEBSRVR -d EXPLAIN > siebsrvr_output.htm
```

To verify the database server:

```
evt -f evt.ini -o HTML -t DBSERVER -d EXPLAIN > dbs_output.htm
```

To have the DB2 software checked on your Siebel server and database server, modify or add values to these lines in *evt.ini* file.

```
[OverrideNameServerInfo]
 DatabasePlatform=Db2Udb
 DatabaseConnStr=<your database connect string>
 TableOwner=<your tableonwer>
 sqlDatabase=
 sqlServer=
 ProcessSection=TRUE
```

When you run EVT, you will be prompted to enter a login ID and password. You should specify a database administrator level username and password, so you can perform the database checks.

4.7.2 Error messages

In our lab, during the installations, we encountered a few Siebel errors. The best approach to troubleshooting is to first look for error messages in the log files and then debug.

We list the issues (problem statements) and the resolution steps we took below:

- **Issue:** You cannot launch the application, `page not found` is displayed.

  Solution: Make sure the version and patches applied on the servers are the ones listed in the System Requirements and Supported Platforms guide.

  – Verify that the Web Server is running.
  – Make sure the port number you used is correct.

- **Issue:** HTTP 500 - Internal Server error.

  Solution:

  – Make sure the `eapps.cfg` has the correct parameters and is available.
  – Make sure the Web Server Administrator has proper access privileges on the SWSE/public directory.

- **Issue:** The server you are accessing is either busy or experiencing difficulties.

  Solution: Verify the following tasks:

  – Web Server and the Siebel Servers are up and running.
  – Parameters in eapps.cfg are correct.
  – Permissions on SWEApps directory is given to all users.

- **Issue:** Login failed for Login name: `guest1`

  Solution: Make sure that the `guest1` user has the Anonymous User responsibility.

- **Issue:** You cannot connect using server manager. An error that 0 out of 1 servers were found.

  Solution: Check whether or not you can connect to the database using the ODBC command.

5

Data in a Siebel Enterprise

This chapter discusses the following topics:

- Governing model for Siebel standard data
- Methods available for loading data
- Maintaining data quality
- Archival of obsolete data

5.1 Governing model for Siebel data

Data integrity is very important for Siebel implementations. Information about customers, opportunities, service requests, and other business objects must be accurate and up-to-date. High quality data is a prerequisite in providing excellence in customer service.

The Siebel data model employs object-oriented architecture features. Data is highly normalized, keeping data redundancy to a minimum. Each record is identified with an automatically generated value, ROW_ID, which is permanent and not dependent in any way on data stored in the record. Data within a record can be changed as needed without impacting relationships between records.

Referential integrity among objects is enforced by the Siebel application. Specialized tools and features are provided for loading and maintaining data to preserve referential relationships.

Use Siebel Data Quality (SDQ) for data matching, data cleansing, and data enrichment. Address correction, standard capitalization and standardized formatting of information are supported. Some features of SDQ require licensing from third-party vendors.

5.2 Data loading methods

Siebel provides a powerful suite of tools to manage loading data into the Siebel Enterprise. The Siebel application establishes and maintains referential relationships, regardless of the original source of the data.

5.2.1 Enterprise Integration Manager

Siebel Enterprise Integration Manager (EIM) is a large volume batch interface process that allows bidirectional exchange of data between Siebel applications and other applications. EIM provides organizations with ongoing batch interface requirements a mechanism to:

- Import high volume data scheduled to run at night or during other slow periods, thereby avoiding network slowdowns or interference with other applications. EIM is the method of choice for data conversions and other large-scale data movement activities.
- Export data from base tables for transfer to external systems. Export can also be used to move incorrectly formatted data from Siebel base tables to EIM tables to be modified and reimported.

- EIM can be used to delete obsolete data from Siebel base tables. EIM will maintain referential integrity by removing all child records and intersection tables.
- Two or more database rows can be merged into a single row. This could be required due to corporate takeovers or identification of duplicate data entry for a given account, contact, or other record.

Siebel does not support loading, deleting, or modifying data directly within physical tables due to the risk of loss of referential integrity. EIM maintains referential integrity by resolving the foreign key relationships during the EIM process.

EIM architecture and configuration

EIM reads a configuration file that specifies the EIM process to perform (import, update, merge, delete, or export) using the appropriate parameters. The EIM configuration file (the default file is `default.ifb`) is an ASCII text file of extension type `.IFB`, commonly referred to as the IFB file, that resides in the Siebel `Server/admin` directory. Before you can run an EIM process, you must edit the contents of the EIM configuration file to define the processes for EIM to perform. Note that if you use Unicode for your implementation, then you must save the IFB file as a Unicode text file.

EIM can be used to perform bulk imports, exports, updates, and deletes through intermediate database tables called EIM tables. The process flow between the Siebel database and external databases is illustrated in Figure 5-1:

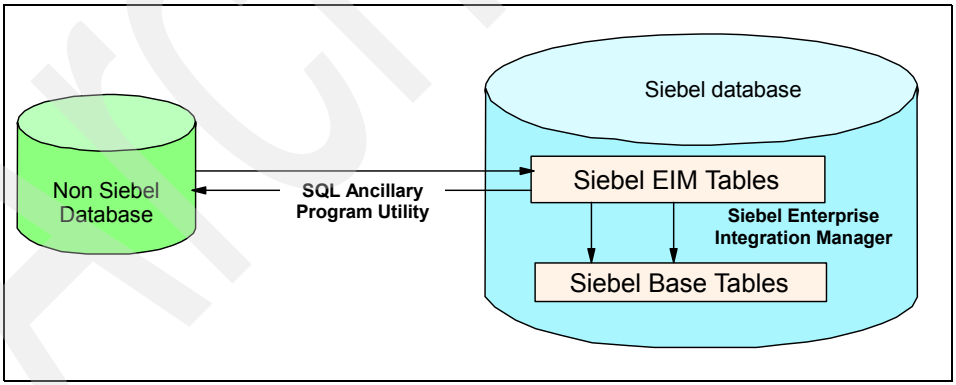

Figure 5-1 EIM process flows[1]

[1] Siebel Systems, Reprinted by Permission

EIM is one of the server components in EAI Component Group that needs to be enabled if it is to be used. Refer to *Siebel Enterprise Integration Manager Administration Guide* in Siebel Bookshelf for detailed information.

EIM data quality

We recommend that data is preprocessed using techniques such as `ForceCase`, Data Cleansing, and Data Matching (deduplication) before loaded into the Siebel base tables.

- Case values can be defined for base columns in Siebel Tools. EIM will adjust the case value of an EIM column according to the `ForceCase` property of the corresponding base table.
- Data cleansing to standardize the structure of data for example to standardize and correct the accounts, contacts, prospects, or business addresses.
- Data matching (decapitation) to be applied to identify the possible duplicate-record matches for account, contact, and prospect records. Ensuring that the data is of good quality is very important.
- Data enrichment to get information via third-party tools such as D&B.

EIM performance considerations

Throughput of EIM can be very important to meeting go-live dates and other time critical goals. There are many architectural and configuration steps you can take to either maximize EIM throughput or allow EIM to coexist with online users.

Things to consider for better EIM performance:

- Make sure the data is clean. If the data is not clean, you may end up running extra jobs/scripts to do the cleanup.
- Working with smaller batch sizes can lead to better performance in some circumstances. It is best to test your implementation with single-threaded EIM jobs with differing batch sizes to determine the optimal batch size. It is common to find that one batch size is not optimal for all sets of tables. For example, during testing you may find that loading Assets runs better at a larger batch size than Contacts. If tens of millions of records need to be brought in during a data conversion, these differences in throughput can add up to hours of elapsed time and finding the optimal batch size is a value-added effort.
- In the IFB, only reference and use the tables that you need, unnecessary listing of tables can lead to spending CPU resources that could be better used on other processing. Use the `ONLY BASE TABLES` and `IGNORE BASE TABLES` parameters to include needed tables or exclude unneeded tables.
- Use the `ONLY BASE COLUMNS` and `IGNORE BASE COLUMNS` parameters to limit processing to just the columns required.

- Use separate IFB configurations and separate batches for update and insert activity. Segregating these two types of activity will reduce the quantity and complexity of the SQL generated. EIM will not have to try to determine if each record is an insert or an update before proceeding if the IFB is configured for either just insert or just update.
  - This is critical when the volume of data to manage is large. The simpler the SQL, the more throughput will be achieved for a given hardware configuration.
  - Isolation of inserts and updates will improve the concurrency of parallel batch processing. Running multiple insert-only batches does not present concurrency issues so long as there are no duplicate records in concurrent batches.
  - Use of the `INSERT ROWS` and `UPDATE ROWS` parameters can limit a given IFB to insert or update activity. For example, if `INSERT ROWS = FALSE` and `UPDATE ROWS = TRUE`, then only update processing can occur.
- Do not allow the IFB to invoke runstats. Doing so will prevent running EIM jobs in parallel and waste resources repeatedly gathering statistics for the same objects. A better approach is to run one batch first of each type of EIM processing to be used, run runstats on the EIM table, and then the rest of the batches. Initially some columns of the EIM table are empty, after the first batch, these columns are populated. By running runstats on the EIM table, DB2 will use the latest runstats which show that certain columns are populated. This will result in a better plan of action from DB2.
- For initial conversions, temporarily remove unused indexes from both target and EIM tables that are not needed for performance or uniqueness. DB2 Explain can be used to determine which indexes are not required during EIM processing. In most cases, only the unique indexes and a handful of delivered indexes are required at this point. Each index is an object that the database engine must maintain. For large loads, it is more efficient to drop these indexes, perform the conversion, and rebuild the indexes in parallel. To ensure integrity, unique indexes must not be removed at any time.
- For initial loads or for implementations that do not have remote users, ensure that transaction logging is disabled. If this is a new implementation, all remote users will require an initial extract. Having remote users enabled will slow down processing without adding value.
- Running EIM jobs in parallel will greatly increase throughput. It is easier to run insert processing in parallel due to the records being unique. If updates are to be processed, duplicate updates to the same record running in different batches could lead to deadlocks, timeouts, and the possibility of a newer update being processed before an older update, which would leave the record in an incorrect state.

- If you delete old data out of EIM table before you load data into it, use load/import with replace option. This will load the data faster into the EIM table. If you keep old data around, then reorganizing the EIM table before a load on the unique index will enhance the performance of an EIM job.
- Using `COMMIT EACH TABLE` and/or `COMMIT EACH PASS` will reduce locking overhead. EIM jobs are restartable in the event of a failure. Note that these two parameters should be `FALSE` for delete processing to prevent the possibility of creating orphaned records.

Example 5-1 shows a portion of the IFB file.

Example 5-1 IFB file content

```
[IMPORT EIM_ORDER]
   USING SYNMONYMS = FALSE
   TYPE = IMPORT
   BATCH = 99
   TABLE = CIM_order
   INSERT ROWS = TRUE
   UPDATE ROWS = FALSE
   COMMIT EACH PASS = TRUE
   COMMIT EACH TABLE = TRUE
   ROLLBACK ON ERROR = TRUE
   LOG TRANSACTIONS = FALSE
ONLY BASE TABLES = S_ORDER
ONLY BASE COLUMNS = S_ORDER.BU_ID \,
and the remaining required columns
```

Example 5-1 shows that many of these recommendations were followed:

- The IFB is insert-only due to `INSERT ROWS = TRUE` and `UPDATE ROWS = FALSE`.
- `RUNSTATS` will not be run by the IFB as `UPDATE STATISTICS = FALSE`.
- Siebel Remote tables will not be populated because `LOG TRANSACTIONS = FALSE`.
- Commits are being performed to reduce locking overhead.
- Activity has been limited to only the required table and columns.

By having EIM do only work that is really necessary and correctly managing the database schema, you can attain a high volume of throughput.

5.2.2 Enterprise Application Integration

Organizations face an ever increasing need to integrate and streamline business processes. Data stores are becoming more interdependent and must be kept

up-to-date with changes from many sources. The Siebel Enterprise Application Integration (EAI) and Siebel Universal Application Network (UAN) have been developed to meet the challenges of enterprise data integration. UAN is part of the solutions included in Customer Data Integration (CDI).

UAN leverages the integration framework provided by EAI. EAI is the integrated toolset provided to support integration of Siebel Enterprise Applications with other external systems. EAI provides components for integration with external applications and technologies, such as ERP, legacy, or even another Siebel Application system. EAI works with third-party solutions such as IBM MQSeries® and other vendor products.

The architecture is designed to seamlessly integrate with third-party software:

- It supports real-time and batch integration.
- It enables you to build a consistent view of all data from different applications.
- It provides configurable components and a standard set of interfaces.
- It provides connectors and adapters to develop an integration with other technologies.
- It provides prebuilt business processes to connect to back office applications such as SAP applications.
- It provides support for XML, COM, Java, and HTTP-based standards.

The architecture of Siebel EAI is shown in Figure 5-2 on page 116.

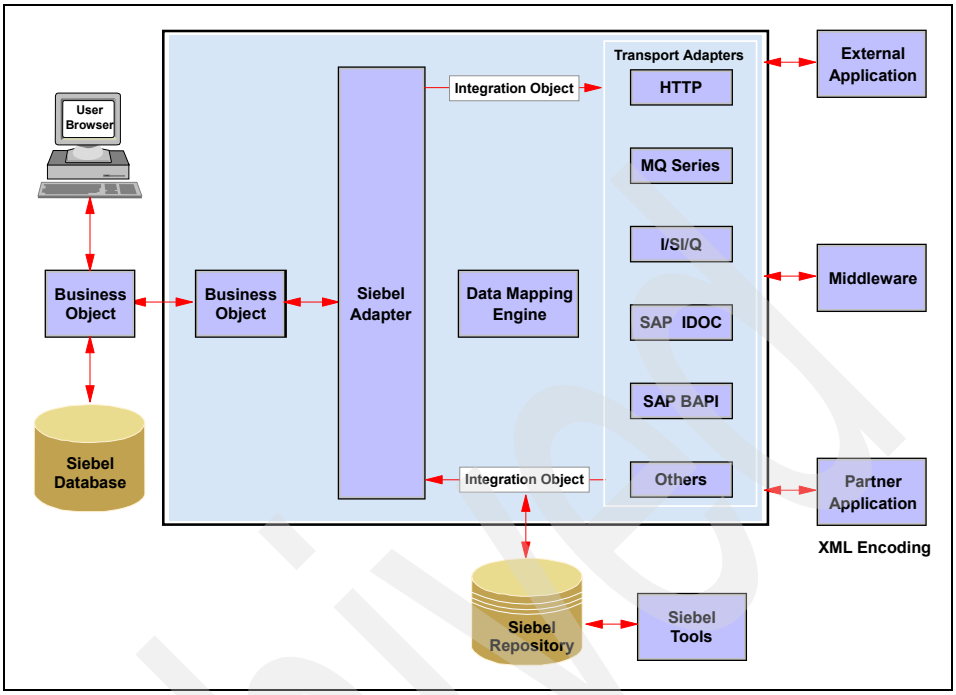

Figure 5-2 Siebel EAI architecture[2]

The Exchange of data using EAI can be any one of the following types:

- **Data sharing**: Using Siebel Business Application to view the data from external systems and not replicate it in Siebel database.
- **Replication:** Replication of data would mean data in both the systems. Replication of data can be real-time or batch replication.
- **User Interface (UI) layer integration**: UI in an external system which can be based on data stored in Siebel database. This method is used to keep data in two or more systems, one of which is Siebel Enterprise Application, in-sync.
- **Programming Interface**: External COM or Java-based programs can programmatically call Siebel objects to view, insert, update, or delete Siebel data.

Depending on the data exchange requirement, you must select the appropriate type. The incoming data has to be of good quality. You must enable the EAI Component Group for using EAI.

[2] Siebel Systems, Reprinted by Permission.

Refer to Siebel documentation *Overview: Siebel Enterprise Application Integration* for detailed information on the data import process.

There are several techniques available to improve the performance of EAI:

- Use as little scripting as possible. Place required scripting either before or after the EAI Siebel Adapter call rather than within it.
- Disable logging when logging is not required.
- Keep integration object sizes as small as possible. For example, inactivate unnecessary fields or integration components.
- Analyze SQL for possible improvements. One approach to finding slower SQL statements is to enable logging by setting the event SQL to level 4 in the component running EAI, then check the logs for slow statements. You can also enable component event logging for workflow process execution and individual EAI business services. By enabling these events, you can get a high level overview of where performance is slow. The event names in Siebel are:
  - WFPerf: Workflow performance for individual steps
  - EAITransportPerf: Any EAI transport performance (that is, HTTP, MQSeries, JMS, File, and so on)
  - EAISiebAdptPerf: EAI Siebel Adapter performance
  - WebSvcPerf: Web Services performance

  Another alternative is using the excellent tools provided in DB2 UDB V8. See Chapter 7, “Database server monitoring” on page 167, for details about DB2 UDB monitoring.

5.2.3 Siebel programming interfaces

Siebel object interfaces allow external applications to access Siebel business objects, thereby supporting integration between Siebel applications and external applications through standard protocols such as:

- Microsoft® Component Object Model (COM)
- Java interfaces
- Built-in scripts

These interfaces form a source to exchange or update small amounts of data between a third-party application and a Siebel Enterprise.

COM Interfaces

You can access Siebel COM object interfaces in four ways:

- COM Data Server

- COM Data Control
- Siebel Web Client Automation Server
- Siebel Mobile Web Client Automation Server

Java Interfaces

The Siebel Java Data Bean provides an interface to an external application through Siebel Application Object Manager. The Siebel Java Data Bean is a set of libraries which users can use to build applets, servlets, JSPs, and Java applications to interface with Siebel objects.

Built-in scripts

You can use programming languages such as Siebel VB, Siebel eScript, and Browser Script to access Siebel methods and events to access Siebel objects. Refer to Siebel documentation *Siebel Object Interfaces Reference* for more details.

It is important to ensure that the data imported through interfaces is quality data. Ensure the proper usage of the Siebel Objects and Methods, optimizing algorithms when using Siebel Interfaces. Ensure proper separation of code between configuration, Siebel VB/eScript/JavaScript, and external code which uses Siebel programming interfaces.

5.2.4 Siebel Web Client

Users use the Siebel Web Client for data entry by using one of the following methods:

- **Create Record:** Users can create records through different methods such as clicking a button, choosing an option from a menu, or using a short cut.
- **Importing data:** Data can be imported on certain windows of the user interface. To import data, navigate to the windows on which to import. Click the menu button, and then click **Import**. In the Import dialog window, you can:
  - Click **Browse** and select the input file to import.
  - Select the Input Format: Select either *Comma Separated Text File* or *Tab Delimited Text File*.
  - Select the Input Source: Select either *Auto Mapping* or *Predefined Mapping*.
  - Specify the Conflict resolution: You have three options:
    - Reject Import Record
    - Overwrite Existing Record
    - Create Additional Record
  - You also need to select the mapping method:

- If you select *Pre-defined Mapping*, the tool will use the default mappings.
- If you select *Auto Mapping*, the tool will display the field mapping information for you to verify and then proceed with the import.

When the import is finished, a status dialog box appears. You can click **OK** to close the dialog or click **View Log** to review information about the import.

The data captured by either of the methods for a particular entity could span across multiple tables in a Siebel database. Therefore, it is important to input good quality data. You can write a routine to validate/cleanse the data and then import the data into Siebel.

5.3 Data quality maintenance

Data quality can be compromised in many ways. Misspelled names, leading blanks, and special characters are several of the more common problems. Errors such as these can originate from poor data entry standards on the Siebel user interface or from records originating in other systems.

Poor data quality can reduce return-on-investment in several ways.

- Difficulty in finding customer records can lead to longer call durations as the operator tries to find an existing customer record. This can create frustration for both the call center employee and the customer.
- Employee responses to data quality issues can create performance and scalability problems. Search methods to find elusive records can involve the use of leading wildcard, case insensitive searches, repeated searches, or retrieval of unusually large numbers of records and scrolling through them looking for the right entry.

Duplicate records can be created. If records for an existing customer are not found, new records for that customer may be created. This may effectively lose some customer history depending on which record the next call center agent happens to retrieve.

5.3.1 Identifying data quality problems

Database administrators are not commonly directly involved with detection, prevention, or resolution of data quality issues. There is one exception. User attempts to find the correct customer record can create situations where the DB2 team will be engaged due to performance and scalability issues.

A common but often unexpected indication of poor data quality are performance problems. These can manifest as user complaints, the IT department noticing a decline in scalability or the discovery of poorly performing queries on the database server. Database administrators will become involved to try to solve these symptoms.

The database administrator may notice a few of the following occurring:

- The same SQL runs sub-second most of the time, but sometimes very slowly.
- Wildcards are used for many queries resulting in inferior access plans. Note that performance problems with trailing wildcards can be resolved with a Siebel user property, Use Literals For Like. However, this user property will not help with leading wildcards because an index match cannot be made. For details about the Use Literals For Like user property, please consult the *Siebel Developer's Reference Version 7.8* in Siebel Bookshelf.
- Case insensitive SQL is found when the feature is disabled globally for the implementation.

Examples to illustrate each of these symptoms follow:

- The database administrator finds periodically very slow SQL statements that consume large quantities of CPU with high I/O impact but not every time the SQL runs:
  - Running DB2 application snapshots shows the same SQL statement running either sub-second or for an extended period of time, sometimes minutes.
  - In discussion with user management, it is found that users typically entered most of the search string and appended a trailing wildcard. The appropriate Use Literals For Like user property can be enabled and this will usually work well.
  - It is found that some records could only be found by putting a wildcard in the first character.
  - An examination of the data may show the presence of a subset of records where the search string formatted with leading spaces. The damaged records can be exported by EIM, modified in the EIM tables, and reimported. The source of the incorrectly formatted records should be determined. They could come from an external system or the source could be a training issue.
- Case insensitive SQL is a common user recourse to data quality issues. Users can create case insensitive SQL, regardless of configuration settings when running queries if they have been taught how to do so.

– Depending on the size of the table being read and the presence or absence of other supporting query fields, performance can vary considerably. Database administrators may see SQL similar to this:

```
WHERE (LAST_NAME LIKE ? OR LAST_NAME LIKE ? AND UCASE(LAST_NAME) =
UCASE(CAST(? AS VARCHAR(4000))))
```

– There may be more or fewer “OR LIKE” predicates depending on the setting of the case insensitivity factor.

– Users notice that these queries are much slower than not using case insensitive techniques, especially on larger tables. Database administrators will usually detect these as well due to scalability issues. Compared to running queries that return no records or the wrong records, using case insensitivity may seem a better approach from the user’s perspective.

It is unreasonable to expect the user community to understand the scalability impacts of some of these search methods. Communication with the user community and management is the key to finding the underlying reasons for poorly formed queries. It is important to determine why these queries are being created in an inefficient manner. In many cases, these are training issues. In some cases, the poorly performing queries are the result of data quality problems.

5.3.2 Maintaining data quality

One method for maintaining data quality is to correctly format data before it is stored in the Siebel database. Preventing data quality problems is preferable to correcting them after the fact in many cases.

By entering data in a consistent format, you can prevent many problems. The correct record is easily found using efficient query methods.

Data quality problem prevention

Entering data with the correct format is the best approach to maintaining data quality. Siebel feature `ForceCase` field property and Siebel Data Quality Matching Server can be used for this purpose.

Use of ForceCase field property

This field property is a standard feature of Siebel products. For fields defined as `DTYPE_TEXT`, the field property `ForceCase` can be configured to help consistently format fields such as first and last name. The options are:

- ForceCase = “Upper”: When the user steps off of the field, all characters will automatically be converted to all upper case.

- ForceCase = “Lower”: Will convert all characters to lower case when the user steps off of the field.
- ForceCase = “FirstUpper”: Will convert the first character of each field to upper case after the user steps off of the field. This leaves the other letters unchanged.

This field property only works for letters A-Z (ASCII) and can help to standardize user interface input.

For large scale inserts, EIM adjusts the case value of an EIM column according to the `ForceCase` property of the corresponding base table column. For details on the use of `ForceCase` with EIM, please consult the *Siebel Enterprise Integration Manager Administration Guide* in Siebel Bookshelf.

Siebel Data Quality in real-time mode

The Siebel Data Quality (SDQ) Matching Server is available for deduplication of account, contact, and prospect data. The Matching Server can run in real-time mode to detect duplicates as data is entered to prevent data quality from becoming compromised.

The SDQ Universal Connector integrates Siebel with third-party vendors for data cleansing and standardization in real time as data is entered. Siebel partners with leading vendors in this space. Details are available on Web site:

http://www.siebel.com

Note that real-time data cleansing may require more costly licenses than batch methods. Cost-benefit analysis should be undertaken to determine the appropriate approach for a given situation.

The ability to detect and manage data quality problems as data is entered provides tangible business benefits by:

- Prevention of the creation of duplicate records avoids time consuming research after the fact to determine which records are duplicates and how they should be merged.
- Integration to data quality vendors allows centralized administration of data quality standards, preventing the addition of invalid or incorrectly formatted data into the database.
- For Siebel implementations that are integrated with external systems, prevention of incorrect data entry is even more important. Inappropriate or inaccurate data can be propagated quickly to numerous external systems making subsequent correction efforts much more difficult and time consuming.

Data cleansing

Once data quality has been compromised, you undertake data cleansing to correct and standardize the data.

Using EIM and direct SQL

Running SQL updates against the Siebel tables to make corrections to data that has already been added to the Siebel database is only supported when the data is contained in EIM tables. Incorrect records could be exported to the EIM tables, updated as needed, and reimported.

Direct SQL against base tables is not supported to prevent changes that could adversely affect referential relationships. You can make exceptions to this rule by requesting a Non-Standard Change Request.

Siebel Data Quality in batch mode

Both Siebel Data Quality (SDQ) Matching Server and SDQ Universal Connector can be run in batch mode to analyze data which has already been entered into the database.

As data volumes increase, the overhead of cleansing or deduplicating the entire database will become unacceptable. We recommend that you use date ranges or other criteria to limit the number of records checked in each batch run and avoid rechecking previously cleansed records.

Consider table space disk capacity when implementing SDQ for deduplication:

- The table `S_DEDUP_RESULT` will be populated with as many as six times the number of rows that are being deduplicated. The `S_DEDUP_RESULT` table must be manually emptied or it will continue to grow. This is not considered a base transactional table and direct SQL is permitted.
- The tables `S_PER_DEDUP_KEY`, `S_ORG_DEDUP_KEY`, and `S_PRSP_DEDUPKEY` will also be about six times larger than the number of rows being deduplicated.
- Placing these tables in a dedicated table space will prevent problems with other tables if the table space containing the tables becomes full. A dedicated buffer pool will prevent flushing needed records from the transactional buffer pools.

For information about the deployment and administration of third-party data cleansing products, please consult the vendor's documentation. For more details about SDQ, please consult the *Siebel Data Quality Administration Guide* in Siebel Bookshelf. Information on data quality product partners can be found at:

http://www.siebel.com

Siebel File System

`sfscleanup` command line utility is provided to clean up the Siebel File System. `sfscleanup` utility checks to make sure every file in the Siebel File System is a valid file. Siebel File System can contain orphaned, ancient, and invalid files which will be removed by the `sfscleanup` utility. *Orphaned* files are created when a user deletes a parent record, and its child records are not deleted. *Ancient* files are created when the database has an associated record, but the version number is different. Invalid files are files that are not Siebel attachments.

Run the `sfscleanup` utility during off peak hours. Cleaning of the Siebel File System is a two step process, creating a report and then cleaning up the Siebel File System. The `sfscleanup` utility is located in the `bin` directory under Siebel server root directory. The `sfscleanup` command has the following flags:

/u	User ID
/p	User ID password
/f	Location of the Siebel File System
/c	OBDC data source
/d	Table owner
/x	Name and location of the output file
/m	Directory to which discarded files are moved
/n	Y (old versions are removed)
/g	Y (remove non-Siebel attachments)
/r	Y to generate a report file

Example 5-2 shows a sample report generated by `sfscleanup`.

Example 5-2 Sample sfscleanup report

```
Kanaga ..>$ ./sfscleanup /u sadmin /p sadmin /d siebel /c siebsrvr_siebaix /f
/siebelfs /x /tmp/sfscleanup.log /r Y
Connecting to the database...
Opening cached file iterator...
Found attachments for S_ACTIVITY_ATT
Found attachments for S_ACCNT_ATT
Found attachments for S_AUDIT_TRAIL
Found attachments for S_COLLAB_CMD
Found attachments for S_LIT
Found attachments for S_DCP_PKG_ITEM
Found attachments for S_DOC_CORR
Found attachments for S_DOC_PPSL
Found attachments for S
Found attachments for S_OPTY_ATT
Found attachments for S_SRCH_MBLI_ATT
Found attachments for S_UPG_KIT
Found attachments for S_UPG_KIT_IARG
Found attachments for S_USERLIST_ATT
100 % done. Processing INVALIDS_ORPHANS...
```

```
606/606 files processed. 0 error(s) encountered.
        CURRENT:        210
         ORPHAN:        322
        ANCIENT:        74
        INVALID:        0
```

After you have analyzed the `sfscleanup` report, then invoke the `sfscleanup` command again with `/g` to remove garbage or non-Siebel files. Example 5-3 shows execution of `sfscleanup` with `/g` flag. 322 orphan files have been removed.

Example 5-3 sfscleanup with the /g flag set to Y

```
kanaga..>/siebel/siebsrvr/bin/sfscleanup /u sadmin /p sadmin /d siebel /c
siebsrvr_siebaix /f /siebelfs /x /tmp/sfscleanup.log /g Y
Connecting to the database...
Opening cached file iterator...
Found attachments for S_COLLAB_CMD
Found attachments for S_LIT
Found attachments for S_DCP_PKG_ITEM
Found attachments for S_DOC_CORR
Found attachments for S_DOC_PPSL
Found attachments for S_USERLIST_ATT
100 % done. Processing INVALIDS_ORPHANS...
284/284 files processed. 0 error(s) encountered.
        CURRENT:        210
         ORPHAN:        0
        ANCIENT:        74
        INVALID:        0
```

You invoke the `sfscleanup` with the `/n` flag to remove the old versions of Siebel attachments as shown in Example 5-4. 74 ancient files have been removed.

Example 5-4 sfscleanup with the /n flag set to Y

```
kanaga..>/siebel/siebsrvr/bin/sfscleanup /u sadmin /p sadmin /d siebel /c
siebsrvr_siebaix /f /siebelfs /x /tmp/sfscleanup.log /n Y
Connecting to the database...
Opening cached file iterator...
Found attachments for S_DCP_PKG_ITEM
Found attachments for S_DOC_CORR
Found attachments for S_DOC_PPSL
Found attachments for S_LIT
Found attachments for S_USERLIST_ATT
100 % done. Processing INVALIDS_ORPHANS...
210/210 files processed. 0 error(s) encountered.
        CURRENT:        210
         ORPHAN:        0
```

```
ANCIENT:        0
INVALID:        0
```

Refer to the *Siebel Administration Guide* in the Siebel Bookshelf for more information about `sfscleanup`.

5.4 Data archiving

Over time, large quantities of older service requests, orders, and obsolete data of many kinds will cause the Siebel database to contain a large amount of unused data. Removal of unnecessary data will improve the performance of database maintenance jobs, reduce the elapsed time for database recovery and disaster recovery, and help control data storage costs.

Accomplish archiving obsolete data through the use of EIM or by using third-party products.

Using EIM for data purging

Deletion of OLTP entries using EIM can be a viable solution for implementations where important business information from transactions has been placed into an external system such as the Siebel Data Warehouse.

This requires a two-step process:

1. High level records to be deleted, such as accounts, contacts, service requests, orders, and so forth are exported to the EIM tables. There is no cascade functionality in the export process.
2. EIM delete is then run against the exported records. EIM will perform a cascade delete which will remove the parent records and all their dependent records.

Using this two-step process requires a comprehensive understanding of the Siebel data model to ensure that orphaned records are not created as part of this process.

Third-party archival solutions

Siebel partners that provide archiving solutions include:

- Princeton Softech Active Archiving Solutions for data archiving and automated testing.

  More information can be found at the Web site:

  http://www.princetonsoftech.com/products/activearchivesolutions/c2_techcenter_services-siebelCRMedition.asp

- Applimation Informia Archive

  More information can be found at the Web site:

  http://www.applimation.com/data_management/archiving.asp

6

Database administration

In this chapter, we discuss several of the important DB2 UDB utilities used by database administrators. These utilities include database backup, recovery, table and index reorganization, statistics collection, and governor.

This chapter discusses the following utilities:

- Backup
- Recovery
- Reorg
- Runstats
- Governor

6.1 Backup

The single most important item you can possess, that will prevent catastrophic data loss in the event of storage media failure, data corruption, and user data modification, is a database backup.

A database backup is also required for various other reasons such as:

- Database migration/upgrade
- Application software upgrade
- Setup of production and test environments
- Database copies for developers
- High availability and disaster recovery
- Hardware upgrade

6.1.1 Logging concepts

Database transaction logs are crucial for recovery because they keep track of changes made to the database objects and data. DB2 UDB uses a write-ahead logging scheme in which it writes transactions to the logs before writing the changes to the database.

There are two types of logging for DB2 UDB databases:

- *Circular logging*
  In this implementation, logs are recycled as soon as they are no longer necessary for crash recovery. The default logging method, the primary log files record all the changes and are reused when the changes are committed and the primary log files are no longer necessary for crash recovery. When the limit of primary log files has been reached, the secondary log files are allocated. This type of logging supports only crash and version recovery.
- *Archive logging*
  Archive logging does not overwrite log files. Instead, it creates additional logs to record all transactions.This type of logging supports rollforward recovery. You need to enable archive logging to perform online and incremental backup. The archive logs are the files that contain the records associated with completed transactions (in other words, transactions that have been externalized to the database).

The new database parameter `LOGARCHMETH1` is now used to specify the archival method. The following command changes the logging method of database `QA78RBA` to archive logging:

```
db2 update db cfg for qa78rba using LOGARCHMETH1 LOGRETAIN
```

It is mandatory to make an offline backup of your database when you change from circular to archival logging.

> **Note:** `LOGRETAIN` and `USEREXIT` configuration parameters have been replaced by `LOGARCHMETH1` in DB2 UDB V8.2. `LOGRETAIN` and `USEREXIT` parameters are still supported for back-level compatibility.

6.1.2 Types of backup

DB2 UDB supports three types of backup:

- Offline backup
- Online backup
- Incremental backup

Offline backup

The database is offline (inaccessible to users) while a offline backup is performed. During an offline backup, an exclusive connection to the database is made by the DB2 agent performing the offline backup. Also, this backup is referred to as a *full database backup*. This method of backup is supported for both circular logging and archive logging.

Online backup

Online backups are performed while users continue to access the database. Therefore, there may be changes to the data between the time the backup starts and when it completes. Because of this, online backup can only be performed on databases where archive logging has been enabled.

When performing an online backup, you can choose to make a backup of the entire database, incremental backup, or selected table space backup. Only a full online backup of a database or table space supports point-in-time recovery. To perform a point-in-time recovery, you have to restore the online backup and then roll forward either to the end of logs or a point in time.

Incremental backups

There are two kinds of incremental backups:

- Incremental:
  An incremental backup image is a copy of all database data that has changed since the most recent, successful, full backup operation has been performed.

  Figure 6-1 on page 132 illustrates the incremental backup.

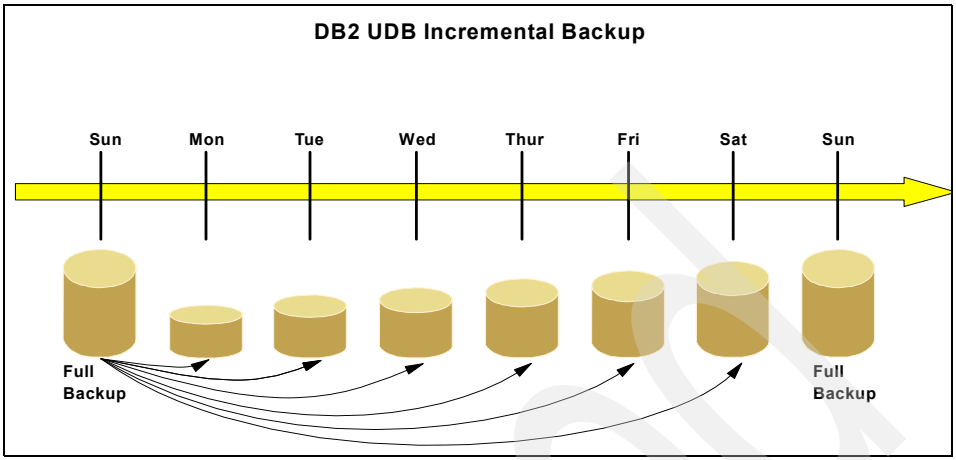

Figure 6-1 Incremental backup

- Delta:
  A delta, or incremental delta, backup image is a copy of all database data that has changed since the last successful backup (full, incremental, or delta). Figure 6-2 illustrates the delta backup.

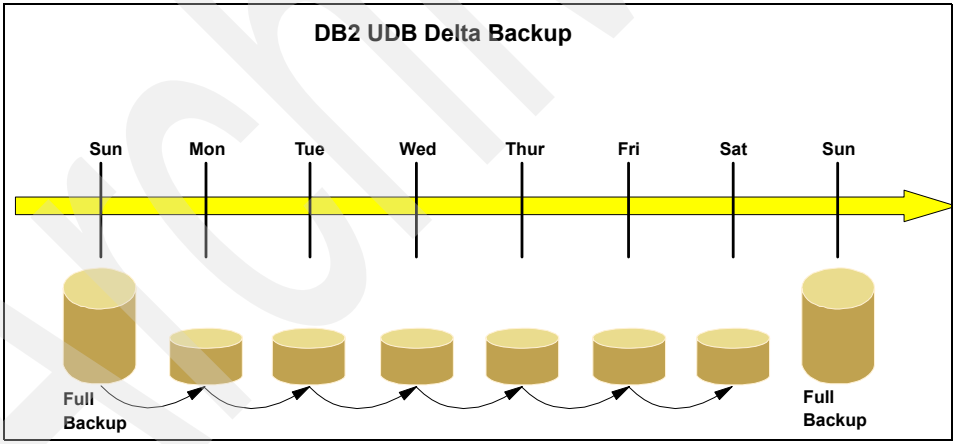

Figure 6-2 Delta backup

To perform an incremental or delta backup, you need to set the parameter `TRACKMOD` in database configuration to `ON`, as shown:

```
db2 update db cfg for qa78rba using TRACKMOD on
```

6.1.3 Backup command and authorities required

The `BACKUP DATABASE` command creates a backup copy of a database or a table space. The syntax for the backup command is:

```
BACKUP DATABASE database-alias [USER username [USING password]]
[TABLESPACE (tblspace-name [ {,tblspace-name} ... ])] [ONLINE]
[INCREMENTAL [DELTA]] [USE {TSM | XBSA} [OPEN num-sess SESSIONS]
[OPTIONS {options-string | options-filename}] | TO dir/dev
[ {,dir/dev} ... ] | LOAD lib-name [OPEN num-sess SESSIONS]
[OPTIONS {options-string | options-filename}]]
[WITH num-buff BUFFERS] [BUFFER buffer-size] [PARALLELISM n]
[COMPRESS [COMPRLIB lib-name [EXCLUDE]] [COMPROPTS options-string]]
[UTIL_IMPACT_PRIORITY [priority]] [{INCLUDE | EXCLUDE} LOGS] [WITHOUT
PROMPTING]
```

The simplest form of the database backup command is shown in Example 6-1.

Example 6-1 Offline database backup command

```
db2 backup database QA78RBA to /dbbkup/qa78rba/ucsdb2
Backup successful. The timestamp for this backup image is : 20050710131120
```

The backup can also be performed using the Control Center.

To open a Control Center:

- In Windows, click **Start** → **Programs** → **IBM DB2** → **General Administration Tools** → **Control Center**.
- In Linux, open the IBM DB2 folder on the desktop and click **Control Center**.
- In UNIX, enter the `db2cc` command from a terminal session.

Right-click on the database and select **Backup**.

Back up of database to disk provides the fastest I/O. However, disks are expensive for backup storage. Running the database backup to multiple disks/ devices at the same time will decrease backup time.

Authorities required for database backup

Performing backup requires one of the following authorizations:

- System Administrator (SYSADM)
- System Control (SYSCTRL)
- System Maintenance (SYSMAINT)

6.1.4 Performing backup on Siebel File System

Whenever a backup of Siebel database is performed (full, incremental, or delta), the Siebel File System also needs to be backed up using operating system tools.

An example of a backup for Siebel File System to a tape device on AIX is as follows:

```
/usr/sbin/backup -f '/dev/rmt0' -'0' /siebelfs
```

Note: You also can use IBM Tivoli® Storage Manager for System Backup and Recovery (also known as SysBack™) to back up Siebel File System. SysBack provides system administrators and other system users with a simple, efficient way to back up and recover data from a command line or a SMIT menu-driven interface. SysBack lets you recover all or part of the system, and it also supports incremental backup/restore.

6.1.5 Backup image file format

Table 6-1 shows the naming convention for DB2 UDB backup files on the disk. You will notice a slight difference in the naming convention by platform. In UNIX/Linux, the file name is the concatenation of several elements separated by periods. In Windows, the file name is a four level subdirectory tree where the last element is the backup file.

When the backup image is written to a tape device, the file names are not created, but the information described in the below table is stored in the backup header for verification purposes.

Table 6-1 Database backup file format

Operating system	Database backup image file format
UNIX/Linux	DB_alias.Type.Inst_name.NODEnnnn.CATNnnnn.timestamp.Seq_num Example: `QA78RBA.0.ucsdb2.NODE0000.CATN0000.20050726092907.001`
Windows	DB_alias.Type\Inst_name\NODEnnnn\CATNnnnn\yyyymmdd\hhmmss.Seq_num Example: `QA78RBW.0\DB2\NODE0000\CATN0000\20050725\095656.001`

The elements of a backup image file and their explanations are shown in Table 6-2 on page 135:

Table 6-2 Backup file elements and their explanations

Element Type	Explanation
Database alias	A 1- to 8- character database alias name that was specified when the backup utility was invoked.
Type	Type of backup operation, where: 0 represents a full database-level backup, 3 represents a table space-level backup, and 4 represents a backup image generated by the LOAD...COPY TO command.
Instance name	A 1- to 8- character name of the current instance that is taken from the `DB2INSTANCE` environment variable.
Node number	The node number. In non-partitioned database systems, this is always `NODE0000`. In partitioned database systems, it is `NODExxxx`, where `xxxx` is the number assigned to the node in the `db2nodes.cfg` file.
Catalog node number	The node number of the catalog node for the database. In non-partitioned database systems, this is always `CATN0000`. In partitioned database systems, it is `CATNxxxx`, where `xxxx` is the number assigned to the node in the `db2nodes.cfg` file.
Timestamp	A 14-character representation of the date and time at which the backup operation was performed. Time value specified is interpreted as a Coordinated Universal Time (CUT). The timestamp is in the form yyyymmddhhnnss, where: yyyy represents the year (1995 to 9999) mm represents the month (01 to 12) dd represents the day of the month (01 to 31) hh represents the hour (00 to 23) nn represents the minutes (00 to 59) ss represents the seconds (00 to 59)
Sequence number	A 3-digit number used as a file extension.

6.1.6 DB2 UDB processes related to database backup/restore

Understanding the DB2 UDB spawned backup/restore processes will help you optimize the backup/restore performance. In Table 6-3 on page 136, you can find the DB2 UDB processes and their definitions.

Table 6-3 Main backup/restore processes

Process name	Description	Remark
`db2bm`	The buffer manipulator of backup/restore is the process used to read from a table space during a backup operation, and to write to a table space during a restore operation.	One process per parallelism value.
`db2med`	This process handles the reading from the database table space for backup and writing to the database table spaces during restore and load.	One process per target writing location.

Figure 6-3 shows the backup process model illustrating the parallelism.

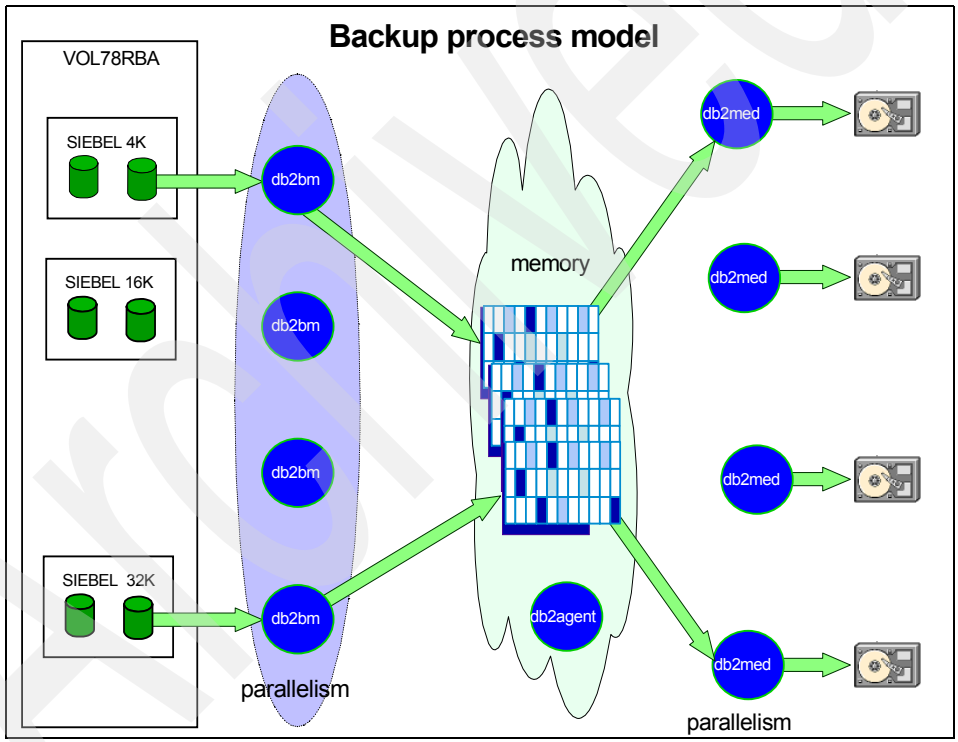

Figure 6-3 Backup process model

The backup command in Example 6-2 on page 137 shows multiple virtual disk targets with a parallelism value of 6.

Example 6-2 Backup command with multiple targets and with PARALLELISM option

```
KANAGA.Almaden.ibm.com:aixdba:/dbbkup>db2 “backup database VOL78RBA to
"/dbbkup/scenario", "/dbbkup/scenario", "/dbbkup/scenario", "/dbbkup/scenario",
"/dbbkup/scenario", "/dbbkup/scenario" with 12 buffers buffer 1024 parallelism
6  without prompting”
```

Here is the list of db2bm & db2med processes when the backup command in Example 6-2 ran.

```
KANAGA.Almaden.ibm.com:aixdba:/dbbkup> ps -ef |grep db2bm
  ucsdb2 319644 290834   1   Jul 15      -  0:09 db2bm.397392.3 0
  ucsdb2 524288 311448   0   Jul 15      -  4:08 db2bm.397392.2 0
  ucsdb2 548980 311448   0   Jul 15      -  1:29 db2bm.397392.5 0
  ucsdb2 737384 307350   0 10:54:10      -  0:01 db2bm.397392.0 0
  ucsdb2 741482 307350   0 10:54:10      -  0:00 db2bm.397392.4 0
  ucsdb2 745580 307350   4 10:54:10      -  0:37 db2bm.397392.1 0
KANAGA.Almaden.ibm.com:aixdba:/dbbkup> ps -ef |grep db2med
  ucsdb2 749678 307350   2 10:54:10      -  0:51 db2med.397392.0 0
  ucsdb2 753776 307350   3 10:54:10      -  0:42 db2med.397392.1 0
  ucsdb2 757874 307350   0 10:54:10      -  0:50 db2med.397392.2 0
  ucsdb2 761972 307350   1 10:54:10      -  0:42 db2med.397392.3 0
  ucsdb2 766070 307350   1 10:54:10      -  0:42 db2med.397392.4 0
  ucsdb2 770168 307350   2 10:54:10      -  0:43 db2med.397392.5 0
```

6.1.7 Database history file

The history file keeps the record of administrative operations, including:

- Backup (all types)
- Restore and rollforward operations
- The events such as table space administration, reorganize table, drop tables, and LOAD.

This file is created at the time of the creation of the database and gets updated when one of the above events occurs. You can access the history file by issuing the following command:

```
LIST HISTORY {BACKUP | ROLLFORWARD | REORG |
CREATE TABLESPACE | ALTER TABLESPACE | DROPPED TABLE | LOAD |
RENAME TABLESPACE | ARCHIVE LOG}
{ALL | SINCE timestamp |CONTAINING {schema.object_name | object_name}}
FOR [DATABASE] database-alias
```

You can view the database backup information by using the LIST HISTORY BACKUP command. This information gives you valuable input to restore processes. Example 6-3 on page 138 shows the backups that were performed on the database QA78RBA.

Example 6-3 Backup history command

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 list history backup all for qa78rba

                    List History File for qa78rba

Number of matching file entries = 1


 Op Obj Timestamp+Sequence Type Dev Earliest Log Current Log  Backup ID
 -- --- ------------------ ---- --- ------------ ------------ --------------
  B  D  20050726092907001   F    D  S0000000.LOG S0000000.LOG
 ----------------------------------------------------------------------------
  Contains 9 tablespace(s):

  00001 SYSCATSPACE
  00002 USERSPACE1
  00003 SIEBEL_4K
  00004 SIEBEL_4KI
  00005 SIEBEL_16K
  00006 SIEBEL_16KI
  00007 SIEBEL_32K
  00008 SIEBEL_32KI
  00009 SYSTOOLSPACE
 ----------------------------------------------------------------------------
    Comment: DB2 BACKUP qa78rba OFFLINE
 Start Time: 20050726092907
   End Time: 20050726093058
     Status: A
 ----------------------------------------------------------------------------
  EID: 4250 Location: /dbbkup
```

In the above sample output, “Op” means operation, and the value “B” stands for backup, the “Obj ID” signifies the granularity of the backup, and “D” means it is a full database backup, and type “F” signifies that it is offline backup. The “Earliest Log” signifies that this is the first log required for the rollforward operation, and “Current Log” signifies that this was the last log that was written to when the backup completed.

Pruning the history file

The `PRUNE HISTORY` command is used to delete entries from the database history file. This may be necessary if the file becomes excessively large when the retention period is high (long duration). `PRUNE HISTORY` command syntax is as follows:

```
prune history <timestamp> with force option
```

In Example 6-4, the `PRUNE HISTORY` command will remove the entries for all restores, loads, table space backups, and full database backups taken before and including `20050718203138` from the database history file.

Example 6-4 Prune history command

```
KANAGA.Almaden.ibm.com:aixdba:/aixdba> db2 prune history 20050718203138 with
force option
DB20000I  The PRUNE command completed successfully.
```

Note: The size of the database history file is controlled by the `REC_HIS_RETENTN` database configuration parameter that specifies a retention period (in days) for the entries in the file. Even if the number for this parameter is set to zero (0), the most recent full database backup (plus its restore set) is kept. The retention period has a default value of 366 days. The period can be set to an indefinite number of days by using -1. In this case, explicit pruning of the file is required.

6.1.8 Backup examples

DB2 UDB backup has several features and options, but you should select the appropriate strategy that will protect your data and meet your business requirements.

The following examples show you how to perform backup and introduce you to the new features of backup in DB2 UDB V8.

Offline database backup

Before taking the offline database backup, ensure no applications are connected to the database; otherwise, you will receive an `SQL1035N` error message. In Example 6-5, we first check if there are any database connections using the `LIST APPLICATIONS` command. If we find any application connected to the database, then use the `FORCE APPLICATION` command to force off the connections. A backup of database `QA78RBA` is then performed to the storage location `/dbbkup/qa78rba/ucsdb2`. The option "`WITHOUT PROMPTING`" key phrase specifies the backup will run unattended and that any actions which normally require user intervention will return an error message.

Example 6-5 Offline database backup for QA78RBA

```
KANAGA.Almaden.ibm.com:aixdba:/aixdba>db2 list applications for database
qa78rba

Auth Id  Application  Appl.     Application Id              DB       # of
         Name         Handle                                Name    Agents
------- ------------ ---------  -------------------------- -------- -----
```

```
SADMIN   siebmtshmw     93      G901275C.05DC.08E311191234  QA78RBA   1

KANAGA.Almaden.ibm.com:aixdba:/aixdba>db2 "force application (93)"
DB20000I  The FORCE APPLICATION command completed successfully.
DB21024I  This command is asynchronous and may not be effective immediately.

KANAGA.Almaden.ibm.com:aixdba:/aixdba>db2 backup database QA78RBA to
/dbbkup/qa78rba/ucsdb2 without prompting
Backup successful. The timestamp for this backup image is : 20050710131120
```

Full backup with COMPRESS option

COMPRESS is a new option introduced in DB2 UDB V8 on the database backup command. Using the COMPRESS option saves storage. The restore utility recognizes the compressed database backup and automatically decompresses the backup copy. No additional parameter is needed during restore.

Example 6-6 shows you how to perform a backup with COMPRESS option and also demonstrates the benefits of this new feature. The backups are performed on two databases, namely, QA78RBA of size 4.6 GB and VOL78RBA of size 201 GB.

Example 6-6 Offline database backup of QA78RBA with COMPRESS option

```
KANAGA.Almaden.ibm.com:aixdba:/aixdba> db2 backup database qa78rba to
/dbbkup/scenario compress without prompting

Backup successful. The timestamp for this backup image is: 20050714140058
```

The size of the backup file for QA78RBA without COMPRESS option is 4795 MB. With the COMPRESS option, the file size is reduced to 810 MB.

The size of the backup file for VOL78RBA without compress option is 202 GB. With the COMPRESS option, the file size is reduced to 33 GB.

The compress option is saving a good amount of storage space.

Important: Using the compression feature may result in additional CPU overhead due to the compression computations. However, the media I/O time is decreased because the image size is smaller. The overall backup/restore performance impact depends on whether CPU or media I/O is a bottleneck of the system.

Online backup with INCLUDE LOGS option

The database restore operation requires transaction logs when restoring the database with online image copy. The INCLUDE LOGS option will backup the required transaction log files with online database backup image. The benefit of

this option is that you do not have to decide which log files are required to guarantee the consistency of the restored database. This provides some protection against the deletion of log files required for a successful recovery. Further, if you need to ship backup images to a disaster recovery site, you do not have to send the log files separately. Example 6-7 shows the online backup with `INCLUDE LOGS` option.

Example 6-7 Online backup with INCLUDE LOGS option

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 backup db qa78rba online to /dbbkup
include logs
Backup successful. The timestamp for this backup image is : 20050718112404
```

Online table space level backup

A table space level backup is supported when the database is in archive logging mode. When a table space is backed up online, it remains available for use and all modifications to the data stored in that table space are recorded in the transaction log files. However, when an online restore or online rollforward recovery operation is performed against a table space, the table space itself is not available for use until the operation has completed. Example 6-8 shows online table space backup.

Example 6-8 Table space level backup

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 “backup database QA78RBA tablespace (
SIEBEL_4K, SIEBEL_4KI ) online to ‘/dbbkup/OnlineTest’ with out prompting”

Backup successful. The timestamp for this backup image is : 20050718113658
```

`SIEBEL_4K` and `SIEBEL_4KI` are two data table spaces residing in the `QA78RBA` database to be backed up. This command will produce a backup image of the specified table spaces within a database at the target location `/dbbkup/OnlineTest`.

6.1.9 VLDB backups

In the current on demand world, DBAs are often faced with maintaining large DB2 UDB databases. In this section, we discuss some of the advancements in system/storage architecture and database configuration parameters that will help back up these large databases.

- **Storage architecture:**

  There are product features such as FLASHCOPY that are available on storage systems such as IBM ESS, which will flash a database from one area of storage to other. This flash copy can then be backed up using the normal

process to achieve “zero” downtime for backups on the primary production server. This is the best available solution to backup: quickly copy/duplicate your database to another environment/machine.

If the backend for your storage is an enterprise storage server, separating the areas where the database and the backup volumes are (isolating them by adapters) decreases the I/O contention during a database backup.

- **System architecture:**

  Adding more I/O paths between the database server and storage will increase the throughput of data. If your database is on Enterprise storage, then increasing the number of Fibre Channel cards that are connected to a high speed storage switch can decrease the time taken by backups.

- **Database layout:**

  For the backup of a large database, the size of table space (amount of data held) has impact on database backup performance. The backup job performs better for a larger number of table spaces with smaller amounts of data than a smaller number of table spaces with larger amounts of data. Therefore, if possible, try to spread the tables out, because having a large number of table spaces helps database backup performance.

- **Buffers:**

  The default 1024 buffer size will work great for small to medium databases. For large databases, play with this parameter to get better streaming through I/O adapters.

6.2 Recovery overview

The rebuilding of the database is called *recovery*. We discuss various recovery methods in this section. You should choose the recovery method that is best suited to your business environment.

DB2 UDB provides three types of recovery:

- Crash recovery
- Version recovery
- Rollforward recovery

Crash recovery

Crash recovery is the process by which the database is moved back to a consistent and usable state. This is done by rolling back incomplete transactions and completing committed transactions that were still in memory when the crash occurred.

When the AUTORESTART database configuration parameter is enabled, the first connect to the database will automatically issue the RESTART DATABASE command when it is required. This will cause the system to read the logs which were active at the time of the crash and redo or undo the transactions as appropriate. Once this operation has completed, the database will be in a consistent state.

You can check the database configuration parameter AUTORESTART using the GET DB CFG command. Following is an example from the AIX environment:

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 get db cfg for qa78rba |grep restart
Auto restart enabled                        (AUTORESTART) = ON
```

If the AUTORESTART parameter is set to OFF, you will need to explicitly issue the RESTART DATABASE command in order to initialize crash recovery. Once the restart process completes, you can then connect to the database.

Issue the RESTART command as shown in the example below:

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 restart db qa78rba
DB20000I  The RESTART DATABASE command completed successfully.
```

Version recovery

Version recovery is the restoration of a previous version of the database, using an image that was created during a backup operation. This process provides recovery from media, hardware, operational, and software problems to the last offline backup. Therefore, the more frequent the backups, the more current the recovered database will be. You must schedule and perform a full backup of the database on a regular basis. As the time between the backups increases, so will the amount of loss of data. Table space recovery is not supported by the version recovery.

Rollforward recovery

Rollforward recovery uses a online or a offline backup copy of the database or table space to replace the current data and then applies log records to recover changes made after the backup image was created. The recovery can be to a point in time or the last committed unit of work.

Rollforward recovery using a recent backup will require fewer log records than a roll forward using an older backup, so the frequency of backups will still be a consideration for the database administrator.

To use the database rollforward recovery method, you must ensure that log archiving is enabled to facilitate the online backup, and that the log files are created and available for the rollforward process.

Use the `GET DB CFG` command to check the database configuration parameter for log archiving:

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 get db cfg |grep -i method
 First log archive method                 (LOGARCHMETH1) = LOGRETAIN
 Second log archive method                (LOGARCHMETH2) = OFF
```

6.2.1 Restore command and authorities required

The `RESTORE DATABASE` command rebuilds a damaged or corrupted database that has been backed up using the DB2 backup utility. The restore command supports the restore of the full database image, the incremental/delta backup image, and the table space level backup image.

Following is the syntax for the database restore command:

```
RESTORE DATABASE source-database-alias
{ restore-options | CONTINUE | ABORT }
[USER username [USING password]] [TABLESPACE [ONLINE] |
TABLESPACE (tblspace-name [ {,tblspace-name} ... ]) [ONLINE] |
HISTORY FILE [ONLINE] | LOGS [ONLINE] | COMPRESSION LIBRARY [ONLINE]]
[INCREMENTAL [AUTOMATIC | ABORT]] [USE {TSM | XBSA} [OPEN num-sess
SESSIONS][OPTIONS {options-string | options-filename}] |
FROM dir/dev [{,dir/dev} ... ] | LOAD shared-lib [OPEN num-sess SESSIONS]
[OPTIONS {options-string | options-filename}]] [TAKEN AT date-time]
[TO target-directory] [INTO target-database-alias] [LOGTARGET directory]
[NEWLOGPATH directory] [WITH num-buff BUFFERS] [BUFFER buffer-size]
[DLREPORT file-name] [REPLACE HISTORY FILE] [REPLACE EXISTING] [REDIRECT]
[PARALLELISM n] [COMPRLIB lib-name] [COMPROPTS options-string]
[WITHOUT ROLLING FORWARD] [WITHOUT DATALINK] [WITHOUT PROMPTING]
```

Authorities required for database restore

The restore operation requires following authorities:

- SYSADM, SYSCTRL, or SYSMAINT authority is required to restore to an existing database.
- SYSADM or SYSCTRL authority is required to restore to a new database.

6.2.2 Rollforward command and authorities required

The `ROLLFORWARD` command recovers a database by applying transactions recorded in the database log files. This command is invoked after a database or a table space has been restored. The restore operation will automatically place the database in “Roll-forward pending” state unless the `WITHOUT ROLLING FORWARD` option has been specified with the `RESTORE` command.

The `ROLLFORWARD` command syntax is as follows.

```
ROLLFORWARD DATABASE database-alias [USER username [USING password]]
[TO {isotime [ON ALL DBPARTITIONNUMS] [USING LOCAL TIME] | END OF LOGS
[On-DbPartitionNum-Clause]}] [AND {COMPLETE | STOP}] |
{COMPLETE | STOP | CANCEL | QUERY STATUS [USING LOCAL TIME]}
[On-DbPartitionNum-Clause] [TABLESPACE ONLINE | TABLESPACE (tblspace-name
[ {,tblspace-name} ... ]) [ONLINE]] [OVERFLOW LOG PATH (log-directory
[{,log-directory ON DBPARTITIONNUM db-partition-number} ... ])]
[NORETRIEVE][RECOVER DROPPED TABLE dropped-table-id TO export-directory]
On-DbPartitionNum-Clause:
  ON {{DBPARTITIONNUM | DBPARTITIONNUMS} (db-partition-number
  [TO  db-partition-number] , ... ) | ALL DBPARTITIONNUMS [EXCEPT
  {DBPARTITIONNUM | DBPARTITIONNUMS} (db-partition-number
  [TO db-partition-number] , ...)]}
```

Example 6-9 shows how to use the `ROLLFORWARD DATABASE` command from the Command Line Processor (CLP).

Example 6-9 ROLLFORWARD command from the Command Line Processor (CLP)

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 rollforward database qa78rba to end
of logs and complete

                                Rollforward Status

 Input database alias                   = qa78rba
 Number of nodes have returned status   = 1

 Node number                            = 0
 Rollforward status                     = not pending
 Next log file to be read               =
 Log files processed                    = S0000006.LOG - S0000008.LOG
 Last committed transaction             = 2005-07-25-23.50.44.000000
```

Authorities required for roll forward

The following authorities are required to perform a rollforward recovery:

- SYSADM
- SYSCTRL
- SYSMAINT

6.2.3 Recover database command

The `RECOVER DATABASE` command combines the functionality of the `RESTORE DATABASE` and the `ROLLFORWARD DATABASE` commands, and does not require you to specify which database backup image must be restored, or which log files are needed for point-in-time (PIT) recovery.

The following command is to restore and roll forward a database to a particular point in time or to the end of the logs:

```
RECOVER DATABASE database-alias [TO {isotime
[USING LOCAL TIME | USING GMT TIME] [ON ALL DBPARTITIONNUMS] |
END OF LOGS [On-DbPartitionNum-Clause]}] [USER username
[USING password]] [USING HISTORY FILE (history-file [{, history-file
ON DBPARTITIONNUM db-partition-number} ... ])] [OVERFLOW LOG PATH
(log-directory [{,log-directory ON DBPARTITIONNUM db-partition-number} ...
])][COMPRLIB lib-name] [COMPROPTS options-string]
On-DbPartitionNum-Clause:
  ON {{DBPARTITIONNUM | DBPARTITIONNUMS} (db-partition-number
  [TO  db-partition-number] , ... ) | ALL DBPARTITIONNUMS [EXCEPT
  {DBPARTITIONNUM | DBPARTITIONNUMS} (db-partition-number
  [TO db-partition-number] , ...)]}
```

Example 6-10 shows how to use the RECOVER DATABASE command through the CLP:

Example 6-10 Recover database command through the CLP

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 recover database qa78rba to
2005-07-25-16.41.00.000000
                                Rollforward Status

 Input database alias                   = qa78rba
 Number of nodes have returned status   = 1

 Node number                            = 0
 Rollforward status                     = not pending
 Next log file to be read               =
 Log files processed                    = S0000006.LOG - S0000008.LOG
 Last committed transaction             = 2005-07-25-16.40.03.000000

DB20000I  The RECOVER DATABASE command completed successfully.
```

Authorities required for database recover

The authorities required for using the RECOVER DATABASE command are:

- SYSADM, SYSCTRL, or SYSMAINT authority is required to recover to an existing database.
- SYSADM or SYSCTRL authority is required to recover to a new database.

6.2.4 Database recovery examples

Different recovery methods are discussed in the sections that follow, and you need to decide which recovery method is best suited to your business environment.

The following examples show how to use restore, rollforward, and recovery commands.

Full database restore

Example 6-11 shows the restore of a full offline database image to an existing database. This command will restore the database image which is saved at `/dbbkup/restore` with a timestamp of `20050724214913` to the existing `QA78RBA` database. This command will effectively overwrite the old `QA78RBA` database files with the backed up database image files. To identify the timestamp of the backup, use the `LIST HISTORY` command as explained in 6.1.7, “Database history file” on page 137.

Example 6-11 `RESTORE DATABASE` command issued through the CLP

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 "restore database qa78rba from
'/dbbkup/restore' taken at 20050724214913"
SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I  The RESTORE DATABASE command completed successfully.
```

Database restore with `LOGTARGET` option

The advantage of including logs with the backup images is discussed in “Online backup with INCLUDE LOGS option” on page 140.

Example 6-12 shows the restore of a database with the `LOGTARGET` option. When restoring the database, specify the directory where you want DB2 UDB to place the log files.

Example 6-12 `RESTORE` command with `LOGTARGET` option

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 restore db qa78rba from
/dbbkup/OnlineTest logtarget /qa78rbalogs
SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I  The RESTORE DATABASE command completed successfully.

kanaga.almaden.ibm.com:aixdba:/aixdba>db2 "rollforward db qa78rba to end of
logs and stop overflow log path (/qa78rbalogs)"
```

```
                         Rollforward Status

 Input database alias                   = qa78rba
 Number of nodes have returned status   = 1

 Node number                            = 0
 Rollforward status                     = not pending
 Next log file to be read               =
 Log files processed                    = S0000005.LOG - S0000005.LOG
 Last committed transaction             = 2005-07-17-23.54.50.000000

DB20000I  The ROLLFORWARD command completed
successfully.
```

Point in time recovery

Example 6-13 shows the restore of a database using an online backup image and subsequently performing rollforward to a point in time.

The ROLLFORWARD command will rollforward all logs located in the log directory specified in the database configuration file for QA78RBA up to and including the stated point in time. The "AND COMPLETE" key phrase completes the rollforward recovery process by rolling back incomplete transactions and turning off the roll forward pending state of the database.

Example 6-13 Point in time recovery

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 restore database QA78RBA from
/dbbkup/PIT
SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I  The RESTORE DATABASE command completed successfully.

kanaga.almaden.ibm.com:aixdba:/aixdba>db2 connect to QA78RBA
SQL1117N  A connection to or activation of database "QA78RBA" cannot be made
because of ROLL-FORWARD PENDING.  SQLSTATE=57019

kanaga.almaden.ibm.com:aixdba:/aixdba>db2 rollforward database QA78RBA to
2005-07-25-12.06.00.000000 using local time and complete

                                 Rollforward Status

 Input database alias                   = QA78RBA
 Number of nodes have returned status   = 1

 Node number                            = 0
 Rollforward status                     = not pending
```

```
 Next log file to be read               =
 Log files processed                    = S0000006.LOG - S0000007.LOG
 Last committed transaction             = 2005-07-25-12.05.24.000000

DB20000I  The ROLLFORWARD command completed
successfully.
```

Redirected restore

You can restore a DB2 UDB online or offline database backup image and redirect it into another DB2 database that is running under the same type of operating system. With the redirected restore, you can:

- Change the number and size of containers per table space.
- Change the name and location of table space containers.

But it is impossible to add and remove table spaces, change the page size of a table space, or change the type of your table space (from SMS to DMS and vice versa).

The DB2 UDB command statements required to perform a redirected restore are:

- `RESTORE DATABASE INTO` command
- `SET TABLESPACE CONTAINERS` command for each table space in the database
- `RESTORE DATABASE CONTINUE` command

It is important to remember that the entire redirected restore operation must be invoked from the same session; otherwise, `SQL0900N` is returned and the restore operation fails. One way to ensure that this does not happen is to create and run a script that contains all three parts of the redirected restore process.

You also need to restore DB2 UDB log files and issue a `ROLLFORWARD DATABASE` command against the target database.

Table 6-4 shows our lab setup for a redirect restore scenario. In this scenario, we use redirected restore to restore a database from one system to another and also demonstrate the process to change the table space container paths. For this scenario, we performed the backup using the online backup command.

Table 6-4 Lab setup for redirect restore

DB Server Name	DB Name	Log Path
PUGET - DB2 UDB on Windows (Source)	QA78RBW	C:\qa78rbwlogs
WISLA - DB2 UDB on Windows (Target)	QA78RBW	C:\Databaselogs

1. Before you plan for a redirect restore, you should write down the table space type and size (in other words, the high water mark or largest size) of the source database using the following command:

```
db2 "list tablespaces show detail"
```

2. The following command creates a new database called QA78RBW on WISLA. The redirect option is used to specify the container paths. You will receive the SQL1277N message that tells you that you can define the containers as required.

```
db2 "restore db QA78RBW from C:\backups into QA78RBW redirect without
prompting"
```

3. We want to redirect all the table spaces associated with the database on PUGET by setting new container paths on WISLA:

```
db2 "set tablespace containers for 0 using (path 'C:\Databases\sms\syscat')"

db2 "set tablespace containers for 1 using (path 'C:\Databases\sms\temp16k')"

db2 "set tablespace containers for 2 using (path 'C:\Databases\sms\user')"

db2 "set tablespace containers for 3 using (path 'C:\Databases\sms\temp4k')"

db2 "set tablespace containers for 4 using (path 'C:\Databases\sms\temp32k')"

db2 "set tablespace containers for 5 using (file 'C:\Databases\dms\siebel_4k'
533632)"

db2 "set tablespace containers for 6 using (file 'C:\Databases\dms\siebel_4ki'
443552)"

db2 "set tablespace containers for 7 using (file 'C:\Databases\dms\siebel_16k'
113728)"

db2 "set tablespace containers for 8 using (file 'C:\Databases\dms\siebel_16ki'
158368)"

db2 "set tablespace containers for 9 using (file 'C:\Databases\dms\siebel_32k'
10912)"

db2 "set tablespace containers for 10 using (file
'C:\Databases\dms\siebel_32ki' 24064)"

db2 "set tablespace containers for 11 using (path 'C:\Databases\sms\systool')"

db2 "set tablespace containers for 12 using (path
'C:\Databases\sms\systooltemp')"
```

In the SET commands above, the table space IDs 0, 1, 2, 3, and 4 are used to specify the table space that will be assigned to the new container path. All five table spaces are System Managed Spaces (SMS) with the containers pointing to the C:\Databases\sms table spaces directory. Note, you only need to create the C:\tablespaces\sms directory. The execution of the SET command will create the directories SYSCAT, TEMP16K, USER, TEMP4K, and TEMP32K.

The Database Managed Spaces (DMS) table spaces have a slightly different syntax. The table space IDs 5, 6, 7, 8, 9, and 10 are DMS table spaces with the containers pointing to C:\Databases\dms.

In case you want to determine what other table spaces reside in the database, execute the following command in the same session command window.

```
db2 "list tablespaces show detail"
```

Specify new table space containers for each table space placed in the "Restore Pending" state by executing a SET TABLESPACE CONTAINERS for each appropriate table space. Keep in mind that SMS table spaces can only use PATH containers, while DMS table spaces can only use FILE or DEVICE containers.

4. After the table space containers are set, continue the database restore by executing the command:

```
db2 "restore database QA78RBW continue"
```

5. Rolling forward the log files is necessary since the database backup was performed online. After shipping the log files from the source to the target database server, we perform a rollforward operation on the target database server.

```
db2 "rollforward database QA78RBW to end of logs and stop"
```

6.3 RUNSTATS

RUNSTATS is a utility that updates statistics that are used by the optimizer to determine the fastest path to the data. With DB2 UDB V8.2, the number of RUNSTATS options has increased considerably. Understanding how to use RUNSTATS, with both the existing options and the new ones, can help you get optimal performance for your database.

Role of DB2 optimizer and catalog statistics

When optimizing SQL queries, the decisions made by the SQL compiler are heavily influenced by the optimizer's model of the database contents. The

optimizer uses this data model to estimate the cost of alternative access paths that can be used to resolve a particular query. A key element in the data model is the set of statistics gathered about the data contained in the database and stored in the system catalog tables. This includes statistics for tables, nicknames, indexes, columns, and user-defined functions (UDFs). A change in the data statistics can affect the choice of access plan selected as the most efficient method of accessing the desired data.

Examples of the statistics available which help define the data model to the optimizer include:

- The number of pages in a table and the number of pages that are not empty.
- The degree to which rows have been moved from their original page to other (overflow) pages.
- The number of rows in a table.
- Statistics about individual columns such as the number of distinct values in a column.
- The degree of clustering of an index; that is, the extent to which the physical sequence of rows in a table follows an index.
- Statistics about the index such as the number of index levels and the number of leaf pages in each index.
- The number of occurrences of frequently used column values.

RUNSTATS command

The `RUNSTATS` command updates statistics about the physical characteristics of a table and the associated indexes. The `RUNSTATS` command can provide different levels of statistics as shown in the following two tables, Table 6-5 for basic statistics and Table 6-6 for extended statistics:

Table 6-5 Basic statistics

Database object	RUNSTATS command
Table	**RUNSTATS ON TABLE** <tablename>
Index	**RUNSTATS ON TABLE** <tablename> **FOR INDEXES ALL**
Both tables and indexes	**RUNSTATS ON TABLE** <tablename> **AND INDEXES ALL**

Table 6-6 Enhanced statistics

Database object	RUNSTATS command
Table	**RUNSTATS ON TABLE** <tablename> **WITH DISTRIBUTION**

Database object	RUNSTATS command
Index	**RUNSTATS ON TABLE** <tablename> **FOR DETAILED INDEXES ALL**
Both tables and indexes	**RUNSTATS ON TABLE** <tablename> **WITH DISTRIBUTION AND DETAILED INDEXES ALL**

RUNSTATS command in Siebel environment

With Siebel applications, we recommend you use the following options as shown in Example 6-14:

Example 6-14 RUNSTATS command in Siebel environment

```
db2 runstats on table siebel.s_evt_act with distribution and indexes
all shrlevel change
```

In the example, the option "WITH DISTRIBUTION" specifies that distribution statistics are requested, the option "AND INDEXES ALL" specifies that statistics will be updated on both the table and its indexes, and the option "SHRLEVEL CHANGE" specifies that other users can read from and write to the table, while statistics are being gathered.

updatestats.sql script for RUNSTATS

The updatestats.sql is a Siebel provided script which invokes a C stored procedure called siebstat via the ODBC interface, and updates the statistics for those tables listed in the updatestats.sql file. It scans the tables and indexes to gather statistics about each DB2 UDB object, such as the number of rows in the table, the number of unique key values in the index, and so on. By editing the updatestats.sql file, you can create multiple updatestats scripts to update statistics on groups of tables and indexes.

To run updatestats.sql use the odbcsql command:

```
odbcsql /s DATASOURCE_NAME /u username /p password /v separator
$DBSRVR_ROOT/db2udb/updatestats.sql TABLEOWNER_NAME
```

The loadstats.sql script is a Siebel specific script to load the statistics required for the operation of the Siebel application into the DB2 catalog tables. You can locate the script, loadstat.sql, in the $DBSRVR_ROOT/db2udb directory (UNIX) or the DBSRVR_ROOT\db2udb directory (Windows).

To run loadstats.sql, use the odbcsql command:

```
odbcsql /s DATASOURCE_NAME /u username /p password /v separator
$DBSRVR_ROOT/db2udb/loadstats.sql TABLEOWNER_NAME
```

Important: Make sure you execute the `loadstats.sql` script after the completion of the `RUNSTATS` command. Also, we recommend you run `RUNSTATS` only on those tables contained in the `updatestats.sql` file. The tables that were omitted were left out on purpose and could hinder performance and operability if you run `RUNSTATS` on these tables.

We recommend that you run `RUNSTATS` on all catalog and Siebel tables daily if possible.

`RUNSTATS` command enhancements

In DB2 UDB V8.2, the `RUNSTATS` command is enhanced to improve the performance of statistics collection and to provide additional options. The `RUNSTATS` utility now has the ability to:

- Collect additional statistics, such as statistics about column combinations, and prefetching statistics about the table, index, and index-to-table relationship.
- Accept a list of index names (previously available only with the API).
- Accept a list of columns about which statistics are to be collected.
- Accept distribution statistics limits: `NUM_FREQVALUES` and `NUM_QUANTILES` values at the table level (without having to change the configuration parameters, and then disconnect and reconnect all users).
- Accept individual column `NUM_FREQVALUES` and `NUM_QUANTILES` values.
- Perform a faster (sampled) collection of `DETAILED` index statistics. We do not recommend running `DETAILED` index statistics for Siebel tables.

Checking to see if `RUNSTATS` has been run

It is important to routinely check to see if `RUNSTATS` has been run on tables or not. This section provides method of identifying the tables that require `RUNSTATS`.

The easy way to determine whether the `RUNSTATS` utility has been run or not is to query the system catalog tables.

If the time of the last statistics update is not current, it may be a good idea to run the `RUNSTATS` command for the table and indexes. Use the command shown in Example 6-15 on page 155 to check the statistics time. If the `RUNSTATS` utility has never been run, the value of “-1” will be stored for the columns `NLEAF` and `NLEVELS` in the `SYSCAT.INDEXES` table. After the initial `RUNSTATS` was run, there will be a value higher than “-1” stored in the `NLEAF` and `NLEVELS` columns, and the timestamp when the `RUNSTATS` was run will be stored for the `STATS_TIME`.

Example 6-15 Check `stats_time` from system catalog table

```
SELECT  stats_time, char(tabname,40), nleaf, nlevels, stats_time FROM
syscat.indexes

SELECT char(tabname,40) FROM syscat.tables  WHERE type = 'T' AND stats_time is
null

SELECT char(indname,40) FROM syscat.indexes WHERE stats_time is null
```

You can also look for tables and indexes that have statistics that are over a certain number of days old. Example 6-16 shows the statement that will list all the tables which have statistics collected over seven days ago.

Example 6-16 Statistics that are older than a week

```
SELECT tabname FROM syscat.tables WHERE stats_time < current timestamp - 7 days
>
SELECT indname FROM syscat.indexes WHERE stats_time < current timestamp - 7
days>
```

The system catalog tables also benefit from having `RUNSTATS` performed on them.

Database configuration parameter

The `STAT_HEAP_SZ` or the statistics heap size database configuration parameter indicates the maximum size of the heap used in collecting statistics using the `RUNSTATS` command. It is allocated when the `RUNSTATS` utility is started, then freed when it completes. `STAT_HEAP_SZ` is part of agent private memory. When gathering distribution statistics, we recommend you increase the `STAT_HEAP_SZ` parameter so that more columns can fit into the heap. Wide tables would also require more memory for processing.

Generating `RUNSTATS` CLP script

The following DB2 UDB statement can be used to generate the `RUNSTATS` statements for the database.

```
db2 “SELECT 'db2 runstats on table ' || rtrim(tabschema) || '.' ||
char(tabname,40) || ' with distribution and indexes all shrlevel change' FROM
syscat.tables WHERE type = 'T' ORDER BY tabschema, tabname”
```

The result can be piped to a file to generate a script.

REBIND

After performing a RUNSTATS, packages should be rebound so that their static SQL can take advantage of the most recent system statistics. Use the DB2RBIND command to rebind all the database packages:

```
db2rbind dbname -l logfile.out ALL
```

> **Note:** In the UNIX environment, if you are running the db2rbind command from the <DB2_ROOT>/sqllib/bin, you may not have the write permission to this directory. You will need to specify the output message file path to write the message to the directory where you have the write permission.

6.4 DB2 UDB table reorganization

Table and index maintenance is a critical success factor in Siebel implementations. Appropriate and timely maintenance must be applied to ensure performance and scalability characteristics, in line with user and management expectations. Setting up a schedule to perform table maintenance is an important part of maintaining user productivity and achieving return-on-investment targets.

In this section, we discuss:

- Siebel schema changes from Siebel 6 to Siebel 7.
- How DB2 UDB data reorganization is used in a Siebel environment and how to handle volatile tables.
- Why this maintenance is important for optimal performance of your implementation.
- Methods to determine which tables need reorganization.
- Upgrade considerations.

6.4.1 DB2 UDB data reorganization in a Siebel environment

Database administration is about being responsible for maintaining the health of the physical database schema. The physical health of tables and indexes is critical for customers to achieve performance, scalability, and ROI goals.

What drives the need to reorganize?

Insert, update, and, to a lesser degree, delete processing will, over time, affect the physical location of data in storage. Inserted data that should be stored on contiguous pages becomes spread out as free space is exhausted. Updates to columns that produce rows too long for their original locations become relocated.

Delete activity may leave large numbers of empty pages which should be coalesced. As the degree of clustering is reduced, numerous performance and scalability issues can and will occur.

The DB2 UDB reorganization process will restore clustering, correct row relocations, remove entries flagged for deletion, and adjust free space.

Clustering indexes in the Siebel schema

Prior to Siebel 7, the Siebel schema did not support clustering indexes for DB2 UDB on UNIX and Windows operating systems.

The Siebel document, FAQ 1486: *DB2 and REORG: Normal Database Maintenance - when is a REORG done?* was written for Siebel 6 and DB2 UDB V6 implementations to provide direction for using DB2 UDB reorganization utilities in Siebel 6.

For Siebel 7 implementations, use the Siebel document FAQ 2072: *How can REORG be used for Database Maintenance with Siebel 7 and DB2 Version 7 or 8?*

With the release of Siebel 7, the Siebel schema on Windows, UNIX, and, starting with Siebel 7.8, Linux, includes clustering indexes on nearly all tables. Inclusion of clustering indexes provides improved performance, enhanced scalability, and reduces the need for reorganizations of data tables. The use of clustering indexes is helpful in the following ways:

- Clustering indexes groups data on physically contiguous storage. This can greatly enhance the performance of 1: M joins as well as range scans by keeping the required rows together, thereby reducing I/O.
- During inserts, DB2 UDB will attempt to keep the new rows in the clustering order as much as possible.
- Prefetch usage will be optimized for queries using the clustering index.

Determining if reorganization is necessary

There are several factors to consider in deciding whether or not a given table or index should be reorganized. These include:

- For tables with clustering indexes, determine if the cluster ratio or normalized cluster factor for a given table is less than 80%. If so, DB2 will consider the table to be out of cluster. Almost every Siebel table has a clustering index for DB2 UDB deployments. If the 80% cluster threshold is not met on a table of any size, performance and scalability implications are likely. Problems with the effectiveness of prefetch can occur at cluster ratios higher than 80%, therefore, tables should be maintained to prevent clustering from falling below

the 80% threshold. This is one of the most important maintenance tasks for overall performance and scalability.

- Check the `OVERFLOW` column of table `SYSCAT.TABLES`. This will provide information on the number of relocated rows. Table snapshots can also be used to determine how frequently relocated rows are being used by the application.
- There are three columns on the `SYSCAT.INDEXES` table that can be queried to find out the effectiveness of prefetch on the index:
  - *AVERAGE_RANDOM_FETCH_PAGES* is defined as the average number of random table pages between sequential page accesses when fetching table rows using the index. As the number of random pages increase, the effectiveness of prefetch decreases. In extreme cases, prefetch against an index could become disabled, increasing I/O and reducing throughput.
  - *AVERAGE_SEQUENCE_FETCH_GAP* is defined as the gap between sequences when fetching table rows using the index. This measures leaf pages where sets of random pages for a given index occur between sets of sequential leaf pages. A high value for this field is an indication that the table is disorganized or poorly organized with respect to the index.
  - *AVERAGE_SEQUENCE_FETCH_PAGES* is defined as the average number of index pages accessible in sequence. This is the number of index pages that the prefetchers can detect as being in sequence. The higher the value, the more efficiently prefetchers can operate on the index.
- On the `SYSCAT.INDEXES` table, the `NUM_EMPTY_LEAFS` column has the number of leaf pages with all RIDS marked for deletion; the value of `NUMRIDS_DELETED` shows the number of RIDS available for deletion on index pages excluding those RIDS on leaf pages on which all row identifiers are as marked deleted. These two columns are good inidcators for large number row deletion.

Usually, checking the condition of the clustering index will be sufficient to determine if reorganization is needed for a given table. `REORGCHK` can be used to check the condition of tables and indexes in an implementation. `REORGCHK` does not flag the clustering index, so additional analysis is needed to determine clustering status.

Example 6-17 shows the SQL statement which will list the tables having less than 80% clustered.

Example 6-17 SQL to identify tables with less than 80% clustered[1]

```
SELECT     substr(indschema, 1, 25), substr(tabname, 1, 25), indname,
           int(clusterfactor * 100)
FROM       syscat.indexes
```

[1] Siebel System, Reprinted by Permission.

```
WHERE      indextype = 'CLUS'
           AND int(clusterfactor * 100) < 80
           AND clusterfactor > 0
ORDER BY   indschema, tabname;
```

Special handling for volatile and append mode tables

There are a few tables that are not defined with clustering indexes. There are approximately 57 tables that are defined as Volatile. These few tables cannot support clustering indexes.

The Volatile tables in any environment can be identified with the sample SQL in Example 6-18:

Example 6-18 SQL to identity reorganization candidates for Volatile tables [2]

```
SELECT     substr(tabname, 1, 25)
FROM       syscat.tables
WHERE      volatile = 'C'
ORDER BY   tabname
```

The Activities table, S_EVT_ACT, is set to Volatile, but is not used by Siebel Remote. S_EVT_ACT is also set to Append Mode. This setting encourages the insertion of new rows at the end of the table. The Activities table is the only transactional table with these settings. Table reorganizations are unlikely for this table but could become necessary if row relocations occur for a significant number of rows which has been the case for some customers. If reorganization of S_EVT_ACT becomes necessary, use the index S_EVT_ACT_P1 on which to reorganize. The tables S_OPTY_POSTN, S_OPTY_TERR, and S_ORG_EXT are similarly defined. All are in Append Mode, and, as a result, none have defined clustering indexes. Use of data archiving could provide a second reason for reorganizing these tables. Removal of large numbers of records could make reorganization desirable to free storage, reset free space, and rebuild indexes.

The tables in Append Mode in any environment can be identified with the sample SQL in Example 6-19:

Example 6-19 SQL to identify reorganization candidate for Append Mode tables[3]

```
SELECT     substr(tabname, 1, 25)
FROM       syscat.tables
WHERE      append_mode = 'Y'
ORDER BY   tabname
```

[2] Siebel System, Reprinted by Permission
[3] Siebel, Reprinted by Permission

Upgrade considerations

For projects upgrading from Siebel 6 with DB2 UDB V6 to any version of Siebel 7 and DB2 UDB V8, it is important to recall that the Siebel 6 schema did not include clustering indexes. By default, tables implemented in Siebel 6 were reorganized to cluster on their P1 indexes. With the advent of Siebel 7, clustering indexes are part of the Siebel schema. During the upgrade from Siebel 6, you must allocate time to reorganize the tables from the P1 clustering to the Siebel 7 clustering. This step is very important to avoid performance and scalability problems with the Siebel 7 rollout.

DB2 UDB V8 introduces type-2 indexes. These new indexes improve database concurrency by reducing locking on index pages. For those implementations, which are upgrading to DB2 UDB V8 from a previous release of DB2 UDB, there is a step you must take to convert type-1 indexes to type-2 indexes. You can generate the following index reorganization command from the system catalog to convert indexes to type-2:

```
reorg indexes all for table <table name> convert
```

This command will have no effect if your tables are already defined with type-2 indexes.

Table reorganization methods

There are two methods of table reorganization available. They are listed here with the advantages and disadvantages of each:

- Classic table reorganization

  This method creates a shadow copy of the table. The shadow copy can be built in a temporary table space if there is insufficient capacity in the table's table space:

  - Advantages:
    - Fastest method for reorganization, especially if the reorganization takes place in the table's table space.
    - All indexes are completely rebuilt.
    - Read-only applications will have access to the original table for most of the time that the reorganization is running.
  - Disadvantages:
    - The shadow copy can require twice as much space as the original table.
    - Read-only access is available only during the first phases of the reorganization.

    - If the reorganization encounters any problems, it must be restarted from the beginning.
  - Classic table reorganization is best adapted to environments that have maintenance windows available.
- In-place table reorganization

  This method allows better access to the data during the reorganization.
  - Advantages:
    - Read and write access to table is possible.
    - Table reorganization can be paused and restarted. This is very helpful for implementations with known busy periods.
  - Disadvantages:
    - Elapsed time of the reorganization is longer than for classic reorganization.
    - Index organization will not occur. You may need to reorganize the indexes at a later time.
    - Log space requirements are greater as an in-place reorganization is logged.
    - Table must have type-2 indexes for in-place reorganization.
    - Tables with extended indexes cannot use in-place reorganization. Siebel applications support extended indexes, so there may be tables in an implementation that cannot use this type of reorganization.
  - In-place reorganization is well-suited to implementations with 24 x 7 availability requirements or small maintenance windows.

Proactive reorganization planning

As noted earlier, if a table with a clustering index becomes less than 80% clustered, there can be serious performance and scalability implications. There are several methods which you can use to proactively ensure that important data objects do not become out of cluster:

- Set up a schedule to periodically reorganize all populated tables.
- Use scripts or third-party tools to determine which tables should be reorganized.
- Create a forward looking procedure.

Some implementations have generous maintenance windows. In these cases, all tables and indexes which are subject to insert, update, and delete activity can be reorganized on a fixed schedule. Implementations that are not customer or partner-facing or which have relatively small tables can fall into this category.

Increasingly, implementations have large tables, users covering numerous time zones, and must keep maintenance windows to a minimum. Determining what maintenance is necessary and how frequently it is necessary will maximize the usefulness of available downtime.

A simple approach is to track the larger and more active tables in an implementation. For example, the sample script in Example 6-17 on page 158 can be modified to check for cluster ratios or normalized cluster factor of 95%, 90%, and 85%. For example after running `RUNSTATS`, we see that `S_CONTACT` is 95% clustered on its cluster key, `S_CONTACT_U1`. Two weeks later, S_CONTACT_U1 crosses the 90% clustering threshold. Database administrators will not want to risk the table becoming less than 80% clustered, so they can schedule this table to be reorganized sometime in the next two weeks. Over time, potential schedules will become apparent for all of the most important and large tables in the system.

It is important to include smaller tables in this analysis. Smaller tables generally have less activity, so it usually requires fewer resources and less time to determine the correct maintenance for smaller tables.

6.5 Query control

One of the major tasks for a DBA is to identify and tune complex and runaway queries. Especially in the case of the Siebel Application where the user can submit impromptu queries, the need for a tool to identify both the query and the user, prioritize, and analyze is high.

6.5.1 DB2 Query Patroller

DB2 Query Patroller is a powerful query management system that you can use to proactively and dynamically control the flow of queries against your DB2 UDB database in the following key ways:

- Define separate query classes for queries of different sizes to better share system resources among queries and to prevent smaller queries from getting stuck behind larger ones.
- Give queries submitted by certain users high priority so that these queries run sooner, such as priority for users over batch jobs.
- Automatically put large queries on hold so that they can be canceled or scheduled to run during off-peak hours.
- Track and cancel runaway queries.

- Generate reports that assist in identifying trends in database usage, such as which objects are being accessed, and which individuals or groups of users are the biggest contributors to the workload.

The features of Query Patroller allow you to regulate your database's query workload, so that small queries and high-priority queries can run promptly and you can use your system resources efficiently. In addition, you can collect and analyze information about completed queries to determine trends across queries, heavy users, and frequently used tables and indexes.

6.5.2 Siebel Query Cancel function

Recently, Siebel has introduced a feature which allows the user to cancel a query from their workstation.

To invoke the Query Cancel function to cancel a query that has been running too long:

View → Options → Query

6.5.3 Home-grown query cancel scripts

Example 6-20 is a home-grown, simplified query control script. You can easily expand this script to capture the following data:

- SQL being executed by the ID.
- Track how long each of the distinct SQLs ran before it was forced off.
- Number of times a particular user connection was cancelled.
- Number of times a particular SQL has been cancelled.
- Run `EXPLAIN` on SQL that was executing at the time of cancellation.
- Set soft and hard limits on which queries are monitored and killed.
- Send a note to specific user to notify them their query was cancelled.

Example 6-20 Simple query cancel script

```
#-----------------------------------------------------------------------------------
# Program : easykill#
# Desc    : Force of named applications after DB2 preset TIMEOUT is reached for
duration of current Unit Of Work
#-----------------------------------------------------------------------------------
#-----------------------------------------------------
# Number of minutes before application forced off db2
#-----------------------------------------------------
. $HOME/.profile
TIMEOUT=15
MAILLIST="sreeman@us.ibm.com"
# Define temp work files
```

```
TEMPFILE1=/home/aixdba/out/db2process.work
TEMPFILE2=/home/aixdba/out/db2agent.work
TEMPFILE3=/home/aixdba/out/db2force.work
TEMPFILE4=/home/aixdba/out/db2force.work.sorted
TEMPFILE5=/home/aixdba/out/db2users.work
rm $TEMPFILE1
rm $TEMPFILE2
rm $TEMPFILE3
#-------------------------------------------------------
# Get application agent details for selected db2 processes
#-------------------------------------------------------
db2 connect to vol78rba
db2 list applications show detail | grep Executing | grep siebel.exe | awk
'{print $3, $1, $12}' > $TEMPFILE1
cat $TEMPFILE1 | awk '{ (pos1= index($3,":")) ; hour = substr($3,pos1-2,2) ;
min = substr($3,pos1+1,2)
 print $1, $2, hour, min }' >> $TEMPFILE2
#---------------------------------------------
# Get current date and calculate no of minutes
#---------------------------------------------
let r1=`date +%H`
let r2=`date +%M`
let r3="($r1)*60+($r2)"
#-----------------------------------------------------------------------
# Read line record file, calculating no of minutes and calc difference
#-----------------------------------------------------------------------
let count=0
while read AGENT USERID HOUR MINUTE ;
do
let p1="($HOUR)*60+($MINUTE)"
let diff="($r3)-($p1)"
#------------------------------------------------------------------------
# Get users first, last name, email and work phone number from database
#------------------------------------------------------------------------
db2 "select substr(a.fst_name,1,10), substr(a.last_name,1,15),
sustr(a.email_addr,1,20), substr(a.work_ph_num,1,15) from siebel.s_contact a,
siebel.s_employee b where b.login='$USERID' and b.par_row_id=a.row_id" | head
-n +4 | tail -n -1 > $TEMPFILE5
read FIRST LAST EMAIL PHONE < $TEMPFILE5
print Difference is $diff r3=$r3 p1=$p1
 if [[ $diff -ge $TIMEOUT ]]
then let count=count+1 ; print "db2 \"force application ($AGENT)\" #USERID=
$USERID UOW =$diff mins [$FIRST,$LAST,$EMAIL,$PHONE]" >> $TEMPFILE3
 fi
done < $TEMPFILE2
db2 terminate
#---------------------------
# Run the db2 force commands
#---------------------------
```

```
chmod 700 $TEMPFILE3
$TEMPFILE3
#-------------------------------------------------------------------------------
# Sort file by highest number of minutes elapsed, only send mail if users are
forced off
#-------------------------------------------------------------------------------
sort -t= +2 -nr $TEMPFILE3 > $TEMPFILE4
if [[ count -ge 1 ]]
 then mail -s easykill $MAILLIST <$TEMPFILE4
  else print "No users forced off"
fi
cat $TEMPFILE4
```

7

Database server monitoring

In this chapter, we introduce multiple tools that are useful for monitoring and tuning the Database Environment. DB2 Universal Database (UDB) offers several tools that help you monitor and tune your database and applications.

This chapter describes the following tools that are part of DB2 UDB:

- Snapshot monitor
- Event monitor
- Explain utilities
- DB2 UDB diagnostic log
- Health Center/Memory Visualizer/DB2 Memory tracker command
- Design Advisor
- DB2 problem determination

In addition to the DB2 UDB tools, we also describe:

- DB2 Performance Expert
- IBM Tivoli Monitoring for Databases
- OS monitoring

7.1 Tools for monitoring

Monitoring helps you analyze your system. Use Monitoring if you have a transaction that is running for a long time, are experiencing scalability problems, receive user complaints about performance and you need to find the cause and develop solutions. Monitor the production environment first, and then identify the problem. Verify the fix in a test environment by running system and unit test, before you apply the fix in the production environment.

The database manager parameter, `MON_HEAP_SZ`, determines the amount of memory allocated in pages for monitoring your database system. The default value is not designed for high volume monitoring activities. In a Siebel Environment, we suggest setting it to a minimum of 512.

7.1.1 Snapshot monitor

The *snapshot monitor* captures information about the database and any connected applications at a specific time. This is one of the most important tools available for a DBA. This tool is available on all platforms. It provides detailed information about sorts, tables, locks, SQL statements, and buffer pools.

Snapshots are useful for determining the status of a database environment. Taken at regular intervals, they also provide useful information for observing trends and foreseeing potential problems.

Before running a snapshot, ensure that the switch is turned on for the information you are trying to capture. If a particular monitor switch is off, the data for that switch will not be collected. The database manager (DBM) keeps track of all the snapshot monitoring switch settings.

Table 7-1 shows the monitor switches and information they can collect. The DBM configuration parameters shown in the table are dynamic and no longer require a recycle of the DB2 UDB instance to activate the changes.

Table 7-1 Data returned by snapshot monitor

Monitor switch	DBM parameter	Information provided
BUFFERPOOL	`DFT_MON_BUFPOOL`	Number of reads and writes from/to buffer pool and disk; time taken
LOCK	`DFT_MON_LOCK`	Number of locks held; number of deadlocks; what is locked with which lock mode; lock escalations
SORT	`DFT_MON_SORT`	Number of heaps used; sort overflow; total sorts; total sort times

Monitor switch	DBM parameter	Information provided
STATEMENT	DFT_MON_STMT	Start/stop time; statement identification
TABLE	DFT_MON_TABLE	Rows read; rows written
TIMESTAMP	DFT_MON_TIMESTAMP	Timestamps
UOW	DFT_MON_UOW	Start/stop time; completion status

Note: TIMESTAMP monitor switch is on by default. You can also enable Monitor switch settings for a given session. Changes to the switch settings at the session level only affect the session for which the switch was changed.

How to take a snapshot

The example shown below collects information on all switches.

Note: Turning switches on for the session does not affect setting the DBM parameter. For example, the **db2 update monitor switches using bufferpool on** command does not update the DFT_MON_BUFPOOL DBM parameter.

1. Turn on the snapshot switches for a session using the command:

   ```
   db2 update monitor switches using table on lock on sort on uow on statement
   on bufferpool on
   ```

2. Reset the monitor counters:

   ```
   db2 reset monitor all
   ```

3. <Run workload>
4. Take snapshot using the following command:

   ```
   db2 get snapshot for all on <dbname>
   ```

5. Turn off the snapshot switches using command:

   ```
   db2 update monitor switches using table off lock off sort off uow off
   statement off bufferpool off
   ```

Information in a database snapshot

In Example 7-1 on page 169, we explain what to look for in a database snapshot in typical Siebel OLTP applications.

Example 7-1 Database Snapshot

```
Database Snapshot
```

```
Database name                                = VOL78RBA
Database path                                =
/db2/vol78rba/cat/db2inst1/NODE0000/SQL00001/
Input database alias                         = VOL78RBA
Database status                              = Active
Catalog database partition number            = 0
Catalog network node name                    =
Operating system running at database server= AIX 64BIT
...
Application connects                         = 2552

Locks held currently                         = 53523
Lock waits                                   = 181
Time database waited on locks (ms)           = 4906
Lock list memory in use (Bytes)              = 17509888
```

Note: Time database waited on locks (ms) are affected by system load. The more processes you have running, the higher this wait time value is.

```
Deadlocks detected                           = 0
Lock escalations                             = 0
```

Note: Lock escalation also can cause deadlocks. For example, suppose a read-only application and an update application both are accessing the same table. If the update application has exclusive locks on too many rows in the table, the database manager might try to escalate the locks on this table to an exclusive table lock. However, the table lock held by the read-only application will cause the exclusive lock escalation request to wait. If the read-only application requires a row lock on a row already locked by the update application, this creates a deadlock. To avoid this kind of problem, modify the update application to lock the table exclusively when it starts or increase the size of the lock list and `MAXLOCKS` configuration parameters.

Lock escalation converts row locks into table locks, therefore reducing concurrency on shared objects in the database.

```
Exclusive lock escalations                   = 0

Agents currently waiting on locks            = 0
Lock Timeouts                                = 0
Number of indoubt transactions               = 0

Total Private Sort heap allocated            = 396
Total Shared Sort heap allocated             = 0
Shared Sort heap high water mark             = 0
```

```
Total sorts                                 = 380688
Total sort time (ms)                        = 8420
Sort overflows                              = 770
Active sorts                                = 22
```

Note: The ratio of number of sort overflows with respect to the total sorts is small. For this reason, increasing the sort heap size may have little impact on performance. We recommend a value of 64 to 256 for Siebel CRM databases.

With statement switch turned on, you can capture and identify statements which are performing large numbers of sorts. These statements may benefit from additional tuning to reduce the number of sorts.

...

```
Buffer pool data logical reads              = 11891007
Buffer pool data physical reads             = 133520
Buffer pool temporary data logical reads    = 110165
Buffer pool temporary data physical reads   = 0
Asynchronous pool data page reads           = 586
Buffer pool data writes                     = 53215
Asynchronous pool data page writes          = 52721
Buffer pool index logical reads             = 25559134
Buffer pool index physical reads            = 217371
Buffer pool temporary index logical reads   = 0
Buffer pool temporary index physical reads  = 0
Asynchronous pool index page reads          = 7
Buffer pool index writes                    = 177835
Asynchronous pool index page writes         = 176847
```

Note: Increasing buffer pool size will generally improve the buffer pool hit ratio. A large buffer pool improves the performance of the database, because the pages can be accessed faster from memory than from the disk.

In conjunction with `POOL_DATA_P_READS`, `POOL_INDEX_P_READS`, and `POOL_INDEX_L_READS`, you can calculate the overall buffer pool hit ratio using the following formula:

```
1 - ((pool_data_p_reads + pool_index_p_reads)
     / (pool_data_l_reads + pool_index_l_reads))
```

In the above example, the buffer pool hit ratio is:

```
1 - ((133520 + 217371) / (18891007 + 25559134)) = .9921*100= 99.21%
```

...

```
No victim buffers available                = 412413
LSN Gap cleaner triggers                   = 0
Dirty page steal cleaner triggers          = 2968
Dirty page threshold cleaner triggers      = 1432
```

Note: DB2 UDB periodically and asynchronously writes changed data pages from buffer pool to disk. Calculate the percentage of page cleaner invocations using the formula:

```
Dirty page steal cleaner triggers / (Dirty page steal cleaner triggers +
Dirty page threshold cleaner triggers + LSN Gap cleaner triggers)
```

In the above example, the ratio is 67%. It indicates that you do not have enough page cleaners (`NUM_IOCLEANERS` database configuration parameter) defined. As a general rule of thumb, the `NUM_IOCLEANERS` should not be greater than the number of CPUs in your DB2 UDB server.

```
...

Commit statements attempted                = 95215
Rollback statements attempted              = 0
Dynamic statements attempted               = 8056522
Static statements attempted                = 190734
Failed statement operations                = 3797
Select SQL statements executed             = 6082457
Update/Insert/Delete statements executed   = 397234
DDL statements executed                    = 0

Log pages read                             = 0
Log read time (sec.ns)                     = 0.000000004
Log pages written                          = 160913
Log write time (sec.ns)                    = 149.000000004
Number write log IOs                       = 138329
Number read log IOs                        = 0
Number partial page log IOs                = 48781
```

Note: Monitor write I/Os per second for writing log data to the disk to determine if the current disk speed is adequate for logging.

```
Package cache lookups                      = 6421428
Package cache inserts                      = 3
Package cache overflows                    = 0
Package cache high water mark (Bytes)      = 83852701
Application section lookups                = 8152041
Application section inserts                = 588
```

Note: You can calculate the package cache hit ratio using the following formula:

```
1 - (Package Cache Inserts / Package Cache Lookups)
```

A ratio close to 100% indicates that most of the dynamic SQL statements utilized by the application are cached in `PCKCACHESZ` memory.

7.1.2 Event monitor

Snapshot Monitors let you "take a picture" of your system at a given time. *Event monitors* allow you to collect information about the database and any connected applications when specified events occur. This tool is available on all platforms. You can define an event monitor by the type of event or events you want to monitor. You can monitor events such as database, tables, deadlocks, table spaces, buffer pools, connections, statements, and transactions.

With DB2 UDB V8, you can store the monitor data in tables, files, or named pipes.

Note: By default, all databases have a default event monitor named `DB2DETAILDEADLOCK`, which records detailed information about deadlock events. The `DB2DETAILDEADLOCK` event monitor starts automatically when the database starts.

Use the following procedure to set up the event monitor:

1. Connect to database using the command:

   ```
   db2 connect to <dbname>
   ```

2. Make sure there is only one event monitor named `STBP` for the entire database. Drop the event monitor `STBP` using the command: (in case it already exists)

   ```
   db2 drop event monitor STBP
   ```

3. Create the directory for the event monitor to write binary files, using the command:

   ```
   mkdir c:\evmon
   ```

4. Create the event monitor with a preferred name. In our example, we name it `STBP`. Specify the types of events to be monitored, and the option to manually start the event montior. Create the event monitor using the command:

```
db2 create event monitor STBP for statements, bufferpools write to file
c:\evmon maxfiles 50 maxfilesize 1024 manualstart
```

5. Activate event monitor using the command:

```
db2 set event monitor STBP state 1
```

6. <Run workload>
7. Deactivate event monitor using the command:

```
db2 set event monitor STBP state 0
```

8. Disconnect from the database using the command:

```
db2 connect reset
```

9. The file for the event monitor is a binary stream of logical data groupings. You can format this data using the `db2evmon` command. This productivity tool reads in event records from an event monitor's files or *pipe*, then writes them to the screen (standard output).

```
db2evmon -db <dbname> -evm STBP
```

Example 7-2 shows a sample of the event monitor output. We have noted the area that you want to examine, while reading the output.

Example 7-2 Partial statement event monitor output

```
8) Statement Event ...
  Appl Handle: 73
  Appl Id: *LOCAL.ucsdb2.050428230132
  Appl Seq number: 0007

  Record is the result of a flush: FALSE
  ------------------------------------------
  Type     : Dynamic
  Operation: Describe
  Section  : 201
  Creator  : NULLID
  Package  : SQLC2E06
  Consistency Token  : AAAAAcEU
  Package Version ID  :
  Cursor   : SQLCUR201
  Cursor was blocking: TRUE
  Text     : SELECT T1.CONFLICT_ID, T1.LAST_UPD, T1.CREATED, T1.LAST_UPD_BY,
T1.CREATED_BY, T1.MODIFICATION_NUM, T1.ROW_ID, T1.ROW_ID, T1.PARTNER_BRANCH,
...
FROM siebel.S_ASSET T1 INNER JOIN siebel.S_ASSET_POSTN T2 ON T1.PR_POSTN_ID =
T2.POSITION_ID AND T1.ROW_ID = T2.ASSET_ID INNER JOIN siebel.S_PARTY T3 ON
T2.POSITION_ID = T3.ROW_ID
...
WHERE (T1.TYPE_CD = 'Fin Account') FOR FETCH ONLY OPTIMIZE FOR 1 ROW
```

```
  ------------------------------------------
  Start Time: 04/29/2005 00:22:10.127557
  Stop Time:  04/29/2005 00:23:52.141291
  Exec Time:  102.013734 seconds
  Number of Agents created: 1
  User CPU: 5.830000 seconds
  System CPU: 1.340000 seconds
```

Note: User CPU and System CPU utilizations are high. The execute time is close to two minutes, which is rather long for applications such as a call center.

```
Fetch Count: 1
  Sorts: 0
  Total sort time: 0
  Sort overflows: 0
Rows read: 59787
Rows written: 0
  Internal rows deleted: 0
  Internal rows updated: 0
  Internal rows inserted: 0
```

Note: The high value in Rows read indicates a potential problem. This might be an indication that statistics are not up to date or proper indexes are missing.

```
Bufferpool data logical reads: 59787
  Bufferpool data physical reads: 6877
  Bufferpool temporary data logical reads: 0
  Bufferpool temporary data physical reads: 0
  Bufferpool index logical reads: 241343
  Bufferpool index physical reads: 2512
  Bufferpool temporary index logical reads: 0
  Bufferpool temporary index physical reads: 0
  SQLCA:
   sqlcode: 0
   sqlstate: 00000
```

7.1.3 Explain utilities

The *Explain* tool provides the detailed information about the access plan chosen by the DB2 optimizer while executing SQL statements. This tool is available on all platforms.

Once you have isolated the problem queries, generate the explain plans for the queries involved.

How to use the Explain tool

Following is the procedure to explain SQL statements:

1. Connect to database using the command:

   ```
   db2 connect to <dbname>
   ```

   For example,

   ```
   db2 connect to VOL78RBA
   ```

2. The Explain tables must be created before explain can be invoked. The `EXPLAIN.DDL` script file is located in `<db2instance home>/sqllib/misc directory`. Create explain tables using the command:

   ```
   db2 -tvf EXPLAIN.DDL
   ```

3. Create the explain plan for the SQL statement using command:

   ```
   db2 explain plan for "< place your SQL statement here>"
   ```

   For example,

   ```
   db2 explain plan for "select row_id from SIEBEL.S_CONTACT where SRC_ID =
   '1+13A+11' "
   ```

4. Use the *db2exfmt* tool to format the contents of the explain tables into a readable output using command:

   ```
   db2exfmt -d <database name> -e <schema> -g TIC -w -1 -n % -s % -# 0 -o
   <output file>
   ```

   For example,

   ```
   db2exfmt -d VOL78RBA -e ucsdb2 -g TIC -w -1 -n % -s % -# 0 -o explain.out
   where,
   -d dbname Name of the database containing packages.
   -e schema Explain table SQL schema.
   -g TIC
   -g Graph plan.
   T Include total cost under each operator in the graph.
   I Include I/O cost under each operator in the graph.
   C Include the expected output cardinality of each operator in the graph.
   -w timestamp Explain time stamp.
   -l Respect case when processing package names.
   -n name Name of the source of the explain request (SOURCE_NAME).
   -s schema SQL schema or qualifier of the source of the explain request
   (SOURCE_SCHEMA).
   -# sectnbr Section number in the source. To request all sections, specify
   zero.
   -o outfile Output file name.
   ```

Example 7-3 shows the text-based Explain plan we generated using the above steps.

Example 7-3 Text-based Explain plan:

```
DB2 Universal Database Version 8.1, 5622-044 (c) Copyright IBM Corp. 1991, 2002
Licensed Material - Program Property of IBM
IBM DATABASE 2 Explain Table Format Tool



******************** EXPLAIN INSTANCE ********************

DB2_VERSION: 08.02.1
SOURCE_NAME: SQLC2E06
SOURCE_SCHEMA: NULLID
SOURCE_VERSION:
EXPLAIN_TIME: 2005-07-21-22.05.10.509412
EXPLAIN_REQUESTER: UCSDB2

Database Context:
----------------
   Parallelism: None
   CPU Speed: 5.628769e-07
   Comm Speed: 100
   Buffer Pool size: 120000
   Sort Heap size: 2048
   Database Heap size: 4800
   Lock List size: 10000
   Maximum Lock List: 30
   Average Applications: 5
   Locks Available: 192000
```

Note: The optimizer uses the above configuration parameters to calculate the least cost-based access plan. For example, access plans can differ due to disk speed, speed, and number of CPUs.

The DB2 optimizer makes its decisions based on database statistics that are stored in the system catalog.

```
Package Context:
---------------
   SQL Type: Dynamic
   Optimization Level: 3
   Blocking: Bock All Cursors
   Isolation Level: Cursor Stability
```

```
---------------- STATEMENT 1  SECTION 203 ----------------
   QUERYNO: 1
   QUERYTAG:
   Statement Type: Select
   Updatable: No
   Deletable: No
   Query Degree: 1

Original Statement:
------------------
select ROW_ID
from SIEBEL.S_CONTACT
where SRC_ID = '1+13A+11'


Optimized Statement:
-------------------
SELECT Q1.ROW_ID AS "ROW_ID"
FROM SIEBEL.S_CONTACT AS Q1
WHERE (Q1.SRC_ID = $VARGRAPHIC_string$)

Access Plan:
-----------
   Total Cost: 75.0443
   Query Degree: 1

Rows
           RETURN
           (   1)
            Cost
             I/O
             |
           2.04807
           FETCH
           (   2)
           75.0443
              3
          /---+---\
     2.04807    3.54116e+06
     IXSCAN   TABLE: SIEBEL
     (   3)      S_CONTACT
     50.0379
        2
       |
   3.54116e+06  ---> Number of rows in the table SIEBEL.S_CONTACT about
3,541,160
 INDEX: SIEBEL
  S_CONTACT_F4  ---> DB2 Optimizer is utilizing an index SIEBEL.S_CONTACT_F4
Base Table Schema:SIEBEL Base Table Name: S_CONTACT Columns in index: SRC_ID
```

Note: If table SIEBEL.S_CONTACT did not have correct index defined or if statistic on the table was not current, the DB2 Optimizer would have done Table Scan instead of index scan. In example here, table scan for the table SIEBEL.S_CONTACT with 3.5 Million rows in this scenario would have been very expensive.

...

The db2exfmt and Visual Explain utilities require the DB2 Explain tables to be created for each user who is executing the explain commands. The db2expln and dynexpln utilities allow dynamic SQL to be explained without using DB2 Explain tables. Further details about the db2expln, dynexpln, and Visual Explain utilities can be found at the DB2 Information Center:

http://publib.boulder.ibm.com/infocenter/db2help/index.jsp

7.1.4 DB2 UDB diagnostic log

DB2 UDB provides a diagnostic capturing mechanism. It writes information into a file called *db2diag.log*. By default, this file is located in:

- On UNIX platforms:

  $INSTHOME/sqllib/db2dump directory

  Where INSTHOME is the database instance home directory

- On Intel platforms:
  - If the DB2INSTPROF environment variable is not set:

    x:\Program Files\IBM\SQLLIB\%DB2INSTANCE%

  - If DB2INSTPROF is set:

    x:\%DB2INSTPROF%\%DB2INSTANCE%

  Where x:\SQLLIB is the drive reference and directory specified in the DB2PATH registry variable.

In most situations, the default paths are adequate and do not need to be changed.

In db2diag.log, you can find informational messages, such as parameters that need to be changed, errors that occurred, and lock escalations. The value of the database manager parameter DIAGLEVEL defines the level of diagnostic information that will be captured. The levels are:

- 0: No diagnostic data captured

- 1: Severe errors only
- 2: All errors
- 3: All errors and warning (default)
- 4: All errors, warnings, and informational messages

Level 3 is sufficient for most implementations. Using level 4, you can capture additional information, such as the Java environment that is used by DB2. The IBM Support also needs this information for problem determination. The `db2diag.log` file is manually managed by the administrator. This file can be deleted without impacting the DB2 UDB system. DB2 UDB will automatically recreate the file if it does not exist.

7.1.5 Health Center/Memory Visualizer/DB2MTRK

This section provides a brief description of the following tools:

- Health Center
- Memory Visualizer
- DB2 Memory tracker command

Heath Center

The Health Center is a graphical interface tool that constantly monitors the health of the instance and database. With the tool you can:

- Set up thresholds for key performance and health indicators for performance.
- Notify the database administrator when these thresholds are exceeded.

This tool is available on all platforms.

If a predefined threshold has been exceeded (for example, a particular table space has a utilization rate of 80% or more), the Health Monitor will raise an alert.

When an alert is raised, the Health Center tool sends alert notifications by e-mail or to a pager address, and takes preconfigured actions if configured.

How to start Health Center

The Health Center can be invoked using the command line or GUI interface.

- To start the Health Center from the command line, use command:

  ```
  db2hc
  ```

- To start Health Center on Windows:

  Start → **Programs** → **IBM DB2** → **Monitoring Tools** → **Health Center**

Memory Visualizer

The *Memory Visualizer* tool graphically displays the memory used by DB2 UDB. It allows database administrators to monitor the memory-related performance of an instance and all of its databases organized in a hierarchical tree. This tool is available on all platforms.

With the Memory Visualizer, you can:

- Show or hide data in various columns of the memory utilization on selected components for a DB2 instance and its databases.
- Plot the graph in real time for the selected components.
- Change settings for individual memory components by updating configuration parameters.
- Load performance data from a file into a Memory Visualizer window.
- Save the performance data.

How to start Memory Visualizer

The Memory Visualizer can be invoked using the command line or GUI interface.

- To start the Memory Visualizer from the command line, use the command:

  db2memvis

- To start Memory Visualizer on Windows:

  Start → **Programs** → **IBM DB2** → **Monitoring Tools** → **Memory Visualizer**

DB2 Memory tracker command

db2mtrk is a text-based memory tracker tool, equivalent to Memory Visualizer.

How to use db2mtrk

The DB2 Memory tracker can be invoked using the command line as seen below.

- Run the following command while a backup is running to monitor the memory usage:

  db2mtrk -i -p -v

Example 7-4 shows you the snapshot of db2mtrk while the backup is running. During the backup operation, you can see that the backup/restore memory is allocated.

Example 7-4 Memory in use while backup is running

```
C:\Program Files\IBM\SQLLIB\BIN>db2mtrk -i -p -v
Tracking Memory on: 2005/07/27 at 13:31:54
```

```
Memory for instance

   Backup/Restore/Util Heap is of size 18268160 bytes
   Package Cache is of size 131072 bytes
   Catalog Cache Heap is of size 65536 bytes
   Buffer Pool Heap is of size 1179648 bytes
   Buffer Pool Heap is of size 655360 bytes
   Buffer Pool Heap is of size 393216 bytes
   Buffer Pool Heap is of size 262144 bytes
   Buffer Pool Heap is of size 196608 bytes
   Lock Manager Heap is of size 278528 bytes
   Database Heap is of size 3604480 bytes
   Database Monitor Heap is of size 16384 bytes
   Other Memory is of size 7585792 bytes
   Total: 32636928 bytes

No active agents
```

- Issue the command, after the backup has finished:

  db2mtrk -i -p -v

Example 7-5 shows you the snapshot of db2mtrk when the backup has finished. You can see that the backup/restore memory is released.

Example 7-5 Memory when a backup has finished

```
C:\Program Files\IBM\SQLLIB\BIN>db2mtrk -i -p -v
Tracking Memory on: 2005/07/27 at 13:45:45

Memory for instance

   Database Monitor Heap is of size 16384 bytes
   Other Memory is of size 7438336 bytes
   Total: 7454720 bytes

No active agents
```

Further details about the Health Center, Memory Visualizer, and db2mtrk can be found at DB2 Information Center:

http://publib.boulder.ibm.com/infocenter/db2help/index.jsp

7.1.6 Design Advisor

The DB2 UDB *Design Advisor* tool can help you significantly improve your workload performance. The task of selecting which indexes, MQTs, clustering dimensions, or partitions to create for a complex workload can be quite complex.

The tool identifies all of the objects that might need to be tuned for performance during your workload. The tool is available on all platforms. The tool will generate recommendations for:

- New indexes
- New materialized query tables (MQTs)
- Conversion to multidimensional clustering (MDC) tables
- Repartitioning tables
- Deletion of indexes and MQTs unused by the specified workload (through the GUI tool)

The recommendations are based on one or more SQL statements provided by the user. A group of related SQL statements is known as a *workload*. Users can rank the importance of each statement in a workload and specify the frequency at which each statement in the workload is to be executed.

The Design Advisor outputs a DDL CLP script that includes `CREATE INDEX`, `CREATE SUMMARY TABLE (MQT)`, and `CREATE TABLE` statements to create the recommended objects.

You can start the Design Advisor from the Control Center or from the command line. For example, start the Design Advisor from the command line using the command:

```
db2advis -d <dbname> -S "<SQL Statement>"
```

You can find further details about the Design Advisor at the DB2 Information Center:

http://publib.boulder.ibm.com/infocenter/db2help/index.jsp

In Example 7-6, we show you how to use *Design Advisor* from the command line to improve the performance of a particular statement.

Example 7-6 Design Advisor command

```
db2advis -d qa78rba -s "select firstname, lastname from siebel.s_usertest where
empno between'000010'and '000060'"

Using user id as default schema name. Use -n option to specify schema
execution started at timestamp 2005-07-25-14.46.06.054000
Recommending indexes...
total disk space needed for initial set [   0.009] MB
total disk space constrained to         [   4.427] MB
Trying variations of the solution set.
Optimization finished.
  1  indexes in current solution
 [ 26.0000] timerons  (without recommendations)
```

```
[  0.0307] timerons  (with current solution)
[99.88%] improvement
```

Note: The tool indicates disk space necessary for the indexes and estimates the percentage improvement for the workload.

```
--
--
-- LIST OF RECOMMENDED INDEXES
-- ===========================
-- index[1],    0.009MB
   CREATE INDEX "SIEBEL"."IDX507252146100000" ON "SIEBEL"."EMPLOYEE" ("EMPNO"
ASC, "LASTNAME" ASC, "FIRSTNME" ASC) ALLOW REVERSE SCANS ;
   COMMIT WORK ;
   RUNSTATS ON TABLE "SIEBEL"."S_TESTUSER" FOR INDEX
"SIEEBEL"."IDX507252146100000" ;
   COMMIT WORK ;
```

Note: The tool gives you DDL for creating indexes and generating statistics for tables.

7.1.7 DB2 UDB problem determination utility

The DB2 UDB utility *db2pd* is a problem determination tool which collects extremely detailed information about specific variables from DB2 memory sets. `db2pd` runs fast, since it does not acquire any locks or latches and runs outside of the DB2 engine resources. It is a standalone utility shipped with the DB2 engine. It can be executed from the command line with optional interactive mode. It gives the database administrator a close view into the DB2 engine. It helps users monitor and troubleshoot the potential problems. This tool is available on all platforms.

`db2pd` tool collects information without acquiring any locks. Based on the application business process, it is possible to retrieve information that is changing while `db2pd` is collecting information. There are two benefits to collecting information without latching, which include faster retrieval and no competition for engine resources.

`db2pd` provides many options to display information about the database server. Use this tool for troubleshooting, problem determination, monitoring, and performance tuning of the database and connected applications.

Table 7-2 shows the `db2pd` option, the description of the option, and its corresponding scope of an instance or a database.

Table 7-2 Description and scopes of `db2pd` options

db2pd option	**Description**	**Scope**
`agents`	Returns information about agents	Instance
`applications`	Returns information about applications	Database
`bufferpools`	Returns information about buffer pools	Database
`catalogcache`	Returns information about catalog cache	Database
`dbcfg`	Returns information about database configuration parameters settings	Database
`dbmcfg`	Returns information about database manager configuration parameters settings	Instance
`dynamic`	Returns information about the execution of dynamic SQL	Database
`fcm`	Returns information about fast communication manager	Instance
`help`	Returns help information of db2pd command	N/A
`logs`	Returns information about the logs	Database
`locks`	Returns information about the locks	Database
`mempools`	Returns information about memory pools	Both
`memsets`	Returns information about memory sets	Both
`osinfo`	Returns information about operating systems	Instance
`recovery`	Returns information about recovery activity	Database
`report`	Returns information about cached SQL statements that were reoptimized using REOPT ONCE option applications	Database
`reorg`	Returns information about tables reorganization	Database
`static`	Returns information about the execution of static SQL and packages	Database
`sysplex`	Returns information about the list of servers associated with the database alias for all databases or for a particular database	Instance

db2pd option	Description	Scope
tablespace	Returns information about the table spaces	Database
tcbstats	Returns information about tables and indexes	Database
transaction	Returns information about an active transaction	Database
version	Returns information about the current DB2 version and level	Instance

How to use db2pd Tool

db2pd tool can be invoked with options from the command line:

- Get information about the current DB2 UDB level and current operating system using the command:

  ```
  db2pd -version -osinfo
  ```

- Get the DB2 UDB memory information once every two seconds for five times using the command:

  ```
  db2pd -mempools -repeat 2 5
  ```

- Determine who is holding a lock on the database and send output to filename *lock.out* using the command:

  ```
  db2pd -database qa78rba -locks -transaction -agents -file lock.out
  ```

Further details about the db2pd can be found at the DB2 Information Center:

http://publib.boulder.ibm.com/infocenter/db2help/index.jsp

or IBM white paper *The db2pd Tool:*

http://www.ibm.com/developerworks/db2/library/techarticle/dm-0504poon2/

7.2 DB2 Performance Expert

DB2 Performance Expert (DB2 PE) is a powerful tool to resolve DB2 performance problems. It is a workstation-based tool for monitoring, performance analysis, and tuning the database and its applications. It allows you to monitor applications, system statistics, and system parameters in real time and in historical mode. The tool also helps you analyze performance bottlenecks, and it provides tuning recommendations to improve overall performance.

DB2 PE allows you to manage a heterogeneous mix of DB2 systems from a single user interface. It integrates performance monitoring, reporting, buffer pool analysis, and a Performance Warehouse function. It supports a DB2 server

running on Windows, AIX, HP-UX, z/OS®, Linux, Linux on zSeries®, and Sun Solaris operating environments.

DB2 PE Tool provides many capabilities:

- Determining whether or not an index would improve performance.
- Checking the need to reorganize tables.
- Checking if there are sufficient agents to handle the workload.
- Examining frequently used SQL statements from package cache.
- Helping analyze sort performance, resolve lock conflicts, analyze buffer pools, and monitor system health.
- Monitoring and analyzing the performance of DB2 and DB2 applications to simplify performance tuning.
- Including a database for storing performance data and analysis tools that lets you:
  - Save DB2 snapshot and event monitoring data (for SQL, database, and buffer pool activities) and create reports for investigation and trend analysis.
  - Configure and schedule the report and load process from the workstation GUI functions.
  - Define and apply analysis functions to identify performance bottlenecks.
- Providing expert analysis, a real-time online monitor, and a wide range of reports for analyzing and optimizing DB2 applications and SQL statements. The online monitor includes an extended system overview, graphs in detailed panels, improved navigation, and filtering.
- Supplying exception conditions to anticipate emerging DB2 performance and availability problems.
- Allowing you to generate buffer pool reports in multiple formats, including tables, pie charts, and diagrams.
- Providing full support for DB2 Enterprise Server Edition with the Data Partitioning Feature, enabling highly sophisticated performance monitoring of large enterprise systems.
- Monitoring DB2 Connect™ gateways, including databases and system-related information.

Further details about DB2 PE can be found at the following Web sites:

http://www-128.ibm.com/developerworks/db2/library/techarticle/dm-0409schuetz

http://www.redbooks.ibm.com/abstracts/SG246470.html?Open

7.3 Tivoli Monitoring

IT infrastructure is literally the heart of business operations, so ensuring the reliability and availability of IT resources is vital for business success. Given the increased complexity of e-business infrastructures, intelligent performance, and availability management tools are essential for proactive identification and resolution of IT problems before they impact business performance. These tools enable you to cost-effectively monitor individual resource performance while simultaneously gauging resource function and availability across your heterogeneous operating environment.

Ensuring peak performance and availability cost-efficiently through intelligent management software solutions from Tivoli will help you meet and exceed both internal and external service level agreements and reduce your total cost of ownership.

IBM Tivoli systems and applications monitoring solutions enable you to deploy a single monitoring solution for most or all of the resources in your environment. This allows your system monitors to share a common reporting engine, graphical user interface, and data repository. IBM Tivoli Monitoring provides monitoring for essential system resources, to detect bottlenecks and potential problems, and to automatically recover from critical situations. Tivoli Monitoring saves system administrators from manually scanning through extensive performance data before they can solve problems. Using industry best practices, Tivoli Monitoring can provide immediate value to the enterprise.

IBM Tivoli Monitoring for Databases helps ensure the availability and optimal performance of DB2, Oracle, Informix®, and Microsoft SQL Server database servers.

Through the use of best practices incorporated within IBM Tivoli Monitoring for Databases, the typical database administrator dilemma of determining what to monitor, when to monitor, and how to interpret and act upon the monitoring results is eliminated, leaving more time for the administrator to focus on more complex business-critical tasks. IBM Tivoli Monitoring for Databases provides routine, consistent monitoring that helps anticipate and correct problems before database performance and customer confidence decline.

All data captured by monitoring the databases is delivered through an intuitive user interface and made available through historical and real-time reports. IBM Tivoli Monitoring for Databases provides an out-of-the-box set of monitors for quick deployment and activation leveraging IBM best practices. Custom monitors, thresholds, and tasks can also be defined by the database administrator.

Through the implementation of IBM Tivoli Monitoring for Databases, database administrators are alerted when key performance and resource allocation

problems are detected. This solution helps customers maximize their return on investment through increased efficiency of their IT staff, improved compliance to service-level objectives, and reduced cost of database system administration and deployment.

Tivoli monitors preset thresholds and takes automatic, corrective actions. Tivoli provides end-to-end management solutions for databases from a single centralized console. Tivoli can e-mail or page the administrator when Tivoli detects a problem such as file system size increasing above a threshold, or a process going down.

For more information about Tivoli Monitoring, access the following Web sites:

http://www.ibm.com/tivoli/

http://www.ibm.com/software/tivoli/products/monitor-db/

The typical processes and file systems monitored by Tivoli for a Siebel implementation are:

Table 7-3 lists the AIX processes and file systems monitored by Tivoli for a Siebel Gateway Name Server.

Table 7-3 Siebel Gateway Name Server monitored processes and file systems

Siebel Gateway Name Server	Process status/file system threshold
siebsvc – s gtwyns	process up or down
/siebel	file System threshold of 90%
/siebelfs	file System threshold of 90%

Table 7-4 lists the AIX processes and file systems monitored by Tivoli for the Siebel Application Server.

Table 7-4 Siebel Application Server monitored processes and file systems

Siebel Application Server	Process status/file System threshold
siebsvc -s siebsrvr	process up or down
/siebel	file system threshold of 90%

Table 7-5 lists the AIX processes and file systems monitored by Tivoli for Siebel Web Server.

Table 7-5 Siebel Web Server monitored processes and file systems

Siebel Web Server	Process status/file system threshold
/usr/IBMIHS/bin/httpd	process up or down
/usr	file system threshold of 90%
/siebel	file system threshold of 90%

Table 7-6 lists the AIX processes and file systems monitored by Tivoli for DB2 UDB.

Table 7-6 DB2 UDB database processes and file systems

DB2 UDB database	Process status/file system threshold
db2sysc	process up or down
db2wdog	process up or down
db2 filesystems	file system threshold of 90%
/db2logs	file system threshold of 90%
/db2logarch	file system threshold of 90%

Table 7-7 lists the services and disk sizes monitored by Tivoli for Siebel Enterprise and DB2 UDB database on Windows operating systems.

Table 7-7 Siebel Enterprise and DB2 UDB database on Windows

Siebel Enterprise/DB2 UDB database	Service/Disk size threshold
Siebel Gateway Name Server installation disk	disk size threshold of 90%
Siebel Application Server installation disk	disk size threshold of 90%
Siebel File System disks	disks size threshold of 90%
Siebel Gateway Name Server service	service up or down
Siebel Application Server service	service up or down
DB2-InstanceName-NodeNumber service	service up or down
DB2 License Server service	service up or down
DB2 Security Server service	service up or down
IIS admin service or IBMIHS service	service up or down

Example 7-7 shows a sample Tivoli Monitoring report on an AIX server.

Example 7-7 Sample Tivoli Monitoring Report

```
Profile Monitor      Arg         Last      Bad                 Good

kanaga daemon       clinfo        up   critical -> "down"   warning -> "up"
kanaga daemon       clsmuxpd      up   critical -> "down"   warning -> "up"
kanaga daemon       clstrmgr      up   critical -> "down"   warning -> "up"
kanaga daemon       db2sysc       up   critical -> "down"   warning -> "up"
kanaga daemon       db2wdog       up   critical -> "down"   warning -> "up"
kanaga daemon       inetd         up   critical -> "down"   warning -> "up"
kanaga DaemonStatus db2sysc       up   critical -> "down"   warning -> "up"
kanaga diskusedpct  /             35   critical ->> 95      warning -<< 95
kanaga diskusedpct  /aixdba       78   critical ->> 95      warning -<< 95
kanaga diskusedpct  /db2/fs2      24   critical ->> 90      warning -<< 90
kanaga diskusedpct  /db2/fs5      52   critical ->> 60      warning -<< 90
kanaga diskusedpct  /db2logarch   19   critical ->> 90      warning -<< 90
kanaga diskusedpct  /db2logs      69   critical ->> 90      warning -<< 90
kanaga diskusedpct  /home         51   critical ->> 95      warning -<< 95
kanaga diskusedpct  /opt/Tivoli   41   critical ->> 95      warning -<< 85
kanaga diskusedpct  /siebel       21   critical ->> 95      warning -<< 95
kanaga diskusedpct  /siebelfs     27   critical ->> 95      warning -<< 95
kanaga diskusedpct  /tmp          20   critical ->> 95      warning -<< 95
kanaga diskusedpct  /ucsdb2       19   critical ->> 90      warning -<< 90
kanaga diskusedpct  /usr          86   critical ->> 98      warning -<< 98
kanaga diskusedpct  /var          38   critical ->> 95      warning -<< 95
```

7.4 OS monitoring

This section provides a brief description of available monitoring tools on UNIX and Windows platforms.

7.4.1 Monitoring tools on the UNIX platform

This section provides a brief description of several UNIX commands available for monitoring operating systems.

In Table 7-8, we show you commonly used UNIX (`vmstat`, `iostat`) and AIX (`nmon`, `svmon`, and `vmtune`) tools to monitor various resources.

Table 7-8 Suggested tool usage

Performance problem	Suggested tools
High CPU utilization	vmstat, nmon
Not enough memory	vmstat, vmtune, nmon, svmon

Performance problem	Suggested tools
Paging and/or swapping	vmstat, vmtune, nmon
Disk or Filesystem	iostat, nmon

This section describes the command and the sample output of the following topics:

- vmstat
- vmtune
- nmon
- iostat
- svmon

vmstat

The *vmstat* command provides compact information about various system resources. The `vmstat` command reports statistics about kernel threads in the run and wait queue, memory, paging, disks, interrupts, system calls, context switches, and CPU activity. The reported CPU activity is a percentage breakdown of user mode, system mode, idle time, and waits for disk I/O.

`vmstat` at 5 second intervals using command:

```
vmstat 5
```

In Example 7-8 on page 192, we show you the output from `vmstat` at 5 second intervals.

`r` (run queue) and `b` (wait queue) show the number of threads placed in the relevant queue per second.

The `cpu` columns show the percentage of user, system, idle, and wait I/O time.

If the number of memory pages in the '`fre`' column gets below a certain value, the system starts paging out memory to get more free pages.

Example 7-8 output from vmstat at 5 seconds interval

```
System Configuration: lcpu=4 mem=4096MB
kthr     memory             page              faults          cpu
----- ------------ ------------------------ ---------------- -----------
 r  b   avm   fre  re  pi  po  fr   sr   cy  in   sy   cs    us sy id wa
 6  6 520476  491   0   0   0 1031  928   0 445  714   367   29 27 23 21
 3 21 520586  520   0   0   0 7308 9389   0 5163 30425 13427 18 18  7 57
 3 15 520586  504   0   0   0 7039 16774  0 4955 28656 14262 17 17  9 56
 3 19 520476  490   0   0   0 7449 42387  0 4587 26876 13662 17 19  9 55
 4 18 520476  497   0   0   0 6147 12357  0 4342 21762 12159 17 18  7 57
```

```
 3 23 520476  508   0   0   0 7982 19154  0 4571 20997 11919 12 13  7 68
 2 16 520500  504   0   0   0 7261 81258  0 4426 31792 15226 17 21 10 51
 2 20 520559  486   0   0   0 7418 83479  0 4351 27750 13686 16 18  8 58
 3 24 520805  531   0   0   0 8114 11710  0 4415 29698 12140 17 23  6 54
 3 19 520267  506   0   0   0 7453 35372  0 4841 32532 14798 20 20  7 53
```

Note: Note the high number of blocked threads and high CPU wait time.

The ‘r’ column of `kthr`, shows the average number of kernel threads that are in the run queue per second. This field indicates the number of threads that can be run.

The ‘b’ column of `kthr` details an average number of kernel threads in the wait queue per second. These threads are waiting for resources or I/O. Threads are also located in the wait queue when waiting for one of their thread pages to be paged in. This value is usually near zero. But if the run-queue value increases, the wait-queue normally also increases.

The ‘wa’ column of CPU details the percentage of time the CPU was idle with pending local disk I/O. A ‘wa’ value over 25 percent could indicate that the disk subsystem might not be balanced properly, or it might be the result of a disk-intensive workload.

---

vmtune

The *vmtune* displays and dynamically modifies various kernel parameters.

Invoke `vmtune` with output using as follows:

```
/usr/samples/kernel/vmtune -a
```

Example 7-9 shows output of the `vmtune -a` command.

Example 7-9 Output from `vmtune -a`

```
hd_pendqblked = 0
```

Note: The Logical Volume Manager (LVM) uses a construct called a "pbuf" to control a pending disk I/O. A non-zero value indicates that LVM had to wait for pbufs.

```
psbufwaitcnt = 0
fsbufwaitcnt = 2996
```

Note: The new counters are incremented whenever a file system buffer structure (bufstruct) was not available and the Virtual Memory Manager puts a thread on the VMM wait list.

```
rfsbufwaitcnt = 0
xpagerbufwaitcnt = 0
```

Note: This value is incremented whenever a bufstruct on an Enhanced JFS file system is not available. If the kernel must wait for a free bufstruct, it puts the process on a wait list before the start I/O is issued, and will wake it up once a bufstruct has become available. It may be appropriate to increase if striped logical volumes or disk arrays are being used.

nmon

The *nmon* displays a variety of system information such as CPU, disk, memory utilization, network throughput, various kernel parameters, and so on.

In Example 7-10, we show you CPU utilization and memory utilization at the 2 second refresh rate.

Example 7-10 Sample `nmon` *output*

```
scripts/nmon64 v8d [H for help]  Hostname=kanaga  Refresh=2.0secs 11:04.23 CPU
Utilisation                +-------------------------------------------------+
CPU    User%  Sys% Wait% Idle|

 0      1.5   4.0   0.0  94.5|s>                                              |
 1      2.0   0.5   0.0  97.5|U>                                              |
 2      1.0   3.5   0.0  95.5|s>                                              |
 3      0.0   0.5   0.0  99.5|>                                               |

+-------------------------------------------------+
        1.1   2.2   0.0  96.6|s>                                              |

+-------------------------------------------------+
Memory Use  Physical    Virtual    Paging pages/sec  In     Out  VM parameters
% Used        19.5%       0.6%      to Paging Space   0.0    0.0 numperm   2.2%
% Free        80.5%      99.4%      to File System    3.5    0.0 minperm  18.8%
MB Used      800.3MB      2.9MB    Page Scans         0.0        maxperm  75.2%
MB Free     3295.6MB    509.1MB    Page Cycles        0.0        minfree 120
Total(MB)   4095.9MB    512.0MB    Page Reclaim       0.0        maxfree 128
```

iostat

The *iostat* command is the fastest way to get a first impression, whether or not the system has a disk I/O-bound performance problem. The tool also reports CPU statistics. The `iostat` command displays statistics for disk, adapter, and system throughput.

In Example 7-11, we show you disk activities. The `% tm_act` column lists the percentage of time the disk was busy which represents bandwidth utilization. The `tps` column shows the number of transfers per seconds on each disk.

Invoke `iostat` at 5 second intervals as following:

```
iostat 5
```

Example 7-11 Output from `iostat` at 5 second intervals

```
System configuration: lcpu=4 disk=9

tty:      tin         tout   avg-cpu:  % user    % sys     % idle    % iowait
          0.5         13.8                0.1       0.1       99.7       0.1

Disks:        % tm_act     Kbps      tps    Kb_read   Kb_wrtn
hdisk2           0.4       3.6       0.9      38230    131880
hdisk0           0.1       8.2       0.2     231690    161052
hdisk1           0.0       0.1       0.0        634      6100
dac0             0.0       0.0       0.0          0         0
dac0-utm         0.0       0.0       0.0          0         0
dac1             0.0       6.1       0.1     290363       184
dac1-utm         0.0       0.0       0.0          0         0
hdisk3           0.0       6.1       0.1     290363       184
cd0              0.0       0.0       0.0          0         0
```

svmon

The *svmon* displays various memory statistics for an individual process or selected processes.

In Example 7-12 on page 196, we show you the memory statistics for process ID (PID).

Note: Use **`ps -ef`** command to display PID.

Invoke `svmon` as following:

```
svmon -P 188590
```

Example 7-12 Output from `svmon -P 188590`

```
Pid    Command  Inuse      Pin     Pgsp  Virtual 64-bit Mthrd  16MB
188590 iostat   16226     7644      178    15065      N     N     N

Vsid   Esid Type Description            PSize  Inuse   Pin Pgsp Virtual
0         0 work kernel                  s      11344  7641   94 11424
1e42e     - clnt /dev/hd1:8258           s      3205      0    -     -
1c09d     d work shared library text     s      1517      0   84  3498
c4dc      2 work process private         s       106      3    0   106
1a4ea     f work shared library data     s        37      0    0    37
134c3     1 clnt code,/dev/hd2:57827     s        16      0    -     -
6496      - clnt /dev/hd2:37594          s         1      0    -
```

7.4.2 Monitoring tools on the Windows platform

DB2 UDB on the Windows platform is well integrated with common Windows utilities. Microsoft provides several performance monitoring tools with Windows 2000. Additional tools and information are available in the Windows Resource Kit at the following Web site:

http://www.microsoft.com/windows2000/techinfo/reskit/default.mspx

Table 7-9 lists tools commonly used to monitor performance problems.

Table 7-9 Suggested tool usage on Windows

Performance problem	Suggested tools
High CPU utilization	Task Manager, Performance Monitor
Not enough memory	Task Manager, Performance Monitor
Paging and/or swapping	Performance Monitor
Disk or filesystem	Task Manager, Performance Monitor

This section provides brief overviews of the following topics:

- Performance Monitor
- Task Manager

Performance Monitor

Performance Monitor provides a graphical display and logging of a number of Windows 2000 components, such as processor, memory, and so on. The components are organized in the list of “Performance Object:” which has many associated counters. For example, the “Process” performance object has counters such as %Processor Time, %User Time, and so on. The “DB2

Database Manager:” performance object has counters such as Number of MAX AGENT overflow, Number of idle agents, and so on.

Task Manager

The *Task Manager Processes* tab provides useful information about processes running on the system. By default, the Processes tab displays Image Name, PID, CPU, CPU Time, and Mem Usage columns. The additional columns can be selected through the View/Select Columns Taskbar options.

The Task Manager can be invoked by:

```
Cntrl+Alt+Delete select Task manager
```

8

Performance tuning

DB2 UDB V8.2 has over 150 new features in application development, “lights-out” (remote management) operations, high availability (HADR), manageability, security, and so on. The following play a key role in performance tuning:

- Instance configuration
- DB2 registry variables
- Database configuration
- Database design
- System (environment) configuration
- Application design

We take you through Siebel recommended values and provide tuning consideration for DB2 UDB database and operation systems to make it perform better in a Siebel environment. The application design is not addressed in this redbook since it is controlled by Siebel and any changes to application should only be made by Siebel.

8.1 Performance tuning introduction

Performance tuning is a task that needs to be carried out after the system is set up and continued as long as your system is up and running. Performance tuning starts from system planning time at which time a set of performance requirements will be set. Once the system is up and running, constant system monitoring provides a foundation for ongoing system tuning.

Performance tuning should be done carefully; do not tune arbitrarily. Tune for specific components in the enterprise environment. Consider the following when you tune:

- **Do you have the latest fixes?**

  A number of performance issues arise from hardware, operating systems, and their patches. Check the Siebel Product Documentation section of Support Web, which provides the latest editions of Bookshelf, documentation updates, alerts on new issues with suggested resolutions, and release notes. It includes the "System Requirements and Supported Platforms" (SRSP). For the latest recommended hot fixes and the fixes for which Siebel is certified, visit Siebel support Web site:

  http://supportweb.siebel.com

- **Check and balance**

  Make sure you have documented processes on what you are monitoring and changing. You should track your changes and the improvements. Do not make too many changes at once, because it will be hard to locate and understand which changes helped and which ones hurt. Always track your changes. Tracking your changes helps both in the short and long term. Tracking these changes can provide valuable information to your team members. In the short term, it will let you trace back to any changes that need to be rolled back due to their negative impact on performance. In the long term, it will help you determine whether it is time to buy new hardware or just upgrade.

- **Tune with a reason**

  Tuning for the sake of tuning is a bad decision. Tuning arbitrarily can be costly. If you can track the system usage, I/O, memory usage, file system usage, network bandwidth, and so on, the collected data can help you tune the right component. It will also give you numbers for projecting your environment into the future. This will help you consider the whole system, instead of just the database.

- **Tune earlier, not later**

  Tune your application and database server while they are still in the development and testing stages. This will help you reduce the number of

issues you can run into during production. Also, the first few changes might take the least amount of time to figure out and provide the biggest return. As you keep tuning, the amount of time taken increases while the amount of gain starts to decrease.

- **Make the right goals**

  Setting an arbitrary goal such as all transactions under three seconds sounds nice in theory, but might be a costly one to achieve. Set the expectation for users and understand what you need to do to achieve them. The unreachable goals become visible especially at month-end or quarter-end report times.

For known issues with DB2 UDB on AIX, go to the following Web site:

http://www.ibm.com/support/docview.wss?rs=71&uid=swg21165448

For the operating systems and hardware matrix on which DB2 UDB is supported, use this Web site:

http://www.ibm.com/software/data/db2/udb/sysreqs.html

8.2 Initial DB2 UDB configuration parameter settings

This section lists the DB2 UDB parameters that should be set for an out-of-the-box install for DB2 UDB instance and database in a Siebel environment. These parameters can be found in the following guide in Siebel Bookshelf:

Siebel Installation Guide for UNIX: Servers, Mobile Web Clients, Tools

8.2.1 Database manager configuration

Table 8-1 shows a list of recommended database manager configuration parameters that have to be set on DB2 UDB for a Siebel install.

Table 8-1 Siebel recommended database manager configuration settings

Parameter	Explanation	Setting
SHEAPTHRES	Sort heap threshold (4 KB)	200000 Deployments with 3,000 or more concurrent users and using over 5 GB of RAM can increase this to 300000.
DIR_CACHE	Directory cache support	YES

Parameter	Explanation	Setting
ASLHEAPSZ	Application support layer heap size	15
RQRIOBLK	Maximum requester I/O block size (bytes)	65535
MON_HEAP_SZ	Database monitor heap size (4 KB)	128 (minimum)
QUERY_HEAP_SZ	Query heap size (4 KB)	16384
KEEPFENCED	Keep Fenced process	YES
MAXAGENTS	Maximum number of existing agents	20000 (minimum)
NUM_INITAGENTS	Initial number of agents in pool	10
NUM_POOLAGENTS	Number of agents in the agent pool kept active at all times	80
MAX_COORDAGENTS	Maximum number coordinating agents	MAXAGENTS
INDEXREC	Index re-creation time	RESTART
INTRA_PARALLEL	Enable intra-partition parallelism	NO

8.2.2 Database configuration

Table 8-2 shows a list of recommended database configuration parameters that have to be set on DB2 UDB database for a Siebel install.

Table 8-2 Siebel recommended database configuration settings

Parameter	Explanation	Setting
DFT_DEGREE	Degree of parallelism (1= turn query parallelism off).	1

Parameter	Explanation	Setting
DFT_QUERYOPT	Default query optimization class. This parameter only takes effect on the database server and affects the Siebel Server components, such as Siebel EIM or Siebel Remote. Queries run through the user interface (UI) are not affected by this setting. They take the value from the Siebel system preference or you can override them at the business component level using Siebel Tools.	3
DBHEAP	Database heap (4 KB).	7429 (32-bit) 10000 (64-bit)
CATALOGCACHE_SZ	Catalog cache size (4 KB).	5558 (32-bit) 8000 (64-bit)
LOGBUFSZ	Log buffer size (4 KB).	On AIX, 128 (32-bit) 512 (64-bit) On Windows, 256
UTIL_HEAP_SZ	Utilities heap size (4 KB).	5000 (32-bit) 10000 (64-bit)
LOCKLIST	Maximum storage for lock list (4 KB).	25000 (The setting should never be smaller than this, but can be increased.)
APP_CTL_HEAP_SZ	Maximum application control heap size (4 KB). Controls the number of users that can be included within one connection to the database.	900
STMTHEAP	Minimum setting. If needed, this parameter should be incremented in 1048 blocks.	40960
STAT_HEAP_SZ	Statistics heap size (4 KB).	14000 (32-bit) 20000 (64-bit)
MAXLOCKS	Percentage of lock lists per application.	20 (32-bit) 30 (64-bit)
LOCKTIMEOUT	Lock time out.	300

Parameter	Explanation	Setting
CHNGPGS_THRESH	Changed pages threshold.	30
NUM_IOCLEANERS	Number of asynchronous page cleaners.	Number of CPUs.
INDEXSORT	Index sort flag.	YES
SEQDETECT	Sequential detect flag.	YES
DFT_PREFETCH_SZ	Default prefetch size (4 KB).	32
SORTHEAP	Sort list heap (4 KB). Lower values should be used for development environments; higher values for production. However, increasing this value can lead to insufficient memory on the database server. Also, this parameter may need to be set below the recommended range if you have a high number of Siebel users. Therefore, you need to always monitor database server memory and performance to find the best setting for your environment.	1000-5000
LOGRETAIN	Sequential or circular log files.	RECOVERY. Set this parameter to RECOVERY in a production environment. Otherwise, you will lose data if your database crashes. When LOGRETAIN is set to RECOVERY, you must also activate USEREXIT or implement another method to manage the archived logs, so that LOGPATH does not fill up.
MAXAPPLS	Maximum number of active applications.	Twice the number of users.
AVG_APPLS	Average number of active applications.	Start with a default of 1.
MAXFILOP	Maximum DB files open per application.	500

Parameter	Explanation	Setting
LOGFILSIZ	Log file size (4 KB).	40000
LOGPRIMARY	Number of primary log files.	25 - 50 The value of LOGPRIMARY and LOGSECOND together may not exceed 256.
LOGSECOND	Number of secondary log files.	Up to 103 The value of LOGPRIMARY and LOGSECOND together may not exceed 256.
ESTORE_SEG_SZ	Server deployment with more than 4 GB of RAM can take advantage of this extended storage parameter. Use of this parameter also improves application sorting. Attach 4 KB and 16 KB buffer pools.	Initially 16000 but can be up to 65536. (only for 32-bit instances)
NUM_ESTORE_SEGS	See explanation of ESTORE_SEG_SZ.	Initially 16 (only for 32-bit instances) We recommend setting it to '0' and increase it during performance testing phase.
SOFTMAX	Percent log file reclaimed before soft checkpoint.	80
APPLHEAPSZ	Default application heap (4 KB).	2500
PCKCACHESZ	Package cache size (4 KB).	40000
NUM_IOSERVERS	Number of disks on which the database resides.	Number of disks.

8.2.3 DB2 registry variables

Table 8-3 on page 206 shows a list of DB2 environment and registry variables you have to set, and their recommended settings on DB2 UDB for a Siebel install.

Table 8-3 Siebel recommended DB2 UDB registry variable settings

Parameter	Explanation	Setting
DB2MEMDISCLAIM	AIX only. When set to YES, DB2 agents explicitly request that the database server disassociate the reserved paging space from freed memory. Affects how DB2 frees shared memory.	YES
DB2MEMMAXFREE	AIX only. Maximum amount of unused memory in bytes retained by DB2 processes. It affects how DB2 frees shared memory and causes DB2 to release memory as soon as the size of the DB2 agent goes above the listed value.	1000000
DB2_HASH_JOIN	Turns off hash joins in the optimizer.	ON
DB2_MMAP_WRITE	AIX only. You should evaluate this setting for your particular configuration and environment.	OFF
DB2_MMAP_READ	AIX only. You should evaluate this setting for your particular configuration and environment.	OFF
DB2_CORRELATED_ PREDICATES	When set to YES, the optimizer is able to determine whether or not predicates in a query are related, which permits DB2 to calculate the filter factor more accurately.	YES
DB2_PIPELINED_PLANS	Tells the DB2 optimizer to favor pipeline execution plans.	ON
DB2_INTERESTING_KEYS	Limits the number of execution plans generated by the DB2 optimizer.	ON
DB2_PARALLEL_IO	Useful when using RAID devices. For more information, see your DB2 vendor documentation.	* (asterisk)
DB2_STRIPED_ CONTAINERS	Useful when using RAID devices. For more information, see your DB2 vendor documentation.	ON

Parameter	Explanation	Setting
EXTSHM	AIX only. Use this parameter only when you have the Siebel Database and the Siebel Server on the same AIX machine. EXTSHM must be set when the DB2 UDB database is created and must be included in the script that starts it. The parameter also must appear in the sqllib/db2profile file for the DB2 UDB server. Additionally, you should include this parameter in the script that starts the DB2 client. After changing any of these settings, you must perform a db2stop, then db2start to implement the changes in your DB2 UDB database.	ON
DB2ENVLIST	AIX only. When starting a DB2 UDB server and running EXTSHM, EXTSHM must be part of the DB2 environment. This parameter must be set when the database is created.	EXTSHM
DB2_NO_PKG_LOCK	To bind the Siebel package with siebbind, this package must be off.	OFF
DB2CODEPAGE	Specifies the code page of the data presented to DB2 for database client application.	DONOT Set this to a value that is not supported by your OS.
DB2DBDFT	Specifies the database alias name of the database to be used for implicit connects.	<*dbname*>

8.3 Siebel configuration

In this section, we look at a few of the Siebel configuration parameters that are relevant to DB2 UDB. These are published in Siebel Bookshelf.

8.3.1 Siebel configurations for database

The Siebel Enterprise has many components other than the database that can be configured for high performance and scalability. Siebel Bookshelf has a Performance Tuning Guide for this purpose which explains in great detail the different architecture and infrastructure areas that you can tune.

In this section, our attempt is to only highlight those aspects of Siebel configuration that directly or indirectly impact database performance. We have grouped the parameters at the appropriate Siebel architecture level and you can tune them to gain optimal performance.

- **Siebel Application Object Manager (OM)**

  This is a multithreaded process on the Siebel Server. Each thread communicates with the database among other components. By default, the database connections in OM are not pooled and they are closed when the user session terminates. Depending on your implementation, database connections may be pooled to support sharing and/or persistence. The advantage of having persistent connections is that the cost of creating and closing database connections often can be avoided, thereby gaining performance. Tuning the `MaxTasks`, `MaxSharedDbConns`, and `MinSharedDbConns` based on user scenarios can enhance performance. Keep in mind that when more than one user shares a connection, there is a potential for blocking each other. For example, if a user has a long running query, the other users sharing that connection may have to wait and this can cause severe performance issues. You should also consider whether you are using database authentication or an external authentication like LDAP and accordingly consider the cost of creating database connections. For a detailed explanation and guidelines, please refer to the Siebel Bookshelf.

  The OM also has two other parameters that you can use for tuning performance:

  – SQL cursor cache
    The `DSMaxCachedCursors` parameter can be adjusted to specify the number of SQL cursors per database connection. This parameter can be adjusted based on the CPU and memory availability on the server hosting the OM. Consider increasing the number if you are using connection pooling.

  – SQL data cache

    SQL Data cache is controlled through two parameters:

    - `DSMaxCachedDatasetsPerProcess` controls the global data cache.
    - `DSMaxCachedDatasets` controls the per-connection data cache.

  Consider tuning these parameters after monitoring the CPU and memory usage/availability on the Siebel server.

- **Siebel Workflow**

  Siebel Workflow helps to automate business processes and has two components, *Workflow Processes* and *Workflow Policies*, that are created during this process of automation. There are three tables on the database that are used by Workflow Policies: `S_ESCL_REQ`, `S_ESCL_STATE`, and `S_ESCL_ACTN_REQ`. The DBA must be aware of these key tables and should

monitor them on a regular basis. These tables need to be maintained by adjusting the table space, buffer pool, and table parameters. If any of these tables becomes very large, then you should review the number of workflow policies being monitored. Also, sometimes a policy may be deactivated without removing the database triggers that will continue to insert rows which may not be acted upon. Make sure that no outstanding records are left behind in these tables after a Workflow Policy is deleted or retired.

- **Siebel Tools/Business components**

  The DBA needs to be aware of the fact that the default query optimization class set on the database is only applicable to Siebel Server components like EIM and not to queries coming from the user interface. While tuning the system, if there is a need to adjust the query optimization class, you should adjust the Siebel system preference named *DB2: Default Opt Level*. In case the requirement is to change the optimization class for specific business components, then this should be done in Siebel Tools.

- **Customer configurations**

  While customizing the Siebel application through Siebel Tools, you must keep in mind that there is a delicate balance in the Siebel architecture that is maintained by using multiple database features such as indexes, cursors, efficient SQL generation, and so on. When the Siebel application is customized using Siebel tools, you must take care to create or modify objects without breaking this critical balance. Some of these are listed here for ready reference and as a means to do a quick sanity check. Please refer to the Performance Tuning Guide on Siebel Bookshelf for complete details.

  - Avoid using sort specifications on non-indexed columns or joined columns.
  - Avoid or limit the use of case insensitivity.
  - Limit the number of business components in a view to avoid unnecessary population of data at run time.
  - Limit the number of fields in business components or applets to avoid generating a query that may be too large to be processed.
  - Limit the number of required fields, because these must always be retrieved from the database.
  - Limit the number of records returned for a business component by adding a search specification.
  - Limit the number of joins, extension tables, and primary ID fields in business components.
  - Prefer using inner join rather than outer joins.
  - Remove sort buttons that are unnecessary for your business requirement.

- Avoid calculated fields that do COUNT or SUM.
- Control Predefined Queries (PDQs): Either do not allow users to create their own PDQs or if you do, add indexes to optimize performance of slow PDQs.
- Set granular positions and responsibilities for Siebel users. This can substantially limit the amount of data retrieved from the database.

8.4 Operating system configuration considerations

Before installing Siebel software and DB2 UDB, a carefully planned architecture of servers and the network must be in place, based on the planned number of users who will use the application. The number of servers necessary, their capacity in terms of number and speed of CPUs, and the amount of memory are important considerations. You can engage Siebel Expert Services to help with architecture reviews. In this section, we discuss some server settings for Windows and AIX which can impact performance. We recommend you pay special attention to these parameters when entering the initial settings and adjust later if necessary.

Windows servers

For Windows servers, you must consider the following network settings and TCP/IP parameters. You can edit these through `regedit`. Modifying the registry can cause serious problems that may require you to reinstall the operating system.

- `MaxFreeTcbs`
  This registry variable determines the number of control blocks that are created to support connections. The default value in general may not be sufficient to support the number of users planned in the application.
- `MaxHashTableSize`
  This parameter goes hand in hand with the `MaxFreeTcbs` and needs to be adjusted as well.
- `TcpTimedWaitDelay`

  This setting determines the length of time that a connection will stay in the TIME_WAIT state before being closed. The default is 240 seconds which on a busy server will limit the maximum connections to around 200/sec. Reducing this setting will increase the maximum connection limit.
- `TcpWindowSize`

  This parameter determines the maximum TCP receive window size offered by the system. The receive window specifies the number of bytes a sender may transmit without receiving an acknowledgment.

- `MaxuserPort`

  Determines the highest port number TCP can assign when an application requests an available user port from the system. Adjusting the `MaxUserPort` registry entry (which does not exist by default) to a larger value reduces the likelihood of collisions by allowing Windows a greater range of port values to choose from when allocating ports.

AIX servers

Tuning all aspects of a database server is key to a highly available and high performing database server. OS tuning should not be an afterthought, normally it is the easiest one to check and fix.

- Most operating systems cache the file they read. Change the mount options to release buffered cache. This can enhance performance. On AIX, the mount option *rbrw* is recommended.
- Check the advantages between JFS and JFS2 before you make the final decision. There are advantages of using JFS2 over JFS, especially with AIX 5.3. Depending on the AIX and DB2 UDB levels, there may be potential issues. Refer to this IBM Web site for details:

  http://www.ibm.com/support/docview.wss?rs=71&context=SSEPGG&uid=swg21165572&loc=en_US&cs=utf-8&lang=en+en
- Learn and practice tools like `vmtune64`, `svmon`, `nmon`, `iostat`, and `vmstat` as shown in Chapter 7, “Database server monitoring” on page 167. They will help you later in case of issues to determine problems and also to analyze performance.

Network and TCP/IP parameters

Consider the following TCP/IP parameters:

- `tcp_sendspace`
  This is a kernel variable that specifies the default socket send buffer size. The default size may be low and may need to be adjusted to a higher value.
- `tcp_recvspace`
  Again, a kernel variable for default socket receive buffer size, this may need adjustment to higher value. The values you choose for these two parameters will depend on the machine type and adapter.
- `sb_max`
  Based on the values of send and receive default buffer sizes, this variable may also need adjustment.

Virtual Memory Manager

The Virtual Memory Manager (VMM) manages the memory on AIX. You can use the `vmtune` command to adjust parameters affecting the system’s memory usage.

However, these remain in effect only as long as the server is up and any changes are lost at next reboot. To make these changes permanent, you should make an entry in the `/etc/inittab`. The parameters to consider are:

- -b: specifies the number of file system buffer structures. Increasing the value helps write performance for large write sizes.
- -c: specifies the number of clusters processed by write behind. You may consider setting this to 0 (zero) to disable write behind.
- -f and -F: these two control page stealing and depending on the amount of RAM available on your system, these may be adjusted to higher values.
- -R: this parameter controls the maximum number of pages to be read ahead and increasing this to a higher value can benefit large sequential reads.
- -u: this parameter can be very beneficial for large I/Os, if we are using raw devices. Consider increasing the default for the database server if using raw devices.
- -p and -P: these affect the VMM page replacement.

8.5 Tuning the DB2 UDB environment

There are a number of variables that affect the performance of DB2. The default values are generally good and normally would not need any changes. But to get optimal performance for some Siebel environments, changes may be required. In this section, we discuss changes that you can make to an up and running environment to help with performance. Make sure you have tested these before applying them in production.

The IBM DB2 UDB manual *Administration Guide: Performance V8*, SC09-4821-01, provides rich performance tuning information. The following is a link to DB2 UDB V8.2 manualsL

http://www.ibm.com/software/data/db2/udb/support/manualsv8.html

8.5.1 DB2 UDB registry and environment variables

In this section, we discuss some of the DB2 UDB environment variables that impact performance. Also, you can find the system variables at the DB2 Information Center:

http://publib.boulder.ibm.com/infocenter/db2help/index.jsp?topic=/com.ibm.db2.udb.doc/admin/r0005658.htm

AIX and Windows registry variables

The following registry variables apply to both AIX and Windows systems:

- **DB2_FORCE_APP_ON_MAX_LOG**:

  This registry variable specifies the action DB2 will take when the MAX_LOG configuration parameter value is exceeded. If set to TRUE, the application is forced off the database and the unit of work is rolled back.

- **DB2_PARALLEL_IO**:

  This registry variable is used to change the way DB2 UDB calculates the I/O parallelism of a table space. When I/O parallelism is enabled (either implicitly by the use of multiple containers, or explicitly by setting DB2_PARALLEL_IO), it is achieved by issuing the correct number of prefetch requests. Each prefetch request is a request for an extent of pages.

  Values for this parameter are:

  - *[:number-of-disks]:
    Meaning every table space will have parallel I/O enabled. A value or symbol can be provided after a colon (:) to define the default number of disks per container for all table spaces. A default of 6 (the value for a RAID-5 device) is used if the number of disks is not specified. The asterisk (*) is used to tell DB2 UDB that all table spaces will use the ratio between the prefetch size and the extent size values as the degree of parallelism when issuing prefetch requests. The value after the asterisk and colon (that is, *:n, where "n" is the number), is used by DB2 UDB to calculate the prefetch size of a table space if the table space uses the AUTOMATIC option.
  - TablespaceID[:number-of-disks]
    A comma-separated list of defined table spaces. To define the number of disks per container for that table space, specify a value following a colon after each table space ID. This value can be a numeric value or one of the symbols found in the description that follows.
  - A combination of the preceding two value types with each value separated by a comma.

- **DB2_STRIPED_CONTAINERS**:

  This parameter is used when the table spaces reside on RAID arrays. This parameter works in junction with DB2_PARALLEL_IO. In previous versions, this registry variable was used to create table spaces with an extent-sized tag. However, because this is now the default behavior, this registry variable no longer has any effect.

- **DB2_OVERRIDE_BPF:**

  There are times when you may have gone overboard setting the amount of buffer pool pages that you can allocate to the database. When the allocation exceeds the amount of physical memory on the box, the system will start heavily paging out. If the allocation exceeds physical memory, plus paging space, it can make the environment unstable. This parameter is really

valuable when you plan to restore a production database to a test environment where the amount of memory available is not the same.

When set to a positive number of 4K pages, this parameter specifies the size of the buffer pool (in pages) that will be created at database activation or the first time a connection is established. The default value is null.

- **DB2MEMMAXFREE:**

  Specifies the maximum number of bytes of unused private memory that is retained by DB2 processes before unused memory is returned to the operating system.

  The values range from 0 to 2.0e+32 bytes. The default is 8,388,608 bytes. Siebel recommends you set this value to 1000000 at the start.

- **DB2_ENABLE_BUFPD:**

  Specifies whether or not DB2 uses intermediate buffering to improve query performance. The buffering may not improve query performance in all environments. Testing should be done to determine individual query performance improvements. The default value is `ON`.

- **DB2_HASH_JOIN:**

  The `DB2_HASH_JOIN` registry variable is now set to `ON` by default in DB2 UDB Version 8.

  For Siebel CRM database, Siebel recommends you set this value to `ON`.

- **DB2_PIPELINED_PLANS:**

  For Siebel CRM database, Siebel recommends you set this value to `ON`.

- **DB2_INTERESTING_KEYS:**

  For Siebel CRM database, Siebel recommends you set this value to `ON`.

- **DB2_EVALUNCOMMITTED:**

  When enabled, will allow, where possible, table or index access scans to defer or avoid row locking until a data record is known to satisfy predicate evaluation. With this variable enabled, predicate evaluation may occur on uncommitted data. It is applicable only to statements using either Cursor Stability or Read Stability isolation levels. For index scans, the index must be a type-2 index.

  Setting this to `ON` can help reduce lock contention.

- **DB2_SKIPDELETED:**

  When enabled, allows statements using either Cursor Stability or Read Stability isolation levels to unconditionally skip deleted keys during index access and deleted rows during table access. With `DB2_EVALUNCOMMITTED` enabled, deleted rows are automatically skipped, but uncommitted

pseudo-deleted keys in type-2 indexes are not, unless DB2_SKIPDELETED is also enabled.

Setting this to ON can help reduce lock contention.

For UNIX platforms

In this section, we discuss DB2 UDB environment variables specific to UNIX platforms.

- **DB2MEMDISCLAIM**:

  The DB2MEMDISCLAIM registry variable controls whether DB2 agents explicitly request that AIX disassociate the reserved paging space from the freed memory. A DB2MEMDISCLAIM setting of YES results in smaller paging space requirements, and possibly less disk activity from paging. A DB2MEMDISCLAIM setting of NO will result in larger paging space requirements, and possibly more disk activity from paging. In some situations, such as if paging space is plentiful and real memory is so plentiful that paging never occurs, a setting of NO provides a minor performance improvement.

  Siebel recommends setting this to YES.

- **DB2_PINNED_BP**:

  This variable is used to specify the database global memory (including buffer pools) associated with the database in the main memory on AIX operating systems. Keeping this database global memory in the system main memory allows database performance to be more consistent.

  For example, if the buffer pool is swapped out of the system main memory, database performance deteriorates. The reduction of disk I/O by having the buffer pools in system memory improves database performance. If other applications require more of the main memory, allow the database global memory to be swapped out of main memory, depending on the system main memory requirements. Default is NO.

  Setting the value to YES, will pin the shared memory in RAM, but will limit amount of dynamic growth allowed for buffer pools.

- **DB2_MMAP_READ and DB2_MMAP_WRITE**:

  These parameters are both set to allow DB2 UDB to use MMAP as an alternate method of I/O. In most environments, MMAP should be used to avoid operating system locks when multiple processes are writing to different sections of the same file.

  When these variables are set to ON, data that is read to and written from the DB2 buffer pools bypasses the AIX memory cache. If you have a relatively small DB2 buffer pool, and you cannot or choose not to increase the size of this buffer pool, you should take advantage of AIX memory caching by setting DB2_MMAP_READ and DB2_MMAP_WRITE to OFF.

For Windows

In this section, we discuss DB2 UDB environment variables specific to Windows platforms.

- **DB2NTMEMSIZE**:

  Windows NT® requires that all shared memory segments are reserved at DLL initialization time in order to guarantee matching addresses across processes. DB2NTMEMSIZE permits the user to override the DB2 defaults on Windows NT if necessary.

  The memory segments, default sizes, and override options are:

  – Database Kernel: default size is 16777216 (16 MB); override option is DBMS:<number of bytes>.

  – Parallel FCM Buffers: default size is 22020096 (21 MB); override option is FCM:<number of bytes>.

  – Database Admin GUI: default size is 33554432 (32 MB); override option is DBAT:<number of bytes>.

  – Fenced Stored Procedures: default size is 16777216 (16 MB); override option is APLD:<number of bytes>.

  More than one segment may be overridden by separating the override options with a semi-colon (;). For example, to limit the database kernel to approximately 256K, and the FCM buffers to approximately 64 MB, use:

  ```
  db2set DB2NTMEMSIZE = DBMS:256000;FCM:64000000
  ```

  In most situations, the default values should be sufficient.

- **DB2NTNOCACHE**:

  All the functions for this registry variable are now rolled into the CREATE TABLESPACE and ALTER TABLESPACE commands.

- **DB2NTPRICLASS**:

  Sets the priority class for the DB2 instance (program *DB2SYSCS.EXE*). There are three priority classes:

  – NORMAL_PRIORITY_CLASS (the default priority class)
  – REALTIME_PRIORITY_CLASS (set by using “R”)
  – HIGH_PRIORITY_CLASS (set by using “H”)

  Care should be taken when using this variable.

  The Default Value is null. It can be set to R, H, and any other value.

  Also refer to DB2PRIORITIES and SetPriorityClass().

8.5.2 DB2 UDB database manager parameters

In this section, we look at several database manager (DBM) configuration parameters that affect the performance of a database.

Some DBM configuration parameters are now immediate. Refer to the IBM DB2 UDB document: *Administration Guide: Performance V8*, SC09-4821-01, for details.

- **SHEAPTHRES**:

  This parameter sets the limit on how much memory is allocated to shared and private sorts. Private and shared sorts use memory from two different memory sources. The size of the shared sort memory area is statistically predetermined at the time of the first connection to a database based on the value of `SHEAPTHRES`. The size of the private sort memory area is unrestricted. Adjust this value based on amount of memory on the system.

  > **Note:** On normal OLTP environments, start with a value of about 100000. The `SHEAPTHRES` value should be higher than "total private sort heap allocated". Another way to calculate this value is `(number of concurrent agents running against the database) * (sortheap value set at the database level)`.

- **ASLHEAPSZ:**

  The application support layer heap represents a communication buffer between the local application and its associated agent. This buffer is allocated as shared memory by each database manager agent that is started. Depending on the amount of memory you have, you may want to increase this value.

  > **Note:** If your application is running on a memory constrained system, you might choose to reduce the value of this parameter. If your queries are generally very large, requiring more than one send and receive request and your system is not constrained by memory, you may choose to increase the value of this parameter.

- **NUM_POOLAGENTS:**

  This parameter determines the maximum size of the idle agent pool. Idle agents can be used as parallel subagents or as coordinator agents. If more agents are created than is indicated by the value of this parameter, they will be terminated when they finish executing their current request.

> **Note:** In Siebel CRM, we suggest you monitor your idle agents and set this to a value higher than suggested in the out-of-the-box install. This will lower the cost of creation and termination of agents.

- **NUM_INITAGENTS:**

  This parameter determines the initial number of idle agents that are created in the agent pool at DB2START time. Increase this value as the number of APPSERVERS that you deploy increases.

- **MAXAGENTS**:

  This parameter indicates the maximum number of database manager agents, whether coordinator agents or subagents, available at any given time to accept application requests. This value should be set to a minimum of MAXAPPLS, but this value can be tuned depending on the amount of memory on your environment. On small environments, decrease this value. On production environments, monitor and adjust the value until agents stolen from another application are equal to 0.

- **MAX_COORDAGENTS:**

  For Siebel, we recommend you set this value equal to MAXAGENTS.

- **DFT_MON_*:**

  There are six monitors that can be turned on to capture different information about how the database and database manager are performing. We suggest turning them on (if you are not memory constrained) to capture as much information as possible, so you can use it later.

> **Note:** Run the GET SNAPSHOT command for the entire database at several intervals during a day and tabulate and use snapshot information to tune your database and database manager configurations.

- **MON_HEAP_SZ:**

  This parameter determines the amount of the memory, in pages, to allocate for database system monitor data. Memory is allocated from the monitor heap when you perform database monitoring activities such as taking a snapshot, turning on a monitor switch, resetting a monitor, or activating an event monitor. If you are not memory-constrained, increase it to a value where you can take a snapshot for the whole day without running out of memory in this heap.

  We recommend leaving INTRA_PARALLEL to NO and MAX_QUERYDEGREE to 1 for the Siebel database. Changing these can have negative effects on performance results.

8.5.3 DB2 UDB database parameters

In this section, we look at several database parameters that affect the performance of a database.

- **SORTHEAP:**

  This parameter defines the maximum number of private memory pages used for private sorts, or the maximum number of shared memory pages used for shared sorts. If the sort is a private sort, then this parameter affects agent private memory. If the sort is a shared sort, then this parameter affects the database shared memory. Each sort has a separate sort heap that is allocated as necessary by the database manager. This sort heap is the area where data is sorted. If directed by the optimizer, a smaller sort heap than the one specified by this parameter is allocated using information provided by the optimizer.

  Watch your `db2diag.log` for messages. You may want to decrease this value (lower than the Siebel out-of-the-box recommendation), when you see that you have a lot of small sorts. Also, monitor “Sort overflows” in snapshot. This will help you in determine the optimum value.

  An optimal value for medium to large Siebel databases is between 64 and 256.

  > **Note:** When you increase `SORTHEAP`, look at the `SHEAPTHRES` value also. Having proper indexes to match your queries will reduce the usage of sort heap (monitor “Total sorts” in snapshot).

- **PKGCACHESZ:**

  This parameter is allocated out of the database shared memory, and is used for caching of sections for static and dynamic SQL statements on a database. Caching packages allows the database manager to reduce its internal overhead in the case of dynamic SQL by eliminating the need for compilation.

  This caching of the section for a dynamic SQL statement can improve performance especially when the same statement is used multiple times by applications connected to a database. This is particularly important in a transaction processing application such as Siebel CRM.

  PKGCACHESZ should be larger than “Package cache high water mark (Bytes)” in database snapshot. By increasing PKGCACHESZ, you can drive “Package cache overflows” to 0.

  The Package Cache Hit Ratio (PCHR) should be as close to 100% as possible. Calculate with the following formula:

  ```
  PCHR = (1-("Package cache inserts" / "Package cache lookups"))*100
  ```

Pay attention to Global shared memory, which is calculated by:

```
buffer pools + dbheap + util_heap_sz + pkgcachesz + aslheapsz + locklist +
approx 10% overhead
```

- **CATALOGCACHE_SZ:**

  This parameter is allocated out of the database shared memory, and is used to cache system catalog information. Caching catalog information at individual partitions allows the database manager to reduce its internal overhead by eliminating the need to access the system catalogs to obtain information that has previously been retrieved.

  The value suggested in Siebel out-of-the-box should be applied, but if you add a lot of custom tables you may want to change the value.

- **CHNGPGS_THRESH:**

  You can use this parameter to specify the level (percentage) of changed pages at which the asynchronous page cleaners will be started. Asynchronous page cleaners will write changed pages from the buffer pools to disk before the space in the buffer pool is required by a database agent. As a result, database agents should not have to wait for changed pages to be written out so that they can use the space in the buffer pool. This improves overall performance of the database applications.

  For normal operations, the Siebel out-of-the-box value of 30 is pretty good. But in specialized cases where you are performing EIM and so on, setting this value to a higher number helps, since the records that are being inserted are sequential and the same agents are doing the work.

- **AVG_APPLS:**

  When running DB2 in a multi-user environment, particularly with complex queries and a large buffer pool, you may want the SQL Optimizer to know that multiple query users are using your system so that the optimizer should be more conservative in assumptions of buffer pool availability.

  Start with a value of 1 and increase it depending on the number of complex SQLs that will be running at any given time. Be conservative with this one.

  On Unix, run the following command in order to understand the number of applications running at any given time. Take an average and set the value.

  ```
  db2 “list applications show detail”|grep -c Executing
  ```

- **LOCKLIST:**

  This parameter indicates the amount of storage that is allocated to the lock list. There is one lock list per database and it contains the locks held by all applications concurrently connected to the database. When the percentage of the lock list used by one application reaches `MAXLOCKS`, the database manager

will perform lock escalation, from row to table, for the locks held by the application.

Tune the value for this parameter using the database monitor. This parameter works along with `MAXLOCKS`.

- **`MAXLOCKS`:**

  Lock escalation is the process of replacing row locks with table locks, reducing the number of locks in the list. This parameter defines a percentage of the lock list held by an application that must be filled before the database manager performs escalation.

  When the number of locks held by any one application reaches this percentage of the total lock list size, lock escalation will be attempted for the locks held by that application. In busy systems, the connection attempting the escalation will usually fail and roll back on a timeout. Lock escalation also occurs if the lock list runs out of space.

  The following formula allows you to set `MAXLOCKS` to allow an application to hold twice the average number of locks: `MAXLOCKS = 2 * 100 / MAXAPPLS`.

- **`LOCKTIMEOUT`:**

  This parameter specifies the number of seconds that an application will wait to obtain a lock. This helps avoid global deadlocks for applications. For OLTP environments, start with a setting of 30 seconds and change it depending on your requirements.

  Lock timeout happens for several reasons, but the most common are:

  – An application holding a lock for an extended period of time. For Siebel, make sure the `TXN_ISOLATION` is set to 1 in `db2cli.ini` on the Application Server.

  – Insert and update queries are not tuned.

- **`NUM_IOCLEANERS`:**

  This parameter allows you to specify the number of asynchronous page cleaners for a database. These page cleaners write changed pages from the buffer pool to disk before the space in the buffer pool is required by a database agent. As a result, database agents should not have to wait for changed pages to be written out so that they can use the space in the buffer pool. This improves overall performance of the database applications.

  Set this to the number of CPUs. All asynchronous page cleaners become active at one time. If you set this to a number higher than number of CPUs, you can block other processing and hurt performance.

- **NUM_IOSERVERS:**

  I/O servers are used on behalf of the database agents to perform prefetch I/O and asynchronous I/O by utilities such as backup and restore. This parameter specifies the number of I/O servers for a database. The number of I/O for prefetching and utilities in progress for a database cannot exceed this number at any time. An I/O server waits while an I/O operation that it initiated is in progress. Non-prefetch I/O are scheduled directly from the database agents and as a result are not constrained by `NUM_IOSERVERS`.

  We recommend setting this value to the number of disks (even if they are in a RAID array) across which the database is spread, plus one. But do not go over four times the number of CPUs.

- **MAXFILOP:**

  Leave this at the Siebel out-of-the-box specified value.

- **LOGBUFSZ:**

  Leave this at the Siebel out-of-the-box specified value.

- **LOGFILSIZ:**

  This parameter defines the size of each primary and secondary log file. The size of these log files limits the number of log records that can be written to them before they become full and a new log file is required.

  This has a direct effect on your High Availability and disaster recovery solution, if you use logshipping. Having this value pretty high may improve performance, but at the same time increases the chance to lose a higher number of transactions. Set this to a low value that will help in disaster recovery at the same time, and make sure you have enough log to do large transaction processing (such as full table updates, and so on).

- **MIRRORLOGPATH:**

  This parameter allows you to specify a string of up to 242 bytes for the mirror log path. The string must point to a path name, and it must be a fully qualified path name, not a relative path name. Have log mirrored can help disaster recovery and keep your system available.

- **LOGPRIMARY:**

  The primary log files establish a fixed amount of storage allocated to the recovery log files. This parameter allows you to specify the number of primary log files to be preallocated. You may want to increase or decrease the number depending on the value of `LOGFILSIZ`. Making this a high number would mean additional time needed at db activation. Increase this number when you see that secondary logs are being allocated often. The number of log primary plus number of secondary should be less or equal to 256.

- **LOGSECOND:**

  This parameter specifies the number of secondary log files that are created and used for recovery log files (only as needed). When the primary log files become full, the secondary log files (of size LOGFILSIZ) are allocated one at a time as needed, up to a maximum number as controlled by this parameter. An error code will be returned to the application and the database will be shut down if more secondary log files are required than are allowed by this parameter.

- **DFT_DEGREE:**

  For Siebel, we recommend setting this value equal to 1.

- **DFT_QUERYOPT:**

  For Siebel, we recommend setting this value equal to 3.

  If changes are needed, change optimization at Business component level.

- **APP_CTL_HEAP_SZ:**

  Adjust this parameter's value beyond the recommended value based on the number of users.

8.6 Lock wait/deadlock/lock escalation

Locks are used to provide concurrency control and prevent uncontrolled data access. The database manager places locks on buffer pools, tables, table blocks, or table rows. A lock associates a database manager resource with an application, called the *lock owner*, to control how other applications access the same resource.

8.6.1 Locking issues

Locking issues may arise under various conditions and need to be investigated. The main locking issues that you may encounter are: lock wait, lock timeout, lock escalations, and deadlock.

Lock waits and timeout

Lock waits are a common and natural occurrence in any multi-user application environment. A well-designed application will minimize lock waits by using appropriate locking levels when accessing shared data. However, from time to time, excessive numbers of lock waits or lock waits for extended periods may give rise to user complaints about poor performance.

Lock timeout detection is a database manager feature that prevents applications from waiting indefinitely for a lock to be released. For example, a transaction

might be waiting for a lock held by another user's application, but the other user has left the workstation without allowing the application to commit the transaction that would release the lock. To avoid stalling an application in such a case, set the `LOCKTIMEOUT` database configuration parameter to the maximum time that any application should wait to obtain a lock.

Setting this parameter helps avoid global deadlocks, especially in distributed unit of work (DUOW) applications. If the time that the lock request is pending is greater than the `LOCKTIMEOUT` value, the requesting application receives an error and its transaction is rolled back.

Consider the following scenario where Application A tries to acquire a lock which is already held by Application B. Once the waiting time is reached, DB2 UDB will terminate and roll back the transaction in Application A and returns an SQL error code to Application A as seen in Example 8-1 with reason code 68.

Example 8-1 SQL reason code for lock timeout

```
SQL0911N  The current transaction has been rolled back because of a deadlock
or timeout.  Reason code "68".  SQLSTATE=40001
```

Lock escalation

Lock escalation is a DB2 UDB internal mechanism that reduces the number of locks held. In a single table, locks are escalated to a table lock from many row locks, or for multi-dimensional clustering (MDC) tables, from many row or block locks. Lock escalation occurs when applications hold too many locks of any type. Lock escalation can occur for a specific database agent if the agent exceeds its allocation of the lock list. Such escalation is handled internally, the only externally detectable result might be a reduction in concurrent access on one or more tables.

Lock escalation can occur in two scenarios:

- A single application requests a lock that will exceed its allowable number of locks.
- An application triggers lock escalation because the maximum number of database locks on the system has been exceeded.

In both cases, the database manager will attempt to free up memory allocated to locking by obtaining table locks and releasing existing row locks. The desired effect is to make more lock memory available for additional applications.

The following two database parameters have a direct effect on lock escalation:

- `LOCKLIST`:
  This is the amount of storage (in units of 4K pages) that is allocated to hold

the list of all locks held by all applications concurrently connected to the database.

- `MAXLOCKS:`
  The allowable percentage of `LOCKLIST` that can be used by a single application.

Tuning and monitoring may be necessary to find a balance for these values. Workload and query behavior dictate locking patterns and how applications will use lock memory.

Deadlock

A deadlock is created when one application is waiting for another application to release a lock on data. Each of the waiting applications is locking data needed by another application. Mutual waiting for the other application to release a lock on held data leads to a deadlock. The applications can wait forever for each other's lock until an external event causes one or both of them to roll back. A deadlock may occur in the following manner:

- Application A locks record 1.
- Application B locks record 2.
- Application A attempts to access record 2, but waits since Application B already holds a lock on this record.
- Application B then tries to access record 1, but waits since Application A already holds a lock on this record.

In this situation, both Application A and Application B will wait indefinitely for each other to release their lock until an external event causes one or both of them to terminate the transaction and release the lock. The terminated transaction will be rolled back.

The application which receives a rollback will get the following SQL error code as shown in Example 8-2.

Example 8-2 SQL reason code for deadlock

```
SQL0911N  The current transaction has been rolled back because of a deadlock
or timeout.  Reason code "2".  SQLSTATE=40001
```

The frequency at which the DB2 deadlock detector checks for deadlocks can be controlled using `DLCHKTIME` (measured in milliseconds, from 1000 to 600000). Setting this value high will cause the deadlock check time to be increased, but the overhead of deadlock detection will be reduced. This could potentially result in applications stuck in deadlock for prolonged periods of time. Setting the deadlock check time to a smaller value allows for deadlocks to be detected

sooner; however, it also introduces additional overhead on the database for checking.

8.6.2 Tools used to monitor locking issues

The following are the main tools which will help you gather information about locks. These tools and their usage are described in Chapter 7, “Database server monitoring” on page 167.

- `LIST APPLICATION` command
- Snapshot monitoring
- Event monitoring
- Administration notification log
- Health Center
- db2pd
- OS tools

The following section focuses on a few of the above mentioned tools.

8.6.3 Identifying the lock issues

The lock issues are often presented by a slow response query. Before troubleshooting the lock issues, it always good to use OS tools to eliminate possible causes due to elements such as network, CPU, Disk I/O, and paging. If the problem is not due to any of these element, then the investigation can proceed with lock issue identification and troubleshooting.

Identifying lock waits and timeout

A good starting point for identifying the lock issues is the `LIST APPLICATION` command as shown in Example 8-3.

Example 8-3 `LIST APPLICATION` command

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 list applications show detail

CONNECT Appl Handle Application Id                 Status          DB Name
--------------------------------------------------------------------------------
AIXDBA    8        *LOCAL.ucsdb2.050730200025     UOW Waiting     VOL78RBA
AIXDBA    9        *LOCAL.ucsdb2.050730200029     Lock-wait       VOL78RBA
```

`LIST APPLICATION` command shows the information of active applications in a database. The application handle is the agent ID which uniquely identifies an active application. The status field shows the current status of application status.

This example shows that there are two applications that are currently connected to the database VOL78RBA. The application handle 8 is in “Lock-wait” status.

The starting point in the lock issue investigation is snapshot. To gather the snapshot information, we turned on the session monitor switches as shown in Example 8-4.

Example 8-4 Switch ON session monitors

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 update monitor switches using
bufferpool on lock on sort on statement on table on uow on
```

To further investigate the application lock wait situation, use the GET SNAPSHOT with the option LOCK as shown in Example 8-5.

Example 8-5 Snapshot command for locks

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 get snapshot for locks on vol78rba
```

This snapshot command produces an output for locks which has three sections in it, they are:

- Database section
- Application section
- List of locks section

The database section as shown in Example 8-6 shows the current lock situation on the database. Based on the output, we can determine that:

- There are six locks held.
- Out of two applications which are currently connected, there is one application which is waiting on locks.The same is also evident in LIST APPLICATION command output.

Example 8-6 Snapshot for locks (database section)

```
Database name                              = VOL78RBA
Database path                              =
/db2/fs1/vol78rba/ucsdb2/NODE0000/SQL00001/
Input database alias                       = VOL78RBA
Locks held                                 = 6
Applications currently connected           = 2
Agents currently waiting on locks          = 1
Snapshot timestamp                         = 07/31/2005 05:06:54.446965
```

The application section is shown in Example 8-7 on page 228. Based on the output, we can determine that:

- The application handle 9 has application status of “Lock-wait”.

- The application handle 8 has application status of “UOW Waiting” meaning DB2 UDB is waiting for the user/application to respond.
- The application handle 8 is holding an exclusive lock and application with handle 9 is requesting a conflicting update lock on the same database object causing the lock wait situation.

Example 8-7 Snapshot for locks (application section)

```
Application handle                          = 9
Application ID                              = *LOCAL.ucsdb2.050730200029
Sequence number                             = 0003
Application name                            = db2bp
CONNECT Authorization ID                    = AIXDBA
Application status                          = Lock-wait
Status change time                          = 07/31/2005 05:06:51.245902
Application code page                       = 1208
Locks held                                  = 3
Total wait time (ms)                        = 3201

  ID of agent holding lock                  = 8
  Application ID holding lock               = *LOCAL.ucsdb2.050730200025
  Lock name                                 = 0x0007018F000CA4030000000052
  Lock attributes                           = 0x00000000
  Release flags                             = 0x40000000
  Lock object type                          = Row
  Lock mode                                 = Exclusive Lock (X)
  Lock mode requested                       = Update Lock (U)
  Name of tablespace holding lock           = TBS_16K
  Schema of table holding lock              = SIEBEL
  Name of table holding lock                = S_CONTACT
  Lock wait start timestamp                 = 07/31/2005 05:06:51.245902

Application handle                          = 8
Application ID                              = *LOCAL.ucsdb2.050730200025
Sequence number                             = 0005
Application name                            = db2bp
CONNECT Authorization ID                    = AIXDBA
Application status                          = UOW Waiting
Status change time                          = 07/31/2005 05:02:14.942174
Application code page                       = 1208
Locks held                                  = 3
Total wait time (ms)                        = 0
```

Example 8-8 on page 229 shows the list of locks section. This section of the output shows the lock information of each application.

With this information, the DBA can communicate to the application owner for options to resolve the lock situation.

Example 8-8 Snapshot for locks (list of locks section)

```
List Of Locks (For Application handle= 9)
Lock Name                    = 0x0007018F000000000000000054
 Lock Attributes              = 0x00000000
 Release Flags                = 0x40000000
 Lock Count                   = 1
 Hold Count                   = 0
 Lock Object Name             = 399
 Object Type                  = Table
 Tablespace Name              = TBS_16K
 Table Schema                 = SIEBEL
 Table Name                   = S_CONTACT
 Mode                         = IX

List Of Locks (For Application handle= 8)
 Lock Name                    = 0x0007018F000CA4030000000052
 Lock Attributes              = 0x00000020
 Release Flags                = 0x40000000
 Lock Count                   = 1
 Hold Count                   = 0
 Lock Object Name             = 828419
 Object Type                  = Row
 Tablespace Name              = TBS_16K
 Table Schema                 = SIEBEL
 Table Name                   = S_CONTACT
 Mode                         = X

Lock Name                    = 0x0007018F000000000000000054
 Lock Attributes              = 0x00000000
 Release Flags                = 0x40000000
 Lock Count                   = 1
 Hold Count                   = 0
 Lock Object Name             = 399
 Object Type                  = Table
 Tablespace Name              = TBS_16K
 Table Schema                 = SIEBEL
 Table Name                   = S_CONTACT
 Mode                         = IX
```

Note: Prior to DB2 UDB V8.1, the only way to capture snapshot monitor data was to execute the `GET SNAPSHOT` command or call its corresponding API from an application program. With DB2 UDB V 8.1, you can also collect snapshot monitor data by constructing a query that references snapshot monitor table functions.

Resolution of lock wait and timeout

Some of the following changes are not applicable to Siebel, since it is a packaged application. Minimizing customizations is recommended.

The lock wait can be minimized by performing the following:

- Issue `COMMIT` statements at the right frequency.
- Specify `FOR FETCH ONLY` clause in `SELECT` statement. (Siebel out-of-the-box feature)
- When performing many updates, issue `LOCK TABLE` if required.
- Use `ALTER TABLE` with the LOCK parameter statement appropriately.
- Perform `INSERT`, `UPDATE`, and `DELETE` at the end of a unit of work if possible.
- Choose the appropriate isolation level. (Siebel out-of-the-box feature)
- Release read locks using `WITH RELEASE` option of `CLOSE CURSOR` statement if acceptable.
- Avoid lock escalations impacting concurrency by tuning `LOCKLIST` and `MAXLOCKS` database configuration parameter.

Identifying lock escalations

The DB2 UDB administration notification file is a good starting point for debugging the lock escalation issue. The administration notification file is where DB2 UDB writes most of the administration messages whenever a significant event occurs. This is intended for the use of database and system administration.

The type of event and the level of detail of the information gathered are determined by the `NOTIFYLEVEL` configuration parameter. In our lab setup, we had a default value of 3. The notification file on UNIX platforms is a text file called *<instance>.nfy in* `DIAGPATH` (parameter from DBM configuration) and on Windows it is written to the *Event Log.*

Example 8-9 is a sample output of the administration notification log:

Example 8-9 Administration notification log file (ucsdb2.nfy)

```
ADM5500W  DB2 is performing lock escalation.  The total number of locks
```

```
currently held is "17", and the target number of locks to hold is "8".

2005-07-30-20.42.53.687230   Instance:ucsdb2   Node:000
PID:336054(db2agent (VOL78RBA) 0)   TID:1   Appid:*LOCAL.ucsdb2.050730204201
data management  sqldEscalateLocks Probe:3   Database:VOL78RBA

ADM5502W  The escalation of "12" locks on table "SIEBEL  .S_CONTACT" to lock
intent "X" was successful.
```

Based on the output, we can determine that:

- DB2 is performing a lock escalation.
- Number of locks currently held (17) and number of locks required (8) before lock escalation.
- The application that caused the escalation had an application ID of `LOCAL.ucsdb2.050730204201`.
- The row level locks on table “`SIEBEL.S_CONTACT`” are converted into table level lock “X”.

More detailed information can be obtained by a snapshot monitor. Example 8-10 shows the snapshot command with `database` option and a sample of the output.

Example 8-10 Snapshot on database to identify lock escalations

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 get snapshot for database on vol78rba

...
Locks held currently                        = 7
Lock waits                                  = 4
Time database waited on locks (ms)          = 36091
Lock list memory in use (Bytes)             = 13328
Deadlocks detected                          = 0
Lock escalations                            = 7
Exclusive lock escalations                  = 4
Agents currently waiting on locks           = 0
Lock Timeouts                               = 0
Number of indoubt transactions              = 0
...
```

Based on the output, we can determine that:

- There are 7 lock escalations reported.
- There are 4 exclusive lock escalations, meaning the exclusive row locks are converted into an exclusive table lock. This also means no other application can access any row in the entire table.

- Impact of this can be seen in line “Lock waits” which is 4 and the applications had wait for a long time as indicated in “Time database waited on locks”.
- The key point to observe here is “Lock list memory in use” and when this value is approaching the value of `LOCKLIST` database parameter then there will be a lot of lock escalations.

Resolving lock escalations

Lock escalation can be resolved by tuning the applications and/or tuning the database parameters:

- Database parameter tuning.

  When the “lock list memory in use” is approaching the size of the lock list, it gives you a clear indication of lock escalation. The utilization of lock list is measured as a percentage of memory consumed, where a high percentage represents an unhealthy condition. The `LOCKLIST` memory utilization is calculated using the formula:

  ```
  (LOCKLIST in use / (LOCKLIST * 4096)) * 100
  ```

  The values from the above Example 8-10 on page 231 show the memory utilization as:

  ```
  (13328/(4*4096)) = 81%
  ```

  So if the utilization is more than 50%, you will need to increase LOCKLIST size and if the percentage is low yet there are lock escalations then increasing `MAXLOCK` and `LOCKLIST` will help. In this scenario, the `LOCKLIST` parameter is set to a low value to simulate the lock issue. For the recommended value, refer to 8.2.2, “Database configuration” on page 202.
- Tuning the application.

  For tuning the application, refer to “Resolution of lock wait and timeout” on page 230.

> **Note:** Tuning database parameters may yield more concurrency at the cost of high CPU usage to acquire locks and release them. By tuning the application, you are compromising on concurrency by changing the locking strategy of the applications. Introducing frequent commits will benefit in both the cases.

Identifying deadlocks

To monitor deadlocks, you can use the snapshot monitor at the database level as well as at the application level. The event monitor is a very useful tool for deadlock analysis and it can be left running for a long time since the overhead is less compared to snapshot.

To start with, the snapshot monitor is used to see if there are any deadlocks.

Use the `GET SNAPSHOT` command with `database` option.

Example 8-11 provides a valuable insight into locks and deadlocks. As seen in the output sample, there is one deadlock detected and if you find this number abnormally high, you can continue your investigation with event monitor.

Example 8-11 Output of database snapshot

```
Locks held currently                        = 4
Lock waits                                  = 2
Time database waited on locks (ms)          = Not Collected
Lock list memory in use (Bytes)             = 2240
Deadlocks detected                          = 1
Lock escalations                            = 0
Exclusive lock escalations                  = 0
Agents currently waiting on locks           = 0
Lock Timeouts                               = 0
Number of indoubt transactions              = 0
```

After the preliminary investigation, the monitor switches are turned off as shown in Example 8-12.

Example 8-12 Turn off the monitor switches

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 “update monitor switches using
BUFFERPOOL off, LOCK off, SORT off, STATEMENT off, TABLE off, UOW off”
```

To proceed with the investigation, an event monitor is created to detect deadlocks. Use the following commands to create an event monitor as shown in Example 8-13.

Example 8-13 Create an event monitor

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 connect to vol78rba
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 "create event monitor seblmon for
deadlocks with details write to file '\tmp\output'"
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 "set event monitor seblmon state 1"
```

Note: The event monitor state can be checked using the following SQL statement:

```
db2 "SELECT evmonname, EVENT_MON_STATE(evmonname) FROM
syscat.eventmonitors"
```

The event trace, which is written to path `\tmp\output` is read using the `db2evmon` command as shown in Example 8-14 on page 234.

Example 8-14 db2evmon (event monitor productivity tool)

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2evmon -path \tmp\output
```

The `db2evmon` command produces the output shown in Example 8-15.

Note: Remember that the following is just an example and direct SQL against Siebel tables is not supported. This is an exercise only.

There are two sections in the output:

- The Event Log Header contains the basic information about the event monitor, instance, and database.
- Other sections in the output have information on the two participating applications in a deadlock and its lock details.

Example 8-15 Event monitor output

```
                              EVENT LOG HEADER
  Event Monitor name: SEBLMON
  Server Product ID: SQL08021
  Version of event monitor data: 7
  Byte order: BIG ENDIAN
  Number of nodes in db2 instance: 1
  Codepage of database: 1208
  Territory code of database: 1
  Server instance name: ucsdb2
-------------------------------------------------------------------------
Database Name: VOL78RBA
  Database Path: /db2/fs1/vol78rba/ucsdb2/NODE0000/SQL00001/
  First connection timestamp: 07/31/2005 05:00:25.898506
  Event Monitor Start time:   07/31/2005 05:28:27.153579
-------------------------------------------------------------------------
3) Deadlock Event ...
  Deadlock ID:   1
  Number of applications deadlocked: 2
  Deadlock detection time: 07/31/2005 05:31:10.961346
  Rolled back Appl participant no: 2
  Rolled back Appl Id: *LOCAL.ucsdb2.050730200029
  Rolled back Appl seq number: : 0005

4) Connection Header Event ...
  Appl Handle: 9
  Appl Id: *LOCAL.ucsdb2.050730200029
  Appl Seq number: 0005
  DRDA AS Correlation Token: *LOCAL.ucsdb2.050730200029
  Program Name    : db2bp
```

```
  Authorization Id: AIXDBA
  Execution Id    : aixdba
  Codepage Id: 1208
  Territory code: 1
  Client Process Id: 553002
  Client Database Alias: VOL78RBA
  Client Product Id: SQL08021
  Client Platform: Unknown
  Client Communication Protocol: Local
  Client Network Name: kanaga.almaden.ibm.c
  Connect timestamp: 07/31/2005 05:00:29.436147

5) Deadlocked Connection ...
  Deadlock ID:   1
  Participant no.: 2
  Participant no. holding the lock: 1
  Appl Id: *LOCAL.ucsdb2.050730200029
  Appl Seq number: 0005
  Appl Id of connection holding the lock: *LOCAL.ucsdb2.050730200025
  Seq. no. of connection holding the lock: 0005
  Lock wait start time: 07/31/2005 05:31:01.803163
  Lock Name       : 0x0007018F000CA4030000000052
  Lock Attributes : 0x00000000
  Release Flags   : 0x40000000
  Lock Count      : 1
  Hold Count      : 0
  Current Mode    : none
  Deadlock detection time: 07/31/2005 05:31:10.980656
  Table of lock waited on      : S_CONTACT
  Schema of lock waited on     : SIEBEL
  Tablespace of lock waited on : TBS_16K
  Type of lock: Row
  Mode of lock: X   - Exclusive
  Mode application requested on lock: U   - Update
  Node lock occurred on: 0
  Lock object name: 828419
  Application Handle: 9
  Deadlocked Statement:
    Type     : Dynamic
    Operation: Execute Immediate
    Section  : 203
    Creator  : NULLID
    Package  : SQLC2E06
    Cursor   :
    Cursor was blocking: FALSE
    Text     : UPDATE siebel.S_CONTACT SET WORK_PH_NUM ='1008529252',
MODIFICATION_NUM = 3, LAST_UPD_BY = '0-1', LAST_UPD = CURRENT_TIMESTAMP WHERE
ROW_ID ='88-24L7E' AND MODIFICATION_NUM = 3
  List of Locks:
```

```
    Lock Name                   : 0x0007018F000CA4010000000052
    Lock Attributes             : 0x00000020
    Release Flags               : 0x40000000
    Lock Count                  : 1
    Hold Count                  : 0
    Lock Object Name            : 828417
    Object Type                 : Row
    Tablespace Name             : TBS_16K
    Table Schema                : SIEBEL
    Table Name                  : S_CONTACT
    Mode                        : X   - Exclusive

    Lock Name                   : 0x0007018F000000000000000054
    Lock Attributes             : 0x00000000
    Release Flags               : 0x40000000
    Lock Count                  : 2
    Hold Count                  : 0
    Lock Object Name            : 399
    Object Type                 : Table
    Tablespace Name             : TBS_16K
    Table Schema                : SIEBEL
    Table Name                  : S_CONTACT
    Mode                        : IX  - Intent Exclusive

  Locks Held:    4
  Locks in List: 4

6) Connection Header Event ...
  Appl Handle: 8
  Appl Id: *LOCAL.ucsdb2.050730200025
  Appl Seq number: 0005
  DRDA AS Correlation Token: *LOCAL.ucsdb2.050730200025
  Program Name    : db2bp
  Authorization Id: AIXDBA
  Execution Id    : aixdba
  Codepage Id: 1208
  Territory code: 1
  Client Process Id: 536666
  Client Database Alias: VOL78RBA
  Client Product Id: SQL08021
  Client Platform: Unknown
  Client Communication Protocol: Local
  Client Network Name: kanaga.almaden.ibm.c
  Connect timestamp: 07/31/2005 05:00:25.898506

7) Deadlocked Connection ...
  Deadlock ID:   1
  Participant no.: 1
```

```
  Participant no. holding the lock: 2
  Appl Id: *LOCAL.ucsdb2.050730200025
  Appl Seq number: 0005
  Appl Id of connection holding the lock: *LOCAL.ucsdb2.050730200029
  Seq. no. of connection holding the lock: 0005
  Lock wait start time: 07/31/2005 05:31:00.210997
  Lock Name       : 0x0007018F000CA4010000000052
  Lock Attributes : 0x00000000
  Release Flags   : 0x40000000
  Lock Count      : 1
  Hold Count      : 0
  Current Mode    : none
  Deadlock detection time: 07/31/2005 05:31:11.015177
  Table of lock waited on      : S_CONTACT
  Schema of lock waited on     : SIEBEL
  Tablespace of lock waited on : TBS_16K
  Type of lock: Row
  Mode of lock: X   - Exclusive
  Mode application requested on lock: U   - Update
  Node lock occurred on: 0
  Lock object name: 828417
  Application Handle: 8
  Deadlocked Statement:
    Type     : Dynamic
    Operation: Execute Immediate
    Section  : 203
    Creator  : NULLID
    Package  : SQLC2E06
    Cursor   :
    Cursor was blocking: FALSE
    Text     : UPDATE siebel.S_CONTACT SET WORK_PH_NUM ='4081112222',
MODIFICATION_NUM = 3, LAST_UPD_BY = '0-1', LAST_UPD = CURRENT_TIMESTAMP WHERE
ROW_ID ='88-24L6D' AND MODIFICATION_NUM = 3
  List of Locks:
      Lock Name                  : 0x0007018F000CA4010000000052
      Lock Attributes            : 0x00000000
      Release Flags              : 0x40000000
      Lock Count                 : 1
      Hold Count                 : 0
      Lock Object Name           : 828417
      Object Type                : Row
      Tablespace Name            : TBS_16K
      Table Schema               : SIEBEL
      Table Name                 : S_CONTACT
      Mode                       : U   - Update

      Lock Name                  : 0x0007018F000CA4030000000052
      Lock Attributes            : 0x00000020
      Release Flags              : 0x40000000
```

```
    Lock Count                  : 1
    Hold Count                  : 0
    Lock Object Name            : 828419
    Object Type                 : Row
    Tablespace Name             : TBS_16K
    Table Schema                : SIEBEL
    Table Name                  : S_CONTACT
    Mode                        : X   - Exclusive

    Lock Name                   : 0x0007018F000000000000000054
    Lock Attributes             : 0x00000000
    Release Flags               : 0x40000000
    Lock Count                  : 2
    Hold Count                  : 0
    Lock Object Name            : 399
    Object Type                 : Table
    Tablespace Name             : TBS_16K
    Table Schema                : SIEBEL
    Table Name                  : S_CONTACT
    Mode                        : IX  - Intent Exclusive

  Locks Held:    5
  Locks in List: 5
```

Based on the output, we can determine that:

- There are two participating agents with application handles 9 and 8.
- Application handle 9 is updating a row and has a row lock and exclusive lock on the table `S_CONTACT`.
- Application handle 8 is also updating the same row and has a row lock and exclusive lock on the table `S_CONTACT`.

In this situation, both application handle 8 and application handle 9 will wait indefinitely for each other's lock until an external event causes one or both of them to roll back.

Example 8-16 shows the steps to deactivate and drop the event monitor.

Example 8-16 Deactivate and drop the event monitor

```
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 "set event monitor seblmon state 0"
kanaga.almaden.ibm.com:aixdba:/aixdba>db2 "drop event monitor seblmon"
```

Note: In DB2 UDB version 8, the deadlock event monitor provides more information which can further help system and database administrators determine why deadlocks occur. For example, the monitor now identifies which statements are involved in the deadlock, and it pinpoints which locks each application involved in the deadlock is holding.

Resolving deadlocks

There are three possible causes for deadlocks:

- Lock wait
- Lock escalations
- Application logic and user SQL

For resolution or reduction of lock wait and lock escalations, use the techniques mentioned in “Resolving lock escalations” on page 232.

It is worth considering the following points to reduce deadlocks caused due to application logic and user SQL:

- Release all locks held before switching contexts.
- Do not access a given object from more than one context at a time.
- Avoid concurrent DML, if possible.

For other suggestions, refer to “Resolution of lock wait and timeout” on page 230.

8.6.4 New feature of DB2 UDB V8.2 in the locking context

An individual session can now specify a lock wait mode strategy, which is used when the session requires a lock that it cannot obtain immediately. The strategy indicates whether the session will:

- Return an SQLCODE and SQLSTATE when it cannot obtain a lock.
- Wait indefinitely for a lock.
- Wait a specified amount of time for a lock.
- Use the value of the lock timeout database configuration parameter when waiting for a lock.

The lock wait mode strategy is specified through the new `SET CURRENT LOCK TIMEOUT` statement, which sets of the `CURRENT LOCK TIMEOUT` special register. The `CURRENT LOCK TIMEOUT` special register specifies the number of seconds to wait for a lock before returning an error indicating that a lock cannot be obtained.

Traditional locking approaches can result in applications blocking each other. This happens when one application must wait for another application to release its lock. Strategies to deal with the impact of such blocking usually provide a mechanism to specify the maximum acceptable duration of the block. That is the amount of time that an application will wait prior to returning without a lock. Previously, this was only possible at the database level by changing the value of the `LOCKTIMEOUT` database configuration parameter.

8.7 Buffer pool

Buffer pool is the place where all of the data pages are stored as the data is being read from disk or newly added data is being inserted. Buffer pool plays an important role in reducing the I/O to the disk. Of all the DB2 UDB configuration parameters, buffer pool has the biggest impact on database server performance.

Properly defining buffer pools is one of the major keys to having a well performing system. With 32-bit operating systems, it is important to be aware of the limit on shared memory, which restricts a database's buffers pools (that is, database global memory) from exceeding the following limits (64-bit systems do not have such a limit):

- AIX: 1.75 GB
- Linux: 1.75 GB
- Solaris: 3.3 GB
- HP-UX: 800 MB
- Windows: 2-3 GB (Use '/3GB' switch in `boot.ini` on Windows NT and Windows 2000)

Use the following formula for calculating your approximate database global memory usage:

```
buffer pools + dbheap + util_heap_sz + pkgcachesz + aslheapsz +
locklist + approx. 10% overhead
```

When you create a DB2 UDB database, a default 4K buffer pool is automatically created. For Siebel out-of-the-box implementation, you need three buffer pools of different sizes, namely, 4K, 16K, and 32K. Once the database is created, you can create three new buffer pools and drop the `IBMDEFAULTBP` (only a suggestion so your buffer pool names are uniform and easy to identify).

Figuring out how many buffer pools you need and how big they should be is always a task involving skill, experience, and testing. You should carefully evaluate the number of buffer pools you want to create. If you have a lot of memory, then create one buffer pool for each temporary table space and one for

each table space. You always want to create separate buffer pools so you can assign them to different table spaces, where the data is grouped by usage.

In an OLTP database system, smaller multiple buffer pools may be better than a few huge ones. By having multiple buffer pools, you can pin the frequently used tables in a separate buffer pool and away from tables where the data is infrequently used. The idea is to keep the most frequently accessed rows in a buffer pool. Sharing a buffer pool with tables that are accessed randomly or infrequently can cause “polluting” of your buffer pool, by consuming space and possibly pushing a frequently accessed row to disk for a row which may never be accessed again. Keeping indexes in their own buffer pool can also significantly improve performance when indexes are heavily used (for example, index scans).

In the Siebel environment, we suggest the following for machines with small to medium-sized memory. This is a rule of thumb and will work for most Siebel installs.

- A large 4K buffer pool for indexes
- A medium 4K buffer pool for data and temporary table space
- A medium 16K buffer pool for data and temporary table space
- A medium 32K buffer pool for data and temporary table space

Tip: The largest tables identified in Siebel sizing review should be placed in separate table spaces. Doing it at the beginning will provide great flexibility in buffer pool usage later in the life of the project.

In a large implementation with a lot of memory, separate out the temporary table spaces. In general, three or four large properly allocated buffer pools with one or two small ones for random data tables are better than multiple small ones.

Do not break it out into too many buffer pools, since this can lead to side effects such as:

- Multiple buffer pools to manage.
- Too much time will be spent figuring out which tables need to go where.
- You may spend time on moving tables around, when it may be better spent on monitoring.

Never allocate more memory than you have available or you will incur costly OS memory paging. Generally speaking, it may be hard to know how much memory to initially allocate to each buffer pool without monitoring. We recommend allocating about 70-80% of the available memory to the buffer pools on a dedicated DB server.

Note: Ensure that you have adequately tested the effects allocating 70-80% of memory to buffer pool. In some rare cases, we have seen paging due to this high setting.

8.8 Table space

This is a key DBA job and involves mapping of logical database design to physical storage. All of the tuning we have done until now would be useless if you have not mapped the storage properly.

There are two types of table spaces in DB2 UDB, system managed space (SMS) and database managed space (DMS). In SMS, the operating system is responsible for allocating and managing storage space allocated to the database. The table spaces grow over time as the data is being inserted into the database. In DMS, the database is responsible for allocating storage space and managing it.

We recommend using DMS for all table spaces except temporary table space. In DMS, you have two options:

- Raw devices are logical volumes (AIX and Linux) and drive partition (Windows).
- Files which are of a preallocated size.

Starting with DB2 UDB V8, you can dynamically redistribute/decrease table space size. This helps in better database management.

Note: If you add containers to your production database during prime time since you are running out of space, be careful that it will start the redistribution of data right away unless you specify deferred.

In Siebel out-of-the-box implementation, the following table spaces exist:

- System catalog table space
- 4K temporary table space
- 16K temporary table space
- 32K temporary table space
- 4K data table space
- 16K data table space
- 32K data table space
- 4K index table space

For better performance, we recommend you split out the index, in other words, use something like the example below. The names in parentheses are arbitrary names.

- System catalog table space
- 4K temporary table space
- 16K temporary table space
- 32K temporary table space
- 4K data table space (`siebel_4k`)
- 16K data table space (`siebel_16k`)
- 32K data table space (`siebel_32k`)
- 4K index table space for tables in `siebel_4k` (`siebel_4ki`)
- 16K index table space for tables in `siebel_16k` (`siebel_16ki`)
- 32K index table space for tables in `siebel_32k` (`siebel_32ki`)

Notes: Siebel indexes are in a separate index table space, out-of-the-box. If you want to have more than one index table space, then you would need to take additional steps.

We created all of our index table spaces with page sizes of 4k. By using 16k or 32k, we would have wasted some space in the table space. In DB2 UDB, you can fit 255 records per page. There is a restriction creating an index, the sum of the stored lengths of the specified columns must not be greater than 1024.

As you understand the data and the usage better, you can split out tables such as `S_PARTY`, `S_CONTACT`, and `S_ORG_EXT`, `and` so on for versatility and separation of data.

You should assign multiple containers to each table space regardless of whether it is temporary or data. DB2 UDB writes data in a round robin fashion across containers using one extent at a time. Try to separate your data and index data to separate areas of your storage, in other words, allocate containers for `siebel_4k` and `siebel_4ki` on different adapters of the storage. Share storage areas between `siebel_4k and siebel_16ki`, so that if there is a query running with index only access, it will pull data from `siebel_4ki` and `siebel_16ki` which are in two different areas of storage.

Please pay attention to extent size when you create the table spaces. Once they are created, they cannot be easily altered. The rule of thumb is based on the average size of a table in the table space:

- Less than 25 MB, use an extent size of 8
- Between 25 and 250 MB, use an extent size of 16
- Between 250 MB and 2 GB, use an extent size of 32
- Greater than 2 GB, use an extent size of 64

If the table space resides on a disk array, set the extent size to the stripe size (that is, data written to one disk of the array).

Prefetch size can be changed easily using `ALTER TABLESPACE` command. The optimal setting seems to be:

```
Prefetch Size = (# Containers of the table space on different physical
disks) * Extent Size
```

If the table space resides on a disk array, set it as:

```
PREFETCH SIZE = EXTENT SIZE * (# of non-parity disks in array).
```

9

High availability and disaster recovery

High availability is the requirement for the application to be available for both incoming and outgoing data requests. The incoming data requests include users (sales) and interfaces feeding data to the system, the outgoing data requests are users (call center) and downstream applications such as executive reports. The goal is decrease or eliminate downtime.

Disaster recovery is the requirement/ability to protect and restore data when a catastrophic failure (hardware and/or other natural/human-made disaster) happens. Without a disaster recovery plan/execution in place, this kind of failure can lead to the total loss of data beyond recovery.

Siebel (CRM) is considered as a critical enterprise application and hence a need for high availability and the ability to recover from disaster.

This chapter provides the features in DB2 UDB that provide these capabilities and how they can be utilized in a Siebel database application environment. We also describe how to enable high availability and disaster recovery for Siebel applications. We go through features, what to consider, and finally how we have set up the test/lab environment. DB2 UDB is full of features that reduce the planned downtime. In this chapter, our concentration is on unplanned downtime and how to deal with it. DB2 UDB has a high availability and disaster recovery (HADR) solution built into the software starting at V8.2.

9.1 Components of a Siebel Enterprise

Siebel Enterprise can be broken down into different parts so we can plan for a good high availability and disaster recovery solution.

- Database server:

  Database server is the foundation of a Siebel Enterprise. This is where all of the data is stored. Availability of the database server is key for the application to be up and running.

- Siebel Gateway Name Server:

  Siebel Gateway Name Server holds the entire enterprise configuration information. If the Siebel Gateway Name Server fails, no new components can be added or started and Enterprise Server Administration functions are unavailable.

  High Availability is recommend for Siebel Gateway Name Server to avoid single point of failure.

- Siebel Server:

  If you have multiple Siebel Servers running that are not clustered, these should be load-balanced. There are two types of load balancing methods for Siebel Server components. They are Siebel load balancing and third-party HTTP load balancers. Siebel load balancing is part of the Siebel installation process. The third-party load balancers can be hardware-based such as an intelligent network switch or software-based.

  If the requirements for the Siebel Server are 24 hours and seven days a week, then high availability is recommend for Siebel Server.

  The redundancy in a Siebel Server can be increased by having multiple Siebel Servers with a load balancer.

- Siebel File System:

  Siebel File System is a shared file system directory. All Siebel Servers require access to the Siebel File System. Since Siebel File System is also a single point of failure, it is recommended to mirror the high speed RAID disks which contain the Siebel File System. Siebel File System can be placed on the Siebel Gateway Name Server and high availability can be configured for the Siebel File System also.

- Siebel Web Server:

  Install at least two Siebel Web Servers and use HTTP load balancers to load balance the HTTP connections. Set up the load balancer to allow user requests to fail from one Siebel Web server to another.

9.2 Database server

Database server is comprised of physical database server, network, and storage. For a good high availability solution, you need redundancy in each of these components and they are interdependent on each other. They work together to provide a complete solution.

- Physical database server:

  You chose DB2 UDB with Siebel because you wanted a DBMS that is fast, high performing, reliable, and scalable with good query and utility performance. To have a good database environment, you also have to pick hardware that is reliable with built-in redundancies such as IBM pSeries servers. Running these servers with AIX 5.2 or higher gives you multiple options for high availability.

  DB2 UDB ships special scripts and commands that support various clustering software on all the platforms that DB2 UDB supports. These scripts can be modified by DBAs to map to their high availability solution. These scripts define how DB2 UDB will behave when a failure or a recovery of a system takes place.

  In a DB2 UDB single-partitioned environment, the script performs the following actions (depending on how you have configured the HA cluster):

  - If the primary server fails, it will start DB2 UDB on the standby server.
  - If the failed primary server becomes available (reintegrates) into the HA cluster, it will stop the database on the standby server and start it on the primary server.

- Network:

  A key part of high availability is to be able to communicate with the database server. Providing separate communication paths for each of the following components will increase your availability.

  - Users:
    This part of communications is done on a public network and having multiple routes for these communications is key. Have more than one card in your machine such that you can fail over from one network card to another. Additionally, having these network cards connected to separate routers just in case of a router failure provides an extra layer of protection.
  - Intra-application communications:
    This part of the network should be isolated and communication should be done on a private network. To provide zero latency in these communications, we suggest having two high performance network cards, connected to a high availability network server. Keep the service

communications such as backups to network storage manager on a separate network path.

- Storage:

  Since the data you have needs to be stored, having a good storage subsystem is key to database availability. RAID5 is the simplest form of highly available storage. To do RAID5, you need at least three drives. If you have a storage subsystem such as IBM ESS, it is highly scalable, high performing, load balancing, and high availability storage.

Regardless of which high availability solution you pick, the following three things play an important role:

- Time needed to detect a failure (referred to as heartbeat).
- Time needed to move the necessary resources (network, storage, and so on) to the new machine.
- Time needed for crash recovery to be complete. Always expect your system to be highly active.

9.2.1 HA options for DB2 UDB server

There are multiple design solutions available for high availability. In this section, we discuss three most popular solutions:

- Idle standby
- Active standby
- Mutual takeover

Idle standby

In an idle standby, there is an extra server that will take over the resources from the primary database server. Which means that there would be no degradation in performance after failover. Figure 9-1 on page 249 shows a simple layout.

At the first look, this may look more costly since there is a idle machine in the framework. But this extra machine can be put to work by using it as a test and/or development system. Since the standby is of the same size as the production, the benchmark and performance numbers would be more closer to production. Also, DB2 UDB offers special pricing in such an environment.

In Figure 9-1 on page 249, you will notice that there are three network adapters configured:

- First is the boot adapter, this is the IP address from which the machine boots.

- The second adapter is where all the private network communications are going through such as heartbeat monitor, network storage for backups, and so on.
- The third adapter has the IP address that is active all the time. This is used for all service operations such as problem determination, during failover. This is also used by the maintenance team to perform maintenance on weekends.

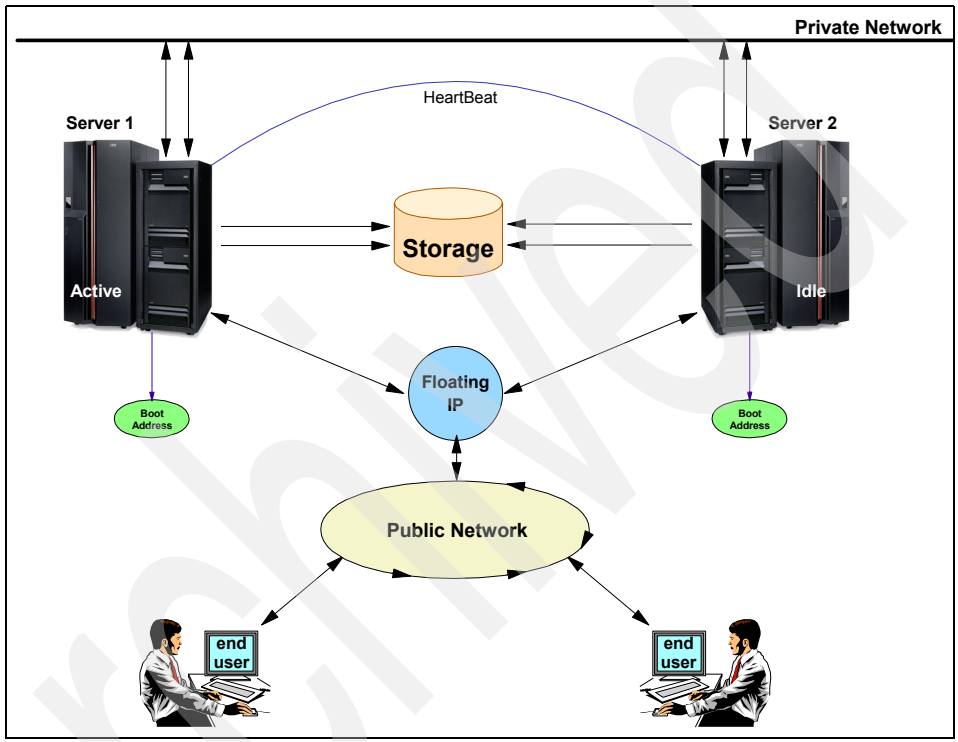

Figure 9-1 Simple layout of a idle standby HACMP solution

Table 9-1 gives the advantages and disadvantages of idle standby.

Table 9-1 Pros and cons of an idle standby solution

Pros	Cons
Simple high availability solution	Cost may be high, since there is a machine that is idle.
No changes at Siebel Server (no need to make changes to node directory at the db2 client)	Does not protect against storage hardware failure. (This can be solved by having RAID5 or buying new network storage solution).

Pros	Cons
Performance would not be hindered if the standby/failover server matches the specifications of primary server.	Performance might be affected if the failover server is smaller than the primary server.
User may never notice an outage (Siebel application does a reconnect if it senses loss of DB2 connection).	
Database is always in sync.	

Active standby

As the name suggests, the standby server will carry the workload from primary server and also its own normal workload after failover. In this approach, there may not be any performance degradation after the failover. Most people would implement such a solution, where the secondary server is running a reporting, audit, and temporary database. If there is a performance degradation then those databases are shut down to give full power to the primary database. DB2 UDB has a separate pricing for such an environment. Figure 9-2 on page 251 shows the test configuration for active standby.

In the picture you will notice that we have three network adapters configured:

- First is the boot adapter, this is the IP address from which the machine boots.
- The second adapter is where all the private network communications are going through such as a heartbeat monitor, network storage for backups, and so on.
- The third adapter has the IP address that is active all the time. This is used for all service operations such as problem determination, during failover. This is also used by maintenance team to perform maintenance on weekends.

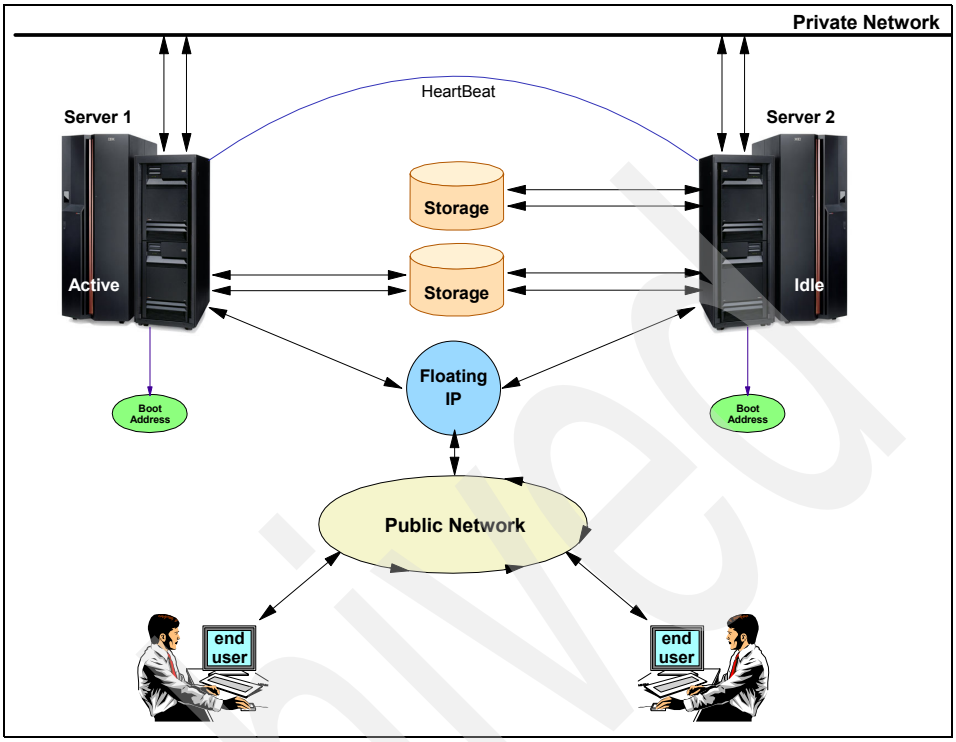

Figure 9-2 Simple active standby HACMP solution

Table 9-2 lists the advantages and disadvantages of active standby HA solution.

Table 9-2 Pros and cons of an active standby HA solution

Pros	Cons
Simple high availability solution.	Performance may be affected since the failover server is also carrying workload for other applications running on it. (This can be minimized by shutting down non-critical applications.)
Cost is minimal as the failover server is performing work.	Does not protect against storage hardware failure. (This can be solved by having RAID5 and/or buying new network storage solution.)
No changes at Siebel Server (no need to make changes to node directory at the db2 client).	

Pros	Cons
User may never notice an outage. (Siebel application does a reconnect if it senses loss of DB2 connection).	
Database is always in sync.	

Mutual takeover

In this approach, the two servers mutually take over each other's workload while still performing their workload. Depending on their normal workload, you may or may not see any changes in performance. DB2 UDB has separate pricing for such an environment.

Figure 9-3 illustrates the mutual takeover solution.

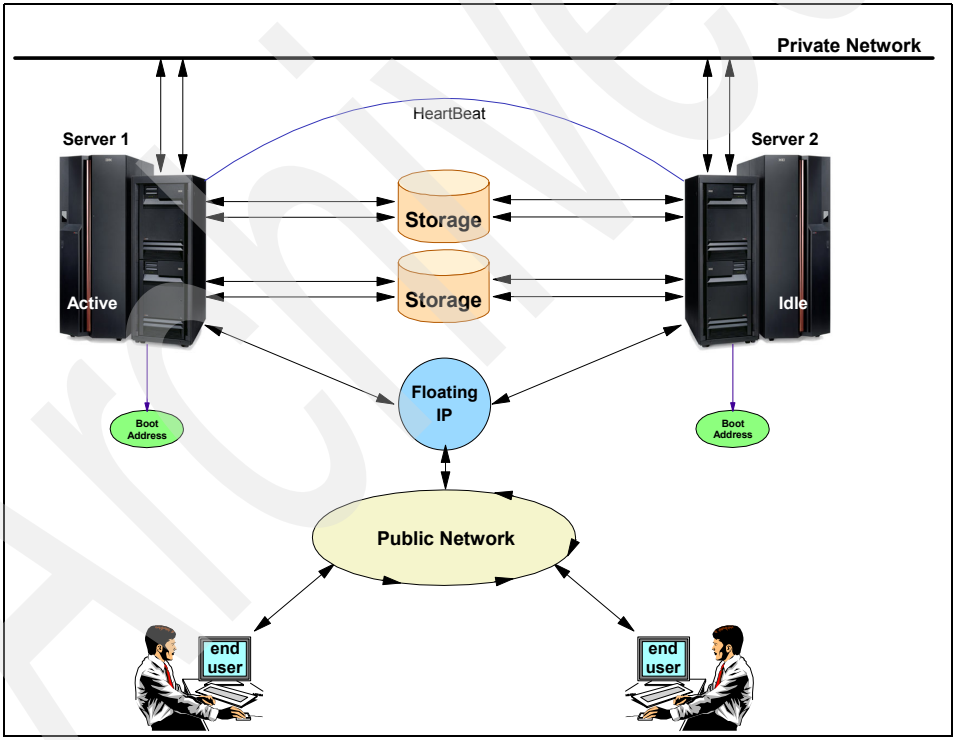

Figure 9-3 Layout of a mutual takeover solution

In the picture, you will notice that we have three network adapters configured.

- First is the boot adapter, this is the IP address from which the machine boots.

- The second adapter is where all the private network communications are going through such as the heartbeat monitor, network storage for backups, and so on.
- The third adapter has the IP address that is active all the time. This is used for all service operations like problem determination, during failover. This is also used by the maintenance team to perform maintenance on weekends.

Table 9-3 lists the advantages and disadvantages of the mutual takeover HA solution.

Table 9-3 Pros and cons of a mutual takeover HA solution

Pros	Cons
Simple high availability solution.	Performance may be affected, since the failover server is also carrying workload for other applications running on it. (This can be minimized by shutting down non-critical applications.)
Cost is minimal as the failover server is performing work.	Does not protect against storage hardware failure. (This can be solved by having RAID5 and/or buying new network storage solution.
No changes at Siebel Server (no need to make changes to node directory at the DB2 client).	
Users may never notice an outage. (Siebel application does a reconnect if it senses loss of DB2 connection).	
Database is always in sync.	

9.2.2 Information on the lab environment

In our lab environment, we demonstrate clustering at AIX and Windows environments.

AIX

At AIX, the HA solution (High-Availability Cluster Multi-Processing (HACMP) is used to host cluster servers `KANAGA` and `ATLANTIC`. The application server is DB2 UDB database server. We provide the steps for setting up idle standby and active standby. The general setup procedures are:

- Install and configure HACMP on the clustered servers.
- Set up DB2 UDB databases server for clustering.

- Configure HACMP Application Server.

The following are the system specifications.

- Server-1 system information:
  - Hostname = `KANAGA`
  - p630, 4-way, 4 GB RAM, 3 x 16 GB internal SCSI drives, three network cards
- Server-2 system information:
  - Hostname = `ATLANTIC`
  - p630, 4-way, 4 GB RAM, 3 x 16 GB internal SCSI drives, three network cards
- A DS4500 storage has been connected to both `KANAGA` and `ATLANTIC`. 1 TB of storage space has been assigned to both servers.

Windows

Microsoft Cluster Services (MSCS) is used to host cluster servers `WISLA` and `LOCHNESS`.

The following are the system specifications:

- Server-1 system information:
  - Hostname = `WISLA`
  - Net Vista, 4-way, 2 GB RAM, 72 GB internal drive, two network cards
- Server-2 system information:
  - Hostname = `LOCHNESS`
  - Net Vista, 4-way, 2 GB RAM, 72 GB internal drive, two network cards
- Both systems share a storage area.

9.2.3 Installation and configuration of HACMP on AIX for DB2 UDB

The high availability clustering solution on AIX is called HACMP (High-Availability Cluster Multi-Processing).

HACMP does the following:

- It continuously sends the heartbeats between the servers in the cluster to identify if and when any of them are down.
- It provides tools to define and manage the group of resources that participate in the high availability setup, including shared disks, network adapters, and application servers that will handle the starting and stopping of the database management subsystem.

- It handles the failing over of the resource groups from one system to the other as needed after a failure or after reinsertion of a failed server back in to the cluster.

HACMP terminology

A brief description of HACMP terminology is given below:

- **Cluster:** A collection of systems organized into a network for the purpose of sharing resources and communicating with each other.
- **Node:** A node is a server running AIX and HACMP that is defined as a part of the cluster. Nodes may share a set of resources-disks, volume groups, file systems, network, network IP addresses, and applications.
- **Resource:** Resources are components of the cluster configuration that can be moved from one node to another.
- **Resource Group:** All resources necessary to provide High Availability application are grouped together in a resource group.
- **Service IP Address:** The service IP address is an IP address used for client access. The service IP address can be a shared service IP address or Node-bound service IP address. A shared service IP address is configured on multiple nodes and is part of a resource group that can be active only on one node at a time. The Node-bound service IP address is configured on only one node and is not shared by multiple nodes. Service IP address is also referred to as Floating IP Address.
- **IP Address takeover** (IPAT): IPAT is the mechanism for recovering the service IP address by moving it to another network adapter or another node, when the initial physical network adapter fails. IPAT can be configured two ways:
  - *IPAT via IP replacement*: The service IP address replaces the boot IP address.
  - *IPAT via IP aliasing*: The service IP address is aliased onto an existing communication interface, without changing the base address of the interface.
- In our lab environment, we used two IBM pSeries 630 servers called `KANAGA` and `ATLANTIC`. We installed AIX 5L V5.2 ML4. To find the latest fixes available for HACMP and AIX 5L, refer to:

  http://techsupport.services.ibm.com/server/hacmp

Installation and configuration of HACMP

This section describes the HACMP installation and configuration procedures.

Installing and configuring HACMP

Following are the steps you follow to install and configure.

1. **HACMP planning**

   DB2 UDB ESE software should be installed on local disks on the database servers. DB2 instances and database should be created on the shared disks volume group. We chose the shared disks access to be non-concurrent access mode. Only one node will have access to the shared disks in non-concurrent access mode.

   While creating the shared volume groups on the shared disks, make sure the volume group's major numbers are the same on all nodes. **Activate volume groups automatically** at system restart should be set to `NO`. Make sure the user ID's uids and group's gids are the same on both the servers. Make sure that **`rsh`** can be used. To check if `rsh` is enabled in `/etc/inetd.conf`, execute the following command:

```
#cat /etc/inetd.conf |grep shell
shell   stream  tcp6    nowait  root    /usr/sbin/rshd          rshd
```

   Make sure `/.rhosts` file is configured properly. A sample `/.rhosts` file from `KANAGA` is shown in Example 9-1.

   Execution of **`rsh atlantic`** and **`rsh kanaga`** should work.

Example 9-1 /.rhosts file from KANAGA

```
+ atlantic
atlantic.almaden.ibm.com
atlantic
hakan
```

   The `/etc/hosts` file should contain service IP address, standby IP address, and boot IP address. A sample `/etc/hosts` file from `KANAGA` is shown in Example 9-2.

Example 9-2 /etc/hosts file from KANAGA

```
127.0.0.1       loopback localhost      # loopback (lo0) name/address
# IP addresses for en0 (Boot IP addresses - net_ether_01)
9.1.39.90       kanaga kanaga.almaden.ibm.com kanaga
9.1.39.92       atlantic atlantic.almaden.ibm.com atlantic

# Ip addresses for en1 (Administration Use)
9.1.39.66       kanagak kanaga.almaden.ibm.com
9.1.39.67       atlantick atlantic.almaden.ibm.com

# Ip addresses for en2 (Secondary Network - net_ether_02)
192.168.0.5     atlantics atlantic.almaden.ibm.com atlantics
192.168.0.4     kanagas  kanaga.almaden.ibm.com kanagas

# Service IP Address
```

Table 9-4 shows the HACMP cluster planning information. The cluster name is db2k:

Table 9-4 HACMP cluster information

Cluster name	**db2k**
Participating Nodes	KANAGA, ATLANTIC
Service IP Address	hakan - 9.1.39.156
Application Server Name	db2_siebel
Resource Group Name	db2_rg
Share Volume Group Name	datavg

Table 9-5 shows the HACMP network information:

Table 9-5 TCP/IP network adapters

Node name	**Adapter name**	**Adapter function**	**IP address**	**Interface name**
KANAGA	KANAGA	boot	9.1.39.90	en0
KANAGA	KANAGAK	administrator use	9.1.39.66	en1
KANAGA	KANAGAS	standby	192.168.0.4	en2
ATLANTIC	ATLANTIC	boot	9.1.39.92	en0
ATLANTIC	ATLANTICK	administrator use	9.1.39.67	en1
ATLANTIC	ATLANTICS	standby	192.168.0.5	en2
	HAKAN	service	9.1.39.156	en0

Table 9-6 shows the HACMP application server information:

Table 9-6 Application Server information

Server Name	db2_siebel
Start script	/usr/es/sbin/cluster/events/start_db2_siebel
Stop script	/usr/es/sbin/cluster/events/stop_db2_siebel

Table 9-7 on page 258 shows the HACMP resource group information:

Table 9-7 Resource Group information

Resource Group Name	db2_rg
Participating Nodes	KANAGA, ATLANTIC
Service IP Address	HAKAN
application server	db2_siebel
Volume Group	datavg
Startup Policy	Online on Home node only
Failover Policy	Failover to next priority node in the list
Fallback Policy	Never Fallback

2. **HACMP software Installation:**

   Install HACMP software on both the nodes. To install the HACMP software on a server node, perform the following steps:

   a. If you are installing directly from the installation media, such as a CD-ROM or from a local repository, enter fast path command:

      ```
      smitty install_all
      ```

      SMIT displays the *Install and Update from ALL Available Software* window.

   b. Enter the device name of the installation medium or install directory in the *Input device/directory for software* field and press Enter.

   c. To select the software to install, press F4 for a software listing, or enter **all** to install all server and client images. Select the packages you want to install according to your cluster configuration. Make sure you choose **Yes** in the *Accept license agreement*.

   After installing HACMP software on both the servers, reboot the servers.

   To check the consistency of the installation we ran **`lppchk -v`** and **`lppchk -c cluster*`**. **`lppchk –v`** will check if the installed file sets are in a consistent state and will show the file sets that need to be installed for the system to return to a consistent state. **`lppchk –c cluster*`** will check if all the cluster installation file sets are in a consistent state. If cluster communication daemon `clcomd` process is not running, start `clcomd` on the servers with the following command: **`startsrc –s clcomdES`**

3. **Define a cluster:**

   Use AIX command:

   ```
   smit hacmp
   ```

Select Extended Configuration → Extended Topology Configuration → Configure a HACMP cluster → Add/Change/Show an HACMP Cluster

Enter a name of the cluster in ASCII text string. It can be different from the host name. In our lab environment, the name of the cluster was `db2k`, see Figure 9-4.

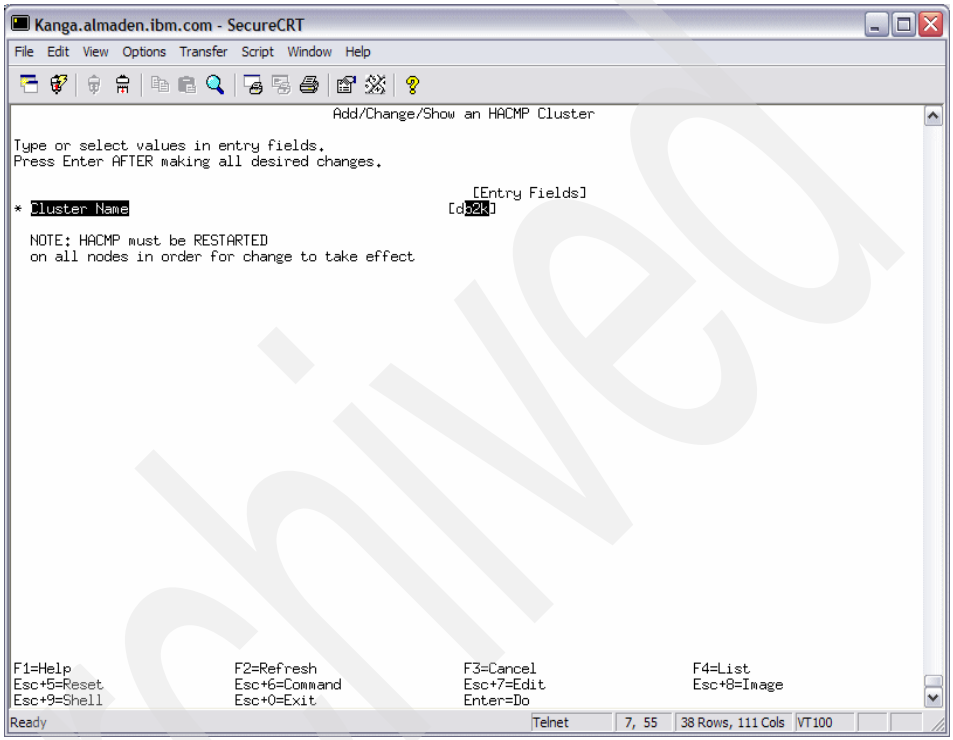

Figure 9-4 Add HACMP cluster name

4. **Define cluster nodes**:

   Use `SMIT`:

   ```
   smit hacmp
   ```

   Select Extended Configuration → Extended Topology Configuration → Configure HACMP nodes → Add a Node to the HACMP cluster

   Enter the *Node name* and *Communication Path to the Node* as shown in Figure 9-5 on page 260.

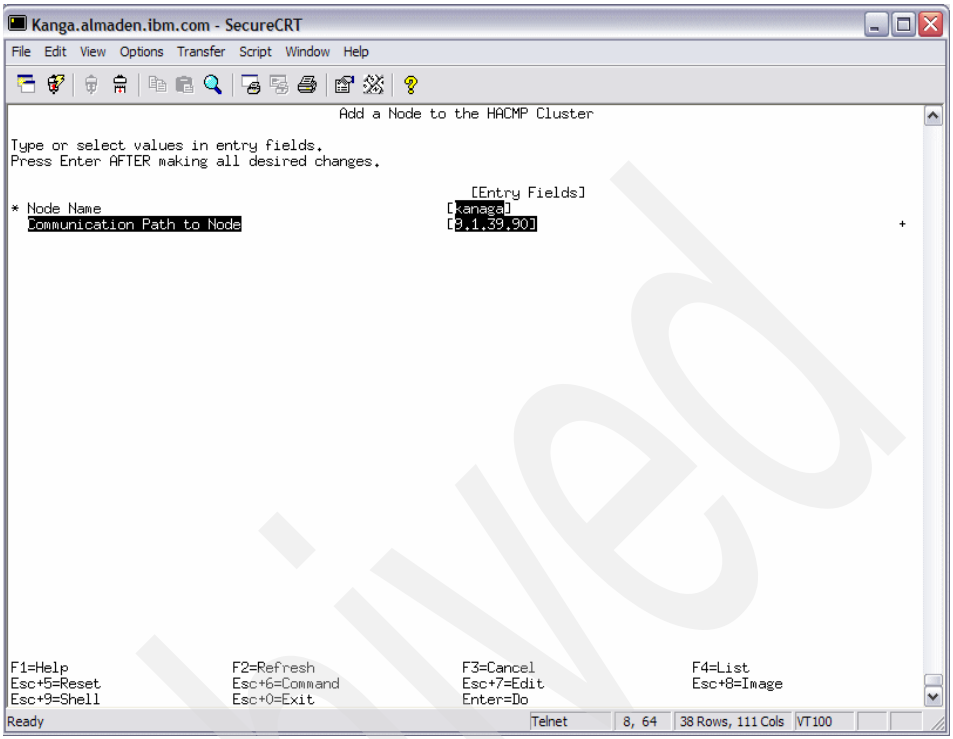

Figure 9-5 Add kanaga node to the cluster

Figure 9-6 on page 261 shows how to add the second node to the cluster.

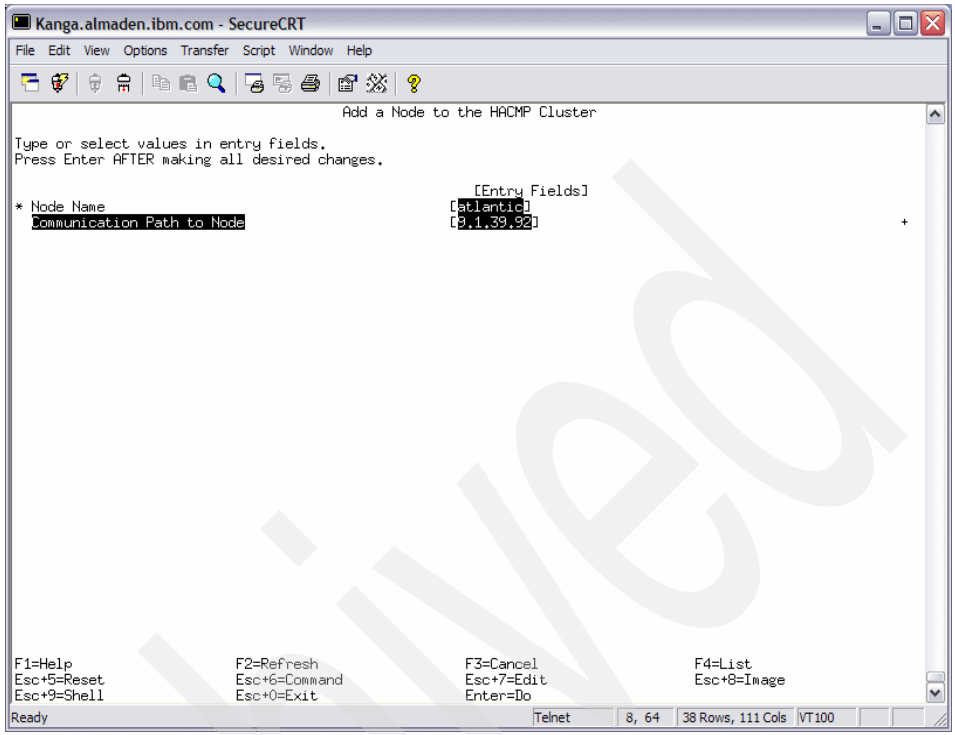

Figure 9-6 Addition of the second node

5. **Define networks**

   Once you have defined all cluster node names, continue setting up your cluster topology by defining the HACMP networks and associated adapters. You can define the cluster topology by defining the networks and adapters or you can take advantage of the automatic network discovery feature. The `/etc/hosts` file must contain all adapter labels and associated IP addresses for HACMP to gather adapter information for creating networks. The automatic network discovery feature simplifies the process of defining the HACMP cluster topology. Using SMIT, you select the type of network to add (IP-based or non IP-based). Then you can run the automatic discovery process either locally or cluster-wide. To perform automatic network discovery, execute the following:

   ```
   smit hacmp
   ```

   Extended Configuration → Discover HACMP-related Information from Configured Nodes

   Automatic network discovery discovers network interfaces and disks on both the nodes. It gathers information about the configured physical networks and

associated network adapters, and automatically creates the necessary selection lists for you to complete the HACMP network and adapter definitions. The output of Automatic Network Discovery is shown in Example 9-3.

Example 9-3 Shows the output of the Automatic Network Discovery

```
Discovering IP Network Connectivity

Retrieving data from available cluster nodes. This could take a few
minutes....
Discovered [9] interfaces
IP Network Discovery completed normally
Discovering Volume Group Configuration
Initializing..
Gathering cluster information, which may take a few minutes...
Processing...
Storing the following information in file
/usr/es/sbin/cluster/etc/config/clvg_config

kanaga:

Hdisk: hdisk0
PVID: 0009cddaea97bf61
VGname:rootvg
VGmajor:active
Conc-capable:Yes
VGactive:No
Quorum-required:Yes
Hdisk: hdisk1
PVID: 0009cdda43c9dfd5
VGname:rootvg
VGmajor:active
Conc-capable:Yes
VGactive:No
Quorum-required:Yes
Hdisk: hdisk2
PVID: 0009cdda43ca4d87
VGname:rootvg
VGmajor:active
Conc-capable:Yes
VGactive:No
Quorum-required:Yes
Hdisk: hdisk3
PVID: 0009cddaf2342b62
VGname:datavg
VGmajor:46
Conc-capable:No
VGactive:No
```

```
Quorum-required:Yes
FREEMAJORS:47...

atlantic:

Hdisk: hdisk0
PVID:  0009cdcaeb48d3a3
VGname:rootvg
VGmajor:active
Conc-capable:Yes
VGactive:No
Quorum-required:Yes
Hdisk: hdisk1
PVID:  0009cdcac26dbb7c
VGname:rootvg
VGmajor:active
Conc-capable:Yes
VGactive:No
Quorum-required:Yes
Hdisk: hdisk2
PVID:  0009cdcab5657239
VGname:rootvg
VGmajor:active
Conc-capable:Yes
VGactive:No
Quorum-required:Yes
Hdisk: hdisk3
PVID:  0009cddaf2342b62
VGname:datavg
VGmajor:46
Conc-capable:No
VGactive:No
Quorum-required:Yes  .
FREEMAJORS:47...

F1=Help                  F2=Refresh                F3=Cancel
Esc+6=Command            Esc+8=Image               Esc+9=Shell
Esc+0=Exit                /=Find                     n=Find Next
```

To define a network, perform the following via SMIT:

```
smit hacmp
```

Select Extended Configuration → Extended Topology Configuration → Configure HACMP Networks → Add a Network to the HACMP Cluster

Choose **ether** for ethernet networks.

Figure 9-7 on page 264 shows the configuration of networks - ethernet.

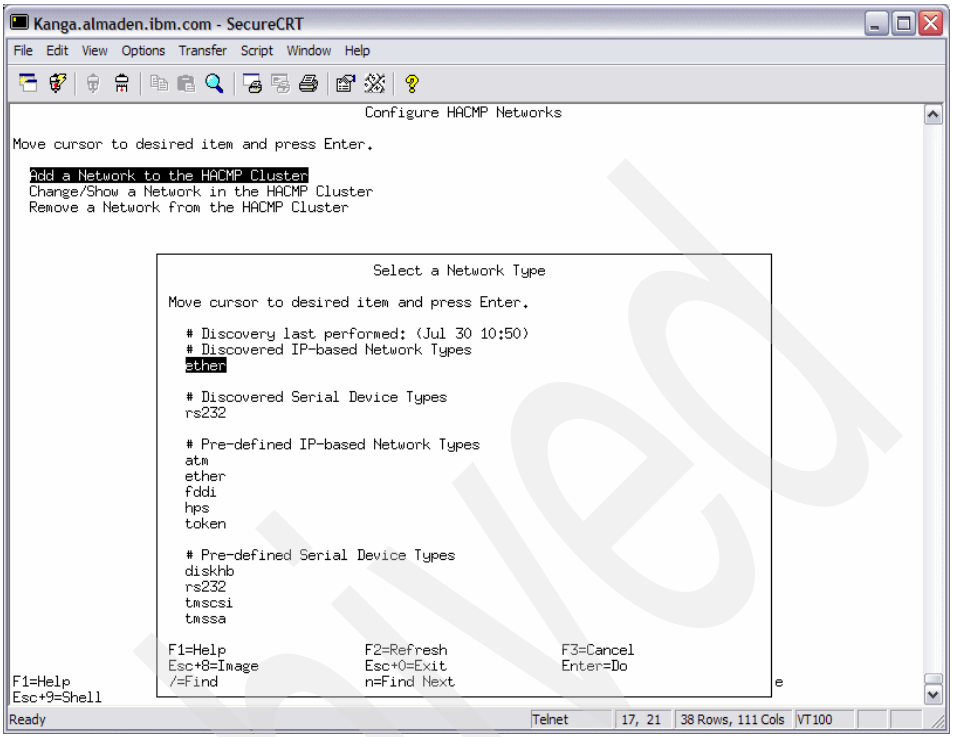

Figure 9-7 Configuration of HACMP networks: ethernet

The subnetwork mask for the first network is 255.255.255.0. In our lab environment, we have chosen to implement IPAT via IP replacement, where the service IP address replaces the boot IP address. Since IPAT via IP replacement was chosen, **Enable IP Address Takeover via IP Aliases** should be set to `NO`. Figure 9-8 on page 265 shows adding ethernet network to the HACMP cluster.

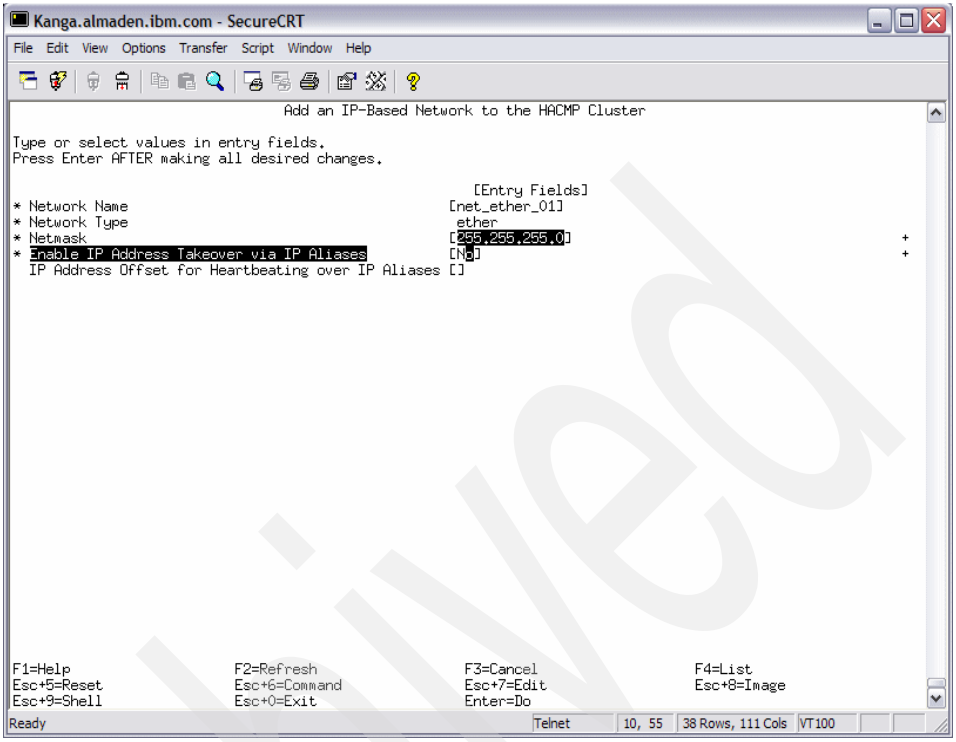

Figure 9-8 Addition of ether network to the HACMP cluster

Figure 9-9 on page 266 defines the standby ethernet network.

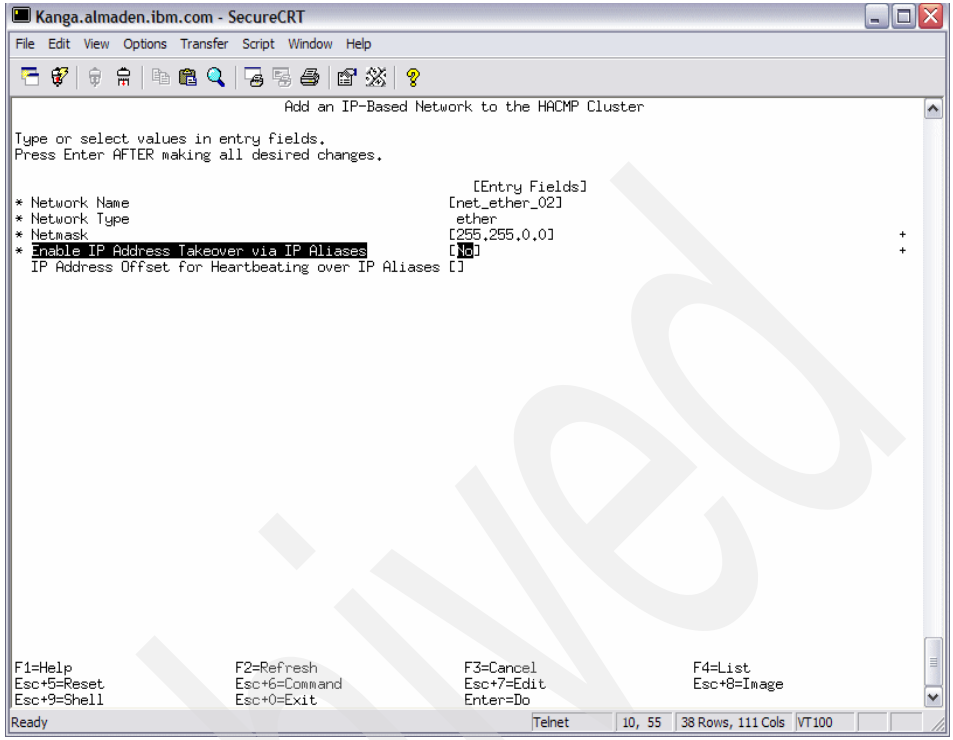

Figure 9-9 Addition of standby ether network

We defined a serial device for the `db2k` cluster for informational purpose only. We did not use the serial device for failover purposes. Figure 9-10 on page 267 shows how to define a serial network to the HACMP cluster.

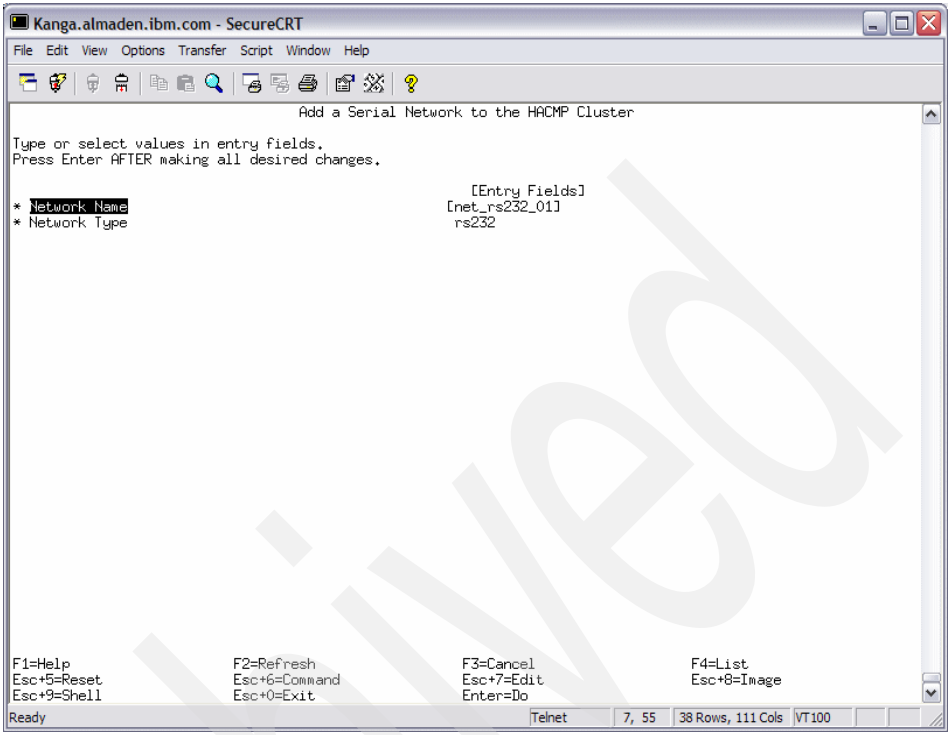

Figure 9-10 Addition of a serial network to the HACMP cluster

6. **Communication Interfaces/Devices:**

   We have already defined the networks to the HACMP cluster, so the next step is to assign the network interfaces to each network. Invoke HACMP configuration tool:

   smit hacmp

   Extended Configuration → Extended Topology Configuration → Configure HACMP Communication Interface/Devices → Add Communication Interfaces/Devices

   Choose **Add Discovered Communication Interface and Devices** since HACMP discovery ran and discovered all the communication interfaces and devices on both the nodes. Figure 9-11 on page 268 shows how to configure HACMP communication interfaces via discovered interfaces/devices.

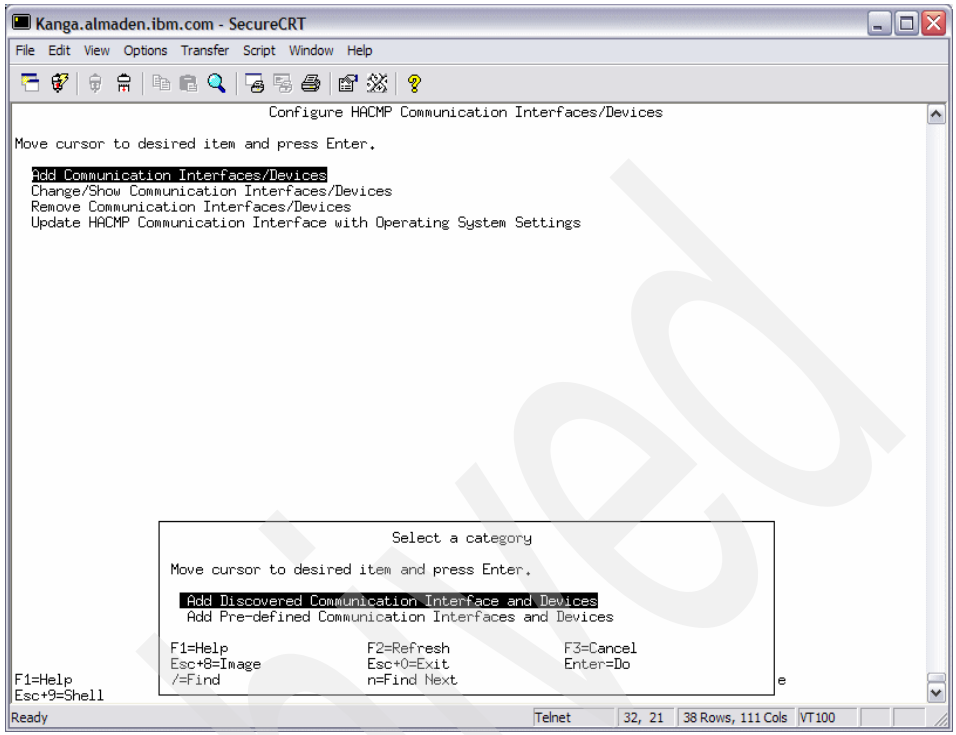

Figure 9-11 Configure HACMP Communication Interface via discovered interfaces/devices

Choose Communication Interfaces as shown in Figure 9-12 on page 269, and press Enter.

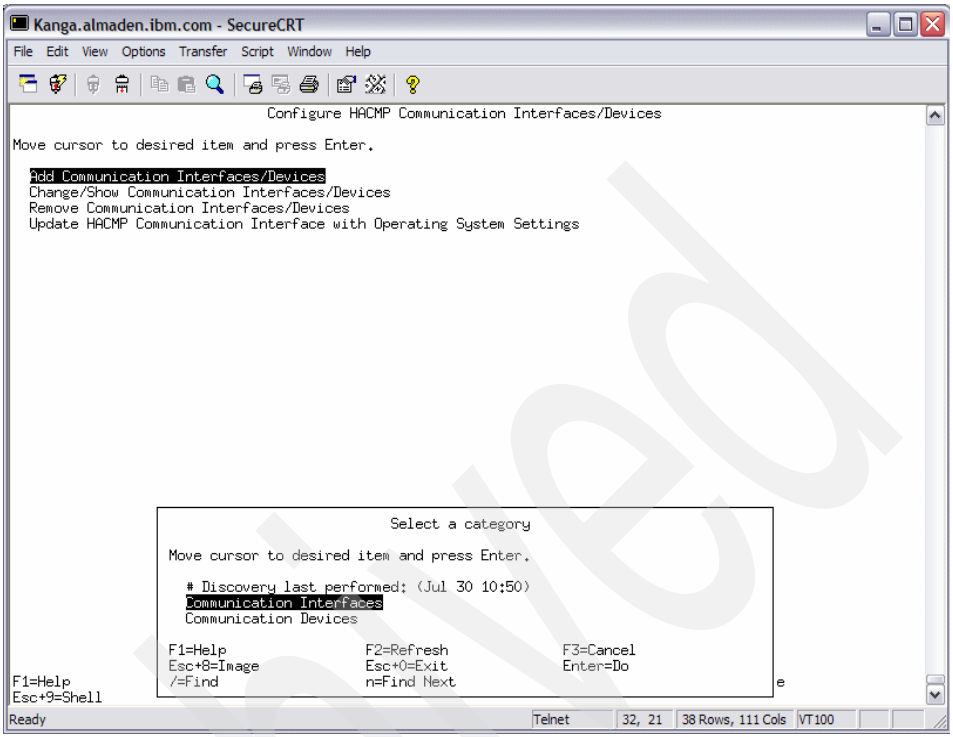

Figure 9-12 Selection of communication Interface

In Figure 9-13 on page 270, choose **net_ether_01** network created earlier and press Enter.

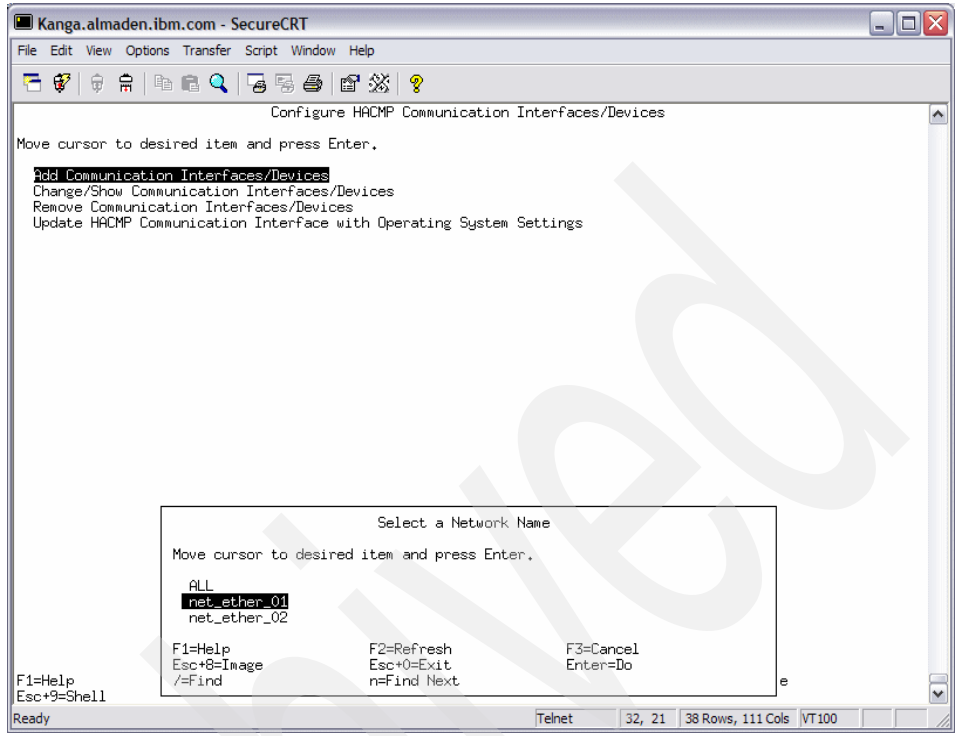

Figure 9-13 Choose the network for ether network interface

In Figure 9-14 on page 271, choose **en0** interface for net_ether_01 on both `KANAGA` and `ATLANTIC` and press Enter.

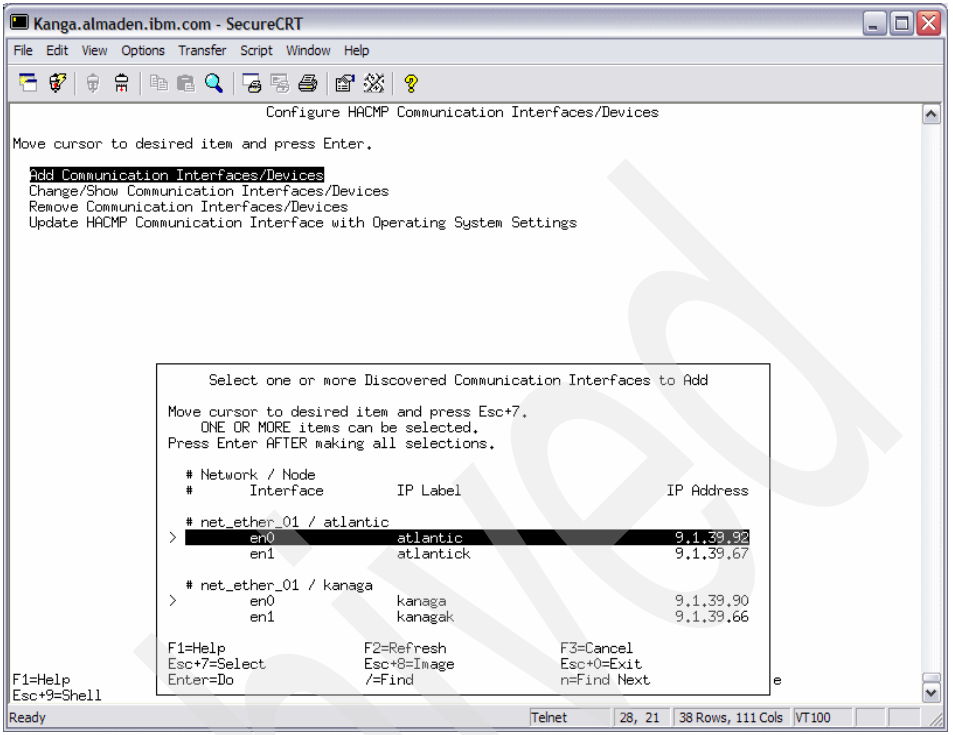

Figure 9-14 Choose en0 as the interface for net_ether_01 network

Add the standby communication interface *en2* in the same manner as *en0*. Choose **en2** interface for network net_ether_02 as shown in Figure 9-15 on page 272.

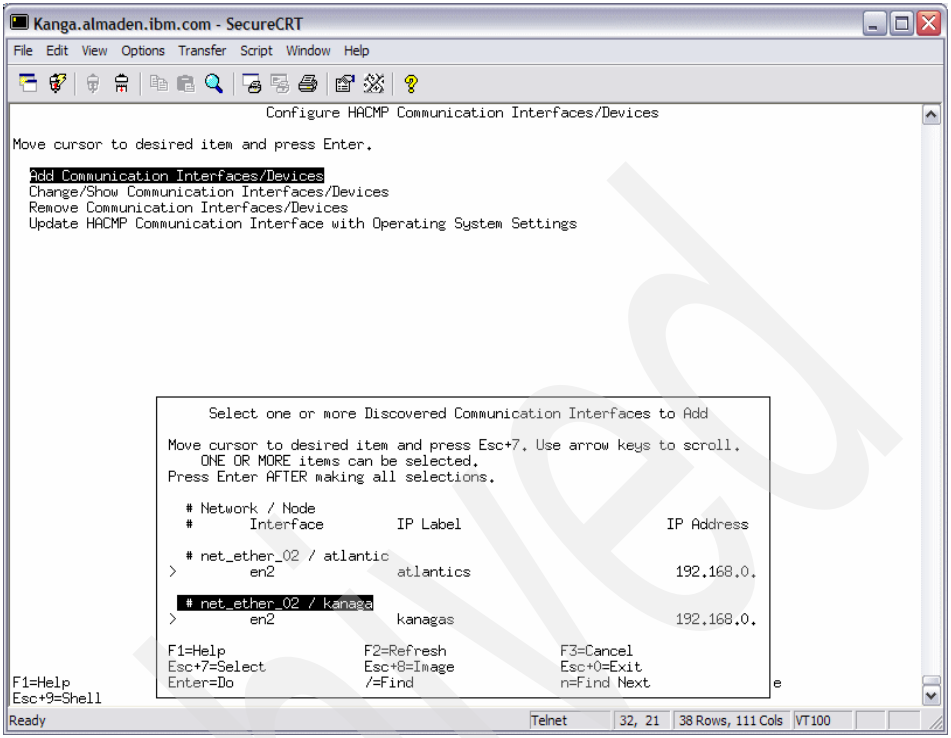

Figure 9-15 Configuring standby adapter

Choose **communication devices** instead of communication interfaces as shown in Figure 9-12 on page 269 to define serial devices. Figure 9-16 on page 273 shows the addition of a serial device to the HACMP cluster.

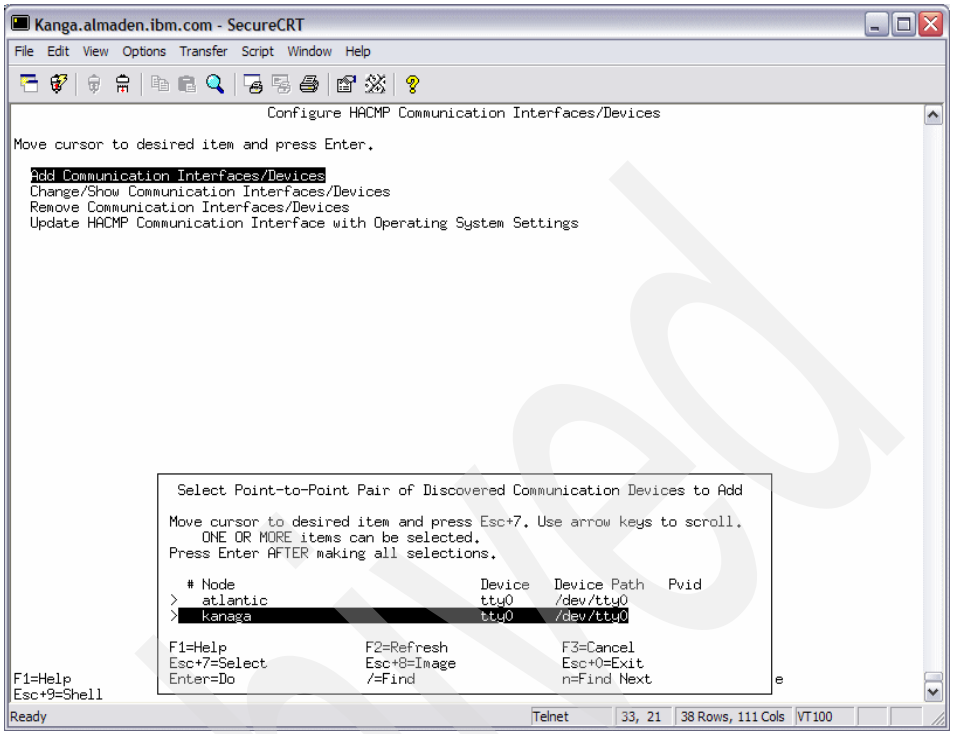

Figure 9-16 Addition of serial device

Verify and synchronize HACMP configuration before defining service IP address and creating the application servers.

7. **Define HACMP Service IP Address**

   Use HACMP utility in SMIT: `smit hacmp`.

   Select Extended Configuration → Extended Resource Configuration → HACMP Extended Resource Configuration → Configure HACMP Service IP Labels/Addresses → Add a Service IP Label/Address.

   Since we choose IPAT via IP replacement, we need to choose **Configurable on Multiple Nodes** as shown in Figure 9-17 on page 274.

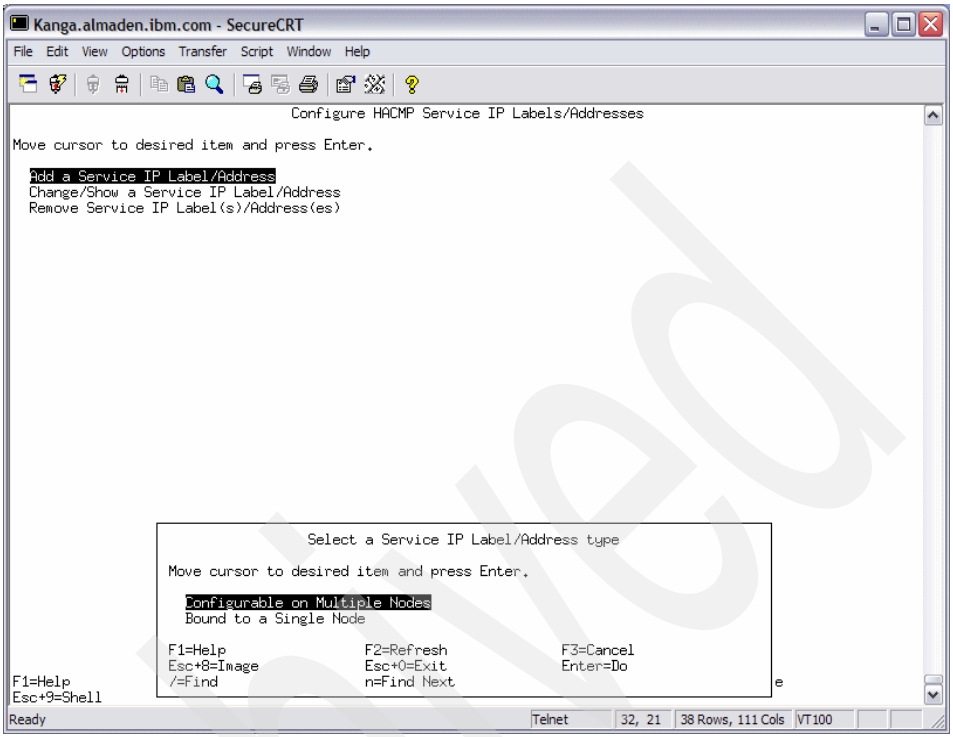

Figure 9-17 Service IP address configuration on Multiple Nodes

In Figure 9-18 on page 275, choose **net_ether_01** since we are defining Service IP Address on the boot network and press Enter.

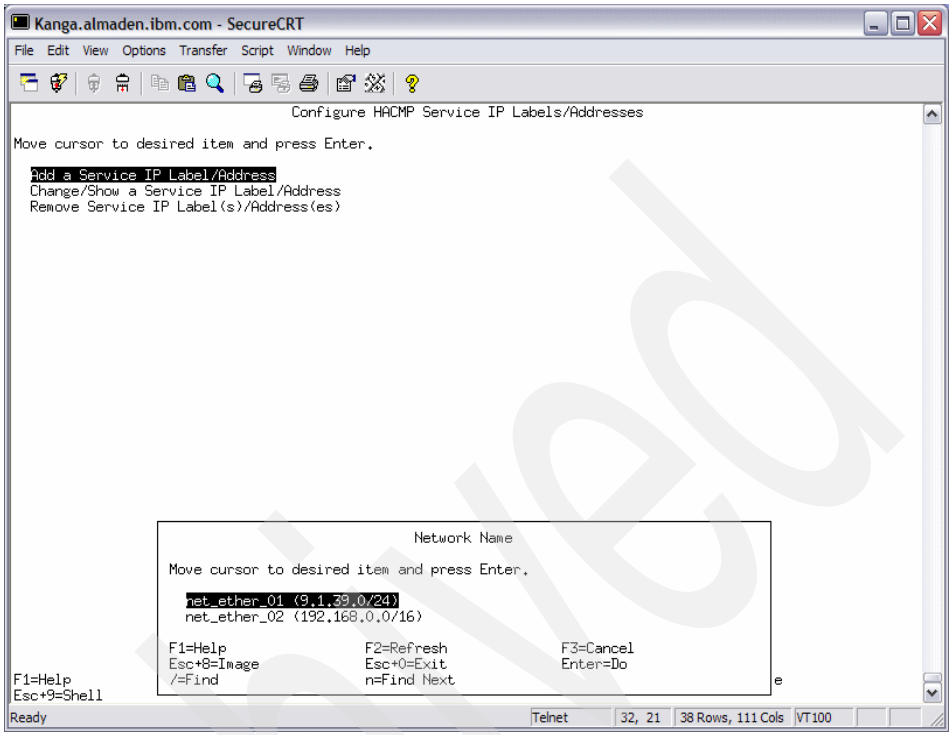

Figure 9-18 Service IP address on the primary network interface

In Figure 9-19 on page 276, choose the service IP Label/Address by pressing F4. It was defined in the `/etc/hosts` on both the nodes. In our lab environment, the service IP Address is `hakan`.

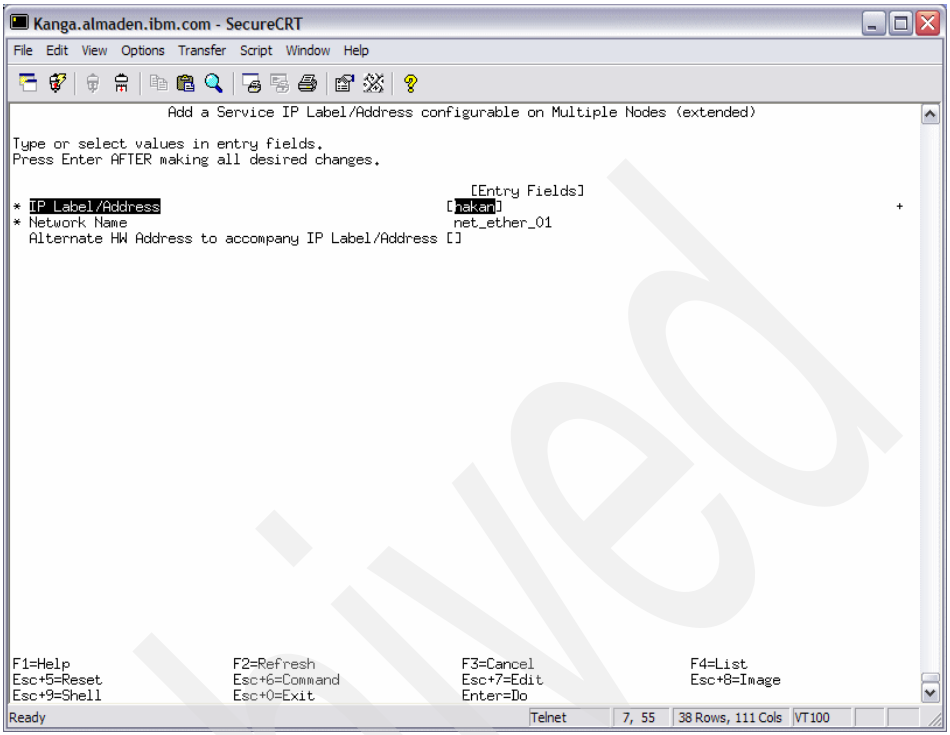

Figure 9-19 Defining Service IP address

8. **Application Server**

   Application Server is the application that needs high availability. In our case, the application is DB2 UDB database server. The application start script and stop script are custom scripts. These custom scripts call other scripts to start and stop DB2 instance. All the scripts are located in Appendix A.3, “HACMP scripts” on page 351.

   Use HACMP utility in SMIT to set up application server: `Smit hacmp`

   Select Extended Configuration → Extended Resource Configuration → HACMP Extended Resource Configuration → Configure HACMP Applications → Configure HACMP Application Servers → Add an Application Server

   Figure 9-20 on page 277 shows the creation of Application Server with start and stop scripts.

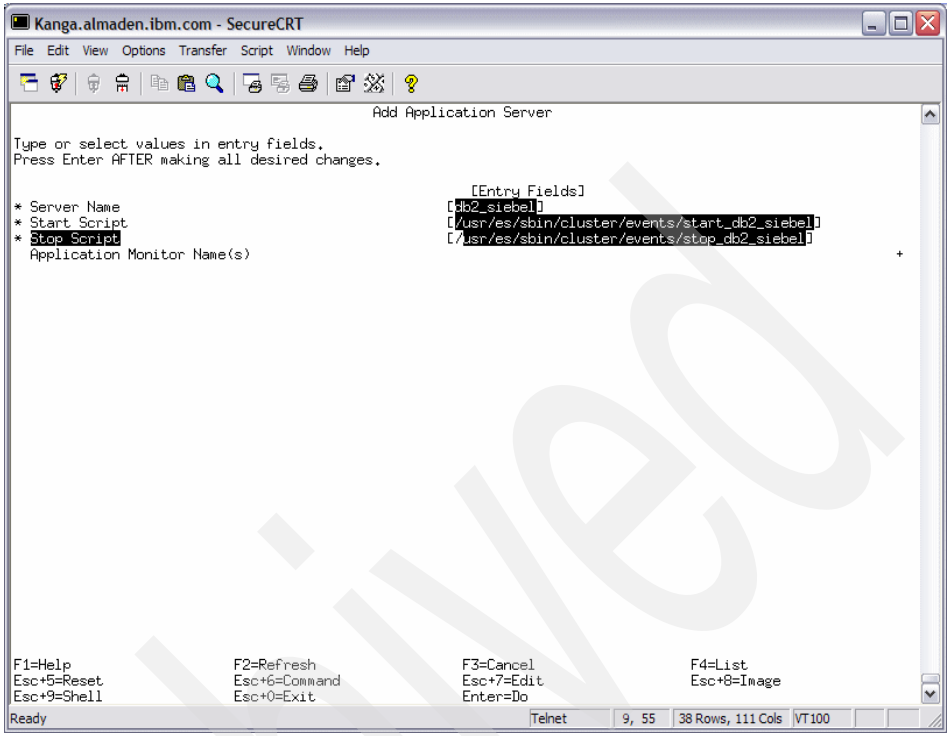

Figure 9-20 Add Application Server

9. **Define Resource Group**

   Use HACMP utility in SMIT: `Smit hacmp`

   Select Extended Configuration → Extended Resource Configuration → HACMP Extended Resource Group Configuration → Add a Resource Group

   Enter the Resource Group Name which is unique. Press F4 in Figure 9-21 on page 278 to get the list of nodes that will own or take over this resource group. It is important to enter the node with the highest priority first, followed by the nodes with lower priorities, in the desired order.

   Choose the Startup Policy, Failover Policy, and Fallback Policy based on your requirements.

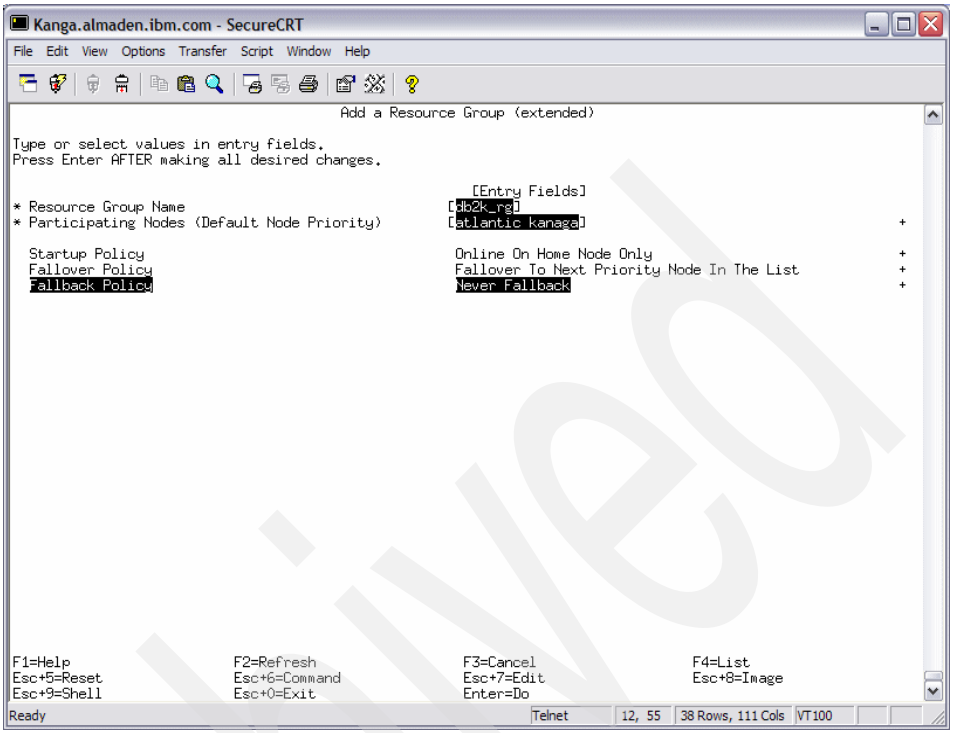

Figure 9-21 Add a resource group

Figure 9-23 on page 288 shows adding resources to the newly created resource group `db2k_rg`.

`smit hacmp`

Select Extended Configuration → Extended Resource Configuration → HACMP Extended Resource Group Configuration → Change/Show Resources and Attributes for a Resource Group

For Service IP address, press F4 and choose the Service IP Address `hakan` that we have defined. For Application Servers, press F4 and choose the application server `db2_siebel` that we have defined. For Volume Groups, press F4 and choose the shared volume group `datavg`.

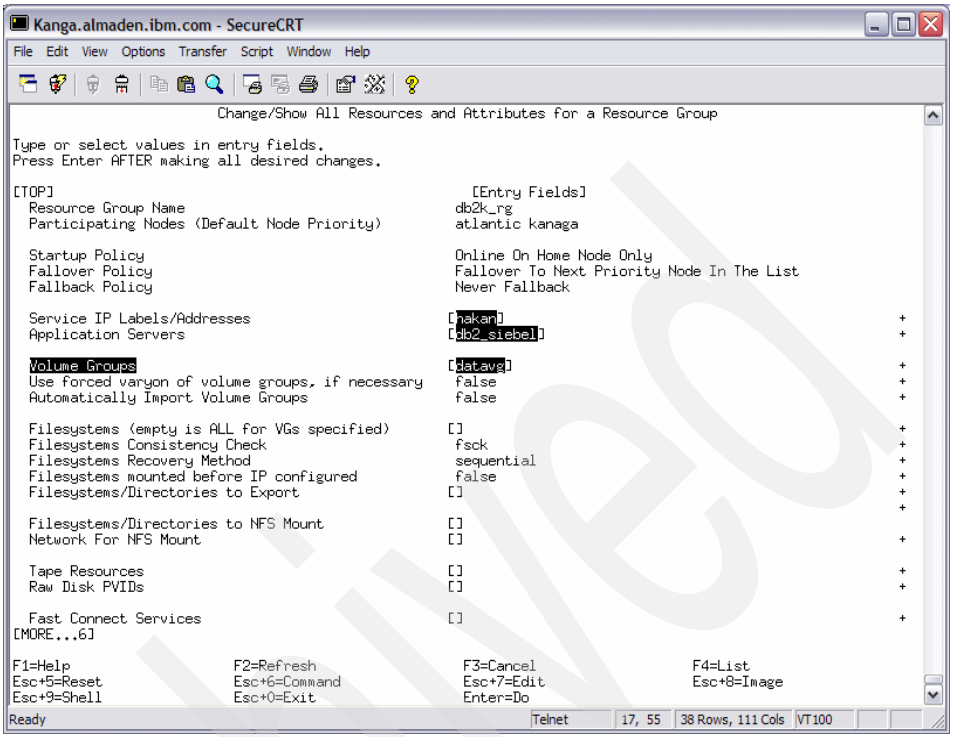

Figure 9-22 Adding resources to a resource group

9.2.4 Setting up idle standby for DB2 UDB server

To set up an idle standby, follow these steps:

1. Connect and define the same storage area to be available to both server 1 (primary) and server 2 (standby).

   After the drives were configured on both servers, we create the volume group `datavg` on `hdisk3` on `KANAGA`, so that we can use it as a shared volume group for the database between the two servers. Example 9-4 shows the disk information on `KANAGA`.

   Example 9-4 KANAGA disk information

```
lsdev -Cc disk
hdisk0 Available 1S-08-00-8,0  16 Bit LVD SCSI Disk Drive
hdisk1 Available 1S-08-00-9,0  16 Bit LVD SCSI Disk Drive
hdisk2 Available 1S-08-00-10,0 16 Bit LVD SCSI Disk Drive
hdisk3 Available 1Z-08-01      1742-900 (900) Disk Array Device

lspv
```

```
hdisk0          0009cddaea97bf61 rootvg          active
hdisk1          0009cdda43c9dfd5 rootvg          active
hdisk2          0009cdda43ca4d87 rootvg          active
hdisk3          0009cddaf2342b62 datavg          active
```

We did not have to create the datavg on ATLANTIC because it is a shared volume group. Following are steps to configure datavg on ATLANTIC.

- **varyoffvg datavg** (on KANAGA)
- **exportvg datavg** (on KANAGA)
- **importvg -L datavg hdisk3** (on ATLANTIC)
- **varyonvg datavg** (on ATLANTIC)

Example 9-5 shows the disk information on ATLANTIC. hdisk3 with datavg is listed under ATLANTIC physical disk list.

Example 9-5 Disk information on ATLANTIC

```
lsdev -Cc disk
hdisk0 Available 1S-08-00-8,0  16 Bit LVD SCSI Disk Drive
hdisk1 Available 1S-08-00-9,0  16 Bit LVD SCSI Disk Drive
hdisk2 Available 1S-08-00-10,0 16 Bit LVD SCSI Disk Drive
hdisk3 Available 1Z-08-01      1742-900 (900) Disk Array Device
lspv
hdisk0          0009cdcaeb48d3a3 rootvg          active
hdisk1          0009cdcac26dbb7c rootvg          active
hdisk2          0009cdcab5657239 rootvg          active
hdisk3          0009cddaf2342b62 datavg          active
```

During normal operation in our idle standby, hdisk3 will only be defined on the primary server KANAGA. We vary off the volume group on ATLANTIC and vary on KANAGA to proceed with the next steps.

2. Create logical volumes and file systems.

   Table 3-5 on page 35 shows the file systems and logical volumes on the shared storage area (hdisk3) we created. For commands to create volume group (vg), logical volume (lv), and file systems (fs), refer to Chapter 4, “Installation and configuration” on page 47. All the file systems we created are mounted with options **rbrw**. All the file systems are created with auto mount turned off.

   We then unmounted all the file systems residing in datavg and varied off and exported datavg on KANAGA. We then imported and varied on datavg on ATLANTIC. HACMP will mount these file systems at failover time.

   All file systems and logical volumes have permissions set to 775, with *owner.group* name set to aixdba.aixdbag. User IDs ucsdb2 and sebldb2 are part of aixdbag group.

Note: The time taken for failover will increase with the number of file systems and raw devices that have to fail over from primary to standby server. For smaller servers, try to minimize the number of devices. You may be able to get away with just DMS table spaces on file systems. For large scale databases, where performance is key, you would have several raw devices and therefore longer failover time.

3. Creation of user IDs, groups, and DB2 instances

   Ensure that the user IDs and groups created on both servers are identical, including the ID numbers. This has to match perfectly otherwise you will have problems during failover. For the list of User IDs and groups with their respective numbers, refer to Chapter 4, “Installation and configuration” on page 47.

   In our setup, we have two DB2 instances, `ucsdb2` and `sebldb2`. `ucsdb2` is a 64-bit instance and `sebldb2` is a 32-bit instance. Even though the database is on 64-bit instance, we need the 32-bit instance, since Siebel currently cannot directly communicate with a 64-bit instance.

   Section 3.6, “Users and groups” on page 39, shows a list of user IDs and their usage.

   We used the following commands to create the DB2 instances on `KANAGA` from `/usr/opt/db2_08_01/instance`.

   To create the 64-bit instance:

   ```
   ./db2icrt -a SERVER_ENCRYPT -p 8100 -s ese -w 64 -u fencdb2 ucsdb2
   ```

   To create the 32-bit instance:

   ```
   ./db2icrt -a SERVER_ENCRYPT -p 8200 -s ese -w 32 -u sebldb2 sebldb2
   ```

   To set up DB2 instances to failover from one system to another, the instances would have to be created on both servers. But the instance home directory should be on shared volume group. Once the instances are created on `KANAGA`, we stopped the instances on `KANAGA` and deleted the `/home/sebldb2/sqllib and /home/ucsdb2/sqllib` to allow instance creation on `ATLANTIC`.

   We then varied off `datavg` on `KANAGA` and varied it on `ATLANTIC`.

   Using the same instance creation command from above to create the instances on `ATLANTIC`, we then made sure that we can start and stop DB2 instances on `ATLANTIC` (standby) server.

   Once that is done, we varied off the `datavg` on `ATLANTIC` and varied it on `KANAGA`.

In a HACMP setup, depending on configuration, you can set up in such a way that, if the primary server comes back online, DB2 is shut down on failover (standby)) server and then all resources are moved to the primary server and DB2 is brought back online. In our lab setup, we have configured the HACMP not to fall back to primary (`KANAGA`) server until we are ready. There are two reasons for it:

- If the primary server has failed, you want to be able to bring it back online and do the problem determination before we fail the database back to the primary server.
- You may want to wait until end-of-business before the failback takes place, because this will cause additional prime time outage to the customer.

The `datavg` is varied off on `KANAGA` (primary server) so that we can do the configuration for HACMP.

Note: From here on, any changes made to primary server also need to be applied to standby server. Such as any user IDs created, altered, dropped (security vulnerability), applying DB2 patches, operating system patches, and changing system configuration parameters like `MAXUPROC` and so on.

HACMP parameters for DB2 UDB server

We are almost ready to test the idle standby setup for DB2 UDB server with HACMP. Before we do that, we discuss several items that will affect failover.

- Heartbeat monitor:

  The heartbeat rate should be set to at least 8 seconds. In theory, you can check a heartbeat every second, but in reality you want it to be higher like 2 seconds. When you multiply it by the heartbeat cycle time (4 seconds), you get 8 seconds. we suggest setting it to a value of 8 or higher.

  For heavily used (overloaded) OLTP applications, the heartbeat rate would be higher. The same applies for heavy transaction-based databases such as OLAP. Also check to make sure that the network is not overloaded otherwise the heartbeat packets can be lost.

- DB2 UDB database configuration parameters which affect failover performance:

  - `CHNGPGS_THRESH`:
    Percentage of changed/dirty pages in the buffer pool, at which the asynchronous page cleaners are triggered to clean the dirty pages out of buffer pool. A dirty page is one that has been changed in memory and has not yet been written to disk. Lowering this number means that there are smaller numbers of changed pages in buffer pool. This improves

performance and helps during HA failover as it reduces the time for crash recovery. We recommend set this value around 20 - 30.

– `NUM_IO_CLEANERS`:
This parameter specifies the maximum number of asynchronous page cleaners triggered when `CHNGPGS_THRESH` has been met. This value should at least be set to the number of CPUs on the system. Increasing this parameter may lead to lowering crash recovery time during soft crashes.

– `SOFTMAX`:
Setting `SOFTMAX` to a lower value will write the log control file to the disk more often. This will reduce the amount of time taken during a crash recovery. Refer to the IBM DB2 UDB document: *Administration Guide: Performance V8*, SC09-4821-01, for more information. We suggest having this value set at 20-30.

Synchronize and verify

Start the HACMP cluster on `KANAGA` and then on `ATLANTIC`. To start HACMP clusters:

`smit hacmp`

System management → Manage HACMP Services → Start Cluster Services

HACMP logs are located in `/tmp/hacmp.out` and in `/var/ha/log directory`.

Once the cluster processes are up on both the servers, you can test the status of the cluster.

`Smit hacmp`

Problem Determination Tools → View Current State

Example 9-6 shows a current state of the HACMP cluster.

Example 9-6 Shows the current HACMP state

```
Obtaining information via SNMP from Node: kanaga...


_____________________________________________________________________
Cluster Name: db2k
Cluster State: UP
Cluster Substate: STABLE
_____________________________________________________________________


Node Name: atlantic             State: DOWN

Network Name: net_ether_01       State: DOWN
```

```
 Address: 9.1.39.92       Label: atlantic           State: DOWN

 Network Name: net_ether_02       State: DOWN

 Address: 192.168.0.5     Label: atlantics          State: DOWN

 Network Name: net_rs232_01       State: DOWN


Node Name: kanaga               State: UP

Network Name: net_ether_01       State: UP

 Address: 9.1.39.156      Label: hakan              State: UP
 Address: 9.1.39.90       Label: kanaga             State: DOWN


 Network Name: net_ether_02       State: DOWN

Address: 192.168.0.4     Label: kanagas            State: UP

 Network Name: net_rs232_01       State: DOWN



Cluster Name: db2k

Resource Group Name: db2k_rg
Startup Policy: Online Using Distribution Policy
Failover Policy: Failover To Next Priority Node In The List
Fallback Policy: Never Fallback
Site Policy: ignore
Node            State
--------------- ---------------
kanaga          ONLINE
atlantic        OFFLINE
```

To test if the HACMP fails over, stop the cluster on the primary node or move the resource. To stop the cluster :

smit hacmp

System management → Manage HACMP Services → Stop Cluster Services

To move the resource db2k_rg to the secondary node:

```
smit hacmp
```

System Management → HACMP Resource Group and Application Management → Move a Resource Group to Another Node

Log into the secondary node and check to see if the file systems on the shared volume group are mounted and if DB2 instance has started.

For more information on HACMP, refer to the following documentation:

- HACMP library is available at:

  http://www.ibm.com/servers/eserver/pseries/library/hacmp_docs.html

- *IBM HACMP for AIX V5.X Certification Guide* Redbook, SG24-6375, can be accessed at:

  http://www.redbooks.ibm.com/abstracts/sg246375.html

9.2.5 Setting up active standby for DB2 UDB server

The procedure of setting up active standby servers is almost identical as the procedure of setting up idle standby. For an active standby, additional logical volumes are needed to do local functions when the primary server is up and running. Following are the additional steps for setting up active standby servers, refer to 9.2.4, "Setting up idle standby for DB2 UDB server" on page 279 for details in each steps:

1. Install and configure HACMP on both servers.
2. Connect and define the same storage area to be available to both server 1 and server 2.

   We moved contents of `hdisk2` to `hdisk0` and `hdisk1` on `ATLANTIC` and created a new volume group called `localvg` on `ATLANTIC`, so that we can host a local database.

   Table 9-8 lists the file systems and logical volumes (lv) we created on `localvg`:

Table 9-8 Additional logical columns needed on a active standby

lv name	Size	Mount point	Usage
userdb2lv	2 GB	/home/userdb2	userdb2 home directory
vpath1cont001lv	1 GB	-	raw device for DMS table space
vpath1cont002lv	1 GB	-	raw device for DMS table space

All file systems and logical volumes have permissions set to 775, with `owner.group` name set to `aixdba.aixdbag`. User IDs `userdb2`, `ucsdb2`, and `sebldb2` are part of `aixdbag` group.

3. Creation of user IDs, groups, and DB2 instances

   Create a group `userdb2g` on `ATLANTIC`.

   Create a userid called `userdb2` on `ATLANTIC`.

   To set up DB2 instances to failover from one system to another, the instances would have to be created on both servers. We create DB2 instances on `KANAGA` and `ATLANTIC`:

   – `ucsdb2`: 64-bit instance
   – `sebldb2`: 32-bit instance

   We also created the instance for `userdb2` using:

   ```
   /db2icrt -a SERVER_ENCRYPT -p 8200 -s ese -w 32 -u userdb2 userdb2
   ```

   Once that is done, we varied off the datavg on `ATLANTIC` and varied it on `KANAGA`.

   Note: We created two instances (`ucsdb2` and `sebldb2`) on `KANAGA` and three (`ucsdb2, sebldb2,` and `userdb2`) on `ATLANTIC`. At any given time, `ucsdb2` and `sebldb2` will only be active on one machine.

Note: From here on, any changes made to primary server also need to be applied to standby server. Such as IDs created, altered, dropped (security) vulnerability), DB2 patches, operating system patches, and system configuration parameters such as maxuproc and so on. Depending on the situation, changes executed on failover server may not be applied to primary server, in that there may be changes applied to `userdb2` id on `ATLANTIC`, that we will not carry to `KANAGA`.

The Synchronizing and verifying procedure for active standby setup are the same as idle standby.

9.2.6 DB2 UDB clustering on Windows

The high availability clustering solution on Windows is called Microsoft Cluster Services (MSCS). The concept of cluster involves taking two or more computers and organizing them to work together to provide high availability. If one server stops functioning, a process called failover automatically shifts the work to another server in the cluster. Clustering is a way to ensure system availability in

the event of component failure, such as a lost CPU, failed power supply, and so on.

In the Windows environment, the HA design solutions idle standby, active standby, and mutual takeover are also applicable. In this section, we provide the detailed implementation steps for idle standby.

Setup idle standby

In an idle standby, there is an extra server that will take over the resources from the primary database server. In idle standby, there would be no degradation in performance after failover.

Lab environment

In our lab environment, we have two servers called `WISLA` and `LOCHNESS`. We considered `WISLA` to be the primary cluster server and `LOCHNESS` the secondary server. At this point, it is assumed that both machines are connected to the shared disk array.

A static IP address and network name are required for the cluster IP address and cluster network name. Once the cluster is brought online, all clients will access the cluster by the network name or IP address. A heartbeat/private network must be installed between the two machines. Create a partition in the shared disk for the quorum disk. The quorum disk is used to store cluster configuration database, checkpoints, and log files that help manage the cluster. The quorum disk partition can be as big as 500 MB.

The following is the cluster information.

- Cluster (domain) name = `db2cluster`
- Domain userid/password = `db2admin`/********
- Cluster IP address = 9.1.39.180
- Public IP address for `WISLA` = 9.1.39.167
- Private IP Address for `WISLA` = 10.0.0.101
- Public IP address for `LOCHNESS` = 9.1.39.169
- Private IP Address for `LOCHNESS` = 10.0.0.100

Cluster installation and configuration

Before installing and configuring the cluster, shut down the secondary server (`LOCHNESS`). Insert the Microsoft Windows 2000 Advanced Server CD into `WISLA` to add the cluster component to the server.

Select **Cluster Service** component as shown in Figure 9-23 on page 288 and click **Next**.

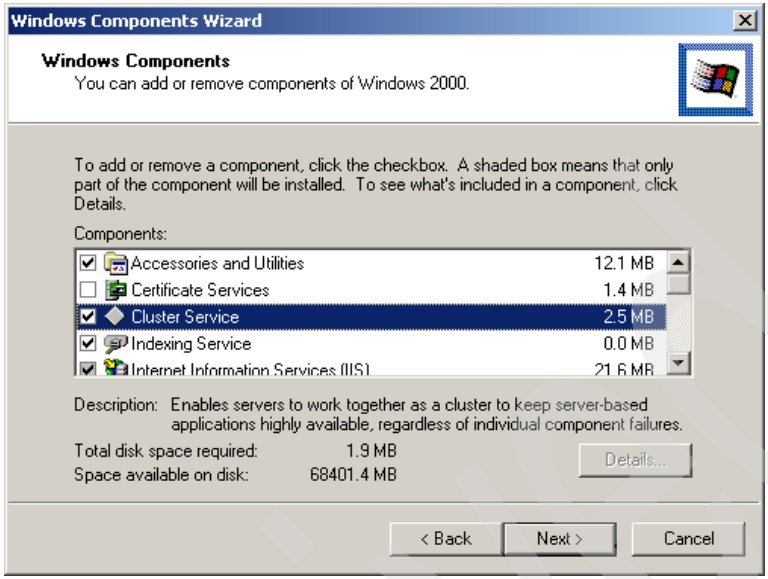

Figure 9-23 Cluster Service installation

Once the required cluster components are installed, the cluster configuration wizard will help you configure the cluster. Click **Next** on the Welcome window of Cluster Service Configuration Wizard.

Click **I Understand** button in the Hardware configuration window and click **Next**.

`WISLA` is the first server in the new cluster. Select **The first node in the cluster** button and click **Next** as shown in Figure 9-24 on page 289.

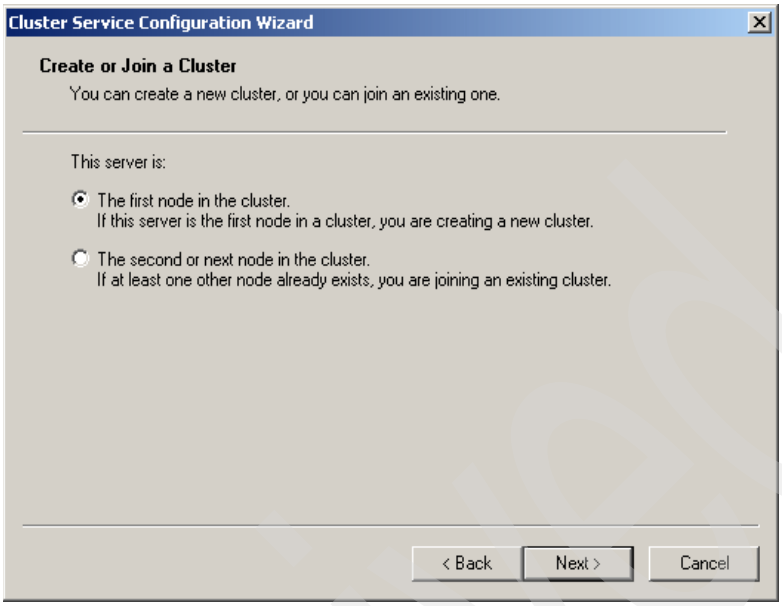

Figure 9-24 Creating a new cluster.

The name of the cluster must be specified as shown in Figure 9-25 and click **Next** to create a new cluster. We used the name `DB2CLUSTER`.

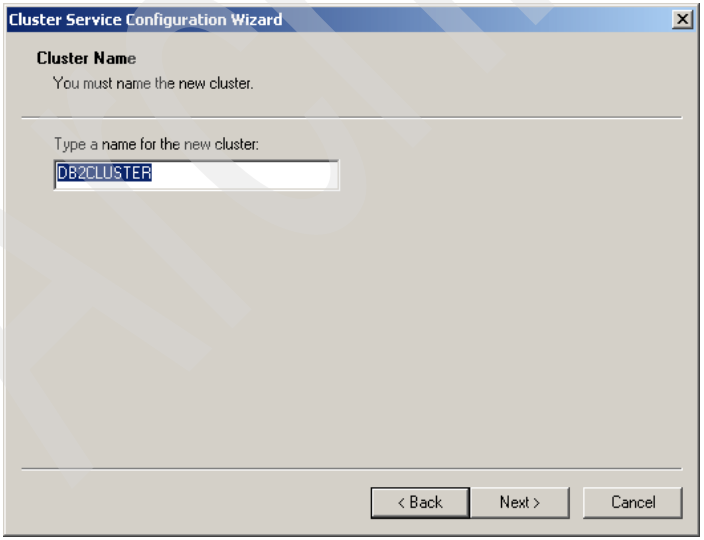

Figure 9-25 The cluster name

Add or remove the disks that are to be managed by the cluster as shown in Figure 9-26.

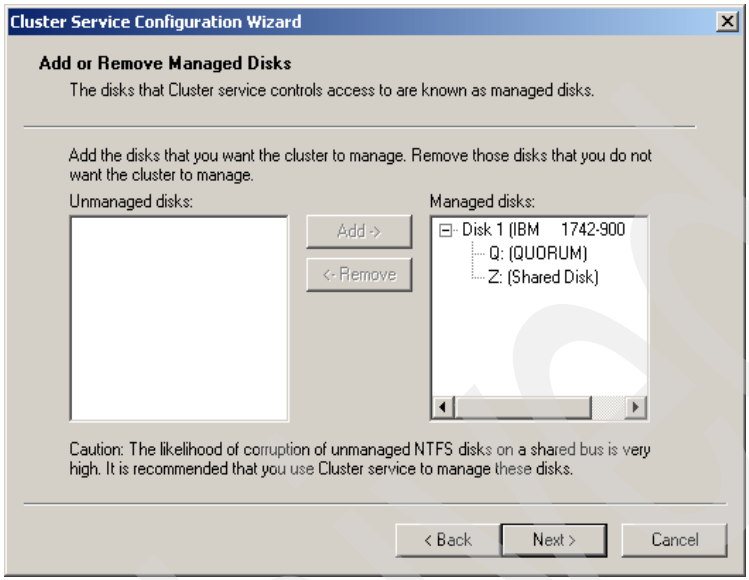

Figure 9-26 Add or remove disks to be managed by the cluster

Specify the location of the quorum disk that has already been created and assigned as Q: drive in Figure 9-27.

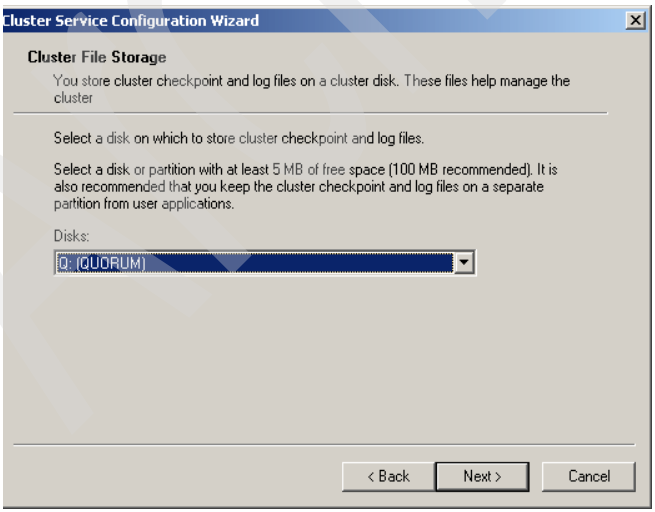

Figure 9-27 Location of the quorum disk

Define the public network as shown in Figure 9-28. Select **Enable this network for cluster use** and click **Next**. You can choose this public network for all communications. We chose the public network just for client access, since we have a separate private network between the two servers.

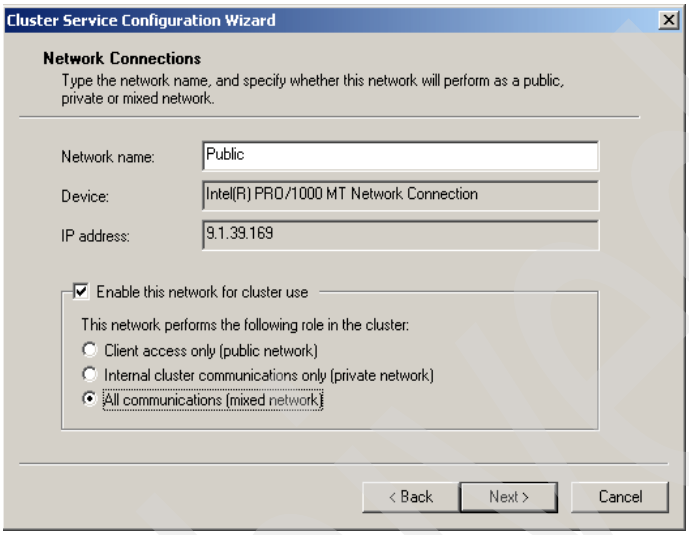

Figure 9-28 Define public network

In Cluster IP Address window (Figure 9-29), specify the cluster IP address and subnet mask.

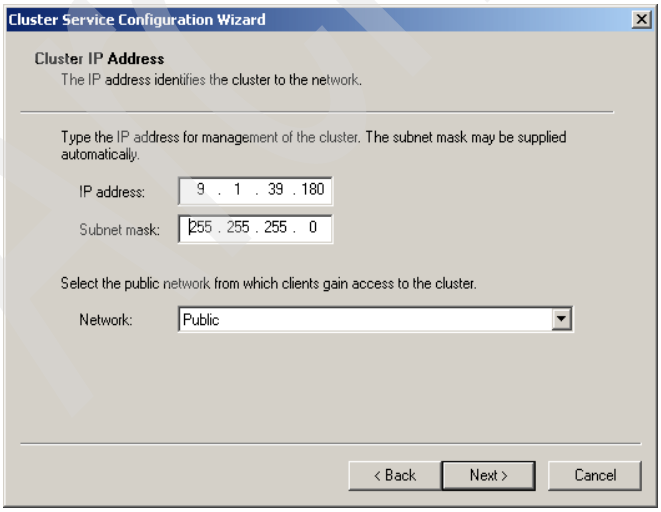

Figure 9-29 Specify cluster IP address and subnet mask

This completes the installation and configuration of the cluster service on the primary cluster server `Wisla`. Click **Finish** to complete the configuration. The cluster service will be started automatically after completion of the installation.

Start the Cluster Administration tool on `Wisla` to manage the cluster, by using **Start** → **Control Panel** → **Administrative Tools** → **Cluster Administration**. Figure 9-30 shows the *Cluster Administration* window with all cluster resources online.

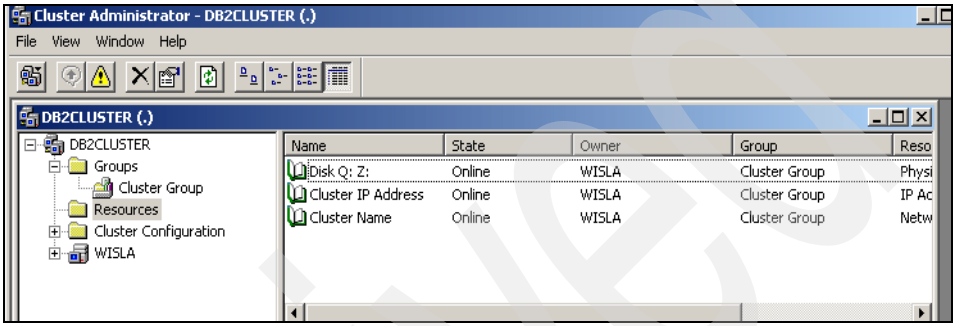

Figure 9-30 Cluster Administration window on Wisla

Cluster Installation and configuration on the secondary server

Start the secondary server. Follow the same process of installing and configuring the primary server to set up the secondary server, `LOCHNESS`.

Lochness is the second server in the new cluster. In the *Create or Join a Cluster* window, select **Second or next node in the cluster** to add `LOCHNESS` to the cluster as shown in Figure 9-31 on page 293.

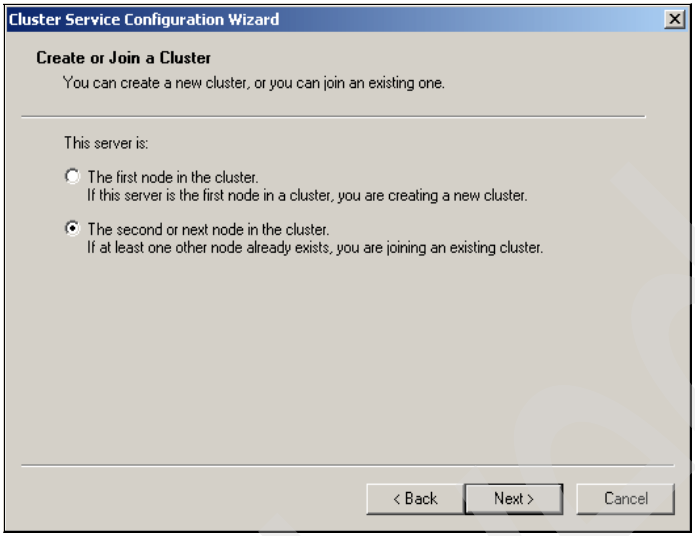

Figure 9-31 Adding the second node to the cluster

Provide the name of the cluster as `DB2CLUSTER` to add `LOCHNESS` to the cluster as shown in Figure 9-32:

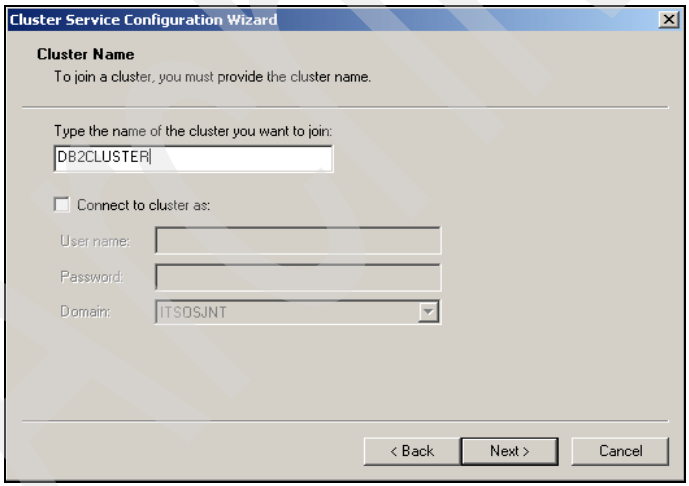

Figure 9-32 Add the name of the cluster to join the cluster

Provide the cluster user name and password in *Select an Account* window.

Select **Finish** when cluster installation is complete. The cluster service is automatically started.

All of the cluster resources, Cluster IP address, Cluster Name and Disks are currently owned by `WISLA` the primary cluster server. In the *Cluster Administration* window, expand DB2CLUSTER, expand Groups and then right-click on the Cluster Group and select **Move Group** as shown in Figure 9-33.

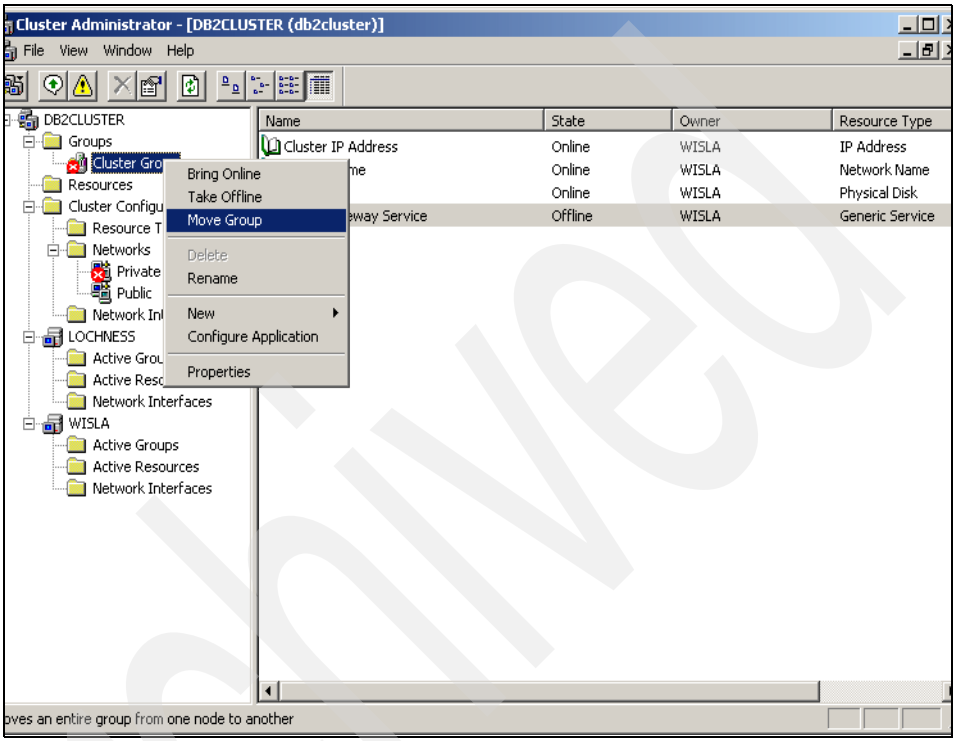

Figure 9-33 Move resource to other servers

All the cluster resource should move to `LOCHNESS` and should come online. This completes the cluster installation and configuration. The next sections describe creation of cluster resources for DB2 UDB and Siebel Gateway Name Server, Siebel File System, and Siebel Server.

DB2 UDB configuration on Microsoft cluster

Install DB2 UDB software on both servers. Ensure that you installed DB2 UDB software on the local drive or on a non-shared drive.

Use the following steps to configure DB2 UDB failover.

1. Create DB2 instance on the primary system.

   During the install, a default db2 instance named `DB2` was created. We prefer instance named `db2admin` so we dropped the instance `DB2` on both servers

and created a new one using the following command on the primary (`WISLA`) server only.

Follow the directions in Chapter 4, “Installation and configuration” on page 47 for configuring DB2 instances.

> **Note:** Do *not* create instance on the failover box.

2. Alter the service *DB2 - DB2ADMIN - 0* to start manually on the primary (`WISLA`) server using the Windows services as shown in Figure 9-34.

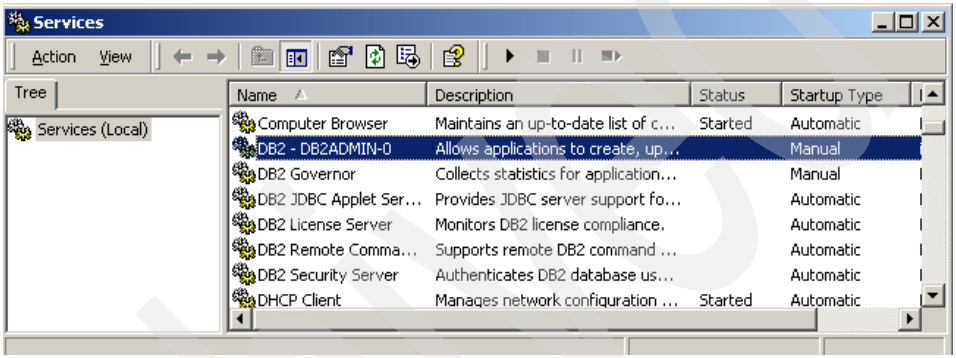

Figure 9-34 Windows services dialog box

3. If the DB2 Instance is running, stop it using **`db2stop`** command.
4. To create the resource type on primary server (`WISLA`), run the following command:

```
db2wolfi i
```

if you get a message “`Error : 183`”, it means that the resource type has already been created.

When you click on resource type on *Cluster Administration* window, a window similar to Figure 9-35 on page 296 will be displayed.

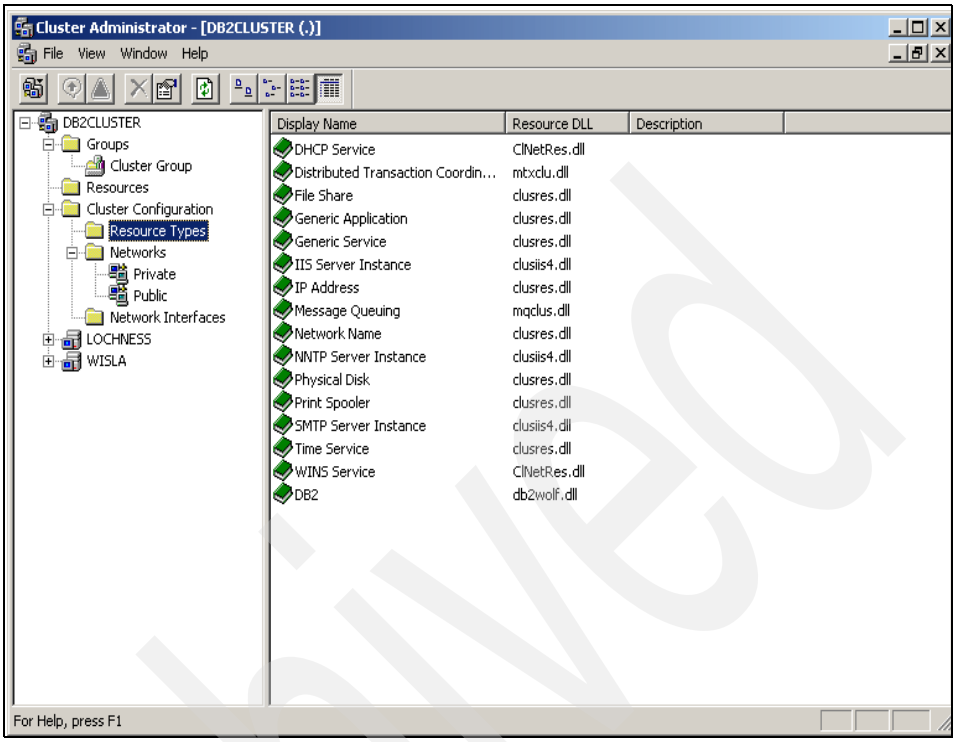

Figure 9-35 Cluster Administration window showing resource types

5. To transform the `db2admin` instance on primary server (`WISLA`) to a clustered instance, create a directory `db2profpath` on the shared drive (Z:) for the instance profile path and run the following command.

```
C:\>db2iclus migrate /i:db2admin /c:db2cluster /m:wisla /p:z:\db2profpath
DBI1912I The DB2 Cluster command was successful.
```

   Running the above command changed the DB2 instance on primary server (`WISLA`) in to a clustered instance and added `WISLA` as the first server in the clustered instance.

6. To add additional servers to the clustered instance, run the following command. Make sure the new server can access the location of the DB2 profile path.

   We ran the following command to add `LOCHNESS`:

```
C:\>db2iclus add /i:db2admin /c:db2cluster /m:lochness
/u:itsosjnt\db2admin,itso13sj
DBI1912I The DB2 Cluster command was successful.
```

   Repeat this command for each additional server you want to add to the cluster.

The Cluster Administration window should look similar to the Figure 9-36.

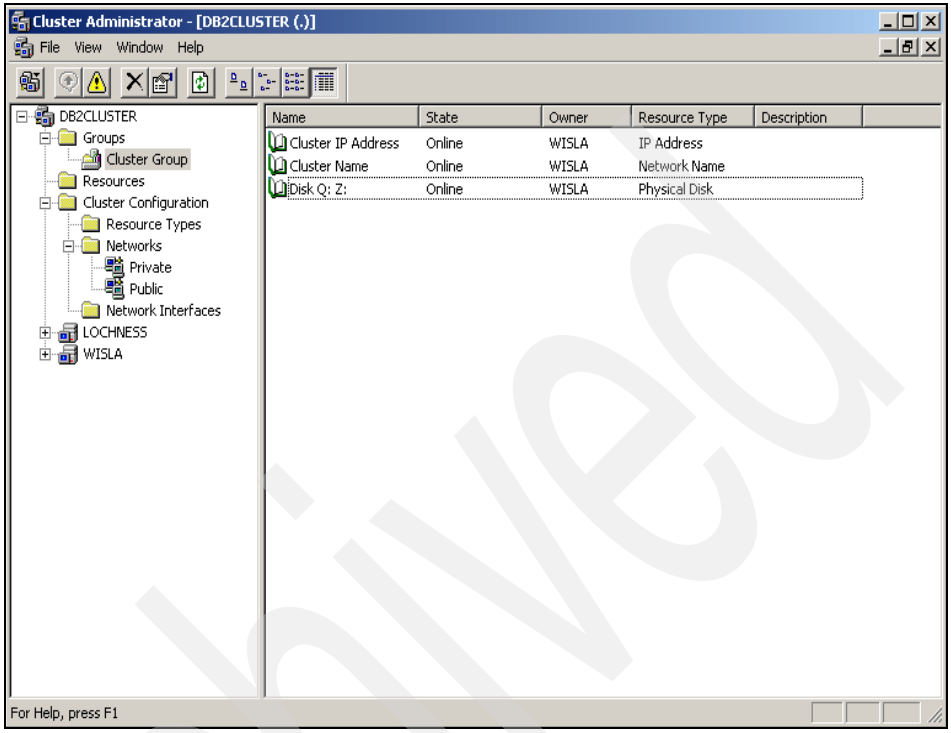

Figure 9-36 Cluster Administration window showing cluster group

7. Create resource.

   From the *Cluster Administrator* window, create a new resource of type **DB2** that will belong to **Cluster Group** as shown in Figure 9-37 on page 298

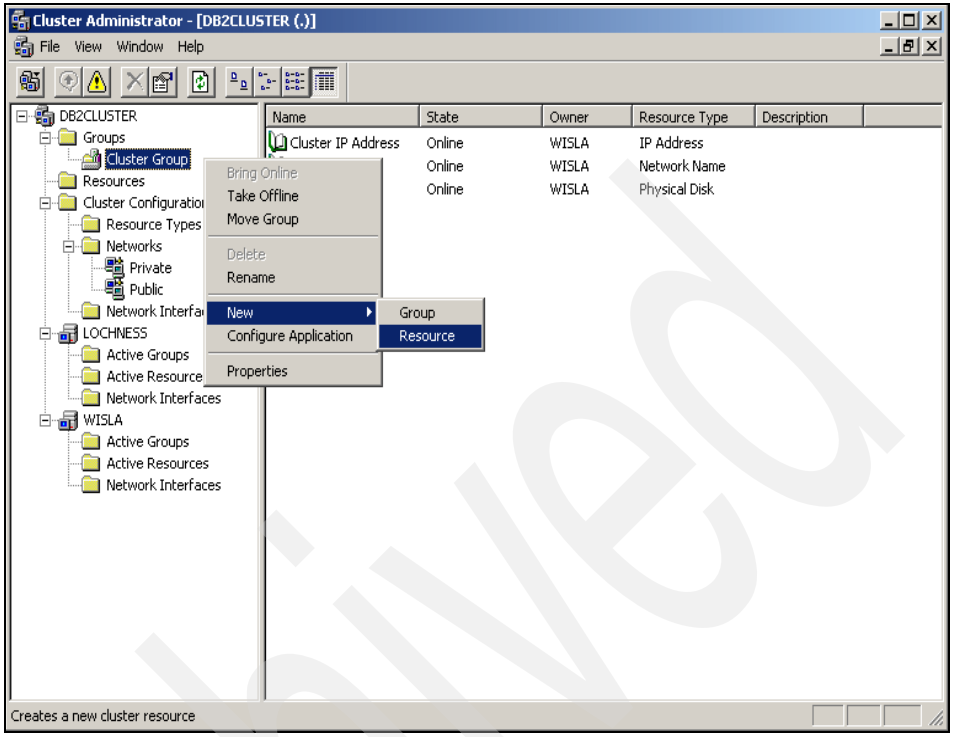

Figure 9-37 Cluster Administration window shows adding a new resource

The name of this resource must be exactly identical to the instance name; therefore, the resource is called **db2admin** as shown in Figure 9-38 on page 299.

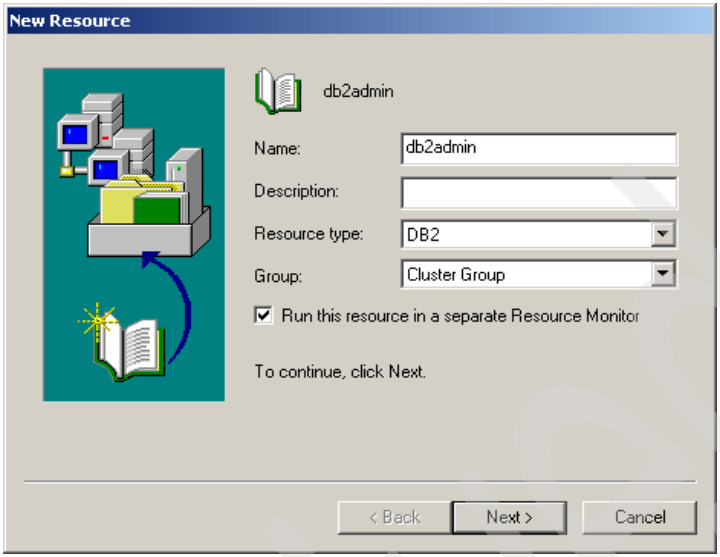

Figure 9-38 Adding a new resource to the cluster

Choose both primary (`WISLA`) and failover (`LOCHNESS`) servers as possible owners as shown in Figure 9-39.

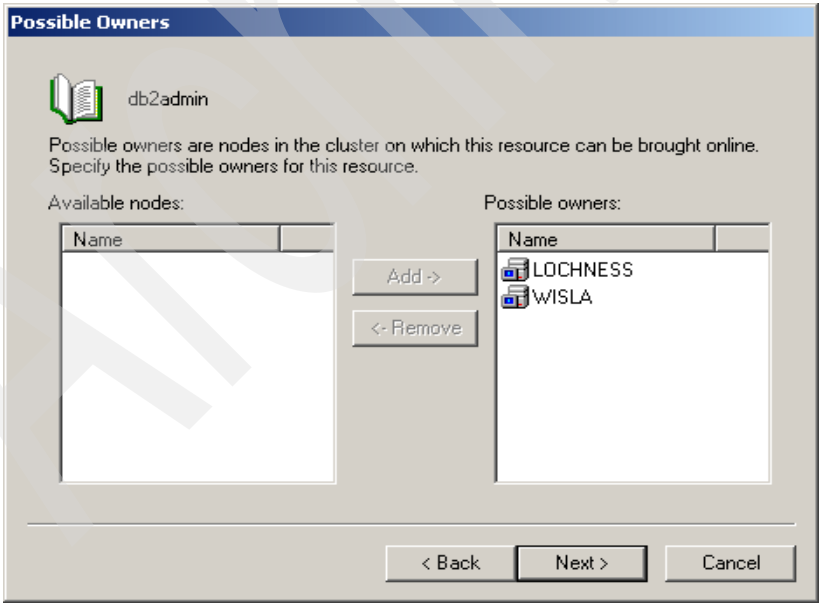

Figure 9-39 Adding possible owners for the new resource

When Cluster Administrator prompts for dependencies associated with the DB2 resource, ensure it is dependent on Cluster IP, Cluster Name, and two disks Q (quorum) and Z (shared data drive) and click **Finish** as shown in Figure 9-40, Figure 9-41, and Figure 9-42 on page 301.

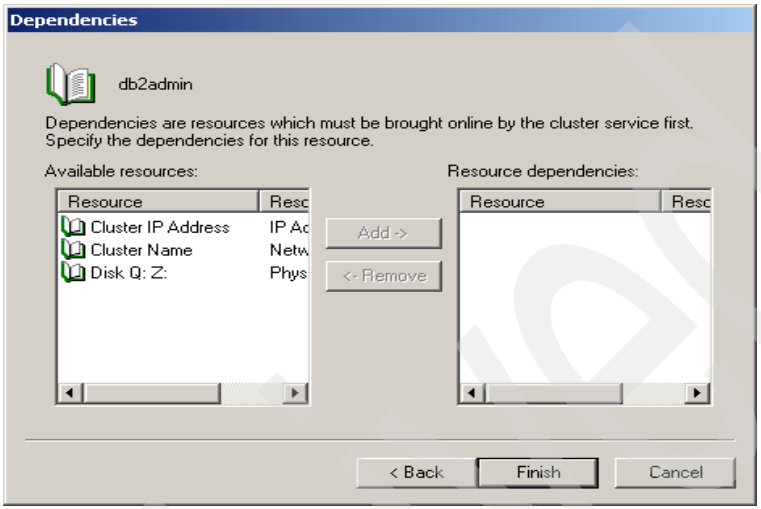

Figure 9-40 Possible dependencies to the new resource

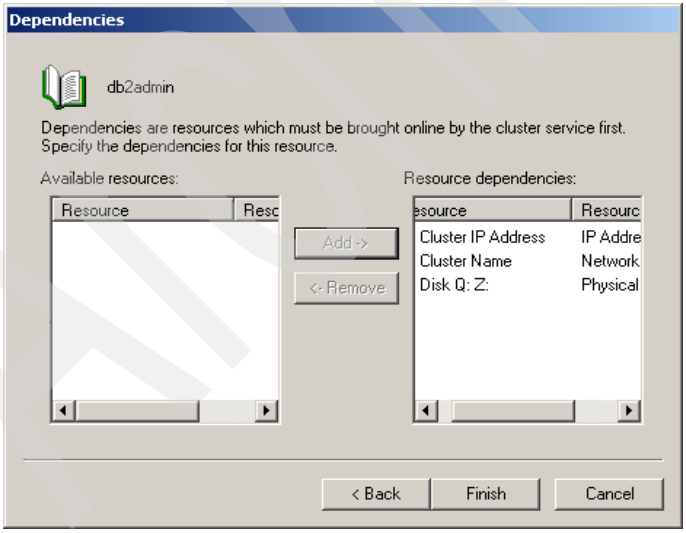

Figure 9-41 Added dependencies to the new resource

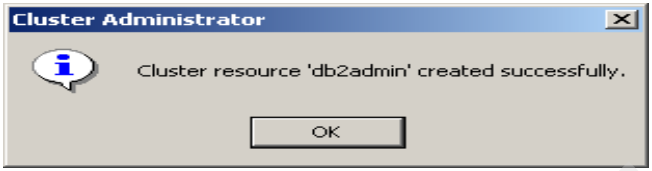

Figure 9-42 A new resource was added successfully to the cluster

8. Create or restore all databases on Disk Z.
9. Test the failover configuration.

   To test the configuration, right-click on **Cluster Group**, and select **Move Group**. Once the resources have moved, a window similar to Figure 9-43 will be displayed.

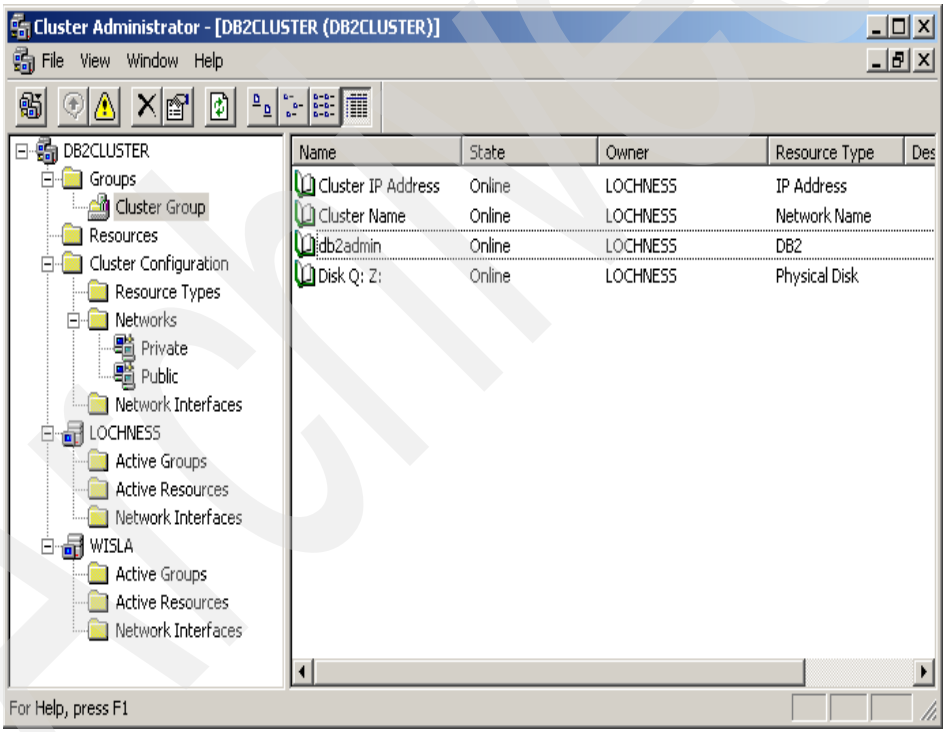

Figure 9-43 Cluster Administrator shows resources assigned (LOCHNESS)

To fail back, do the same on `LOCHNESS` and a cluster administration window similar to Figure 9-44 on page 302 will be displayed.

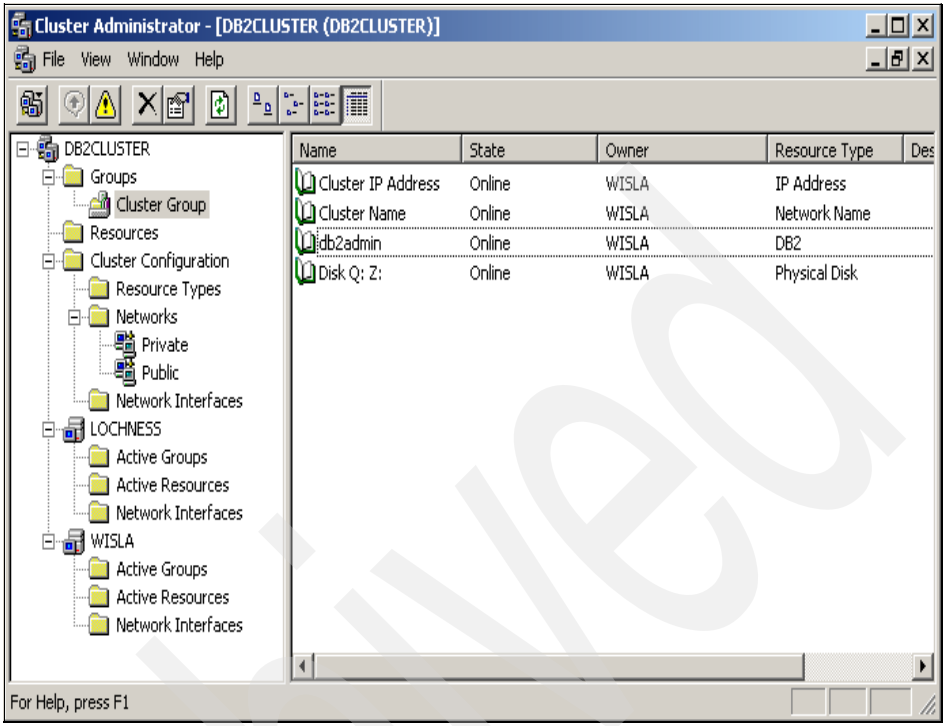

Figure 9-44 Cluster Administrator window showing resources assigned (WISLA)

9.3 Siebel Applications failover

In this section, we discuss failover of Siebel Gateway Name Server, Siebel Server, and Siebel File System. This section describes two scenarios, one on AIX and one on Windows. We show step by step failover process of the Windows scenario. Siebel Applications were installed after Microsoft Windows cluster and AIX HACMP have been configured. Siebel Applications should use the cluster name/service IP address and not the host name during Siebel installation.

You can have idle standby or active standby for Siebel Applications. In an idle standby failover scenario, the primary server has Siebel Gateway Name Server, Siebel Server, and Siebel File System installed on it. The secondary server is idle and will take over the Siebel Applications when a hardware failure occurs on the primary server.

The heartbeat connection between the two failover servers will be used by the servers to check if the other one is up or not. The service IP address, cluster name, and shared storage will be attached to the primary server. When a failover

occurs, service IP address, cluster name, and shared storage will be moved to the secondary server. The private network is used for communication between primary server and secondary server. The private network is optional, but heartbeat connection is required for a high availability solution.

In an active standby failover scenario, the standby server will perform its own activity and can act as secondary server in a failover. Siebel Gateway Name Server and Siebel File System are installed on the primary server. The Siebel Server is installed in local or non-shared disks on the secondary server. Siebel Server will not be part of the high availability configuration. The secondary server does not fail over to the primary server, therefore, Siebel Server will be unavailable if secondary server has hardware/network failures.

9.3.1 HACMP for Siebel Applications

HACMP installation and configuration is described in Section 9.2.3, “Installation and configuration of HACMP on AIX for DB2 UDB” on page 254. For Siebel Application failover, custom Siebel Gateway Name Server and Siebel Server start and stop scripts should be written. These scripts are called from the Application Server start and stop scripts. In our Lab environment, the application server name is `db2_siebel`. The `start_db2_siebel` script can be modified to call `start_gateway` and `start_siebel` scripts. In the same way the `stop_db2_siebel` script can be modified to call `stop_gateway` and `stop_siebel` scripts. If the Siebel File System is created on the shared volume group, then after a failover Siebel File System is automatically mounted by HACMP. The `start_db2_siebel`, `stop_db2_siebel`, `start_gateway`, `stop_gateway`, `start_siebel`, and `stop_siebel` scripts are located in Appendix A.3, “HACMP scripts” on page 351.

9.3.2 Windows clustering for Siebel Applications

Refer to 9.2.6, “DB2 UDB clustering on Windows” on page 286 for installing and configuring cluster manager on Windows systems.

The name of the cluster was `db2cluster` and has the following resources already configured in our cluster group.

- IP address resource
- Disk Z resource
- Hostname resource

The servers in our lab environment are `WISLA` and `LOCHNESS`. The primary server is `WISLA` and Z drive is the shared clustered drive. Siebel Gateway Name Server is installed on the Z: drive on the primary cluster server `WISLA`.

The following steps lead you through implementation of failover for Siebel Gateway Name Server.

Defining Gateway cluster service

Using **Start** → **Programs** → **Administration Tools** → **Cluster Administrator** on primary cluster server `WISLA` to bring up the cluster administration window. Create a new cluster resource by selecting **Cluster Group**, right-clicking and selecting **New** → **Resource** in the cluster administration window.

In *New Resource* window (Figure 9-45), provide a name and description for the gateway cluster resource. Make sure the resource type is **Generic Service** and do not select *Run this resource in a separate Resource Monitor*.

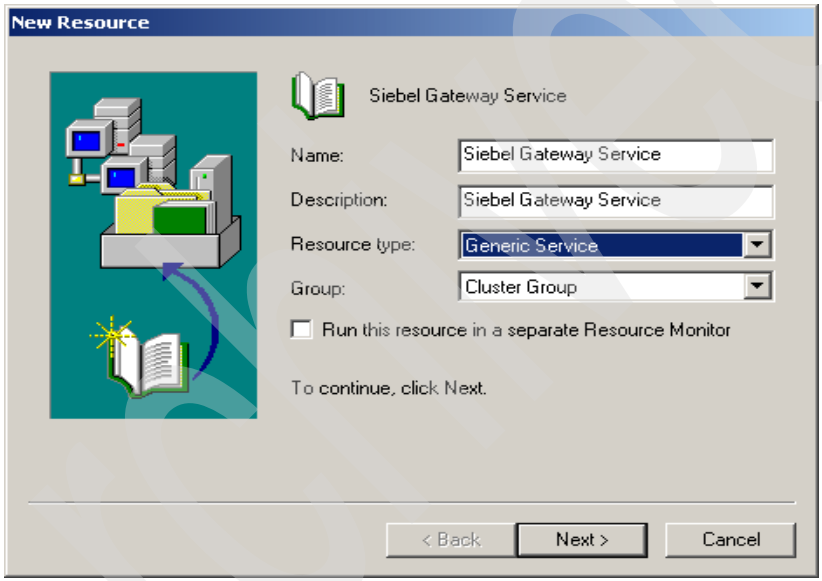

Figure 9-45 Define the Siebel Gateway Service

In *Possible Owners* window (Figure 9-46 on page 305), choose both the servers as possible owners of the resource.

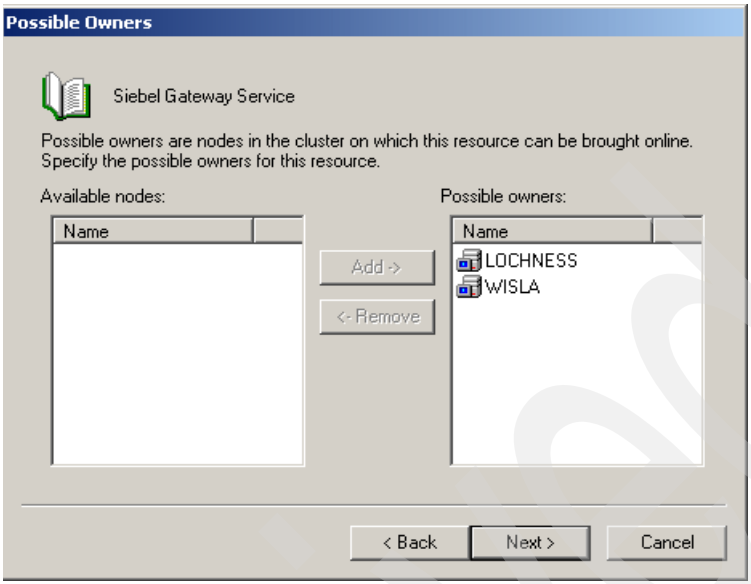

Figure 9-46 Define possible cluster owners

In *Dependencies* window (Figure 9-47), choose the dependencies of the Siebel Gateway resource. The dependencies are the cluster IP address, cluster Name, disk Q, and disk Z.

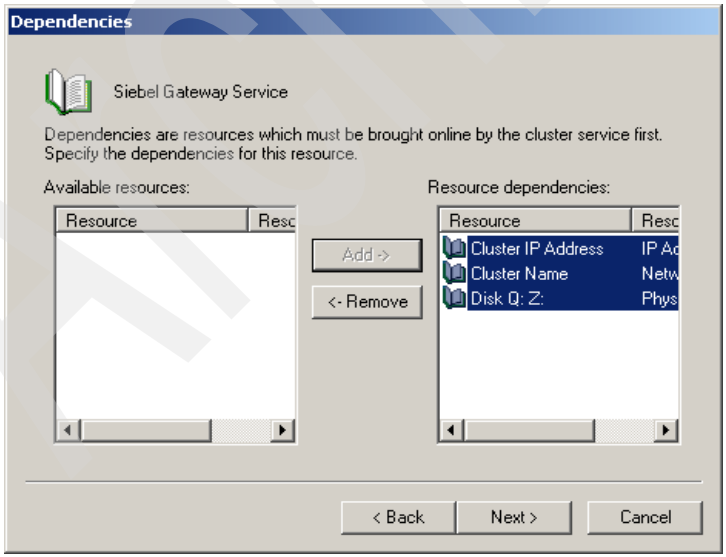

Figure 9-47 Define Siebel Gateway Service resource dependencies

In *Generic Service Parameters* window (Figure 9-48), provide **gtwyns** as the Siebel Gateway Name Server service name. Select **Use Network Name for computer name**.

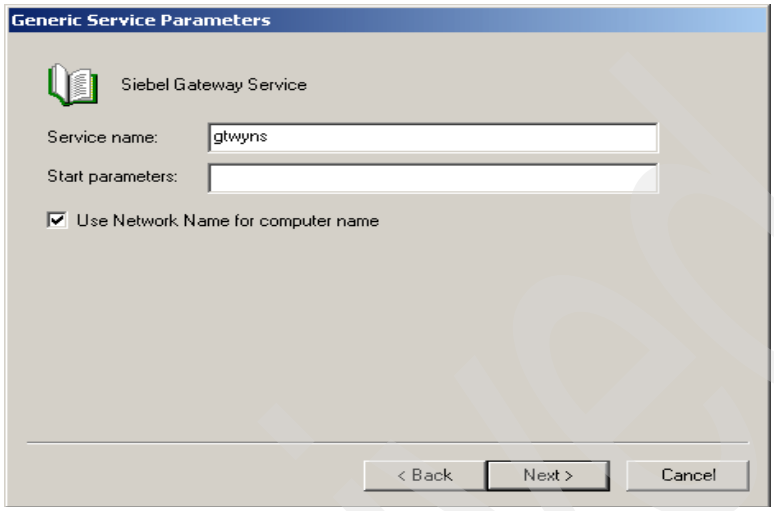

Figure 9-48 Define the gtwyns service name

In Registry Replication (Figure 9-49), choose the default **Root Registry Key**.

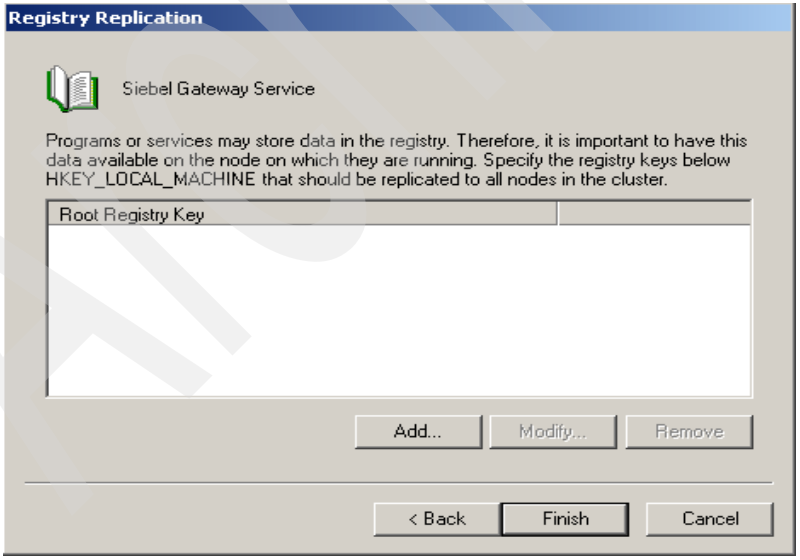

Figure 9-49 Define the Root Registry Key

A confirmation that the cluster resource “Siebel Gateway Service” has been successfully created will appear. In the cluster administration window, the Siebel Gateway Service will appear and will be offline. Right-click the Siebel Gateway Server and select **Bring online**, and wait for a minute to make sure it stays online.

Take the Gateway Cluster service offline on `WISLA`. Right-click the Siebel Gateway Service and select **Take Offline**. Fail the Cluster Group over to `LOCHNESS` server by right-clicking the cluster group and selecting **Move Group** as shown in Figure 9-33 on page 294.

Define Siebel Gateway Cluster on secondary server

Using **Start** → **Control Panel** → **Administrative Tools** → **Cluster Administration** to bring up the cluster administration window on the secondary server `LOCHNESS`. All the resources except Siebel Gateway service resource will be online.

Add registry entry on `LOCHNESS` server by running the following command from a DOS prompt as shown in Figure 9-50:

```
Z:\sea782\gtwysrvr\bin\siebctl –s gtwyns –a –g “/f
Z:\sea782\gtwysrvr\admin\siebns.dat” –u DomainName\DomainAccount –p xxxxx
–L enu –h Z:\sea782\gtwysrvr
```

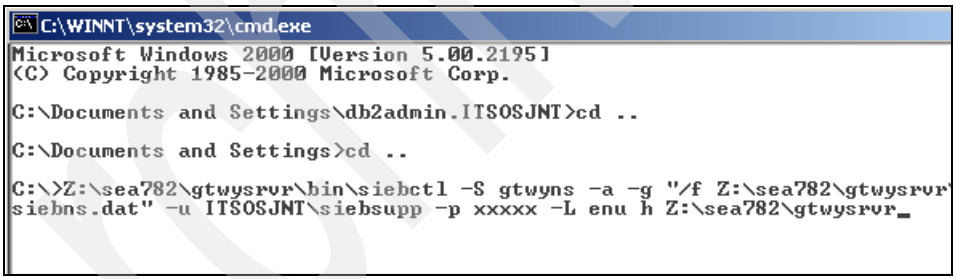

Figure 9-50 Adding Siebel Gateway Name Server Service

Right-click the Siebel Gateway server and select **Bring online**, and wait for a minute to make sure it stays online. Move the Cluster group back to the primary server `WISLA`. This completes the configuration of Siebel Gateway Name Server cluster configuration.

The usage of **`siebctl`** command is as follows:

```
Z:\sea782\gtwysrvr\BIN>siebctl
Program for controlling Siebel Server "NT Services"
Usage: siebctl [ <args> ... ]
   -S <service name> name of the service to invoke (required)
   -i <instance name> name of service "instance"
```

```
   -a                 add the specified service ("-g" required)
   -g <service args> arguments for service (used with "-a")
   -u                 account name for service (required with "-a")
   -p                 password for service (required with "-a")
   -h <home dir>      siebel home directory (defaults to environment
variable SIEBEL_HOME)
   -d                 delete the specified service
   -s                 start the specified service
   -k                 stop the specified service
   -r <prerequisite services>   comma separated list of prerequisite
services
   -t                 timeout in seconds for service shutdown
   -v                 query the specified service
   -A {yes|no}      specify service start type (used with "-a", defaults to
"no")
   -L <language>      message language code
   -q                 quiet mode
```

Clustering the Siebel Server

Clustering the Siebel server is the same as clustering the Siebel Gateway Name Server. The following are the minor differences between Siebel Server clustering and Siebel Gateway Name Server clustering.

Instead of `gtwyns`, the service name parameter is:

```
siebsrvr -i <EnterpriseName_ApplicationServerName
```

Example 9-7 gives the service name parameter for Siebel Server.

Example 9-7 Service name parameter for Siebel Server

```
siebsrvr -i siebwin_app001
where siebwin is the name of the enterprise and app001 is the name of the
Siebel Server.
```

If Siebel Gateway Name Server and Siebel Server are installed on the same server, then choose the Siebel Gateway Name Server as a dependency for Siebel Server.

The **`siebctl`** command is used to create the registry entry on the secondary server as follows:

```
Z:\sea782\siebsrvr\bin\siebctl -S siebsrvr -i "<Siebel EnterpriseName<_<Siebel
Server Name>" -a -g "-g  -e <Gateway Name Server Name> -s <Siebel Server Name>"
-u DomainName\DomainAccount -p <password> -L <Language> -h Z:\sea782\siebsrvr
```

Defining the Siebel File System for failover

The Siebel File System should be placed in a shared disk in the cluster. The Siebel File System should be shared, and all Siebel Servers should access the Siebel File System using the cluster name and not the host name of the server.

In our Lab environment, we add the Siebel File System cluster resource to the existing `db2cluster`. A Siebel File System folder was created in the cluster disk `Z:\sea782\siebelfs`. This Siebel File System was shared, accessible to Siebel Servers.

Using **Start** → **Control Panel** → **Administrative Tools** → **Cluster Administration** to bring up the cluster administration window on the primary server `WISLA`. Figure 9-51 shows how to create a new resource for the Siebel File System.

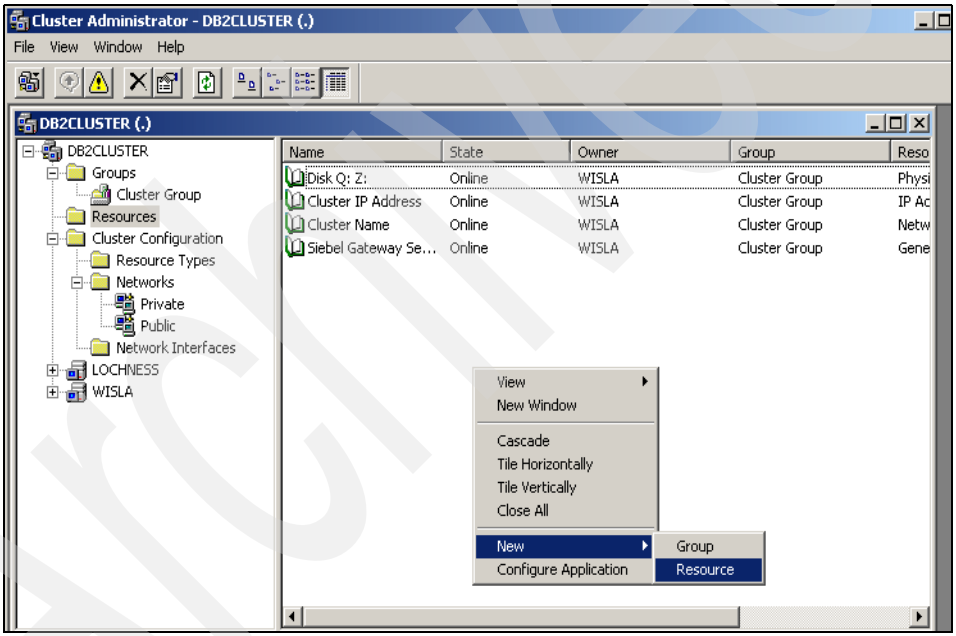

Figure 9-51 Create a new resource window

Figure 9-52 on page 310 shows the definitions of the Siebel filesystem cluster resource. The resource type should be defined as “File Share”.

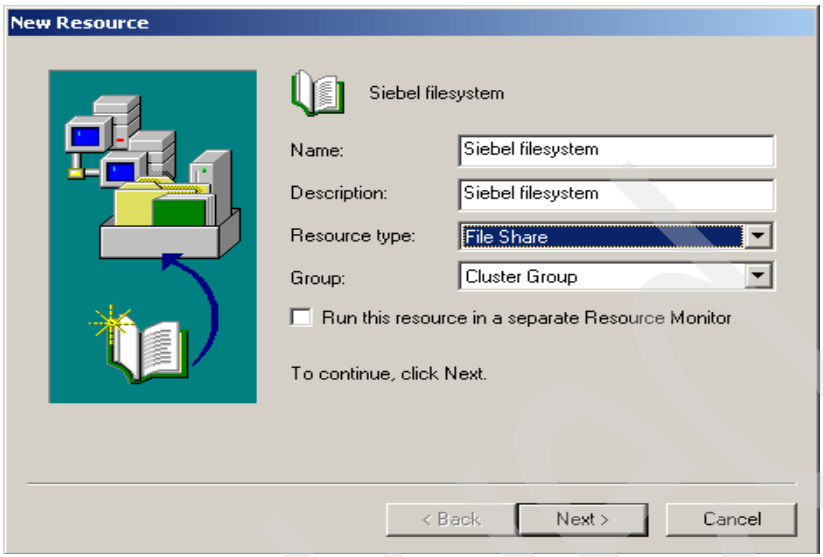

Figure 9-52 Define Siebel filesystem resource

In Figure 9-53, we define the owners of the Siebel File System cluster resource. The owners in our lab environment are `Wisla` and `Lochness`.

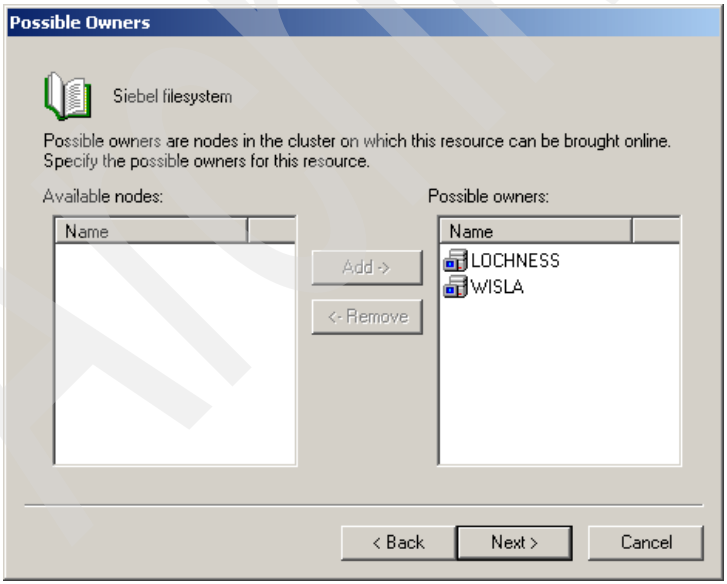

Figure 9-53 Owners of Siebel filesystem resource

In Figure 9-54, we define the dependencies of the Siebel filesystem cluster resource. The dependencies are the Cluster IP address, Cluster name, Disk Q, and Disk Z.

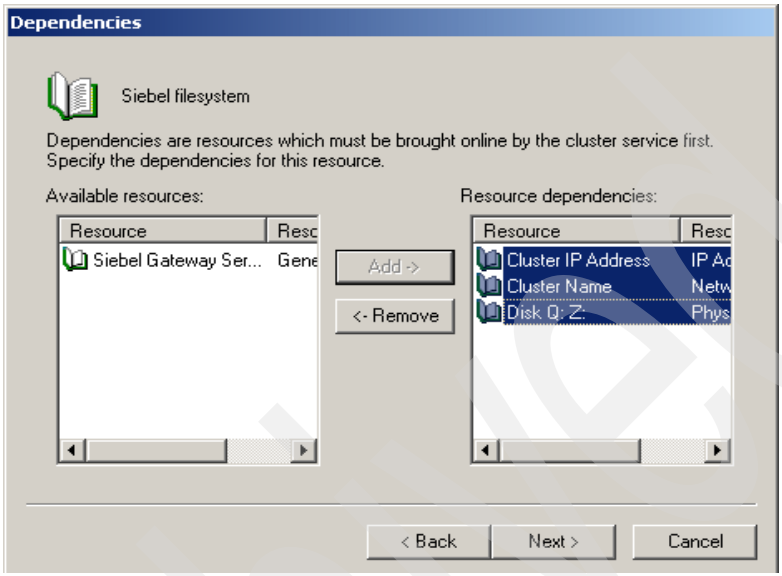

Figure 9-54 Siebel filesystem resource dependencies

In Figure 9-55, we define the Siebel filesystem share name and the location.

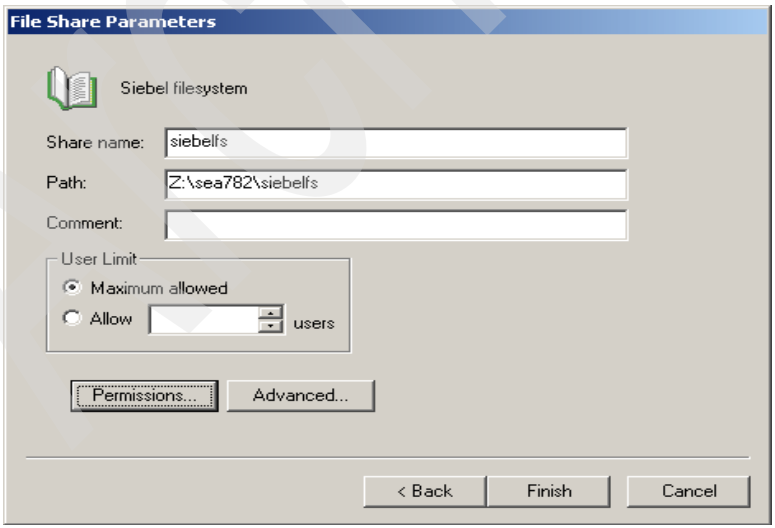

Figure 9-55 Define share name and path of the Siebel filesystem

A confirmation that the cluster resource “Siebel File System” has been successfully created will appear. In the Cluster Administration window, you should see the Siebel File System resource. Right-click on the Siebel File System resource to bring it online. Move the cluster group to the secondary server as shown in Figure 9-33 on page 294. Bring up the Cluster Administration window on `LOCHNESS`, all the resource groups including the Siebel File System resource will be online. The Siebel File System cluster configuration is complete.

9.4 Disaster recovery

Unforeseen situations can happen anytime, it might be as small as a user deleting wrong data or a water damage or flooding, to as big as sabotage by friendly and unfriendly resources. Needless to say natural circumstances such as fire and earthquakes can also happen.

Disaster recovery is about planning and being prepared for unplanned outages. With proper planning and training of the personnel, the data can be saved and restored at an alternate machine/location and resume normal operations.

Following are some of considerations in selecting a disaster recovery options for your business:

- How current should the data be on the recovery site.
- How much time do you have once the disaster is declared.
- What is the budget for the disaster.
- Security of data.
- What kinds of disaster do you want to protect against.
- Can we test our disaster recovery concept before the disaster happens? We recommend a minimum of annual testing of the disaster recovery plan and method and update the plan depending on results.

Here is a sample list of items that need to be tracked for the planning phase:

- The current backup schedule should be updated/changed to meet the HA and disaster recovery requirements.
- For a reliable and secure disaster recovery, backups may need to be sent off-site to a backup storage location. The backup schedule should consider how often tapes should be sent to an off-site storage location. Identify the retention period for tapes to be off-site.
- A copy of device types and software used in the backup solution should be kept along with backups. If you used a 20 GB tape drive to back up your database, it is recommended that you keep a 20 GB tape drive at the same

site as spare, so that you do not have to scramble to find a drive during a disaster recovery.

- There should be documentation and procedures in place regarding the site and type of server equipment where disaster recovery takes place.
- There should be clearly defined time lines by which the systems should be functional after a disaster has been declared.
- A project plan on the sequence of events and teams involved should be prepared.

9.4.1 Recovery for the Siebel environment

In this section, we discuss two of the most common options used in a disaster recovery for a Siebel environment.

Total recovery

Success of a disaster recovery depends on careful planning and consideration. Here is a sample list of items that are needed for a successful recovery for a Siebel environment:

- Operating system backups of the database server, Siebel Web Server, Siebel Gateway Name Server, and the Siebel Server.
- Backup of `/siebel` file system (AIX) or the drive (Windows) of the Gateway Name Server.
- Backup of `/siebel` file system (AIX) or the drive (windows) of the Siebel Server.
- Backup of `/var/adm/siebel` with UNIX operating system for database server, Siebel Web Server, Siebel Gateway Name Server, and the Siebel Server.
- Backup the Siebel File System (`/siebelfs`) immediately after database backup (full, incremental, and delta). OS tools available for backup can be used.
- Backup of the file system where SWSE is installed.
- Ensure the Web server code (`/usr/IBMIHS` in AIX) is backed up with the operating system backup.
- Full, incremental, and delta backups of the database.
- All database closed logs from the database.

Here is a sample list of items that need to be tracked for the execution phase:

- Once the site is declared as a disaster site, call the off-site storage tape location to send backups to the disaster recovery site.

- All required personnel should be informed and all personnel must either be present or connected to the disaster recovery site.
- Install the backup mechanism (tape drive) and make it functional.
- Restore operating systems on the database server, Siebel Gateway Name Server, Siebel Servers, and Siebel Web Servers from the backups.
- Configure the Network connectivity between the following Servers:
  - Siebel Servers and Database Server
  - Siebel Gateway Name Server and Siebel Server
  - Siebel Web Server and Siebel Gateway Name Server
  - Siebel Web Server and Siebel Server
- Restore the database from the full backup. Apply the incremental/delta backups and roll forward to the latest database log.
- Restore the Siebel Gateway Name Server code (`/siebel` file system (AIX) or drive on Windows) from the backups. Start the gateway.
- Restore the Siebel File System (`siebelfs`).
- Restore the Siebel Server code (`/siebel` file system (AIX) or drive on windows). Start the Siebel Servers.
- Restore the Siebel Web Server from the backups. Start the Web server.
- Login to the Siebel application and make sure all parts of the application are working properly.
- Testers should perform documented test procedures to ensure the functionality of the application before users access the application.

Standby recovery

The standby recovery is a method where the standby system will be a mirror image of the primary system.

Here is a sample list of tasks:

- Decide on what type of database solution you would like to use. Refer to 9.4.2, “DB2 UDB recovery options” on page 315 for details.
- Standby solutions can be as much as one day behind production system.
- Any system upgrades/patches/configuration to the production system also must be applied to the standby systems.
- Siebel File System: Daily Siebel File System changes must be copied over to the standby Siebel File System location. A third-party tool can be used for delta file transfer of the Siebel File System.

9.4.2 DB2 UDB recovery options

The disaster recovery site can be next machine, machine in next room, machine on next floor, machine in next building, machine at a remote site in the same city, machine at a remote site in the same country, machine at a remote site in another country. It all depends on what your requirements are for data protection.

There are several options available for disaster recovery. In the following sections, we cover several of the options and their advantages/disadvantages. Most disaster recoveries for DB2 UDB will fall under one of the following options.

- **Version restore**

  This is the simplest method, where you move/copy the backup to another machine and restore your databases. You can enhance this approach by adding incremental/delta backups (as discussed in Chapter 6, “Database administration” on page 129). Table 9-9 shows the advantages and disadvantages of the version restore recovery option.

  Table 9-9 Advantages and disadvantages of version restore

Advantages	Disadvantages
Low cost	Data is not current. Data is current as of the last backup (full, incremental, or delta).
Simple	Additional time needed to do the restore.
Database can be restored on a smaller box to keep the cost lower.	Database at the disaster recovery site is not available for access until disaster recovery happens.

- **Log Shipping**

  This is an extension of version restore. The transaction logs are shipped to the disaster recovery site and are ready to restore the database to the point-in-time of failure. This provides a better solution as compared to version restore because it gives you more complete data.

  To ensure that the logs are available when you need them, the following measurements are recommended:

  – Dual logging:
    This will protect you from storage subsystem failure, also with some of the newer storage systems you can have the second log at a different location but mounted across systems such that DB2 can write logs to it.

  – Userexit:
    This is a program routine that can interface with DB2 and ship the logs once the log has been closed (no open transactions on the log). The shipping method can be as simple as moving it to another file system and

compressing it. Depending on your needs, you can alter the routine to FTP the log to your disaster recovery site. Table 9-10 shows the advantages and disadvantages of log shipping.

Table 9-10 Advantages and disadvantages of log shipping

Advantages	Disadvantages
Reasonably low cost	Data is current to the last closed log.
Time taken for the database to be up and online is low (if the database at the target site is already in rollforward).	Additional time is needed to do the roll forward using the logs.
Database can be restored on a smaller box to keep the cost lower.	Database at the disaster recovery site is not available for access until disaster recovery happens.
Loss of data is kept to a minimum.	There is some data loss that may be unacceptable to some businesses such as banks, and so on.
	if you miss any logs then you are limited to version restore.

- **Data replication using DB2**

  DB2 UDB has a built-in replication process of which you can be take advantage. It is a very easy process and only has two moving parts, namely *Capture* and *Apply*.

  A DBA working with the business can identify the tables to be replicated from the source database to a set of tables in the target database. With this process, you can replicate only the tables you need and not the entire database. Distance is not an issue as long as there are network connectivity and available bandwidth to send the data across in a timely manner.

  The replication process is still asynchronous, since there is lag time for getting the data from the logs to the staging tables to the target database.

  You can use the target database for reporting and so on during normal time. Table 9-11 lists the advantages and disadvantages of data replication using DB2 UDB.

Table 9-11 Advantages and disadvantages of data replication using DB2 UDB

Advantages	Disadvantages
Currency of data available is really close to the source server.	Data is current as of the last run of apply on target.

Advantages	Disadvantages
The database on the target is available instantly (distance is not an issue).	There is an overhead to run replication.
Database can be on a smaller machine, on another platform, or another DBMS (architecture is flexible).	Connectivity is required to the target server all the time.
Loss of data is kept to a minimum.	There is data loss that may be unacceptable to certain businesses such as banks.
No extra cost to buy software.	DDL and database changes will affect.
Target server can do secondary workload.	

- **DB2 Q-replication**

  Q-replication is a high-volume, low-latency replication solution that uses WebSphere MQ message queues to transmit transactions between source and target databases or subsystems. The Q Capture program reads the DB2 recovery log for changes to a source table that you specify. The program then sends transactions as messages over queues, where they are read and applied to targets by the Q Apply program.

  This type of replication offers several advantages:

  – **Minimum latency**

    Changes are sent as soon as they are committed at the source and read from the log.

  – **High-volume throughput**

    The Q Capture program can keep up with rapid changes at the source, and the multi-threaded Q Apply program can keep up with the speed of the communication channel.

  – **Minimum network traffic**

    Messages are sent using a compact format, and data-sending options allow you to transmit the minimum amount of data.

  – **Asynchronous**

    The use of message queues allows the Q Apply program to receive transactions without having to connect to the source database or subsystem. If either of the replication programs is stopped, messages remain on queues to be processed whenever the program is ready. Because the messages are persistent, the source and target remain synchronized even in the event of a system or device failure.

Figure 9-56 illustrates a simple Q-replication setup. It shows the flow of transactional data from source server to the target server.

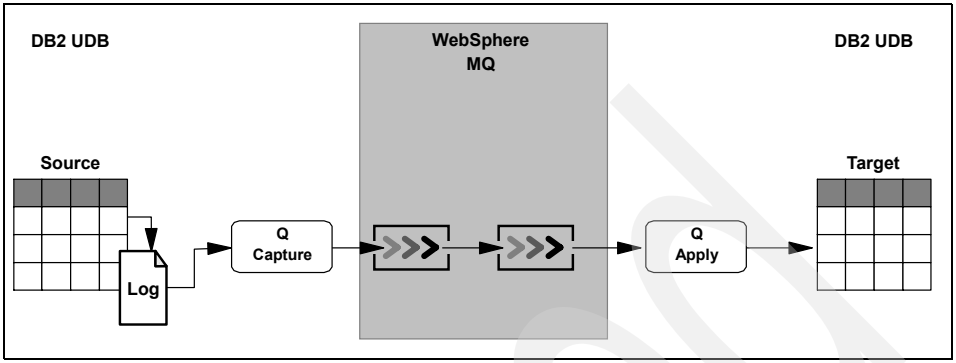

Figure 9-56 Q-replication example

There are three types of Q-replication

- Unidirectional replication
- Bi-directional replication
- Peer-to-Peer replication

For detailed information and how to set up Q-replication, use the following IBM DB2 UDB document, *Replication and Event Publishing Guide and Reference*, SC18-7568.

- **Using latest storage options**

DB2 UDB has built-in features to take advantage of several of the latest storage options such as PPRC, and so on. Using these storage options, you can put the primary database in write suspend and copy the database to a target. The options like flashcopy can perform the copy in a matter of seconds. Once the flash command is complete, you can then set write resume and release the database for users. The database is available for read operations during suspend mode.

The copy of data is done at a volume level in the storage. Using the `db2inidb` utility, you can then bring the database online in one of the three modes:

- **SNAPSHOT**:
  In this mode, `db2inidb` rolls back any open unit of work, and brings the database online to be available. This is most commonly used for creating reports, testing, and development environments.
- **STANDBY**:
  This will put the database in rollforward pending mode and you can run backups of this database, which can be used for a restore and roll forward at a later time. Using this option you can apply the logs and rollforward.

This option is similar to the log shipping option, but decreases/removes the time to restore the database.

– **MIRROR**:
This method will replace the copied volumes over the original and processing would take place in the same location.

Table 9-12 lists the advantages and disadvantages of these latest storage options.

Table 9-12 Advantages and disadvantages of the latest storage options

Advantages	Disadvantages
Time taken for the database to be up and online is short (If the database at the target site is already in rollforward).	Cost is high as advanced hardware is required.
Loss of data is kept to a minimum.	Additional time needed to do the rollforward using the logs.
Redundancy because the data is at two places.	Allocation of space for database at the target is same as source.
Large databases can be copied over a distance in relatively short time.	There is data loss that may be unacceptable to certain businesses such as banks and similar businesses.
All database changes are copied.	If you miss any logs then you are limited to version restore.
	Database at the disaster recovery site is not available for access until disaster recovery happens.

9.5 DB2 HADR

High Availability Disaster Recovery (HADR) provides a new alternative for delivering a high availability solution by replicating data from a source database, called the primary, to a target database, called the standby. HADR provides protection for both partial and complete site failures. Combined with the new automatic client reroute capability, HADR provides transparency to the application regardless of the failure type from hardware, network, or software issues to disaster scenarios like fire. HADR provides multiple levels of protection allowing flexibility in the environment. Additionally, DB2 UDB provides an easy to use wizard that allows the entire configuration to be set up in a matter of minutes.

Figure 9-57 on page 320 illustrates the HADR implementation.

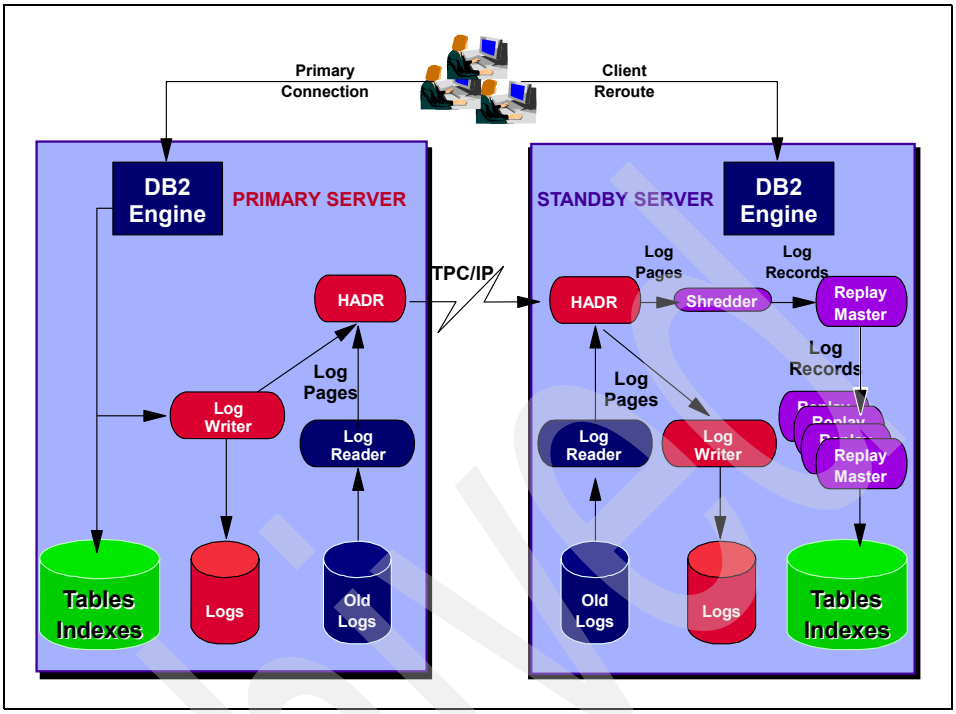

Figure 9-57 Simple DB2 HADR layout

HADR function is available as part of the DB2 UDB Enterprise Server Edition at no extra charge. Users of DB2 UDB Express and DB2 UDB Workgroup Server Editions can add HADR function to their servers by purchasing the DB2 UDB High Availability Disaster Recovery Option.

A failure can be caused by a hardware, network, or software failure. With HADR, the standby database can takeover the workload in a matter of seconds from any of these failures, including disk failure on the primary. Furthermore, you can have clients automatically redirected to the standby database (now the primary database) without the need for changes to your application with the new automatic client reroute facility of DB2 UDB V8.2.

Data changes made on the primary server are sent to the standby database directly from the log buffer of the primary server. Therefore, the two servers stay completely in sync with each other. There are three modes of operation for HADR, synchronous, near synchronous, and asynchronous.

- **Synchronous mode:**

  In synchronous mode, DB2 UDB ensures that the log records written to disk on the primary server are also written to disk on the standby server before an

application receives a successful return code to their commit statements. In this mode, there is a guarantee that no committed transactions will ever be lost as both servers stay completely in sync.

- **Near Synchronous mode:**

  In near synchronous mode, DB2 UDB ensures that the log records being written to disk on the primary server are in memory at the standby server (but may not yet have been written out to disk on the standby) prior to notifying an application that their commit statement was successful. In this mode, there will never be any transactions lost, unless both the primary and standby fail simultaneously.

- **Asynchronous mode:**

  In asynchronous mode, DB2 UDB will write the log buffer to disk on the primary server and ensure the log buffer has been passed down to the TCP/IP socket to be sent over to the standby. In this case, it would be possible to lose a committed transaction if the primary failed and the packets containing the log did not make it to the standby server prior to a takeover.

It is really important to understand that in HADR there are no shared disks as compared to some of the HA solutions discussed in 9.2.1, “HA options for DB2 UDB server” on page 248. Therefore, the standby server will be able to deliver the same performance (if the servers are similarly configured) as the primary server in a failover situation.

The following are the steps required to set up HADR between two servers. These steps can be performed manually or you can simplify the setup by using the HADR Configuration Wizard (from DB2 Control Center), which performs all of the following tasks:

- Identify primary and standby servers and instances
- Backup primary database and restore on standby
- Specify the TCP/IP ports over which HADR will communicate
- Enable client reroute
- Determine the synchronization mode

Once the above five steps are completed, start HADR on the standby and primary servers using the following commands:

```
db2 start hadr on database <database name> as standby
db2 start hadr on database <database name> as primary
```

This will start the HADR processes and bring the standby database in sync with the primary.

For more detail about HADR, refer to IBM DB2 UDB document: *Data Recovery and High Availability Guide and Reference V8*, SC09-4831-01.

10

Siebel Analytics

This chapter describes the Siebel Analytics architecture as it relates to Siebel CRM and OLTP database. The purpose of this chapter is not to delve deeply into the Siebel Analytics product and its components or to discuss install and administration aspects of Siebel Analytics since this redbook's theme is primarily focussed on Siebel CRM 7.8 on DB2 UDB. For readers who have both Siebel CRM as well as Siebel Analytics deployed in their environment, this chapter will be of special significance since it deals with the impact of Siebel Analytics' components on the OLTP database.

In this chapter, we discuss the overall architecture of Siebel Analytics including a discussion about the creation and maintenance of analytics tables, indexes, and triggers on the OLTP database. We also discuss planning considerations for data extraction from the OLTP database during the ETL (Extract Transform & Load) process. Lastly, we discuss some performance considerations that the Siebel Administrator/DBA needs to be aware of when their business runs Siebel Analytics in addition to the SIebel Enterprise Applications.

10.1 Architecture overview

Siebel Analytics architecture is shown in Figure 10-1. The overall architecture is very similar to the Siebel Enterprise Application n-tier architecture. There are equivalent components in Siebel Analytics. Now that you are familiar with the Siebel CRM 7.8 architecture, the comparison is straightforward. For example, both have a browser client; just like the Siebel Web Server Extension (SWSE) on the CRM side, there is a Siebel Analytics Web (SAW) component that takes requests from the browser. Similar to the Siebel Enterprise Server in the CRM solution, Siebel Analytics has its equivalent component named Siebel Analytics Server (SAS). Again, like the Siebel repository and Siebel Tools to manage it, analytics has a Siebel Analytics Repository and Siebel Analytics Administration Tool to manage and configure the repository. Lastly, like the Siebel OLTP database, there is the Siebel Relationship Management Warehouse (SRMW) Database, also sometimes referred to as the OLAP database. The DW can be any DW, not limited to Siebel.

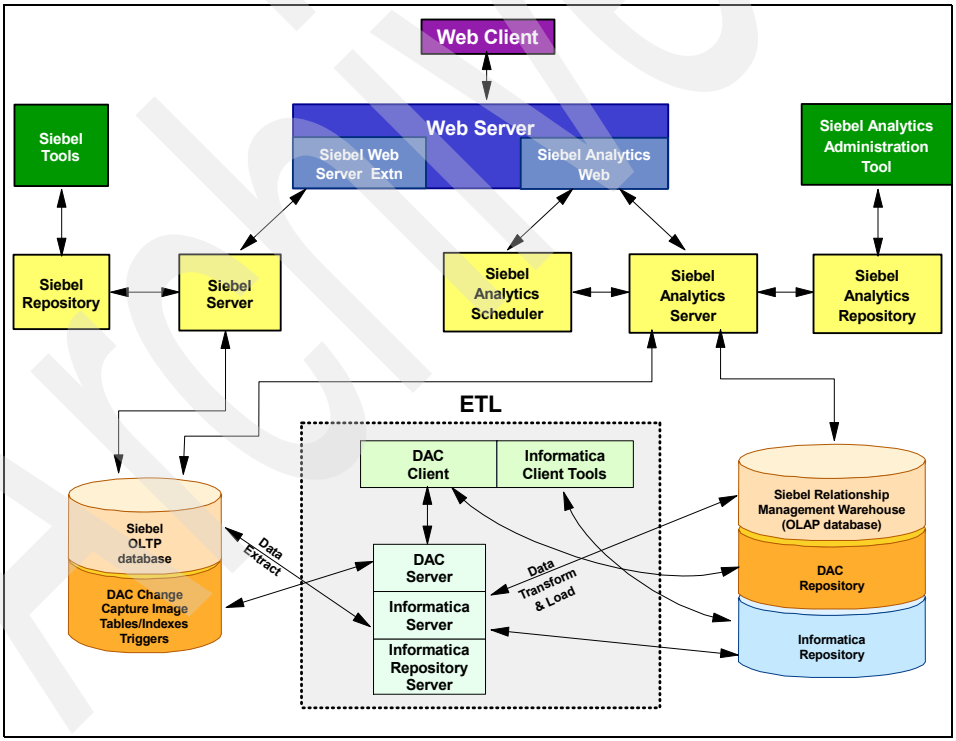

Figure 10-1 Siebel Analytics Architecture as deployed with Siebel Enterprise

Briefly, the process is as follows. The user logs in to the Web client and runs a report either from prebuilt Dashboards or by creating a new report using Siebel

Answers. The HTTP request is passed on to the Siebel Web Server which in turn sends the request to the Siebel Analytics Server. The Siebel Analytics Server converts the logical request into a physical SQL query based on the Siebel Analytics Repository configuration and sends the SQL to the SRMW database. The SQL is processed by the database and the results returned back through the Siebel Analytics Server and Siebel Analytics Web to the browser.

However, despite the architectural similarities, Siebel Analytics is a little different than the Siebel CRM. One major difference in the Siebel Analytics architecture is the ETL (Extract Transform & Load) component. This is a major component in the architecture and is the one that most affects the OLTP database. To load the SRMW database, Siebel Analytics uses prebuilt Informatica workflows/mappings that are stored in the Informatica repository, typically on the SRMW database and are executed by the Informatica server. Siebel Analytics also provides a Datawarehouse Application Console (DAC) that complements Informatica to manage and administer the ETL process, including calls to the Informatica Server for executing the workflows/mappings as well as tasks such as dropping and creating indexes, executing the DB2 utility `RUNSTATS`, and so on. The full features and capabilities of DAC are documented in the *Siebel Datawarehouse Installation and Administration Guide*. In this chapter, we discuss features that are relevant to the scope of this book in general and this chapter in particular.

10.1.1 DAC and Change-Capture process on OLTP database

To understand the impact of the ETL process on the OLTP database, we take a brief look at the data Change-Capture process implemented by Siebel Analytics on the OLTP database. For the purpose of extracting data from the OLTP database and loading it into the SRMW, a set of tables, commonly called the "Image" tables are created and maintained on the OLTP database. These include the `S_ETL_R_IMG_*` tables (R Image tables), the `S_ETL_I_IMG_*` tables (I Image tables) and the `S_ETL_D_IMG_*` tables (D Image tables). The `S_ETL_R_IMG_*` tables help identify rows in the OLTP tables that have been modified or newly created (inserted) during a period of time. For this purpose, the last update column of the source tables' records is compared to the `S_ETL_R_IMG_*` tables to effectively extract the records that fall within the ETL extract window specified in the DAC ETL preferences. The `S_ETL_I_IMG_*` tables are temporary holders that just store the row identifiers that have either changed or been newly created and are used for incremental ETL. To capture deletes in the OLTP database, however, the change capture process uses delete triggers in conjunction with the `S_ETL_D_IMG_*` tables. These triggers are created on the OLTP tables through DAC. When rows are deleted from the OLTP table, the triggers capture the row identifier into the corresponding `S_ETL_D_IMG` table. During the Change-Capture process run by DAC, it moves the deleted row information from the D Image tables to the I Image tables with the `OPERATION` column set to a value of "D".

During ETL, the preconfigured Informatica workflows/mappings, identified as the SDE (Source Dependent Extract) mappings, extract data from source OLTP database tables by querying views created by DAC with names starting with V_* and load the data into staging tables on the SRMW database. The view definition will change depending on whether the ETL run is a full load or incremental load run. In case of a full load, it would be a "`SELECT * FROM <base table>`". While in the case of an incremental load, the view definition joins the base table with the I Image tables. This is done to minimize the impact and duration of the ETL process on the OLTP database.

DAC drops and creates these views during each run (unless specified explicitly in one of the System properties called 'Drop Create Views Always'). These views can be dropped any time and DAC will create them when necessary.

In addition to the Image tables, there are a few more housekeeping tables (for example, `S_ETL_RUN`) and other tables named `S_ETL_*` (for example, `S_ETL_EXCH_RATE`) that are used by the ETL component of Siebel Analytics.

10.2 Creation and maintenance of Siebel Analytics tables in the OLTP database

Now that we understand the Change-Capture process and tables/triggers needed on the OLTP database, we are ready to explore different ways to create these tables. The detailed process for creation of these analytics objects on the OLTP database is described in the Siebel Bookshelf guide, *Siebel Datawarehouse Installation and Administration Guide*. A quick review of the process is in order here:

- Back up the Siebel Repository to a `.dat` file.
- Import the `.SIF` file (provided with Siebel Analytics software install) into the OLTP database using Siebel tools.
- Apply the schema definitions to the Siebel OLTP database.
- Verify the `*IMG*` (image) tables were created.

The Siebel CRM Administrator/OLTP DBA needs to be aware of the above tables and triggers and also be aware of the fact that the ETL process impacts the performance of the OLTP database in a number of different ways. While the ETL process runs, there may be resource contention as well as lock waits, lock escalations, and potential deadlocking on the OLTP database between the ETL connections and OLTP application users' connections.

Understanding this need, as well as the need for easy database administration, including table space level backup, we followed the following process and you

can use it while implementing Siebel Analytics. However, it must be noted that this is an alternate method that DAC provides to the above process of creating Image tables by applying the SIF file.

- Launch DAC client (Figure 10-2), **Start** → **Programs** → **Siebel Analytics** → **Siebel DAC** → **DAC Client**:

Figure 10-2 DAC Login

- Click **Design** tab and then click **Tables** tab to get to the DAC Design window as shown in Figure 10-3 on page 328.

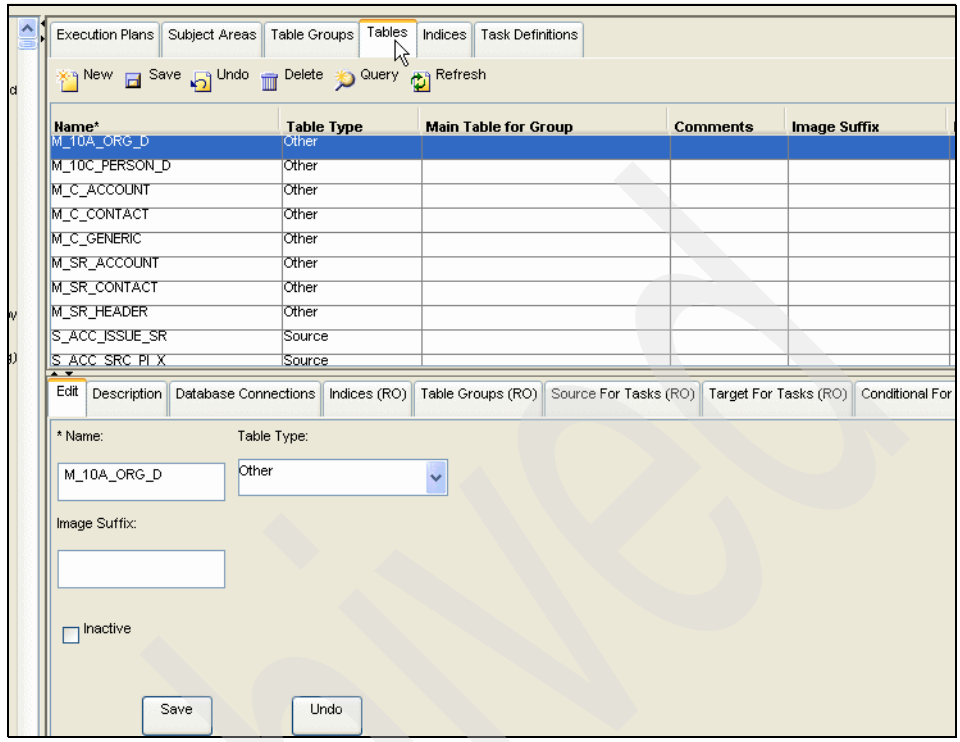

Figure 10-3 DAC Client Design window

- Click **Query** and then click the column named **Image Suffix**. Enter value **!NULL** in the column and click **Go** as in window shown in Figure 10-4 on page 329.

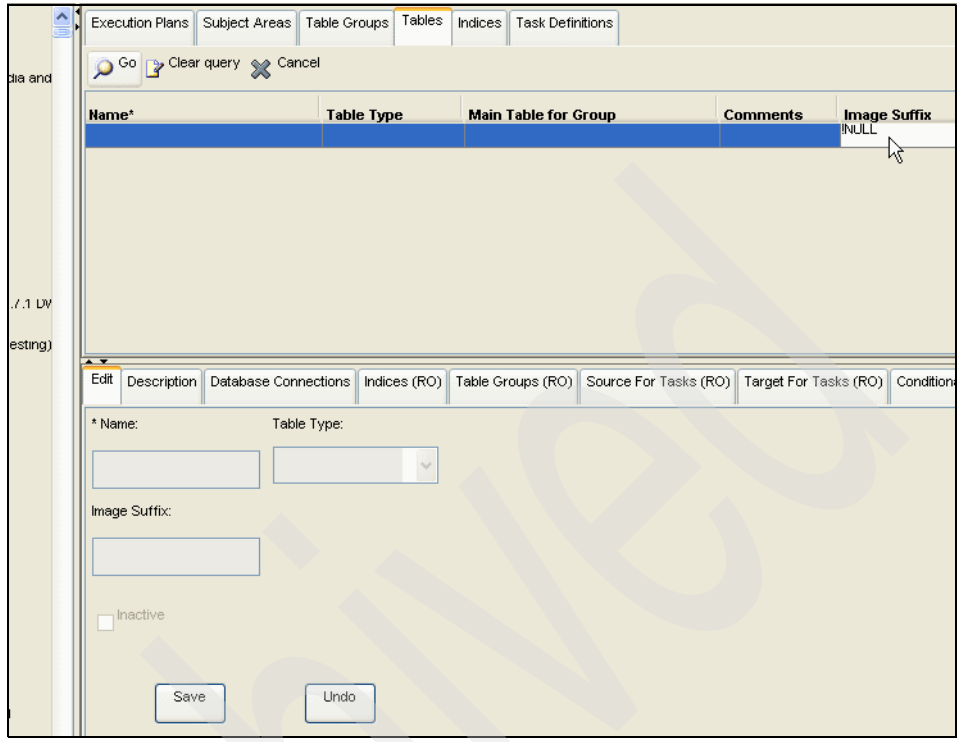

Figure 10-4 DAC Design: Entering image suffix

- The result should look similar to the window shown in Figure 10-5 on page 330.

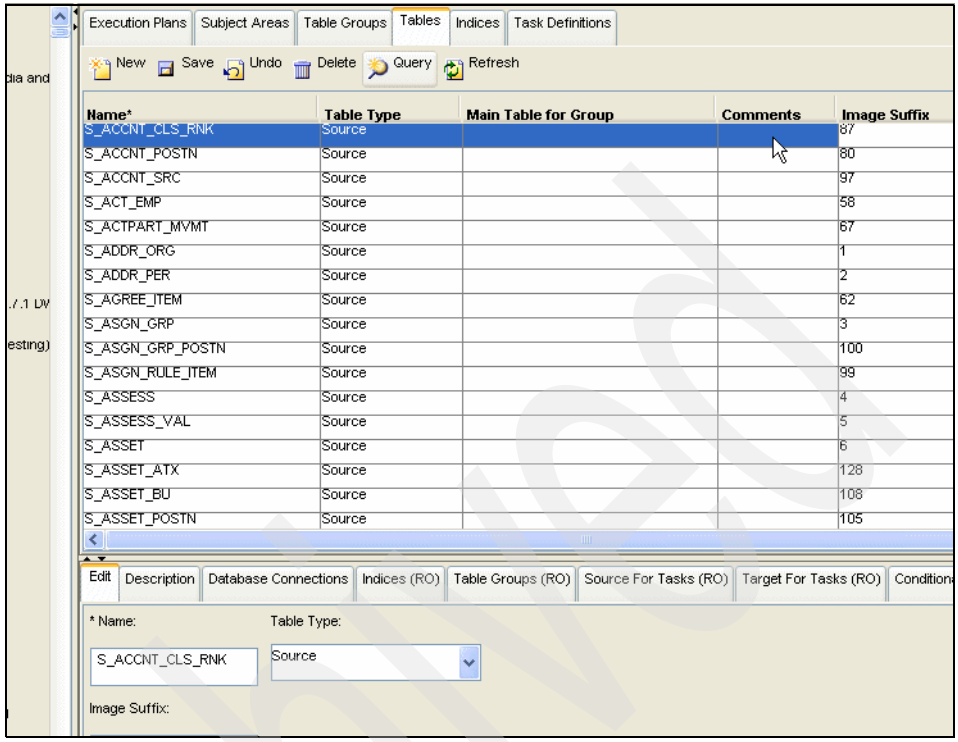

Figure 10-5 DAC Design: Query Results window

- You will notice that this query result set brings up all the table names that have a corresponding Image table. The Image Suffix shows the number suffixed to the image table name corresponding to the particular table. The aim here is to generate a script for the creation of image tables and triggers. From this window, right-click on any row, click **Change Capture scripts** and then click **Generate image and trigger scripts** to come to the window shown in Figure 10-6 on page 331.

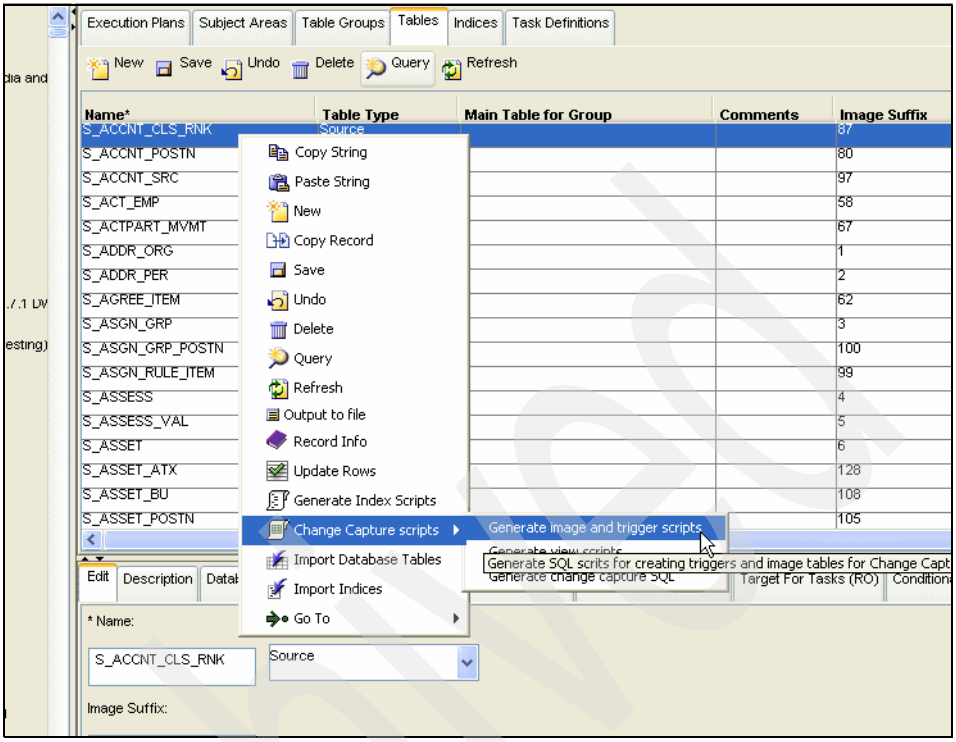

Figure 10-6 DAC Client showing generation of change capture scripts

- This will bring up the options window where we choose the option **All Tables in The List** and also select **DB2** from the "*Database Type*" options. At this time, we also select the **Unicode** option if the database is unicode. Lastly, you can choose to create scripts for Image tables and/or Triggers along with scripts to delete these before creating. These options are depicted in the window in Figure 10-7 on page 332.

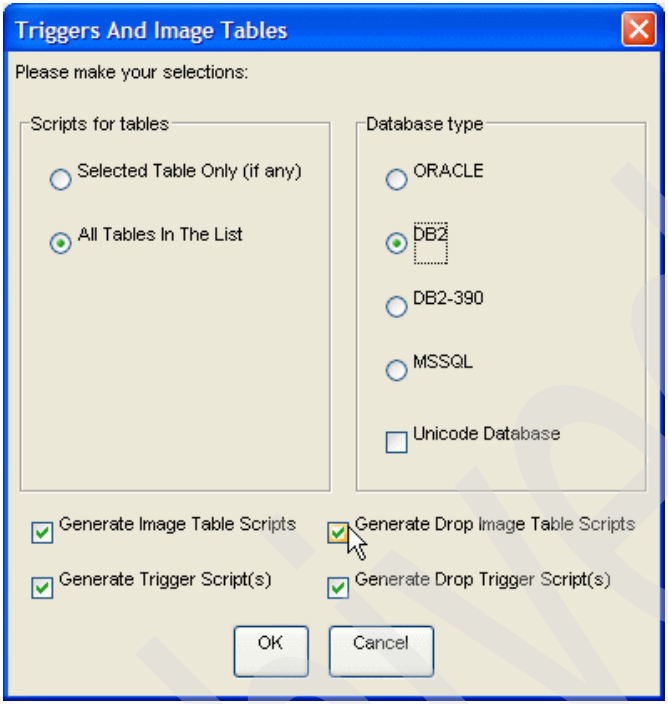

Figure 10-7 Triggers and Image Tables scripts generation options

- Click **OK** and this will generate the SQL script for dropping and creating the image tables and delete triggers on the OLTP database as shown in Figure 10-8 on page 333.

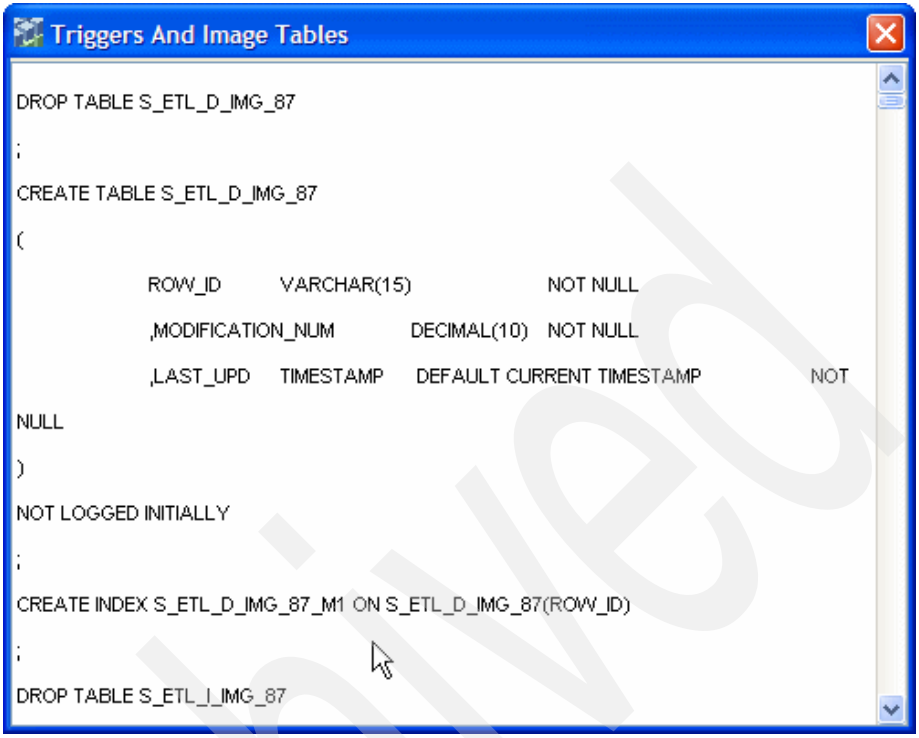

Figure 10-8 Drop and create script for image tables and delete triggers

- From here, select all text in the results window (use Ctrl-A) and copy-paste (Ctrl-C/Ctrl-V) to a notebook or other text-editor file and save it as depicted in Figure 10-9 on page 334.

```
Untitled - Notepad
File  Edit  Format  View  Help
DROP TABLE S_ETL_D_IMG_87
;
CREATE TABLE S_ETL_D_IMG_87
(
        ROW_ID  VARCHAR(15)      NOT NULL
        ,MODIFICATION_NUM       DECIMAL(10)      NOT NULL
        ,LAST_UPD       TIMESTAMP        DEFAULT CURRENT TIMESTAMP       NOT NULL
)
NOT LOGGED INITIALLY
;
CREATE INDEX S_ETL_D_IMG_87_M1 ON S_ETL_D_IMG_87(ROW_ID)
;
DROP TABLE S_ETL_I_IMG_87
;
CREATE TABLE S_ETL_I_IMG_87
(
        ROW_ID  VARCHAR(15)      NOT NULL
        ,MODIFICATION_NUM       DECIMAL(10)      NOT NULL
        ,LAST_UPD       TIMESTAMP        DEFAULT CURRENT TIMESTAMP       NOT NULL
        ,OPERATION      VARCHAR(1)       NOT NULL
)
NOT LOGGED INITIALLY
;
CREATE INDEX S_ETL_I_IMG_87_M1 ON S_ETL_I_IMG_87(OPERATION)
;
CREATE INDEX S_ETL_I_IMG_87_M2 ON S_ETL_I_IMG_87(ROW_ID)
;
DROP TABLE S_ETL_R_IMG_87
;
CREATE TABLE S_ETL_R_IMG_87
(
        ROW_ID  VARCHAR(15)      NOT NULL
        ,MODIFICATION_NUM       DECIMAL(10)      NOT NULL
        ,LAST_UPD       TIMESTAMP        DEFAULT CURRENT TIMESTAMP       NOT NULL
```

Figure 10-9 Drop and create image tables and delete triggers script

- The DBA can now freely modify the DDL script to include such details as schema, table space, and also include any other DDL special clauses needed. This method provides full control to the DBA so they know what database objects are being created and where. This eventually results in better database administration and maintenance.
- An important factor here is that the delete triggers should be created only for those tables for which the installation/business is interested in capturing delete information in the corresponding Data Warehouse tables. Avoiding the capture of deletes when not necessary for business will save a fair amount of processing and time that the ETL would spend on the OLTP database.

10.3 Data extraction from OLTP database for Siebel Analytics

Siebel Analytics uses the Siebel Relationship Management Warehouse (SRMW) as the back end database. SRMW is like any Online Analytical Processing (OLAP) database with a star schema. However, the schema is directly related to the Siebel OLTP database, from which data is extracted using a third-party tool (Informatica). Like any extract process in a data warehousing environment, the

Siebel ETL process impacts the OLTP database. Knowing the details of the ETL and the ways in which it will impact the OLTP database will help the Siebel DBA manage the databases as well as ETL in an optimal way.

Planning

Siebel Analytics and Siebel Relationship Management Warehouse (SRMW) depend on extracting data from the Siebel OLTP database on a regular schedule. The complete ETL process is the subject of discussion in the Siebel training "Analytics: Data Warehouse Developer". However, in this chapter, we only discuss the Extract part of ETL to see how it impacts the Siebel OLTP database as well the Siebel CRM application. As the term implies, Extract is the part where predefined processes and workflows pull data from the OLTP database and load it into the staging tables on the SRMW database. A number of factors can affect the performance and availability of the OLTP database and tables while these Extract jobs are running. The ETL process is designed to finish the extract in the least amount of time with the least interference with the OLTP database users. However, there are certain inherent design features in DB2 UDB which one needs to be aware of to make this process run without hiccups.

Performance

The performance of the Siebel CRM application and OLTP database may have been tuned well and users may be accustomed to a certain level of response times. Then, when Siebel Analytics is deployed alongside the ETL process, if not planned carefully, may cause some hindrances to the performance of OLTP database users. The Siebel DBA should be aware of the following database objects in the OLTP database that are used by Siebel Analytics and that may have an impact on the OLTP users, directly or indirectly.

- Additional indexes

  A few additional indexes are created on the OLTP base tables as part of applying the `.SIF` file while deploying Siebel Analytics. These indexes help run the Extract processes faster so that the ETL window remains a small window and should not be dropped.

- Delete triggers

  The delete triggers have minimal impact, but the DBA should be aware of their existence. The DBA should create delete triggers for only those OLTP tables from which data is being extracted through ETL.

- Collection of table and index statistics

  This is one area which cannot be overemphasized. Collecting statistics is part of any good database maintenance schedule. However, it is all the more necessary to do `RUNSTATS` on base tables before the ETL process runs. This

can be achieved by running the `RUNSTATS` utility regularly, especially after any bulk data load, index drop/creation, or table reorganization.

- Prune Days setting in DAC

  The DAC has a setting called “Prune Days” that can significantly impact the performance of the ETL run.The Prune Days setting determines what records from the OLTP database to extract based on the last ETL refresh date. For a detailed explanation of this and other DAC settings, please refer to the *Siebel Data Warehouse Installation and Administration Guide*. Briefly, if your installation does not use Siebel Remote for on-the-road sales personnel and if the `LAST_UPD` column on the OLTP tables is always current, then you must consider changing the default value of Prune Days to 1 or 2. This will, in effect, result in the ETL processes parsing far fewer records (smaller number of records), thereby resulting in a faster and more efficient extract. This can significantly reduce the overall total ETL timings.

  Prune Days is a global setting for DAC and affects extract from all OLTP tables. However, if after careful analysis of your deployment, you want to set this value differently for different OLTP tables, then Siebel Analytics allows for a provision through the use of the “`ChangeCaptureFilter.xml`” file that can be edited and modified to change the Prune Days for particular tables. This file can be found in the path `<Siebel Analytics installing path>\SiebelAnalytics\DAC\CustomSQLs`.

- Change Capture Filter Mechanism

  SIebel Analytics also provides a filtering mechanism that can restrict the records being extracted from OLTP tables. The “`ChangeCaptureFilter.xml`” file is again used for this purpose. Any valid `WHERE` clause can be added to a table section in this xml file and that will be used by the ETL process to filter and limit the number of records being extracted. Again, this should be done after a careful business analysis and thorough consideration of its impact. This can be particularly helpful for the big tables.

- Performance analysis tools

  The DAC provides excellent tools for ETL performance analysis. We highly recommend using this analysis to improve ETL performance.

- R Image tables

  The ETL process may seem to get slow after running fine for some time. One of the reasons we found was the growth of the R Image tables. These tables keep growing with each ETL run and as a result, the ETL process gradually becomes slow. After careful consideration, your installation should formulate a policy for these tables to either empty these out (delete) or prune (archive) these tables at regular intervals to make the ETL process efficient. Related to this, consider setting a small prune days value for DAC depending on the functionality of remote users implemented in the business.

- Resource contention

  Another factor that can impact performance on the OLTP database is the issue of resource contention between ETL processes and the OLTP users and the DBA needs to be aware of this. This can potentially result in lock waits, lock escalations, lock timeouts, or even deadlocks. The DBA should be on the lookout for these.

  For performance reasons as well as deadlock, timeout, and lock escalation issues, it is highly recommended that while ETL is running, there should be no bulk data batch jobs like EIM or EAI running. This can cause severe performance problems and even result in failure of ETL jobs.

  The DBA also needs to be aware that the Siebel Analytics Repository configuration and Siebel Answers & Dashboards may be impacting the OLTP database while Analytics users run their reports. This is due to the fact that some of the Dashboards and Siebel Answers' queries will try to pull the latest data from the OLTP database and then merge these results with the SRMW data to present a more updated result-set to the user. The OLTP database DBA needs to be aware of this.

A

Set up scripts

In this appendix, we provide a set of sample DB2 UDB scripts and HACMP scripts. We also provide sample Siebel load balancing and eapps.cfg files.

A.1 Database scripts for AIX

This section provides sample DB2 database creation and configuration scripts for an AIX operating system.

A.1.1 Directory creation

-- create.dir

```
#!/bin/ksh
if [ $# != 2 ]
then
  echo "Usage: $0 <instance> <dbname> "
  exit 2
fi
INSTANCE=$1
db=$2
db2 terminate > /dev/null 2>&1
export LD_LIBRARY_PATH=
export export CLASSPATH=
. /home/$INSTANCE/sqllib/db2profile
export LIBPATH=/home/$INSTANCE/sqllib/lib:/usr/lib:/lib
db2 drop db $db
db2 terminate
cd /db2/fs1;rm -r $db;mkdir $db;cd $db;mkdir $INSTANCE;cd $INSTANCE;mkdir
tbspaces;cd /db2/fs1;chmod -R 775 $db;chown -R aixdba.aixdbag $db
cd /db2/fs2;rm -r $db;mkdir $db;cd $db;mkdir $INSTANCE;cd $INSTANCE;mkdir
tbspaces;cd /db2/fs2;chmod -R 775 $db;chown -R aixdba.aixdbag $db
cd /db2/fs3;rm -r $db;mkdir $db;cd $db;mkdir $INSTANCE;cd $INSTANCE;mkdir
tbspaces;cd /db2/fs3;chmod -R 775 $db;chown -R aixdba.aixdbag $db
cd /db2/fs4;rm -r $db;mkdir $db;cd $db;mkdir $INSTANCE;cd $INSTANCE;mkdir
tbspaces;cd /db2/fs4;chmod -R 775 $db;chown -R aixdba.aixdbag $db
cd /dbbkup;rm -r "$db"log;mkdir "$db"log;cd "$db"log;mkdir $INSTANCE;cd
/dbbkup;chown -R aixdba.aixdbag "$db"log;chmod -R 770 "$db"log
cd /dbbkup;rm -r $db;mkdir $db;cd $db;mkdir $INSTANCE;cd /dbbkup;chown -R
aixdba.aixdbag $db;chmod -R 770 $db
cd /db2/fs5;rm -r "$db"log;mkdir "$db"log;cd "$db"log;mkdir $INSTANCE;cd
/db2/fs5;chown -R aixdba.aixdbag "$db"log;chmod -R 770 "$db"log
cd /db2/fs6;rm -r "$db"log;mkdir "$db"log;cd "$db"log;mkdir $INSTANCE;cd
/db2/fs6;chown -R aixdba.aixdbag "$db"log;chmod -R 770 "$db"log
```

A.1.2 Setting DB2 UDB registry variables

-- db2set.all.sh

```
#!/bin/ksh
if [ $# != 1 ]
```

```
then
  echo "Usage: $0 <instance> "
  exit 2
fi
INSTANCE=$1
db2 terminate > /dev/null 2>&1
export LD_LIBRARY_PATH=
export export CLASSPATH=
. /home/$INSTANCE/sqllib/db2profile
export LIBPATH=/home/$INSTANCE/sqllib/lib:/usr/lib:/lib
db2set DB2_HASH_JOIN=NO
db2set DB2_MMAP_WRITE=OFF
db2set DB2_MMAP_READ=OFF
db2set DB2_CORRELATED_PREDICATES=YES
db2set DB2_PIPELINED_PLANS=ON
db2set DB2_INTERESTING_KEYS=ON
db2set DB2_PARALLEL_IO=*
db2set DB2_STRIPED_CONTAINERS=ON
db2set DB2MEMMAXFREE=1000000
db2set DB2MEMDISCLAIM=YES
db2set DB2_NO_PKG_LOCK=OFF
db2set DB2ENVLIST=EXTSHM
db2set DB2COMM=TCPIP
db2set DB2CODEPAGE=1208
db2set DB2AUTOSTART=TRUE

echo
echo "Registry values are now as follows"
echo
db2set -all
```

A.1.3 Database manager configuration parameters

-- dbmcfg.sh

```
#!/bin/ksh

if [ $# != 1 ]
then
  echo "Usage: $0 <instance> "
  exit 2
fi

INSTANCE=$1

db2 terminate > /dev/null 2>&1
```

```
export LD_LIBRARY_PATH=
export export CLASSPATH=
. /home/$INSTANCE/sqllib/db2profile
export LIBPATH=/home/$INSTANCE/sqllib/lib:/usr/lib:/lib

db2 "update dbm cfg using SHEAPTHRES 200000"
db2 "update dbm cfg using DIR_CACHE YES"
db2 "update dbm cfg using QUERY_HEAP_SZ 16384"
db2 "update dbm cfg using ASLHEAPSZ 15"
db2 "update dbm cfg using RQRIOBLK 65535"
db2 "update dbm cfg using MON_HEAP_SZ 512"
db2 "update dbm cfg using KEEPFENCED YES"
db2 "update dbm cfg using NUM_POOLAGENTS 80"
db2 "update dbm cfg using NUM_INITAGENTS 10"
db2 "update dbm cfg using MAXAGENTS 1000 MAX_COORDAGENTS 1000"
db2 "update dbm cfg using INDEXREC RESTART"
db2 "update dbm cfg using MAX_QUERYDEGREE 1"
db2 "update dbm cfg using INTRA_PARALLEL NO"
db2 "update dbm cfg using DISCOVER DISABLE"
db2 "update dbm cfg using DISCOVER_INST DISABLE"
db2 "update dbm cfg using svcename 8100"

# turn on monitor switches
db2 "update dbm cfg using DFT_MON_BUFPOOL ON"
db2 "update dbm cfg using DFT_MON_LOCK ON"
db2 "update dbm cfg using DFT_MON_SORT ON"
db2 "update dbm cfg using DFT_MON_STMT ON"
db2 "update dbm cfg using DFT_MON_UOW ON"
db2 "update dbm cfg using DFT_MON_TABLE ON"

echo
echo "DBM cfg is now as follows:"
echo

db2 get dbm cfg
```

A.1.4 Database creation

-- create_unicode_db.sh

```
#!/bin/ksh

if [ $# != 2 ]
then
  echo "Usage: $0 <instance> <dbname> "
  exit 2
fi

INSTANCE=$1
```

```
db=$2

db2 terminate > /dev/null 2>&1
export LD_LIBRARY_PATH=
export export CLASSPATH=
. /home/$INSTANCE/sqllib/db2profile
export LIBPATH=/home/$INSTANCE/sqllib/lib:/usr/lib:/lib

db2 "CREATE DATABASE $db  ON '/db2/fs1/$db' \
  USING CODESET UTF-8  TERRITORY US \
  COLLATE USING Identity \
  DFT_EXTENT_SZ 32 \
  CATALOG TABLESPACE MANAGED BY DATABASE \
  USING \
   (File '/db2/fs1/$db/$INSTANCE/SYSCATSPACE/syscat.1' 25000, \
    File '/db2/fs2/$db/$INSTANCE/SYSCATSPACE/syscat.2' 25000, \
    File '/db2/fs3/$db/$INSTANCE/SYSCATSPACE/syscat.3' 25000, \
    File '/db2/fs4/$db/$INSTANCE/SYSCATSPACE/syscat.4' 25000 \
   ) \
    EXTENTSIZE 32 \
    PREFETCHSIZE 128"

rc=$?

db2 get db cfg for $db

db2 connect to $db
db2 grant dbadm on database  to user siebel
db2 set event monitor db2detaildeadlock state 0
db2 drop event monitor db2detaildeadlock
db2 connect reset

exit $rc
```

A.1.5 Database configuration parameters

```
#!/bin/ksh


if [ $# != 2 ]
then
  echo "Usage: $0 <instance> <dbname> "
  exit 2
fi


INSTANCE=$1
```

```
db=$2

db2 terminate > /dev/null 2>&1
export LD_LIBRARY_PATH=
export export CLASSPATH=
. /home/$INSTANCE/sqllib/db2profile
export LIBPATH=/home/$INSTANCE/sqllib/lib:/usr/lib:/lib


db2 "update db cfg for $db using DFT_DEGREE 1"
db2 "update db cfg for $db using DFT_QUERYOPT 3"
db2 "update db cfg for $db using DBHEAP 10000"
db2 "update db cfg for $db using CATALOGCACHE_SZ 8000"
db2 "update db cfg for $db using LOGBUFSZ 512"
db2 "update db cfg for $db using UTIL_HEAP_SZ 10000"
db2 "update db cfg for $db using LOCKLIST 25000"
db2 "update db cfg for $db using APP_CTL_HEAP_SZ 900"
db2 "update db cfg for $db using SORTHEAP 1024"
db2 "update db cfg for $db using STMTHEAP 40960"
db2 "update db cfg for $db using PCKCACHESZ 40000"
db2 "update db cfg for $db using STAT_HEAP_SZ 20000"
db2 "update db cfg for $db using MAXLOCKS 30"
db2 "update db cfg for $db using LOCKTIMEOUT 30"
db2 "update db cfg for $db using CHNGPGS_THRESH 30"
db2 "update db cfg for $db using INDEXSORT YES"
db2 "update db cfg for $db using SEQDETECT YES"
db2 "update db cfg for $db using DFT_PREFETCH_SZ 32"

db2 "update db cfg for $db using LOGRETAIN ON"
db2 "update db cfg for $db using USEREXIT ON"

db2 "update db cfg for $db using MAXAPPLS 100"
db2 "update db cfg for $db using AVG_APPLS 25"
db2 "update db cfg for $db using MAXFILOP 500"
db2 "update db cfg for $db using LOGFILSIZ 40000"
db2 "update db cfg for $db using LOGPRIMARY 20"
db2 "update db cfg for $db using LOGSECOND 107"
db2 "update db cfg for $db using SOFTMAX 80"
db2 "update db cfg for $db using APPLHEAPSZ 2500"
db2 "update db cfg for $db using NUM_IOCLEANERS 4"
db2 "update db cfg for $db using NUM_IOSERVERS 5"
db2 "update db cfg for $db using NEWLOGPATH /db2/fs5/"$db"log/$INSTANCE"
db2 "update db cfg for $db using MIRRORLOGPATH /db2/fs6/"$db"log/$INSTANCE"

db2 deactivate db $db
db2 activate db $db


echo
```

```
echo " DB cfg for $db is now as follows: "
echo
db2 get db cfg for $db
```

A.1.6 Creation of buffer pools

-- create_bp.sh

```
#!/bin/ksh

if [ $# != 2 ]
then
  echo "Usage: $0 <instance> <dbname> "
  exit 2
fi

INSTANCE=$1
db=$2

db2 terminate > /dev/null 2>&1
export LD_LIBRARY_PATH=
export export CLASSPATH=
. /home/$INSTANCE/sqllib/db2profile
export LIBPATH=/home/$INSTANCE/sqllib/lib:/usr/lib:/lib

db2 connect to $db

db2 CREATE BUFFERPOOL "BUFF4K"  SIZE 14000 PAGESIZE 4096 NOT EXTENDED STORAGE

db2 CREATE BUFFERPOOL "BUFF16K"  SIZE 14000 PAGESIZE 16384 NOT EXTENDED STORAGE

db2 CREATE BUFFERPOOL "BUFF32K"  SIZE 21750 PAGESIZE 32768 NOT EXTENDED STORAGE

db2 terminate > /dev/null 2>&1

db2 deactivate db $db
db2 activate db $db
```

A.1.7 Creation of temporary table spaces

-- create_temptbs.ddl

```
CREATE TEMPORARY TABLESPACE TEMP4KA IN DATABASE PARTITION GROUP IBMTEMPGROUP
PAGESIZE 4096 MANAGED BY SYSTEM
         USING ('/db2/fs1/vol78rba/ucsdb2/tbspaces/temp4ka1',
                '/db2/fs2/vol78rba/ucsdb2/tbspaces/temp4ka2',
```

```
                '/db2/fs3/vol78rba/ucsdb2/tbspaces/temp4ka3',
                '/db2/fs4/vol78rba/ucsdb2/tbspaces/temp4ka4')
        EXTENTSIZE 32
        PREFETCHSIZE 128
        BUFFERPOOL BUFF4K;

CREATE TEMPORARY TABLESPACE TEMP16KA IN DATABASE PARTITION GROUP IBMTEMPGROUP
PAGESIZE 16384 MANAGED BY SYSTEM
        USING ('/db2/fs1/vol78rba/ucsdb2/tbspaces/temp16ka1',
                '/db2/fs2/vol78rba/ucsdb2/tbspaces/temp16ka2',
                '/db2/fs3/vol78rba/ucsdb2/tbspaces/temp16ka3',
                '/db2/fs4/vol78rba/ucsdb2/tbspaces/temp16ka4')
        EXTENTSIZE 32
        PREFETCHSIZE 128
        BUFFERPOOL BUFF16K;

CREATE TEMPORARY TABLESPACE TEMP32KA IN DATABASE PARTITION GROUP IBMTEMPGROUP
PAGESIZE 32768 MANAGED BY SYSTEM
        USING ('/db2/fs1/vol78rba/ucsdb2/tbspaces/temp32ka1',
                '/db2/fs2/vol78rba/ucsdb2/tbspaces/temp32ka2',
                '/db2/fs3/vol78rba/ucsdb2/tbspaces/temp32ka3',
                '/db2/fs4/vol78rba/ucsdb2/tbspaces/temp32ka4')
        EXTENTSIZE 32
        PREFETCHSIZE 128
        BUFFERPOOL BUFF32K;
```

A.1.8 Creation of Siebel table spaces

```
-- create_tbs.ddl

CREATE REGULAR TABLESPACE SIEBEL_4K IN DATABASE PARTITION GROUP IBMDEFAULTGROUP
PAGESIZE 4096 MANAGED BY DATABASE USING(
DEVICE '/dev/rvpath0cont01lv'  524288,
DEVICE '/dev/rvpath0cont02lv'  524288
        )EXTENTSIZE 32
        PREFETCHSIZE 64
        BUFFERPOOL BUFF4K;

CREATE REGULAR TABLESPACE SIEBEL_4KI IN DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 4096 MANAGED BY DATABASE USING(
DEVICE '/dev/rvpath0cont03lv'  524288,
DEVICE '/dev/rvpath0cont04lv'  524288
        )EXTENTSIZE 32
        PREFETCHSIZE 64
        BUFFERPOOL BUFF4K;
```

```
CREATE REGULAR TABLESPACE SIEBEL_16K IN DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 16384 MANAGED BY DATABASE USING(
DEVICE '/dev/rvpath0cont05lv' 131072,
DEVICE '/dev/rvpath0cont06lv' 131072
         )EXTENTSIZE 32
         PREFETCHSIZE 64
         BUFFERPOOL BUFF16K;

CREATE REGULAR TABLESPACE SIEBEL_32K IN DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 32768 MANAGED BY DATABASE USING(
DEVICE '/dev/rvpath0cont07lv'  65536,
DEVICE '/dev/rvpath0cont08lv'  65536
         )EXTENTSIZE 32
         PREFETCHSIZE 64
         BUFFERPOOL BUFF32K;
```

A.2 Database scripts for Windows

This section provides sample DB2 database creation and configuration scripts for a Windows operating system.

A.2.1 Setting DB UDB registry variables

```
db2set DB2_HASH_JOIN=NO
db2set DB2_CORRELATED_PREDICATES=YES
db2set DB2_PIPELINED_PLANS=ON
db2set DB2_INTERESTING_KEYS=ON
db2set DB2_PARALLEL_IO=*
db2set DB2_NO_PKG_LOCK=OFF
db2set DB2_STRIPED_CONTAINERS=ON
db2set DB2ENVLIST=EXTSHM
db2set DB2COMM=tcpip
db2set DB2CODEPAGE=1208
db2set DB2AUTOSTART=TRUE

echo
echo "Registry values are now as follows"
echo
db2set -all
```

A.2.2 Database manager configuration parameters

```
db2 "update dbm cfg using SHEAPTHRES 200000"
db2 "update dbm cfg using DIR_CACHE YES"
db2 "update dbm cfg using QUERY_HEAP_SZ 16384"
db2 "update dbm cfg using ASLHEAPSZ 15"
```

```
db2 "update dbm cfg using RQRIOBLK 65535"
db2 "update dbm cfg using MON_HEAP_SZ 256"
db2 "update dbm cfg using KEEPFENCED YES"
db2 "update dbm cfg using NUM_POOLAGENTS 80"
db2 "update dbm cfg using NUM_INITAGENTS 10"
db2 "update dbm cfg using MAXAGENTS 1000 MAX_COORDAGENTS 1000"
db2 "update dbm cfg using INDEXREC RESTART"
db2 "update dbm cfg using MAX_QUERYDEGREE 1"
db2 "update dbm cfg using INTRA_PARALLEL NO"
db2 "update dbm cfg using DISCOVER DISABLE"
db2 "update dbm cfg using DISCOVER_INST DISABLE"
db2 "update dbm cfg using svcename 8100"
# turn on monitor switches
db2 "update dbm cfg using DFT_MON_BUFPOOL ON"
db2 "update dbm cfg using DFT_MON_LOCK ON"
db2 "update dbm cfg using DFT_MON_SORT ON"
db2 "update dbm cfg using DFT_MON_STMT ON"
db2 "update dbm cfg using DFT_MON_UOW ON"
db2 "update dbm cfg using DFT_MON_TABLE ON"

echo
echo "DBM cfg is now as follows:"
echo

db2 get dbm cfg
```

A.2.3 Database creation

```
db2 CREATE DATABASE %1 ON 'c:' USING CODESET UTF-8  TERRITORY US  COLLATE
USING Identity   DFT_EXTENT_SZ 32  CATALOG TABLESPACE MANAGED BY DATABASE
USING    (File 'c:\db2\fs1\%1\SYSCATSPACE\syscat.1' 25000,     File
'c:\db2\fs2\%1\SYSCATSPACE\syscat.2' 25000,    File
'c:\db2\fs3\%1\SYSCATSPACE\syscat.3' 25000,    File
'c:\db2\fs4\%1\SYSCATSPACE\syscat.4' 25000 )   EXTENTSIZE 32  PREFETCHSIZE
128
db2 get db cfg for %1
db2 connect to %1
db2 grant dbadm on database  to user siebel
db2 set event monitor db2detaildeadlock state 0
db2 drop event monitor db2detaildeadlock
db2 connect reset
```

A.2.4 Database configuration parameters

```
db2 "update db cfg for %1 using DFT_DEGREE 1"
db2 "update db cfg for %1 using DFT_QUERYOPT 3"
db2 "update db cfg for %1 using DBHEAP 7500"
db2 "update db cfg for %1 using CATALOGCACHE_SZ 5600"
```

```
db2 "update db cfg for %1 using LOGBUFSZ 256"
db2 "update db cfg for %1 using UTIL_HEAP_SZ 5000"
db2 "update db cfg for %1 using LOCKLIST 25000"
db2 "update db cfg for %1 using APP_CTL_HEAP_SZ 900"
db2 "update db cfg for %1 using SORTHEAP 1024"
db2 "update db cfg for %1 using STMTHEAP 40960"
db2 "update db cfg for %1 using PCKCACHESZ 40000"
db2 "update db cfg for %1 using STAT_HEAP_SZ 14000"
db2 "update db cfg for %1 using MAXLOCKS 20"
db2 "update db cfg for %1 using LOCKTIMEOUT 30"
db2 "update db cfg for %1 using CHNGPGS_THRESH 30"
db2 "update db cfg for %1 using INDEXSORT YES"
db2 "update db cfg for %1 using SEQDETECT YES"
db2 "update db cfg for %1 using DFT_PREFETCH_SZ 32"
db2 "update db cfg for %1 using LOGRETAIN ON"
db2 "update db cfg for %1 using USEREXIT ON"
db2 "update db cfg for %1 using MAXAPPLS 100"
db2 "update db cfg for %1 using AVG_APPLS 25"
db2 "update db cfg for %1 using MAXFILOP 500"
db2 "update db cfg for %1 using LOGFILSIZ 40000"
db2 "update db cfg for %1 using LOGPRIMARY 20"
db2 "update db cfg for %1 using LOGSECOND 107"
db2 "update db cfg for %1 using SOFTMAX 80"
db2 "update db cfg for %1 using APPLHEAPSZ 2500"
db2 "update db cfg for %1 using NUM_IOCLEANERS 4"
db2 "update db cfg for %1 using NUM_IOSERVERS 5"
db2 "update db cfg for %1 using NEWLOGPATH c:\db2\fs5\%1"
db2 "update db cfg for %1 using MIRRORLOGPATH c:\db2\fs6\%1"
db2 "update db cfg for %1 using ESTORE_SEG_SZ 65000"

db2 deactivate db %1
db2 activate db %1


echo
echo " DB cfg for $db is now as follows: "
echo
db2 get db cfg for $db
```

A.2.5 Creation of buffer pools

```
-- create_bp.bat
db2 connect to %1
db2 CREATE BUFFERPOOL "BUFF4K"  SIZE 14000 PAGESIZE 4096 NOT EXTENDED
STORAGE
db2 CREATE BUFFERPOOL "BUFF16K"  SIZE 14000 PAGESIZE 16384 NOT EXTENDED
STORAGE
```

```
db2 CREATE BUFFERPOOL "BUFF32K"  SIZE 21750 PAGESIZE 32768 NOT EXTENDED
STORAGE
db2 terminate
db2 deactivate db %1
db2 activate db %1
```

A.2.6 Creation of temporary table spaces

```
-- create_temptbs.ddl

CREATE TEMPORARY TABLESPACE TEMP4KA IN DATABASE PARTITION GROUP IBMTEMPGROUP
PAGESIZE 4096 MANAGED BY SYSTEM
         USING ('c:\db2\fs1\siebeldb\tbspaces\temp4ka1',
                'c:\db2\fs1\siebeldb\tbspaces\temp4ka2',
                'c:\db2\fs1\siebeldb\tbspaces\temp4ka3',
                'c:\db2\fs1\siebeldb\tbspaces\temp4ka4')
         EXTENTSIZE 32
         PREFETCHSIZE 128
         BUFFERPOOL BUFF4K;

CREATE TEMPORARY TABLESPACE TEMP16KA IN DATABASE PARTITION GROUP IBMTEMPGROUP
PAGESIZE 16384 MANAGED BY SYSTEM
         USING ('c:\db2\fs1\siebeldb\tbspaces\temp16ka1',
                'c:\db2\fs1\siebeldb\tbspaces\temp16ka2',
                'c:\db2\fs1\siebeldb\tbspaces\temp16ka3',
                'c:\db2\fs1\siebeldb\tbspaces\temp16ka4')
         EXTENTSIZE 32
         PREFETCHSIZE 128
         BUFFERPOOL BUFF16K;

CREATE TEMPORARY TABLESPACE TEMP32KA IN DATABASE PARTITION GROUP IBMTEMPGROUP
PAGESIZE 32768 MANAGED BY SYSTEM
         USING ('c:\db2\fs1\siebeldb\tbspaces\temp32ka1',
                'c:\db2\fs1\siebeldb\tbspaces\temp32ka2',
                'c:\db2\fs1\siebeldb\tbspaces\temp32ka3',
                'c:\db2\fs1\siebeldb\tbspaces\temp32ka4')
         EXTENTSIZE 32
         PREFETCHSIZE 128
         BUFFERPOOL BUFF32K;
```

A.2.7 Creation of Siebel table spaces

```
-- create_tbs.ddl

CREATE REGULAR TABLESPACE SIEBEL_4K IN DATABASE PARTITION GROUP IBMDEFAULTGROUP
PAGESIZE 4096 MANAGED BY DATABASE USING(
file 'c:\db2\fs1\siebeldb\tbspaces\siebel_4k.1' 524288,
file 'c:\db2\fs2\siebeldb\tbspaces\siebel_4k.2' 524288
```

```
         )EXTENTSIZE 32
         PREFETCHSIZE 64
         BUFFERPOOL BUFF4K;

CREATE REGULAR TABLESPACE SIEBEL_4KI IN DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 4096 MANAGED BY DATABASE USING(
file 'c:\db2\fs3\siebeldb\tbspaces\siebel_4ki.1' 524288,
file 'c:\db2\fs4\siebeldb\tbspaces\siebel_4ki.2' 524288
         )EXTENTSIZE 32
         PREFETCHSIZE 64
         BUFFERPOOL BUFF4K;

CREATE REGULAR TABLESPACE SIEBEL_16K IN DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 16384 MANAGED BY DATABASE USING(
file 'c:\db2\fs1\siebeldb\tbspaces\siebel_16k.1' 131072,
file 'c:\db2\fs2\siebeldb\tbspaces\siebel_16k.2' 131072
         )EXTENTSIZE 32
         PREFETCHSIZE 64
         BUFFERPOOL BUFF16K;

CREATE REGULAR TABLESPACE SIEBEL_32K IN DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 32768 MANAGED BY DATABASE USING(
file 'c:\db2\fs1\siebeldb\tbspaces\siebel_32k.1' 65536,
file 'c:\db2\fs2\siebeldb\tbspaces\siebel_32k.2' 65536
         )EXTENTSIZE 32
         PREFETCHSIZE 64
         BUFFERPOOL BUFF32K;
```

A.3 HACMP scripts

This section provides sample HACMP scripts for DB2 UDB failover, Siebel Gateway Name Server failover, Siebel Server failover, and Siebel File System failover.

A.3.1 start_db2_siebel

```
#!/bin/ksh
#################################################################
# This is the start script for HACMP application server.
# This script calls three other scripts to start db2 instance,
# start the Siebel Gateway Name Server and Siebel Server
#################################################################
su - ucsdb2 -c "/usr/es/sbin/cluster/events/db2.reboot startup"
/usr/es/sbin/cluster/events/start_gateway
```

```
/usr/es/sbin/cluster/events/start_siebel
```

A.3.2 db2.reboot

```
#!/bin/ksh
#*********************************************************************
#*                                                                   *
#* db2.reboot                                                        *
#*                                                                   *
#* This script is used to start or stop a DB2 UDB instance during    *
#* HACMP failover.                                                   *
#* This script expects only one parameter:                           *
#*  startup  - Start the DB2 UDB instance.                           *
#*  shutdown - Stop the DB2 UDB instance.                            *
#*                                                                   *
#*********************************************************************
#* Dependencies:                                                     *
#* $HOME/activate.list contains the list of database to activate     *
#*********************************************************************
#* Define local variables:
#*                                                                   *
#* ACTION     - Parameter 1 folded to lower case.                    *
#* ACTlist    - Full file name of the database activation list.      *
#* MSGfile    - Full file name of the message file.                  *
#* TMPMSGfile - Full file name of the temporary message file.        *
#* MAXlines   - Maximum number of lines in the message file.         *
#* EXITcc     - Return code.                                         *
#*                                                                   *
#*********************************************************************
ACTION=`echo $1 | tr '[A-Z]' '[a-z]'`
ACTlist=$HOME/activate.list
MSGfile=$HOME/db2.reboot.msgs
TMPForceMsg=/tmp/db2.reboot.force.msgs.`whoami`
TMPMSGfile=${MSGfile}.`date | cut -c12-19`
MAXlines=1000
EXITcc=0
Hostname=`hostname|cut -d '.' -f1`
. /$HOME/.profile
#*********************************************************************
#* Write the header message to the temporary message file.           *
#*********************************************************************
DASH="------------------------"
echo "$DASH>"`date`"<$DASH" >$TMPMSGfile
#*********************************************************************
#* If this is a DB2 UDB instance, there should be a db2profile file  *
#* in the sqllib directory.                                          *
#*********************************************************************
```

```
if [ -f $HOME/sqllib/db2profile ] ; then
   #*******************************************************************
   #* Process the request.                                            *
   #*******************************************************************
   case $ACTION in
   startup)
      #****************************************************************
      #* Process a startup request.                                   *
      #****************************************************************
      db2 list applications >/dev/null
      DB2cc=$?
      #****************************************************************
      #* If the instance is not already up, start it.                 *
      #****************************************************************
      if [ $DB2cc -le 2 ] ; then
         echo "The instance is already active." >>$TMPMSGfile
      else
         mv $HOME/sqllib/db2dump/db2diag.log
$HOME/sqllib/db2dump/db2diag.log.`d
ate +%Y%m%d%H%M`
         db2start >>$TMPMSGfile
         DB2cc=$?
         if [ $DB2cc = 0 ] ; then
            sleep 5
            if [ -f $ACTlist ] ; then
               #*********************************************************
               #* Activate all the databases identified in the        *
               #* activation list.                                    *
               #*********************************************************
               while read ACTdb
               do
                  db2 "activate database $ACTdb" >>$TMPMSGfile
               done < $ACTlist
            fi
         else
            echo "The db2start command ended with RC: "$DB2cc"." \
                >>$TMPMSGfile
            EXITcc=DB2cc
         fi
      fi
      ;;
   shutdown)
      #****************************************************************
      #* Process a shutdown request.                                  *
      #****************************************************************
      db2 list applications >>$TMPMSGfile
      DB2cc=$?
         # We will deactivate db prior to the forcing of applications
         if [ -f $ACTlist ] ; then
```

```
            #*************************************************************
            #* Deactivate all the databases identified in the         *
            #* activation list.                                       *
            #*************************************************************
            while read ACTdb
            do
               db2 "deactivate database $ACTdb" >>$TMPMSGfile
            done < $ACTlist
         fi

      #*****************************************************************
      #* If there are applications connected to the instance, force   *
      #* the applications before stopping the instance.               *
      #*****************************************************************
Counter=0
   while [ $Counter -lt 30 ]
      do
         Counter=`expr $Counter + 1`
         sleep 5
         db2 force applications all >$TMPForceMsg
         db2 list applications >/dev/null
         RC=$?                        # Zero indicates still have
connections
         if [ $RC -ne 0 ]
            then
                cat $TMPForceMsg>>$TMPMSGfile
                echo "Force of applications successful in $Counter
attempts">>$
TMPMSGfile
                Counter=`expr $Counter + 30`
         fi
      done
      #*****************************************************************
      #* If the instance is up, issue the db2stop command.            *
      #*****************************************************************
      if [ $DB2cc -le 2 ] ; then
         # Deactivate the database after the force.
         # We should be able to remove this code when version 5.0 and 2.1
         # databases are gone
         if [ -f $ACTlist ] ; then
            #*************************************************************
            #* Deactivate all the databases identified in the         *
            #* activation list.                                       *
            #*************************************************************
            while read ACTdb
            do
               db2 "deactivate database $ACTdb" >>$TMPMSGfile
            done < $ACTlist
         fi
```

```
         db2stop >>$TMPMSGfile
         DB2cc=$?
         if [ $DB2cc = 0 ] ; then
            sleep 5
            echo "The db2stop command was successful. "`date` \
                 >>$TMPMSGfile
         else
            echo "The db2stop command ended with RC: "$DB2cc". "`date` \
                 >>$TMPMSGfile
            echo "Output of list applications show detail:" >>$TMPMSGfile
            db2 list applications show detail >>$TMPMSGfile
            echo "Issuing: 'db2stop force' at "`date` >>$TMPMSGfile
            db2stop force>>$TMPMSGfile
            DB2cc=$?
            if [ $DB2cc = 0 ] ; then
               sleep 5
               echo "The db2stop FORCE  command was successful. "`date` \
                   >>$TMPMSGfile
#               mail -s "db2stop force SUCCESSFUL on $Hostname"
aixdba@us.ibm.co
m <$TMPMSGfile
            else
               echo "The db2stop FORCE command failed with RC: "$DB2cc".
"`date`
 >>$TMPMSGfile
               EXITcc=DB2cc
               mail -s "db2stop force FAILED on $Hostname"
aixdba@us.ibm.com <$T
MPMSGfile
            fi
         fi
      else
         echo "The instance is already inactive." >>$TMPMSGfile
      fi
      ;;
   *)
      #*****************************************************************
      #* Invalid request was made.                                     *
      #*****************************************************************
      echo Invalid action requested: $1. >>$TMPMSGfile
      EXITcc=4
      ;;
   esac
else
   echo "The db2profile file does not exist in the sqllib directory." \
        >>$TMPMSGfile
   EXITcc=4
fi
#*********************************************************************
```

```
#* If an error was detected, mail the temporary message file to the   *
#* appropriate support addresses.                                     *
#**********************************************************************
if [ $EXITcc != 0 ] ; then
   if [ -n "$SYSaddr" ] ; then
#     mail -s "Host: `hostname -s`  Instance: $DB2INSTANCE" \
           $SYSaddr <$TMPMSGfile
   fi
fi
#**********************************************************************
#* Copy the temporary message file to the message file and delete the *
#* temporary message file.  If requested, restrict the size of the    *
#* message file to the specified maximum number of lines.             *
#**********************************************************************
cat $TMPMSGfile >>$MSGfile
if [ -n "$MAXlines" ] ; then
   tail -$MAXlines $MSGfile >$TMPMSGfile
   cp $TMPMSGfile $MSGfile
fi
rm $TMPMSGfile
exit $EXITcc
```

A.3.3 start_gateway

```
#!/bin/ksh
##########################################################################
################
## Script is used to start the Siebel Gateway Nameserver during HACMP
failover.
## script checks for  siebns.dat.
## Starts the gateway using the start_ns command.
##########################################################################
################
echo "Performing slibclean\n"
/usr/sbin/slibclean

if [[ ! -f /siebel/gtwysrvr/sys/siebns.dat ]]; then
echo "Error: Could not find siebns.dat file\n"
echo "Error: Gateway is not going to started\n"
exit 1
fi
echo "Starting the Gateway\n"

su - siebsupp -c ". /siebel/gtwysrvr/siebenv.sh;
/siebel/gtwysrvr/bin/start_ns"
sleep 30
```

```
ps -ef | grep "siebsvc -s gtwyns" | grep -v grep
if (( $? != 0 )); then
    echo "Error: Could not start the gateway\n"
else
    echo " GATEWAY STARTED \n"
                 exit 0
fi
```

A.3.4 start_siebel

```
#!/bin/ksh
##########################################################################
################
## Script is used to start the siebel server during HACMP failover.
## Starts the gateway using the start_server -e enterprise server command.
## The script is for AIX servers with one siebel server.
##########################################################################
################
enterprise=`ls /siebel/siebsrvr/enterprises`
server_name=`ls /siebel/siebsrvr/enterprises/$enterprise`

echo "Running slibclean\n"
/usr/sbin/slibclean

echo  "\nChecking to Make sure gateway server is up\n"
su - siebsupp -c ". /siebel/siebsrvr/siebenv.sh;
/siebel/siebsrvr/bin/srvredit -q -g \$SIEBEL_GATEWAY -e none -z -c
'\$Gateway.VersionString'" > /dev/null
if (( $? != 0 )); then
      echo "\nError: GATEWAY IS NOT UP\n"
      echo "\nError: Exiting the script. Siebel is NOT started\n"
      exit 1
fi

echo "\n\nStarting the siebel server\n"
su - siebsupp -c ". /siebel/siebsrvr/siebenv.sh;
/siebel/siebsrvr/bin/start_server -e $enterprise $server_name"
sleep 60
ps -ef | grep "siebsvc -s siebsrvr" | grep -v grep
if (( $? != 0 )); then
    echo "\nError: Could not start the siebel server $server_name\n"
else
    echo "\nSiebel Server $server_name is UP \n"
    exit 0
fi
```

A.3.5 stop_db2_siebel

```
#!/bin/ksh
##############################################################
# This is the stop script for HACMP application server.
# This script calls three other scripts to stop db2 instance,
# stop the Siebel Gateway Name Server and Siebel Server
##############################################################
su - ucsdb2 -c "/usr/es/sbin/cluster/events/db2.reboot shutdown"
/usr/es/sbin/cluster/events/stop_gateway
/usr/es/sbin/cluster/events/stop_siebel
```

A.3.6 stop_gateway

```
#!/bin/ksh
#########################################################################
################
## Script is used to stop the Siebel Gateway Nameserver during HACMP
## script makes a backup of the siebns.dat
## Tries to stop the gateway with stop_ns if this does not work then
## Kill the siebsvc processes
#########################################################################
################
# check if /siebel/gtwysrvr/sys directory exits to copy the siebns.dat file
if [[ -d /siebel/gtwysrvr/sys ]]; then
        echo "Stopping SIEBEL GW, please wait....\n"
        echo "Making a backup copy of siebns.dat\n"
        cp -p /siebel/gtwysrvr/sys/siebns.dat
/siebel/gtwysrvr/sys/"siebns.dat.`date`"
fi
echo "Stopping the gateway process\n"
su - siebsupp -c ".
/siebel/gtwysrvr/siebenv.sh;/siebel/gtwysrvr/bin/stop_ns"
ps -ef | grep "siebsvc -s gtwyns" | grep -v grep > /dev/null
if [[ $? -eq 0 ]]; then
       echo " Gateway did not stop with stop_ns.\n"
       svc_process=`ps -ef | grep "siebsvc -s gtwyns" | grep -v grep | awk
'{print $2}'`
       echo "Killing the Gateway process with kill -9\n"
       kill -9 $svc_process
       echo " GATEWAY STOPPED \n"
       exit 0
else
       echo " GATEWAY STOPPED \n"
       exit 0
fi

echo "Cleaning up leftover IPCs\n"
```

```
for i in `ipcs -s | grep siebsupp | awk '{print $2}'`
do
        ipcrm -s $i 2>/dev/null
done
for i in `ipcs -m | grep siebsupp | awk '{print $2}'`
do
        ipcrm -m $i 2>/dev/null
done
```

A.3.7 stop_siebel

```
#!/bin/ksh
##########################################################################
################
## Script is used to stop the siebel during HACMP failover
## Script tries to stop the siebel with stop_server if it does not work
then
## Kill the siebsvc processes
##########################################################################
################
# Get the name of the enterprise and server name
enterprise=`ls /siebel/siebsrvr/enterprises`
server_name=`ls /siebel/siebsrvr/enterprises/$enterprise`

echo "Stopping the siebel Server\n"
su - siebsupp -c ".
/siebel/siebsrvr/siebenv.sh;/siebel/siebsrvr/bin/stop_server -e $enterprise
$server_name"
sleep 60
ps -ef | grep "siebsvc -s siebsrvr" | grep -v grep > /dev/null
if [[ $? -eq 0 ]]; then
        echo "Siebel Server did not stop with stop_server command. Trying
reset_server command\n"
        su - siebsupp -c ".
/siebel/siebsrvr/siebenv.sh;/siebel/siebsrvr/bin/reset_server -e
$enterprise $server_name"
        ps -ef | grep -E "siebproc|
siebmtshmw|siebmtsh|siebsess|siebprocmw|siebsh|siebsvc -s siebsrvr" | grep
-v grep > /dev/null
        if [[ $? -eq 0 ]]; then
                echo "Siebel Server did not stop with reset_server command.
Killing the processes\n"
                for i in `ps -ef | grep -E "siebproc|
siebmtshmw|siebmtsh|siebsess|siebprocmw|siebsh| siebsvc -s siebsrvr" | grep
-v grep | awk '{print $2}'`
                do
                        echo "\nkilling the pid $i\n"
                        kill -9 $i 2> /dev/null
```

```
                done
                if [[ -f /siebel/siebsrvr/admin/*.shm ]]; then
                           rm -rf /siebel/siebsrvr/admin/*.shm
                fi
                if [[ -f /siebel/siebsrvr/sys/osdf.$enterprise.$server_name
]]; then
                           rm -rf
/siebel/siebsrvr/sys/osdf.$enterprise.$server_name
                fi
        else
                echo "Siebel server $server_name stopped with reset_server
command\n"
        fi
else
         echo " Siebel Server $server_name STOPPED \n"
         exit 0
fi

echo "Removing any srvrmgr processes\n"
ps -ef | grep srvrmgr | grep -v grep > /dev/null
if [[ $? -eq 0 ]]; then
    for i in `ps -ef | grep srvrmgr | grep -v grep | awk '{print $2}'`
    do
           kill -9 $i  2> /dev/null
    done
fi

echo "Cleaning up leftover IPCs\n"

for i in `ipcs -s | grep siebsupp | awk '{print $2}'`
do
        ipcrm -s $i 2>/dev/null
done
for i in `ipcs -m | grep siebsupp | awk '{print $2}'`
do
        ipcrm -m $i 2>/dev/null
done
```

A.4 Siebel files

This section contains Siebel load balance configuration file `lbconfig.txt` and `eapps.cfg`.

A.4.1 lbconfig.txt

```
#This is the load balance configuration file generated by the Siebel
srvrmgr "generate lbconfig" command.
#It contains two sections. Section one contains load balancing rules to be
used by Siebel session manager.
#Section two is intended for 3rd party load balancers. Before modifying the
content of this file please
#read the chapter on SWSE configuration in the Siebel Bookshelf.

#Section one -- Session Manager Rules:
VirtualServer=1:Atlantic:2321;2:KANAGA:2321;


******************************
#Section two -- 3rd Party Load Balancer Rules

#Component Rules:
/siebaix/loyaltyObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/loyaltyscwObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eloyaltyObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/CRAObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eCommunicationsObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eMediaObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eEnergyObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eEnergyOGCObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eCommWirelessObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eChannelCMEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eSalesCMEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eCustomerCMEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/SMObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eMarketObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eEventsObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/ERMObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/ERMAdminObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/ERMEmbObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eTrainingObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/ePharmaObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eClinicalObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eMedicalObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eSitesClinicalObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eProfessionalPharmaObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eConsumerPharmaObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/ServiceCEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/SalesCEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/UCMObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/PSCcObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/PSeServiceObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eProdCfgObjMgr_enu=KANAGA:2321;Atlantic:2321;
```

```
/siebaix/eSalesObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eCustomerObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/edealerObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/edealerscwObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eautoObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/PManagerObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eChannelObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/WirelessSalesObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/WirelessServiceObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/WirelesseChannelObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/WirelesseServiceObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eConsumerSectorObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eRetailObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eChannelCGObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eConsumerObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eHospitalityObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/sismeObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/EAIObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/CustomAppObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/SCCObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/eServiceObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/ePharmaCEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/MedicalCEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/CGCEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/SIASalesCEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/SIAServiceCEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/SSEObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/SMCObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/SFSObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/htimprmObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/htimObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/FINSObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/FINSConsoleObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeSalesObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeBrokerageObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeBankingObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/INSeServiceObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeChannelObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeCustomerObjMgr_enu=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeEnrollmentObjMgr_enu=KANAGA:2321;Atlantic:2321;

#Server Rules:
*/!1.*=Atlantic:2321;
*/!2.*=KANAGA:2321;

#Round Robin Rules:
/siebaix/loyaltyObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/loyaltyscwObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eloyaltyObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
```

```
/siebaix/CRAObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eCommunicationsObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eMediaObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eEnergyObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eEnergyOGCObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eCommWirelessObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eChannelCMEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eSalesCMEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eCustomerCMEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/SMObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eMarketObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eEventsObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/ERMObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/ERMAdminObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/ERMEmbObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eTrainingObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/ePharmaObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eClinicalObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eMedicalObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eSitesClinicalObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eProfessionalPharmaObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eConsumerPharmaObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/ServiceCEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/SalesCEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/UCMObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/PSCcObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/PSeServiceObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eProdCfgObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eSalesObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eCustomerObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/edealerObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/edealerscwObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eautoObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/PManagerObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eChannelObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/WirelessSalesObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/WirelessServiceObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/WirelesseChannelObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/WirelesseServiceObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eConsumerSectorObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eRetailObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eChannelCGObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eConsumerObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eHospitalityObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/sismeObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/EAIObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/CustomAppObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/SCCObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/eServiceObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
```

```
/siebaix/ePharmaCEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/MedicalCEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/CGCEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/SIASalesCEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/SIAServiceCEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/SSEObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/SMCObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/SFSObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/htimprmObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/htimObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/FINSObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/FINSConsoleObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeSalesObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeBrokerageObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeBankingObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/INSeServiceObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeChannelObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeCustomerObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
/siebaix/FINSeEnrollmentObjMgr_enu/RR=KANAGA:2321;Atlantic:2321;
```

A.4.2 eapps.cfg

A small part of `eapps.cfg` is shown below:

```
[include]
eapps_sia.cfg
eapps_fins.cfg
eapps_sis.cfg

[swe]
Language      = enu
Log           = errors
LogDirectory  = /siebel/swse/log
ClientRootDir = /siebel/swse
SessionMonitor  = FALSE
AllowStats      = TRUE
LogSegmentSize  = 0
LogMaxSegments  = 0
DisableNagle    = FALSE

[ConnMgmt]
EnableVirtualHosts = true
VirtualHostsFile   = /siebel/swse/admin/lbconfig.txt
CertFileName       = $(CertFileName)
CACertFileName     = $(CaCertFileName)
KeyFileName        = $(KeyFileName)
KeyFilePassword    = $(KeyFilePassword)
PeerAuth           = $(PeerAuth)
PeerCertValidation = $(PeerCertValidation)
```

```
[defaults]
EncryptedPassword = TRUE
AnonUserName  = sadmin
AnonPassword  = 6TvuUFl1qg4BAACE2A==
StatsPage     = _stats.swe
HTTPPort      = 80
HTTPSPort     = 443
EnableFQDN    = FALSE
FQDN          = CHANGE_ME
DoCompression   = TRUE
GuestSessionTimeout = 300
SessionTimeout      = 900

[/callcenter_enu]
ConnectString =
siebel.tcpip.none.none://VirtualServer/siebaix/SCCObjMgr_enu
WebPublicRootDir = /siebel/swse/public/enu
WebUpdatePassword = fgS6wDKlAWkBAAD/EA==
```

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 371. Note that some of the documents referenced here may be available in softcopy only.

- *Database Performance Tuning on AIX,* SG24-5511-01
- *DB2 UDB V8 and WebSphere V5 Performance Tuning and Operation Guide,* SG24-7068
- *DB2 UDB ESE V8 Performance Guide for High Performance OLTP and BI*, SG24-6432
- *Siebel 7 Using DB2 UDB V7.x Planning & Installation Guide for AIX/Win2k*, SG24-6415
- *DB2 UDB V8.2 on the Windows Environment,* SG24-7102
- *IBM eServer Certification Study Guide: AIX 5L Installation and System Recovery,* SG24-6183-01
- *HACMP for AIX V5.X Certification Study Guide,* SG24-6375
- *IBM eServer Certification Study Guide: AIX 5L Problem Determination Tools and Techniques,* SG24-6185-01
- *IBM eServer Certification Study Guide: pSeries AIX System Support,* SG24-6199
- *IBM eServer Certification Study Guide: pSeries AIX System Administration,* SG24-6191
- *AIX 5L and Windows 2000: Side by Side,* SG24-4784-02
- *AIX 5L Practical Performance Tools and Tuning Guide,* SG24-6478

Other publications

These publications are also relevant as further information sources:

IBM DB2

- IBM DB2 UDB documentation: *What's New V8,* SC09-4848-01
- IBM DB2 UDB documentation: *Administration Guide: Implementation V8*, SC09-4820-01
- IBM DB2 UDB documentation: *Administration Guide: Performance V8*, SC09-4821-01
- IBM DB2 UDB documentation: *Administration Guide: Planning V8*, SC09-4822-01
- IBM DB2 UDB documentation: *Application Development Guide: Building and Running Applications V8*, SC09-4825-01
- IBM DB2 UDB documentation: *Application Development Guide: Programming Client Applications V8*, SC09-4826-01
- IBM DB2 UDB documentation: *Application Development Guide: Programming Server Applications V8*, SC09-4827-01
- IBM DB2 UDB documentation: *Call Level Interface Guide and Reference, Volume 1*, *V8*, SC09-4849-01
- IBM DB2 UDB documentation: *Call Level Interface Guide and Reference, Volume 2, V8*, SC09-4850-01
- IBM DB2 UDB documentation: *Command Reference V8*, SC09-4828-01
- IBM DB2 UDB documentation: *Data Movement Utilities Guide and Reference V8*, SC09-4830-01
- IBM DB2 UDB documentation: *Data Recovery and High Availability Guide and Reference V8*, SC09-4831-01
- IBM DB2 UDB documentation: *Guide to GUI Tools for Administration and Development*, SC09-4851-01
- IBM DB2 UDB documentation: *Installation and Configuration Supplement V8*, GC09-4837-01
- IBM DB2 UDB documentation: *Quick Beginnings for DB2 Clients V8*, GC09-4832-01
- IBM DB2 UDB documentation: *Quick Beginnings for DB2 Servers V8*, GC09-4836-01
- IBM DB2 UDB documentation: *Replication and Event Publishing Guide and Reference*, SC18-7568

- IBM DB2 UDB documentation: *SQL Reference, Volume 1, V8,* SC09-4844-01
- IBM DB2 UDB documentation: *SQL Reference, Volume 2, V8,* SC09-4845-01
- IBM DB2 UDB documentation: *System Monitor Guide and Reference V8,* SC09-4847-01
- IBM DB2 UDB documentation: *Data Warehouse Center Application Integration Guide Version 8 Release 1*, SC27-1124-01-01
- IBM DB2 UDB documentation: *DB2 XML Extender Administration and Programming Guide Version 8 Release 1*, SC27-1234-01
- IBM DB2 UDB documentation: *Federated Systems PIC Guide Version 8 Release 1*, GC27-1224-01

IBM AIX

- AIX operating system library:

  http://www-1.ibm.com/servers/aix/library/index.html
- High Availability:

  http://www-1.ibm.com/servers/eserver/pseries/ha/
- IBM AIX 5L operating system:

  http://www-1.ibm.com/servers/aix/
- HACMP library:

  http://www.ibm.com/servers/eserver/pseries/library/hacmp_docs.html

Siebel

- Siebel Support Web: *Siebel Systems Requirements and Supported Platforms 7.8*
- Siebel Bookshelf: *Siebel Datawarehouse Installation and Administration Guide*
- Siebel Bookshelf: *Deployment Planning Guide Version 7.8*
- Siebel Bookshelf: *Siebel Developer's Reference Version 7.8*
- Siebel Bookshelf: *Developing and Deploying Siebel eBusiness Applications Version 7.7*
- Siebel Bookshelf: *Going Live with Siebel Business Applications*
- Siebel Bookshelf: *Installation Guide for Microsoft Windows: Servers, Mobile Web Clients, Tools*
- Siebel Bookshelf: *Installation Guide for Unix: Servers, Mobile Web Clients, Tools*
- Siebel Bookshelf: S*ecurity Guide for Siebel Business Applications*

- Siebel Bookshelf: *Siebel Data Quality Administration Guide*
- Siebel Bookshelf: *Siebel Enterprise Integration Manager Administration Guide*
- Siebel Bookshelf: *Siebel Object Interfaces Reference*
- Siebel Bookshelf: *System Administration Guide*
- Siebel Bookshelf: *System Monitoring and Diagnostics Guide for Siebel Business Applications*
- Siebel Tech Note: 261: *Siebel V7 on Microsoft Cluster Server*

Online resources

These Web sites are also relevant as further information sources:

DB2

- Database and Data Management

  http://www.ibm.com/software/data/

  http://www.ibm.com/software/data/highlights/db2tco.html
- DB2 Universal Database

  http://www.ibm.com/software/data/db2/udb/

  http://www.ibm.com/db2/v8

  http://publib.boulder.ibm.com/infocenter/db2help/index.jsp

  http://www.ibm.com/support/docview.wss?rs=71&context=SSEPGG&uid=swg21165572&loc=en_US&cs=utf-8&lang=en+en

  http://www.ibm.com/software/data/db2/udb/support/manualsv8.html
- DB2 developerWorks

  http://www.ibm.com/developerworks/db2/
- DB2 Universal Database V8 Application Development

  http://www.ibm.com/software/data/db2/udb/ad
- DB2 Technical Support

  http://www.ibm.com/software/data/db2/udb/support/index.html
- DB2 Extenders

  http://www.ibm.com/software/data/db2/extenders/

- IBM Manuals for Data Management Products

  http://www.ibm.com/software/data/technical/BOOK/

- DB2 Migrate NOW!

  http://www.ibm.com/db2/migration

Siebel

- Siebel support Web site

  http://supportweb.siebel.com

- Siebel Global Services

  http://www.siebel.com/crm/global-services/software-solutions.shtm

- Siebel Technical Support

  http://www.siebel.com/training-support/index.shtm

Windows

- Windows 2000 Server Home

  http://www.microsoft.com/windows2000/default.asp

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

E

F

G

H

O

P

Q

R

S

T

U

V

W

X

Siebel 7.8 with IBM DB2 UDB V8.2 Handbook

(0.5” spine)
0.475”<->0.875”
250 <-> 459 pages

Siebel 7.8 with IBM DB2 UDB V8.2 Handbook

Step-by-step installation and configuration procedures

Database administration, monitoring, and tuning methods

High availability options and setup steps

This IBM Redbook delivers details about DB2 UDB V8.2 on Siebel 7.8. It outlines the partnership between Siebel Systems and IBM and the benefits of using DB2 UDB to support the Siebel Enterprise. The most commonly used components of the Siebel Enterprise and the DB2 UDB architecture are described.

We provide the planning considerations for running DB2 UDB in Siebel environment. The step-by-step installation and configuration details are followed. We then describe information on methods to populate and maintain data in Siebel tables including data archival techniques and information on ensuring data integrity and data quality.

The database administration, monitoring, and tuning tools provided by DB2 UDB and operating systems are discussed and the tool usage provided. The book also provides in-depth discussion on high availability and disaster recovery options and setup procedure for a Siebel/DB2 UDB environment.

Finally, the book provides information about the components of Siebel Analytics and where these components fit in the overall scheme with Siebel Enterprise.

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

SG24-6766-00 ISBN 0738492817