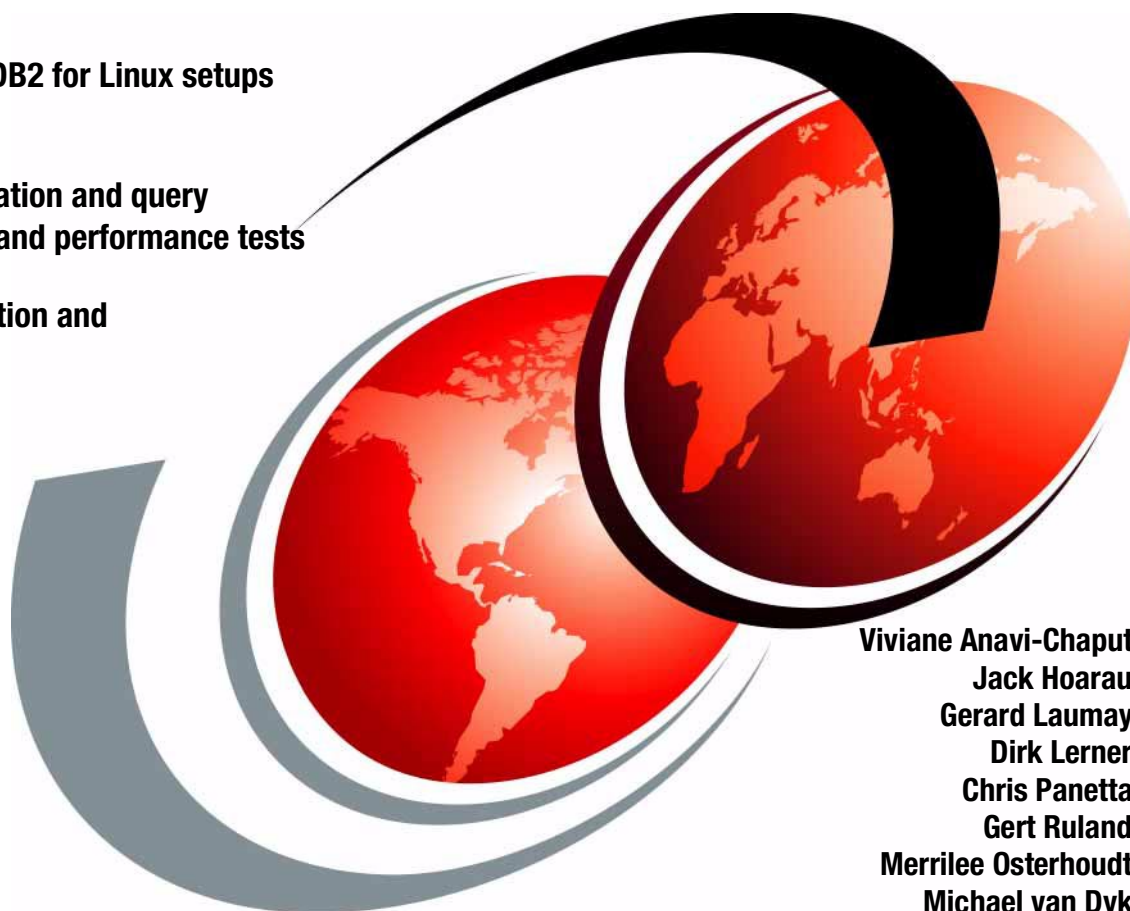


e-Business Intelligence: Leveraging DB2 for Linux on S/390

Linux and DB2 for Linux setups
on S/390

Data population and query
scalability and performance tests

Administration and
monitoring



Viviane Anavi-Chaput
Jack Hoarau
Gerard Laumay
Dirk Lerner
Chris Panetta
Gert Ruland
Merrilee Osterhoudt
Michael van Dyk



International Technical Support Organization

**e-Business Intelligence: Leveraging DB2 for Linux
on S/390**

July 2001

Take Note! Before using this information and the product it supports, be sure to read the general information in “Special notices” on page 205.

First Edition (July 2001)

This edition applies to DB2 for Linux on S/390 V7.1 for use with Linux SuSE V7.0, and DB2 for OS/390 V6.1 for use with the OS/390 V2R8 operating system.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2001. All rights reserved.

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

Contents

Preface	ix
The team that wrote this redbook	ix
Comments welcome	xi
 Chapter 1. Overview of Linux	1
1.1 Linux on S/390	1
1.2 Why Linux on zSeries-S/390?	2
1.2.1 Expanded application portfolio	2
1.2.2 Cost savings	3
1.2.3 Server consolidation and faster time to market	3
1.3 Configurations for Linux on zSeries-S/390	5
 Chapter 2. BI consolidations using DB2 for Linux on S/390	9
2.1 Disadvantages of distributed data marts	9
2.2 Advantages of consolidating data marts	11
2.2.1 Rich end-user functionality	11
2.2.2 Faster time to market	11
2.2.3 Reduced cost of ownership	11
2.2.4 More data marts, more data	13
2.3 Consolidation of networks and gateway servers	15
2.3.1 High speed Joins	16
2.4 Web-enablement of the BI infrastructure	16
2.5 Consolidation of BI application servers and middleware	16
 Chapter 3. Introducing the test environment	19
3.1 Test configurations	21
3.2 Product implementations	22
3.3 Scalability and performance tests	22
3.4 Administration and monitoring	23
 Chapter 4. Setting up Linux on S/390	25
4.1 Base installation of SuSE Linux 7.0 in a native LPAR	26
4.1.1 Acquiring network information	26
4.1.2 Creating the bootstrap tape	27
4.1.3 IPLing the bootstrap tape	29
4.1.4 Initializing network connections	30
4.1.5 Installation tasks performed from a Linux workstation	31
4.2 Invoking YaST and configuring	33
4.2.1 Invoking YaST	33
4.2.2 Performing cleanup and rebooting	46
4.3 Installing SuSE Linux 7.0 under VIF	48

4.3.1	Preparing for installation.	50
4.3.2	Installing VIF on S/390 DASD.	52
4.3.3	Loading VIF Hypervisor from DASD	52
4.3.4	Initializing the Master Linux in RAMdisk	53
4.3.5	Installing the Master Linux on VIF partition 201	55
4.3.6	Setting partition 201 as boot for Master Linux.	60
4.3.7	Making access to VIF commands more convenient.	61
4.4	Implementing the first Linux image under VIF.	62
4.4.1	Allocating more disk space to the VIF system.	62
4.4.2	Creating the new guest image LINUX1.	64
4.4.3	Adding a network connection to Linux image LINUX1.	65
4.4.4	Checking Linux image LINUX1 settings	65
4.4.5	Starting and installing the Linux image	66
4.4.6	Cloning a Linux image from LINUX1	69
4.4.7	Hints and tips	72
4.5	Adding a network connection device to an active Linux	72
4.5.1	Network configuration files under SuSE Linux for S/390	73
4.5.2	Adding a CTC/escon connection.	73
4.5.3	Adding an OSA Express GbE device connection	75
Chapter 5.	Implementing DB2 for Linux on S/390.	79
5.1	DB2 for Linux installation on S/390.	79
5.1.1	Hardware and software requirements	79
5.1.2	Defining DB2 users.	80
5.1.3	Customizing SuSE Linux for DB2 support.	81
5.1.4	Transferring the DB2 archive file to the Linux system on S/390	81
5.1.5	Installing DB2 for Linux.	82
5.1.6	Post-installation	92
5.1.7	Checking the DB2 for Linux installation	93
5.2	DB2 Connect for Linux setup on S/390.	94
5.2.1	Checking DB2 for OS/390 settings	94
5.2.2	DB2 Connect configuration parameters	94
5.2.3	Cataloging the TCP/IP node	95
5.2.4	Cataloging the host database	96
5.2.5	Cataloging the database.	97
5.2.6	Binding utilities and applications to the database	98
5.2.7	Testing access to the host	99
5.3	DB2 Client connectivity	99
5.3.1	DB2 Run-Time client	99
5.3.2	DB2 Administration Client.	103
5.3.3	QMF for Windows accessing DB2 for Linux on S/390	104
5.4	DB2 for Linux on S/390 performance	105
5.4.1	Linux system performance	106

5.4.2 DB2 subsystem performance	106
5.5 DB2 Connect performance	108
5.6 DB2 client performance	110
Chapter 6. Building the data mart with DB2 for Linux	111
6.1 Creating a new DB2 instance	114
6.2 Creating a new database	114
6.3 Buffer pools	120
6.4 Containers	121
6.5 Creating table spaces	121
6.6 Creating tables	125
6.7 Creating indexes	126
6.8 Recommendations for database objects	126
Chapter 7. Scalability and performance tests for data population ..	129
7.1 Test environment	129
7.2 Single distributed SQL statement	131
7.2.1 Federated database	131
7.2.2 Distributed SQL scenario	132
7.3 Select/Insert SQL statements in a C program	134
7.4 DB2 DSNTIAUL and DB2 Load	137
7.5 DB2 High Performance Unload (HPU)	139
7.6 DB2 Export and Import	145
7.7 DB2 Export and Load	148
7.8 Data population performance summary	151
7.9 Other data population tools	152
7.9.1 DB2 DataPropagator	152
7.9.2 DB2 Data Warehouse Center	153
7.9.3 DB2 AutoLoader	155
7.9.4 DB2 for OS/390 V7.1 Unload	155
7.9.5 DB2 DataJoiner	157
Chapter 8. Scalability and performance tests for queries	159
8.1 Processor-intensive queries	161
8.2 I/O-intensive queries	163
8.2.1 I/O-intensive query - no rows returned	163
8.2.2 I/O-intensive query - 40,000 rows returned	165
8.2.3 I/O-intensive query - 40,000 rows returned with Sort	166
8.3 Short queries	168
8.4 Trivial queries	169
Chapter 9. Administrating and monitoring the BI environment.	171
9.1 DB2 administration	171
9.2 DB2 performance tools	173

9.2.1 SQL Explain	173
9.2.2 Snapshot monitoring	174
9.2.3 Event monitors	175
9.2.4 DB2 Performance Monitor	176
9.2.5 DB2 Governor	176
9.2.6 Wizards	176
9.3 Linux performance tools	177
9.3.1 vmstat	177
9.3.2 mpstat	178
9.4 DB2 security administration	179
Appendix A. Teraplex LPAR resource matrix.	181
Appendix B. DB2 for Linux scripts to create the data mart	183
B.1 Creating the database.	183
B.2 Creating the table spaces	183
B.3 Creating the tables	183
B.4 Creating the DB2 buffer pools.	184
Appendix C. Data population scripts	185
C.1 Distributed SQL statement	185
C.2 C program source code with Select/Insert SQL statements	187
C.2.1 C program source code	187
C.2.2 Makefile used to compile the C program	192
C.2.3 Script to run the C program	194
C.3 DB2 DSNTIAUL program and DB2 Load	195
C.3.1 JCL for DB2 DSNTIAUL utility	195
C.3.2 DB2 Load script	196
C.4 DB2 High Performance Unload utility	196
C.5 DB2 Export/Import script.	197
C.6 DB2 Export/Load scripts.	198
Appendix D. Table definitions and query scripts	201
D.1 Table definitions	201
D.2 Q01 I/O-intensive query - no rows returned	202
D.3 Q02 Processor-intensive query	202
D.4 Q03 Trivial query	203
D.5 Q04 Short query and Q05 Short query with table in memory.	203
D.6 Q06 I/O-intensive query - 40,000 rows returned.	203
D.7 Q07 I/O-intensive query - 40,000 rows returned with Sort	203

Appendix E. Special notices	205
Appendix F. Related publications	209
F.1 IBM Redbooks.....	209
F.2 IBM Redbooks collections.....	209
F.3 Other resources	209
F.4 Referenced Web sites.....	210
How to get IBM Redbooks	213
IBM Redbooks fax order form	214
Index	215
IBM Redbooks review	219

Preface

This IBM Redbook discusses leveraging DB2 for Linux on S/390 for new e-BI opportunities. It shows the possible Linux configurations on S/390 and describes the advantages of consolidating UNIX data mart farms on unique IBM e-servers using DB2 for Linux on S/390.

The book describes how to set up DB2 for Linux in a Linux LPAR and how to customize the database connections (using DB2 Connect for Linux) to the DB2 for OS/390 data warehouse residing in an OS/390 LPAR on the same machine.

It also includes early scalability and performance tests on queries and data population techniques run at the IBM S/390 Teraplex Center in Poughkeepsie, New York.

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 Poughkeepsie Center.

Viviane Anavi-Chaput is a Senior IT Specialist for BI and DB2 at the IBM International Technical Support Organization, Poughkeepsie Center. She writes extensively, teaches worldwide, and presents at international conferences on all areas of Business Intelligence and DB2 for OS/390. Before joining the ITSO in 1999, Viviane was a Senior Data Management Consultant at IBM Europe, France. She was also an ITSO Specialist for DB2 at the San Jose Center from 1990 to 1994.

Jack Hoarau is a zSeries Linux architect at IBM Products and Solution Support Center (PSSC) in Montpellier, France. Jack has been with IBM for many years supporting large customers in areas of zSeries, Linux, and data management.

Gerard Laumay is a zSeries Business Intelligence Architect at the Products and Solutions Support Center (PSSC) in Montpellier, France. Part of the technical field support organization group, Gerard has been with IBM for 15 years, working as an architect on S/390 zSeries environments for large French customers.

Dirk Lerner is a database developer and supporter who works for SuSE, Germany. He has three years of experience in administration, designing and

developing databases. He holds a degree in general computer science from the Scientific-Technical Academy Prof. Dr. Gruebler, Isny i.A.

Chris Panetta is a performance consultant working at the IBM S/390 Teraplex Integration Center, New York. He specializes in zSeries performance, focusing on Linux on S/390 and BI strategy on the zSeries-S/390 platform. Chris also designs Linux and BI proof of concepts to support large customer requirements on performance and scalability issues.

Gert Ruland has worked for IBM since 1981 as a system engineer and a sales representative. Until late 2000, he was on assignment at the New Technology Center in Montpellier, France, dealing with Business Intelligence Applications on zSeries. Currently he is a member of the International SAP IBM Competence Center in Walldorf, Germany, dealing with SAP Applications on zSeries.

Merrilee Osterhoudt has been with IBM for 18 years. She started her career in the MVS design, responsible for development and design of operating system components, coordination of new releases and management of Early Support Programs. In 1992, Merrilee joined the IBM Server Group S/390 Business Intelligence solution team involved in the development, test and support of data warehouse solutions on S/390. She is currently the zSeries-S/390 BI Technical Support Leader, responsible for the S/390 Business Intelligence technical support strategy and implementation worldwide.

Michael van Dyk is a database administrator who works for Debis IT Services, South Africa. He has eight years of experience with DB2 OS/390 administration and two years experience with DB2 UDB administration, designing, developing and supporting databases.

Thanks to the following people for their invaluable contributions to this project:

Michael MacIsaac and Richard Conway
International Technical Support Organization, Poughkeepsie Center

Leticia Cruz, Darren Swank, and Michael Tebolt
IBM S/390 Teraplex Integration Center

Caryn Meyers
IBM zSeries Server Group Solution Marketing for BI, US

Comments welcome

Your comments are important to us!

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

- Fax the evaluation form found in “IBM Redbooks review” on page 219 to the fax number shown on the form.
- Use the online evaluation form found at ibm.com/redbooks
- Send your comments in an Internet note to redbook@us.ibm.com

Chapter 1. Overview of Linux

Linux has become the hottest topic in operating systems in recent years, experiencing over 25% growth in 2000, more than any other operating system. So what is Linux? Linux is a UNIX-like operating system initially created by Linus Torvalds as a graduate student in 1991. The objective for developing Linux was to deliver a nonproprietary operating and application development environment, completely independent of underlying hardware architectures, that would offer maximum freedom to move applications from one hardware infrastructure to another by simply recompiling code, without expensive, labor-intensive porting efforts.

Linux is a fully networked 32-bit/64-bit architecture, that supports multiple users, multitasking, and multiprocessors, with a user-friendly Xwindows Graphical User Interface. Continued development and testing of Linux is governed by the Open Source community, which uses the Internet as the primary vehicle for technical exchange. Linux source code can be downloaded free of charge from the Internet, and programmers are free to modify the code. However, the integrity of the official kernel source code is managed and maintained by a strict submission and review process controlled by the Linux Review Board, an international standards body for Linux.

Linux, like other Open Source software, is distributed under the terms of the GNU Public License (GPL), and is packaged and distributed by approved distributors, such as Caldera, Red Hat, SuSE and Turbo Linux. Distributor packages include the Linux operating system code that has been precompiled for specific hardware environments, along with other Open Source applications and middle ware, such as Apache Web Server, SAMBA (file/print serving), Jabber (instant messaging), and IMAP/POP (mail servers). Distributors also offer Linux services and support packages, as does IBM Global Services.

In 1998 IBM announced a commitment to support Linux on all hardware and appropriate software, including the zSeries-S/390 platform. Linux 31-bit is generally available from Linux distributors for S/390 G5 and G6, and on zSeries hardware models. Linux 64-bit support on the zSeries is available on Open Source today and will be available from distributors later in 2001.

1.1 Linux on S/390

The Linux operating system has been running natively on S/390 hardware in test environments since early 1999. In December of 1999, IBM posted the

enablers for the S/390 hardware to the open source community and zSeries enablers became available in 2001. Compiled Linux source code has been available from the Internet by the Marist College Linux Web site since 1999 and has been available from key distributors since mid 2000. Linux 31-bit is supported on S/390 G5 and G6 and on zSeries hardware models. Linux 64-bit support on the zSeries is available on Open Source today and will be available from distributors later in 2001.

Linux on zSeries-S/390 is an ASCII environment that exploits IBM zSeries-S/390 hardware including IEEE floating point. Linux is not a replacement for other IBM operating systems on zSeries-S/390, and will coexist with z/OS-OS/390 and VM/ESA. It supports such UNIX tools as sed, awk and grep, compilers like C, C++, Fortran, Smalltalk and Ada. Network tools like Telnet, ftp, ping, and traceroute are supported as well. IBM software available on Linux for zSeries-S/390 include:

- DB2 Universal Database Enterprise Edition
- WebSphere Application Server Advanced Edition
- Java Development Kit
- Connectors
 - DB2 Connect (packaged with DB2 UDB EE)
 - CICS Transaction Gateway
 - IMS Connect
- MQSeries client
- Tivoli
 - Tivoli Framework
 - Tivoli Storage Manager client

1.2 Why Linux on zSeries-S/390?

There are several advantages to running Linux on the S/390 platform. We describe them in the following sections.

1.2.1 Expanded application portfolio

The nonproprietary environment of Linux opens the door for the zSeries-S/390 platform to perform as an application development and deployment server. The Linux application portfolio will greatly increase the number of applications available to zSeries-S/390 customers who want to continue to leverage the critical advantages of the platform to run business applications on a highly available and reliable architecture. Customers will

now have the flexibility to develop new applications directly on zSeries-S/390 and run them on all Linux-supporting platforms; or to develop and test on Linux on other platforms, such as the desktop, and then run the applications on Linux on zSeries-S/390, with a simple recompile. As applications from independent software vendors (ISVs) become generally available through Linux distributors, customers will be able to deploy them quickly on zSeries-S/390 without any special porting effort.

1.2.2 Cost savings

A new feature designed specifically for the Linux operating environment, called the Integrated Facility for Linux (IFL), is now available on G5, G6 and all zSeries server models. IFL gives you the ability to dedicate processors to the Linux operating system on a logically partitioned machine, transparently to the z/OS-OS/390 operating system environments. Processors dedicated to the Linux LPAR under IFL are priced at a lower rate than those for the z/OS-OS/390 environment. The added capacity, because it is dedicated to Linux workloads, does not increase the software licensing fees of the z/OS-OS/390 environment. The z/OS-OS/390 software pricing is confined to the capacity of only those processors enabled to the z/OS-OS/390 LPARs on the system. (This reflects pricing as of the publishing of this book and may have changed since the publication date. Consult your IBM server sales representative to obtain the most current pricing structure information.)

1.2.3 Server consolidation and faster time to market

All of the great flexibility and openness of Linux combined with the outstanding qualities of service of zSeries-S/390 results in an industrial-strength Linux environment.

zSeries-S/390 has several options for partitioning server resources into multiple logical servers. The PR/SM hardware feature provides the ability to partition the physical machine into as many as 15 logical servers, or LPARs, with dedicated, or shared, CPU and memory. The VM operating system, and the new Virtual Image Facility for Linux, allow you to partition an LPAR horizontally into multiple logical operating system images. These unique partitioning features provide the capability to quickly and easily consolidate a large number of Linux servers onto a single S/390 or zSeries server. This industry-leading zSeries-S/390 technology for dynamically sharing processing resources across multiple logical systems delivers value to the Linux environment not provided by any other architecture on the market today.

Figure 1 illustrates the typical Linux or UNIX application deployment strategy used by most installations deploying applications on non-zSeries-S/390 servers today.

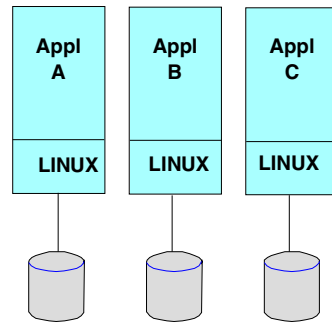


Figure 1. Linux application deployment on server farms

Without the robust operating environment provided by the z/OS-OS/390 operating system and the industry-leading reliability of the zSeries-S/390 hardware, most installations choose to deploy each new application in its own isolated operating system environment, on its own server, with a dedicated data base or data base partition. This leads to large server farms, with applications that cannot be easily integrated, and require complex systems management.

Contrast the configuration in Figure 1 to the flexibility available on the zSeries-S/390 architecture, illustrated in Figure 2.

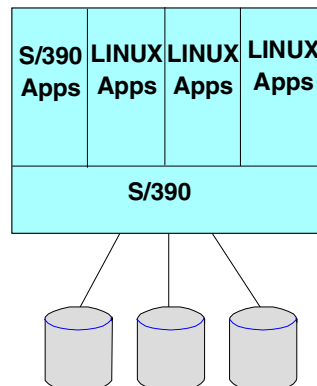


Figure 2. Linux application deployment on S/390

The unique capabilities of zSeries-S/390 for running multiple operating images simultaneously and sharing processing resources dynamically, supports many diverse workloads and multiple applications on a single server, with outstanding interpretability and integration between applications. Database sharing with integrity and ease of systems management are additional unique benefits that zSeries-S/390 brings to the Linux operating environment.

In fact, many customers are blending the data richness of the zSeries-S/390 environments with the web capability of Linux applications to deliver highly integrated, cost effective e-business solutions today.

1.3 Configurations for Linux on zSeries-S/390

There are several options for implementing Linux on zSeries-S/390. The first is to run Linux natively in a dedicated machine or in a Logical Partition (LPAR) under PR/SM. An LPAR where Linux is running enjoys the same PR/SM functions as any other LPAR. Under PR/SM there is the ability to dedicate or dynamically share capacity between Linux LPARs. Figure 3 illustrates a zSeries machine that has been partitioned to support Linux as well as z/OS-OS/390 operating systems.

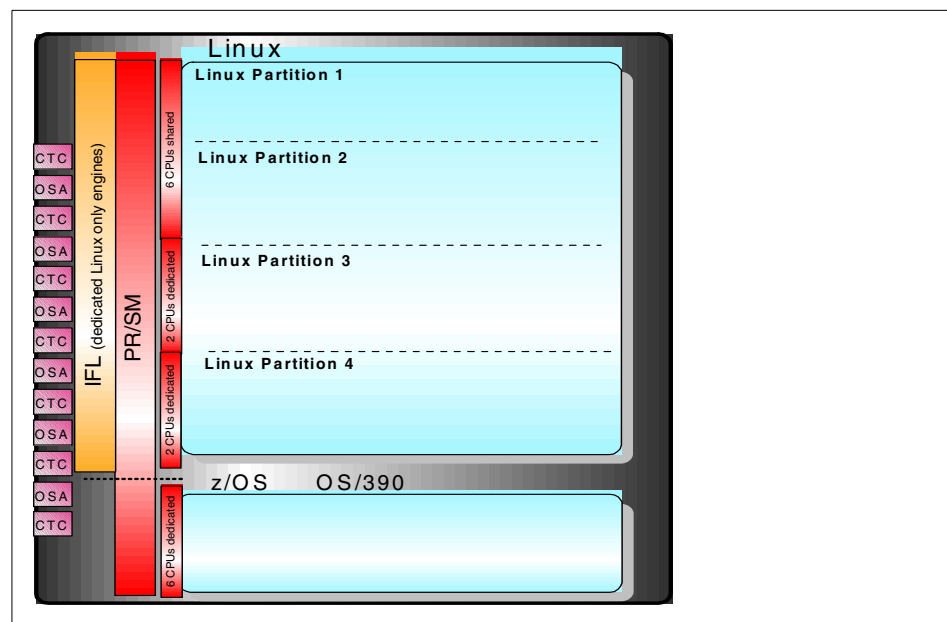


Figure 3. zSeries machine partitioned with Linux and OS/390 operating systems

The first two Linux LPARs share 6 IFL processors. The next two Linux LPARs each have dedicated IFL processors. Capacity can be assigned to each of the Linux LPARs as it is needed, based on business requirements, using standard PR/SM functions such as LPAR capping. The Linux applications running in LPAR 1 and LPAR 2 can take advantage of zSeries-S/390 PR/SM features to dynamically share the capacity of the 6 processors, while insuring that each application receives the minimum capacity it needs.

When set up properly, this means that the applications can effectively utilize the available capacity without requiring reconfiguration or human monitoring and intervention. In this illustration, LPARs 3 and 4 each have 2 processors of dedicated capacity, and LPAR 5 has 6 processors that are dedicated to the z/OS-OS/390 environment. It is important to note that PR/SM does not support sharing of processing resources between IFL LPARs and z/OS-OS/390 LPARs.

Another option is to run Linux in virtual images under VM on non-IFL processors on G5/G6 and zSeries servers. This option allows you to run multiple Linux images within a single LPAR while getting the full benefit of VM, such as resource prioritization and usage reporting.

This implementation provides the most flexibility and maximizes both PR/SM and VM resource management. In this configuration, both Linux and OS/390 operating environments can run in virtual images on a single LPAR and share processing resources. Each image has the isolation and integrity of a single operating system running natively, while sharing the total capacity of the LPAR with the other images. VM allows you to add and remove images dynamically, without requiring a reIPL of the LPAR. Under VM, it is possible to assign priorities to the individual virtual images within the LPAR based on business requirements, and to dynamically change those priorities when business needs change. The disadvantage of this option is that this environment is not supported on IFL processors and would be subject to traditional z/OS-OS/390 software and hardware pricing.

Another, more attractive, VM alternative is the Linux feature on z/VM for zSeries, or Virtual Image Facility (VIF) on G5/G6 server models as shown in Figure 4 on page 7.

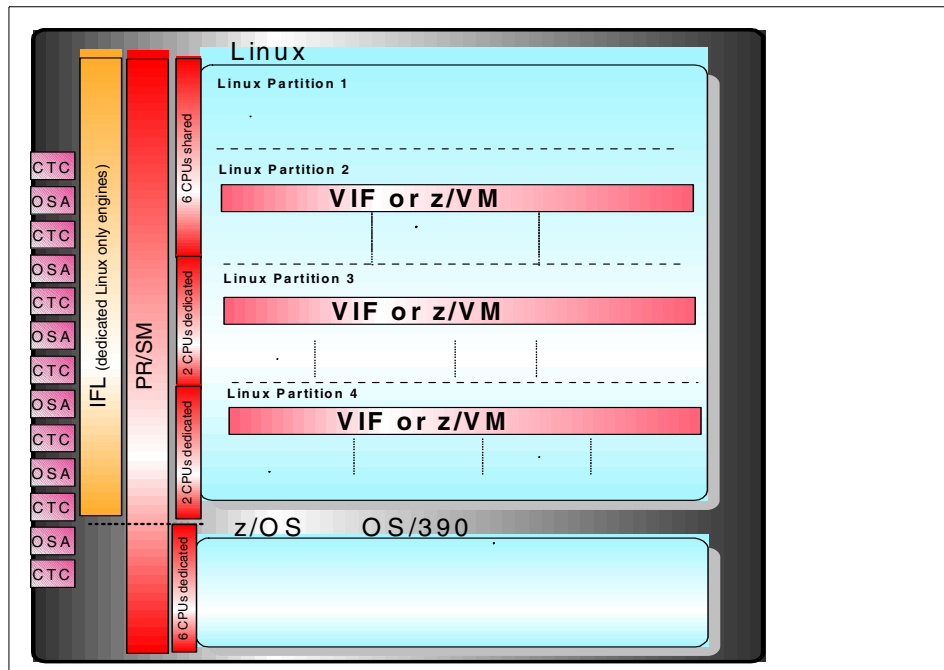


Figure 4. Linux in virtual images under VIF or z/VM

The Linux feature on z/VM is a specially-priced option that will allow zSeries customers to run multiple Linux images under z/VM on IFL processors, getting the full functionality of z/VM as well as the economics of the zSeries Linux pricing structure. An additional function of this feature is the ability to replicate or clone Linux images, providing a fast way to bring new Linux systems up quickly.

There is also a simple GUI for ease-of-use that will minimize the amount of VM skills required to exploit the flexibility of this environment. This feature is only supported on IFL processors. Note: z/OS and z/OS-OS/390 operating systems cannot run on IFL processors.

Virtual Image Facility (VIF) for Linux is an attractively priced, reduced-function adaptation of VM that runs on IFL processors on G5/G6 server models. VIF provides the logical partitioning capability of VM for running multiple Linux images in a single IFL LPAR, along with the cloning functionality and the user-friendly GUI. VIF can only be implemented on IFL processors and cannot support z/OS-OS/390 images. The virtual images running under VIF share the total capacity of the LPAR equally, and you cannot assign priorities to virtual images.

VIF is attractively priced for G5/G6 installations that do not require all of the robust functionality of VM, such as resource prioritization and reporting capabilities, for their Linux environment. (For a complete understanding of the difference in functionality between VIF and VM, consult your zSeries server sales specialist.)

Chapter 2. BI consolidations using DB2 for Linux on S/390

How can the Linux on zSeries-S/390 infrastructure be exploited to build integrated Business Intelligence solutions? In this chapter we discuss the following advantages:

- Consolidation of multiple data marts onto a single server with DB2 for Linux on zSeries-S/390
 - Rich end-user functionality
 - Faster time to market
 - Improved cost of ownership
 - Reduced cost of acquisition
 - Better systems management
 - Efficient use of capacity
 - More data marts, more data
 - Rapid deployment of data marts
 - Population of data marts
 - High availability of marts and BI applications
- Consolidation of network and gateway servers
- Web-enablement of the BI infrastructure
- Consolidation of BI application servers and middleware

2.1 Disadvantages of distributed data marts

Data warehouses, Operational Data Stores (ODS) and data marts are the fundamental data store structures in a BI infrastructure. The ODS and data warehouse are traditionally comprehensive in nature, containing highly detailed data on a corporate-wide scale. These structures are not usually tuned for direct access by business users who are focused on achieving very specific business objectives and have an interest only in a subset of the enterprise data. The data mart is a solution that is customized to meet the needs of end users with business goals focused on a specific business unit, subject matter or function. Data marts usually consist of subsets of the data in the data warehouse, primarily in summarized, rather than detailed, format.

Many zSeries-S/390 installations build their ODS and data warehouse structures on zSeries-S/390, where they can leverage the high availability and workload management capabilities that have made DB2 on z/OS-OS/390 a premier operational data base manager, to facilitate the movement, transformation and management of large volumes of data. However, for the most part, UNIX and NT servers have been the preferred platforms for data

marts. The primary reasons for choosing UNIX platforms for the implementation of data marts are as follows:

- Rich end user functionality - The users of data marts often require rich SQL functionality to manipulate and dissect the data. While DB2 on z/OS-OS/390 has included many enhancements in this area in recent years, DB2 UDB EE and EEE usually deliver these functions before DB2 for z/OS-OS/390 delivers them.
- Time to market - The growth of off-the-shelf BI applications that reduce the need for companies to develop their own code, has been almost exclusively focused on UNIX platforms.
- Cost of acquisition - While it has been recognized and documented by leading industry consultants that zSeries-S/390 provides the lowest total cost of ownership over time, most data marts are funded by business units that are focused on the initial cost of acquisition. It is relatively easy to develop a compelling business case for the initial startup cost of a data mart on a small UNIX or NT server, while the policies of charge-back and resource allocation can add complexity to the zSeries-S/390 environment.

This approach to many distributed data marts has some significant drawbacks which, as the number of data marts increases, can dramatically increase the total cost of ownership:

- The growth of large server farms characterized by poor application integration, complex systems management, increased costs in staffing and network infrastructure
- The overhead and complexity of converting source data in DB2 z/OS-OS/390 format to a UNIX data base format
- The infrastructure and systems management costs, as well as time requirements, of loading and maintaining external marts
- The challenge of meeting the end-user demands for continuous availability of data marts with current data
- The overall reduction in price performance as result of a cumulative growth in latent capacity that cannot be utilized.

With the introduction of DB2 for Linux on zSeries-S/390, zSeries-S/390 is positioned to address these issues and to provide additional value and advantages for consolidating multiple data marts on a single zSeries or S/390 server.

2.2 Advantages of consolidating data marts

There are numerous advantages in consolidating data mart servers on Linux for zSeries-S/390.

2.2.1 Rich end-user functionality

With DB2 UDB Enterprise Edition on Linux, DB2 now brings its rich relational functionality required by data mart end users (such as Automatic Summary Tables, federated databases, as well as rollup and cube functions), to the zSeries-S/390 environment. This offers a new and very attractive opportunity for customers to build DB2 data marts on the same zSeries-S/390 server as the DB2 z/OS-OS/390 data warehouse and operational systems, getting the combined value of the rich end user functionality of DB2 for Linux, and the robust workload management and large-volume, high availability capabilities of DB2 on z/OS-OS/390.

2.2.2 Faster time to market

Time to market is often cited as a reason why customers implement data marts on UNIX platforms. In the past, it has been easier and faster to deploy on a separate isolated system than it is on a production z/OS-OS/390 environment.

With Linux on zSeries-S/390, this is no longer true. Sizing, ordering and waiting for the hardware and software for a new UNIX server system can take significantly longer than defining an LPAR on an existing zSeries-S/390 server, or using Capacity Upgrade on Demand to enable spare processors that are already installed on a server for a new workload. The hardware can be enabled, and the software installed in less than a day on a zSeries-S/390 platform. DB2 for Linux can be acquired in real time with a simple download from the Internet. Installation of an initial Linux system can take as little as a few hours, and DB2 for Linux can be installed and initialized in minutes. The VIF and zVM image replication feature can further reduce the time it takes to implement new data marts on Linux for zSeries-S/390. It is conceivable that a new data mart could be brought on line in less than an hour!

2.2.3 Reduced cost of ownership

Another reason why customers have chosen to implement solutions on the NT/UNIX platform is the cost of the computing model. While industry consultants have long recognized that S/390 total cost of ownership is the most attractive in the industry, the cost of acquisition and initial startup of

individual data mart solutions on zSeries-S/390 is less appealing than it is for smaller UNIX systems.

2.2.3.1 Reduced cost of acquisition

Data mart consolidation on Linux on zSeries-S/390 can significantly reduce the startup costs for data marts on zSeries-S/390.

IFL processors dedicated to Linux workloads are less expensive than processors for z/OS-OS/390 and VM.

OTC pricing for DB2 for Linux and off-the-shelf BI applications running on Linux for z/OS-OS/390 is the same as it is for UNIX platforms.

2.2.3.2 Better systems management

The leading edge systems management features of the zSeries-S/390 architecture have been a key reason why customers have sought to consolidate workloads on S/390 for years. This is also a compelling reason to consolidate data marts on Linux images on zSeries-S/390. This gives you the opportunity to streamline enterprise systems management processes, and reduce the overall cost of ownership of multiple data marts, under a single server.

2.2.3.3 Efficient use of capacity

A key characteristic of separate data mart servers is the inability to leverage latent capacity, or “white space”. Characteristically, individual applications have unique processor utilization requirements based on the business cycle and functional requirements of the business users. Utilization levels will fluctuate up and down throughout the business day, at different times of the week and even at different times of the business cycle. When data marts are running on separate servers, each server must be configured to provide acceptable performance during high utilization periods, resulting in latent capacity during lower utilization.

There are also considerable investments required in infrastructure to meet the peak capacity demands of users. It is very typical for individual data marts on UNIX servers to run at utilization levels that are significantly lower than 100%, on average in the range of 40-60%. These isolated systems have no ability to share unused capacity. When one data mart server is experiencing high utilization and suffering performance degradation, it cannot capitalize on other data marts that may be under utilized and have excess capacity. The cumulative effect is that a large percentage of the total processing resources remains idle a great deal of the time, continually driving up the cost/user over time.

This escalating problem can be addressed by consolidating multiple physical data mart servers on zSeries-S/390, running DB2 for Linux marts on Linux, natively in individual LPARs, or under z/VM or VIF in a single LPAR. The unique partitioning capabilities of PR/SM and VM provide the opportunity to configure a single physical server into multiple logical data mart servers with the capability to share processing resources, while maintaining complete data integrity and workload isolation. In this type of configuration, the unused capacity of the data marts that are experiencing a lull in activity can be exploited by data marts that are experiencing high utilization. Consolidating onto a central server can be particularly attractive when individual marts service users in different geographical locations worldwide, and usage requirements are naturally staggered due to time zone differences.

This consolidation of “server white space” is a key benefit of implementing Linux LPARs on zSeries or S/390. This capability permits multiple applications and workloads to be consolidated on a single physical server, while maintaining the integrity of each application and insuring that each will receive the minimum capacity it requires. This consolidation of data mart applications allows businesses to maximize the return on investment in their BI infrastructure.

2.2.4 More data marts, more data

Over time, business users tend to do increasingly sophisticated analyses on data, and the demand for more data grows on a continual basis. The success of initial decision support systems spawns requirements for new BI applications, deployed across more functions and departments, resulting in the growth of the number of data marts and the amount of data end users need to access.

Along with the growth in data and the number of data marts comes increased demand for 24x7 access to the data marts. This growth presents IT with the challenge of keeping data marts current with decreased time for the data transformation and movement processes to complete. This challenge is complicated even further where there are large numbers of distributed data mart servers.

2.2.4.1 Rapid deployment of data marts

As was mentioned earlier in this chapter, the functionality of VIF and z/VM offers excellent opportunity to rapidly deploy new data marts on Linux, and requires minimal skills in the system environment to use.

Horizontal partitioning of an LPAR into multiple Linux images can be done dynamically, with no need to reconfigure capacity or reIPL.

Data marts can be taken online and offline dynamically.

With the cloning capabilities of VIF on G5/G6 IFL processors and zVM on zSeries IFLs, and a properly defined TCP/IP environment, it is conceivable to define, build and load a new DB2 for Linux data mart, from scratch, in minutes.

2.2.4.2 Population of data marts

A large portion of time related to implementing and maintaining a data mart is devoted to the movement of the data from one platform to another. The primary source of data for decision support systems is most often the operational system on zSeries-S/390. Recognizing this, solution architects have designed data warehouse and ODS infrastructures on DB2 for z/OS-OS/390, so that they can leverage the proximity of the data and the z/OS-OS/390 workload management functionality to maximize the performance of moving large volumes of data into the enterprise warehouse and ODS. Outboard data marts have not been able to capitalize on this proximity to the warehouse and suffer the cost of moving data across networks.

Consolidating several data marts in Linux LPARs on the same physical platform as the data warehouse and ODS brings the data marts into close proximity to the DB2 source systems. Now the transfer of data to the data marts is done between LPARs, eliminating the overhead of network traffic. With 100 gigabit Ethernet cards, the speed of transfer is significantly faster than over the network. The internal OSA card available on the G5/6 and zSeries systems, recognizes the IP address for each LPAR, and transfers the data internally on the card.

IBM has made a statement of direction that in the near term it will deliver a significant enhancement to data transfer rates on the newest zSeries processors. This enhancement will enable communication between LPARs using TCP/IP over high speed internal data transfer pipes. Data will be moved from one LPAR to another, through memory, creating an "internal network". This transfer of data through memory has an additional performance benefit when moving data from one DB2 data base to another running on different LPARs: The number of reads and writes required for the data transfer is reduced from multiple read and write operations to a single read and a single write, which further streamlines the process and reduces the time to load the data marts.

These hardware features can be combined with software solutions for the rapid population of the DB2 for Linux data marts in Linux LPARs. DB2 import/export and data load/unload tools and utilities from IBM and other

software providers can leverage these internal data transfer mechanisms between LPARs to further reduce the time required to maintain the currency of data marts. Batchpipes can also be exploited to add more parallelism to batch update processes, effectively increasing I/O bandwidth and shortening the update window requirements.

The speed with which data can be moved into the system allows tight integration of Linux-based applications and marts with applications of the z/OS-OS/390 environment. This adds up to being able to populate more data marts with more data in a shorter amount of time.

2.2.4.3 High availability of marts and BI applications

To end users, “availability” means that the business application is running and can access the data. Shortening update windows will shorten the time that the data is unavailable due to scheduled updates. However, the reliability of the system environment is key to the availability of the application server and data base server. zSeries-S/390 hardware has unsurpassed reliability , availability and serviceability. The mean time to failure of a zSeries-S/390 server is over 60 years, while most UNIX systems measure availability in days, weeks and months.

Such zSeries-S/390 features as self-error detection and correction, Remote Support Facility, “phone home”, and concurrent microcode updates lead the industry in preventing planned and unplanned outages. Capacity Upgrade on Demand (CUD) enables dynamic increases in capacity to support additional data marts and BI applications without interrupting service to the marts that are already running in existing Linux images.

2.3 Consolidation of networks and gateway servers

We have discussed the many advantages of consolidating data mart servers on Linux for zSeries-S/390. There is an equally powerful story for consolidation of gateway servers that are required for connectivity to DB2 on z/OS-OS/390, into the Linux LPAR. With DB2 for Linux data server in the zSeries or S/390 Linux partition, the workstation clients can access the data mart directly through DRDA as they do today on AIX or Solaris. There is no need to install any additional gateway or connectivity software.

However, many ODBC and JDBC applications need direct access to DB2 data warehouses, ODSs, and data marts on z/OS-OS/390. This connectivity requires such gateway technologies as DB2 Connect. Prior to the introduction of Linux on zSeries-S/390, these gateways had to be distributed on NT/Unix servers. Today, DB2 Connect runs on Linux for zSeries-S/390 and can be

implemented inside a Linux LPAR, eliminating the requirement for DB2 Connect servers, dispersed throughout the network. DB2 Connect Unlimited Edition is an option on DB2 for Linux and you have the choice of getting the connectivity function either through DB2 for Linux or by implementing DB2 Connect in its own Linux image.

2.3.1 High speed Joins

One specific data mart may not contain all of the information that users need to do a complete analysis. Users will often want to generate queries that need to join tables in one data mart with tables that reside in other data marts and/or the data warehouse. When joining tables between a data warehouse and data marts that reside on separate physical servers, performance can be significantly impacted by the overhead of network traffic. With data marts consolidated in LPARs on a single zSeries or S/390 server, cross-table joins of DB2 data bases are accomplished with LPAR-to-LPAR communications and do not have to go over the external network. Performance is even further improved on zSeries with high speed memory-to-memory data transfer.

2.4 Web-enablement of the BI infrastructure

In addition to data accessibility, the introduction of Linux also eases the speed of Web-enabling database applications and eliminates another requirement for a middle-tier server. The Apache Web server is an Open Source product that can be downloaded from the Web free of charge and is packaged by all of the Linux distributors today at no additional charge. Apache can be used to enable the Linux data mart system, as well as the zSeries-S/390 data warehouse.

The IBM WebSphere product can be implemented on a Linux LPAR on zSeries-S/390 for Web-based applications that require a more robust Web solution. If you are running WebSphere on z/OS-OS/390 today, you can continue to do so. Then, as new Web applications are developed, you may want to consider developing them on WebSphere in a Linux LPAR, using the Application Enablement (AE) package of WebSphere. You have the option to run the new applications under WebSphere on Linux, or to maintain the data base and business logic on the z/OS-OS/390 partition where it gets the full benefit of the z/OS-OS/390 workload management capabilities.

2.5 Consolidation of BI application servers and middleware

Linux affords you the opportunity to eliminate the need for the middle-tier server for applications as well. Because Linux is standardized across

hardware platforms, vendors who develop new applications on Linux can make them available on the zSeries-S/390 environment simultaneously with other platforms without any special porting or rework. Installations that do their own application development can develop new applications directly on Linux on zSeries-S/390, and recompile existing Linux applications for deployment on an zSeries-S/390 LPAR. Because Linux is so similar to UNIX, UNIX applications port to Linux quickly and easily. There are many non-IBM BI applications that run on Linux, a number of which are supported on zSeries-S/390 today.

Linux on S/390 opens the door to customers and software developers who want to take advantage of the zSeries and S/390 platform strengths to easily deploy their applications on the mainframe. We have already talked about consolidating the DB2 Connect middleware on Linux, and IBM's Intelligent Miner Scoring became available on Linux for zSeries-S/390 in March, 2001.

Because the Linux operating system is standardized across all hardware platforms, vendors like Microstrategy, Brio, and Business Objects can economically move their middleware, Web servers and applications to the mainframe. Today, many IT organizations develop and maintain their own customized Extraction, Transformation and Load (ETL) routines for cleaning and loading source data on to the z/OS-S/390 data warehouse. Linux on zSeries-S/390 provides an opportunity for price-conscious installations to use less expensive processing cycles to do this processing on IFL processors.

Chapter 3. Introducing the test environment

To assess Linux performance on S/390 and possible BI consolidations using DB2 for Linux on S/390, we built a test environment at the IBM S/390 Teraplex Centers where we could test Linux and DB2 for Linux implementations on S/390, and where we could run a few pertinent query and data population scenarios to observe and compare preliminary BI scalability and performance aspects on consolidated OS/390 and Linux platforms.

As shown in Figure 5 on page 20, we designed two types of scenarios:

- Data population scalability and performance scenarios
 - Transferring data from a DB2 for OS/390 data warehouse to DB2 for Linux data marts
 - Our objectives were to demonstrate the large choice and usability of data population techniques on the Linux on S/390 platform, and give an idea of their scalability and performance behavior on Linux/S390 platform.
- Query scalability and performance scenarios:
 - Running the same set of queries against the DB2 for OS/390 data warehouse and against the DB2 for Linux data mart.
 - Our objectives were to compare and assess that the behavior of the workload is the same on both OS/390 and Linux platforms.

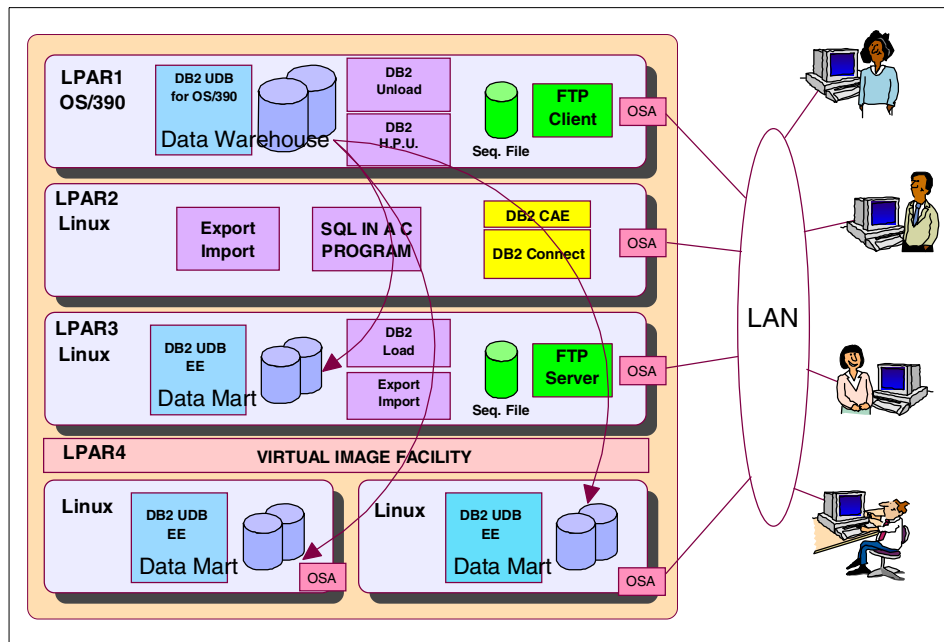


Figure 5. Scenarios for BI scalability and performance

The LPAR design is as follows:

- LPAR1 runs an OS/390 operating system with the DB2 data warehouse.
- LPAR2 runs a Linux operating system and is defined as the network gateway and application partition. We run our query performance tests out of LPAR2 and compare DB2 for OS/390 scalability attributes to DB2 for Linux scalability attributes.
- LPAR3 runs the Linux operating system and is defined as the native Linux data mart partition. Data is extracted from the data warehouse in LPAR1 and is loaded into the data mart in LPAR3.
- LPAR 4 runs the Virtual Image Facility (VIF) environment under which run multiple Linux virtual images housing each a DB2 for Linux data mart. This LPAR offers an alternative for the native Linux data mart. The data population techniques are the same in a virtual Linux image as in a native Linux LPAR.

3.1 Test configurations

To support the scenarios we designed, we implemented the following test configurations. Figure 6 shows the LPAR configurations on the Teraplex G5-turbo we used to run our tests.

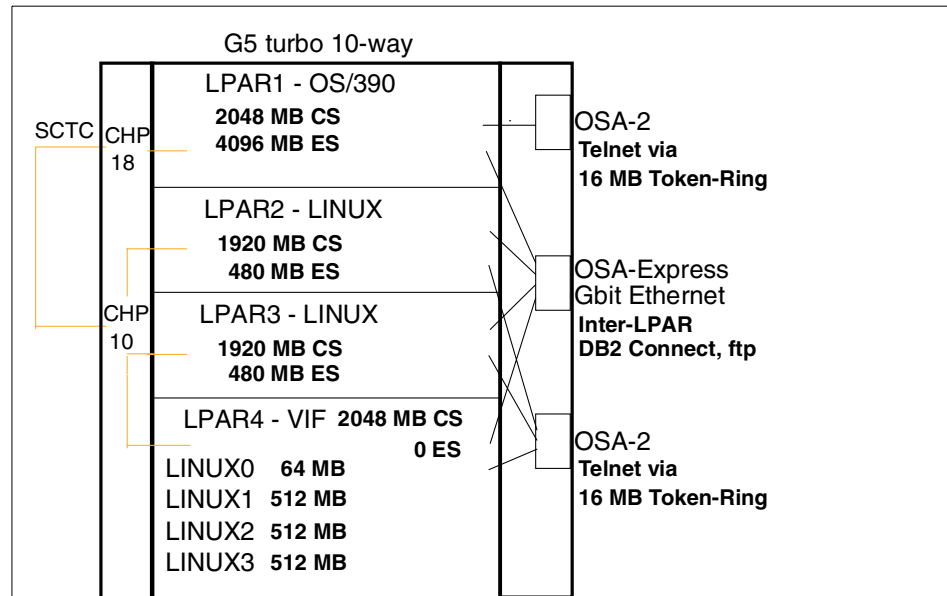


Figure 6. Teraplex system images

- Linux LPARs

Each LPAR used for Linux (LPARs 2,3,4) was dedicated to Linux. The expanded storage on Linux LPARs 2 and 3 was used for swap. The Linux images under VIF had DASD partitions defined for swap.

- Processors

CPs were moved, depending on the test.

CPs could be configured on/off the OS/390 LPAR dynamically. The Linux LPARs had to be deactivated/activated to change the number of CPs.

- Connectivity

Telnet access was provided via OSA-2 cards with Token-Ring connections. Each image, including the Linux VIF images, was allocated a pair of OSA-2 addresses. Inter-LPAR network connectivity was provided via an OSA-Express Gigabit Ethernet card. Four OSA-Express addresses

were allocated for each operating system image. Tests were also done with inter-LPAR network connectivity provided by SCTCs.

- Disks

The DASD containing the system libraries (OS/390, DB2 for OS/390, Linux, DB2 for Linux, etc.) was RAMAC-II DASD with integrated controllers. An Enterprise Storage Subsystem 2105-E20 was used to contain the DB2 tables. All of the DASD subsystems used in the project were shared among the LPARs using channel path IDs configured with Escon Multiple Image Facility (EMIF), which enables sharing of channel paths among LPARs.

More configuration details are provided in the LPAR resource matrix provided in Appendix A, “Teraplex LPAR resource matrix” on page 181.

3.2 Product implementations

Based on this configuration we did the following product implementations:

- Linux on S/390 setup

The implementation of Linux on S/390 is described in Chapter 4, “Setting up Linux on S/390” on page 25.

- DB2 for Linux setup

The implementation of DB2 for Linux on S/390 is described in Chapter 5, “Implementing DB2 for Linux on S/390” on page 79.

- Data mart implementation

The implementation of the data mart using DB2 for Linux is described in Chapter 6, “Building the data mart with DB2 for Linux” on page 111.

3.3 Scalability and performance tests

When the test environment was set up and the Linux data mart was up and running, we ran two sets of scalability and performance tests:

- Scalability and performance tests for data population

The data population tests are described in Chapter 7, “Scalability and performance tests for data population” on page 129.

- Scalability and performance tests for queries

The query tests are described in Chapter 8, “Scalability and performance tests for queries” on page 159.

3.4 Administration and monitoring

We conclude this redbook with considerations on how to administer and monitor a BI Linux environment running on Linux and S/390 platforms. You will find those considerations in Chapter 9, “Administering and monitoring the BI environment” on page 171.

Chapter 4. Setting up Linux on S/390

SuSE Linux 7.0 is an operating system which runs on a variety of platforms and has been recently ported to the S/390 platform where it can run in the following modes:

- Native in an LPAR
- Under VM/ESA
- Under Virtual Image Facility (VIF)

VIF provides many virtual S/390 instances in one LPAR. In each of this instances one Linux system may reside. For the reader who is acquainted with S/390 operating systems, VIF is a stripped down version of VM which only purpose is to support many instances of Linux in one LPAR.

SuSE Linux 7.0 is distributed on CD-ROM or can be downloaded from the web. To install on S/390 you need to set up an a NFS or FTP server where the installation files are stored or the CD file system is *mounted*.

We used an Intel workstation running SuSE Linux 7.0 as an NFS and FTP server hosting the installation files.

Our test configuration included the following:

- A Linux NFS/FTP server on an Intel PC connected to the LAN
- A S/390 LPAR hosting the Linux system
- A userid to access an OS/390 system where we could produce a bootstrap tape

We used the following documentation for installing, available on the SuSE Linux CD-ROM:

Linux for S/390, SG24-4987

SuSE Linux 7.0 Installation, Networking, Know How, by Berlich et al.

The chapter includes the following sections:

- Installing SuSE Linux 7.0 in a native LPAR
- Invoking YaST and configuring
- Installing SuSE Linux 7.0 under VIF
- Implementing the first Linux image under VIF
- Adding a network connection device to an active Linux

4.1 Base installation of SuSE Linux 7.0 in a native LPAR

To install the Linux system native in an LPAR you have to create a bootstrap image on a device that can be IPLed from this LPAR. This is normally a tape. This tape can be created on any system running under OS/390 or VM or VSE.

If your Hardware Management Console (HMC) and system Service Element have the last level of service, you should be able to IPL from a CD or an FTP server (*Load from CD or server* function now available on the HMC), instead of using a tape drive. The LPAR profile should be set as "Linux only LPAR".

The base installation involves the following tasks:

- Acquiring network information
- Creating the bootstrap tape
- IPLing the bootstrap tape
- Performing installation tasks from the Linux workstation

4.1.1 Acquiring network information

You need to identify an NFS or FTP server where you can mount the CDs on which Linux is delivered. You need an operating system on S/390 where you can create a bootstrap tape. Your NFS/FTP server should have a network connection to the S/390 and the LPAR where Linux is going to be installed.

You must first collect the information relative to the network connections you will be using.

In our test scenarios we used the connections listed in Table 1.

Table 1. Our connections

Comments	Name	IP addresses we used	IP addresses you will use
OS/390 FTP port		9.100.203.106	
LINUX LPAR IP address	ntc.demo.mop.ibm.com	9.100.193.003	
Sub-Net Mask		255.255.252.0	
Broadcast address		9.100.195.255	
NFS/FTP Server		9.100.147.136	
Domain name	mop.ibm.com		

4.1.2 Creating the bootstrap tape

You will find a detailed description in *SuSE Linux 7.0 Installation, Networking, Know How*, by Berlich et al.

4.1.2.1 Identify the files to copy to tape

Search for the files listed in the installation guide and the README file on the CD.

- Mount the CD file system on the NFS/FTP server:

```
->mount /dev/cdrom /cdrom
```

- Check if all the files are there:

```
->cd /cdrom/suse/images (change directory)
```

```
->ls (list directory)
```

Transfer and copy the following files:

- Parmfile (boot parameters file)
- Tapeipl.ikr (kernel image file)
- Initrd (initial root file system)

Do not use boot parameter file parmfile.v; it is used for an installation of Linux under VM or VIF.

4.1.2.2 FTP the installation files to OS/390

We chose to FTP from the workstation site as follows:

- Connect to the OS/390 and logon:

```
ftp <OS/390 IP-address>
```

```
User <userid>
```

The system will prompt you for the password:

```
pass <password>
```

- Define the file record format and request space to allocate the receiving data set on the OS/390 system:

```
quote site lrecl=1024 recfm=f primary=300
```

Set the record format to 1024 bytes of fixed length, and allocate 300 blocks for each file.

- Switch to image transfer mode, no translation:

```
bin
```

- Send the files to OS/390:

```
put parmfile '<userid>.linux390.parm.line'
```

```
put intitrdr      '<userid>.linux390.intitrdr.txt'
put tapeipl.ikr  '<userid>.linux390.image.txt'
<user id> is your fully qualified user ID on OS/390
```

- Close the FTP session:

```
quit
```

Now you are done on the LINUX workstation for a while.

4.1.2.3 Create the bootstrap tape

The IPLable tape has a special format, particularly no VOL and DATASET labels. To unlabel the tape we ran the OS/390 job shown in Figure 7.

```
//LINUXUT JOB (1458,MOP),'UNLABEL TAPE',CLASS=A,MSGLEVEL=(1,1),
// MSGCLASS=H,NOTIFY=&SYSUID
//*****
//*
//*      JOB TO OVERWRITE A SL ON A TAPE
//*
//*****
//STEP1      EXEC   PGM=IEBGENER
//SYSPRINT DD      DUMMY
//SYSUT1 DD        DUMMY,DCB=(RECFM=F,BLKSIZE=1024,LRECL=1024)
//SYSUT2 DD DISP=(NEW,PASS),LABEL=(,NL),DSN=DUMMY,
//              DCB=(RECFM=F,LRECL=1024),UNIT=/FA00
//SYSIN      DD      DUMMY
```

Figure 7. OS/390 job to unlabel a tape

To create the IPL tape, we used the job shown in Figure 8 on page 29.

```

//LINUXIPL JOB (SYS0000), 'LINUX BOOT-TAPE',
//TIME=1440,NOTIFY=&SYSUID,REGION=4M,
//          CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//*****
//* LINUX INSTALL: JCL to create the IPL tape
//*****
//TOTAPE  PROC LBL=NOSUCH,FILE=NOSUCH.FILE
//IEBGNR   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT1   DD DSN=&FILE,DISP=OLD,
//          DCB=(LRECL=1024,RECFM=F,BLKSIZE=1024)
//SYSUT2   DD LABEL=(&LBL,NL),DISP=OLD,
//          UNIT=/FA00,
//          VOL=(PRIVATE,RETAIN,SER=LINUX1)
//TOTAPE   PEND
//*
//FILE1    EXEC TOTAPE,LBL=1,FILE=HOARAU.LINUX390.IMAGE.TXT
//FILE2    EXEC TOTAPE,LBL=2,FILE=HOARAU.LINUX390.PARM.LINE
//FILE3    EXEC TOTAPE,LBL=3,FILE=HOARAU.LINUX390.INITRD.TXT
//*

```

Figure 8. OS/390 job to create the bootstrap tape

4.1.3 IPLing the bootstrap tape

After creating the bootstrap tape we are ready to IPL the Linux system from the tape:

- Check that the I/O devices, such as the DASD you want to install Linux on and the network adapter you wish to use for connecting to the network, are available and defined to the LPAR in the active IOCDs.
- IPL your S/390 Linux from the tape.

During the IPL, Linux will start with sensing all the I/O devices defined to the LPAR. If one or some are not responding to the senses, the process could be slow. You can check the results of the sense I/O process by inspecting the System Operating Messages log, on the Hardware Management Console. The messages appear one or two minutes after IPL. Ensure, at least, that the I/O devices you plan to use with Linux are correctly identified (however, if a very large number of I/Os are defined to the LPAR, the HMC display buffer capacity could be exceeded and you could miss the first messages).

- Answer the prompts on the HMC.
- Confirm the software license.

You will be asked to read and confirm the software license. If you do not answer yes to this question, you cannot proceed with the installation. The license conditions are shown on the console in chunks and to get the next chunk you have to send <command/enter> on the console. Finally, you will be asked to accept the license; you have to answer yes in order to proceed.

4.1.4 Initializing network connections

You need to do the following:

- Enter the network device number as you are requested:

0500 (we used our OSA2 device connection)

Recommendation

If you are running in a parallel environment, we strongly recommend that you do not use the auto-detect feature, because this might have an adverse impact on the other LPARs. Specify the dev number directly.

At this point the system loads the corresponding network device driver module (in our case the LAN Channel Station driver to support a Token Ring through an OSA2 card) and initializes the connection to the network.

```
insmod -v lcs noauto -l devno-portno-pairs=0x0500.0;
```

You have to wait for the initialization of the network device driver to finish.

```
cs: tr0 configured as follows read subchannel=1a3 write subchannel=1a4
```

The initialization of the network device driver (tr0 here) has completed.

- Enter the network definitions to set up the TCP/IP stack. You must answer the prompts with the networking information you collected earlier:
 - Full host name
ntcdemo.mop.ibm.com
 - IP address
9.100.193.3
 - Netmask
255.255.252.0
 - Broadcast
9.100.195.255
 - Gateway address
9.100.193.69
 - DNS
9.100.147.136

- Domain (DNS search domain)
mop.ibm.com
- Transfer units
4096

This parameter defines the maximum size of the packets sent through the network.

- Confirm your choices.

The system sends you a confirmation message listing all your inputs, that you have to confirm. A negative answer allows you to return to the first prompt to correct your inputs.

- Chose a temporary password for user root.

The system creates a user root and prompts you for a temporary password:

```
linx390
```

You will need this password during the installation steps described in 4.1.5, "Installation tasks performed from a Linux workstation" on page 31.

- Activate the Telnet connection daemon.

The system then pings the network to see if everything is accessible. You will receive several messages, the last one being:

```
"Network setup finished ..running inetd .."
```

This means that *inetd* has been started and has also started other network services, among them *telnet*. The Linux system, loaded from the image file on the tape into the RAM disk in the LPAR storage, is now ready to be accessed via a Telnet session. For convenience, we will be using *xterm* on the Linux PC we installed to serve as an NFS/FTP server, to telnet to Linux.

At this point you are done with the first part of the installation. The rest of the installation will be performed from the workstation with the assistance of YaST.

4.1.5 Installation tasks performed from a Linux workstation

The following is explained in *SuSE Linux 7.0 Installation, Networking, Know How*, by Berlich et al p. 43.

4.1.5.1 Give access authorizations

We must grant Linux on S/390 authorization to read the CD files. For this purpose, we must update the Export profile on the NFS server.

- Edit the file `/etc/exports` on the LWST.
- Insert the following line in it:

```
/cdrom <L390 ip address> (ro)
```

This will give read-only access to the installation files to Linux on S/390 .

4.1.5.2 Telnet to Linux on S/390

From an *xterm* on the LWST, open a telnet session with the L390 system:

```
telnet 9.100.193.3
```

4.1.5.3 Log in to Linux on S/390 using root

When the session is established, log in as user root:

```
login root
(root install pw)
```

Now we are accessing the Linux system on S/390 LPAR.

4.1.5.4 Activate the swap file system in expanded storage

Because we had spare expanded storage on this LPAR, we decided to install the Linux swap file system in expanded storage, and thus avoid monopolizing a full volume of 3390 for swapping.

The expanded storage device driver to load is *xpram*. The Linux device name is `/dev/slram0` when only one partition of the expanded storage is defined (this is the default when no parameters are used with `insmod`).

```
insmod xpram
/sbin/mkswap /dev/slram0
/sbin/swapon /dev/slram0
```

Check your results with `lsmod` and `free`.

4.1.5.5 Install and format Linux on S/390 system disks

- Install the DASD device drivers and define them to L390:

```
insmod dasd dasd=544c,544d
```

Check with:

```
cat /proc/dasd/device
```

- Perform the low-level formatting of the disks (takes some time):

```
dasdfmt -v -b 4096 -n 544c
dasdfmt -v -b 4096 -n 544d
```

You have completed the DASD installation. You will now be invoking YaST to install a full Linux on the target DASD volume. After completing YaST you will be able to boot Linux on S/390 from the DASD. Keep the tape for backup.

We have defined two DASD volumes, one for the Linux system and one to be used for installing our DB2 UDB file systems.

4.2 Invoking YaST and configuring

We now describe the installation steps using YaST, a utility supplied by SuSE Linux 7.0 .

Note: YaST stands for “yet another setup tool”. This installation tool is exclusive to SuSE Linux distributions.

Invoking YaST is done through a Telnet session to the S/390 Linux system.

YaST presents a sequence of panels with input and/or selection fields. Based on some parameters and values you might have entered during the initial installation, the tool will propose default values for some fields, but you may still overwrite them whenever necessary.

The use of this tool is straightforward and you should have no problems in using it. We include some of the panels to give you the look and feel. The tool is explained in depth in *SuSE Linux 7.0 Installation, Networking, Know How*, by Berlich et al.

To make a selection in a presented panel, you navigate with the <tab>-key, <up-arrow>, <down-arrow> and confirm by pressing <enter>. This applies also if your selection is *Abort*. If there are fields with input to fill in, you navigate as explained, fill in your values, and proceed to the next field. Some selections may show a subpanel, or tiled menu, where you have to select other values or options. Nevertheless, the navigation scheme is always the same.

4.2.1 Invoking YaST

Enter either `YaST` or `yast`.

4.2.1.1 Select the language of the installation



Figure 9. Example of a YaST selection panel

Select **English**.

4.2.1.2 Select the installation media

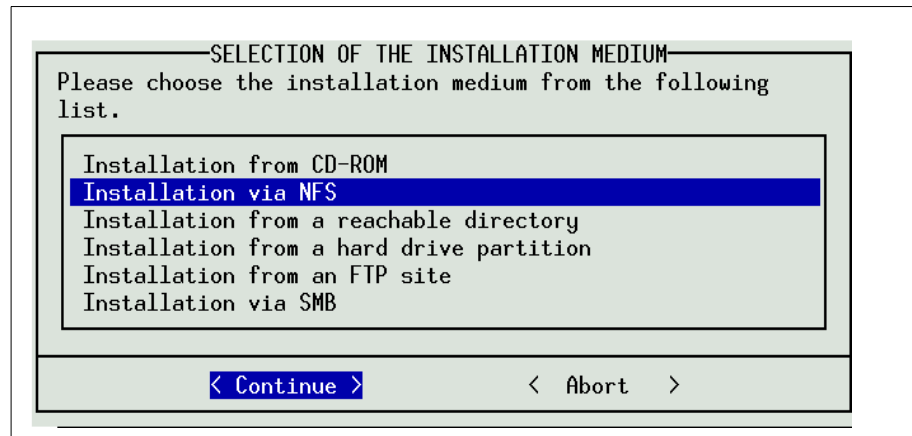


Figure 10. Installation medium

Select **Installation via NFS**.

4.2.1.3 NFS server information

This will bring up the Enter the data for the NFS server panel. On this panel, enter the IP address of the NFS server and the directory where the SuSE CD resides (e.g., /cdrom).

4.2.1.4 Selection of Installation type

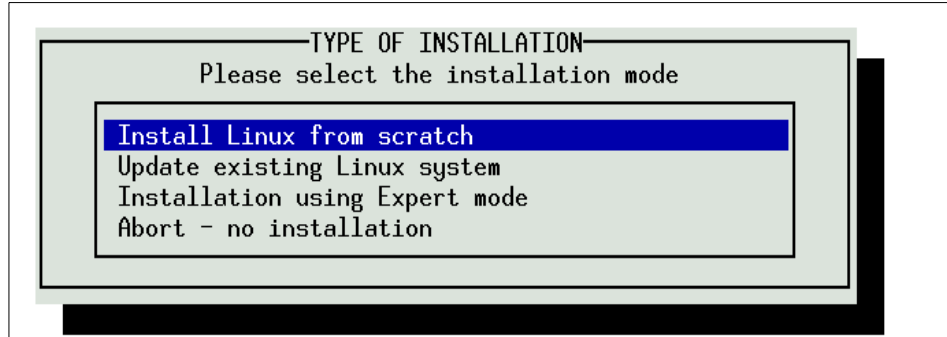


Figure 11. Type of installation

You can choose to do different types of installation, such as:

- Install Linux from scratch
- Update an existing system
- Install in expert mode
- Do not install at all

Select **Install Linux from scratch**.

4.2.1.5 Select Swap partition

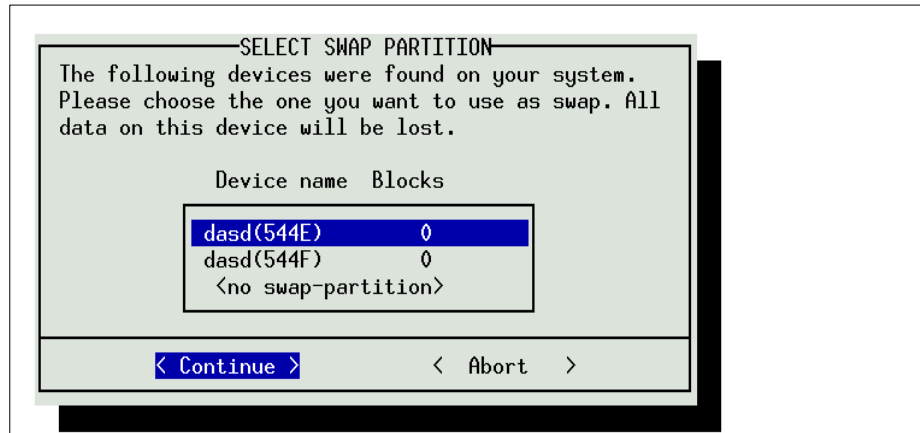


Figure 12. Select swap partition

We chose **no swap-partition**.

In our installation, we reserved expanded storage for the swap file system and kept the second DASD for installing the UDB file system. Your choice might be to put the swap on a separate volume, which you would select by highlighting the corresponding line.

The selected DASD would then be presented as swap on the “Creating file systems” screen.

4.2.1.6 Partitioning hard drives

At the time of writing, hard drive partitioning is not supported yet. There can be only one DASD partition, *partition 1*, on Linux for S/390.

Select **Do not partition**.

4.2.1.7 Creating file systems

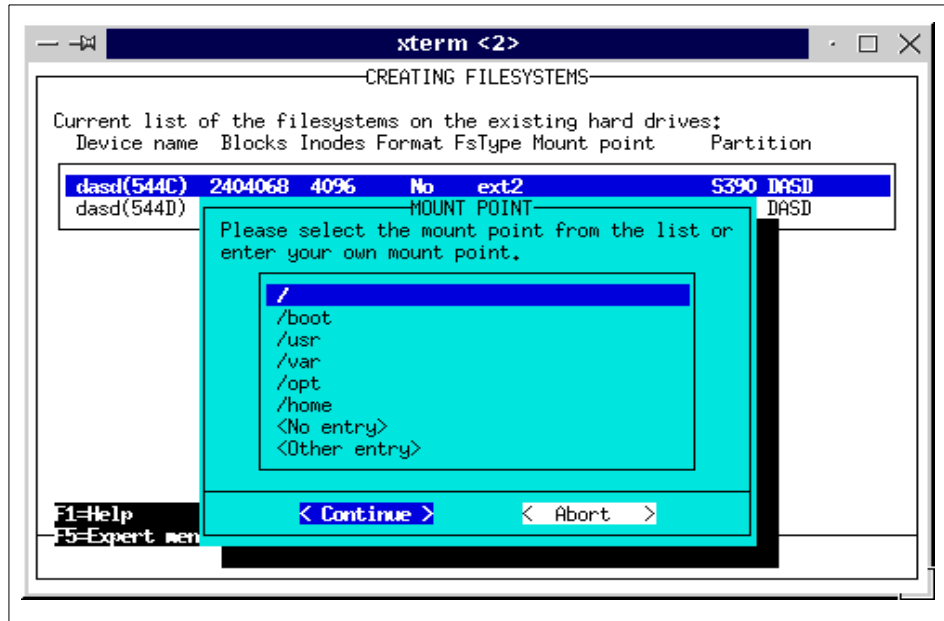


Figure 13. Creating file systems

- Selecting the mount point.

We are installing on a single volume. Everything is going to be referenced from the root mount point: "/".

Select /.

- Confirming the selections.

Because there is a format involved in creating the file system, you will be asked to confirm your selections.

Figure 14 on page 38 shows the confirmation panel. The DASD target is the first DASD (address here is 544C) for Linux. It will be *dasda1* (a=first DASD, 1=partition #1).

Select < Yes >.

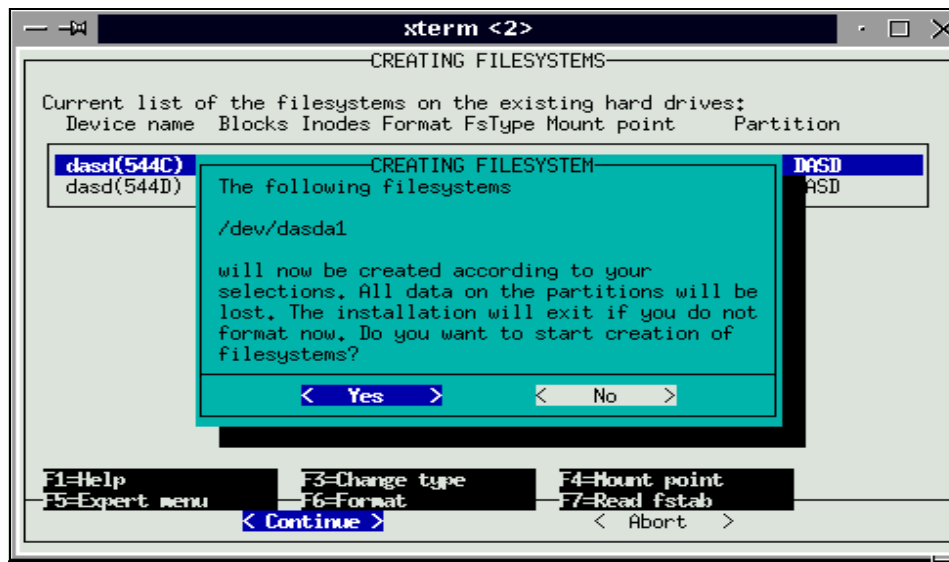


Figure 14. YaST creating file systems - confirmation

4.2.1.8 Load configuration and start installation

So far we have prepared the system for receiving all the data that make up the Linux system. Now we specify what we actually want to install.

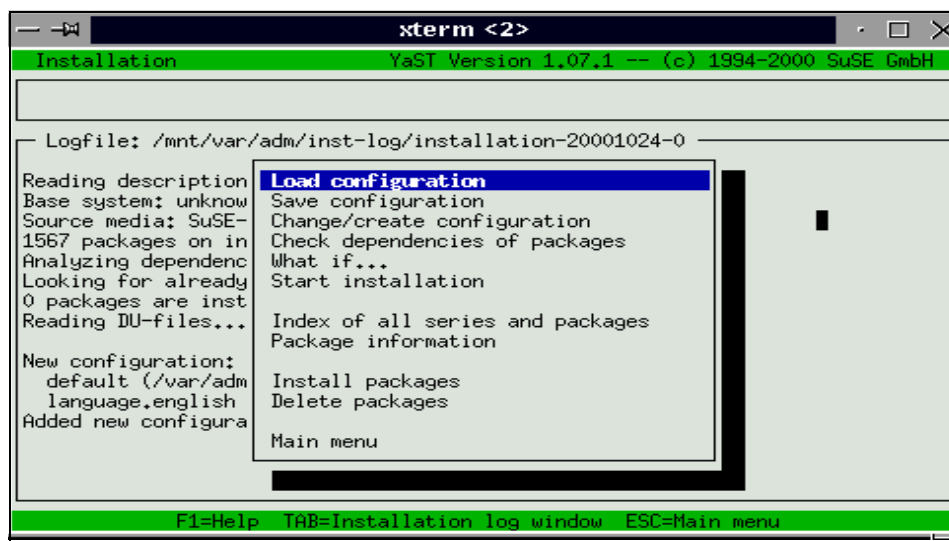


Figure 15. YaST Load configuration

Figure 15 on page 38 allows us to examine the different aspects of the system installation. We do not discuss it here. For further details, refer to *SuSE Linux 7.0 Installation, Networking, Know How*, by Berlich et al.

The installation panel sequence depends on whether you have already been using YaST and YaST has knowledge of what you selected the last time.

Select **Load configuration**.

This selection loads the configuration of the installation you are going to perform based on your previous selections.

4.2.1.9 Select one of the predefined systems

A SuSE Linux system is made of building blocks that are called packages. The selection of packages to install is simplified as YaST provides you with a list of predefined systems.

We chose **SuSE Network oriented system**.

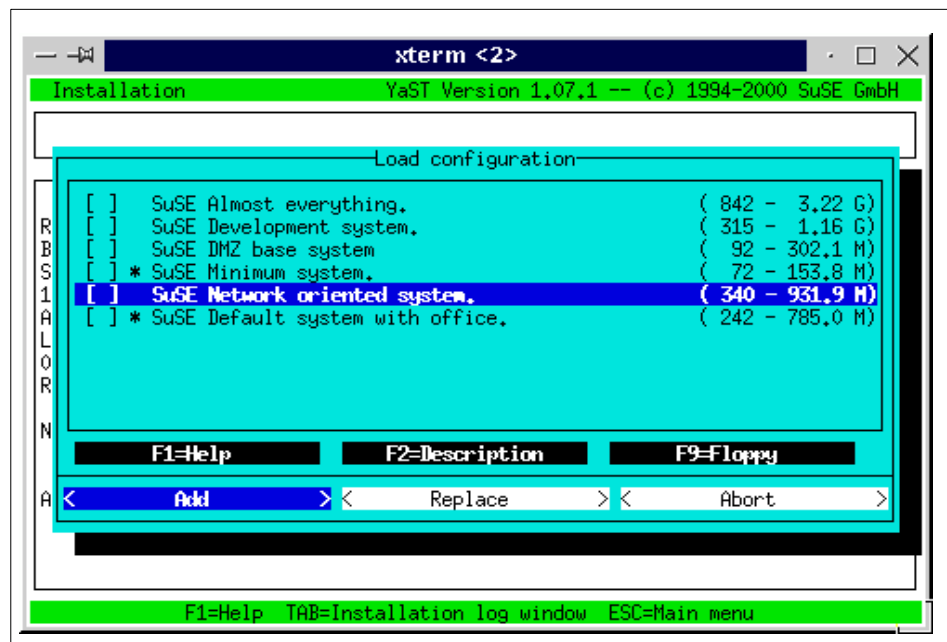


Figure 16. Package selection

4.2.1.10 Start installation

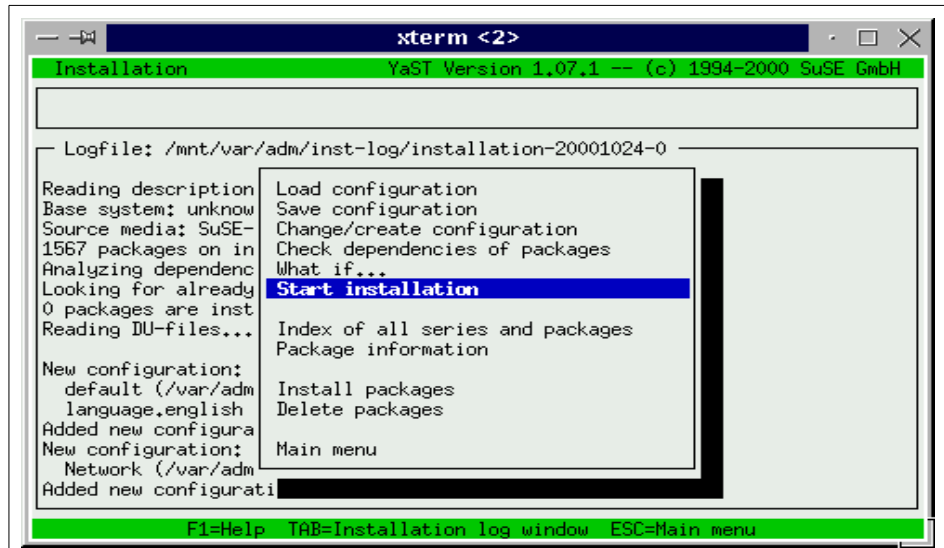


Figure 17. YaST Start installation panel

Since we chose a predefined system, we select **Start installation**.

(Note also the “Check dependencies of packages” option, which automatically checks the consistency of your package selections if you opted to install your own selection of packages, rather than selecting a predefined system.)

4.2.1.11 Progress of installation

YaST shows you the progress of the installation in an information panel. The panel shows which package is currently installing, how many have already been installed, and how many remain to be installed.

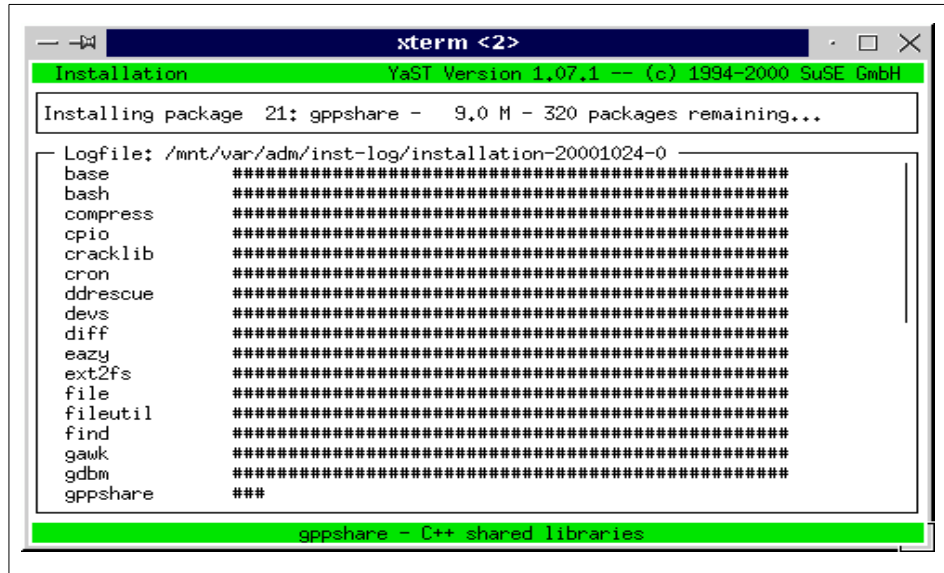


Figure 18. Installation progress panel

The “bar chart”-like graph shows the installation progress.

4.2.1.12 Select kernel

SuSE provides you with two kernels:

- One for VM
- A default kernel

Since we were running native and not under VM, we chose **Default kernel for S/390 (with support for tape IPL)**.

Remark: For an installation under VIF, you have to select the VM kernel.

4.2.1.13 Time zone configuration

The clock the operating system is running on has to be defined. It is based on the system hardware clock plus an eventual offset according to the local time zone.

- Select Local Time: **CET**.
- Adjustment of Hardware Clock: **GMT**.

Note that the adjustment of the hardware clock will probably fail, because in the multi-LPAR environment the hardware clock can be set only at IML time,

before any partition is started. Nevertheless, YaST is a generic tool and has no provision for this special case.

4.2.1.14 Network configuration

Now we have to describe our Network in which the system will run. The defaults are derived from the information previously provided in the first part of the installation. If you are planning to run your system in the same network you configured during the first part of your installation, you accept all the defaults.

- Enter your hostname.

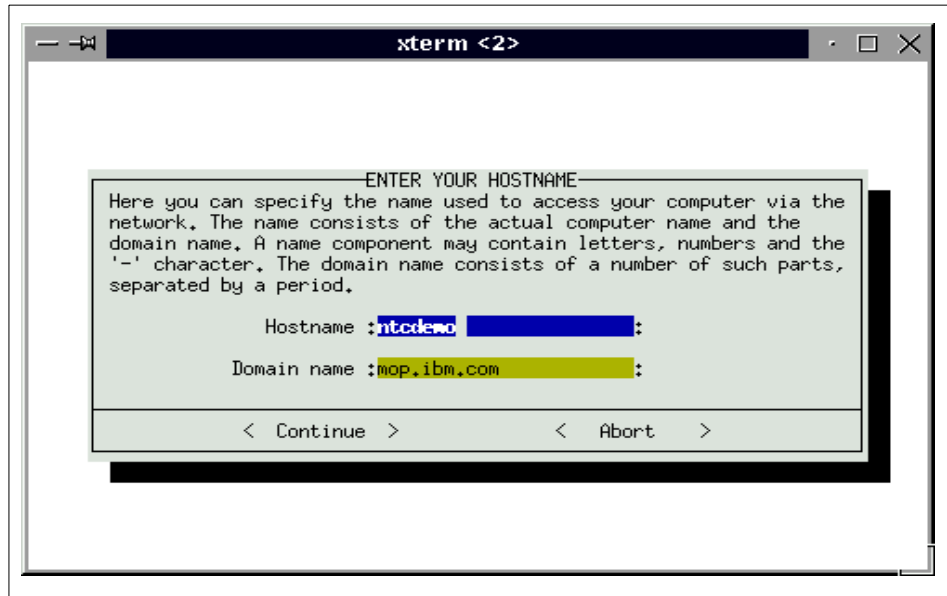


Figure 19. Enter hostname

Linux has prefilled the values we declared in the first part of the install.

Select **Continue**.

- Real Network or Loopback only:

Select **Real Network**.

- DHCP setup.

We did not use DHCP to set up the IP configuration.

Select **No**.

- Type of network

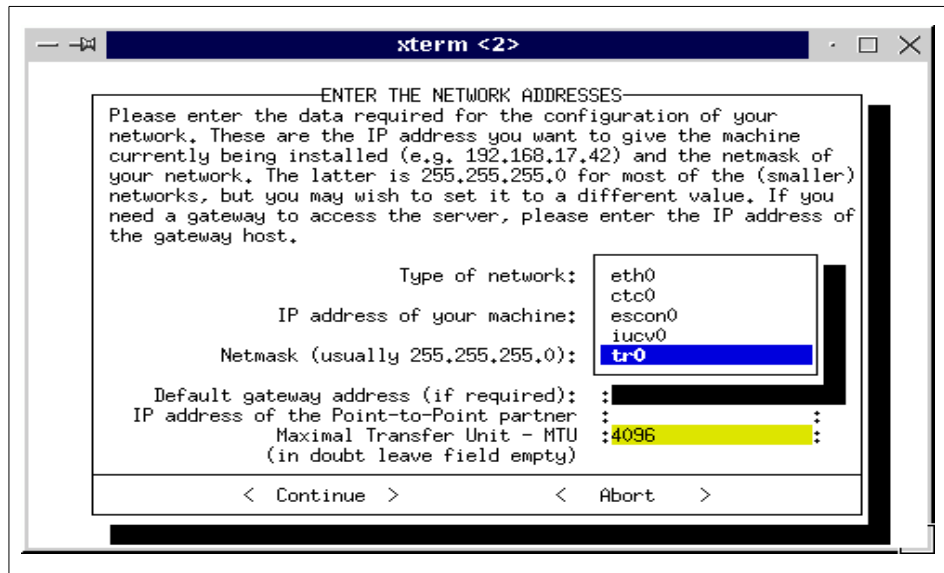


Figure 20. Select type of network

- Network addresses

YaST retrieves the former values of the address parameters if on entering the Type of Network it does not change the initial value you used in the first part of the installation.

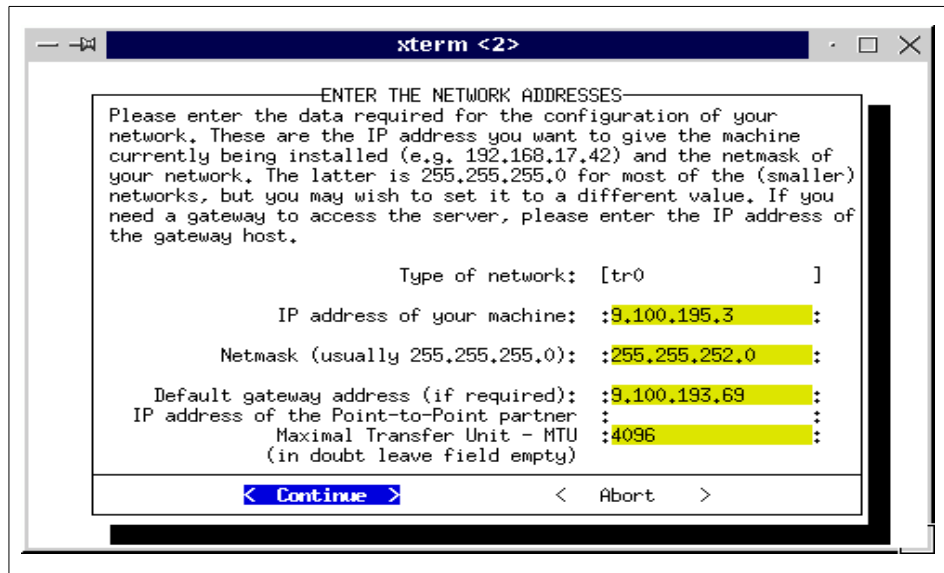


Figure 21. Enter network addresses

- Start inetd at boot-time.

To enable access to network services installed on your system, including Telnet, you must start inetd. You will be asked whether this service should be started at boot-time.

Select **YES**.

- Start Portmapper at boot-time.

Portmap provides services for Remote Procedure Calls (RPC).

Select **YES**.

- Start NFS server.

To provide NFS services, the system has to start a suite of programs at boot-time.

Select **YES**.

- Adjust the News from Address.

This is of secondary importance, but YaST asks for something to be used by the *news* system. Just select **Continue**. The field is prefilled with the full host name and has to be accepted as is.

- Name Server configuration.

Enter the Name Server IP address, if there is one, as well as the search domain name, which in general is equal to your domain name or a subset of it. You can enter more than one value in both fields. Separate the values with a blank space.

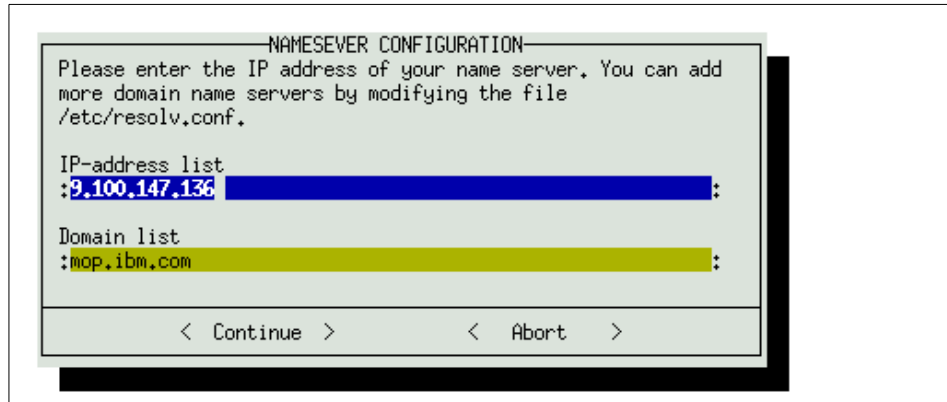


Figure 22. Name Server configuration

- Send Mail configuration.

The Send Mail feature is optional. The choice depends on the kind of services you will be supporting on the Linux server you are installing. You may choose not to have a mail server running by selecting **Do not install /etc/sendmail.conf**.

4.2.1.15 Execute configuration

Now we are done with the configuration and YaST is now installing the configuration files.

You will see the messages shown in Figure 23 on page 46.

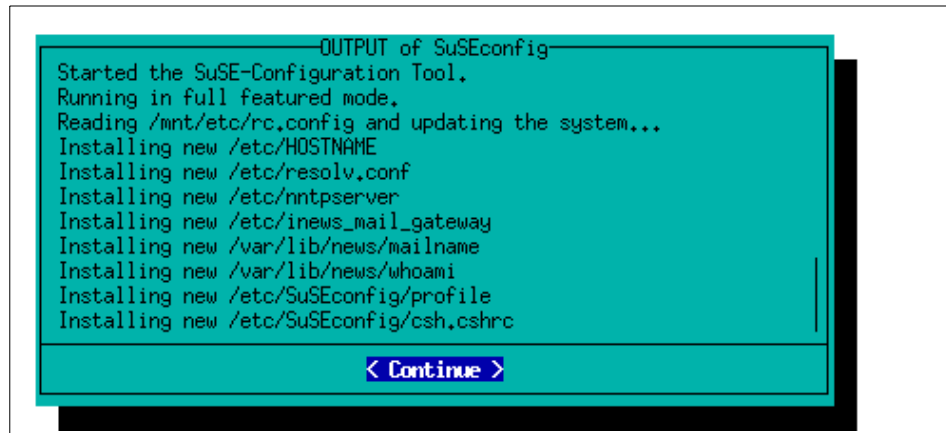


Figure 23. Execute SuSEconfig

4.2.1.16 Shutdown the system

After SuSEconfig execution, the system is installed on your DASD.

You must now stop and bring down the Linux system to reboot it from the newly installed DASD.

Enter either of the following commands:

```
halt or shutdown now -h
```

4.2.2 Performing cleanup and rebooting

You need to execute some cleanup tasks before you can use your system.

4.2.2.1 Booting the system from DASD

On the Hardware Management Console change the Load device address to point to the DASD where the new Linux system is installed. Ensure the load parameter is set to blank.

Activate Load to IPL Linux from the DASD.

Display the Operating System Message on the HMC. After the bootup messages you should be asked to enter a password for root. Post-install scripts will then run to set up the system from the different configuration parameters YaST has collected.

At this point, the installation is done, and the Linux system running on the S/390 LPAR is ready to be used.

4.2.2.2 Swap files in expanded storage

To permanently use the free space in expanded storage as the swap file of your Linux system, do the following:

- Ensure that the *xpram* device driver module is defined to the system. Check into `/etc/modules.conf`. There should be a line like:

```
alias block-major-35 xpram
```

Add this line if it is missing.

- Ensure that *xpram* will be loaded at boot-time with your options:

Check for, or add into `/etc/modules.conf`, a line like:

```
options xpram devs =1 sizes =524288
```

Where:

devs is the number of partitions you wish to set into expanded storage. Each defined partition corresponds to a device ranging from `/dev/slram0` up to `/dev/slram7` (8 partitions is the maximum).

sizes is the partition size, in our case the swap file size, you will be using in expanded storage. We provided all the available expanded storage as a swap file, hence the value of 524288, which is 512 MB expressed in kilobytes.

Note: in testing the parms you might perform the tasks by entering the commands yourself, using the `insmod`, `lsmod`, and `rmmod` commands.

- Save your changes.
- Now add a line into `/etc/fstab` to auto-mount the swap file system when Linux initializes after reboot:

```
/dev/slram0 swap swap pri=5 0 0
```

4.2.2.3 Shut down and reboot your system

After setting up the swap file system configuration, you may shut down the system and reboot it to ensure that all your changes are integrated.

During booting you should see, on the HMC, the messages relative to the expanded storage device initialization.

After booting, check the swap space allocations with the following:

```
free OR top OR cat /proc/swaps
```

Important

Expanded storage does not clear when rebooting Linux, so a swap file system created in expanded storage is not altered and can be reused as is between reboots. However, expanded storage contents are cleared up when executing an IML of the S/390. Consequently, you will lose the swap file system after an IML. You must enter `mkswap` to recreate it and `swapon` to reactivate the swap space.

4.2.2.4 Base installation is finished

In performing all the steps mentioned in this chapter, you have successfully installed a Linux system on S/390 using SuSE Linux. To give you a time estimate: the process took us roughly three hours.

4.3 Installing SuSE Linux 7.0 under VIF

In this section we describe the installation of SuSE Linux under VIF on one of the S/390 native LPARs of our test system.

The base installation of Linux under VIF consists of installing the VIF *Hypervisor* itself and the necessary environment to create, add, delete, and manage multiple guest images of the Linux operating system.

This environment is based on a Linux operating system, the *Master Linux*, running on top of the VIF Hypervisor. Therefore, the base installation includes the installation and configuration of this Master Linux. We used SuSE Linux distribution for S/390 to install the Master Linux.

VIF provides a complete set of commands we can invoke and execute from the Master Linux, for the administration tasks of the other Linux images.

Figure 24 gives an overview of VIF base installation.

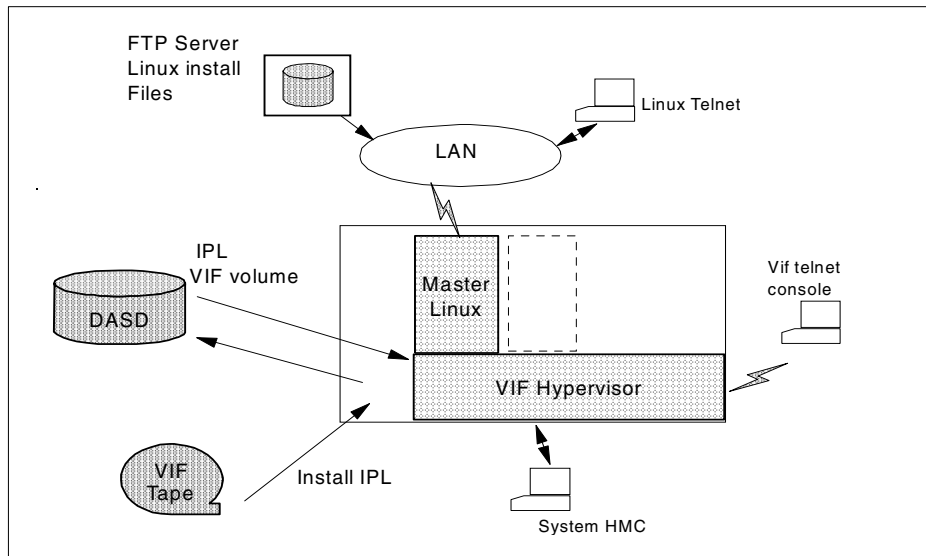


Figure 24. VIF base installation overview

Installing Linux under VIF requires the following:

1. Obtain the product tape and, if one exists, the associated service tape.
2. Identify the installation target DASD, unit *dddd*.

This DASD will become the VIF system boot and initialization volume (we used one volume of a 3390 model 3, Ramac, for our base installation).

3. Collect the necessary definitions of the network, the IP addresses, and the devices to use with your installation.
4. Prepare for the installation:
 - IOCDS and OSA must be initialized and configured to contain the required definitions of the devices to be accessed by the target system LPAR.
 - The DASD volume must have been initialized using ICKDSF and given a *sysres valid* to be used by VIF.
 - The system TOD clock must be properly set.
 - An FTP server, connected to the system network and holding the SuSE Linux for S/390 installation files, must be available.
5. Install VIF on DASD.

Load the LPAR using the product tape to install VIF on DASD. Use the Load parameter to specify the target DASD address.

6. Load from the DASD to IPL the VIF Hypervisor into the LPAR.
7. Access the *Operating System Messages* function of the HMC to initialize the system.

Answer the prompts to enter the definitions relative to the network connections and to download and activate an initial *ramdisk image* of the Master Linux.
8. Via *Telnet*, access the Linux in ramdisk to install the Master Linux on the VIF partition at address 201, using YaST.
9. Reboot the Master Linux from its boot DASD (201).

4.3.1 Preparing for installation

Preparing for the installation involves several tasks.

4.3.1.1 Choose the Type of Network Connection to the LAN

Under VIF there are two possibilities to connect a Linux guest to the LAN. Either use an *external* or an *internal* type of network.

The external type corresponds to a connection to the LAN via physical devices such as OSA2, OSA express, or CTC/escon.

- The internal type implements a point-to-point virtual connection between a Linux guest and the VIF Hypervisor by using, for instance, the Inter User Communication Vehicle feature (IUCV). The VIF Hypervisor TCPIP stack acts as a network gateway to the LAN for the Linux guest.

In our test environment we chose to have a separate access to the LAN for the VIF Hypervisor and for each image of Linux, including the Master Linux.

The VIF Hypervisor, the Linux guest images, and the Master Linux have their own connection to the network via one pair of OSA *I/O device* addresses.

The installation described here does not include the setup required for enabling an internal connection with IUCV.

4.3.1.2 Make SuSE distribution code available on an FTP server

We used a PC installed with SuSE Linux 7.0 where we mounted the CD containing SuSE Linux for S/390.

This PC proved to be helpful for the Telnet sessions with Linux on S/390, mainly when running YaST for installing or configuring the Master Linux and other Linux images. It is sometimes hard to find a Telnet tool that correctly

implements the PF keys required when installing with YaST. In this context Linux on the PC offers a full and compatible environment.

4.3.1.3 Define the access path to the files to be downloaded

To load the *ramdisk* image of the master Linux, the system needs first to retrieve an *installation file* that contains a list of files to transfer from the FTP server.

The files to be downloaded by the VIF installation process are pointed to in the installation file: *suse.ins* (or *susev.ins* for an IUCV connection):

tapeipl.ikr The installation kernel image

initrd The initial ramdisk with YaST

parmfile The default load parameter file

Depending on the way you are mounting the file system of the CD into your FTP server, or if you need to change parameters in the *parmfile*, it could be necessary to adapt the contents of the supplied installation file, as follows:

1. Copy the default installation file, *suse.ins*, found in the *suse/images/* directory of the CD, into a work directory;
2. Edit this copy to insert the appropriate full path name in front of the file names in the list.

The *suse.ins* file contains the following defaults:

```
* SuSE Linux for S/390 Installation/Rescue System (default)
tapeipl.ikr 0x00000000
initrd 0x00800000
parmfile 0x00010480
```

The full pathname of the work directory as well as the filename of the modified copy of *suse.ins* must be collected among other parameters to pass at installation prompts.

4.3.1.4 Collecting the information for the installation process

As a result to the first IPL of the VIF LPAR from the DASD, you will be prompted to enter different information to set up the VIF Hypervisor and the Master Linux. We suggest you collect it before the installation process starts. You will need the following:

- Your choice of sysres valid for VIF
- OSA Device Address and Port Number
- Network type for VIF Hypervisor network
- Network maximum size of a transmitted block (MTU size)

- IP address assigned to VIF Hypervisor and subnet mask
- IP address of gateway to be used by VIF
- Network type for Master Linux
- IP address of Master Linux and subnet mask
- IP address of the FTP server
- The user ID and password for the FTP server
- The full pathname and filename of the installation file

4.3.2 Installing VIF on S/390 DASD

Put the product tape on a tape drive and IPL the LPAR with `AUTOdddd` entered in the Hardware Management Console *load parameter*.

In `dddd`, specify the device address of the DASD where VIF is to be installed.

VIF is copied to the DASD. Wait for the system to enter a *wait state*. It took about 10 to 15 minutes on our G5 to complete (the installation guide says "...up to one hour..."). Observing the tape drive behavior, for the rewind status coming up, could be a good indication of the end of the copy process.

When the wait state occurs, insure that the PSW displays 000E0000 00000000.

To display the PSW, do the following from the Hardware Management Console (HMC):

1. Select the CPC Image where the VIF LPAR is defined.
2. Execute the Single Object Operation task.
3. Find the correct LPAR icon and right-click it.
4. Select logical CP0 in the appearing drop-down list.
5. Find and select **CP Toolbox**.
6. Invoke STOP against the selected CP0.
7. Select **Display or Alter Task**.
8. Expand the list and select **Program Status Word** under Registers.
9. On return, select **Start** to start CP0For.

If the display of the PSW contents is correct, the volume is ready for loading.

4.3.3 Loading VIF Hypervisor from DASD

LOAD the S/390 LPAR from the DASD where VIF has been copied. No load parameters are required.

VIF should come up after one or two minutes.

Activate the *Operating System Messages* on the Hardware Management Console. The HMC displays the following system messages and prompts:

```
11:36:00 EREP records are accumulating for userid $LINOP.  
Reply RESTART at any time to start over  
Enter Hypervisor system residence volume label:  
HCPSED6013A A VM read is pending.
```

Figure 25. First prompt when loading VIF hypervisor for installation

Start the installation of the Master Linux by answering the configuration prompts.

4.3.4 Initializing the Master Linux in RAMdisk

This operation takes place at the System Hardware Management Console. The installation process begins as soon as VIF has completed the load from the DASD.

On the HMC, as a first prompt, you are requested to enter the Hypervisor residence volume label. Subsequent prompts ask for the network parameters required to:

- Define and establish the connection of the VIF Hypervisor and the Linux master Image to the network.
- Connect to the FTP server and find the files to download.
- Boot Linux using a file system in RAM.
- Start the *inetd* services.

Figure 26 is a snapshot of the operating system messages on the HMC when prompting to enter the parameters.

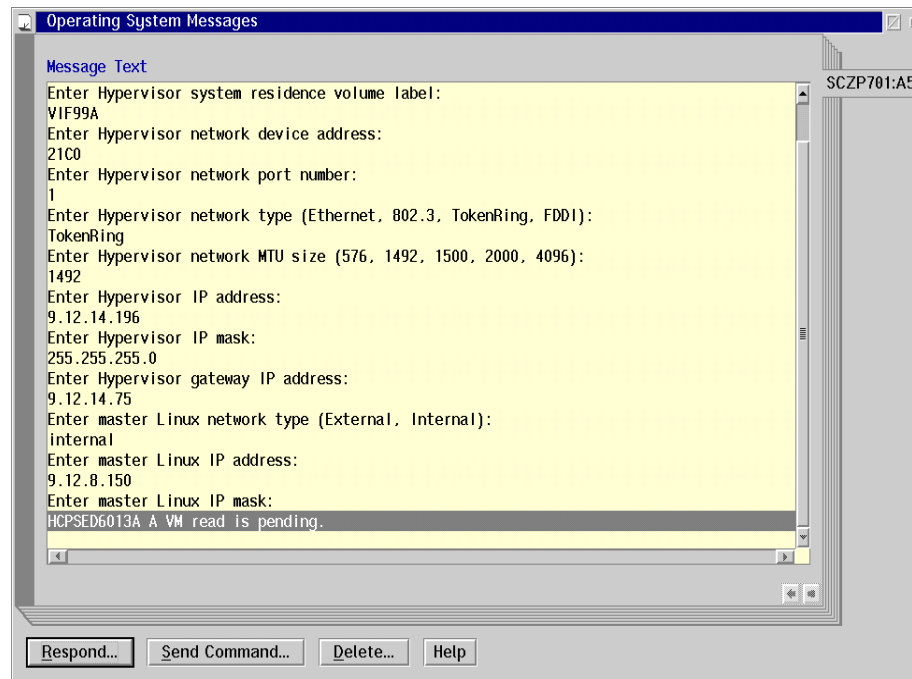


Figure 26. Sample of system messages on the HMC when initializing under VIF

Once the files are correctly transferred from the server, the Linux image is booted from the RAMdisk. Linux initialization starts and asks for the Network device as shown on Figure 27 on page 55.

```

=
==--          Welcome to SuSE Linux 7.0 for IBM S/390          -==
=
First, select the type of your network device:
0) no network
1) OSA Token Ring
2) OSA Ethernet
3) OSA-Express Gigabit Ethernet (experimental)
4) Channel To Channel
5) Escon
6) IUCV (experimental)
Enter your choice (1-6):

```

Figure 27. Selecting the network device

Make the appropriate choice and answer the networking questions. The Linux OS initialization proceeds and completes. It starts up the `inetd` services; in particular, the *telnet* server is started.

From now on the Linux image on RAMdisk is ready to be accessed from a *telnet* session.

The last steps of the installation process use that session to install a “final full” Master Linux on the boot disk (VIF partition 201) using YaST.

4.3.5 Installing the Master Linux on VIF partition 201

The following tasks must be accomplished.

4.3.5.1 Defining the DASD devices available

At startup, VIF automatically allocates two partitions: one with device address 201, one with 203.

As when installing on a native LPAR using SuSE Linux 7.0, you must first tell the system which DASDs are available. The procedure is displayed at login on the *telnet* session screen as shown in Figure 28 on page 56.

```

(none) login: root

>>> >>> >>> >>> >>> >>> SuSE Linux S/390 <<< <<< <<< <<< <<< <<<

1. Enter 'insmod dasd probeonly', then 'rmmod dasd'

2. Choose the device numbers you want to use for SuSE Linux S/390

    !!! BE CAREFUL WHEN SELECTING DASD's - !!!
    !!! YOU MAY DESTROY DATA ON SHARED DEVICES !!!

3. Enter 'insmod dasd dasd=<list of devices>'

    Remember to separate devices by colons (<dev_no>,<dev_no>),
    syntax for ranges is <from_dev_no>-<to_dev_no>
    like
        'insmod dasd dasd=FD00-FD0F,FD40-FD46'

4. Start installation with 'YaST'

5. When YaST has finished, minor modifications of config files may
    be done manually - see documentation for further information.

/root #

```

Figure 28. Telnet session screen

You must *only* use disk 201 to install the master Linux. Disk 203 is already initialized. It contains the VIF code. It will be mounted read-only into the file system to enable access to the set of VIF system management commands.

Enter the commands to allocate the two DASDs. No need for *probeonly*, both device numbers are used. Start executing insmod to load the device driver module into memory and to scan and assign the available DASDs:

```

/root # insmod dasd dasd=201,203
Using /lib/modules/2.2.16/block/dasd.o

```

Check the device's definition in Linux:

```

/root # cat /proc/dasd/devices
0201(ECKD) at (94:0) is dasda:n/f
0203(ECKD) at (94:4) is dasdb:active at blocksize: 1024, 2475 blocks, 2 MB

```

dasda is 840 MB, not shown, since it's not formatted (n/f) yet. You can decide to format dasda now or wait for YaST to do it.

```
/root # dasdfmt -f /dev/dasda -b 4096
```

I am going to format the device /dev/dasda in the following way:

```
Device number of device : 0x201
Major number of device  : 94
Minor number of device  : 0
Labelling device        : yes
Disk label              : INX1 x80405200
Blocksize               : 4096
```

--->> ATTENTION! <---

All data in the specified range of that device will be lost.

Type "yes" to continue, no will leave the disk untouched: yes

Formatting the device. This may take a while (get yourself a coffee).

Finished formatting the device.

Rereading the partition table... done.

Now check the size allocated to dasda:

```
/root # cat /proc/dasd/devices
```

```
0201(ECKD) at (94:0) is dasda:active at blocksize: 4096, 216000 blocks, 843 MB
```

```
0203(ECKD) at (94:4) is dasdb:active at blocksize: 1024, 2475 blocks, 2 MB
```

4.3.5.2 Installing the Master Linux using YaST

Refer to Section 4.2, “Invoking YaST and configuring” on page 33 for more on how to invoke YaST and configure. Here we just tell how we proceeded for the Master Linux we installed.

Selection of the installation medium

You have the choice of using either an NFS server or an FTP server. We chose **Installation from an FTP site**.

Type of installation

We selected **Install Linux from scratch**.

Selecting a swap partition

At this point there are only two predefined partitions. One is reserved for VIF code and the second for the file system. There is not that much space left! We must select **<no swap-partition>** for the Master Linux image.

Creating file systems

To create the file systems you must first select a *mount point* and *format type* for each DASD device. Each setting is done by pressing a PF key. To perform these operations, it is very important that the telnet tool you are using correctly implements the PF key functions.

Figure 29 on page 58 shows the file system prompt screen.

```

-----CREATING FILESYSTEMS-----
Current list of the filesystems on the existing hard drives:
Device name Blocks Inodes Format FsType Mount point Partition
+-----+-----+-----+-----+-----+-----+
| dasd(0201) 863988 4096 Normal ext2 / S390 DASD |
| dasd(0203) 2456 8192 No ext2 /usr/local/vif390 S390 DASD |
+-----+-----+-----+-----+-----+-----+

+-----+-----+-----+-----+-----+
| Creating filesystem on "/dev/dasda1"... |
+-----+-----+-----+-----+-----+

F1=Help          F3=Change type      F4=Mount point
F5=Expert menu   F6=Format           F7=Read fstab

```

Figure 29. SuSE Linux for S/390 file systems creation prompt screen

Select the DASDs by highlighting the line.

Press PF4 to select the mount point of the highlighted DASD.

You need to enter the mount point /(root) for dasd(201). We selected **/usr/local/vif390** for dasd(203), which is the directory entry we chose for the VIF command.

PF6 lets you then choose the format for the highlighted DASD. Select **Normal** for dasd(201) and **no (default)** for dasd(203).

FTP server settings

We chose an installation from an FTP site. We must now enter the server configuration: Server name or IP address and the directory containing the SuSE Linux CD (mount point of the directory, i.e., /mnt/cdrom, but not /mnt/cdrom/suse!).

Uncheck the Anonymous FTP check box and enter userid and password.

Press PF5 to check the connection and then PF10 to proceed.

Loading the configuration

Select **Load configuration** in the presented tiled menu to make the software selection.

Selecting the base software

In our installation under VIF we decided to reserve the Master Linux for system management functions only. We selected **SuSE Minimum system**, in the list of the default configurations presented after we selected **Load configuration**. Other packages will be individually installed later if necessary.

Starting the installation

Select **Start installation** in the tiled option menu to start.

Resolving dependencies

Unresolved dependencies in packages to be installed require decisions; we selected **AUTO** to validate.

Waiting for installation complete message

When the completion message appears, you can proceed as for a normal install of Linux:

- Select the **default kernel** to install (the only possibility)
- Adjust the hardware clock to **local time**.
- Set home hostname and address.
- Select **real network**.
- Do *not* set DHCP client.
- Set Network address and type (tr0; we are connected using OSA2 on Token Ring). The information entered earlier is displayed. Select **Start inetd** TCP/IP daemon.
- Select **Yes** for Port Mapper and NFS server.
- Select **Name server** and enter the name server configuration.
- Do *not* accept installation of the send mail configuration file.

YaST terminates now. The tool *SuSEconfig* is automatically run. All the settings collected into YaST are written to the various configuration files.

At this point DASD 201 is loaded but not ready yet for booting Linux. The *parmfile* might need to be adjusted and device 201 must be, at least, defined as the boot device for the Master Linux.

Note: Do not exit the session yet!

4.3.6 Setting partition 201 as boot for Master Linux

To boot the Master Linux from a newly installed configuration, and automatically start it when VIF starts, you must perform the following steps:

- Check and eventually adapt the new parmfile.
- Run `silos` if the parmfile is modified.
- Run a `vif image set` command to set the boot device.
- Re-IPL the VIF LPAR.

At completion, YaST lets the newly created file systems `umount`. It will be necessary to temporarily `mount` them during the process. The recommended mount point is `/mnt`. Do not forget to execute `umount` prior to exit. Otherwise you might get an error: "....file not cleanly unmounted...." at boot of Linux.

Check the parmfile

There is normally no need to modify it when using OSA2, as we did.

```
mount /dev/dasda1 /mnt
cat /mnt/boot/parmfile
dasd=0201,0203 root=/dev/dasda1 noinitrd
```

Run Silo

This is required only if the parmfile is modified. YaST has already run it.

```
cd /mnt/boot
silo -f image -p parmfile -d /dev/dasda -b iplrckd.boot -t2
```

Unmount dasda

We need a clean mount point for the next step.

```
cd
umount /mnt
```

Set the boot device for master linux: LINUX0

For executing this step, LINUX0 is the name VIF uses to identify the Master Linux image.

Run a `vif image set` command to allocate the boot device to LINUX0. At first you must get access to the directory where the vif code is. The command requires the Hypervisor IP address as a first input parameter.

```
mount /dev/dasdb1 /mnt
cd /mnt
vif 9.12.18.35 image set linux0 boot 201
LINUX0 boot changed to 201
Command Complete
cd
```

```
umount /mnt
```

Stop and re-IPL the VIF LPAR

Now stop the currently running Linux system with `halt`. Watch the Operating system messages on the HMC for completion.

LOAD VIF from the DASD with no parameter (Null in LOAD PARMS).

VIF starts immediately and starts the Master Linux, LINUX0. This shows up in Operating system messages, on the Hardware Management Console.

You are requested to set a *password* for *root*, then post-install scripts run and the Master Linux system starts, as any normal Linux running native on an S/390 LPAR.

4.3.7 Making access to VIF commands more convenient

There are good reasons for creating an environment to make VIF commands more convenient to use:

- The path to the `vif` command is not in the system default search path.
- You are required to enter the Hypervisor IP address in the parameter for every `vif` command.

Here are two possible solutions:

1. Create a shell command called `vif` under `/usr/sbin`:

```
vi /usr/sbin/vif
#!/bin/bash
/usr/local/vif390/vif xx.xx.xx.xx $@
```

Make it executable:

```
chmod 755 /usr/sbin/vif
```

2. Add an *alias* definition in `.profile` of user `root`. This makes `root` the only user authorized to execute the `vif` commands.

```
alias vif="/usr/local/vif390 xx.xx.xx.xx "
```

Where `xx.xx.xx.xx` is the IP address of the VIF Hypervisor, and `/usr/local/vif390` is the mount point we chose for the `vif` code partition, during installation, when creating the file systems.

Now you just need to type `vif` followed by the appropriate substrings and parameters when entering a command.

4.4 Implementing the first Linux image under VIF

In the previous section we saw how we installed VIF and how we set up the Virtual Image Facility environment with SuSE Linux 7.0.

The VIF Hypervisor is now operational. The Master Linux, LINUX0, is up and running and provides the necessary administration interface to the VIF Hypervisor, in the form of a set of `vif` commands. We are ready to create new images of Linux OS running under VIF.

We are now going to see how we used `vif` commands to add LINUX1, the first Linux image, under VIF. We do not describe each command in detail, but rather give snap shots of the steps we executed to set up the new guest on our system.

You will find the list of all `vif` commands in *S/390 Virtual Image Facility for Linux Guide and Reference*.

The main steps for implementing the first guest image are:

- Allocate space for paging and for Linux images.
- Create the new guest image.
- Add a network connection to the guest image.
- Check the image settings.
- Start and install the Linux image.
- Install a Linux configuration using YaST.
- Run post-install to complete the installation.

4.4.1 Allocating more disk space to the VIF system

It is obvious that the operating space allocated so far for VIF and LINUX0 is rather tiny. Adding new guests would certainly create problems due to the lack of available disk space, and more so if the LPAR, where VIF runs, is short on real storage. You will be facing paging problems and abends.

So, adding a Linux guest image requires, first of all, additional disk space.

Two types of disk space need to be increased:

Paging space

Image space

Paging space is required by the VIF Hypervisor to operate when the number of images increases. Image spaces are for the file systems of the Linux images.

For sizing guidelines of these spaces, refer to *S/390 Virtual Image Facility for Linux Guide and Reference*.

You must prepare two sets of volumes: one for paging and one for Linux images. At least one volume is required for each space to install a first image. They must be formatted using ICKDSF and fully identified by their VOLID.

Figure 30 describes VIF disk space usage.

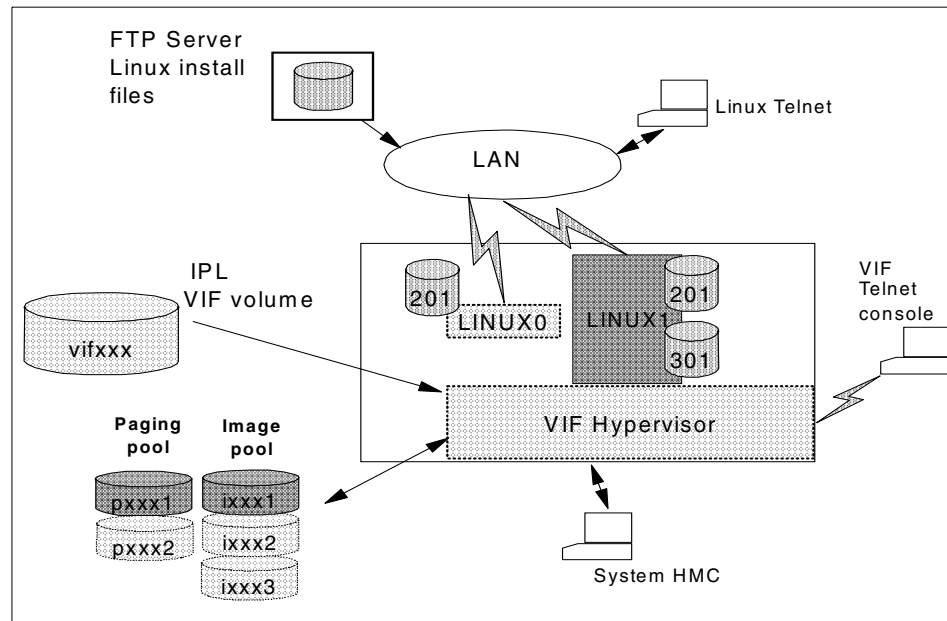


Figure 30. VIF disk space usage

4.4.1.1 Adding paging space to the VIF Hypervisor

Enter the commands from a telnet session with LINUX0:

```
# vif hyp vol add paging 201B VF201B
Command may take up to 32 minutes; please wait
PAGING volume VF201B added
Command complete
```

Checking:

```
# vif query paging
160 of 3168 MB of Hypervisor paging space in use
Command complete
```

4.4.1.2 Adding space for Linux Images

```
# vif hyp vol add image 2006 VF2006
IMAGE volume VF2006 added
Command complete
```

Check the space by querying the free space available:

```
# vif hyp vol map free
VolID      Gap (MB)
VF2006      2347
Command complete
```

4.4.2 Creating the new guest image LINUX1

To create a new Linux image you need to execute these tasks:

- Check the free space available for a new Linux image.
- Create the Linux guest image.
- Assign a partition for the Linux image file system.
- Assign a partition for swapping to the Linux image.
- Display the remaining free space.

4.4.2.1 Create a Linux image on VIF

```
# vif image create linux1
Image LINUX1 created successfully
Command complete
```

4.4.2.2 Assign a partition for Linux1 image boot file system

```
# vif par create linux1 201 1173
PARTITION CREATE completed successfully
Command complete
```

4.4.2.3 Assign a swap partition (optional)

This is optional, depending on the real storage size assigned to the LPAR running VIF and the amount of logical storage you will set for this image.

```
# vif par create linux1 301 1173
PARTITION CREATE completed successfully
Command complete
```

4.4.2.4 Display the remaining free space

```
# vif hyp vol map free
No FREE space exits
Command complete
```

4.4.3 Adding a network connection to Linux image LINUX1

There are two types of network connection available to Linux running under VIF: internal, using IUCV, or external via physical devices, such as an OSA2 card.

In the following examples we use a pair of OSA2 device addresses for directly connecting LINUX1 to the network. We just need to specify the first device address in the command. If we were using an internal connection via IUCV, the command would require the IP addresses of both the VIF Hypervisor and the LINUX1.

```
# vif image net linux1 add 908
Command complete
```

Quick checking:

```
# vif q net
```

LINUX1 has an external network connection via devices 908-909

4.4.4 Checking Linux image LINUX1 settings

LINUX1 image is now fully defined. Before proceeding with the Linux software configuration installation, query the newly created image. You might need to increase the number of CPUs or the storage size of the Linux image, or possibly make some modification.

```
# vif query image linux1
LINUX1 has 1 CPUs and 64M of storage and boots default
LINUX1 has an external network connection via devices 908-909
LINUX1 has a 1173 MB read/write partition as device 201 on volume VF2006
LINUX1 has a 1173 MB read/write partition as device 301 on volume VF2006
Command complete
```

By default VIF assigns 1 CPU and 64 MB of memory when creating images. We changed the storage size to 512 MB:

```
# vif image set linux1 sto 512M
```

Figure 31 on page 66 displays the result of a global system query.

```

linux0:~ # vif query all
LINUX0 has 1 CPUs and 64M of storage and boots 201
LINUX0 has an external network connection via devices 906-907
LINUX0 has a 843 MB read/write partition as device 201 on volume VF201A
LINUX0 has a 3 MB read-only partition as device 203 on volume VF201A
LINUX1 has 1 CPUs and 512M of storage and boots default
LINUX1 has an external network connection via devices 908-909
LINUX1 has a 2347 MB read/write partition as device 201 on volume VF202C
672 of 5504 MB of hypervisor paging space in use
2347 of 9388 MB of Linux image partition space in use
VIF performance: CPU is Green, Paging is Green, I/O is Green
Hypervisor uses IP address 9.12.18.35 via device 904

The following Linux images are active:
LINUX0
Hypervisor level: V1-R1-M0-SL0100
Last boot on 2000-11-10 at 20:52:11
System has been up for 2 days, 17 hours, 26 minutes
00000024 error records are pending
Command Complete

```

Figure 31. Result of a global system query after install of LINUX1

4.4.5 Starting and installing the Linux image

When we first installed the Master Linux, LINUX0, we downloaded the RAMdisk file system image and other installation files from the FTP server. We initially used this RAMdisk to boot the Master Linux and start the installation. The same process is used here for LINUX1.

The `vif image start` command will boot LINUX1 with defaults from SuSE installation files, downloaded at Master LINUX0 installation time.

Perform the following steps to install the new image you created:

- Start the new image to boot LINUX1 from RAMdisk.
- Access the VIF Hypervisor with Telnet to pre-install.
- Install the SuSE Linux configuration with YaST.
- Change the Linux boot device and reboot.
- Run the post-installation.

4.4.5.1 Start the new LINUX1 image

To boot LINUX1 from the current boot device, which is the default here (the RAMdisk), enter the command:

```

# vif image start linux1
Linux image LINUX1 started
Command Complete

```


4.4.5.2 Telnet VIF Hypervisor to start the installation

It's the first time in the installation process that we directly access the VIF Hypervisor by means of a Telnet session.

As opposed to the Master Linux LINUX0, the pre-installation process of any new Linux guest image is performed through a Telnet session with the VIF Hypervisor, not the Hardware Management Console.

Once again it could become “tricky” to communicate with this session if the Telnet tool you are using is not compatible. For instance, the one we used required that we set the *echo* function to echo our keyboard typing in the entry field, on the screen.

Connect the Telnet session with the VIF Hypervisor using the Hypervisor IP address.

Press **Enter** once when you are invited to “...press break key...”.

Enter `Logon linux1`.

Press **Enter** again after “...reconnected...” to display the prompt menu.

Enter the network type as shown in Figure 32. We chose 1, and proceeded with a new Linux install as usual.

```
S/390 Virtual Image Facility for LINUX --VIF      --PRESS BREAK KEY TO
.logon linux1

LOGMSG - 21:02:27 UTC Wednesday 06/14/00
*** S/390 Virtual Image Facility for LINUX ready!
FILES:   NO RDR,   NO PRT,   NO PUN
RECONNECTED AT 15:30:24 UTC MONDAY 11/13/00
!
.
First, select the type of your network device:
0) no network
1) OSA Token Ring
2) OSA Ethernet
3) OSA-Express Gigabit Ethernet (experimental)
4) Channel To Channel
```

Figure 32. Network type selection

4.4.5.3 Telnet to LINUX1 and install SuSE Linux with YaST

Inetd should have started the Telnet service in LINUX1. You can now connect to LINUX1 in RAMdisk to install the configuration of your choice of SuSE Linux for S/390.

We chose **SuSE Network oriented system** from the list of predefined configurations.

4.4.5.4 Change LINUX1 boot device and reboot

After finishing installation with YaST, change the LINUX1 boot configuration to replace the default device (RAMdisk) and specify 201 as the new boot device.

First, change the boot device in VIF with the following command:

```
# vif image set linux1 boot 201
LINUX1 boot changed to 201
Command Complete
```

The LINUX1 image is ready to boot from device 201. Stop and restart the LINUX1.

```
# vif image stop linux1
Linux image LINUX1 stopped
Command Complete
```

```
# vif image start linux1
Linux image LINUX1 started
Command Complete
```

4.4.5.5 Run post-installation of LINUX1 to complete

The first Start of the new image does not make LINUX1 operational yet. SuSE post-installation scripts need to be run to finish the installation:

1. Log on to the LINUX1 image through the Hypervisor Telnet screen.
2. As you did before for LINUX0 (but on the HMC), enter a password for root, when you are prompted.
3. Consequently, SuSE post_install scripts execute to update the various configuration files from the YaST settings. Watch the messages logged by the scripts on the Telnet session screen until you see "...enter root password to start...".

You can disconnect from the Hypervisor Telnet session and open one with LINUX1 to start working with Linux.

The LINUX1 image is now operational!

4.4.6 Cloning a Linux image from LINUX1

With this process you can add a new Linux image, LINUX2 for instance, by duplicating the file system from LINUX1 partition 201 to LINUX2 partition 201. You do not need to run YaST to install LINUX2, but you do need to modify the copied configuration files for the network connection definition of LINUX2.

It is obvious that before starting this operation you must verify that there is the required amount of free space left in the image space and that the paging space is wide enough.

To add a new LINUX2 image, here is how we proceeded on the test system:

1. First stop LINUX1 or insure that it is stopped:

Enter `halt` at a LINUX1 Telnet session, or logon to LINUX1 from the VIF Hypervisor Telnet and enter the same command.

2. Create a new guest image, LINUX2, into VIF:

```
# vif image create linux2
Image LINUX2 created successfully
Command Complete
```

3. Create a partition for LINUX2 swap:

```
# vif par create linux2 301 1173
PARTITION CREATE completed successfully
Command Complete
```

4. Add the network connection for LINUX2:

```
# vif image net linux2 add 90C
NETWORK ADD completed successfully
Command Complete
```

5. Then copy the Linux file system on partition 201 from LINUX1 to LINUX2. VIF automatically creates the partition, of the same size, at the specified address.

```
# vif part copy linux1 201 linux2 201
Command may take up to 84 minutes; please wait
#
```

This took about 35 minutes, not 84, as specified in the returned message.

6. Now start the LINUX2 image. It will boot from the default, which is the RAMdisk, since we have not specified a boot device for LINUX2 yet.

```
# vif image start linux21
Linux image LINUX2 started
Command Complete
```

7. The LINUX2 image is now running with the default boot. We need to logon to LINUX2 through the Hypervisor Telnet session to answer pre-install prompts until *inetd* can start .

- Telnet to VIF Hypervisor and enter logon as linux2.
- Answer the network questions to configure the connection of LINUX2.
 - Network device type and address
 - IP address and subnet mask
 - MTU size

8. At this point we are able to establish a direct Telnet session with the LINUX2 image in RAM to define the DASDs LINUX2 will use. We must load the DASD driver to access both the copied file system and the swap partition, with the following command:

```
insmod dasd dasd=201,301
```

After entering the command, partition 201 is defined as dasda and 301 as dasdb, to the Linux kernel.

9. We must *format* partition 301 and make the *swap* file system, using:

```
dasdfmt -b 4096 -f /dev/dasdb (It Took 10 mins).  
mkswap /dev/dasdb1
```

10. Then mount partition 201 (the "cloned" partition) on /mnt using:

```
mount /dev/dasda1 /mnt
```

11. The cloned partition is now accessible for modification of the configuration files for the LINUX2 image. We have to change the network connection definition to this:

```
vi /mnt/etc/HOSTNAME
```

Change the hostname *name1* to *name2*.

name1 and *name2* are the hostnames for LINUX1 and LINUX2, respectively.

```
vi /mnt/etc/rc.config
```

Change all the occurrences of the IP address of LINUX1 with the IP address of LINUX2, and all the occurrences of the LINUX1 hostname with the LINUX2 hostname. It could include changing the occurrences of the associated subnet mask, too.

We are considering here that LINUX1 and LINUX2 are both using the same type of device and connection to connect to the network. We would have needed to change the network type in /mnt/etc/rc.config and modify /mn/etc/modules.conf to use a different device driver. You might have to

change from the OSA2 Token Ring to OSA express fast ethernet, for example.

12. You must now unmount the file system before exiting the session with LINUX2:

```
umount /mnt
exit
```

13. Now return to a VIF management session (Telnet session with LINUX0) to change the LINUX2 image boot parameter from default to boot from partition 201:

```
# vif image set linux2 boot 201
LINUX2 boot changed to 201
Command Complete
```

14. Stop and start the LINUX2 image to boot it from the “cloned” partition at address 201:

```
# vif image stop linux1
Linux image LINUX1 stopped
Command Complete
```

```
# vif image start linux1
Linux image LINUX1 started
Command Complete
```

15. Open a Telnet session with the LINUX2 image that has now started and run /sbin/SuSEconfig, since rc.config has been manually modified.

```
tplexpdb:~ # SuSEconfig
Started the SuSE-Configuration Tool.
Running in full featured mode.
Reading /etc/rc.config and updating the system...
Installing new /etc/inews_mail_gateway
Installing new /var/lib/news/mailname
Installing new /var/lib/news/whoami
Installing new /etc/SuSEconfig/profile
Installing new /etc/SuSEconfig/csh.cshrc
Executing /sbin/conf.d/SuSEconfig.groff...
Executing /sbin/conf.d/SuSEconfig.kdm...
Executing /sbin/conf.d/SuSEconfig.pam...
Executing /sbin/conf.d/SuSEconfig.perl...
Executing /sbin/conf.d/SuSEconfig.sendmail...
Installing new /etc/sendmail.cf
Executing /sbin/conf.d/SuSEconfig.susehilf...
Executing /sbin/conf.d/SuSEconfig.susehilf.add...
Executing /sbin/conf.d/SuSEconfig.susewm...
Executing /sbin/conf.d/SuSEconfig.yppclient...
Processing index files of all manpages...
```

```
Finished.  
tplexpdb:~ #
```

Guest image LINUX2 is operational.

4.4.7 Hints and tips

Any defined Linux image is automatically started when you stop and start the VIF Hypervisor, even if there is no network connection or partition, or neither, assigned to this image. However, such a “blind” Linux image consumes VIF storage resource like real storage and paging space. In this event, be sure to have enough paging and/or image space available.

To avoid a not properly unmounted DASD partition (Linux message “...partition not cleanly unmounted...,” plus the time required for the file system check to run), we recommend that you *shut down* all started LINUXx images with the regular `halt linux` command (except LINUX0, because you need it to shut down VIF) prior to shutting down the VIF Hypervisor.

Note that SuSE Linux exchanges the DASD names between the root file system DASD and the swap DASD. Don't be surprised if, after rebooting following an installation, you do not find what you expect to be dadsa. This is because SuSE Linux seems to sort the device address values in descending order. In the *boot parmfile* built by SuSE Linux, DASD address 301 comes before DASD address 201, therefore 301 is assigned `/dev/dasda` and 201 `/dev/dasdb`. Check the contents of the *dasd=* parameter, in the boot parmfile, and also find out the boot device name in the *boot=/dev/dasd...* parameter. There is no reason to change it. Just be aware of it.

4.5 Adding a network connection device to an active Linux

To test the execution of intensive queries and the transfer of large data blocks between the UDB instance on Linux S/390 and the DB2 instance on OS/390, we had to install three types of devices to connect the machines across the network.

On each Linux machine, on a native LPAR, we enabled three types of network devices supported today with Linux on S/390:

- The OSA2 card with a Token Ring connection
- The OSA Express Gigabit Ethernet
- The CTC/escon channel

The OSA2 card Token Ring connection was installed and configured at the beginning. All SuSE Linux on S/390 installations were performed the same

way, using a pair of device addresses on an OSA2 card with a Token Ring port. It was the only connection available.

We had to add the other devices to the Linux machines on the fly. Here is how we did it in the SuSE Linux on S/390 environment.

4.5.1 Network configuration files under SuSE Linux for S/390

Under SuSE Linux on S/390, there are two files to edit and modify to manually change the network configuration in order to activate a disabled network device:

`/etc/rc.config`

`/etc/modules.conf`

`rc.config` contains the system parameters collected by YaST that SuSE uses to configure the Linux instance. Among the parameters are those associated with the network, such as the network device name, the IP addresses, the subnet mask, etc.

`modules.conf` specifies device driver modules (and parameters), not built into the kernel, that need to be loaded into memory at initialization time to automatically activate the associated devices.

Linux commands like `insmod`, `rmmod`, and `lsmod`, can be used as bottom line commands to manually load, unload, or list device driver modules. Refer to standard Linux manuals for details on module-related commands.

It is important that after making manual changes in `/etc/rc.config` you also execute *SuSEconfig* to validate these changes.

4.5.2 Adding a CTC/escon connection

We are assuming that all the preparation work is done. The ESCON channels are correctly interconnected, the CTC *I/O devices* are configured in the IOCDs and are made available to the Linux LPAR, and the remote side is ready.

To add a CTC/escon point-to-point connection to Linux, you need:

- The ESCON channel addresses; they must be an even/odd pair of addresses.
- The Linux device name, in the range of `escon0` to `escon7`.
- The Local IP address for this point-to-point connection.
- The Peer IP address.
- The subnet mask, i.e., 255.255.255.0.

- The MTU size.

Before making final changes to the configuration files, you should activate the connection manually to validate the parameters you collected. You can use this set of commands:

```
insmod ctc setup=\"ctc=0,0x0300,0x0301,escon0\"
ifconfig escon0 15.1.1.7 pointopoint 15.1.1.4 netmask 255.255.255.0 mtu
32768 up
```

For the connection to be automatically activated when rebooting, you must update `/etc/module.conf` and `/etc/rc.config`. Enter the networking parameters and the ESCON device name you are activating. Find the same “name=value” pairs as shown in the sample of `rc.config`. There are two sets of values in the sample: one is for device `tr0`; the second is the update to be done to add the `escon0` connection.

Edit `rc.config` as shown in Figure 33.

```
NETCONFIG="_0 _1"
#
# IP Addresses
#
IPADDR_0="9.12.18.47"
IPADDR_1="15.1.1.7"
IPADDR_2=""
IPADDR_3=""

#
# network device names (e.g. "eth0")
#
NETDEV_0="tr0"
NETDEV_1="escon0"
NETDEV_2=""
NETDEV_3=""

#
# parms for ifconfig, simply enter bootp or dhcpclient to use the
# respective service for configuration
# sample entry for ethernet:
#
IFCONFIG_0="9.12.18.47 broadcast 9.12.18.255 netmask 255.255.255.0 mtu 4096 up"
IFCONFIG_1="15.1.1.7 pointopoint 15.1.1.4 netmask 255.255.255.0 mtu 32768 up"
```

Figure 33. Adding TCP/IP configuration for CTC/escon channel connection

Note: `IFCONFIG_0` and `IFCONFIG_1` must be on one line only.

To update `/etc/modules.conf`, either modify existing definitions or add two new definitions like the following:

```
alias escon0 ctc
```



```
options ctc setup="ctc=0,0x0300,0x0301,escon0"
```

Use your values for the device addresses and the ESCON device name.

With our configuration we found that when there are too many CTC/escon channels defined to the LPAR, in the IOCDs, Linux is only capable of assigning the first 8 pairs of channels in sequence. We could not use a channel with an address out of the range of the 8 first pairs, even when adding the *ctc=no auto* parameter in the boot *parmline*. So we modified the IOCDs to only include the range of I/O devices we needed.

4.5.3 Adding an OSA Express GbE device connection

The process for adding an OSA Express GbE connection is similar to the one we used for the CTC/escon channel device connection, i.e., specify the following:

- The device driver modules to be loaded
- The I/O device addresses to use for connecting
- The TCP/IP configuration of the connection

The OSA Express driver on Linux S/390 requires two modules to run: *qdio*, the base protocol driver module, and *qeth* to drive the Gigabit Ethernet interface.

Among the load parameters to pass to the *qeth* module are the I/O device addresses to be used by the connection. An OSA Express GbE connection requires three I/O device addresses to operate (control read, control write, and data). They need to be defined in the IOCDs and made available to the LPAR.

To install a configuration that worked on our Linux LPAR we discovered that we had to really reserve four I/O device addresses, in sequence, on an even boundary. The last one is useless, but must be available. Consequently we adjusted the IOCDs of the LPAR.

We concluded that for connecting with an OSA GbE, Linux requires two pairs of I/O device addresses, in sequence, and the first one must have an even value. Other combinations were rejected by the *qeth* module as invalid (...*"Device or resource busy...Hint: this error can be caused by incorrect module parameters, including IO or IRQ parameters"....*).

To manually load *qdio*, we entered:

```
insmod qdio
```

Prior to loading the GbE interface driver module, we check the device definitions we put in place using `qeth` with the *probeonly* option:

```
tplexpd7:/proc # insmod qeth probeonly
Using /lib/modules/2.2.16/net/qeth.o
```

We then displayed the result of the device definitions stored in `/proc/qeth`:

```
tplexpd7:/proc # cat qeth
```

devnos (hex)	CHPID	cardtype	device	chksum	prio-q'ing	rtr	bufsize
-----	----	-----	-----	-----	-----	---	---
0E04/0E05/0E06	xF0	OSA-Expr. GbE	eth0	no	always q 2	no	64K

Figure 34. Result of `cat/prcc/qeth`

Note that `/proc/qeth` would not have been created if `ismod qeth` had not worked.

We loaded `qeth` without specifying any parameter, since we had only defined four I/O devices in the IOCDs available to the LPAR. We let the module use the *autosensing* of the devices.

```
insmod qeth
```

We then updated `/etc/modules.conf` by adding two lines:

```
options qdio
options qeth"
```

With no parameters, no alias, `qeth` automatically defines `eth0` as the device name for this connection.

We finally added the new TCP/IP configuration parameters to `/etc/rc.config` and ran `SuSEconfig`.

```

NETCONFIG="_0 _1 _2"

#
# IP Addresses
#
IPADDR_0="9.12.18.46"
IPADDR_1="15.1.1.7"
IPADDR_2="14.1.1.7"
IPADDR_3=""
#
# network device names (e.g. "eth0")
#
NETDEV_0="tr0"
NETDEV_1="escon0"
NETDEV_2="eth0"
NETDEV_3=""

IFCONFIG_0="9.12.18.46 broadcast 9.12.18.255 netmask 255.255.255.0 up"
IFCONFIG_1="15.1.1.7 pointopoint 15.1.1.4 netmask 255.255.255.0 mtu 32768 up"
IFCONFIG_2="14.1.1.7 broadcast 14.1.1.255 netmask 255.255.255.0 mtu 1492 up"
IFCONFIG_3=""

```

Figure 35. Adding the TCP/IP configuration for a GbE connection in */etc/rc.config*

Adding OSA Express GbE to the Linux image under VIF

There are no differences between loading the device driver modules and updating the configuration, but you must first add two network connections to the Linux image to attach the set of four I/O devices required by GbE:

```

# vif image net linux1 add E0C
# vif image net linux1 add E0E

```

After adding the device to the Linux image, you must update */etc/modules.conf* and */etc/rc.config*.

Then stop and restart the Linux image:

```

# vif image stop linux1
# vif image start linux1

```

And finally run *SuSEconfig* to finish.

Chapter 5. Implementing DB2 for Linux on S/390

DB2 for Linux V7 is now available on the S/390 platform. You install DB2 for Linux V7 in a S/390 LPAR the same way you install it on any Intel or UNIX workstation platform.

This chapter walks you through the process of installing DB2 for Linux on S/390, and describes how to customize client connections to the host database using the DB2 Connect connection gateway.

5.1 DB2 for Linux installation on S/390

You can install DB2 for Linux in two types of environment:

- Native Linux LPAR
- VIF Linux LPAR

The installation process of DB2 for Linux V7 is the same in both environments and is accomplished through the following steps:

- Address all hardware and software requirements.
- Define the naming conventions and parameter values.
- Transfer the DB2 .tar file to the Linux system on S/390.
- Configure Linux for S/390.
- Install DB2 and customize the following:
 - DB2 EE
 - DB2 Connect
 - Sample database
 - DB2 initialization at system startup
- Start DB2.

In the following sections, we explain these steps in more detail.

5.1.1 Hardware and software requirements

Hardware and software requirements for Linux for S/390 are documented in the IBM Redbook *Linux for S390*, SG24-4987.

Hardware and software requirements for DB2 for Linux V7 are documented in *Quick Beginnings Version 7*, GC09-2970.

The DB2 for Linux setup shell utility requires the korn shell. Install the Public Domain Korn Shell package (pdksh) included in SuSE Linux.

To install pdksh, select **Choose/Install packages** from the YaST main menu, then choose **Create/Change configuration**. This will take you to the Series selection panel. Then choose the series **ap**, which is described as “Programs that don’t need X”.

For more information about how to install this package, refer to the redbook *Linux for S390*, SG24-4987.

Note

A *shell* is an interface that allows a user to work with the operating system. There are different shells available with Linux.

We used the following in our test environment:

- SuSE Linux for S/390 RC6
- DB2 for Linux V7 code release SQL07010 with level identifier 02020105 and informational tokens DB2 v7.1.0.2, s000810, and U472079.

5.1.2 Defining DB2 users

For a first installation, we recommend that you use the DB2 default users as listed in Table 2. Those default users are initially generated by the SuSE Linux installation process for usage by DB2. Those users are also the defaults which DB2 installation would create if they did not already exist at DB2 installation time.

Table 2. Default user definitions for DB2

User	Home directory	Description
db2inst1	/usr/lib/db2/db2inst1	DB2 Instance user
dbas	/usr/lib/db2/db2as	DB2 Administration user
dbfenc1	/usr/lib/db2/db2fenc1	DB2 Fence user for UDFs and Stored Procedures

If you don't want to use the default users provided by SuSE Linux and DB2 for Linux, you can use Table 3 as a worksheet to define your own users. You should add your new users to the group db2iadm1.

Table 3. Worksheet for your own user definitions for DB2

User	Home directory	Description
		DB2 Instance user
		DB2 Administration user
		DB2 Fence user for UDFs and Stored Procedures

A user in Linux is comparable to a user ID in S/390. A user ID in Linux is a number and the user relates to the user ID. You create a new user with the useradd function. For detailed information on how to create users, refer to *Linux for S390*, SG24-4987.

5.1.3 Customizing SuSE Linux for DB2 support

To ensure a smooth installation of DB2 for Linux on S/390, edit and modify the /etc/services file, removing the following entries:

```
db2cdb2inst1    50000/tcp      # Connection port for DB2 instance db2inst1
db2idb2inst1    50001/tcp      # Interrupt port for DB2 instance db2inst1
```

Those entries are inserted by the DB2 setup. If the entries already exist, DB2 setup inserts additional lines with higher port numbers in sequence, resulting in non-default ports definitions for this installation. However, this can cause confusion with the port numbers when, for example, configuring client access to the database.

5.1.4 Transferring the DB2 archive file to the Linux system on S/390

For this step, you first have to transfer the DB2 for Linux archive file (<filename>.tar) from a workstation or ftp-server to the Linux system on S/390. Then you have to unpack this archive file and install DB2 for Linux.

5.1.4.1 Transfer the DB2 archive file

To transfer the DB2 archive file, use the FTP program. Do *not* use the root user for this operation; for security reasons, root is not authorized to perform FTP.

You ftp the DB2 archive file from your workstation to the /tmp directory on Linux for S/390 in bin format as follows:

- On your workstation, go to the directory where the DB2 archive file is located. For example:

```
/tmp/s00810b.familyCD.tar
```

- Connect to Linux for S/390 using FTP and enter your user name and password:

```
ftp 9.12.18.47
```

- Switch to binary mode:

```
bin
```

- Change directory on Linux for S/390 to go to /tmp:

```
cd /tmp
```

- Put the DB2 archive file into the /tmp directory on Linux for S/390:

```
put s00810b.familyCD.tar
```

- Exit the FTP program:

```
bye
```

5.1.4.2 Unpack the DB2 archive file

Login to Linux for S/390 as user root, go to the /tmp directory, and extract the DB2 archive file with the following command:

```
tar -xvf <tarfilename>.tar
```

Go to the /tmp directory and run the following:

```
tar -xvf s00810b.familyCD.tar
```

Unpacking the tar file creates one directory (db2) and three files (db2_deinstall, db2_install, db2setup) in the <tarfilename> directory.

5.1.5 Installing DB2 for Linux

The installation of DB2 for Linux on the S/390 platform has the same look and feel as an installation on any other UNIX platform.

During the installation process, you will select, install, and customize the following DB2 components:

- DB2 Administration Client

Select this option to install the Java Client database administrator.

DB2 Administration Client cannot run on Linux S/390 if Java runtime environment (JRE) is not available. Do not select this option without JRE implemented in your system.

- DB2 UDB Enterprise Edition

This option will install DB2 UDB for Linux Enterprise Edition.

- DB2 Connect Enterprise Edition

Select this option to install DB2 Connect to enable access to host DB2 databases (OS/390 or AS/400).

- DB2 Application Development Client

Select this option if you have to develop applications which access DB2 databases.

- Language option for DB2 Product Messages
- Language option for DB2 Product Library

The DB2 for Linux V7 installation proceeds as follows:

- Log in to Linux on S/390 as user root, go to the /tmp/s00810.familyCD directory, and start the installation with the following command:

```
./db2setup
```

The Install DB2 V7 prompt screen appears as shown in Figure 36. Select the different DB2 options you need as listed in this figure.

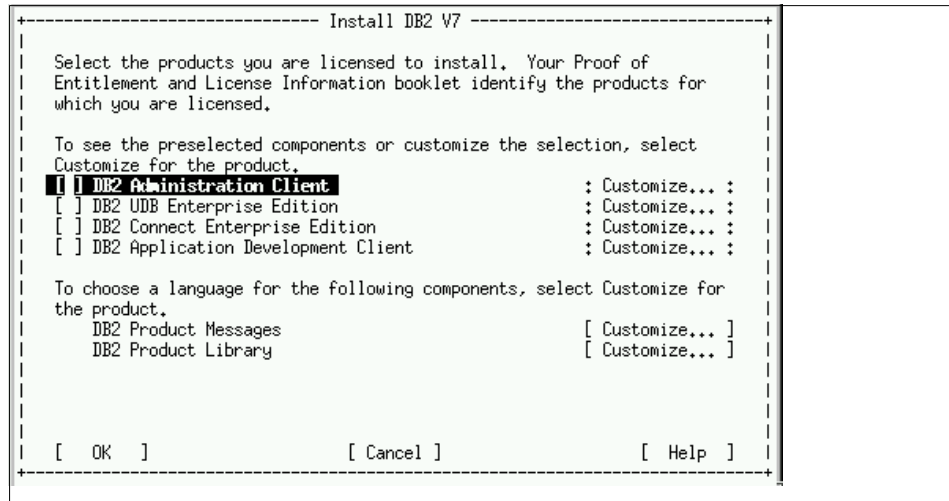


Figure 36. Install DB2 V7

Each time you select an option, either a new text window opens or an asterisk (*) shows in front of your selection. For example, assume you want to select **mychoice**; before you select it, it will appear as follows:

```
[ ] mychoice
```

After you select it, it will appear as follows:

```
[*] mychoice
```

However, if your selection appears like : : **mychoice**, it means that either the option is either not available at that time, or you need to install an additional product to be able to select this option. You can do the following:

- Select **DB2 Administration Client** to install the DB2 Administration Client. Then, select **customize** on the right and choose any option needed for your installation.
- Select **DB2 UDB Enterprise Edition** to install the DB2 for Linux Enterprise Edition. Then, select **customize** on the right and choose any option needed for your installation as shown in Figure 37.

Note: Also remember to select the right Code Page conversion support required in your environment.

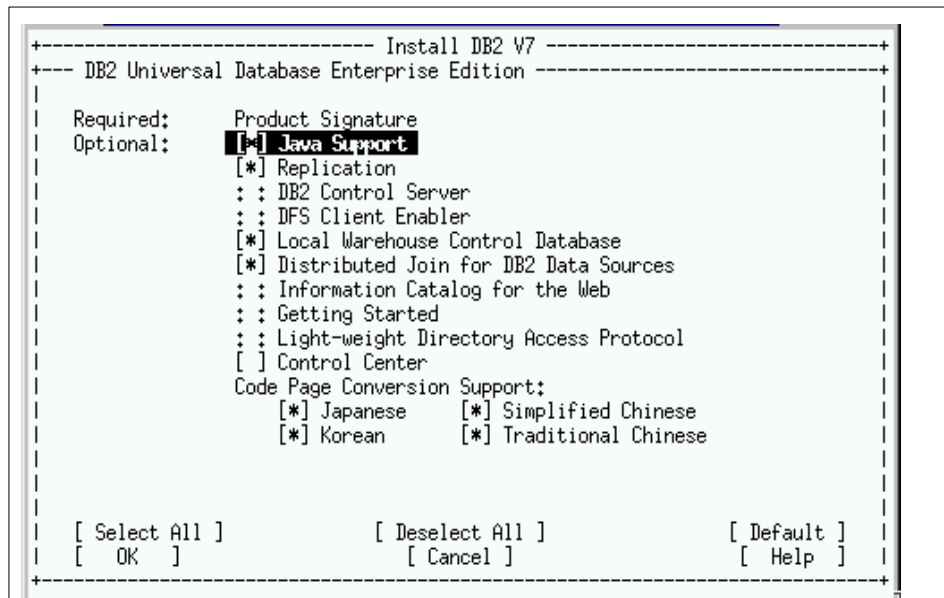


Figure 37. DB2 Universal Database Enterprise Edition

- Select **Application Development Client** to install the Application Development Client. Then, select **customize** on the right and choose any option needed for your installation.
- Select **Customize** on the right to install the language for the DB2 Product Messages, and chose the one you prefer (this screen is not shown).

- Select **Customize** on the right to install the language for the DB2 Product Library, and chose the one you prefer as shown in Figure 38. This selection comes in addition to en_US, which you must *always* choose.

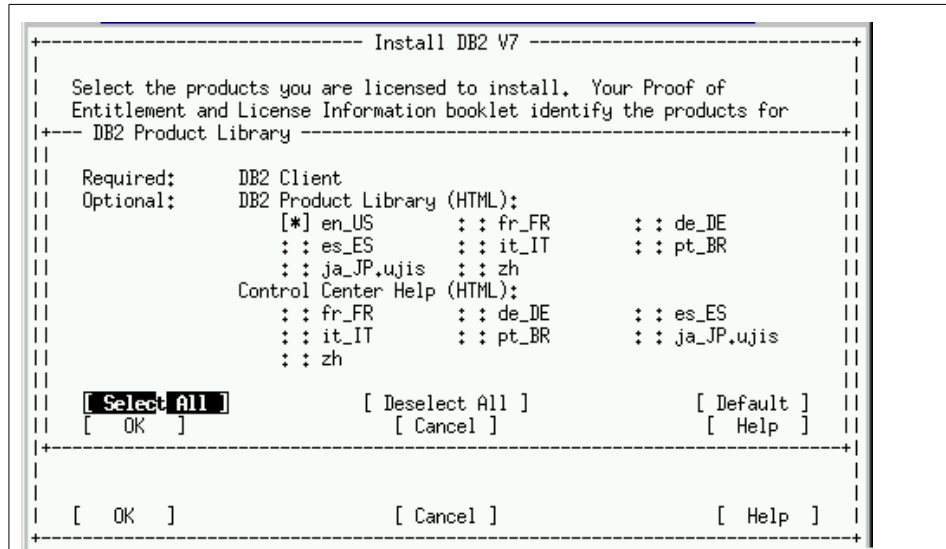


Figure 38. DB2 Product Library

After you confirm all your choices, you proceed with the following settings:

- Select **Create a DB2 Instance**, which prompts the screen shown in Figure 39 on page 87.

```

+----- Create DB2 Services -----+
|+--- DB2 Instance -----+|
||
|| Authentication:
||   Enter User ID, Group ID, Home Directory and Password that will be
||   used for the DB2 Instance.
||
||   User Name      [db2inst1]
||   User ID        [47]          [ ] Use default UID
||   Group Name     [db2iadm1]
||   Group ID       [47]          [ ] Use default GID
||   Home Directory [ <ib/db2/db2inst1]
||   Password       [
||   Verify Password [
||
|| Select Properties to view or change more options. [ Properties... ]
||
|| Select Default to restore all default settings. [ Default ]
||
|| [ OK ] [ Cancel ] [ Help ]
++-----+

```

Figure 39. DB2 instance

It is necessary to create a DB2 instance to setup DB2 Connect. You are asked to enter the User, Group, Home Directory and Password values which will be used for the DB2 Instance User. Refer to Table 3 on page 81 if you want to use your own users, or use the default users to complete the required settings. The password can be left empty to be changed later as described in 5.1.6, “Post-installation” on page 92. We recommend that you leave the default values as shown in Figure 40 on page 88.

- Select **Properties** to view and customize DB2 Instance Properties as shown in Figure 40. Select the type of authentication you want for the DB2 instance.

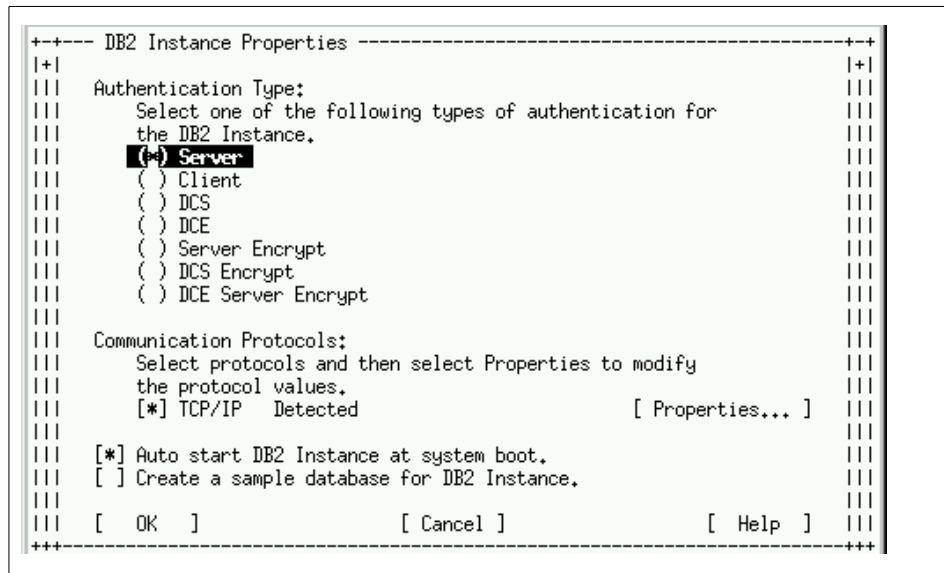


Figure 40. DB2 Instance Properties

- Customize the **Properties** of **TCP/IP** as shown in Figure 41.

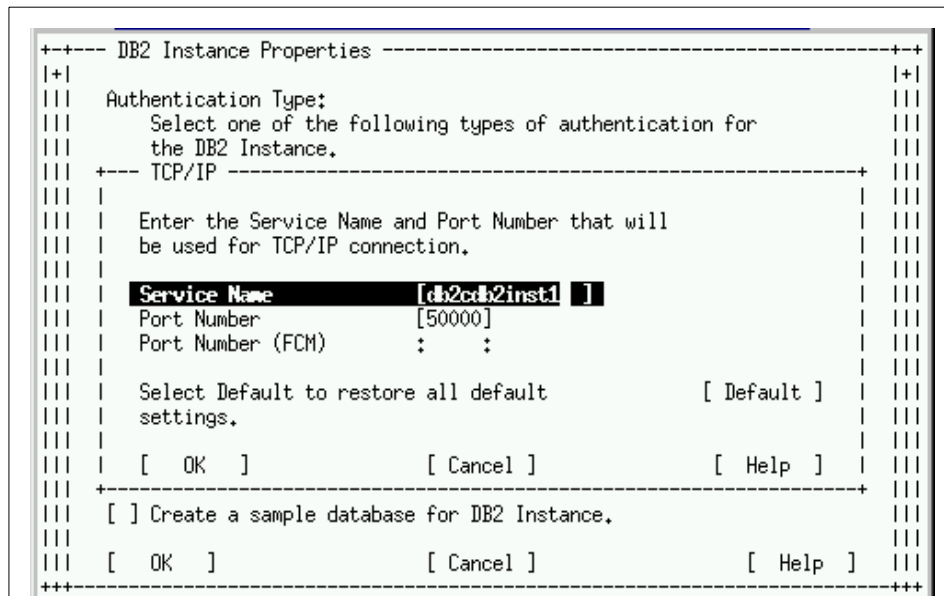


Figure 41. TCP/IP

Referring back to the DB2 Instance Properties, you still have two more choices to make, as shown in Figure 42:

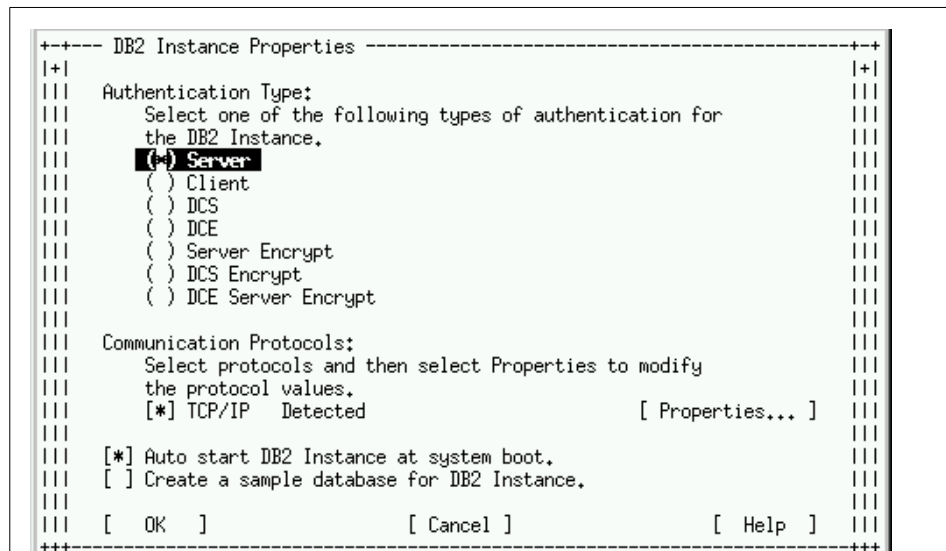


Figure 42. DB2 Instance Properties

If you want to start DB2 Instance at system boot time, select **Auto start DB2 Instance at system boot**.

If you want to set up a sample database for the DB2 instance, select **Create a sample database for DB2 Instance**.

- After some more customizations, you will be prompted to customize the Fenced User (user authorized for UDFs and stored procedures) as shown in Figure 43 on page 90. We recommend that you use the default values.

```

++--- Fenced User -----++
||
|| Fenced user defined functions (UDFs) and stored procedures will
|| execute under this user and group.
||
||
|| Authentication:
|| Enter User ID, Group ID, Home Directory and Password that will be
|| used for the Fenced User.
||
||
|| User Name      [db2fenc1]
|| User ID       [46]           [ ] Use default UID
|| Group Name    [db2fadm1]
|| Group ID      [46]           [ ] Use default GID
|| Home Directory [ib/db2/db2fenc1]
|| Password      [ ]
|| Verify Password [ ]
||
|| Select Default to restore all default settings. [ Default ]
||
|| [ OK ]           [ Cancel ]           [ Help ]
++-----

```

Figure 43. Fenced User

Enter the User, Group, Home Directory and Password required for the DB2 Fenced User. Refer to Figure 3 on page 81 if you want to use your own users, or use the default users to complete the required settings. The password can be left empty to be changed later, as described in 5.1.6, “Post-installation” on page 92.

After finishing the fenced user customization, you continue with the following installation steps:

- DB2 Warehouse Control Database

To set up a DB2 Warehouse Control Database, select **Set up DB2 Warehouse Control Database**.

- Administration Server

Customize the administration server as prompted in Figure 44 on page 91.

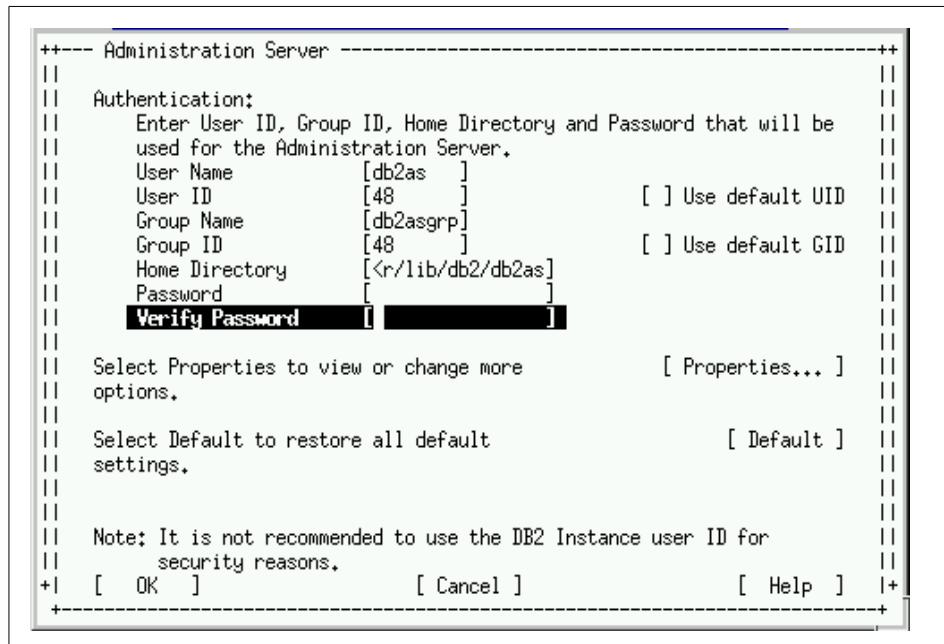


Figure 44. Administration server

Refer to Figure 3 on page 81 if you want to use your own users, or use the default users to complete the requested settings. Enter the User, Group, Home Directory and Password you want to use for the DB2 Fenced User. The password can be left empty, to be changed later, as described in 5.1.6, “Post-installation” on page 92. We recommend that you use the default values.

- Next, select **Properties** to administer the server properties. In the text window **TCP/IP** that now appears, you can enter the Service name and the Port number. We strongly recommend that you use the default values.
- At the end of the installation, you will get a summary report on all selected options. If you agree with your choices, click **continue** and the installation will execute with your customized options.
- When the installation process is finished, you will receive the message **Completed successfully**, along with an overview status report. You can also check this report in /tmp/db2setup.log.
- At the end of the DB2 installation process, the DB2 Installer automatically starts the DB2 server process and the administration server process.

5.1.6 Post-installation

After you have finished the installation process, you have a few more tasks to do to make your DB2 for Linux more user-friendly, as follows:

- Modify the user passwords

Three DB2 users were created during SuSE Linux 7.0 installation with no usable password. You need to modify the passwords created in the previous processes if you want to be able to login with those DB2 users.

Login as user root and issue the following commands:

```
passwd db2inst1
```

You are prompted to enter the new password. Enter the password you want to use.

Also change the password of:

```
passwd db2as
```

Note

You can check the definition of all user IDs in your Linux system in the `/etc/passwd` file, as follows:

```
cat /etc/passwd
```

You can also check in `/etc/shadow` to see if a password has been defined for the DB2 users.

- Create DB2 User Profiles to make the DB2 UDB environment accessible to users.

To create a user .profile file for the users db2inst1 and db2as in their home directories, as specified in Table 2 on page 80 or Table 3 on page 81, do the following:

- Login as user db2inst1 on Linux for S/390.

Create a new user profile (the filename is .profile) for db2inst1:

```
if [ -f sqllib/db2profile ]; then
. sqllib/db2profile
fi

#For DB2 control center (db2cc)
export JAVA_HOME=/usr/lib/jre1.1.8/
db2jstrt
```

- Login as user db2as on Linux for S/390.

Create a new Profile for the db2as user:

Create a new user profile (the filename is .profile) for db2as:

```
if [ -f ~/db2inst1/sqlllib/db2profile ]; then
. ~/db2inst1/sqlllib/db2profile
fi
```

5.1.7 Checking the DB2 for Linux installation

Normally, all DB2 services start at the start time of Linux. If for any reason this does not happen, you should start the DB2 services manually. To start and stop the DB2 database, do the following:

- Login to Linux for S/390 using the user db2inst1.
- To start DB2 UDB, enter:

```
db2start
```

- To stop DB2 UDB, enter:

```
db2stop
```

To start DB2 UDB administration server, do the following:

- Login to the system using db2as.
- To start the DB2 UDB administration server, enter:

```
db2admin start
```

- To stop the DB2 UDB administration server, enter:

```
db2admin stop
```

If you did not install the DB2 Control Center, you can check your installation using the DB2 command line processor to query the sample database which is normally set up during the installation process. To validate your installation, log in to Linux as user db2inst1 and perform some queries against the DB2 sample database. For example:

```
db2 connect to sample
db2 "select * from staff where dept = 20"
db2 list tables
db2 describe table sysibm.systables
db2 connect reset
```

If the sample database has not been installed during the initial installation process, you can set it up later. Be sure to log in with a user such as

db2inst1(SYSADM) which has the authorizations of a system administrator required to execute this operation.

To create the sample database, enter:

```
db2samp1
```

The sample database will automatically be created and catalogued with the alias “sample”.

5.2 DB2 Connect for Linux setup on S/390

If you want to connect to the DB2 for OS/390 host database, you need a connection gateway such as DB2 Connect.

DB2 Connect setup includes the following steps:

- Check DB2 for OS/390 settings to support remote connections.
- Prepare DB2 Connect on Linux for S/390 to access DB2 for OS/390.
- Catalog the TCP/IP Node.
- Catalog the DCS.
- Catalog the database.
- Test the connection.
- Bind the utilities.

5.2.1 Checking DB2 for OS/390 settings

Make sure DB2 for OS/390 is customized to support remote connections. You should check the following with the host database administrator (DBA):

- Is TCP/IP set for the remote connections?
- Are there enough remote connections allowed?
- How is user security handled? (You should not allow everyone to connect to your host system.)
- Is DDF started? (You need to have DDF up and running to be able to connect to the host database.)

5.2.2 DB2 Connect configuration parameters

We recommend that you gather all the configuration parameter values you will need for this configuration *before* you start configuring DB2 Connect. You can use Table 4 on page 95 to identify the required parameter values.

For more details, refer to the IBM DB2 html help *DB2 Connect Quick Beginnings for Linux* at the following Web site in your local directory of DB2:

`doc/html/db2c4/db2c433.htm#HRTCP_TO_HOST_S2B`

Table 4. Configuration parameters for DB2 Connect setup

Parameter	Description	Sample value	Your value
Hostname (hostname) or IP address (ip_address)	Use the hostname or ip_address of the remote host.	9.100.206.103	
Connection Service Name (svcename) or Port number/Protocol (port_number/tcp)	The port number for the DB2 Connect workstation must be the same as the port number that the svcename parameter maps to in the services file at the host database server. (The svcename parameter is located in the database manager configuration file on the host.) This value must <i>not</i> be in use by any other applications, and must be unique within the services file.	ddm-rdb or 446/tcp	
Target database name (target_dbname)	The location name on DB2 for OS/390 system.	N7DBA0	
Local database name (local_dcsname)	An arbitrary local nickname for use by DB2 Connect that represents the remote host.	DB2OS390	
Node name (node_name)	A local alias, or nickname, that describes the node to which you are trying to connect.	NTSIM	
Database alias (database_alias)	An arbitrary local nickname for the remote database. If you do not provide one, it will be the default database name (in our case: local_dcsname)	DB2OS390	

5.2.3 Cataloging the TCP/IP node

To describe the remote node (in this case, the DB2 OS/390 LPAR) you must add an entry in the Linux LPAR's node directory. This entry specifies the chosen alias (node_name), the ip_address (or hostname) and svcename (or port_number) that the client will use to access the remote host.

- Log on to the Linux system as a user with system administration authority (SYSADM) or system controller authority (SYSCTRL). In the Linux for S/390 environment, this is the user db2inst1.

- Catalog the node by entering the following commands:

```
db2 catalog tcpip node node_name remote [hostname|ip_address] server [
svcname|port_number]
db2 terminate
```

For example, to catalog the remote host with the IP address 9.100.206.103 on the node called NTSIM, using the service name ddm-rdb, enter the following:

```
db2 catalog tcpip node NTSIM remote 9.100.206.103 server ddm-rdb
db2 terminate
```

If you need to change the values set with the `catalog node` command, perform the following:

- Run the `uncatalog node` command in the command line processor:

```
db2 uncatalog node node_name
```

- Recatalog the node with the values that you want to use.

- Check your new entry with the following command:

```
db2 list node directory
```

5.2.4 Cataloging the host database

To catalog the remote host database as a Database Connection Service (DCS), perform the following:

- Log on to the Linux system as a user with system administration authority (SYSADM) or system controller authority (SYSCTRL). In the Linux for S/390 environment, this is the user db2inst1.
- Catalog the remote database in DCS by entering the following commands:

```
db2 catalog dcs db local_dcsname as target_dbname
db2 terminate
```

For example, to catalog in DCS the local database name (or alias) DB2OS390, which DB2 Connect will use to access the remote host database called N7DBA0, enter the following:

```
db2 catalog dcs db DB2OS390 as N7DBA0
db2 terminate
```

If you need to change the values set with the `catalog dcs db` command, perform the following:

- Run the `uncatalog dcs db` command in the command line processor as follows:

```
db2 uncatalog dcs db locale_dcsname
```

- Recatalog the dcs db with the values that you want to use.

- Check your new entry with the following command:

```
db2 list dcs directory
```

5.2.5 Cataloging the database

Before a client application can access a remote database, the database must be cataloged on the host system node and on any DB2 Connect workstation nodes that will connect to it. When you create a database, it is automatically cataloged on the host with the database alias (`database_alias`) the same as the database name (in our case, `local_dcsname`).

The information in the database directory, along with the information in the node directory, is used on the DB2 Connect workstation to establish a connection to the remote database.

- Log on to system as a user with System Administrative (SYSADM) or System Controller (SYSCTRL) authority. In the installed Linux on S/390 Environment, this is the user `db2inst1`.
- Catalog the database by entering the following commands:

```
db2 catalog database local_dcsname as alias_name at node node_name
authentication dcs
db2 terminate
```

For example, to catalog the DCS known database DB2OS390 so that it has the local database alias DB2OS390, on the node NTSIM, enter the following:

```
db2 catalog database DB2S390 as N7DBA0 at node NTSIM authentication dcs
db2 terminate
```

You can look at your new entry with the following command:

```
db2 list db directory
```

Note

If you need to change values that were set with the `catalog database` command, perform the following:

- Run the `uncatalog database` command in the command line processor as follows:

```
db2 uncatalog database locale_dcsname
```

- Recatalog the database with the values that you want to use.

5.2.6 Binding utilities and applications to the database

Now you must bind the utilities and applications to the host using the command line from your Linux workstation or from Linux on S/390, as follows:

- Log on to system as a user with System Administrative (SYSADM) or System Controller (SYSCTRL) authority. In the installed Linux on S/390 Environment, this is the user `db2inst1`.
- To bind the utilities and applications to the host, enter the following commands:

```
db2 connect to dbalias user userid using password
db2 "bind path@ddcsmv.s.lst blocking all sqlerror continue messages
    mvs.msg grant public"
db2 connect reset
db2 terminate
```

For example, to enable the `db2` command line to get access to the host system:

```
db2 connect to DB2OS390 user myuserid using mypassword
db2 "bind /usr/IBMDB2/V7.1/bnd/@ddcsmv.s.lst blocking all sqlerror
    continue messages mvs.msg grant public"
db2 connect reset
db2 terminate
```

Note

- In the file `mvs.msg`, you can find all the messages the `bind` command prompted out.
- The user ID and password specified must have the authority to bind applications against the target database.

5.2.7 Testing access to the host

If you have finished configuring the DB2 Connect workstation for access to the host, perform the following to test the connection:

- Log on to system as a user with System Administrative (SYSADM) or System Controller (SYSCTRL) authority. In the installed Linux on S/390 Environment, this is the user db2inst1.
- Enter the following command in the command line processor to connect to the remote host database:

```
db2 connect to dbalias user userid using password
```

If the connection is successful, you can test the connection by entering the following query:

```
db2 "select name from sysibm.systables"
```

When you are finished using the database connection, enter the `connect reset` command to end the database connection, and enter the `exit` command to end the user session.

5.3 DB2 Client connectivity

This section describes the DB2 client installations to access DB2 UDB for OS/390 and DB2 UDB for Linux on S/390 via DB2 Connect. We describe the following DB2 clients:

- DB2 Run-Time Client
- DB2 Administration Client
- QMF for Windows

5.3.1 DB2 Run-Time client

DB2 Run-Time Client runs on a large variety of platforms, including Linux (Intel, S/390), Windows 9x, Windows NT and Windows 2000.

Before starting the installation, you need to ensure that your system meets all the memory, hardware and software requirements. Refer to *Quick Beginnings Version 7*, GC09-2970.

5.3.1.1 Installing DB2 Run-Time client on a Linux workstation

We recommend you use the db2setup utility to install DB2 products on Linux. You can use a simple interface that includes online help. The DB2 setup shell utility requires a korn shell. Install the Public Domain Korn Shell package already included in SuSE Linux (package pkdsh).

To install the DB2 Run-Time Client:

- Log on as a user with root authority.
- Insert and mount the appropriate CD-ROM.
- Change to the directory where the CD-ROM is mounted by entering the `cd /cdrom` command ,where /cdrom is the CD-ROM mount point.
- Change to the /cdrom/db2/linux directory.
- Enter the `./db2setup` command. The install DB2 V7 window opens.
- Select **DB2 Run-Time client**. Press Tab to move between available options and fields. Press Enter to select or deselect an option. Select **OK** to continue the installation process, or select **Cancel** to go back to a previous window.

Next, you configure the client to access a remote DB2 server using the Command Line Processor, as shown in 5.3.1.3, “Configuring DB2 Run-Time client to access DB2 Connect” on page 101.

5.3.1.2 Installing DB2 Run-Time client on a Windows workstation

To install DB2 Run-Time Client on a workstation, perform the following steps:

- Log on to the system with the user account that you want to use to perform the installation.
- Shut down any other programs so that the setup program can update files as required.
- Insert the appropriate CD-ROM into the drive. The auto-run feature automatically starts the setup program. To manually invoke the setup program, click Start and select the **Run** option.
- The DB2 Launchpad opens.
- From the window, you can proceed to the installation by clicking **Install**.

The next step will consist of configuring the client to access a remote DB2 server using the Client Configuration Assistant, as shown in 5.3.1.3, “Configuring DB2 Run-Time client to access DB2 Connect” on page 101.

Note

DB2 clients can connect to DB2 servers two releases later or one release earlier than the client’s release level, as well as to servers at the same release level.

5.3.1.3 Configuring DB2 Run-Time client to access DB2 Connect

You need to configure DB2 Run-Time Client to get access to DB2 Connect. Through DB2 Connect, the client is able to get access to DB2 for OS/390.

Preparing for the configuration

To configure DB2 Run-Time Client, you need to gather some configuration values. You can use Table 5 on this purpose.

Table 5. Configuration values for DB2 for Linux client

Parameter	Description	Sample value	Your value
Hostname (hostname) or IP address (ip_address)	Use the hostname or ip_address of the remote host.	9.100.195.3	
Connection Service Name (svcename) or Port number/Protocol (port_number/tcp)	The port number for the DB2 Connect workstation must be the same as the port number that the svcename parameter maps to in the services file at the host database server. (The svcename parameter is located in the database manager configuration file on the host.) This value must not be in use by any other applications, and must be unique within the services file.	db2cdb2inst1 or 50000/tcp	
Database name (database_name)	The database alias (database_alias) of the remote DB2 Connect database	DB2OS390	
Node name (node_name)	A local alias, or nickname, that describes the node to which you are trying to connect.	DB2CONN	
Database alias (database_alias)	An arbitrary local nickname for the remote database. If you do not provide one, the default database name (in our case: local_dcsname)	OS390	

Catalog the TCP/IP node

To describe the remote node (in this case, DB2 Connect for Linux on S/390,) you must add an entry in the client's node directory. This entry specifies the chosen alias (node_name), the ip_address (or hostname) and svcename (or port_number) that the client will use to access the remote host.

- Log on to system as a user with System Administrative (SYSADM) or System Controller (SYSCTRL) authority. In the installed Linux Client Environment, this is the user db2inst1.
- Catalog the node by entering the following commands:

```
db2 catalog tcpip node node_name remote [hostname|ip_address]
      server [ svcename|port_number]
db2 terminate
```

For example, to catalog the remote host with the IP address 9.100.195.3 on the node called DB2CONN, using the service name db2cdb2inst1, enter the following:

```
db2 catalog tcpip node DB2CONN remote 9.100.195.3 server db2cdb2inst1
db2 terminate
```

You can look at your new entry with the following command:

```
db2 list node directory
```

Note

If you need to change values that were set with the catalog node command, perform the following:

- Run the `uncatalog node` command in the command line processor:

```
db2 uncatalog node node_name
```

- Recatalog the node with the values that you want to use.

Catalog the database

Before a client application can access a remote database, the database must be cataloged on the server node and on any client nodes that will connect to it. When you create a database, it is automatically cataloged on the server with the database alias (`database_alias`) the same as the database name (in this case: `local_dcsname` on DB2 Connect for Linux on S/390).

The information in the database directory, along with the information in the node directory, is used on the DB2 Client to establish a connection to the remote database.

- Log on to system as a user with System Administrative (SYSADM) or System Controller (SYSCTRL) authority. In the installed Linux Environment, this is the user db2inst1.
- Catalog the database by entering the following commands:

```
db2 catalog database database_name as datatabase_alias at node node_name
db2 terminate
```

For example, to catalog the remote database called DB2OS390 so that it has the alias OS390, on the node DB2CONN, enter the following:

```
db2 catalog database DB2OS390 as OS390 at node DB2CONN
db2 terminate
```

You can look at your new entry with the following command:

```
db2 list db directory
```

Note

If you need to change values that were set with the `catalog database` command, perform the following:

- Run the `uncatalog database` command in the command line processor as follows:

```
db2 uncatalog database locale_dcsname
```

- Recatalog the database with the values that you want to use.

Test the client access to the host

If you have finished configuring the DB2 Connect workstation for accessing the host, perform the following to test the connection:

- Log on to system with a user ID which has DB2 System Administrative (SYSADM) or System Controller (SYSCTRL) authority. In the installed Linux Environment, this is the user `db2inst1`.
- Enter the following command in the command line processor to connect to the remote host database:

```
db2 connect to database_alias user userid using password
```

If the connection is successful, you can test the database access by entering the following command:

```
db2 "select name from sysibm.systables"
```

When you are finished using the database, enter the `connect reset` command to end the database connection, and enter the `exit` command to end the user session.

5.3.2 DB2 Administration Client

The DB2 GUI tools are a set of administration and configuration tools which include the Command Center, Control Center, and the Data Warehouse Control Center (at the time of writing, Data Warehouse Control Center is only available for Windows). Additional GUI tools are available on Windows and

OS/2 operating systems. Refer to *DB2 Administration Guide* for more information about DB2 GUI tools.

To run DB2 Administration client on Linux, the IBM Java Runtime Environment (JRE) 1.1.8 is required. JRE is not required for Windows.

TCP/IP is the communication protocol used to connect DB2 client on Linux or Windows to DB2 for Linux on S/390.

A DB2 Administration Client provides the ability for workstations from different platforms to access and administer DB2 databases. These workstations are known as DB2 Administration Clients. The DB2 Administration Client has all the features of DB2 Run-Time Client and also includes all the DB2 Administration tools, documentations, and support for thin clients.

DB2 Administration Client runs on Linux, Windows 9x, Windows NT and Windows 2000 platforms.

5.3.3 QMF for Windows accessing DB2 for Linux on S/390

To query DB2 UDB for Linux tables using QMF for Windows, you have to create the explain tables.

- Log on to system as a user with DB2 System Administrative (SYSADM) or System Controller (SYSCTRL) authority. In the installed Linux on S/390 Environment, this is the user db2inst1.
- Enter the following command in the command line processor to connect to the remote database:

```
db2 connect to dbalias user userid using password
```

- To create the tables in the database, enter the following command:

```
db2 -tf ~/sqllib/misc/EXPLAIN.DDL
```

- To verify that all explain tables are created, enter the following command:

```
db2 list tables
```

Nine tables are created:

Table/View	Schema	Type	Creation time
ADVISE_INDEX	DB2INST1	T	2000-11-18-11.15.52.471636
ADVISE_WORKLOAD	DB2INST1	T	2000-11-18-11.15.53.838437
EXPLAIN_ARGUMENT	DB2INST1	T	2000-11-18-11.15.50.703133
EXPLAIN_INSTANCE	DB2INST1	T	2000-11-18-11.15.46.737557
EXPLAIN_OBJECT	DB2INST1	T	2000-11-18-11.15.50.960304
EXPLAIN_OPERATOR	DB2INST1	T	2000-11-18-11.15.51.767339
EXPLAIN_PREDICATE	DB2INST1	T	2000-11-18-11.15.51.911197
EXPLAIN_STATEMENT	DB2INST1	T	2000-11-18-11.15.49.837970
EXPLAIN_STREAM	DB2INST1	T	2000-11-18-11.15.52.163060

5.4 DB2 for Linux on S/390 performance

This section highlights some performance elements to consider when deploying DB2 for Linux applications.

Use the following list to assess DB2 for Linux performance:

- Are the bufferpool sizes correct?
- Check the bufferpool hit ratios: index, data, and aggregate.
- Are there any issues with sorts?
- Are there indications of excessive locking or contention?
- Ensure the optimization class is appropriate for the workload.
- Ensure the appropriate statistics have been gathered, and that packages have been rebound to take advantage of the latest statistics.
- Check whether any tables need reorganization.
- Verify that prefetch and asynchronous I/O have been correctly configured.
- Verify that query intra-parallelism has been configured correctly.
- Check system logs and DB2 diagnostic logs for indications of possible causes of poor performance.
- Use the Index Advisor to determine if additional indexes will improve performance.
- Check for system bottlenecks: I/O, CPU, memory.

Running the workload you want to check, you can use different tools to gather useful DB2 information, such as:

- The db2batch tool
- The DB2 snapshot monitor

In any case, we recommend that you turn on the bufferpool, lock, sort, table and statement monitor switches before you start to investigate performance issues.

5.4.1 Linux system performance

To check Linux system performance, examine the system error logs for reports of errors that may affect performance, such as disk errors, communication errors, or memory errors. Browse the file located in `/var/log/messages`.

Determine if there are any system bottleneck in I/O, CPU, or memory, as follows:

- I/O
 - Use `sar` or `iostat`. These tools are contained in the `sysstat` package that may be obtained from:
`ftp://sunsite.unc.edu/pub/Linux/system/status`
Or you can install them from SuSE Linux for S/390 CDs.
 - Check the utilization of any disk (it should not be higher than 50%).
 - Check the I/O load distribution across the disks holding DB2 table space containers.
 - Check I/O wait time.
- CPU
 - Use `vmstat`, `iostat`, `mpstat` or `sar` utilities.
 - Check the CPU utilization. In a SMP environment, check that the % Processor utilization for each processor is roughly equal to the others.
- Memory
 - Use `vmstat`, `free` or `sar` utilities.
 - Check for unused memory that could be allocated to the bufferpool or used for other requirements such as sorting, etc.

5.4.2 DB2 subsystem performance

Check the following:

- `db2diag.log`: Are there any indications in the `db2diag.log` file of possible causes for poor performance (for example, index page splitting or index rebuilds)?

- Bufferpool sizes: Check that the bufferpool sizes are adequate and as expected with the following command: `select * from syscat.bufferpools.`
Verify that the NPAGES values in the output of this query of BUFFPAGE in the database configuration are correct for any bufferpools with NPAGES=-1 in SYSCAT.BUFFERPOOLS.
- Query optimization class: Is the query optimization class appropriate for the workload (in general, lower values for complex queries).
- Statistics: Is the query optimization class appropriate for the workload (in general, lower values for OLTP, higher values for complex queries)?
- Table reorganization: Use reorgchk to identify any tables that need reorganization (for example, to compress the table or take advantage of index clustering).
- Prefetching and Asynchronous Page cleaners: For each table space, check that the prefetch size is (from LIST TABLESPACES SHOW DETAIL): extent size x number of containers, for non RAID-5 table spaces; extent size x number of disks in RAID-5 array, for RAID-5 table spaces.
Also check that the number of I/O servers is adequate for the total number of data containers, and that an adequate number of asynchronous page cleaners have been configured.
- Log setup: With circular logging, is the number of primary logs sufficient to prevent regular use of secondary logs? Are the log files separated from data table spaces on physical disk?
- Query Degree: Are the DFT_DEGREE (database configuration), MAX_QUERYDEGREE and INTRA_PARALLEL (database manager configuration) parameters set to sensible values? On SMP systems, DFT_DEGREE and MAX_QUERYDEGREE should be set to 'ANY' and INTRA_PARALLEL to 'YES'. On uni-processor machines, set INTRA_PARALLEL to 'NO'.
- Indexes: Use the Index Advisor to determine if additional indexes will improve performance.
- Bufferpool hit ratio: The DB2 snapshot reports and the output from db2batch give separate logical and physical read information for data and indexes. This information can be combined to give the overall bufferpool hit ratio.

In general, the bufferpool hit ratio should be as near to 100% as possible. How high it can be in reality is a function of the available memory for the bufferpool and the amount of data (both table and index) that needs to be accessed to execute the workload. If the bufferpool hit ratio is significantly below 100% and disk I/O is a bottleneck (shown by high % I/O wait), then

you should consider increasing the size of the bufferpool, including adding more real memory if required.

- **Sorting:** Check the sort overflows, post threshold sorts and piped sorts rejected (= piped sorts requested - piped sorts accepted) snapshot data. If any of these are not zero, then tune the relevant DB2 parameters. Determine if creating additional indexes would remove the need to explicitly sort data.

- **Locking issues:** Check the snapshot output for the following—a high number of lock waits, or a long time spent waiting on locks; lock escalations; deadlocks and lock timeouts (these will cause a failure of a statement, but not necessarily the one in the workload under examination if this is not the only work being performed on the system).

Waiting for a deadlock to be detected, or for a lock to time out, will delay query execution.

- **Background Processing:** Processing of (and I/O related to) traces and other monitoring can affect performance. Check the following: CLI and ODBC tracing is turned off; DB2 auditing is turned off—or if it must be on, that the minimum acceptable level of audit data is being recorded; DB2 event monitoring is turned off, or if it must be on—that the minimum acceptable level of event data is being recorded.

5.5 DB2 Connect performance

DB2 Connect is a member of the DB2 family of products and provides application requester functions for connectivity between DB2 clients and Distributed Relational Architecture (DRDA) hosts, such as DB2 for OS/390.

DB2 Connect is available in two products:

- **DB2 Connect Personal Edition (PE)**, which provides access from a single workstation to a DB2 database residing on a DRDA host
- **DB2 Connect Enterprise Edition (EE)**, which functions as a host data access server, providing managed connectivity from client desktop and Web applications to databases stored on DRDA hosts or DB2 UDB systems.

In addition to DRDA host access, DB2 Connect provides the following functions:

- A run-time environment for database applications that use the following APIs: Microsoft ODBC, DB2 embedded SQL (static and dynamic), DB2 Call Level Interface, JDBC, and Embedded SQL for Java (SQLJ)

- Support for stored procedures at the host
- Support for communicating with the host via either APPC or TCP/IP

In addition, DB2 Connect EE provides the following functions:

- Support for clients on Windows 3.1, OS/2, Windows NT, (%), (*), UNIX, or Apple Macintosh System 7 operating systems
- Support for client connections via popular communication protocols, including NetBIOS, TCP/IP, IPX/SPX, or APPC
- Support for DCE Directory Services for simplified management of network addressing information

DB2 Connect can be used alone or with DB2 UDB on the same machine.

5.5.0.1 DB2 Connect EE performance checklist

The factors that contribute to server requirements for DB2 Connect EE are:

- The number of users
- The transaction throughput
- Other tasks such as file server, print server, etc.

To ensure a successful installation, check the following points:

- Ensure sufficient hard drive space is available for DB2 Connect.
- Ensure sufficient memory is installed to support DB2 Connect (depending on the number of concurrent users).
- Consider using additional network adapters you plan to support many concurrent users.
- Check OS/390 system is properly configured for TCP/IP connection. DB2 for OS/390 Version 5 Release 1, OS/390 Version 1 Release 3 are the minimum system levels needed to support connectivity through TCP/IP.
- Distributed Data Facility (DDF) must be configured and started. A server port number and a resynchronization port number must be defined to support TCP/IP connection requests from remote DRDA clients.

To configure DB2 Connect on the client, you will need the following information:

- TCP/IP hostname or IP address
- The location name of the DB2 subsystem
- The port number associated with the DB2 instance where the target database resides

5.6 DB2 client performance

To ensure a successful installation, check the following points:

- Ensure sufficient hard drive space is available on the client for DB2 Run-Time Client.
- Ensure client machines have sufficient disk space and memory available.
- Ensure the front-end application has been identified for interfacing with DB2 Connect, and that a plan is in place for testing the connection.

Chapter 6. Building the data mart with DB2 for Linux

This chapter describes how to create the DB2 for Linux objects (DB2 instance, database, bufferpools, containers, table spaces, tables and indexes) needed to build a DB2 data mart in a Linux on S/390 environment. We show examples of how to create the linux8 database, buffer pools, one table space and one table. Appendix B, “DB2 for Linux scripts to create the data mart” on page 183 lists all the objects used for our testing.

Building of the DB2 data mart includes the creation of several database objects, which consist of instances, databases, tables, views, indexes, and schemas, and storage objects which consist of table spaces, containers and buffer pools. Figure 45 shows the relationship between DB2 objects.

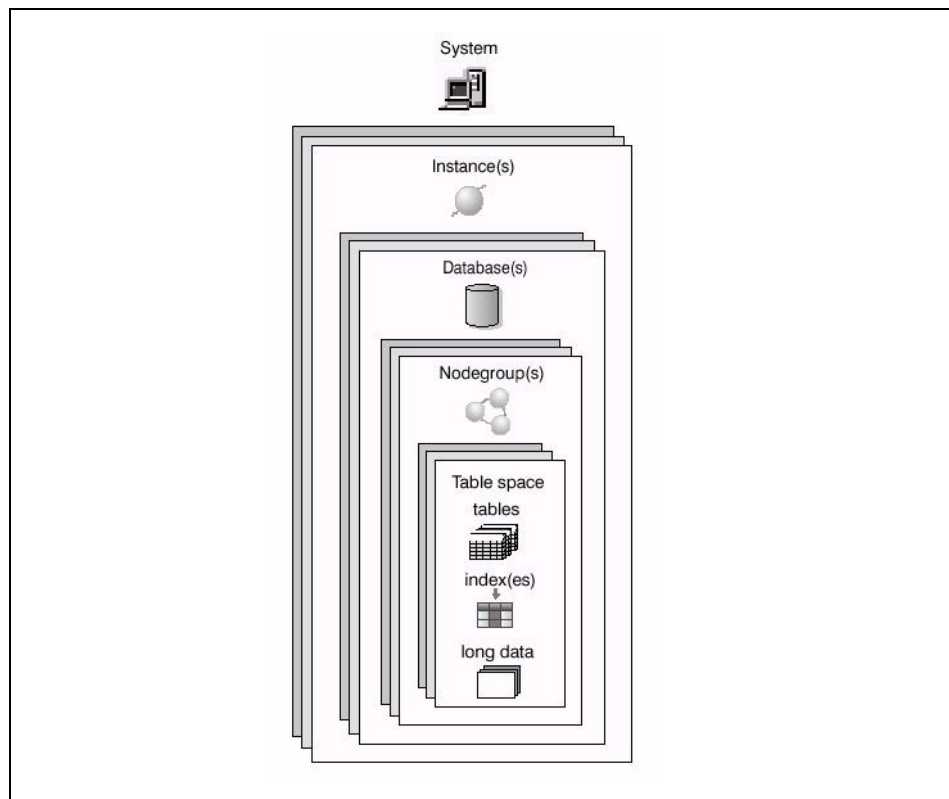


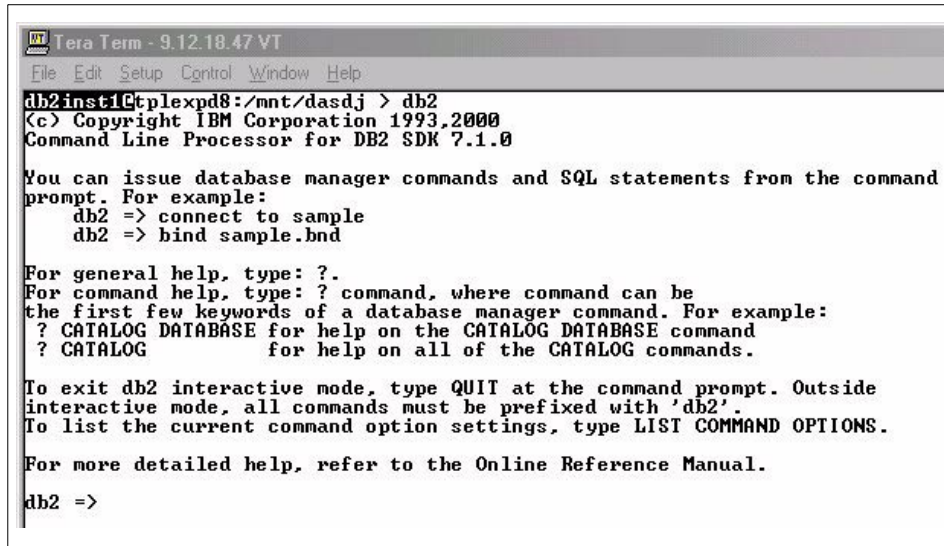
Figure 45. Relationship among DB2 database objects

DB2 commands and statements can be run from the UNIX shell command line by prefixing the command or statement with `db2`. To connect to the linux8

database on the local system, TPLEXPd8, use the command `db2 connect to alias` as shown:

```
db2inst1@tplexpd8:~ > db2 connect to linux8
```

Another way is to invoke the DB2 Command Line Processor (CLP) interactive mode. To invoke the DB2 Command Line Processor, enter the command `db2` as shown in Figure 46.



```
Tera Term - 9.12.18.47 VT
File Edit Setup Control Window Help
db2inst1@tplexpd8:/mnt/dasdj > db2
<C> Copyright IBM Corporation 1993.2000
Command Line Processor for DB2 SDK 7.1.0

You can issue database manager commands and SQL statements from the command
prompt. For example:
  db2 => connect to sample
  db2 => bind sample.bnd

For general help, type: ?.
For command help, type: ? command, where command can be
the first few keywords of a database manager command. For example:
? CATALOG DATABASE for help on the CATALOG DATABASE command
? CATALOG           for help on all of the CATALOG commands.

To exit db2 interactive mode, type QUIT at the command prompt. Outside
interactive mode, all commands must be prefixed with 'db2'.
To list the current command option settings, type LIST COMMAND OPTIONS.

For more detailed help, refer to the Online Reference Manual.

db2 =>
```

Figure 46. DB2 Command Line Processor interactive mode

DB2 commands can then be used without the prefix of `db2`. This mode is called the *interactive mode* of the Command Line Processor. However, an advantage of typing `db2` in front of every command is that it allows you to use standard UNIX shell features like pipe and pagers (`more`, `less`), `tail`, `grep`, and recall of commands from history; this facility is not available in interactive mode. To exit from the DB2 Command Line Processor, type: `quit`.

Following are some examples of how to list connected applications and table spaces:

```
db2inst1@tplexpd8:~ > db2 list applications show detail | more
```

```
db2inst1@tplexpd8:~ > db2 list tablespaces | tail -20
```

```
db2inst1@tplexpd8:~ > db2 list tablespaces | grep Name
```

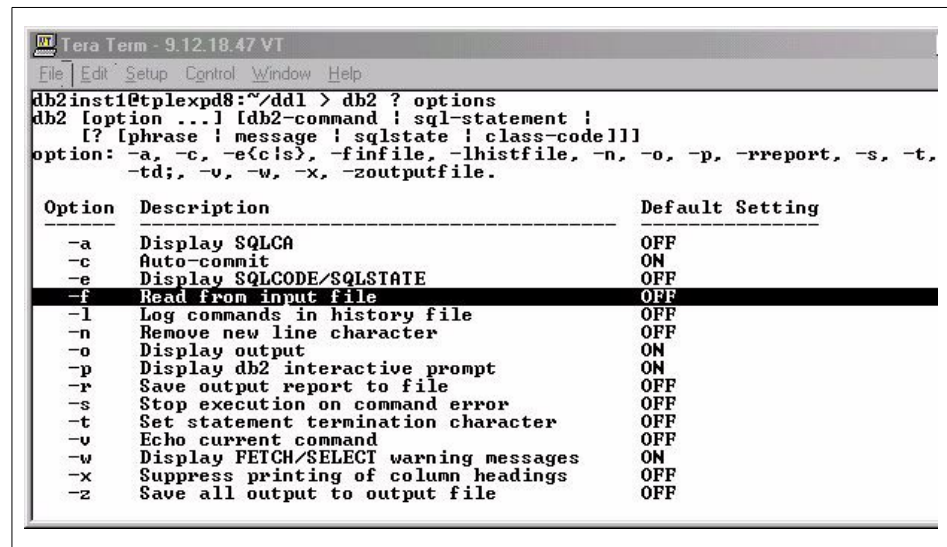
Script files can also be used to contain DB2 commands and statements. To invoke DB2 commands and statements within a script file, use the `-f` option with the `db2` command:

```
db2inst1@tplexpd8:~/ddl > db2 -stvf db2con.sql
```

To list the contents of the script file `db2con.sql`, use the `cat` command as shown:

```
db2inst1@tplexpd8:~/ddl > cat db2con.sql
connect to linux8;
```

Use the `db2 ? options` command to see the meaning of the options used on the command line. Figure 47 shows the output from the `db2 ? options` command.



```
db2inst1@tplexpd8:~/ddl > db2 ? options
db2 [option ...] [db2-command ! sql-statement !
[? [phrase ! message ! sqlstate ! class-code]]]
option: -a, -c, -e{c|s}, -finfile, -lhistfile, -n, -o, -p, -rreport, -s, -t,
        -td;, -v, -w, -x, -zoutputfile.
```

Option	Description	Default Setting
-a	Display SQLCA	OFF
-c	Auto-commit	ON
-e	Display SQLCODE/SQLSTATE	OFF
-f	Read from input file	OFF
-l	Log commands in history file	OFF
-n	Remove new line character	OFF
-o	Display output	ON
-p	Display db2 interactive prompt	ON
-r	Save output report to file	OFF
-s	Stop execution on command error	OFF
-t	Set statement termination character	OFF
-v	Echo current command	OFF
-w	Display FETCH/SELECT warning messages	ON
-x	Suppress printing of column headings	OFF
-z	Save all output to output file	OFF

Figure 47. Output from the `db2 ? options` command

DB2 commands and statements can also be run remotely from a DB2 client on Windows and UNIX, and DB2 also provides GUI tools like DB2 Command Center and Control Center. When connecting from a remote DB2 client, a user and password must be specified on the connect command. For example, to connect from the DB2 client on `TPLEXP7`, use the `user` and `using` options with the `db2 connect to alias` command, as shown:

```
db2inst1@tplexpd7:~ > db2 connect to udblp3ox user myuser using
mypassword
```

6.1 Creating a new DB2 instance

During the installation of DB2 UDB a default instance, DB2INST1, is created by the DB2 setup program. This instance can be used for the data mart databases, as multiple databases can be created per DB2 instance. Each database can be managed on its own; i.e., database tuning parameters, buffer pools, backup, etc.

More instances can be created by running the DB2 setup program and choosing **Create a DB2 Instance** or running the db2icrt program with the necessary options (a reason to create more than one instance might be to separate a test DB2 system from a production DB2 system on the same server). Refer to Figure 39 on page 87 for information about the creation of a DB2 instance; or for an introduction to instances, refer to “Using Multiple Instances of the Database Manager” in *IBM DB2 Administration Guide : Implementation Version 7*, SC09-2944.

Each DB2 instance has its own database manager configuration file. Use the `get database manager configuration` command to list the parameters for the DB2 instance. From a remote DB2 client, first attach to the DB2 instance using the `ATTACH` command and then use the `get database manager configuration` command. To attach from the remote DB2 client on TPLEXP7, use the following command:

```
db2inst1@tplexpd7:~ > db2 attach to udblp3ox user myuser using  
mypassword
```

6.2 Creating a new database

Each DB2 database includes its own set of system catalog tables that describes the logical and physical structure of the data, a configuration file containing the parameter values allocated for the database, and a recovery log with ongoing transactions and archivable transactions. Therefore, three databases in a DB2 instance will have three separate sets of system catalog tables, database configuration file and recovery logs.

DB2 creates a separate subdirectory to store control files (such as log header files) and to allocate containers to default table spaces. You can create new table spaces in the DB2 subdirectory, but they can be stored in various other locations, including devices.

Following is a sample `create database` command to create the linux8 database:

```
db2inst1@tplexpd8:~ > db2 create db linux8 on /mnt/dasdb/database
```


The database is created in the DB2 instance that is defined by the DB2INSTANCE environment variable, or in the DB2 instance to which you have explicitly attached (using the ATTACH command). The DB2INSTANCE environment variable is defined in the db2profile script file in the \$HOME/sqllib directory. During installation of DB2, the db2profile script file is included in the DB2 instance user .profile script file. The .profile script file, located in the home directory of the user, is executed when you log into the UNIX shell (Bash or Korn).

If no ON clause is specified, the database will be created on the path value determined by DFTDBPATH in the database manager configuration file. The initial value is set to the home directory of the DB2 instance owner, DB2INST1. To list the current value, use the `get database manager configuration` command:

```
db2inst1@tplexpd8:~ > db2 get dbm cfg | grep DFTDBPATH
```

Following is the output from the `db2 get dbm cfg` command:

```
Default database path                                (DFTDBPATH) = /home/db2inst1
```

To change the DFTDBPATH value, use the `update database manager configuration` command:

```
db2inst1@tplexpd8:~ > db2 update dbm cfg using DFTDBPATH
/mnt/dasdb/database
```

Following is the output from the `db2 update dbm cfg` command:

```
DB20000I  The UPDATE DATABASE MANAGER CONFIGURATION command completed
successfully.
DB21025I  Client changes will not be effective until the next time the
application is started or the TERMINATE command has been issued. Server
changes will not be effective until the next DB2START command.
```

DB2 will create a subdirectory structure in the /mnt/dasdb/database directory, where /mnt/dasdb is a mount point of the file system on /dev/dasdb1 and database is a user-created directory using the `mkdir` Linux command. The naming scheme used on UNIX-based systems is:

```
specified_path/$DB2INSTANCE/NODEnnnn/SQL00001
```

where:

- *specified_path* is the optional, user-specified location to create the DB2 database.
- *instance name* is defined by the DB2INSTANCE environment variable or the ATTACH command.

- *NODEnnnn* is the node identifier in a partitioned database environment. The first node is NODE0000. Partitioned databases are only available with DB2 UDB EEE.
- *SQL00001* contains files associated with the first database created, and subsequent databases are given higher numbers: SQL00002, and so on.

Three initial table spaces will be created in the database directory structure as SMS table spaces:

- SYSCATSPACE for the catalog.

```
/mnt/dasdb/database/db2inst1/NODE0000/SQL00001/SQLT0000.0
```

- USERSPACE1 for the initial user table space. It can be dropped later.

```
/mnt/dasdb/database/db2inst1/NODE0000/SQL00001/SQLT0001.0
```

- TEMPSPACE1 for the initial system temporary table space.

```
/mnt/dasdb/database/db2inst1/NODE0000/SQL00001/SQLT0002.0
```

When creating the database, the first three initial table spaces can be placed in different directory structures by specifying *containers* for each table space in the create database command. To create a SMS table space for the default user table space in a specified directory, use the following `create database` command:

```
db2inst1@tplexpd8:~ > db2 "create db linux8 on /mnt/dasdb/database user
tablespace managed by system using
('/mnt/dasdc/tspaces/userspace1/container1',
'/mnt/dasdd/tspaces/userspace1/container2') "
```

Note

Notice the double quotes after the db2 command—these are used to prevent the UNIX shell from interpreting the open bracket as a special character.

The first initial placement of the active logs is in `specified_path/$DB2INSTANCE/NODEnnnn/SQL00001/SQLOGDIR/` where: `specified_path` equals `/mnt/dasdb/database` from the create database command; `$DB2INSTANCE` equals `db2inst`; `NODEnnnn` equals `NODE0000`; `SQL00001` equals the first database in the `/mnt/dasdb/database` path.

The specified path can be changed by updating the `NEWLOGPATH` parameter in the database configuration file. The new setting does not become the value of `logpath` until *both* of the following occur:

- The database is in a consistent state, as indicated by the `database_consistent` parameter.
- All users are disconnected from the database.

To list the current values of the recovery logs, use the `get db configuration` command. Use the following command to determine if the database is consistent:

```
db2inst1@tplexpd8:~ > db2 get db cfg for linux8 | grep consistent
```

The output shows that the linux8 database is consistent:

```
Database is consistent                                = YES
```

The next command shows the current value of the `NEWLOGPATH` parameter:

```
db2inst1@tplexpd8:~ > db2 get db cfg for linux8 | grep LOGPATH
```

`NEWLOGPATH` equals spaces as shown by the output from the previous command, no change is pending:

```
Changed path to log files                            (NEWLOGPATH) =
```

To list the current path to the recovery logs, use the following command:

```
db2inst1@tplexpd8:~ > db2 get db cfg for linux8 | grep Path
```

Following is the output from the previous command showing the current path for the recovery logs:

```
Path to log files                                    =
/mnt/dasdb/database/db2inst1/NODE0000/SQL00001/SQLLOGDIR/
```

The value of `NEWLOGPATH` can be changed by using the `update database configuration` command:

```
db2inst1@tplexpd8:~ > db2 update db cfg for linux8 using NEWLOGPATH
/mnt/dasdd/db2logdir
```

Changing the `NEWLOGPATH` is not immediate; as shown by the output from the `update db cfg` command, the database must be consistent and no users connected to the database, in order for the change to take effect:

```
DB20000I The UPDATE DATABASE CONFIGURATION command completed
successfully.
DB21026I For most configuration parameters, all applications must
disconnect from this database before the changes become effective.
```

DB2 diagnostic errors are recorded in the error log file db2diag.log in the path \$HOME/sqllib/db2dump, where the \$HOME environment variable equals the DB2 instance users home directory. This path value can also be changed with the `update database manager configuration` command. To list the current path to db2diag.log, use the following command:

```
db2inst1@tplexpd8:~ > db2 get dbm cfg | grep DIAG
```

Following is the output from the previous command:

```
Diagnostic error capture level          (DIAGLEVEL) = 3
Diagnostic data directory path          (DIAGPATH) =
/home/db2inst1/sqllib/db2dump
```

Use the `update dbm cfg` command to change the path for db2diag.log to /mnt/dasdd/db2dump as shown:

```
db2inst1@tplexpd8:~ > db2 update dbm cfg using DIAGPATH
/mnt/dasdd/db2dump
```

The following output shows that this change is also not immediate; this change will be effective the next time DB2 starts:

```
DB20000I  The UPDATE DATABASE MANAGER CONFIGURATION command completed
successfully.
DB21025I  Client changes will not be effective until the next time the
application is started or the TERMINATE command has been issued.  Server
changes will not be effective until the next DB2START command.
```

The following files are also associated with a database:

- SQLDBCON

This file stores the tuning parameters and flags for the database. Refer to *IBM DB2 Administration Guide : Performance Version 7, SC09-2945* for information about changing database configuration parameters.

- SQLOGCTL.LFH

This file is used to help track and control all of the database log files.

- Syyyyyyy.LOG

Database log files, numbered from 0000000 to 9999999. The number of these files is controlled by the logprimary and the logsecond database configuration parameters. The size of the individual files is controlled by the logfilsiz database configuration parameter. With circular logging, the files are reused and the same numbers remain. With archive logging, the

file numbers increase in sequence as logs are archived and new logs are allocated. When 9999999 is reached, the number wraps.

- **SQLINSLK**

This file helps to ensure that a database is used by only one instance of the database manager.

- **SQLTMPLK**

This file helps to ensure that a database is used by only one instance of the database manager.

- **SQLSPCS.1**

This file contains the definition and current state of all table spaces in the database.

- **SQLSPCS.2**

This file is a backup copy of SQLSPCS.1. Without one of these files, you will not be able to access your database.

- **SQLBP.1**

This file contains the definition of all buffer pools used in the database.

- **SQLBP.2**

This file is a backup copy of SQLBP.1. Without one of these files, you will not be able to access your database.

- **DB2RHIST.ASC**

This file is the database history file. It keeps a history of administrative operations on the database, such as backup and restore operations.

- **DB2RHIST.BAK**

This file is a backup copy of DB2RHIST.ASC

Note

To avoid potential problems, do *not* create directories that use the same naming scheme as DB2, and do not manipulate directories that have already been created by DB2.

Do *not* make any direct changes to the DB2 system files. They can only be accessed indirectly using the documented APIs and by tools that implement those APIs, including the Command Line Processor and the Control Center.

Use the Command Line Processor or Control Center to back up a database or table space.

6.3 Buffer pools

A *buffer pool* is the amount of main memory allocated by DB2 to cache table and index data pages as they are being read from disk, or being modified. The purpose of the buffer pool is to improve system performance. Data can be accessed much faster from memory than from disk; therefore, the fewer times DB2 needs to read from or write to a disk (I/O), the better the performance.

A buffer pool is associated with a table space allowing control over which data shares the same memory areas for data buffering. You can create more than one buffer pool per database with your own naming standard. Buffer pools can also be dropped or altered. DB2 creates a default buffer pool, IBMDEFAULTBP, at database creation.

Following are examples of statements to create, alter and drop buffer pools for the linux8 database:

```
db2inst1@tplexpd8:~ > db2 create bufferpool BP1 size 2000
db2inst1@tplexpd8:~ > db2 create bufferpool BP2 size 2000
db2inst1@tplexpd8:~ > db2 alter bufferpool BP1 size 2500
db2inst1@tplexpd8:~ > db2 drop bufferpool BP2
```

When creating a *new* buffer pool, the database has to be *restarted* in order for the table space assignment to the new buffer pool to take effect. Any table space that is created using the new buffer pool will use an already active

buffer pool of the same page size. The database has to be restarted for the table space assignment to the new buffer pool to take effect.

Altering the buffer pool will only take effect the *next* time the database is started. Although the buffer pool definition is transactional and the changes to the buffer pool definition will be reflected in the catalog tables on commit, no changes to the actual buffer pool will take effect until the next time the database is started. The current attributes of the buffer pool will exist until then, and there will not be any impact to the buffer pool in the interim.

6.4 Containers

A *container* is a physical storage device. It can be identified by a directory name, a device name, or a file name. A container is assigned to a table space. A single table space can span many containers, but each container can belong to only one table space.

DB2 balances the data across *all* the containers that belong to a given table space in a round-robin fashion. The number of pages that DB2 writes to one container before using a different one is called the *extent size*.

Following is an example `create tablespace` statement from the linux8 database with a container-string of `/mnt/dasdc/dmsts/tslinei`.

```
db2inst1@tplexpd8:~ > db2 "create tablespace TST1 managed by database
using (file '/mnt/dasdc/dmsts/tslinei' 256000) extentsize 32
prefetchsize 64 bufferpool BP1"
```

Each container-string can be an absolute or relative directory name. The directory name, if not absolute, is relative to the database directory. DB2 creates any component of the directory name that does not exist. When a table space is dropped, all components created by DB2 are deleted.

DB2 will return an error if the directory identified by container-string exists and the directory contains any files or subdirectories.

6.5 Creating table spaces

A *table space* is a place to store data from tables, indexes and large objects (LOBS). When creating a table, you can decide to have certain objects such as indexes and large object (LOB) data kept separately from the rest of the table data. A table space can also be spread over one or more containers. Figure 48 on page 122 shows the flexibility you have with DB2 table spaces.

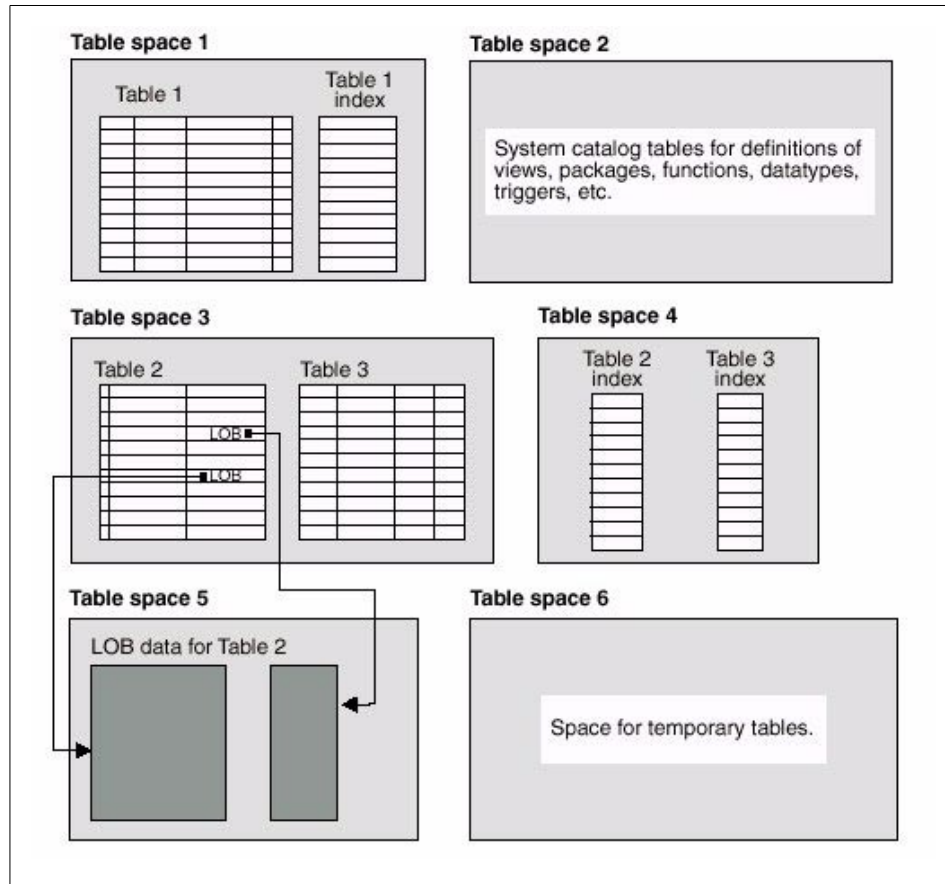


Figure 48. DB2 table spaces flexibility

A table space can be either system-managed space (SMS), or database-managed space (DMS). For an SMS table space, each container is a directory in the file space of the operating system, and the operating system's file manager controls the storage space. For a DMS table space, each container is either a fixed size preallocated file, or a physical device such as a disk, and DB2 controls the storage space. Figure 49 on page 123 shows the physical management of DB2 table spaces.

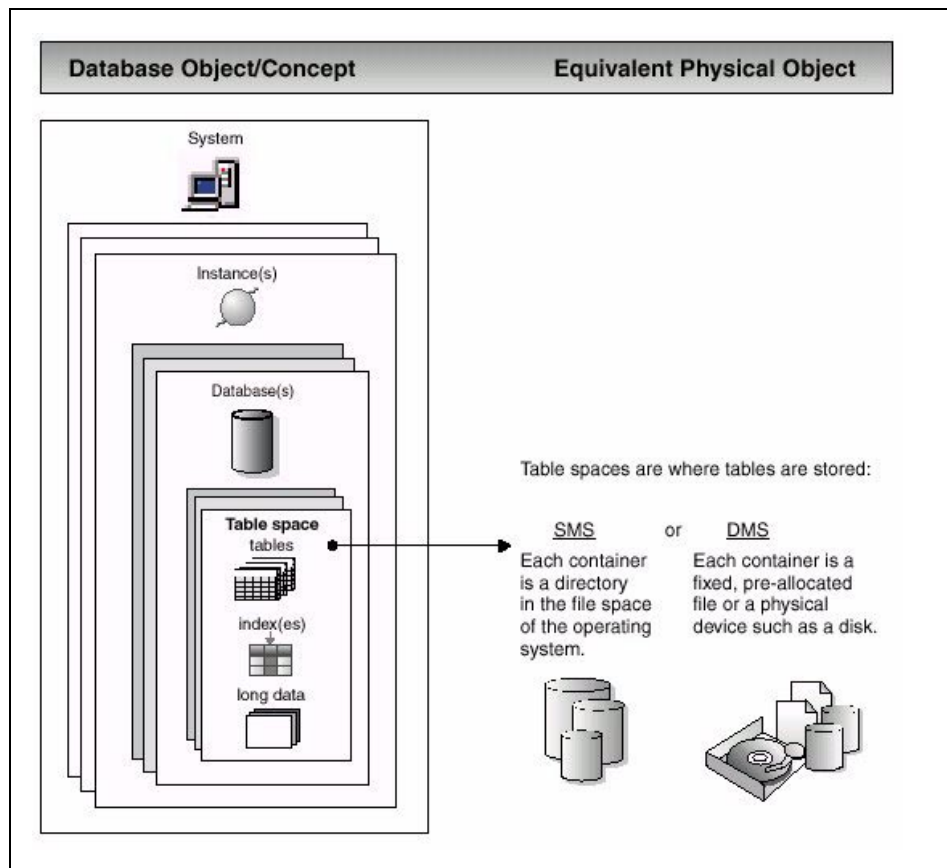


Figure 49. DB2 table space physical management

Three types of table spaces can be created:

- Regular table space stores tables and indexes.
- Temporary table space stores temporary data:
 - System temporary table space stores temporary tables created by DB2 to perform operations such as sorts or joins. A system temporary table space (TEMPSPACE1) is created automatically when a database is created.
 - User temporary table space stores declared global temporary tables. No user temporary table spaces exist when a database is created. At least one user temporary table space should be created with appropriate USE privileges, to allow definition of declared temporary tables.

- Long table space stores long or LOB table columns. It may also store structured type columns. The table space must be a DMS table space.

Following are examples for creating a system temporary table space as system-managed space (SMS) or database-managed space (DMS):

- SMS-managed table space:

```
db2inst1@tplexpd8:~ > db2 "create system temporary tablespace tempsms
managed by system using ('/mnt/dasdc/tspaces/tempsms') extentsize 32
prefetchsize 64 bufferpool BP1"
```

SMS-managed tempsms is a directory with files managed by the filesystem.

- DMS-managed table space:

```
db2inst1@tplexpd8:~ > db2 "create system temporary tablespace tempdms
managed by database using (file '/mnt/dasdc/tspaces/tempdms' 500)
extentsize 32 prefetchsize 64 bufferpool BP1"
```

DMS-managed tempdms is a preallocated file managed by DB2.

SMS table spaces need no space management, and the growth of the file is limited by the amount of available space in the filesystem. The disadvantages of SMS table spaces are lack of performance due to the filesystem overhead, and noncontiguous space allocation, which increases I/O overhead.

DMS table spaces are preallocated and is managed by DB2, and therefore perform better than SMS table spaces. The disadvantage of DMS table spaces is space management. Procedures must be in place to monitor growth and add new containers if needed. Use the `db2 list tablespaces show detail` command to monitor space usage as follows, or use the Control Center:

```
db2inst1@tplexpd8:~ > db2 list tablespaces show detail | less
```

You can scroll the output from the `list tablespaces` command by using the up and down arrow keys until you see the information for your table space. The following output includes used pages, free pages, and high water mark for the table space:

```
Tablespace ID          = 6
Name                   = TSLINEI
Type                   = Database managed space
Contents               = Any data
State                  = 0x0000
Detailed explanation:
    Normal
```

Total pages	= 256000
Useable pages	= 255968
Used pages	= 179840
Free pages	= 76128
High water mark (pages)	= 179840
Page size (bytes)	= 4096
Extent size (pages)	= 32
Prefetch size (pages)	= 64
Number of containers	= 1
Minimum recovery time	= 2000-11-18-15.59.18.000000

When running out of space, use the `alter tablespace` statement with the `add container-clause` as follows:

```
db2inst1@tplexpd8:~ > db2 "alter tablespace tslinei add (file
'/mnt/dasdd/dmsts/tslinei' 256000)"
```

DB2 automatically rebalances the contents of the table space across the containers. Access to the table space is not restricted during the rebalancing.

Note

Creating and displaying DB2 database objects such as buffer pools, table spaces, tables, views, indexes, and schemas requires a database connection:

```
db2inst1@tplexpd8:~ > db2 connect to linux8 user myuser using mypassword
```

To protect your password, do not specify the `using` clause of the `connect` command. The DB2 Command Line Processor will then prompt you for your password.

6.6 Creating tables

The `CREATE TABLE` statement defines a table. The definition must include its name and the names and attributes of its columns. The definition may include other attributes of the table, such as its primary key or check constraints.

Following is the `create table` statement for the T1 table:

```
db2inst1@tplexpd8:~/ddl > cat create_tbl.ddl
```

```
CREATE TABLE T1 (
  L_ORDERKEY INT NOT NULL,
```

```

L_PARTKEY INT NOT NULL,
L_SUPPKEY INT NOT NULL,
L_LINENUMBER INT NOT NULL,
L_QUANTITY REAL NOT NULL,
L_EXTENDEDPRICE REAL NOT NULL,
L_DISCOUNT REAL NOT NULL,
L_TAX REAL NOT NULL,
L_RETURNFLAG CHAR(1) NOT NULL,
L_LINESTATUS CHAR(1) NOT NULL,
L_SHIPDATE DATE NOT NULL,
L_COMMITDATE DATE NOT NULL,
L_RECEIPTDATE DATE NOT NULL,
L_SHIPINSTRUCT CHAR(25) NOT NULL,
L_SHIPMODE CHAR(10) NOT NULL,
L_COMMENT VARCHAR(44) NOT NULL)
IN TST1;

```

The T1 table will be defined in the TST1 tablespace. No indexes will be created, therefore no INDEX IN clause. For a complete list of column data types and options, refer to *IBM DB2 SQL Reference Version 7, SC09-2975*.

6.7 Creating indexes

All indexes from one table can only be in one specific table space specified on the create table statement, but a table space can contain the indexes of more than one table.

The INDEX IN clause of CREATE TABLE identifies the table space in which any indexes on the table will be created. This option is allowed only when the primary table space specified in the IN clause of CREATE TABLE is a DMS table space.

The specified table space must exist, and must be a REGULAR DMS table space over which the authorization ID of the CREATE TABLE statement has USE privilege.

6.8 Recommendations for database objects

Separate tables and indexes by specifying different table spaces in the CREATE TABLE statement. To improve I/O performance, place the containers on different physical storage devices and controllers if possible.

Use SMS table spaces for test application data, and use DMS table spaces for production data.

When creating a database, use an SMS table space for the catalog table space in most cases. For temporary table spaces, use SMS table spaces if you cannot determine the size for temporary space needed for most queries; otherwise, use DMS table spaces for the best performance. Monitor space problems on the temporary table spaces and add more containers if needed.

Ensure also that the DB2 logs are on a separate device from your application data and indexes.

Chapter 7. Scalability and performance tests for data population

Data population scalability and performance are the most important issues in data warehouse and data mart environments. When transferring large amounts of data from the S/390 to the data marts residing outside the S/390, network traffic can be a big issue. This is why data mart consolidation on a single S/390 server where the data warehouse data also resides (with appropriate network connectivity links such as OSA-2, OSA-Express, CTC, Virtual CTC, and Hipersockets), can be a unique opportunity to improve network traffic performance between the data warehouse and the data marts, and reduce data population batch windows.

At the IBM S/390 Teraplex Integration Center we tested a few simple data population scenarios for preliminary scalability and performance evaluation of those techniques on the DB2 for Linux on S/390 platform. The scope of our tests was an atomic-level analysis of simple data extraction and insertion techniques. We tested the following data population scenarios for source data residing on S/390:

- SQL-based scenarios, including:
 - Single distributed SQL statement
 - Select/Insert SQL statements in a C program
- DB2 utilities-based scenarios, including:
 - DB2 DSNTIAUL, FTP, and DB2 Load
 - DB2 High Performance Unload
 - DB2 Export and Import
 - DB2 Export and Load

7.1 Test environment

All scenarios used a DB2 for OS/390 data warehouse and DB2 for Linux data marts, all consolidated on S/390.

Figure 50 on page 130 describes the data population scenarios we implemented in our test environment.

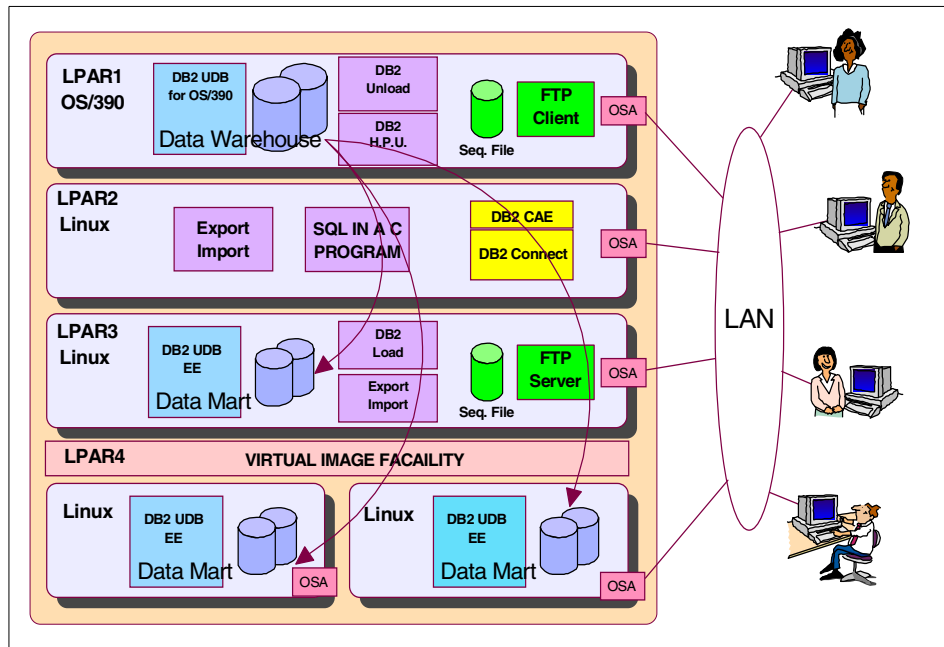


Figure 50. Data population test scenarios

Data was extracted from the data warehouse and inserted/loaded into a data mart using the following configuration:

- LPAR1 runs the OS/390 operating system with the DB2 data warehouse.
- LPAR2 runs the Linux operating system and is defined as the network gateway and application partition.
- LPAR3 runs the Linux operating system and is defined as the native Linux data mart partition. Data comes from the data warehouse in LPAR1 and is loaded in the data mart in LPAR3.
- LPAR 4 runs the Virtual Image Facility (VIF) environment, under which run multiple Linux virtual images, each housing a DB2 for Linux data mart. This LPAR offers an alternative for the native Linux data mart. The data population techniques are the same in a virtual Linux image as in a native Linux LPAR.

Our data model was based on three tables with the following characteristics:

Table 6. Characteristics of tables used in data population test

Name of table	No. of columns	Avg. length of row	No. of rows
T1	16	11	6.3 million
T2	9	17	1.4 million
T3	4	49	25

You can find the DDL statements we used to define those three tables in Appendix 2.

We evaluated each scenario, transferring different amounts of data from the data warehouse T1 table to the data mart:

- 1 MB
- 5 MB
- 100 MB
- 1 GB

We used 1 CP.

We used no index and did not investigate the index effect on cases where the table space scan proved to be inefficient.

7.2 Single distributed SQL statement

A great performer among all our data population tests was the single *distributed SQL* statement enabled by a federated database, which was DB2 for Linux on S/390, in our case.

7.2.1 Federated database

A *federated database* system is a database management system (DBMS) that supports applications or users submitting distributed SQL statements referring to two or more DBMSs in a single SQL statement.

DB2 UDB version 7.1 for Linux on S/390 provides support for distributed requests across DBMSs. For example, with a distributed request you can perform a UNION operation between a DB2 table and an Oracle view in the same SQL statement.

A DB2 federated system provides location transparency for database objects. A DB2 federated system consists of a DB2 UDB instance which serves as a federated database, and one or more data sources. Applications connect to the DB2 federated database and send their queries to it. Figure 51 shows the DB2 federated database; this database retrieves the data from the referenced data sources (such as the DB2 family and Oracle), consolidates the result set, and returns it to the application. The distributed SQL function supported by the DB2 federated database is actually the first phase of integration of DB2 DataJoiner into DB2 Universal Database.

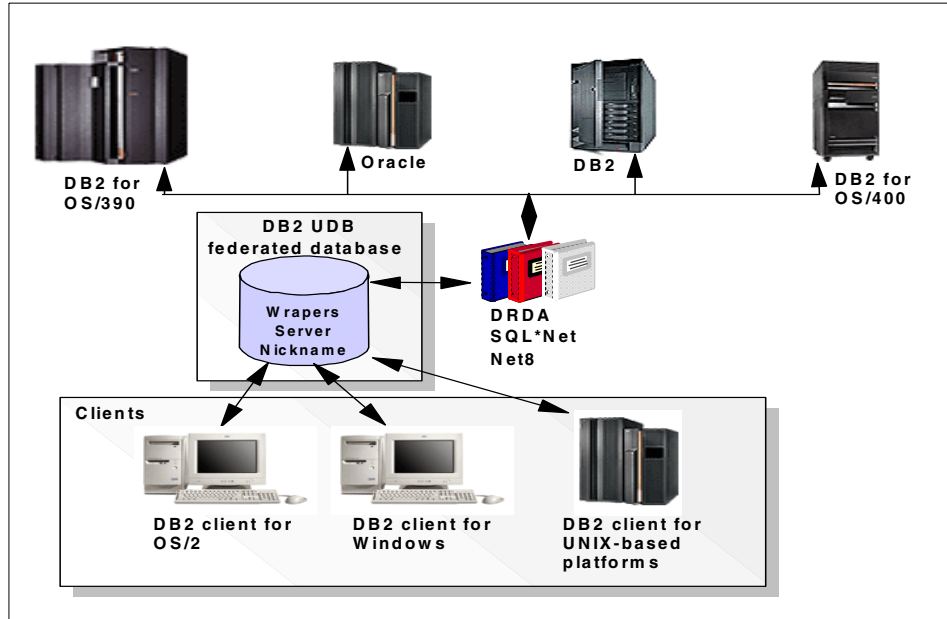


Figure 51. DB2 federated database system

For more details about DB2 federated database support, refer to *DB2 Administration Guide: Planning*, SC09-2946.

7.2.2 Distributed SQL scenario

In our test environment, DB2 for Linux on LPAR3 acted as the federated database as well as the data mart. We tested the distributed SQL function using the DB2 for Linux federated database on LPAR3 to read from the DB2 data warehouse on LPAR1, and insert into the data mart residing on the same LPAR3.

Figure 52 illustrates our distributed SQL statement scenario.

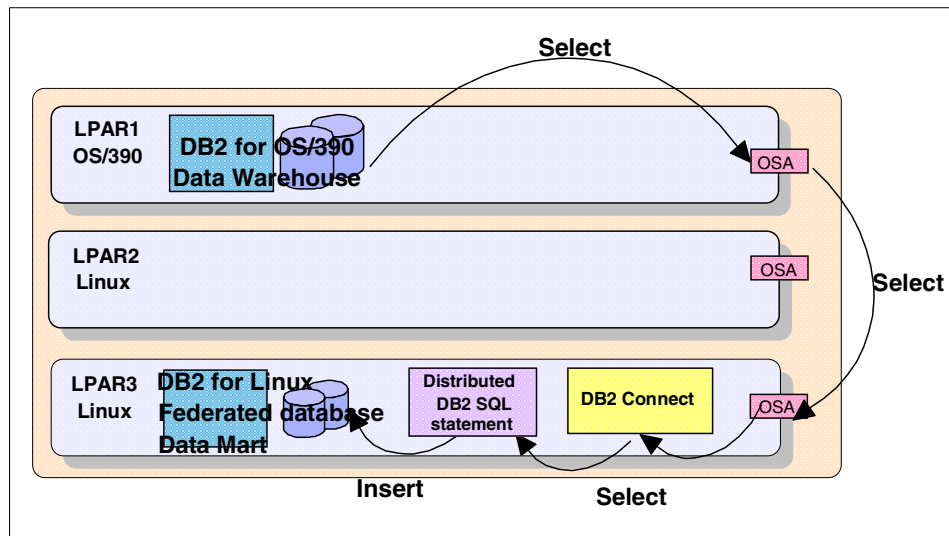


Figure 52. Single SQL statement combining both Select and Insert clauses

From the DB2 Control Center in LPAR3 we issued the following distributed SQL statement:

```
insert into marttbl1 select * from line390 where l_orderkey < 7895169
```

where `marttbl1` is the DB2 for Linux table, and `line390` is the DB2 for OS/390 table.

Using DB2 Connect for Linux, the federated system connects to DB2 for OS/390 in LPAR1 and reads the rows from the data warehouse. Next, the federated system inserts locally the rows in the DB2 for Linux data mart.

A detailed explanation of the implementation of this solution is described in C.1, “Distributed SQL statement” on page 185.

We tested the distributed SQL scenario against Table 1 to transfer 1 GB of data volume. This function demonstrated great performance potential, as we compared its results with the other tools we tested to scan 1 GB of data.

Figure 53 on page 134 shows the rows/second rate summary for 1 GB data.

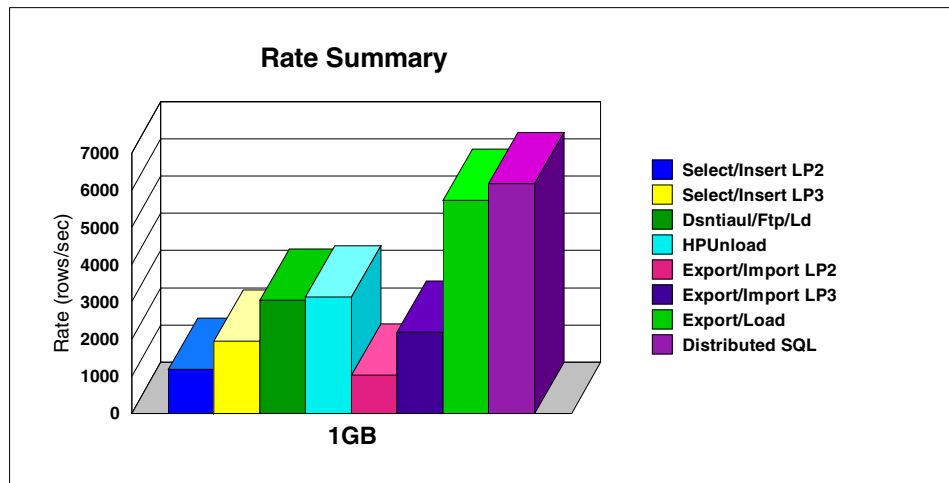


Figure 53. Positioning of Distributed SQL scanning 1 GB data

In Figure 53, the last bar on the right represents the results in terms of number of rows transferred per second when scanning and inserting 1 GB of data. The single distributed SQL proved to be the best performer in our test cases. The following factors proved to be beneficial:

- There is no external networking overhead for the connection between LPAR1 and LPAR3; we benefit from the performance of the OSA connection.
- The system performs a block fetch from DB2 for OS/390 followed by iterative inserts into DB2 for Linux, in one execution of SQL statement.

7.3 Select/Insert SQL statements in a C program

The C program extracts data from the data warehouse and inserts it into the data mart. The program issues a Select statement to read rows from the source T1 table and inserts them into the target table as follows:

- Connect to the DB2 source database.
- Connect to the DB2 target database.
- Open a cursor.
- Select the columns in T1 table using an *orderkey* statement (which enables the selection of the right amount of data needed to perform scalability tests).
- Set connection to the target database.
- Insert the columns into the target database.
- Set connection back to the source database.

- Go to the next row.
- Commit the changes in the target database and release the connections.

Appendix 2 provides the sample C program and related scripts and make files we used to run this program.

Figure 54 shows two implementations of the data population scenario using Select/Insert SQL statements in a C program.

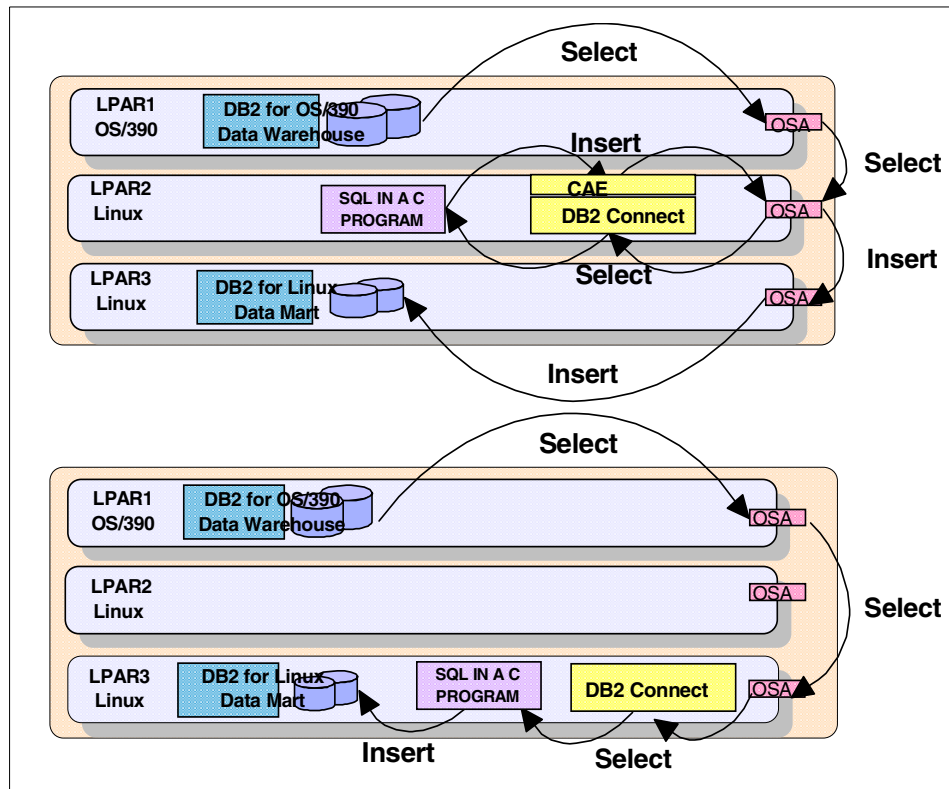


Figure 54. Select/Insert SQL statements in a C program

In the first implementation, the C program runs in LPAR2. The program issues a Select statement that goes through DB2 Connect, the Open System Adapters (OSA) and reaches the DB2 for OS/390 in LPAR1. DB2 for OS/390 sends the data back to the C program, which issues an Insert statement. The data is sent through DB2 Client Application Enabler (CAE), the OSA adapter and reaches DB2 for Linux in LPAR3, which writes the data in the target DB2 table.

In the second implementation, the C program runs in LPAR3 next to the data mart. The program issues a Select statement that goes through DB2 Connect, the Open System Adapters (OSA) and reaches the DB2 for OS/390 in LPAR1. DB2 for OS/390 sends the data back to the C program which issues an Insert statement. The data is sent locally through DB2 Client Application Enabler (CAE) to DB2 for Linux, which writes the data in the target DB2 table.

As shown in Figure 55, the second implementation performs better than the first one. The figure shows the number of rows transferred per second for the four test cases (1 MB, 5 MB, 100 MB, 1 GB of data transfer) when the C program runs in LPAR2, and when it runs in LPAR3 next to the data mart.

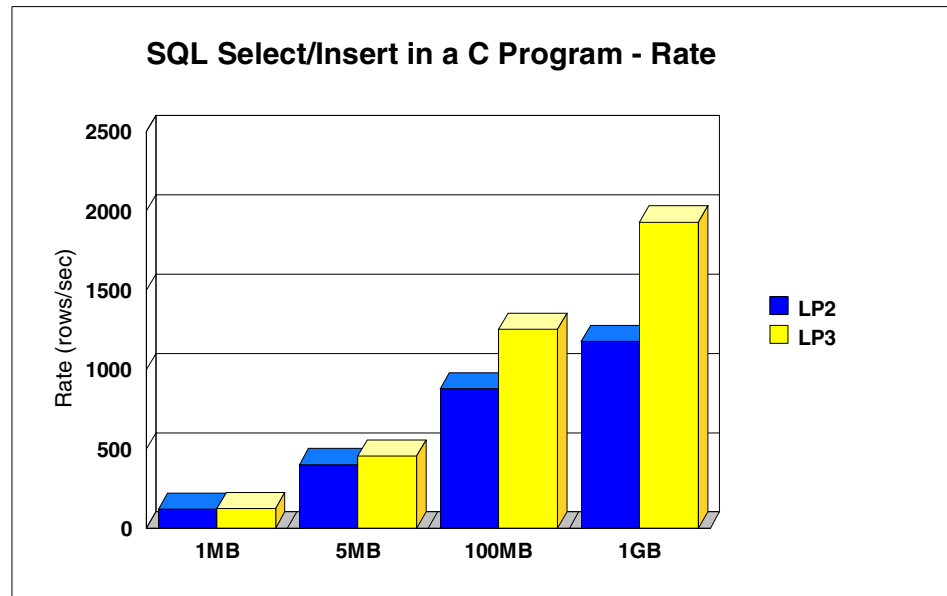


Figure 55. Transfer rate for SQL Select/Insert in a C program in different LPARs

The test clearly shows that the program performs better when it executes locally in the same LPAR as the data mart. The Inserts perform better when they access locally the data mart in LPAR3.

The source code we used for this test is listed in C.2, "C program source code with Select/Insert SQL statements" on page 187.

7.4 DB2 DSNTIAUL and DB2 Load

DB2 DSNTIAUL utility runs in LPAR1. It unloads the data from the DB2 for OS/390 data warehouse. The data is stored in an OS/390 sequential file, and then transferred directly to the Linux LPAR3 using FTP in an ASCII format. The format conversion is done by the FTP Server. The FTP Server can run either in the OS/390 environment with a Linux FTP client, or in the Linux environment with an OS/390 FTP client. After the sequential file is transferred to the LPAR1 environment, it can be loaded in the DB2 for Linux data mart using the DB2 Load utility.

Figure 56 illustrates the scenario using DB2 DSNTIAUL and Load utilities.

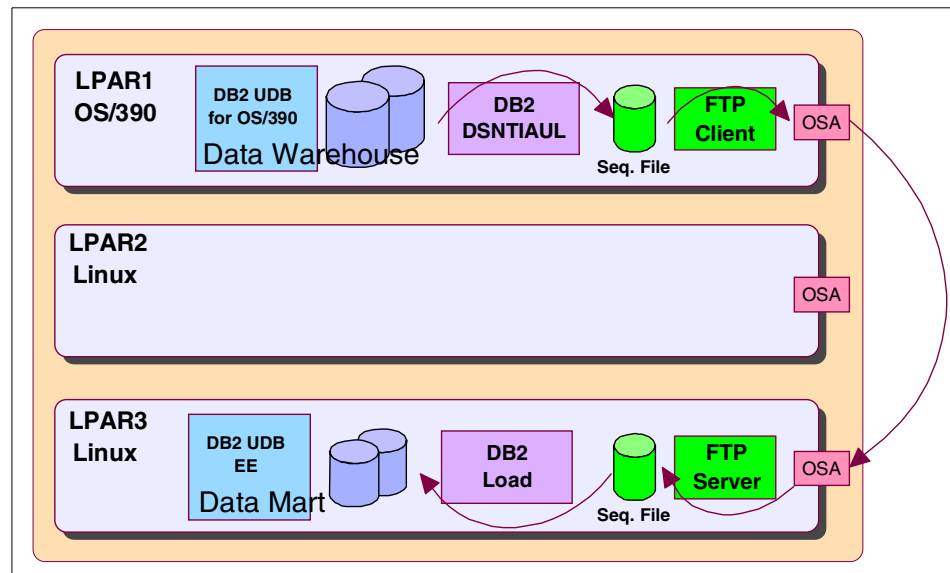


Figure 56. DB2 DSNTIAUL and Load utilities

DB2 DSNTIAUL runs locally next to the source data warehouse in LPAR1 and unloads the table T1 in a sequential file, which is then FTPed using OSA to a sequential file in LPAR3. The FTP client runs in LPAR1 and the FTP server runs in LPAR3. DB2 Load runs locally next to the data mart in LPAR3 and loads the content of the sequential file into the target data mart residing on the same LPAR.

Figure 57 on page 138 shows the total number of rows per second transferred with DSNTIAUL, FTP, and Load utilities.

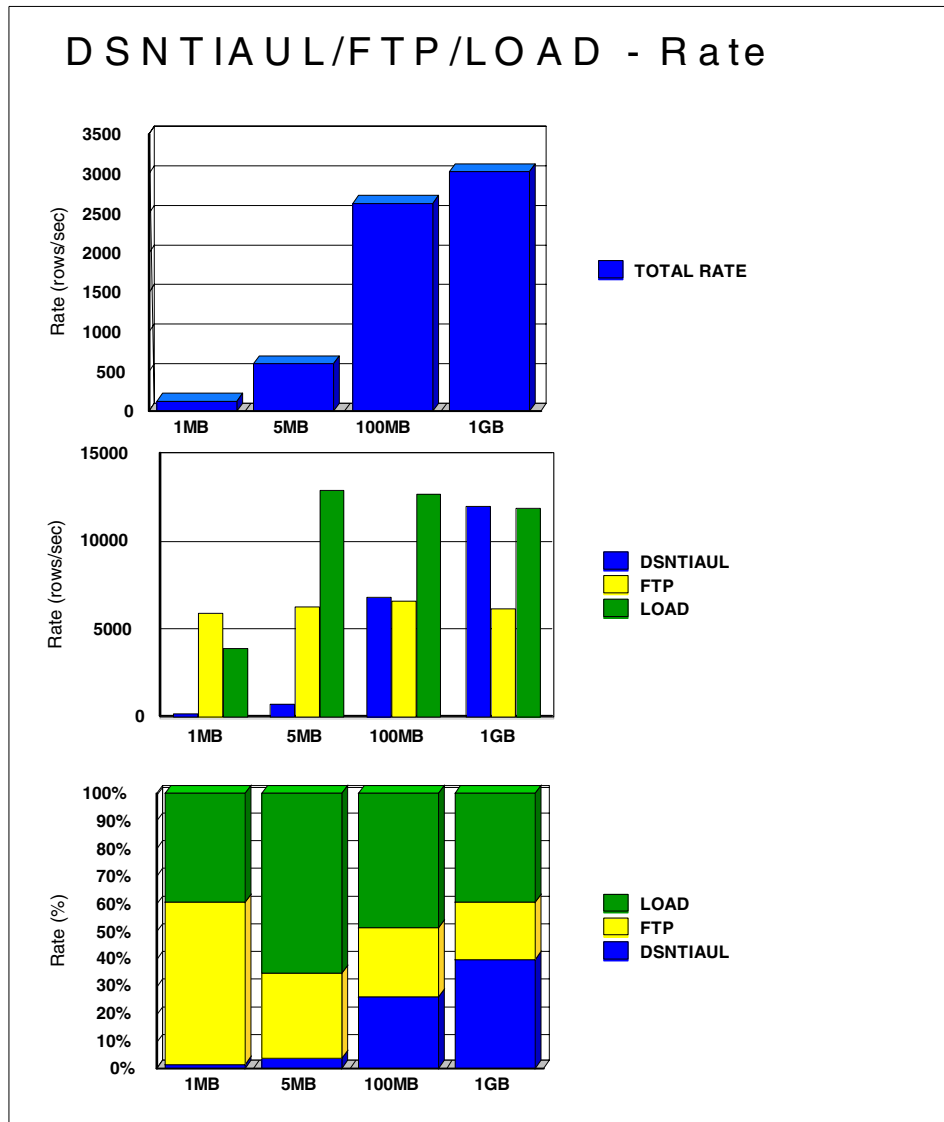


Figure 57. Rows/second transferred with DSNTIAUL/, FTP, and Load utilities

This test showed that the transfer rate, especially for DSNTIAUL, is much more efficient with large volumes than with small volumes of data. When a small number of rows is returned, most of the time is spent scanning the table space, which is not very efficient. Small volumes of data would most likely benefit from indexes, but we did not test this case.

7.5 DB2 High Performance Unload (HPU)

IBM DB2 High Performance Unload (HPU) for OS/390 V2, program number 5655-E69, is a new, high-speed DB2 utility for unloading DB2 tables from either a table space or an image copy. Tables are unloaded to one or more files based on a format specified by the user.

HPU performs sequential reading and accessing DB2 data at high speed. HPU can scan a table space and create output files in the format you need them in. It also lets you unload partitioned table spaces in parallel.

Other capabilities include multiple output formats (with the opportunity to tailor your own), a comprehensive and powerful SELECT statement, and the ability to unload from image copies as well as active tables.

HPU can do multiple unloads of the same table space or image copy data. HPU has functions to help you manage and control the unload activity. HPU is “outside” DB2, directly accessing the VSAM or sequential files which contain the table space or image copy data set, respectively.

HPU design provides superior performance, especially in CPU time.

HPU performs the following tasks:

- It unloads table spaces quickly.
- It executes, simultaneously, several unloads accessing the same table space.
- It unloads against an image copy to eliminate interference with DB2 production databases. The image copy can be:
 - The last full image copy.
 - Any full image copy.
 - An incremental image copy (cannot be compressed by DB2).
- It unloads selected rows and columns.
- It unloads every n rows and maximum rows.
- It generates load control statements for subsequent reload.
- It inspects, modifies, or discards DB2 rows, using the HPU user exit.

Figure 58 on page 140 gives an overview of the HPU.

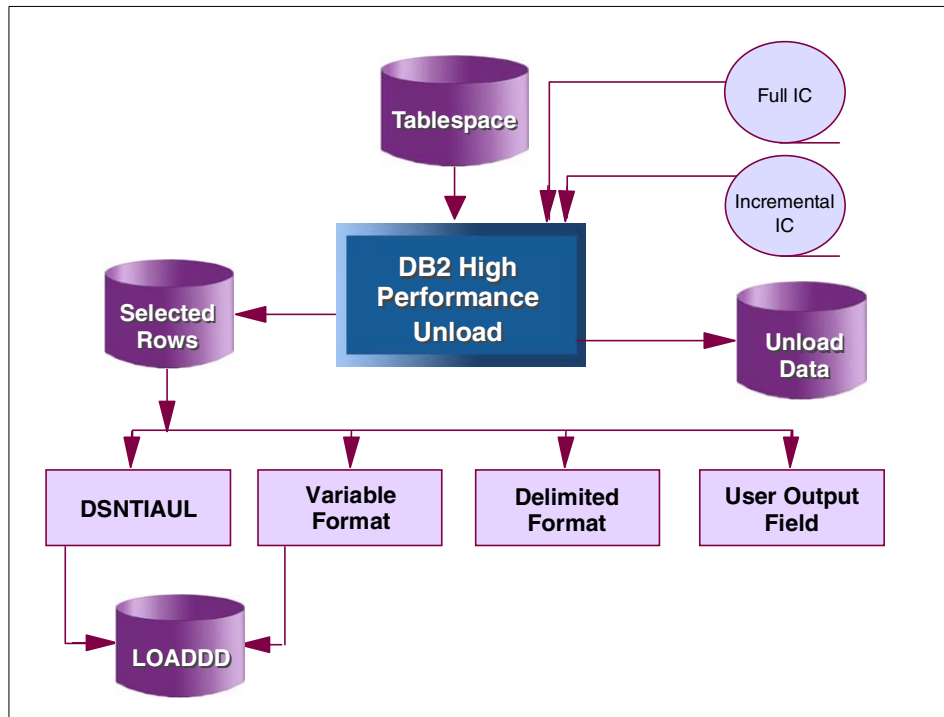


Figure 58. DB2 High Performance Unload utility overview

HPU runs on any hardware supported by DB2 for OS/390. DB2 for OS/390 Version 3 (or later) must be installed.

HPU provides an extremely fast way to sequentially read and share a DB2 table space among multiple unloads. It scans a table space and creates the number of output files you specify, in the format you specify. The output format can be:

- DSNTIAUL-compatible
- VARIABLE, which lets you quickly create variable-length records
- DELIMITED, which lets you quickly create a delimited file that can be exported to another platform
- USER, which lets you specify virtually any type of conversion so that your output appears as you want it

You can code as many SELECT statements as necessary for tables belonging to the same table space. Different output files can be created during the same unload process at almost no additional cost. For example,

you can unload a list of customers having payments due this week, and another list of customers having birthdays on the first day of the week. These lists could be created in a single execution of HPU at a fraction of the cost required by traditional dual unload executions. This high level of performance is achieved while producing multiple outputs by executing processes in parallel against a single read of the table space.

HPU can also be run against image copies of the table space, thereby avoiding interference with DB2 production databases.

To maximize performance, HPU uses buffering and synchronization techniques.

- Buffering

When reading data rows, HPU directly accesses the VSAM clusters containing the table space. This direct use of VSAM takes maximum advantage of VSAM's buffering capability, which lets an entire cylinder be read with a single I/O.

- Synchronization

HPU allows the simultaneous execution of several unloads that are accessing the same table space; it does this by synchronizing the unloads. Each row extracted from a table space is offered successively to each of the unloads executed under DB2 Unload control, so that a single read satisfies several requests.

Using HPU is simple. There is an UNLOAD command and an optional SELECT statement. The syntax of the SELECT statement is compatible with the syntax of the DB2 SELECT statement.

Once the UNLOAD operation is complete, HPU is operational. By optimizing sequential reads of the table space, HPU will reduce both the elapsed and the CPU time needed to execute the unloads.

Figure 59 on page 142 shows the data population scenario using the DB2 High Performance Unload utility.

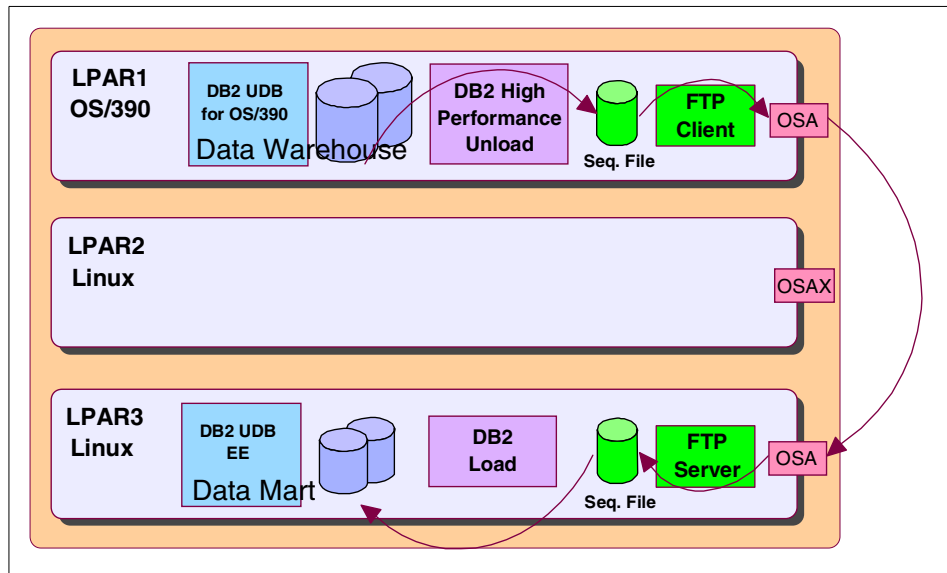


Figure 59. The DB2 High Performance Unload utility

HPU runs in LPAR1 under OS/390. It creates a sequential file that is transferred using FTP, through the network, to the LPAR3 partition where the data mart to be populated resides. Then, the DB2 Load utility is used locally on LPAR3 to load the data mart tables.

Figure 60 on page 143 shows the test results of the HPU/FTP/LOAD scenario. The solution is much more efficient with large volumes of data. Here, also, small volumes of data would benefit from an index.

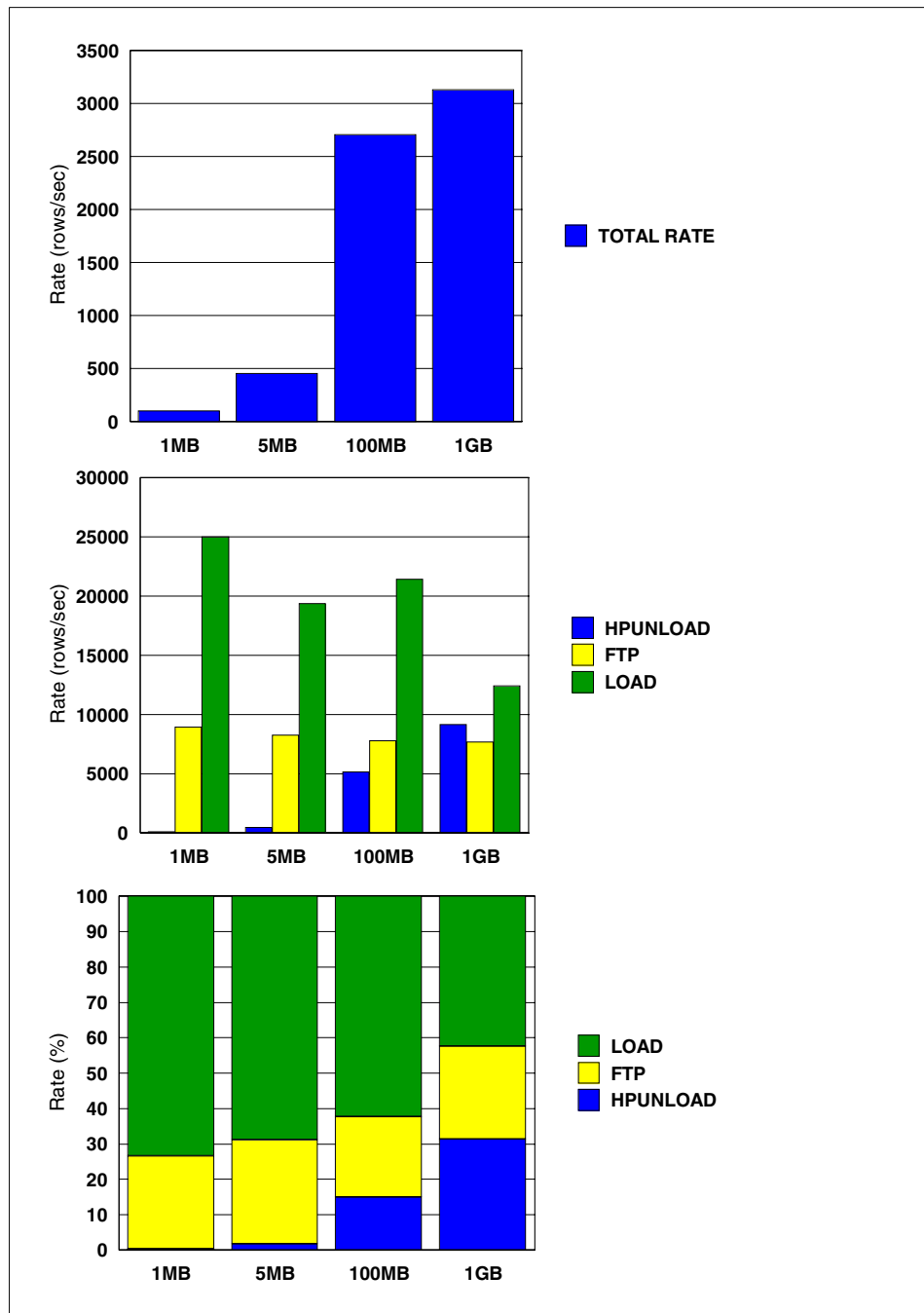


Figure 60. Rows/second transferred with HPU, FTP, and Load utilities

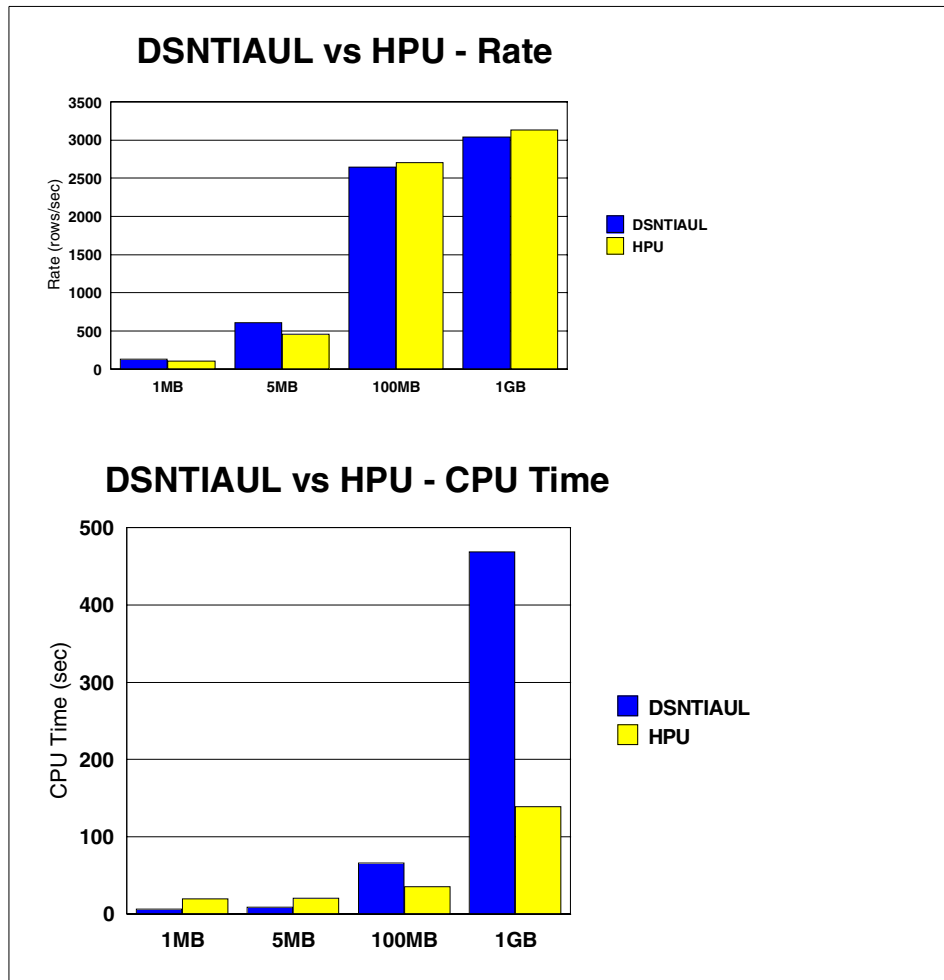


Figure 61. DSNTIAUL and HPU comparison

Figure 61 shows a comparison between DSNTIAUL and HPU. From an elapsed time perspective, they performed equivalently; however, HPU has CPU savings associated.

7.6 DB2 Export and Import

DB2 Export and Import utilities do not run on the OS/390 platform. In our scenario, they run in the Linux environment. The DB2 Export utility unloads data from a DB2 table to a sequential file. The supported output file format is the PC Integrated Exchange Format (PC/IXF). PC/IXF format is a structured description of a database table that contains an external representation of the internal table. A table with attributes that are compatible with the data must exist before you can import to it. Import through DB2 Connect cannot create a table, because Insert is the only supported option.

Figure 62 illustrates the scenario using the DB2 Export/Import utilities.

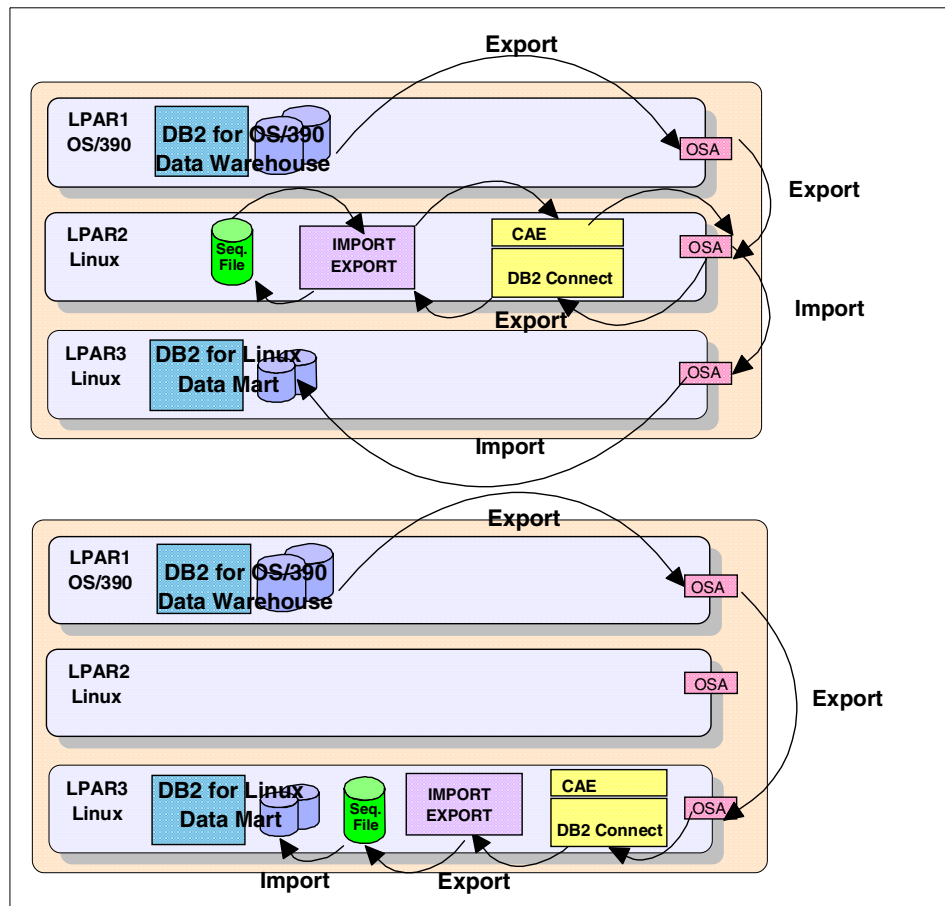


Figure 62. DB2 Export and Import utilities

The DB2 Import utility inserts data from a sequential input file into a DB2 table. If the table receiving the imported data already contains data, you can either replace or append to the existing data.

In the first implementation of our scenario, the DB2 Export/Import utilities ran in LPAR2. Using DB2 Connect, the script connected first to the source DB2 for OS/390 database in LPAR1. It then issued an Export command to unload data from the DB2 for OS/390 table into a sequential file on LPAR2. This file is defined using the PC/IXF format. Then, using the CAE, the script connected to the target DB2 for Linux database in LPAR 3 and issued an Import command to insert the sequential file contents into the target table in LPAR3.

In the second implementation of our scenario, the DB2 Export/Import utilities ran locally next to the data mart in LPAR3. The mechanism was the same, but the performance of the Import was much better when the inserts into the data mart were done locally, as shown in Figure 63 on page 147. This is because we avoided the additional network connection with LPAR2. Export also performed better when large number of rows are returned, which justifies the table space scan. When fewer rows are returned, an index on the source table should help extract the rows faster.

The sample scripts used to populate DB2 Linux data mart from DB2 OS/390 data warehouse using DB2 Export / Import utilities are listed in C.5, "DB2 Export/Import script" on page 197.

Export/Import - Rate

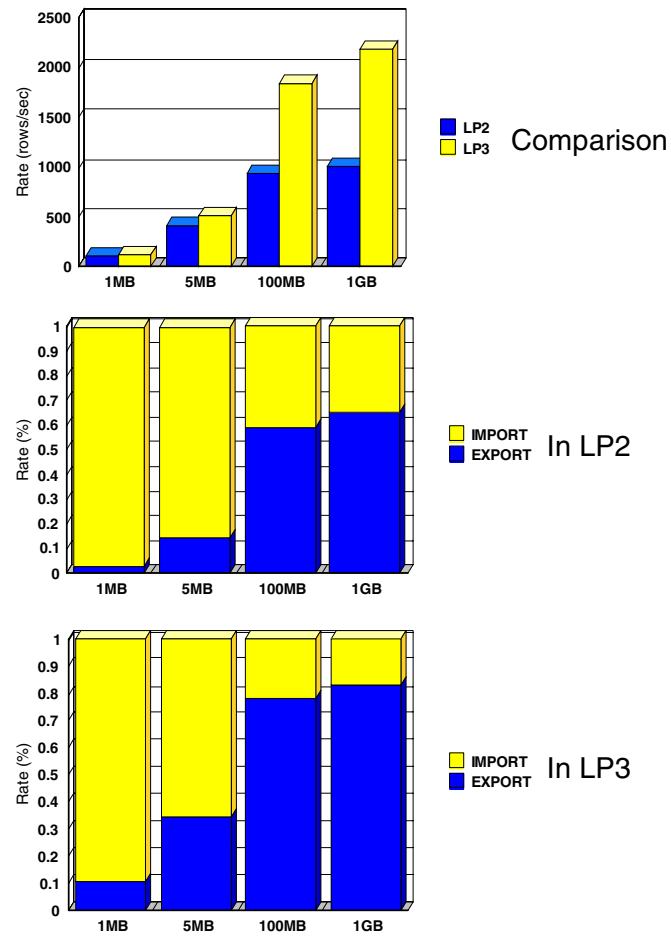


Figure 63. Export/Import rate in LP2 and LP3

7.7 DB2 Export and Load

This scenario is similar to the previous one, but used the DB2 Load utility instead of the DB2 Import utility to load the data into the target database.

The Load utility is capable of efficiently moving a large quantity of data into newly created tables, or into tables that already contain data. The Load utility can use a PC/IXF file previously created by the Export utility.

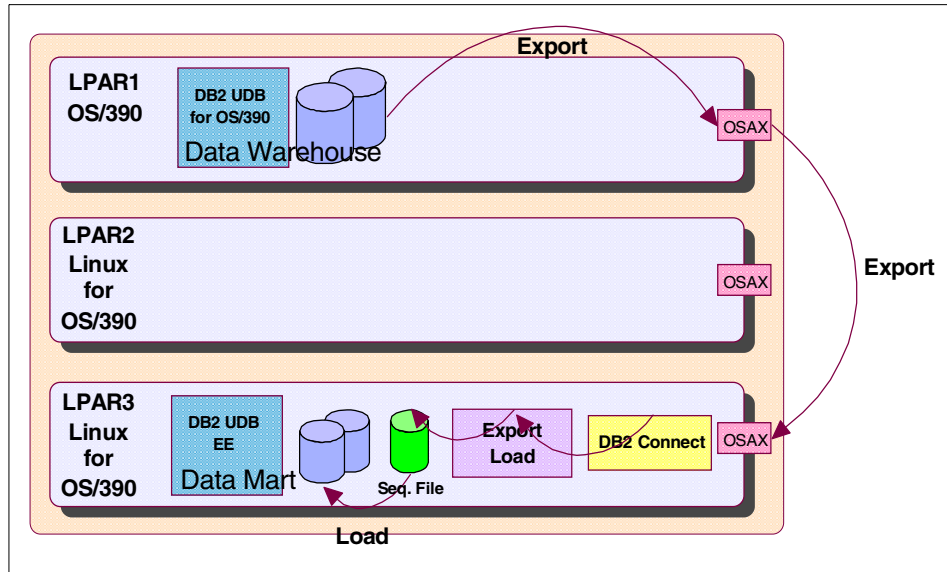


Figure 64. Data population scenario using DB2 Export and Load utilities

Export and Load can run in LPAR2 or LPAR3. These utilities perform better when they run in LPAR3, local to the data mart, because this avoids additional connections to intermediary LPARs.

Figure 65 on page 149 shows the row transfer rate of Export/Load in LPAR3. They perform better with large volumes of data.

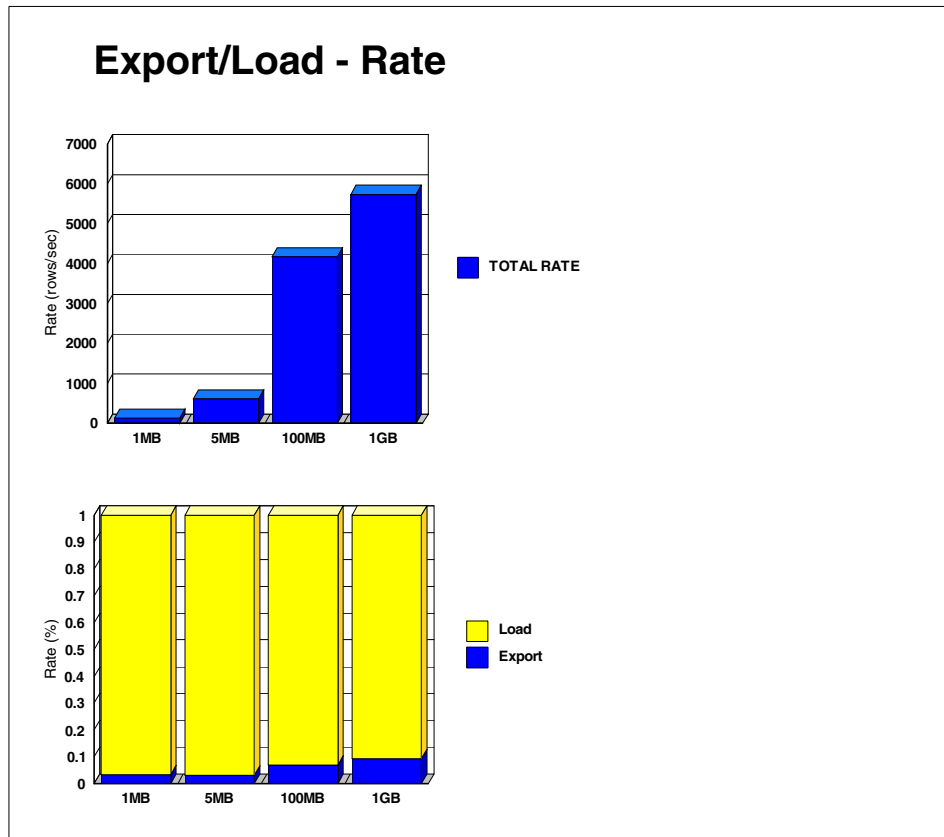


Figure 65. Export and Load rate in LP3

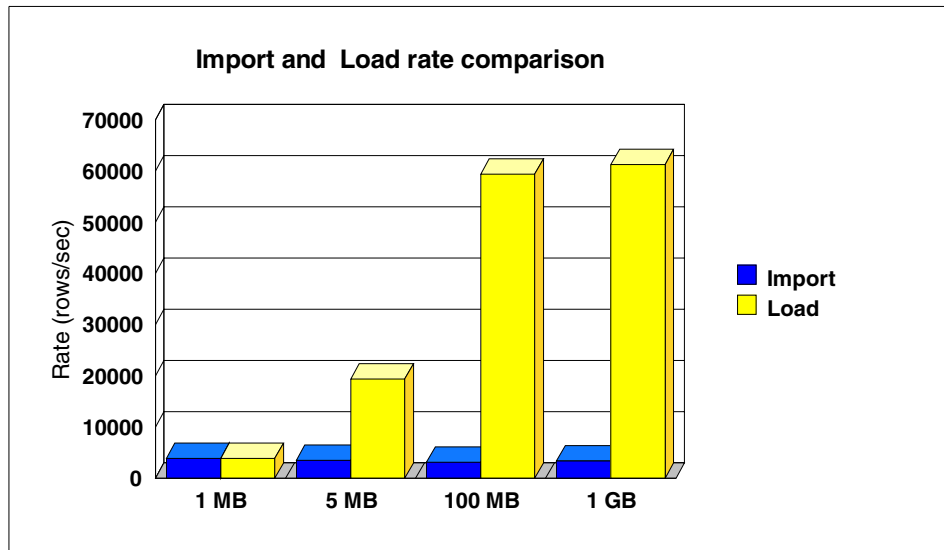


Figure 66. Import and Load rate comparison

Figure 66 shows the Import and Load rate comparison. The Load utility is faster than the Import utility because it writes formatted pages directly into the database, while the import utility performs SQL INSERTs.

C.6, "DB2 Export/Load scripts" on page 198 has the sample scripts used to populate DB2 Linux data mart from DB2 OS/390 data warehouse using DB2 Export/Load utilities.

7.8 Data population performance summary

The following figures summarize the performance results, in terms of rows/second, of the different tests we run. Figure 67 shows the rows/second rate comparisons for the four test cases we ran (1 MB, 5 MB, 100 MB, 1 GB result sets) in all our data population scenarios.

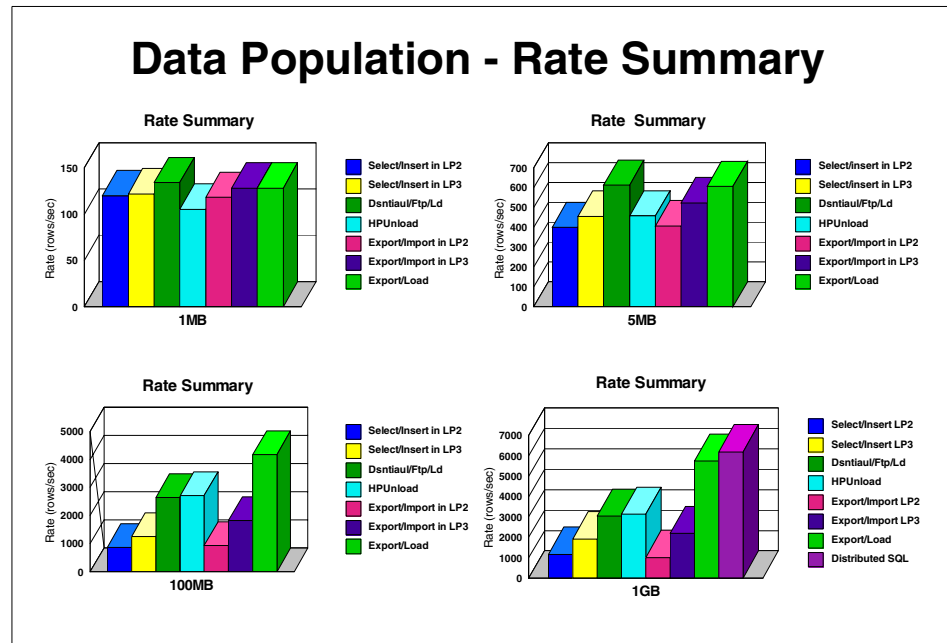


Figure 67. Rate summary

By running the data population scenarios in four different test cases, we wanted to observe the primitives of each tool and show an example of how they could perform in a given environment. Our objective was not to benchmark them, but rather to test their functionality in the DB2 for Linux on S/390 environment and observe some basic performance behavior.

A large array of data population tools is available for operating with the Linux on S/390 platform. Even though some tools perform better than others in certain circumstances, each is very capable, and any of these methods can accomplish your data population tasks. The method you choose depends on your personal preferences, the availability of the tools on your site, your familiarity with them, your performance requirements, data availability objectives, and batch window issues. The charts in Figure 67 can help in making a selection of the tools to be used for data population purposes.

7.9 Other data population tools

Although we did not test them, the following data population tools are worth considering when choosing a data population method:

- DB2 DataPropagator
- DB2 Data Warehouse Center
- DB2 AutoLoader
- DB2 for OS/390 V7.1 Unload
- DB2 DataJoiner

We provide more detail about these tools in the next few sections.

7.9.1 DB2 DataPropagator

IBM DB2 DataPropagator is a replication product for relational data. You can use it to replicate changes between any DB2 relational databases.

DB2 DataPropagator consists of three main components:

- The administration interfaces
- The change-capture mechanisms
- The apply program

You can set up these components using the DB2 Control Center to create a control table, which stores your replication criteria.

The Capture program can run, for example, on DB2 for OS/390 and capture changes to the source tables. The changes are copied into a change data table, where they are stored until the target system is ready to copy them. The Apply program can run on a target system such as DB2 for Linux on S/390, read the changes from the change data table, and copy them to the target tables. The Apply process is entirely SQL-based.

You can use the DB2 Control Center to do the following:

- Set up the replication environment
- Define source and target tables
- Specify the timing of automated copying
- Specify SQL enhancements to the data
- Define relationships between the source and the target tables

Figure 68 on page 153 summarizes the DataPropagator process.

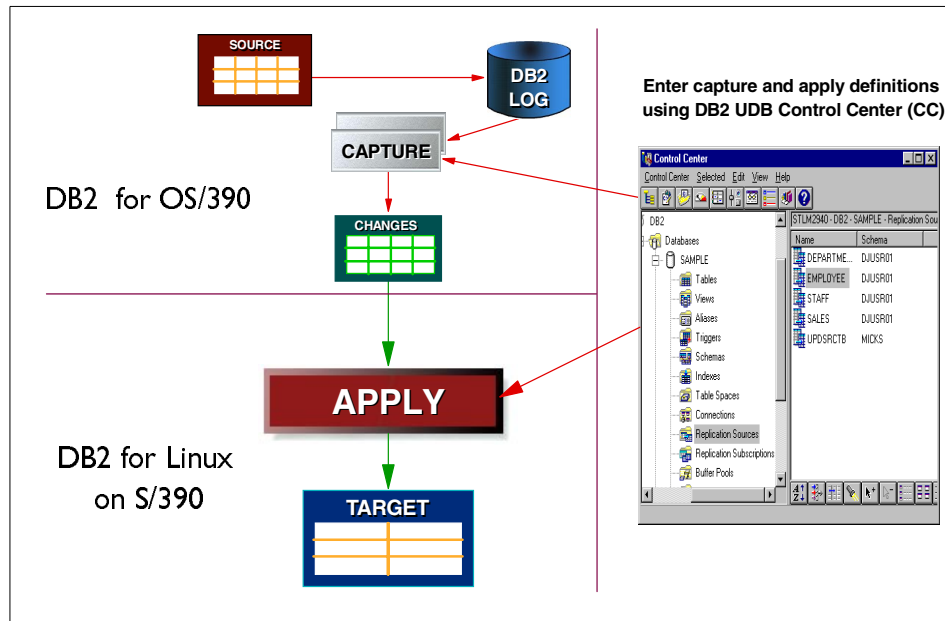


Figure 68. The DB2 DataPropagator process

7.9.2 DB2 Data Warehouse Center

Figure 69 on page 154 shows the DB2 Warehouse Manager feature for OS/390, which provides the following components:

- A restricted license for Version 7 of the DB2 Universal Database Enterprise Edition, which delivers the database warehouse management infrastructure and OLAP Starter Kit. This infrastructure includes:
 - The DB2 Data Warehouse Center (DWC), a new graphical user interface integrated with the DB2 Control Center which provides the administration for building and managing data warehouses.
 - A Warehouse manager which runs on Microsoft Windows NT and controls all the operations defined through the Data Warehouse Center.
 - A Warehouse agent for Windows NT which executes local operations on behalf of the components of the DB2 Warehouse Manager feature.
 - The OLAP Starter Kit, which is a limited user license for DB2 OLAP Server and the OLAP Integration Server for building and deploying OLAP applications.

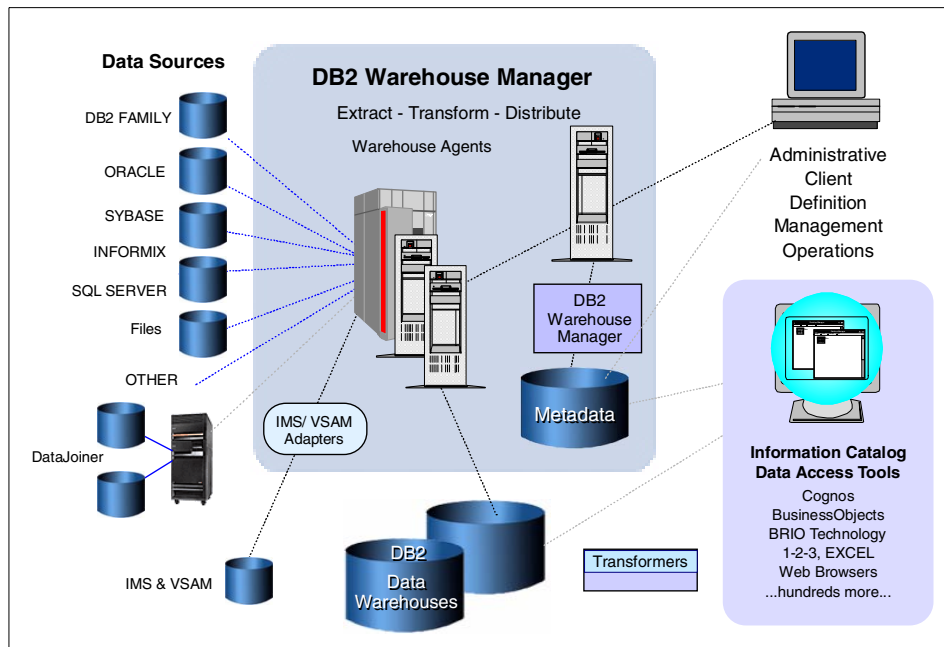


Figure 69. DB2 Warehouse Manager components

- A Warehouse agent for OS/390, which executes OS/390-based processes on behalf of the components of the DB2 Warehouse Manager feature.
- Prebuilt programs for performing a variety of OS/390-specific tasks, such as file transfer protocol (FTP), submitting job control language (JCL), and triggering server or client programs.
- Warehouse transformers for OS/390, which are stored procedures or user-defined functions. These perform complex transformations that are commonly used in warehouse development, data cleansing, key generation, and statistical calculations.
- The Enhanced Information Catalog Manager, which improves the Web interface.
- The QMF family of products including QMF for OS/390, QMF High Performance Option (HPO,) and QMF for Windows.

You can use the Data Warehouse Center (DWC) to move data from an operational database or a data warehouse to a data mart. You can also use the DWC to define the structure of the operational databases, called *sources*. You can specify how operational the data is to be moved and transformed for the warehouse. You can model the structure of the tables in the

warehouse database, called *targets*, or build the tables automatically as part of the process of defining the data movement operations.

The Data Warehouse Center uses the following DB2 functions to move and transform data:

- **SQL** to select data from sources and insert the data into targets. You can also use SQL to transform the data into its warehouse format. You can use the Data Warehouse Center to generate the SQL, or you can write your own SQL.
- **Load and Export utilities** to export data from a source, and load the data into a target. These utilities are useful if you need to move large quantities of data.
- **Replication** to copy large quantities of data from the data warehouse sources into a warehouse target, and then capture any subsequent changes to the source data.
- **Transformer stored procedures** to clean up the data and then aggregate it into fact and dimension tables. You can also use the Data Warehouse Center to move your data into an OLAP database.

7.9.3 DB2 AutoLoader

The DB2 UDB AutoLoader utility can be used in a partitioned database environment to load data across all or some of the partitions at the same time.

The AutoLoader utility does the following:

- It transfers data from a OS/390 system to a Linux S/390 system.
- It partitions that data in parallel.
- It loads the data simultaneously on the corresponding database partitions.

In a partitioned database, large amounts of data are located across many partitions. Partitioning keys are used to determine on which database partition each portion of the data resides. The data must be split before it can be loaded at the correct database partition.

7.9.4 DB2 for OS/390 V7.1 Unload

One issue to consider when transferring data from a table managed by DB2 for OS/390 to a table managed by DB2 for Linux on S/390 is *data compatibility*. We need to convert EBCDIC data used on OS/390 to ASCII format used on workstation platforms, but the REORG UNLOAD EXTERNAL

utility of DB2 for OS/390 cannot be used to do this data transfer into DB2 for Linux on S/390.

DB2 UDB for OS/390 Version 7.1 introduces a new utility that is an extension to the REORG UNLOAD EXTERNAL utility, but which allows you to generate data in an external format that can be used by DB2 for Linux on S/390. ASCII format can be specified for the output data.

In DB2 for OS/390 Version 7, you can unload data from a table space or an image copy by using the new UNLOAD utility. In most cases, the UNLOAD utility is faster than the DSNTIAUL program, especially when you activate partition parallelism. UNLOAD is easier to use than REORG UNLOAD EXTERNAL.

As shown in Figure 70, UNLOAD allows you to perform the following tasks:

- Unload data from an image copy data set
- Unload data from multiple partitions in parallel
- Select data by using a syntax similar to the SQL SELECT statement
- Sample rows by table
- Use field selection, ordering, and formatting options
- Specify SHRLEVEL CHANGE or REFERENCE

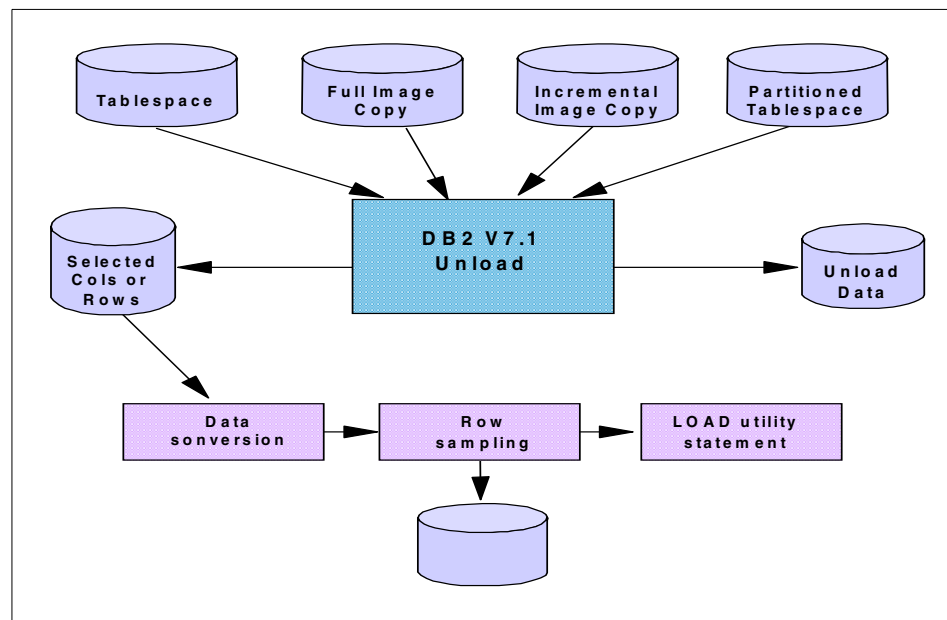


Figure 70. Unloading data using the DB2 V7.1 UNLOAD utility

7.9.5 DB2 DataJoiner

To implement a data mart on DB2 for Linux on S/390, you may need to populate it with data coming from external platforms.

DB2 DataJoiner provides transparent SQL access and join capability across heterogeneous data sources.

Using DB2 DataJoiner, you can view all your data (relational, nonrelational, local, remote, and geographic) as if it were local data. With a single SQL statement, you can access and join tables located across multiple data sources without needing to know the source location.

DB2 DataJoiner features are:

- Native support for popular relational data sources (DB2 Family, Informix, Microsoft SQL Server, Oracle, Sybase SQL Server, Teradata, and others).
- Support for nonrelational data sources (through Classic Connect V2.1.1).
- Client access (using DB2 clients) from a multitude of platforms, including Java™ (using JDBC).
- Integrated replication administration.
- DDL statements to easily create, drop, and alter data source mappings, users, user-defined and built-in functions and data types.
- Excellent performance and intelligent use of pushdown and remote query caching.
- Version 2.1.1 is available for Windows NT, Sun Solaris, and AIX. Version 1 is available for AIX and HP-UX.

Chapter 8. Scalability and performance tests for queries

For the query tests, we did not analyze all aspects of performance, but rather examined a set of primitive attributes, such as I/O, CPU, and memory. We wanted to observe the behavior of the workload on both OS/390 and Linux platforms. We expected to see a similar behavior on the Linux platform as the one we observed on the OS/390 platform.

We ran the same set of queries against DB2 for OS/390 tables and DB2 for Linux tables to compare their behavior on both platforms. We ran the queries in LPAR2 as shown in Figure 71.

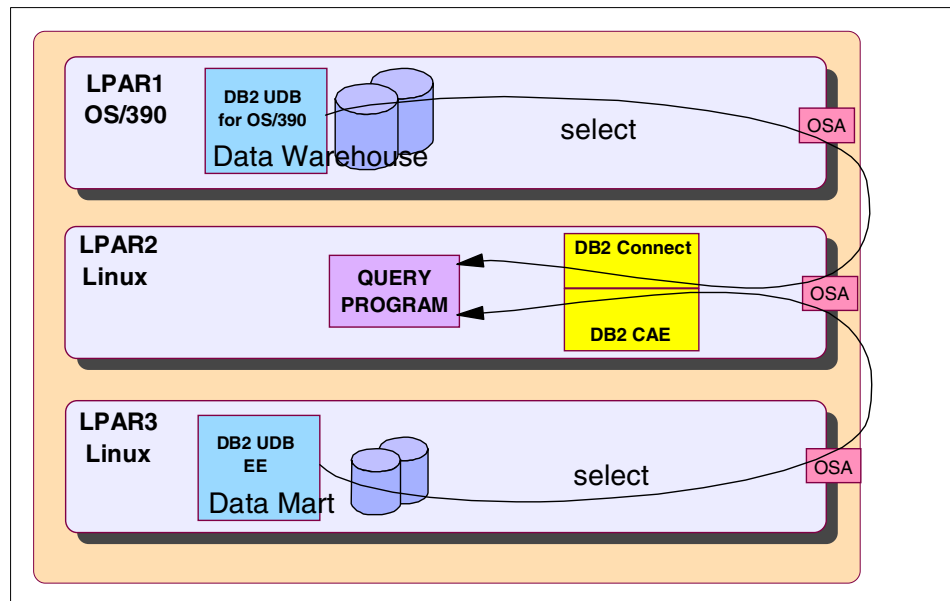


Figure 71. Query scenarios

The table definitions and scripts we used to run our queries are listed in Appendix D, “Table definitions and query scripts” on page 201.

Data model

The data model of our test scenarios consisted of three tables with no indexes defined on them. Table 7 on page 160 summarizes the characteristics of the DB2 tables we used for the query tests. Each table was

stored on a different disk and assigned to a different bufferpool. The size of each bufferpool was set to 2000 pages.

Table 7. Characteristics of tables used in the queries

Name of table	# columns	Avg. length of row	# rows
T1	16	135	6,001,215
T2	3	191	5
T3	140		1,500,000

Test cases

Except for Q02, we measured all our queries using one processor at different concurrency levels:

- To set the baseline, we ran each query once.
- We ran two instances of the same query concurrently.
- We ran three instances of the same query concurrently.
- We ran eight instances of the same query concurrently.

To achieve these measurements, we developed some simple bash scripts. We measured the elapsed time (ET) on the database server side with a resolution of 1 second. We present our measurements in normalized form; that is, the baseline measurement with concurrency level one is set to one, while the other results are shown in proportion to this baseline.

Workload

We defined our workload with the following types of queries and interests:

- Processor-intensive queries
 - Processor scalability
- I/O-intensive queries
 - I/O scalability
 - Network traffic scalability
 - Sort scalability
- Short queries
 - Memory scalability and buffer pool efficiency
- Trivial queries
 - Is what seems trivial on OS/390 also trivial on Linux?

We discuss these queries in more detail in the following sections.

8.1 Processor-intensive queries

Q02 is a processor-intensive query. It performs a table space scan on table T1, computes summary and average fields, and performs a sort. The calculations and sort involved qualify this query as processor-intensive. The objective with Q02 is to observe processor scalability.

This query is used to gain insight into how well DB2 for Linux would be able to take advantage of the zSeries and S/390 scalability attributes, as compared to DB2 for OS/390.

We ran the same tests on both systems and compared the processor scalability with 1 processor and 1,2,4,8 queries, then with 8 queries and 1,2,4,8 processors, as shown in Figure 72.

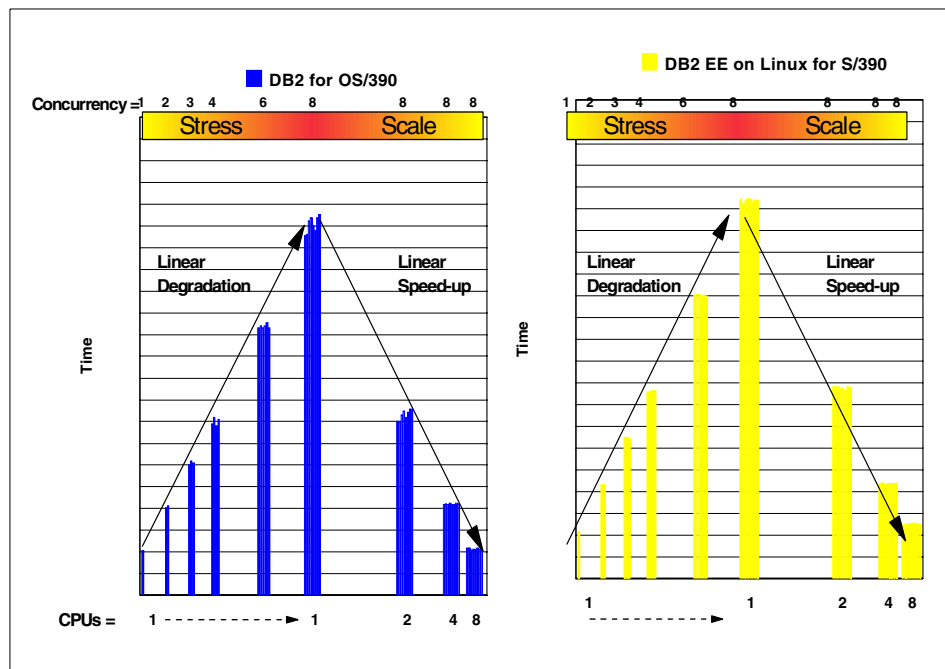


Figure 72. Scalability and stress test for CPU-intensive query (Q02)

- Test 1 - increase query concurrency, but not capacity

We measured completion time of 1 query, 1 processor. We then added queries one at a time, up to 8 concurrent queries. We took measurements at each new query concurrency level. DB2 for OS/390 and DB2 for Linux

performed equivalently, with linear degradation of response times as we increased the query concurrency.

- Test 2 - increase capacity, static workload of 8 concurrent queries

We then added processors in increments of 2, up to 8 processors, without adding concurrent queries. We took measurements at each increment. DB2 for OS/390 and DB2 for Linux performed equivalently, and query response times decreased linearly as we added processors.

OS/390 measurements for Q02 are presented in Table 8.

Table 8. Normalized ET measurements for Q02 on OS/390

#queries/#processors	Average normalized ET	Standard deviation
1/	1.00	n/a
2/1	1.96	0.02
4/1	3.88	0.23
8/1	8.18	0.20
8/2	3.96	0.18
8/4	2.02	0.01
8/8	1.03	0.02

Linux measurements for Q02 are presented in Table 9.

Table 9. Normalized ET measurements for Q02 on Linux

#queries/#processors	Normalized ET	Standard deviation
1/1	1.00	n/a
2/1	1.96	0.02
4/1	3.80	0,23
8/1	8.18	0,20
8/2	3.96	0.18
8/4	2.02	0.01
8/8	1.03	0.02

In our preliminary tests, DB2 for Linux scaled well.

The script for Q02 is provided in Appendix D.3, “Q02 Processor-intensive query” on page 202.

8.2 I/O-intensive queries

We tested three variations of I/O-intensive queries:

- I/O-intensive query - no rows returned
- I/O-intensive query - 40,000 rows returned
- I/O-intensive query - 40,000 rows returned with Sort

The objectives were to observe:

- I/O scalability
- Network traffic scalability
- Sort scalability

8.2.1 I/O-intensive query - no rows returned

Q01 was an I/O-intensive query which scans table T1 and returns no rows; see Appendix D.2, “Q01 I/O-intensive query - no rows returned” on page 202.

I/O scalability

We ran 1,2,3,8 queries, each running against a table on a different disk volume. By spreading the tables across the disk volumes, we could maintain the same elapsed time as when we ran only one query. We could effectively scale the I/O system and increase the MB/sec factor by spreading the tables on different disk volumes.

Q01 showed a minimal impact on ET of 2% and 4% when running 2 or 3 queries concurrently. Actually, we read the same table, which fits into the cache of the I/O system; thus we did not overload the I/O subsystem.

We observed the same I/O scalability on both OS/390 and Linux platforms for this I/O-intensive query.

Figure 73 shows Q01 normalized ETs at different query concurrency levels on OS/390.

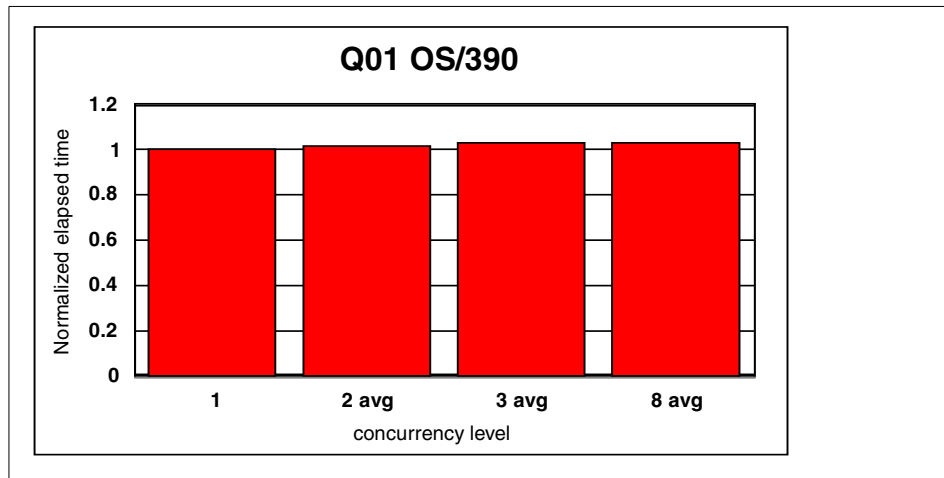


Figure 73. Q01 normalized elapsed time at different query concurrency levels

OS/390 measurements for Q01 are presented in Table 10.

Table 10. Normalized ET results for Q01 on OS/390

Concurrent runs	Average normalized ET	Standard deviation
1	1.00	0.00
2	1.02	0.00
3	1.04	0.00
8	1.04	0.00

Linux measurements for Q01 are presented in Table 11.

Table 11. Normalized ET results for Q01 on Linux

Concurrent runs	Average normalized ET	Standard deviation
1	1.00	0.00
2	1.02	0.00
3	1.04	0.00
8	1.04	0.00

8.2.2 I/O-intensive query - 40,000 rows returned

Q06 performed the same table space scan on table T1 as Q01. The difference was that Q01 does not return any row, and Q06 returns 40,000 rows. In comparing Q01 and Q06 response time, we got some insight into the delays caused by the network traffic for returning 40,000 rows.

Network traffic scalability

The normalized ETs for Q06 at different concurrency levels on OS/390 are presented in Table 12.

Table 12. Normalized ETs for Q06 at different concurrency levels on OS/390

Concurrent runs	Average normalized ET	Standard deviation
1	1.00	n/a
2	2.00	0,01
3	2.79	0,30
8	7.80	0.55

Q01 performed the same table space scan as Q06; the difference was that Q01 returns no rows and Q06 returns 40,000 rows. If we divide the ET of Q06 by the ET of Q01, we get an idea of the network transfer cost of the 40,000 rows expressed in relative ET as shown in Table 13. Compared to Q01, Q06 runs longer with an elongation of ET of 7%, due to the 40,000 rows returned in Q06.

Table 13. Relative ET for transferring 40,000 rows on OS/390

Query concurrency level 1	Result set	Relative ET
Q01	0	1.00
Q06	40,000	1,07

The behavior is the same on the Linux platform. The normalized ETs for Q06 at different concurrency levels on Linux are presented in Table 14.

Table 14. Normalized ETs for Q06 at different concurrency levels on Linux

Concurrent runs	Average normalized ET	Standard deviation
1	1.00	n/a
2	2.00	0,01
3	2.79	0,30
8	7.80	0.55

The relative ETs for transferring 40,000 rows on Linux is presented in Table 15. Q06 ran longer than Q01, with an elongation of 8% of its ET.

Table 15. Relative ETs for transferring 40,000 rows on Linux

Query concurrency level 1	Result set	Relative ET
Q01	0	1.00
Q06	40,000	1,08

See the script for Q06 in Appendix D.6, “Q06 I/O-intensive query - 40,000 rows returned” on page 203.

8.2.3 I/O-intensive query - 40,000 rows returned with Sort

Q07 scanned table T1 and returned the same result set as Q06 (that is, 40,000 rows), but this result set was sorted. This query gives some insight in the sort behavior.

Sort scalability

Table 16 shows the normalized ETs for Q07 at different concurrency levels on OS/390.

Table 16. Normalized ETs for Q07 at different concurrency levels on OS/390

Concurrent runs	Average normalized ET	Standard deviation
1	1.00	n/a
2	2.00	0,01
3	2.79	0,30
8	7.80	0.55

Q07 performed the same table space scan as Q06 and transferred the same number of returned rows (that is, 40,000), but in addition, Q07 performed a sort of the 40,000 rows. If we divide the ET of Q07 by the ET of Q06, we get an idea of the relative ET spent for the sort. The relative ET for sorting 40,000 rows on OS/390 is presented in Table 17. Q07 has an elongation of 5% of its ET, which represents the cost of Sort.

Table 17. Relative ETs for sorting 40,000 rows on OS/390

Query concurrency level 1	Result set	Relative ET
Q06	0	1.00
Q07	40,000	1,05

The normalized ETs for Q07 at different concurrency levels on Linux are presented in Table 18.

Table 18. Normalized ETs for Q07 at different concurrency levels on Linux

Concurrent runs	Average normalized ET	Standard deviation
1	1.00	n/a
2	2.00	0,01
3	2.79	0,30
8	7.80	0.55

The relative ETs for sorting 40,000 rows on Linux is presented in Table 19.

Table 19. Relative ETs for sorting 40,000 rows on Linux

Query concurrency level 1	Sorted result set	Relative ET
Q06	0	1.00
Q07	40,000	1,01

The relative ET for Sort in the Linux environment is negligible (1%). The reason for this is because DB2 for Linux systematically sorts the rows whether you ask for it or not! So the comparison, with and without explicitly requested Sort, shows no difference! We double-checked the relative ET by

doubling the returned rows (80,000) and noticed no ET elongation when we explicitly asked for a sort.

The script for this query is provided in Appendix D.7, “Q07 I/O-intensive query - 40,000 rows returned with Sort” on page 203.

8.3 Short queries

We observed short queries that read the table data from disk, and short queries that read the table data from buffer pools in memory. The objective was to observe memory scalability and DB2 buffer pool efficiency.

Memory scalability and buffer pool efficiency

Q04 and Q05 were short queries designed to demonstrate the memory usage through bufferpool allocations. The table T3 they scan can be read from disk or it can be read from memory if there are enough buffer pools allocated to that table. We increased the size of the bufferpool in order to be able to load the entire table into the bufferpool. The query was run twice, as Q04 and Q05, and we measured the ET:

- The ET of Q04 includes the disk reads.
- The ET of Q05 benefits from the fact that the table is already in memory.

The first time we ran Q04, the table is read from the disk and loaded into the associated buffer pool. The second time we ran Q04 (or Q05), the table is already in the buffer pool. The behavior on both platforms is the same.

Table 20 shows the ET comparison between Q04 and Q05 on OS/390.

Table 20. Normalized ETs for Q04 and Q05 OS/390

Query concurrent runs	Normalized ET	Standard deviation
Q04	1.00	n/a
Q05	0,14	n/a

Q05 normalized ETs at different concurrency levels on OS/390 are presented in Table 21.

Table 21. Normalized ETs for Q05 at different concurrency levels on OS/390

Concurrent runs	Average normalized ET	Standard deviation
1	1.00	n/a

Concurrent runs	Average normalized ET	Standard deviation
2	1.50	0,00
3	2.50	0,00
8	5,75	0.50

Table 22 shows the ET comparison between Q04 and Q05 on Linux. The behavior is the same as on OS/390.

Table 22. Normalized ET for Q04 and Q05 Linux

Query	Normalized ET	Standard deviation
Q04	1.00	n/a
Q05	0,14	n/a

Q05 normalized ETs at different concurrency levels on Linux are presented in Table 23. The behavior is the same as on OS/390.

Table 23. Normalized ETs for Q05 at different concurrency levels on Linux

Concurrent runs	Average normalized ET	Standard deviation
1	1.00	n/a
2	1.50	0,01
3	5.48	0,30
8	9.64	0.55

The scripts for Q04 and Q05 are provided in Appendix D.5, “Q04 Short query and Q05 Short query with table in memory” on page 203.

8.4 Trivial queries

Query Q03 performed a table space scan on table T3. The table had only 5 rows. This was a trivial query with an elapsed time always at 0. The measurement is not significant, which is normal for a trivial query. We observed the same behavior on both OS/390 and Linux platforms.

The script for this trivial query is provided in Appendix D.4, “Q03 Trivial query” on page 203.

Chapter 9. Administrating and monitoring the BI environment

This chapter lists the DB2 UDB administration tools available to administer and monitor the DB2 UDB environment on Linux S/390. Different administration topics are presented:

- DB2 administration tools
- DB2 performance tools
- Linux performance tool
- DB2 backup/restore tools

9.1 DB2 administration

Many of the tasks described in this redbook can be performed using different interfaces:

- The Command Line Processor, which allows you to execute SQL statements and DB2 utility functions. For more information about using the the DB2 command processor, see *IBM DB2 Command Reference Version 7*, SC09-2951.
- The Application Programming Interface (API), which allows you to execute DB2 utility functions within an application program. For more information about using the application programming interface, see *IBM DB2 Administrative API Reference Version 7*, SC09-2947.
- The Control Center, which allows you to graphically perform administrative tasks such as configuring the system, managing directories, backing up and recovering the system, scheduling jobs, and managing media. The Control Center also contains Replication Administration, to graphically set up the replication of data between systems. Further, the Control Center allows you to execute DB2 utility functions through a graphical user interface.

There are different methods to invoke the Control Center, depending on your platform. For example, use the `db2cc` command on a command line, (on OS/2) select the Control Center icon from the DB2 folder, or use start panels on Windows platforms.

There are more administration tools you can use from the Control Center toolbar:

- **Satellite Administration Center**
This allows you to administer DB2 Satellite Servers.
- **Command Center**
This tool enables you to issue DB2 commands, SQL statements and operating system commands; recall previous commands; and scroll through access plans for queries.
- **Script Center**
This allows you to create, run and schedule scripts containing operating-system-level commands and DB2 commands.
- **Alert Center**
The Alert Center notifies you when thresholds that you have set have been exceeded.
- **Journal**
This allows you to view the status of jobs and view the recovery history log and messages log.
- **Information Center**
This provides quick access to the information in the DB2 product manuals and sample programs.
- **License Center**
This displays the status of your license and allows you to configure your system for proper license monitoring.

Reorgs, Runstats and other scripts can also be scheduled by using cron on Linux. You can generate DDL by using the db2look program or Control Center. See *IBM DB2 Administration Guide : Implementation Version 7*, SC09-2944, for administering DB2 UDB with Control Center.

The tools for administering DB2 are part of the Administration Client, a selectable component with each of the DB2 Universal Database products. The Administration Client is also available on a set of CD-ROMs that include the Administration Clients for all the operating systems on which DB2 is available. They allow you to install and use the Administration Client on any workstation: it does not matter whether your database servers are local or remote, or what operating system the database servers are running on.

The tools enable you to perform the same functions from a Graphical User Interface as you could from the Command Line Processor. These functions include the entering of DB2 commands, SQL statements, or system commands. With the tools you do not have to remember complex statements or commands, and you get additional assistance.

Note

The Administration Client is an installation option.

9.2 DB2 performance tools

In general, you should undertake performance tuning when you want to improve the cost-benefit ratio of your system. Specific goals could include:

- Processing a larger, or more demanding, workload without increasing processing costs (for example, increasing the workload without buying new hardware or using more processor time)
- Obtaining faster system response times, or higher throughput, without increasing processing costs
- Reducing processing costs without negatively affecting service to your users

DB2 UDB provides several tools to capture and analyze performance data.

9.2.1 SQL Explain

The SQL explain facility is part of the SQL Compiler (optimizer) that can be used to capture information about the environment where the static or dynamic SQL statement is compiled. The information captured allows you to understand the structure and potential execution performance of SQL statements, including:

- Sequence of operations to process the query
- Cost information
- Predicates and selectivity estimates
- Statistics for all objects referenced in the SQL statement at the time of the explain

Detailed optimizer information that allows for in-depth analysis of an access plan is kept in explain tables separate from the actual access plan.

To create these plan tables, use the DDL statements in *\$HOME/sqllib/misc/EXPLAIN.DDL*.

There are three ways to get information from the explain tables:

- Write your own queries.
- Use the *db2exfmt* tool in the *\$HOME/sqllib/bin* directory.
- Use Visual Explain (to view explain snapshot information). The Visual Explain tool is invoked from the Control Center.

Other explain tools include:

- *db2expln*

This provides a relatively compact overview of what operations will occur at run-time by examining the actual access plan generated.

- *dynexpln*

This provides a quick way to explain dynamic SQL statements that contain no parameter markers.

To populate the explain tables, use the following:

- The EXPLAIN statement captures information about the access plan chosen for the supplied explainable statement and places this information into the explain tables.
- The EXPLAIN bind option stores information in the explain tables about the access plans chosen for each SQL statement in the package.
- The EXPLSNAP bind option stores explain snapshot information in the explain tables.

For detailed information about the SQL Explain Facility and interpreting the output, refer to *IBM DB2 Administration Guide : Performance Version 7*, SC09-2945.

9.2.2 Snapshot monitoring

Taking a snapshot gives you information for a specific point in time. Use the *get snapshot for* command to display current statistics for certain events according to the *get snapshot* command. For example, you can obtain a list of locks held by applications connected to a database by taking a database lock snapshot. This can be used to resolve lock-wait conditions. First, turn on the LOCK switch (*update monitor switches*), so that the time spent waiting for locks is collected.

Refer to *IBM DB2 Command Reference Version 7*, SC09-2951 and *IBM DB2 System Monitor Guide and Reference Version 7*, SC09-2956 for command options and for information about how to interpret the output.

Note

To obtain a snapshot of a remote DB2 instance, you must first attach to the DB2 instance.

9.2.3 Event monitors

You can request the database manager to automatically log monitor data to files when specific events occur. This allows you to collect information about transient events that are difficult to monitor through snapshots, such as deadlocks and transaction completions. Collecting system monitor data with an event monitor is a three-step process:

1. Create the event monitor.

An event monitor is created by using the `create event monitor` SQL statement and specifying the events to monitor. You can also use `db2emcrt` on Windows to create and activate event monitors for local and remote DB2 systems.

2. Activate the event monitor.

An event monitor is activated by using the `set event monitor state` SQL statement. To activate an event monitor as soon as the database is started, create the event monitor with the `AUTOSTART` clause.

3. Read the trace produced.

Reading an event monitor trace file can be done using `db2evmon`. You can also use `db2eva`, a GUI tool, on OS/2 and Windows.

Note

The event monitor trace files *must* be available on the machine where you invoked `db2eva`.

Use the `FLUSH EVENT MONITOR` statement to force monitor values to the event monitor output writer.

Refer to *IBM DB2 System Monitor Guide and Reference Version 7*, SC09-2956, for information about interpreting the output from the event monitors.

9.2.4 DB2 Performance Monitor

The Performance Monitor tool is invoked from the Control Center by right-clicking on the object and activating monitoring for the object. The Performance Monitor tool can be customized to create warnings and alarms, monitoring certain thresholds you have defined.

The Performance Monitor enables you to do the following:

- Detect performance problems
- Prevent problems from occurring
- Analyze trends
- Analyze the performance of applications
- Tune databases

See *IBM DB2 Administration Guide : Implementation Version 7*, SC09-2944, for detailed information on using the DB2 Performance Monitor from Control Center.

9.2.5 DB2 Governor

You use the governor to monitor and change the behavior of applications that run against a database.

For example, a rule may indicate that an application is using too much resource. In this situation, the governor may change the application's priority or force it off the database, according to the instructions you specified in the governor configuration file.

You use the `db2gov` governor front-end utility to start or stop the governor. See *IBM DB2 Administration Guide : Performance Version 7*, SC09-2945, for information about configuring and running the governor.

9.2.6 Wizards

There are wizards integrated with DB2 that will assist you in completing some performance-related administration tasks. The wizards take you through each task, one step at a time. Wizards are available through the Control Center and the Client Configuration Assistant.

The Performance Configuration wizard assists you in tuning the performance of a database by updating configuration parameters to match your business requirements. Other wizards are available to assist in the improvement of performance of individual tables and general data access, including the Create Table, Index, and Configure Multisite Update wizards. You can find the

wizards at the Control Center by clicking on an object with the *right mouse button*.

9.3 Linux performance tools

Different performance and monitoring tools are available on Linux S/390. This section gives an overview of vmstat and mpstat.

9.3.1 vmstat

The vmstat tool reports virtual memory statistics.

The command `vmstat` reports information about processes, memory, paging, block IO, traps, and CPU activity.

The syntax is the following:

```
vmstat [-V] [-n] [delay [count]]
-V prints the version
-n causes the headers not to be reprinted regularly
delay is delay between the updates in seconds
count is the number of updates
```

For example, the following command reports the information to you every two seconds and stops after ten reports:

```
vmstat 2 10
```

The report of `vmstat` has the following format:

procs			memory				swap		io		system		cpu			
r	b	w	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	
0	0	0	396	5280	305720	117828	0	0	11	1	0	19	0	2	98	
0	0	0	396	5280	305720	117828	0	0	0	0	0	8	0	2	98	

The report has the following format:

- procs
 - r: The number of processes waiting for run time
 - b: The number of processes in uninterruptable sleep
 - w: The number of processes swapped out but otherwise runnable
- memory
 - swpd: The amount of virtual memory used (kB)
 - free: The amount of idle memory (kB)
 - buff: The amount of memory used as buffers (kB)
 - cache: The amount of cache (kB)

- swap
 - si: Amount of memory swapped in from disk (kB/s)
 - so: Amount of memory swapped to disk (kB/s)
- io
 - bi: Blocks write to disk (block/s)
 - bo: Blocks read from disk (block/s)
- system
 - in: The number of interrupts per second, including the clock
 - cs: The number of context switches per second
- cpu
 - us: The time of cpu the user need
 - sy: The time of cpu the system need
 - id: Idle time of the cpu

9.3.2 mpstat

The mpstat tool reports processors-related statistics

The `mpstat` commands shows you the activity for each processor available in your system.

The mpstat command syntax is the following:

```
mpstat [ -P { cpu | ALL } ] [ -V ] [ interval [ count ] ]
```

- V prints the version.
- P
 - cpu is the number of the cpu. Note that number 0 is the first processor. With the command `cat /proc/cpuinfo` you can see the numbers of your processors.
 - ALL indicates that statistics are reported for all processors. It is the default.
- interval is the delay between each statistic.
- count is the number of statistics.

As an example, the following command shows you the activity of the second processor with an interval of 5 seconds:

```
mpstat -P 1 5
```


The report of mpstat has the following format:

	CPU	%user	%nice	%system	%idle	intr/s
12:39:05						
12:39:10	1	0.10	0.00	1.00	48.90	0.00
12:39:15	1	1.80	0.00	9.30	38.90	0.00

The report has the following format

- CPU

This shows the processor number. The keyword all indicates that statistics are calculated as averages among all processors.

- %user

This shows the percentage of CPU utilization that occurred while executing at the user level (application).

- %nice

This shows the percentage of CPU utilization that occurred while executing at the user level with nice priority.

- %system

This shows the percentage of CPU utilization that occurred while executing at the system level (kernel).

- %idle

This shows the percentage of time that the CPU or CPUs were idle.

- intr/s

This shows the total number of interrupts received per second by the CPU (or CPUs).

9.4 DB2 security administration

Database security is one of the most important responsibilities of the database administrator. Before implementing a database in a production environment, you should define your objectives for database accesses, and specify who shall have access to what and under what circumstances.

To protect data and resources associated with a database server, DB2 uses a combination of external security services and internal access control information. You must pass some security checks before you are given access to the database server, database data or resources.

The first step in database security is called *authentication*, where you must prove that you are who you say you are.

The second step is called *authorization*, where the database manager decides if you are allowed to perform the requested action, or access the requested data.

Authentication of a user is completed using a security facility outside of DB2. The security facility can be part of the operating system, a separate product or, in certain cases, may not exist at all.

On UNIX-based systems, the security facility is in the operating system itself. DCE Security Services is a separate product that provides the security facility for a distributed environment. The security facility requires two items to authenticate a user: a user ID and a password.

The user ID identifies the user to the security facility. By supplying the correct password (information known only to the user and the security facility), the user's identity (corresponding to the user ID) is verified.

Authorization is the process whereby DB2 obtains information about an authenticated DB2 user, indicating the database operations that user may perform, and what data objects that user may access. With each user request, there may be more than one authorization check, depending on the objects and operations involved.

Authorization is performed using DB2 facilities. DB2 tables and configuration files are used to record the permissions associated with each authorization name. The authorization name of an authenticated user, and those of groups to which the user belongs, are compared with the recorded permissions. Based on this comparison, DB2 decides whether to allow the requested access.

There are two types of permissions recorded by DB2: privileges and authority levels.

For more detailed information about DB2 security options, refer to *DB2 Administration Guide: Implementation*.

Appendix A. Teraplex LPAR resource matrix

LPAR	LP1	LP2	LP3	LP4
Hostname	tplexpd4	tplexpd7	tplexpd8	tplexpd6
OS	OS/390	Linux	Linux	VIF
OS Level	V2R9	SuSE 7.0	SuSE 7.0	VIF GA+
Load Addr	7194	2018	5014	201A
DB2	R6	Connect Beta2	UDB EE Beta2	
DB2 Port	477	50000	50000	
DB2 Location	TPLEXDBLX			
DB2 Instance		db2inst1	db2inst1	
OSA-2 Addr	900	604	606	904
OSA-2 CHPID	D8	84	84	D8
OSA-2 port	1	1	1	0
OSA-2 IP Addr	9.12.17.25	9.12.18.46	9.12.18.47	9.12.18.35
OSA-2 DBLX Alias		DBLX	DBLX	
CTC Addr(LP1-LP2)	350	300		
CTC IP Addr	15.1.1.4	15.1.1.7		
CTC Addr(LP1-LP3)	360		300	
CTC IP Addr	12.1.1.4		12.1.1.8	
OSA-X Addr	E00-3	E04-7	E08-B	
OSA-X IP Addr	14.1.1.4	14.1.1.7	14.1.1.8	
OSA-X DBLX Alias		DBLXOX	DBLXOX	

LPAR	LP4	LP4	LP4	LP4
Hostname	linux0	tplexpda	tplexpdb	tplexpdc
OS	Linux (VIF)	Linux (VIF)	Linux (VIF)	Linux (VIF)
OS Level	SuSE 7.0	SuSE 7.0	SuSE 7.0	SuSE 7.0
Load Addr	201 (Virtual)	201 (Virtual)	201 (Virtual)	201 (Virtual)
DB2		UDB EE Beta2	UDB EE Beta2	UDB EE Beta2
DB2 Port		50000	50000	50000
DB2 Location				
DB2 Instance		db2inst1	db2inst1	db2inst1
OSA-2 Addr	906	908	90C	90E
OSA-2 CHPID	D8	D8	D8	D8
OSA-2 port	0	0	0	0
OSA-2 IP Addr	9.12.18.48	9.12.18.49	9.12.18.58	9.12.18.59
OSA-2 DBLX Alias		DBLX	DBLX	DBLX
OSA-X Addr		E0C-F	E10-3	E14-7
OSA-X IP Addr		14.1.1.10	14.1.1.11	14.1.1.12
OSA-X DBLX Alias		DBLXOX	DBLXOX	DBLXOX

Figure 74. Teraplex LPAR resource matrix

Appendix B. DB2 for Linux scripts to create the data mart

DB2 for Linux scripts to create the data mart include the following sections:

- Creating the database
- Creating the table spaces
- Creating the tables
- Creating the DB2 buffer pools

B.1 Creating the database

We used the following script to create the DB2 for Linux database:

```
create db linux8 on /mnt/dasdb/database;
```

B.2 Creating the table spaces

We used the following script to create table spaces TST1, TST2 and TST3.

```
create tablespace TST1
managed by database
using (file '/mnt/dasdd/dmsts/TST1' 256000)
extentsize 32
prefetchsize 64
bufferpool BP1;
```

```
create tablespace TST2
managed by database
using (file '/mnt/dasde/dmsts/TST2' 256000)
extentsize 32
prefetchsize 64
bufferpool BP2;
```

```
create tablespace TST3
managed by database
using (file '/mnt/dasdh/dmsts/TST3' 256000)
extentsize 32
prefetchsize 64
bufferpool BP5;
```

B.3 Creating the tables

We used the following scripts to create DB2 tables T1, T2 and T3.

```
--CREATE TABLE T1 (
--T1_C01 INT NOT NULL,
```

```

--T1_C02 INT NOT NULL,
--T1_C03 INT NOT NULL,
--T1_C04 INT NOT NULL,
--T1_C05 REAL NOT NULL,
--T1_C06 REAL NOT NULL,
--T1_C07 REAL NOT NULL,
--T1_C08 REAL NOT NULL,
--T1_C09 CHAR(1) NOT NULL,
--T1_C10 CHAR(1) NOT NULL,
--T1_C11 DATE NOT NULL,
--T1_C12 DATE NOT NULL,
--T1_C13 DATE NOT NULL,
--T1_C14 CHAR(25) NOT NULL,
--T1_C15 CHAR(10) NOT NULL,
--T1_C16 VARCHAR(44) NOT NULL)
-- IN TST1;

--CREATE TABLE T2 (
--T2_C01 INT NOT NULL,
--T2_C02 INT NOT NULL,
--T2_C03 CHAR(1) NOT NULL,
--T2_C04 REAL NOT NULL,
--T2_C05 DATE NOT NULL,
--T2_C06 CHAR(15) NOT NULL,
--T2_C07 CHAR(15) NOT NULL,
--T2_C08 INT NOT NULL,
--T2_C09 VARCHAR(79) NOT NULL)
-- IN TST2;

--CREATE TABLE T3 (
--T3_C01 INTEGER NOT NULL,
--T3_C02 CHAR(25) NOT NULL,
--T3_C03 INTEGER NOT NULL,
--T3_C04 VARCHAR(152) NOT NULL)
-- IN TST3;

```

B.4 Creating the DB2 buffer pools

We used the following scripts to create the DB2 bufferpools.

```

create bufferpool BP1 size 2000;
create bufferpool BP2 size 2000;
create bufferpool BP3 size 2000;
create bufferpool BP4 size 2000;
create bufferpool BP5 size 2000;
create bufferpool BP6 size 2000;
# Disconnect from target

```

Appendix C. Data population scripts

Data population scripts include the following sections:

- Distributed SQL statement
- C program source code with Select/Insert SQL statements
- DB2 DSNTIAUL and DB2 Load utilities
- DB2 High Performance Unload utility
- DB2 Export/Import scripts
- DB2 Export/Load scripts

C.1 Distributed SQL statement

In the following example, assume:

- The DB2 for OS/390 Distributed Data Facility has been configured, and uses port number 446 to listen for incoming requests.
- The DB2 for OS/390 location name, as specified in the bootstrap data set, is TESTDBLX.
- The IP address of the OS/390 system to be contacted is 9.12.17.25.

The following statements define the DB2 for OS/390 subsystem to a DB2 instance on UDB for Linux:

```
db2 catalog tcpip node db2on390 remote 9.12.17.225 server 123
```

```
db2 catalog dcs db db2on390 as TESTDBLX
```

```
db2 catalog database db2on390 as db2on390 at node db2on390 authentication DCS
```

Then, to connect to the DB2 for OS/390 subsystem, the following command can be used:

```
db2 connect to db2on390 user userid using password
```

Where userid and password are the userid and password defined to the security software on the OS/390 system for user authentication.

The following scenario is a simple example of how you can populate the DB2 UDB EE data mart on Linux with data from the DB2 for OS/390 data source, using SELECT/INSERT.

In this example:

- linux9a is the local alias name for the DB2 data mart on Linux.
- lp1390 is the arbitrary name assigned to identify the DB2 for OS/390.
- S390USER is the authorized userid on DB2 for OS/390.

- DB2INST1 is the authorized userid on DB2 for Linux.
- marttbl1 is the table on DB2 for Linux data mart.

EXAMPLE:

```
db2 => connect to linux9a
```

Database Connection Information

Database server = DB2/LINUX 7.1.0
 SQL authorization ID = DB2INST1
 Local database alias = LINUX9A

```
db2 => catalog tcpip node db2on390 remote 9.12.17.225 server 123
```

DB20000I The CATALOG TCP/IP NODE command completed successfully.
 DB21056W Directory changes may not be effective until the directory cache is refreshed.

```
db2 => create wrapper drda
```

DB20000I The SQL command completed successfully.

```
db2 => CREATE SERVER lp1390 TYPE DB2/390 VERSION 6.1 WRAPPER DRDA
AUTHORIZATION S390USER PASSWORD XXXXX OPTIONS ( NODE 'db2on390', DBNAME
'TESTDBPX' )
```

DB20000I The SQL command completed successfully.

```
db2 => CREATE USER MAPPING FOR db2inst1 SERVER lp1390 OPTIONS (
REMOTE_AUTHID 'S390USER', REMOTE_PASSWORD 'XXXXX' )
```

DB20000I The SQL command completed successfully.

```
db2 => CREATE NICKNAME line390 FOR lp1390.linux.T1
```

DB20000I The SQL command completed successfully.

```
db2 => insert into marttbl1 select * from line390 where T1_C01 < 7895169
```

DB20000I The SQL command completed successfully.

C.2 C program source code with Select/Insert SQL statements

This section includes:

- C program source code
- Makefile to compile the C program
- Script to run the C program

C.2.1 C program source code

A simple C program can be used to get data from the Data Warehouse and insert it into the Data Mart. We wrote a C program to select from the source database and insert into the target database.

The C program does the following:

- Connect to the DB2 source database.
- Connect to the DB2 target database.
- Open a cursor.
- Select the columns in the table using an orderkey statement. The selection reads the right amount of data needed to perform the scalability tests.
- Set connection to the target database.
- Insert the columns into the target database.
- Set connection back to the source database.
- Go to the next row.
- Commit the changes in the target database and finally, release the connections.

The program runs with a two-phase commit connection.

```

/*****
**      selins.sqc
**
**      PURPOSE :
**          This is a small C program to select from source and
**          insert into target. One way of getting data from DW to DM.
**          insert into target. It is not a generic C program to handle any source and
**          target table.
**
**      STRUCTURES USED :
**          sqlca
**
**      APIs USED :
**
**      FUNCTIONS DECLARED :
**          'C' COMPILER LIBRARY :
*****/
```

```

**      stdio.h - printf
**      string.h - strncpy
**      string.h - strncat
**
**      OTHER :
**      external : [in the file UTIL.C]
**      check_error :      Checks for SQLCODE error, and prints out any
**                          related information available.
**
**      EXTERNAL DEPENDENCIES :
**      - Ensure existence of database for precompile purposes.
**      - Precompile with the SQL precompiler (PREP in DB2)
**      - Bind to a database (BIND in DB2)
**      - Compile and link with the IBM Cset++ compiler (AIX and OS/2)
**        or the Microsoft Visual C++ compiler (Windows)
**        or the compiler supported on your platform.
**
** For more information about these samples see the README file.
**
** For more information on programming in C, see the:
** - "Programming in C and C++" section of the Application Development Guide
** For more information on building C applications, see the:
** - "Building C Applications" section of the Application Building Guide.
**
** For more information on the SQL language see the SQL Reference.
**
*****/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sqlenv.h>
#include <sqlcodes.h>
#include <sqllda.h>
#include "utilemb.h"
#ifdef DB268K
    /* Need to include ASLM for 68K applications */
    #include <LibraryManager.h>
#endif
EXEC SQL INCLUDE SQLCA ;

EXEC SQL BEGIN DECLARE SECTION ;
    char dbcon1[9] ;
    char dbcon2[9] ;
    char userid[9] ;
    char passwd[19] ;
    char userid2[9] ;
    char passwd2[19] ;
    char altstr[255];
    long worder;
    long wT1_C01;
    long wT1_C02;
    long wT1_C03;
    long wT1_C04;
    double wT1_C05;
    double wT1_C06;
    double wT1_C07;
    double wT1_C08;
    char wT1_C09;
    char wT1_C10;
    char wT1_C11[11];
    char wT1_C12[11];
    char wT1_C13[11];
    char wT1_C14[26];

```

```

        char wTl_C15[11];
        char wTl_C16[45];
EXEC SQL END DECLARE SECTION ;

/* 'check_error' is a function found in the util.c program */
/* #define CHECKERR( CE_STRING ) if ( check_error( CE_STRING, &sqlca ) != 0 ) return -1 ;
*/
#define SQLSTATE sqlca.sqlstate
int process_cursor() ;

int main( int argc, char *argv[] ) {

    char constr[] = "Connect to";
    char tempstr[256];
    int rc;
#ifdef DB268K
    /*
    Before making any API calls for 68K environment,
    need to initial the Library Manager
    */
    InitLibraryManager(0,kCurrentZone,kNormalMemory) ;
    atexit(CleanupLibraryManager) ;
#endif
    printf( "Sample C program selins : Select from source and insert into target\n" ) ;

    /* Initialize the connection to a database. */
    if ( argc == 8 ) {
        strcpy( dbcon1, argv[1] ) ;
        strcpy( dbcon2, argv[2] ) ;
        strcpy( userid, argv[3] ) ;
        strcpy( passwd, argv[4] ) ;
        strcpy( userid2, argv[5] ) ;
        strcpy( passwd2, argv[6] ) ;
        worder = atoi( argv[7] );

        printf( "Before target\n" ) ;

        strcpy( tempstr, constr ) ;
        strcat( tempstr, dbcon2 ) ;
        strcat( tempstr, "\0" ) ;

        EXEC SQL CONNECT TO :dbcon2 USER :userid2 USING :passwd2 ;
        EMB_SQL_CHECK( tempstr ) ;

        printf( "Connected to database %s\n", dbcon2 ) ;

        strcpy( altstr, "ALTER TABLE lineitem2 ACTIVATE NOT LOGGED INITIALLY" );
        EXEC SQL EXECUTE IMMEDIATE :altstr;
        EMB_SQL_CHECK( "ALTER TABLE" ) ;

        printf( "Before source\n" ) ;

        strcpy( tempstr, constr ) ;
        strcat( tempstr, dbcon1 ) ;
        strcat( tempstr, "\0" ) ;

        EXEC SQL CONNECT TO :dbcon1 USER :userid USING :passwd ;
        EMB_SQL_CHECK( tempstr ) ;

        printf( "Connected to database %s\n", dbcon1 ) ;

    }
    else {

```

```

        printf( "\nUSAGE: selins [sourcedb targetdb useridsrc passwdsrc useridtrg passwdtrg
orderkey]\n\n" );
        return( 1 ) ;
    } /* endif */

    rc = process_cursor();

    EXEC SQL RELEASE ALL ;
    EMB_SQL_CHECK( "RELEASE CONNECTIONS" ) ;

    return( 0 ) ;

} /*main */

/*****
 * FUNCTION : process_cursor
 * This function opens the source table cursor and inserts
 * the rows into the target table.
 *****/
int process_cursor() {

long int counter = 0;

/*****
/* Declare cursors for returning result sets.
*****/
EXEC SQL
    DECLARE linecursor CURSOR FOR
    Select
        T1_C01 ,
        T1_C02 ,
        T1_C03 ,
        T1_C04 ,
        T1_C05 ,
        T1_C06 ,
        T1_C07 ,
        T1_C08 Real ,
        T1_C09 ,
        T1_C10 ,
        T1_C11 ,
        T1_C12 ,
        T1_C13 ,
        T1_C14 ,
        T1_C15 ,
        T1_C16
    From T1
    Where T1_C01 < :worder Optimize for 10000 rows;

    EXEC SQL OPEN linecursor ;
    EMB_SQL_CHECK( "OPEN ORDER" ) ;
    EXEC SQL FETCH linecursor INTO
        :wT1_C01,
        :wT1_C02,
        :wT1_C03,
        :wT1_C04,
        :wT1_C05,
        :wT1_C06,
        :wT1_C07,
        :wT1_C08,
        :wT1_C09,
        :wT1_C10,
        :wT1_C11,
        :wT1_C12,

```

```

:wTl_C13,
:wTl_C14,
:wTl_C15,
:wTl_C16;

EMB_SQL_CHECK( "FETCH ORDER" ) ;

/* fetch and insert rows from cursor */

while ( SQLCODE == 0 ) {
    counter++ ;

    EXEC SQL SET CONNECTION :dbcon2;
    EMB_SQL_CHECK( "SET CONNECTION TARGET" );

    /* do insert */
    EXEC SQL INSERT INTO lineitem2 VALUES
        (:wTl_C01,
         :wTl_C02,
         :wTl_C03,
         :wTl_C04,
         :wTl_C05,
         :wTl_C06,
         :wTl_C07,
         :wTl_C08,
         :wTl_C09,
         :wTl_C10,
         :wTl_C11,
         :wTl_C12,
         :wTl_C13,
         :wTl_C14,
         :wTl_C15,
         :wTl_C16);
    EMB_SQL_CHECK( "INSERT ORDER" ) ;

    EXEC SQL SET CONNECTION :dbcon1;
    EMB_SQL_CHECK( "SET CONNECTION SOURCE" );

    EXEC SQL FETCH linecursor INTO
        :wTl_C01,
        :wTl_C02,
        :wTl_C03,
        :wTl_C04,
        :wTl_C05,
        :wTl_C06,
        :wTl_C07,
        :wTl_C08,
        :wTl_C09,
        :wTl_C10,
        :wTl_C11,
        :wTl_C12,
        :wTl_C13,
        :wTl_C14,
        :wTl_C15,
        :wTl_C16;

    EMB_SQL_CHECK( "FETCH ORDER" ) ;

} /* endwhile */

EXEC SQL CLOSE linecursor ;
EMB_SQL_CHECK( "CLOSE ORDER" ) ;

```

```

EXEC SQL COMMIT ;
EMB_SQL_CHECK( "COMMIT" ) ;

printf( "\n %d record(s) selected and inserted\n\n", counter ) ;

return (0);
} /* process_cursor */
/* end of program : SELINS.SQC */

```

C.2.2 Makefile used to compile the C program

This is the makefile used to compile the C program:

```

# (C) COPYRIGHT International Business Machines Corp. 1998, 2000
# All Rights Reserved.
# US Government Users Restricted Rights - Use, duplication or
# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
# Makefile for DB2 Universal Database
# C sample programs -- LINUX operating system
# Enter one of the following commands
# make <app_name>          - Builds the program designated by <app_name>
# make clean               - Erases intermediate files
# make cleanall            - Erases all files produced in the build process,
#                           except the original source files
# The makefile contains the following sections:
#   1 -- COMPILERS + VARIABLES
#   2 -- MAKE CATEGORIES
#   3 -- COMMANDS TO MAKE INDIVIDUAL SAMPLES
#####
#               1 -- COMPILERS + VARIABLES
#####
# This file assumes the DB2 instance path is defined by the variable HOME.
# It also assumes DB2 is installed under the DB2 instance.
# If these statements are not correct, update the variable DB2PATH.
DB2PATH = $(HOME)/sqllib
# Use the cc compiler
CC=gcc
# The compiler options.
CFLAGS= $(EXTRA_CFLAGS) -I$(DB2PATH)/include
CFLAGSMT= $(EXTRA_CFLAGS) -I$(DB2PATH)/include -D_REENTRANT
# The required libraries
LIBS= -L$(DB2PATH)/lib -Wl,-rpath,$(DB2PATH)/lib -ldb2
LIBSMT= -L$(DB2PATH)/lib -Wl,-rpath,$(DB2PATH)/lib -ldb2 -lpthread
# To connect to a remote SAMPLE database cataloged on the client machine
# with another name, update the DB variable.
# Set DB, UID and PWD if neccesary

DB=dblx
UID=myuser

```

```

PWD=mypassword

COPY=cp
ERASE=rm -f
#####
# 2 -- MAKE CATEGORIES
#           2a - make clean
#           2b - make cleanall
#####
#*****
#           2a - make clean
#*****
clean :\
    cleangen \
    cleanemb
cleangen :
    $(ERASE) *.o *.map message.*
    $(ERASE) *.DEL *.TXT *.IXF
cleanemb :
    $(ERASE) selins.c
    $(ERASE) selins2.c
#*****
#           2b - make cleanall
#*****
cleanall : \
    clean
    $(ERASE) *.bnd
    $(ERASE) selins
    $(ERASE) selins2
#####
# 3 -- COMMANDS TO MAKE INDIVIDUAL SAMPLES
#           3a - utilities
#           3b - embedded SQL
#####
#*****
#           3a - utilities
#*****
utilemb.c : utilemb.sqc
    ./embprep utilemb $(DB) $(UID) $(PWD)
utilemb.o : utilemb.c
    $(CC) -c utilemb.c $(CFLAGS)
#*****
#           3b - embedded SQL
#*****
selins.c : selins.sqc
    ./embprep selins $(DB) $(UID) $(PWD)
selins : selins.c utilemb.o

```

```

$(CC) -o selins selins.c utilemb.o $(CFLAGS) $(LIBS)
selins2.c : selins2.sqc
./embprep selins2 $(DB) $(UID) $(PWD)
selins2 : selins2.c utilemb.o
$(CC) -o selins2 selins2.c utilemb.o $(CFLAGS) $(LIBS)

```

C.2.3 Script to run the C program

This is the script we used to run the C program.

```

#!/bin/bash
#
# First delete table
#####
db2 connect to udblp3o2 user db2inst1 using db2inst1
db2 drop table T1
db2 -tf T1.ddl
db2 alter table T1 append on
db2 connect reset
#
# Initialize parameters
#####
SRC=$1
TRG=$2
USR1=$3
PWD1=$4
USR2=$5
PWD2=$6
KEY=$7

clear
#
# Start of Measurement
#####
STARTTIME=`date +%s`
echo
echo Starttime: $STARTTIME "<->" `date`
echo -----
echo
./selins $SRC $TRG $USR1 $PWD1 $USR2 $PWD2 $KEY

ELAPSEDTIME=`date +%s`
echo
echo Enddate: $ELAPSEDTIME "<->" `date`
echo -----
echo
#
# End of Measurement
#####
echo
echo Elapsedtime: $(( $ELAPSEDTIME-$STARTTIME ))
echo -----
echo

```

C.3 DB2 DSNTIAUL program and DB2 Load

This section includes the following:

- JCL for DB2 DSNTIAUL utility
- JCL for DB2 Load utility

C.3.1 JCL for DB2 DSNTIAUL utility

This is the JCL we used to run the DSNTIAUL utility.

```
//UNLDT1 JOB CLASS=A,TIME=NOLIMIT,REGION=5M,MSGCLASS=H,
//          MSGLEVEL=(1,1)
//*JOBPARM SYSAFF=*
//JOBLIB DD DSN=DB2610.RUNLIB.LOAD,DISP=SHR
//DELETE EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=H
//SYSIN DD *
DELETE TPLEX.LINUXRAW.T1
//*
//UNLOAD EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(DBLX)
RUN PROGRAM(DSNTIAUL) PLAN(DSNTIB61) PARM('SQL')
//SYSPUNCH DD SYSOUT=*
//SYSREC00 DD DSN=TPLEX.LINUXRAW.T1,
//  DISP=(NEW,CATLG),UNIT=SYSDA,
//  DCB=(RECFM=FB,BUFNO=100),
//  SPACE=(CYL,(1900,100),RLSE)
//SYSIN DD *
SELECT
    DIGITS(T1_C01)
    ,DIGITS(T1_C02)
    ,DIGITS(T1_C03)
    ,DIGITS(T1_C04)
    ,CHAR(T1_C05)
    ,CHAR(T1_C06)
    ,CHAR(T1_C07)
    ,CHAR(T1_C08)
    ,T1_C09
    ,T1_C10
    ,T1_C11
    ,T1_C12
    ,T1_C13
    ,T1_C14
```

```

,T1_C15
,CHAR(T1_C16,44)
FROM LINUX.T1;

```

C.3.2 DB2 Load script

This is the script we used to load T1 table in DB2 for Linux on S/390.

```

load from /mnt/dasdi/T1.tbl
of asc
method L (1 12, 13 24, 25 36, 37 48, 49 60, 61 75, 76 90, 91 105, 106
106, 108 108, 110 119, 123 132, 136 145, 149 173, 174 183, 184 227)
replace into T1
statistics yes;

```

C.4 DB2 High Performance Unload utility

This is the JCL we used to run the DB2 High Performance Unload utility.

```

//HPUT1 JOB ,MSGLEVEL=(1,1),MSGCLASS=H,CLASS=A
/*JOBPARM SYSAFF=*
//*****
/*          DB2 UNLOAD JCL                                *
/*          IN THIS SAMPLE :                               *
/*          - THE DB2 SUBSYSTEM IS                        DB2P      *
/*          - THE DB2 UNLOAD LOAD MODULES ARE             *
/*          IN THE LOADLIB                                &VIZ004    *
/*          - THE EXECUTION REPORT WILL BE                *
/*          WRITTEN ON THE DDNAME                          SYSPRINT   *
//*****
//STEP1 EXEC PGM=INZUTILB,REGION=0M,DYNAMNBR=99,
//          PARM='DBL1,DB2UNLOAD'
//STEPLIB DD DSN=DB2610.SDSNEXIT,DISP=SHR
//          DD DSN=DB2610.SDSNLOAD,DISP=SHR
//          DD DSN=DB2610.HINZ110.SINZLINK,DISP=SHR
//SYSIN DD *
          UNLOAD TABLESPACE LINUX10.TST1
          UNLDDN UNLDDN1
          UNLMAXROWS 7689
/*
//SYSPRINT DD SYSOUT=*
//UNLDDN1 DD DSN=TPLEX.LINUXRAW.HUPT1,DISP=(,CATLG),
//          SPACE=(CYL,(1800,200),RLSE)

```

C.5 DB2 Export/Import script

This is the script we used to export and import DB2 data.

```
#!/bin/bash
#####
# Initialize parameters
#
# SRC source alias
# TRG target alias
# USR1 source userid
# PWD1 source password
# USR2 target userid
# PWD2 target password
# SRCTBL source table name
# TRGTBL target table name
# KEY order key
#####
SRC=$1
TRG=$2
USR1=$3
PWD1=$4
USR2=$5
PWD2=$6
SRCTBL=$7
TRGTBL=$8
KEY=$9

clear
#
# First connect to source
#####
db2 connect to $SRC user $USR1 using $PWD1
#
# Start export from source
#####
STARTTIME=`date +%s`
echo
echo Starttime: $STARTTIME
echo -----
echo
db2 export to $SRCTBL.ixf of ixf "select * from $SRCTBL where T1_C01 < $KEY"

EXPENDTIME=`date +%s`
#
# Disconnect from source
#####
db2 disconnect $SRC
#
# Connect to target
#####
db2 connect to $TRG user $USR2 using $PWD2
#
# Start import into target
#####
IMPORTTIME=`date +%s`
echo
echo Importtime: $IMPORTTIME
echo -----
echo
db2 import from $SRCTBL.ixf of ixf "replace into $TRGTBL"
ELAPSEDTIME=`date +%s`
```

```

echo
echo Enddate: $ELAPSEDTIME
echo -----
echo
#
# End of Measurement
#####
echo
echo Exporttime : (($EXPENDTIME-$STARTTIME))
echo Importtime : (($ELAPSEDTIME-$IMPORTTIME))
echo Elapsedtime : (($ELAPSEDTIME-$STARTTIME))
echo -----
echo
#
# Disconnect from target
#####
db2 disconnect $TRG

```

C.6 DB2 Export/Load scripts

This is the script we used to export and load DB2 data.

```

#!/bin/bash
#####
# Initialize parameters
#
# SRC source alias
# TRG target alias
# USR1 source userid
# PWD1 source password
# USR2 target userid
# PWD2 target password
# SRCTBL source table name
# TRGTBL target table name
# KEY order key for T1
#####
SRC=$1
TRG=$2
USR1=$3
PWD1=$4
USR2=$5
PWD2=$6
SRCTBL=$7
TRGTBL=$8
KEY=$9
clear
#
# First connect to source
#####
db2 connect to $SRC user $USR1 using $PWD1
#
# Start export from source
#####
STARTTIME=`date +%s`
echo
echo Starttime: $STARTTIME
echo -----
echo
db2 export to $SRCTBL.ixf of ixf "select * from $SRCTBL where T1_C01 < $KEY"
#
# Disconnect from source

```

```
#####
db2 disconnect $SRC
#
# Connect to target
#####
db2 connect to $TRG user $USR2 using $PWD2
#
# Start import into target
#####
IMPORTTIME=`date +%s`
echo
echo Importtime: $IMPORTTIME
echo -----
echo
db2 load from $SRCTBL.ixf of ixf replace into $TRGTBL
ELAPSEDTIME=`date +%s`
echo
echo Enddate: $ELAPSEDTIME
echo -----
echo
#
# End of Measurement
#####
echo
echo Elapsedtime: $(( $ELAPSEDTIME-$STARTTIME ))
echo -----
echo
#
# Disconnect from target
#####
db2 disconnect $TRG
```

Appendix D. Table definitions and query scripts

This appendix includes table definitions used in the query scenarios, as well as the query scripts.

D.1 Table definitions

```
CREATE TABLE T1
  (T1_C01 INT NOT NULL
  ,T1_C02 INT NOT NULL
  ,T1_C03 INT NOT NULL
  ,T1_C04 INT NOT NULL
  ,T1_C05 REAL NOT NULL
  ,T1_C06 REAL NOT NULL
  ,T1_C07 REAL NOT NULL
  ,T1_C08 REAL NOT NULL
  ,T1_C09 CHAR(1) NOT NULL
  ,T1_C10 CHAR(1) NOT NULL
  ,T1_C11 DATE NOT NULL
  ,T1_C12 DATE NOT NULL
  ,T1_C13 DATE NOT NULL
  ,T1_C14 CHAR(25) NOT NULL
  ,T1_C15 CHAR(10) NOT NULL
  ,T1_C16 VARCHAR(44) NOT NULL
  )
  IN LINUX10.TST1;
COMMIT;
```

```
CREATE TABLE T2
  (T2_C01 INTEGER NOT NULL
  ,T2_C02 CHAR(25) NOT NULL
  ,T2_C03 VARCHAR(152) NOT NULL
  )
  IN LINUX10.TST2;
COMMIT;
```

```

CREATE TABLE T3
  (T3_C01 INT NOT NULL
  ,T3_C02 INT NOT NULL
  ,T3_C03 CHAR(1) NOT NULL
  ,T3_C04 REAL NOT NULL
  ,T3_C05 DATE NOT NULL
  ,T3_C06 CHAR(15) NOT NULL
  ,T3_C07 CHAR(15) NOT NULL
  ,T3_C08 INT NOT NULL
  ,T3_C09 VARCHAR(79) NOT NULL
  )
  IN LINUX10.TST3;
COMMIT;

```

D.2 Q01 I/O-intensive query - no rows returned

```

select
  *
from
  DB.T1
where
  T1_C09 = 'value';

```

D.3 Q02 Processor-intensive query

```

select
  count(*) ,
  min(T1_C05) ,
  max(T1_C05) ,
  avg(T1_C05) ,
  min(T1_C06) ,
  max(T1_C06) ,
  avg(T1_C06) ,
  sum(T1_C06) as SUM_T1_C06 ,
  sum(T1_C06*(1-T1_C07)) as SUM_T1_C06_M07 ,
  sum(T1_C06*(1-T1_C07)*(1+T1_C08)) as SUM_T1_C06_M07_M08 ,
  min(T1_C07) ,
  max(T1_C07) ,
  avg(T1_C07) ,
  min(T1_C08) ,
  max(T1_C08) ,
  avg(T1_C08) ,
  min(T1_C08) ,
  max(T1_C08) ,
  avg(T1_C08) ,

```



```
        count_big(*) as CNT
from
    DB.T1
where
    T1_C01 <= 1920000000;
```

D.4 Q03 Trivial query

```
select
    count(*)
from
    DB.T2;
```

D.5 Q04 Short query and Q05 Short query with table in memory

```
select
    count(*)
from
    DB.T3;
```

D.6 Q06 I/O-intensive query - 40,000 rows returned

```
select
    *
from
    DB.T1
where
    T1_C01 <= 40000;
```

D.7 Q07 I/O-intensive query - 40,000 rows returned with Sort

```
select
    *
from
    DB.T1
where
    T1_C01 <= 40000
order by
    T1_C15;
```

Appendix E. Special notices

This publication is intended to help technical professionals who plan to install and use DB2 for Linux on S/390 for new BI opportunities. The information in this publication is not intended as the specification of any programming interfaces that are provided by Linux or DB2 for Linux. See the PUBLICATIONS section of the IBM Programming Announcement for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter 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 the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.



Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers

attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

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

e (logo)® 	Redbooks
IBM ®	Redbooks Logo 
AIX	OS/2
AS/400	OS/390
AT	PR/SM
CICS	QMF
CT	RAMAC
Current	RS/6000
DataJoiner	S/390
DataPropagator	SP
DB2	System/390
DB2 Connect	VM/ESA
DB2 Universal Database	WebSphere
DRDA	Wizard
ECKD	XT
Home Director	400
MQSeries	Lotus
Netfinity	

The following terms are trademarks of other companies:

Tivoli, Manage. Anything. Anywhere., The Power To Manage., Anything. Anywhere., TME, NetView, Cross-Site, Tivoli Ready, Tivoli Certified, Planet Tivoli, and Tivoli Enterprise are trademarks or registered trademarks of Tivoli Systems Inc., an IBM company, in the United States, other countries, or both. In Denmark, Tivoli is a trademark licensed from Københavns Sommer - Tivoli A/S.

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

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

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of

Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

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

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

Appendix F. Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

F.1 IBM Redbooks

For information on ordering this publication, see “How to get IBM Redbooks” on page 213.

- *Linux for S/390*, SG24-4987

F.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at ibm.com/redbooks for information about all the CD-ROMs offered, updates and formats.

CD-ROM Title	Collection Kit Number
IBM System/390 Redbooks Collection	SK2T-2177
IBM Networking Redbooks Collection	SK2T-6022
IBM Transaction Processing and Data Management Redbooks Collection	SK2T-8038
IBM Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
IBM AS/400 Redbooks Collection	SK2T-2849
IBM Netfinity Hardware and Software Redbooks Collection	SK2T-8046
IBM RS/6000 Redbooks Collection	SK2T-8043
IBM Application Development Redbooks Collection	SK2T-8037
IBM Enterprise Storage and Systems Management Solutions	SK3T-3694

F.3 Other resources

These publications are also relevant as further information sources:

- *IBM DB2 UDB for UNIX Quick Beginnings Version 7*, GC09-2970
- *IBM DB2 Connect Personal Edition for Linux Quick Beginnings*, GC09-2962
- *IBM DB2 Administration Guide: Implementation Version 7*, SC09-2944
- *IBM DB2 Administration Guide: Performance Version 7*, SC09-2945
- *IBM DB2 Administration Guide: Planning Version 7*, SC09-2946
- *IBM DB2 System Monitor Guide and Reference Version 7*, SC09-2956

- *IBM DB2 Command Reference Version 7*, SC09-2951
- *IBM DB2 SQL Reference Version 7*, SC09-2974, SC09-2975
- *IBM DB2 Data Movement Utilities Guide and Reference Version 7*, SC09-2955
- *IBM DB2 Administration API Reference Version 7*, SC09-2947
- *IBM DB2 Application Building Guide Version 7*, SC09-2948
- *IBM DB2 Application Development Guide Version 7*, SC09-2949
- *SuSE Linux 7.0 Installation, Networking, Know How* (available on SuSE Linux CD-ROM)

F.4 Referenced Web sites

These Web sites are also relevant as further information sources:

- DB2 publications
`http://www.ibm.com/cgi-bin/software/db2www/library/pubs.d2w/report`
- Linux on S/390 software
`http://linux.s390.org/`
- IBM Mart homepage
`http://www-1.ibm.com/servers/eserver/zseries/zmart/`
- DB2 UDB EE Download site
`http://www-4.ibm.com/software/data/db2/udb/downloads.html`
- QMF homepage and download site
`http://www.rocketsoftware.com/qmf/index.html`
- IBM Mart Installation Assistance
`http://www-1.ibm.com/servers/eserver/zseries/zmart/contact_us.html`
- Linux for zSeries homepage
`http://www-1.ibm.com/servers/eserver/zseries/os/linux/`
- Download sites for Linux on zSeries Distributors
`http://www-1.ibm.com/servers/eserver/zseries/os/linux/dist.html`

- IBM Linux homepage
<http://www.ibm.com/linux/>
- Linux.Online
<http://www.linux.org/>

How to get IBM Redbooks

This section explains how both customers and IBM employees can find out about IBM Redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** ibm.com/redbooks

Search for, view, download, or order hardcopy/CD-ROM Redbooks from the Redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

Redpieces are Redbooks in progress; not all Redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

Send orders by e-mail including information from the IBM Redbooks fax order form to:

	e-mail address
In United States or Canada	pubscan@us.ibm.com
Outside North America	Contact information is in the "How to Order" section at this site: http://www.elink.ibm.link.ibm.com/pbl/pbl

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	Country coordinator phone number is in the "How to Order" section at this site: http://www.elink.ibm.link.ibm.com/pbl/pbl

- **Fax Orders**

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	Fax phone number is in the "How to Order" section at this site: http://www.elink.ibm.link.ibm.com/pbl/pbl

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the Redbooks Web site.

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and Redbooks by accessing the IBM Intranet Web site at <http://w3.itso.ibm.com/> and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at <http://w3.ibm.com/> for redbook, residency, and workshop announcements.

IBM Redbooks fax order form

Please send me the following:

Title	Order Number	Quantity

First name	Last name
------------	-----------

Company

Address

City	Postal code	Country
------	-------------	---------

Telephone number	Telefax number	VAT number
------------------	----------------	------------

<input type="checkbox"/> Invoice to customer number	
---	--

<input type="checkbox"/> Credit card number	
---	--

Credit card expiration date	Card issued to	Signature
-----------------------------	----------------	-----------

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

Index

Numerics

64-bit architecture 1

A

administration 171
Alert Center 172
Application Programming Interface 171
applications
 servers 17
 web-enablement 16
archive file 81
authentication 179
authorization 180

B

binding utilities and applications 98
bootstrap tape 27, 28, 29
buffer pools 120

C

capacity usage 12
cloning 69
command line processor 171
consolidations 9
 application servers 16
 data marts 11
 gateway 15
 middleware 16
 networks 15
containers 121
Control Center 171
cost
 acquisition 12
 ownership 11
CTC/escon 72, 73

D

data marts 13
 building 111
 consolidation 11
 deployment 13
 distribution 9
 population 14
data model 131, 159

data population
 other tools 152
 performance summary 151
 scalability and performance 129
database security 179
DB2 administration 171
DB2 administration client 103
DB2 AutoLoader 155
DB2 client connectivity 99
 DB2 administration client 103
 DB2 Run-Time client 99
 QMF for Windows 104
DB2 Connect for Linux 94
 configuration parameters 94
 host database 96
 performance 108
 checklist 109
 setup 94
 TCP/IP node 95
DB2 Data Warehouse Center 153
DB2 DataJoiner 157
DB2 DataPropagator 152
DB2 DSNTIAUL 137
DB2 DSNTIAUL and DB2 Load 137
DB2 Export and Import 145
DB2 Export and Load 148
DB2 for Linux 79
 archive file 81
 commands 112
 customization 81
 DB2 instance 114
 default users 80
 installation 79, 82
 objects 111
 performance 105, 106
 background processing 108
 bufferpool hit ratio 107
 bufferpool sizes 107
 indexes 107
 locking 108
 log setup 107
 prefetching 107
 query degree 107
 query optimization 107
 sorting 108
 statistics 107
 table reorganization 107

- post-installation 92
- requirements 79
- DB2 for Linux data mart 111
 - buffer pools 120
 - containers 121
 - database 114
 - indexes 126
 - recommendations 126
 - table spaces 121
 - tables 125
- DB2 for OS/390 settings 94
- DB2 for OS/390 V7.1 Unload 155
- DB2 Governor 176
- DB2 High Performance Unload 139
- DB2 instance 114
- DB2 Performance Monitor 176
- DB2 performance tools 173
- DB2 Run-Time client 99
 - configuring 101
 - performance 110
 - TCP/IP node 101
- DB2 security administration 179
- DB2 users 80
- disk space 62
 - image space 62
 - paging space 62
- distributed data marts 9
- DSNTIAUL and HPU 144

E

- end user functionality 11
- event monitors 175

F

- federated database 131
- file systems 37, 57, 64
- FTP server 50, 58

H

- hard drive 36
- hints and tips 72

I

- I/O intensive queries
 - no rows returned 163
 - Q01 163
- I/O intensive query 163

- 40,000 rows returned 165
 - Q06 165
- 40,000 rows returned with Sort 166
 - Q07 166
- I/O scalability 163
- ICKDSF 63
- INETD 53, 59
- inetd 44
- Information Center 172

J

- joins 16
- journal 172

K

- kernels 41

L

- License Center 172
- Linux image 62, 64
 - adding an OSA Express Giga Bit Ethernet connection 77
 - cloning 69
 - hints and tips 72
 - installation 66
 - network connection 72
 - post-installation 68
 - starting 66
- Linux on S/390 1, 25
 - access authorizations 31
 - available softwares 2
 - bootstrap tape 27
 - cleanup 46
 - configuration files 73
 - configurations 5
 - file system 37
 - FTP 27
 - installation 26
 - Linux kernel 41
 - load configuration 38
 - network configurations 42
 - network connections 30
 - network information 26
 - packages 39
 - performance 106
 - CPU 106
 - I/O 106

- memory 106
- swap file system 32
- system disks 32
- time-zone configuration 41
- Virtual Image Facility 6
- virtual images under VM 6
- YaST 33
- Linux performance tools 177
- Linux workstation 31
- load parameter file 51

M

- master Linux 48, 53, 55, 57, 60
- monitoring 171
- mpstat 178

N

- name server 45
- network configuration 42, 73
- network connection 72
- network traffic scalability 165
- news 44
- NFS 34, 35, 44

O

- OSA Express Giga Bit Ethernet 72
 - adding a connection 75
- OSA2 72

P

- packages 39
- parmfile 51, 60
- portmapper 44, 59
- privileges 180
- processor intensive queries 161
 - Q02 161

Q

- QMF for Windows 104
- query
 - scalability and performance 159

R

- Remote Procedure Calls 44

S

- Satellite Administration Center 172
- scalability and performance tests
 - query 159
- scalability and performance tests
 - data population 129
- Script Centre 172
- Select/Insert SQL statements in a C program 134
- Send Mail feature 45
- server farms 4
- server node
 - catalogue database 97, 102
- silo 60
- single distributed SQL statement 131
- Snapshot monitoring 174
- sort scalability 166
- SQL Explain 173
- SuSE Linux 7.0 25
 - customizing for DB2 81
 - installation 26
 - under VIF 48
- swap file 32, 36, 47
- swap partition 36, 57, 64
- systems management 12

T

- test environment 19, 129
 - product implementations 22
 - scalability and performance tests 22
 - test configurations 21
 - test objectives 19
 - test scenarios 19
- time to market 11
- time-zone configuration 41

V

- VIF 48
 - base software 59
 - collecting information 51
 - configuration 59
 - dependencies 59
 - disk space 62
 - file systems 57
 - free space 65
 - FTP 50
 - hypervisor 48, 52
 - installation 52
 - kernel 59

- Linux image 62
- master Linux 48, 53, 56, 57
- network connection 50, 65
- parmfile 51
- settings 65
- Telnet 67
- Virtual Image Facility 7
- virtual images under VM 6
- vmstat 177

W

- wizards 176
- workload 160

X

- xpram 32

Y

- YaST 33, 57, 68

IBM Redbooks review

Your feedback is valued by the Redbook authors. In particular we are interested in situations where a Redbook "made the difference" in a task or problem you encountered. Using one of the following methods, **please review the Redbook, addressing value, subject matter, structure, depth and quality as appropriate.**

- Use the online **Contact us** review redbook form found at ibm.com/redbooks
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Document Number	SG24-5687-00
Redbook Title	e-Business Intelligence: Leveraging DB2 for Linux on S/390
Review	<div></div> <div></div> <div></div> <div></div> <div></div> <div></div>
What other subjects would you like to see IBM Redbooks address?	<div></div> <div></div> <div></div>
Please rate your overall satisfaction:	<input type="radio"/> Very Good <input type="radio"/> Good <input type="radio"/> Average <input type="radio"/> Poor
Please identify yourself as belonging to one of the following groups:	<input type="radio"/> Customer <input type="radio"/> Business Partner <input type="radio"/> Solution Developer <input type="radio"/> IBM, Lotus or Tivoli Employee <input type="radio"/> None of the above
Your email address: The data you provide here may be used to provide you with information from IBM or our business partners about our products, services or activities.	<input type="radio"/> Please do not use the information collected here for future marketing or promotional contacts or other communications beyond the scope of this transaction.
Questions about IBM's privacy policy?	The following link explains how we protect your personal information. ibm.com/privacy/yourprivacy/



e-Business Intelligence: Leveraging DB2 for Linux on S/390

(0.2" spine)
0.17" <-> 0.473"
90 <-> 249 pages



Redbooks

e-Business Intelligence: Leveraging DB2 for Linux on S/390

**Linux and DB2 for
Linux setups on
S/390**

**Data population and
query scalability and
performance tests**

**Administration and
monitoring**

This IBM Redbook discusses leveraging DB2 for Linux on S/390 for new e-BI opportunities. It shows the possible Linux configurations on S/390 and describes the advantages of consolidating UNIX data mart farms on unique IBM e-servers using DB2 for Linux on S/390.

The book describes how to set up DB2 for Linux in a Linux LPAR and how to customize the database connections (using DB2 Connect for Linux) to the DB2 for OS/390 data warehouse residing in an OS/390 LPAR on the same machine.

It also includes early scalability and performance tests on queries and data population techniques run at the IBM S/390 Teraplex Center in Poughkeepsie, New York.

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