

IBM Tivoli Identity Manager

Performance Tuning Guide



IBM Tivoli Identity Manager

Performance Tuning Guide



Note

Before using this information and the product it supports, read the information in "Notices" on page 103.

Edition notice

Note: This edition applies to version 5.1 of IBM Tivoli Identity Manager, (product number 5724-C34) and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 2007, 2010.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

About this publication

The IBM® Tivoli Identity Manager *Performance Tuning Guide* provides information on tuning middleware for IBM Tivoli Identity Manager versions 5.0 and 5.1. It includes tuning settings for:

WebSphere®,

Database servers (IBM DB2®, Oracle, and MS SQL)

Directory servers (IBM Tivoli® Directory Server and Sun ONE Directory Server)

IBM Tivoli Directory Integrator

IBM Tivoli Identity Manager application

IBM Tivoli Identity Manager adapters.

This edition includes a troubleshooting, best practices, and regular maintenance sections as well. This publication is a working document and is updated as more information becomes available

Intended audience

This publication is for system and security administrators who install, maintain, or administer software on their computer systems. Readers are expected to understand system and security administration concepts. Additionally, the reader must understand administration concepts for the following types of products:

- Database servers
- Directory servers
- Application servers

Publications

Read the descriptions of the IBM Tivoli Identity Manager library. To determine which additional publications you might find helpful, read "Prerequisite publications" on page v and "Related publications" on page vi. After you determine the publications you need, the instructions in "Accessing publications online" on page vi.

IBM Tivoli Identity Manager library

The publications in the Tivoli Identity Manager technical documentation library can be found at the following URL:

<http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itim.doc/>

The publications in the Tivoli Identity Manager technical documentation library are organized into the following categories:

- Release information
- Online user assistance
- Server installation and configuration
- Problem determination
- Technical supplements
- Adapter installation and configuration

Release Information

IBM Tivoli Identity Manager Quick Start Guide helps you install a base configuration of IBM Tivoli Identity Manager.

IBM Tivoli Identity Manager Information Center provides software and hardware requirements for IBM Tivoli Identity Manager and additional fix, patch, and other support information. This publication also includes known limitations, problems, and workarounds.

Online user assistance:

IBM Tivoli Identity Manager Information Center provides online help topics and an information center for all Tivoli Identity Manager administrative tasks.

Server installation and configuration

Tivoli Identity Manager Server Installation and Configuration Guide provides installation and configuration information for Tivoli Identity Manager.

IBM Tivoli Identity Manager Separate System Upgrade and Data Migration Guide provides upgrade and data migration information for Tivoli Identity Manager.

Problem determination

Tivoli Identity Manager Problem Determination Guide provides problem determination, and logging information for Tivoli Identity Manager.

Tivoli Identity Manager Messages Guide provides message information for IBM Tivoli Identity Manager.

Database and schema information

IBM Tivoli Identity Manager Database and Schema Reference describes some of the data structures used by IBM Tivoli Identity Manager.

Technical supplements:

The following technical supplements are provided by developers or by other groups who are interested in this product:

- Redbooks® and white papers are available at <http://www.redbooks.ibm.com/>.
- Technotes are available at <http://www.redbooks.ibm.com/redbooks.nsf/tips/>.
- Field guides are available at http://www.ibm.com/software/sysmgmt/products/support/Field_Guides.html.
- For an extended list of other Tivoli Identity Manager resources, search the following IBM developerWorks® Web at <http://www.ibm.com/developerworks/>.

Adapter installation and configuration

The IBM Tivoli Identity Manager Server technical documentation library also includes an evolving set of platform-specific installation documents for the adapter components of an IBM Tivoli Identity Manager implementation.

Locate adapter documentation at <http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itim.doc/>

Performance and tuning

IBM Tivoli Identity Manager Performance Tuning Guide provides information to help you optimize the use of resources for IBM Tivoli Identity Manager.

Skills and training

Additional skills and technical training information might be available at the following websites:

- IBM Professional Certification at <http://www.ibm.com/certify/>
Search on **identity manager** to locate available classes and certification offerings.
- Virtual Skills Center for Tivoli Software at <http://www.cgselearning.com/tivoliskills/>
- Tivoli Education Software Training road maps at <http://www-01.ibm.com/software/tivoli/education>
- Tivoli Technical Exchange at <http://www.ibm.com/software/sysmgmt/products/support/>

Prerequisite publications

To use the information in this book effectively, you must know the products that are prerequisites for IBM Tivoli Identity Manager. Publications are available from the following locations:

Operating systems

- **AIX:** <http://publib.boulder.ibm.com/infocenter/pseries/v5r3/topic/>
- **Sun Solaris:** <http://docs.sun.com/app/docs/prod/solaris.10>
- **Microsoft Windows Server 2003:**
 - Support: <http://www.microsoft.com/windowsserver2003/support/default.mspx>
 - Documentation: <http://www.microsoft.com/windowsserver2003/proddoc/default.mspx>
- **Microsoft Windows Server 2008:**
 - Support: <http://www.microsoft.com/windowsserver2008/en/us/support.aspx>
 - Documentation: <http://www.microsoft.com/windowsserver2008/en/us/productdocumentation>.
- **Red Hat Linux:** <http://www.redhat.com/docs/>
- **SUSE Linux:** <http://www.novell.com/documentation/suse.html>

WebSphere Application Server:

- Hardware and software requirements: <http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp>
- Support: <http://www.ibm.com/software/webservers/appserv/was/support/>
- Information center: <http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp>

IBM DB2 Database

- Support: <http://www.ibm.com/software/data/db2/udb/support.html>
- Information center: <http://publib.boulder.ibm.com/infocenter/db2help/index.jsp>
- Documentation: http://www-306.ibm.com/software/data/db2/support/db2_9/ and <http://www.ibm.com/software/data/db2/udb/support/manualsv9.html>
- DB2 product family: <http://www.ibm.com/software/data/db2/>

- Fix packs by version: <http://www-1.ibm.com/support/docview.wss?rs=71&uid=swg21255572>
- System requirements: <http://www.ibm.com/software/data/db2/udb/sysreqs.html>

IBM Tivoli Directory Server

- Support: <http://www.ibm.com/software/sysmgmt/products/support/>
- Information center: <http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/>

IBM Tivoli Directory Integrator:

- Support: <http://www.ibm.com/software/sysmgmt/products/support/>
- Information center: <http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?toc=/>

Related publications

The Tivoli Software Library provides various Tivoli publications such as white papers, data sheets, demonstrations, Redbooks, and announcement letters. The Tivoli Software Library is available at <http://www.ibm.com/software/tivoli/literature/>.

Accessing terminology online

The IBM Terminology Web site consolidates the terminology from IBM product libraries in one convenient location. You can access the Terminology Web site at the following Web address:

<http://www.ibm.com/software/globalization/terminology>

Accessing publications online

IBM posts publications for this and all other Tivoli products, as they become available and whenever they are updated, to the Tivoli Documentation Central Web site at <http://www.ibm.com/tivoli/documentation>

Note: If you print PDF documents on other than letter-sized paper, set the option in the **File** → **Print** window that allows Adobe Reader to print letter-sized pages on your local paper.

Ordering publications

You can order many Tivoli publications online at <http://www.elink.ibm.com/publications/servlet/pbi.wss>.

You can also order by telephone by calling one of these numbers:

- In the United States: 800-879-2755
- In Canada: 800-426-4968

In other countries, contact your software account representative to order Tivoli publications. To locate the telephone number of your local representative, perform the following steps:

1. Go to <http://www.elink.ibm.com/publications/servlet/pbi.wss>.
2. Select your country from the list and click **Go**.
3. Click **About this site** in the main panel to see an information page that includes the telephone number of your local representative.

Accessibility

Accessibility features help users with a physical disability, such as restricted mobility or limited vision, to use software products successfully. With this product, you can use assistive technologies to hear and navigate the interface. You can also use the keyboard instead of the mouse to operate all features of the graphical user interface.

For additional information, see the *Accessibility features for IBM Tivoli Identity Manager* topic in the IBM Tivoli Identity Manager Information Center.

Tivoli technical training

For Tivoli technical training information, refer to the following IBM Tivoli Education Web site at <http://www.ibm.com/software/tivoli/education>.

Tivoli user groups

Tivoli user groups are independent, user-run membership organizations that provide Tivoli users with information to assist them in the implementation of Tivoli Software solutions. Through these groups, members can share information and learn from the knowledge and experience of other Tivoli users. Tivoli user groups include the following members and groups:

- 23,000+ members
- 144+ groups

Access the link for the Tivoli Users Group at www.tivoli-ug.org.

Support information

If you have a problem with your IBM software, you want to resolve it quickly. IBM provides the following ways for you to obtain the support you need:

Online

Access the Tivoli Software Support site at <http://www.ibm.com/software/sysmgmt/products/support/index.html?ibmprd=tivman>. Access the IBM Software Support site at <http://www.ibm.com/software/support/probsub.html>.

IBM Support Assistant

The IBM Support Assistant is a free local software serviceability workbench that helps you resolve questions and problems with IBM software products. The Support Assistant provides quick access to support-related information and serviceability tools for problem determination. To install the Support Assistant software, go to <http://www.ibm.com/software/support/isa>.

Troubleshooting Guide

For more information about resolving problems, see the problem determination information for this product.

Conventions used in this publication

This publication uses several conventions for special terms and actions, operating system-dependent commands and paths, and margin graphics.

Typeface conventions

This publication uses the following typeface conventions:

Bold

- Lowercase commands and mixed case commands that are otherwise difficult to distinguish from surrounding text
- Interface controls (check boxes, push buttons, radio buttons, spin buttons, fields, folders, icons, list boxes, items inside list boxes, multicolumn lists, containers, menu choices, menu names, tabs, property sheets), labels (such as **Tip:**, and **Operating system considerations:**)
- Keywords and parameters in text

Italic

- Citations (examples: titles of publications, diskettes, and CDs)
- Words defined in text (example: a nonswitched line is called a *point-to-point line*)
- Emphasis of words and letters (words as words example: "Use the word *that* to introduce a restrictive clause."; letters as letters example: "The LUN address must start with the letter *L*.")
- New terms in text (except in a definition list): a *view* is a frame in a workspace that contains data.
- Variables and values you must provide: ... where *myname* represents....

Monospace

- Examples and code examples
- File names, programming keywords, and other elements that are difficult to distinguish from surrounding text
- Message text and prompts addressed to the user
- Text that the user must type
- Values for arguments or command options

Definitions for HOME and other directory variables

The following table contains the default definitions that are used in this guide to represent the HOME directory level for various product installation paths. You can customize the installation directory and HOME directory for your specific implementation. If this is the case, you need to make the appropriate substitution for the definition of each variable represented in this table.

The value of path varies for these operating systems.

For Windows, the default path is drive:\Program Files.

For UNIX/Linux, the default path is /opt.

| Path Variable | Default Definition | Description |
|---------------|---|---|
| DB_HOME | Windows: <i>path</i> \IBM\SQLLIB UNIX/Linux: <i>path</i> /ibm/db2/V9.1 <i>path</i> /ibm/db2/V9.5 | The directory that contains the DB2 Database for IBM Tivoli Identity Manager. |

| Path Variable | Default Definition | Description |
|-------------------------|--|--|
| DB_INSTANCE_HOME | <p>Windows: drive:\dbinstancename</p> <p>Solaris: /export/home/dbinstancename</p> <p>Other UNIX/Linux: /home/dbinstancename</p> | The directory that contains the DB2 instance for IBM Tivoli Identity Manager. |
| ITDS_HOME | <p>Windows Version 6.1 <i>path</i>\IBM\LDAP\V6.1</p> <p>Windows Version 6.2 <i>path</i>\IBM\LDAP\V6.2</p> <p>UNIX/Linux: Version 6.1 <i>path</i>/ibm/ldap/V6.1</p> <p>UNIX/Linux: Version 6.2 <i>path</i>/ibm/ldap/V6.2</p> | The directory that contains the IBM Tivoli Directory Server code. |
| ITDS_INSTANCE_HOME | <p>Windows: <i>path</i>\idsslapd-instance_owner_name</p> <p>The value of drive might be C:\ on Windows systems. An example of instance_owner_name might be ldapdb2. For example, the log file might be C:\idsslapd-ldapdb2\logs\ibmslapd.log.</p> <p>UNIX/Linux: /home/instance_owner_name/idsslapd-instance_owner_name</p> <p>Solaris: /export/home/instance_owner_name/idsslapd-instance_owner_name</p> <p>An example of instance_owner_name might be ldapdb2. For example, the log file might be /export/home/ldapdb2/idsslapd-ldapdb2/logs/ibmslapd.log</p> | The directory that contains the IBM Tivoli Directory Server Version 6.0 or Version 6.1 instance. |
| ITDI_HOME | <p>Windows Version 6.1.1 <i>path</i>\IBM\TDI\V6.1.1</p> <p>Windows Version 7.0 <i>path</i>\IBM\TDI\V7.0</p> <p>UNIX/Linux Version 6.1.1: <i>path</i>/IBM/TDI/V6.1.1</p> <p>UNIX/Linux Version 7.0: <i>path</i>/IBM/TDI/V7.0</p> | The directory that contains the IBM Tivoli Directory Integrator Server code. Also, where adapters are installed. |
| ITIM_HOME | <p>Windows: <i>path</i>\IBM\itim</p> <p>UNIX/Linux: <i>path</i>/IBM/itim</p> | The base directory that contains the IBM Tivoli Identity Manager code, configuration, and documentation. |
| TIVOLI_COMMON_DIRECTORY | <p>Windows: <i>path</i>\IBM\tivoli\common</p> <p>UNIX/Linux: <i>path</i>/IBM/tivoli/common</p> | The central location for all serviceability related files, such as logs and first-failure capture data. |

| Path Variable | Default Definition | Description |
|----------------------|---|--|
| WAS_HOME | Windows: <i>path\IBM\WebSphere\AppServer</i> UNIX/Linux: <i>path/IBM/WebSphere/AppServer</i> | The directory that contains the WebSphere Application Server code. |
| WAS_PROFILE_HOME | Windows: <i>path\IBM\WebSphere\AppServer\profiles\profile_name</i> UNIX/Linux: <i>path/IBM/WebSphere/AppServer/profiles/profile_name</i> | The directory that contains the WebSphere Application Server custom profile. |
| WAS_NDM_PROFILE_HOME | Windows: : <i>path\IBM\WebSphere\AppServer\profiles\Dmgr01</i> UNIX/Linux: <i>path/IBM/WebSphere/AppServer/profiles/Dmgr01</i> | The directory that contains the WebSphere Application Server Network Deployment Manager profile. |

Operating system-dependent variables and paths

This guide uses the Windows convention for specifying environment variables and for directory notation.

When using the UNIX/Linux command line, replace %variable% with \$variable for environment variables, and replace each backslash (\) with a forward slash (/) in directory paths. The names of environment variables are not always the same in Windows and UNIX/Linux. For example, %TEMP% in the Windows operating system is equivalent to /tmp in a UNIX/Linux operating system.

Note: If you are using the bash shell on a Windows system, you can use the UNIX conventions.

Contents

| | |
|--|------------|
| About this publication | iii |
| Intended audience | iii |
| Publications | iii |
| IBM Tivoli Identity Manager library | iii |
| Prerequisite publications | v |
| Related publications | vi |
| Accessing terminology online | vi |
| Accessing publications online | vi |
| Ordering publications | vi |
| Accessibility | vii |
| Tivoli technical training | vii |
| Tivoli user groups | vii |
| Support information | vii |
| Conventions used in this publication. | vii |
| Typeface conventions | viii |
| Definitions for HOME and other directory variables | viii |
| Operating system-dependent variables and paths | x |

| | |
|--|----------|
| Chapter 1. Tuning for high-yield performance improvements | 1 |
|--|----------|

| | |
|--|----------|
| Chapter 2. The initial tuning | 3 |
|--|----------|

| | |
|---|----------|
| Chapter 3. Resource allocation | 5 |
| Allocating memory | 5 |
| Allocating processor usage. | 6 |
| Allocating disk space for storage | 6 |

| | |
|--|----------|
| Chapter 4. Tuning IBM WebSphere Application Server | 9 |
| Adjusting the Java virtual machine size | 9 |
| Configuring WebSphere Performance Monitoring Infrastructure. | 10 |
| Configuring WebSphere JDBC connections | 11 |
| Performance implications for Java 2 Security | 12 |

| | |
|---|-----------|
| Chapter 5. Tuning IBM HTTP Server | 13 |
| Optimizing IBM HTTP Server connections | 13 |
| Enabling content compression for the IBM HTTP Server | 14 |
| Improving the caching of static content served from the IBM HTTP Server | 14 |
| Edge Side Include caching | 16 |
| Configuring the Edge Side Include cache size | 16 |
| Configuring the Edge Side Include cache timeout | 17 |

| | |
|--|-----------|
| Chapter 6. Tuning IBM Tivoli Identity Manager | 19 |
| Configuring LDAP connection pooling | 19 |
| Configuring list controls | 19 |
| Configuring report data synchronization. | 20 |
| Configuring report batch sizes | 21 |

| | |
|--|----|
| Configuring e-mail notifications | 21 |
| Using the recycle bin | 22 |
| Disabling the recycle bin | 22 |
| Emptying the recycle bin | 23 |
| Working with reconciliations | 24 |
| Limiting attributes returned from the adapter | 24 |
| Reducing policy enforcements | 24 |
| Configuring reconciliation threads. | 25 |
| Configuring the maximum duration of a reconciliation | 26 |
| Configuring paged searches | 26 |
| Enabling server-side sorting | 28 |
| Configuring the ACI cache | 28 |
| Controlling the size of the database | 29 |

| | |
|---|-----------|
| Chapter 7. IBM Tivoli Identity Manager adapters | 31 |
| Tuning the Microsoft Active Directory adapter. | 31 |
| Configuring attributes returned during an Active Directory reconciliation | 31 |
| Configuring the number of threads for the Active Directory adapter | 32 |
| Tuning the LDAP adapter | 33 |
| Tuning the RACF adapter | 33 |

| | |
|---|-----------|
| Chapter 8. Tuning Tivoli Directory Integrator | 35 |
| Configuring logging levels for Tivoli Directory Integrator | 35 |
| Using the DSML connector with Tivoli Directory Integrator | 36 |
| Tuning the RMI Dispatcher | 36 |
| Configuring timeouts for large reconciliations | 36 |
| Configuring the number of concurrently running assembly lines | 37 |
| Configuring assembly line caching | 37 |

| | |
|--|-----------|
| Chapter 9. Database servers used with IBM Tivoli Identity Manager | 39 |
| Tuning IBM DB2. | 39 |
| Enabling the self-tuning memory manager | 39 |
| Configuring row-level compression | 41 |
| Configuring database connections for DB2 databases | 42 |
| Configuring table spaces for IBM DB2 databases | 44 |
| Table compression candidates for the IBM Tivoli Identity Manager database | 47 |
| Configuring transaction logs for DB2 databases | 48 |
| Configuring database application heaps | 49 |
| Configuring automatic statistics collection for the IBM Tivoli Identity Manager database | 49 |
| Updating IBM Tivoli Identity Manager database statistics for DB2 databases | 51 |
| Changing the maximum number of open files | 52 |

| | |
|---|----|
| Adjusting lock list and maximum locks | 53 |
| Changing the lock timeout | 53 |
| Disabling the EXTSHM parameter on AIX | 53 |
| Improving disk I/O performance | 54 |
| Tuning Oracle | 54 |
| Configuring the init.ora configuration file | 54 |
| Configuring database connections for Oracle databases | 54 |
| Enabling XA recovery operations | 55 |
| Configuring open cursors. | 55 |
| Configuring table spaces for Oracle databases | 56 |
| Configuring IBM Tivoli Identity Manager indexes for Oracle databases | 58 |
| Updating IBM Tivoli Identity Manager database statistics for Oracle databases | 59 |
| Tuning Microsoft SQL Server | 59 |
| Configuring indexes on Microsoft SQL Server databases | 60 |

Chapter 10. Directory servers supported by IBM Tivoli Identity Manager 63

| | |
|--|----|
| Tuning Tivoli Directory Server | 63 |
| Configuring cache sizes | 63 |
| Configuring paging parameters. | 65 |
| Configuring database buffer pools for the Tivoli Directory Server database. | 66 |
| Disabling file system caching | 67 |
| Table compression candidates for the IBM Tivoli Directory Server database. | 68 |
| Configuring transaction logs for the Tivoli Directory Server database. | 69 |
| Configuring database statement heaps | 70 |
| Configuring system limits | 70 |
| Configuring attribute indexes for Tivoli Directory Server | 71 |
| Configuring DB2 indexes | 72 |
| Configuring automatic statistics collection for the Tivoli Directory Server database | 73 |
| Updating Tivoli Directory Server database statistics | 74 |
| Configuring the maximum open files. | 75 |
| Disabling hash joins | 76 |

| | |
|---|----|
| Improving disk I/O performance | 76 |
| Tuning Sun ONE Directory Server. | 76 |
| Configuring the All IDs Threshold value | 76 |
| Configuring attribute indexes for Sun ONE Directory Server. | 78 |
| Configuring cache sizes | 79 |
| Configuring the referential integrity plug-in | 80 |

Chapter 11. Improving operating system performance 81

Chapter 12. Best practices 83

Chapter 13. Planning a maintenance schedule 87

Chapter 14. Troubleshooting IBM Tivoli Identity Manager 89

| | |
|---|----|
| Sun ONE Directory Server slow query performance | 89 |
| Tivoli Directory Server outages | 91 |
| Tivoli Directory Server slow queries | 91 |
| Governing policy search errors | 93 |
| Java OutOfMemory errors | 94 |
| Transaction rollback errors | 94 |

Chapter 15. Identifying performance bottlenecks 97

Chapter 16. Monitoring system resources 99

| | |
|---|-----|
| Using IBM Tivoli Monitoring scripts | 99 |
| Enabling DB2 monitoring. | 99 |
| Collecting DB2 snapshots. | 99 |
| Configuring the DB2 statement monitor | 100 |
| Using the DB2 statement monitor | 100 |
| Calculating the buffer pool hit ratio | 101 |

Notices 103

Index 107

Chapter 1. Tuning for high-yield performance improvements

Small changes in indexes and memory allocation to the database can yield large performance improvements.

There are several thousand different parameters that you can modify to tune WebSphere Application Server, the IBM Tivoli Identity Manager product, directory servers, and database servers. When setting up an acceptance or production environment, read each topic and perform the applicable tuning for your systems. The database statistics tuning are a vital part of the IBM Tivoli Identity Manager product performance.

If you are setting up a test environment and want to get started as quickly as possible, focus on the following areas:

“Adjusting the Java virtual machine size” on page 9

IBM Tivoli Identity Manager, version 5.0 and 5.1, runs on 64-bit JVMs on supported platforms. Using a 64-bit JVM, you can allocate 2 GB or more of memory. You might need to allocate more memory for very large (more than 6 million accounts) reconciliations.

“Configuring buffer pools for the IBM Tivoli Identity Manager database” on page 43

DB2 buffer pools must be large enough so that most table searches can read directly from memory instead of the disk. You can measure this value by looking at the hit ratio for the buffer pools.

“Configuring attribute indexes for Tivoli Directory Server” on page 71

Indexing the attributes on which applications search increases Tivoli Directory Server performance. Tivoli Directory Server indexes automatically translate into DB2 indexes when you update the Tivoli Directory Server schema for those attributes.

“Configuring attribute indexes for Sun ONE Directory Server” on page 78

You can increase Sun ONE Directory Server performance by indexing the attributes on which applications search.

“Updating IBM Tivoli Identity Manager database statistics for DB2 databases” on page 51

DB2 requires statistics on the number of rows in the tables and available indexes to efficiently execute queries. DB2 version 9 can update the statistics automatically, or you can manually update the statistics.

“Updating IBM Tivoli Identity Manager database statistics for Oracle databases” on page 59

You must gather and update database statistics at regular intervals. Intervals can be one week to one month on a production IBM Tivoli Identity Manager system or after processing a large amount of data.

Chapter 2. The initial tuning

You can implement most tuning in either a new or an existing environment. When tuning the database in a new environment, you must prime your database statistics for better performance.

To prime the statistics, start by loading a small set of users and accounts and updating the database statistics. For DB2, use the RUNSTATS command and the corresponding manual cardinality tuning. Failing to prime the database can result in poor performance or transaction rollbacks.

Consider enabling automatic statistics collection for DB2, version 9, databases.

Related tasks

“Updating IBM Tivoli Identity Manager database statistics for DB2 databases” on page 51

DB2 requires statistics on the number of rows in the tables and available indexes to efficiently execute queries. DB2 version 9 can update the statistics automatically, or you can manually update the statistics.

“Updating IBM Tivoli Identity Manager database statistics for Oracle databases” on page 59

You must gather and update database statistics at regular intervals. Intervals can be one week to one month on a production IBM Tivoli Identity Manager system or after processing a large amount of data.

“Updating Tivoli Directory Server database statistics” on page 74

DB2 requires information about the number of rows in the tables and what indexes are available so that it can efficiently fulfill queries. If Tivoli Directory Server database is running DB2, version 9, you can set RUNSTATS to run automatically. Version 9 is the default for Tivoli Directory Server, version 6.1. RUNSTATS eliminates the need for running it manually.

“Configuring automatic statistics collection for the IBM Tivoli Identity Manager database” on page 49

Administrators can configure automatic statistics collection so that DB2 automatically updates database statistics. Automatic collection eliminates the necessity of manually running a periodic statistics collection against the database.

“Configuring automatic statistics collection for the Tivoli Directory Server database” on page 73

Administrators can use automatic statistics collection so that DB2 automatically updates the necessary database statistics. Automatic collection eliminates the necessity of manually running a periodic statistics collection against the database.

Chapter 3. Resource allocation

Use the correct tuning values for the memory, CPU, and disk resources to avoid over allocating them.

Managing tuning values becomes more complex when more than one middleware component run on the same system. An example is running the IBM Tivoli Identity Manager server, DB2, and Tivoli Directory Server all on the same server. Regardless of configuration, you must calibrate the following resources:

- “Allocating memory”
- “Allocating processor usage” on page 6
- “Allocating disk space for storage” on page 6

Allocating memory

You can adjust how much memory middleware components use. When calculating how to allocate memory to middleware components, keep in mind the following considerations.

- If middleware memory settings are too high, the operating system might swap out memory to disk if the physical memory is exceeded. Memory swapping results in poor performance. After setting up or changing the memory values, monitor the memory and swap space. If anything is swapped out to disk, readjust the settings to correct the problem.
- The 32-bit processes can allocate only 2 GB (AIX, Windows, and some Linux kernels) to 4 GB (Solaris) of RAM. If you configure a 32-bit process to allocate more than the OS-specific limit on process memory, the application might halt or unexpectedly fail. Tivoli Directory Server is an example of this type of application. The memory limit for 64-bit processes is large enough to not be an issue.
- IBM Tivoli Directory Server and Sun ONE Directory Server have internal caches that contribute to the size of their processes. The size of the process must not exceed 2 GB on 32-bit platforms such as Windows. When the LDAP server reaches the 2 GB limit, it refuses new connections and fails. For IBM Tivoli Directory Server, the entry cache size limit determines the number of entries in the cache, not the size of the cache. The size of each cache entry varies based on the IBM Tivoli Identity Manager configuration and any extensions to the base IBM Tivoli Directory Server schema. In rare cases, the default cache values might exceed the 2 GB limit.
- Buffer pools account for a large amount of the memory used by IBM DB2. The application control heaps, the sort heaps, and the statement heaps also use memory. In addition to database-wide memory heaps, each database connection results in memory allocations. Do not overlook these per-connection memory requirements when computing how much memory to allocate to IBM DB2.
- A large part of the WebSphere Application Server memory usage is the JVM size. The size of the JVM does not set an upper bound on the amount of memory that the WebSphere Application Server uses.
- Operating system limits can prevent processes from accessing all available memory. Confirm the appropriate `ulimit` values for your system to ensure that they do not artificially limit the amount of memory available. Determine the

limits using `ulimit -a`. Increase memory and file limits to high or unlimited values before starting IBM Tivoli Identity Manager or related middleware.

Related tasks

“Configuring database connections for DB2 databases” on page 42

DB2 requires enough memory for all possible JDBC connections to run statements without using swap space. If the system does not have sufficient memory, consider decreasing the maximum sizes for the JDBC Data Sources connection pools.

Chapter 4, “Tuning IBM WebSphere Application Server,” on page 9

Regardless of the installation type (single server or cluster), you can think of the IBM Tivoli Identity Manager server as two components: WebSphere Application Server (the J2EE application server running the application) and the IBM Tivoli Identity Manager application itself. You must tune both components.

Allocating processor usage

All IBM Tivoli Identity Manager components are processor-intensive so you must consider how to manage CPU for optimum performance.

Both IBM Tivoli Directory Server and IBM DB2 are multithreaded (and multiprocess in the case of DB2 applications) that show optimum performance on a multiprocessor server.

Even in a well-tuned environment the system bottlenecks might vary between the processor, memory and disk on the IBM Tivoli Identity Manager server, the directory server, and the database server. Deploying the IBM Tivoli Identity Manager server, the directory server, and the database server on separate servers might improve performance. If separate servers are not possible, put the database and directory server on a server with a high performance disk configuration. Use multiple disks and a high performance RAID configuration to provide fast read and write capacity.

Allocating disk space for storage

Each middleware component uses different amounts of disk space for various purposes.

- WebSphere Application Server and the IBM Tivoli Identity Manager application use disk space beyond their installation size because of log files. These log files include the `msg.log` and `trace.log` files. Adjust the number of archives and size of the `msg.log` and `trace.log` files in the `enRoleLogging.properties` file.
- IBM Tivoli Directory Server uses disk space from both the IBM Tivoli Directory Server process (log files like `ibmslapd.log`) and the IBM DB2 database. IBM Tivoli Directory Server uses system-managed space (SMS) table spaces so that the system can manage the amount of disk space used. You cannot specify the upper boundaries of SMS table spaces, so you monitor the amount of disk space used to prevent the drive from becoming full.
- The IBM Tivoli Identity Manager DB2 database uses directory-managed space (DMS) table spaces. These table spaces require manual allocation of disk space for the database. IBM Tivoli Identity Manager enables `autoresize` on these DMS table spaces so they can grow as needed.
- In addition to the table spaces for the database data, IBM DB2 uses disk space for the transaction logs. Configure the transaction logs to ensure enough disk space for the log files.
- IBM Tivoli Identity Manager creates Oracle data files so that they can grow as needed.

Related tasks

“Configuring transaction logs for the Tivoli Directory Server database” on page 69
DB2 keeps logs during transaction processing. During large transactions, the default log number and sizes might be too small and cause transaction rollbacks. Increase the size and number of log files available to DB2.

“Configuring transaction logs for DB2 databases” on page 48
DB2 keeps logs during transaction processing. During large transactions, the default log number and sizes might be too small and cause transaction rollbacks. Increase the size and number of log files to resolve this issue.

“Configuring table spaces for IBM DB2 databases” on page 44
IBM Tivoli Identity Manager uses a database managed space (DMS) table space to store data. This type of table space performs better than system managed space (SMS) table spaces, but you must preallocate disk space for the database to use. The tables spaces created by the installer have autoresize enabled and grow as needed.

“Configuring open cursors” on page 55
IBM Tivoli Identity Manager uses prepared statements through the WebSphere Application Server JDBC interface. Each prepared statement requires an open cursor in Oracle. If you receive an error message about too many open cursors, you can increase the maximum number of open cursors.

Chapter 4. Tuning IBM WebSphere Application Server

Regardless of the installation type (single server or cluster), you can think of the IBM Tivoli Identity Manager server as two components: WebSphere Application Server (the J2EE application server running the application) and the IBM Tivoli Identity Manager application itself. You must tune both components.

WebSphere Application Server provides a variety of settings for tuning the environment.

Related information



WebSphere Application Server product documentation

Adjusting the Java virtual machine size

IBM Tivoli Identity Manager, version 5.0 and 5.1, runs on 64-bit JVMs on supported platforms. Using a 64-bit JVM, you can allocate 2 GB or more of memory. You might need to allocate more memory for very large (more than 6 million accounts) reconciliations.

About this task

For cluster installations, IBM Tivoli Identity Manager uses two application servers per node: one for the application and one for the messaging (JMS) engine. All JVMs in this topic are used by the application, not the messaging engine. The JVM used by the messaging engine application server uses default values.

The IBM Tivoli Identity Manager regular installer sets the maximum JVM size to 1024 MB and the initial size to 512 MB. These values are adequate for most small and medium systems. If your server has available RAM, increase the maximum JVM size to 2048 MB for 64-bit JVMs or 1280 MB for 32-bit JVMs.

The IBM Tivoli Identity Manager Launchpad Single-server installer sets the maximum JVM size to 512 MB. This size is adequate for Proof Of Concept and demonstration environments. Increase the maximum JVM size if you have adequate memory.

Important: Setting the maximum heap memory value too high on 32-bit JVMs can cause memory allocation problems in Java. Problems occur when the memory limit is reached for 32-bit processes. Do not set JVM heap sizes on 32-bit Windows platforms higher than 1280 MB even if the system has more available memory.

The maximum heap size on 64-bit JVMs can be much higher than 2 GB. A larger value larger can result in long delays during full garbage collections. Do not set the maximum JVM size higher than necessary. Typically, 4 GB is an adequate maximum

Do not set the JVM heap size larger than the physical RAM. The WebSphere Application Server suffers significant performance degradation if the operating system swaps out the JVM to swap space. Setting the heap size larger than the physical RAM can cause slow user interface (UI) performance, transaction rollbacks, timeouts, and high disk utilization.

Use the following parameters to set the JVM heap size:

initial_jvm_heap_size

Specifies the initial size of the JVM heap in megabytes.


max_jvm_heap_size


Specifies the maximum size of the JVM heap in megabytes. Use 2048 MB for 64-bit JVMs and 1280 MB for 32-bit JVMs.


Procedure

1. Open the WebSphere Integrated Solutions Console.
2. Expand the **Servers** list.
3. Select **Application Servers**.
4. Select the application (not JMS) server to manage.
5. Expand the **Java and Process Management** list under the **Server Infrastructure** pane.
6. Select **Process Definition**.
7. Select **Java Virtual Machine** from the **Additional Properties** pane on the right.
8. Set the **Initial Heap Size** with *initial_jvm_heap_size*.
9. Set the **Maximum Heap Size** with *max_jvm_heap_size*.
10. Click **OK**.
11. Save the settings to the master configuration.
12. Repeat this procedure for each IBM Tivoli Identity Manager server.
13. Restart all application servers for the changes to take effect.

Related information

 [Understanding the IBM Java Garbage Collector](#)
Find out how objects are allocated in the Java heap for garbage collection.

 [Tuning Garbage Collection with the Sun 5.0 Java Virtual Machine](#)
See information on the general features of the Sun JVM garbage collection and tuning options to take the best advantage of those features.

 [Sun Java HotSpot VM Options](#)
See information on typical command-line options and environment variables that can affect the performance characteristics of the Java HotSpot Virtual Machine.

Configuring WebSphere Performance Monitoring Infrastructure

Disable or adjust Performance Monitoring Infrastructure to prevent performance degradation for the Administrative Console.

About this task

By default, WebSphere Application Server has the Performance Monitoring Infrastructure (PMI) enabled and set at the Basic level. At this level, URIRequestCount and URIServiceTime monitoring is enabled. Enabling both parameters causes performance problems when using the Administrative Console because unique URLs are generated. To prevent performance degradation, disable Performance Monitoring Infrastructure entirely or disable these specific flags.

Tip: Consider disabling Performance Monitoring Infrastructure entirely unless you are actively pursuing a performance-related problem.

Procedure

1. Open the WebSphere Integrated Solutions Console.
2. Expand the **Monitoring and Tuning** list.
3. Select **Performance Monitoring Infrastructure (PMI)**.
4. Select the server you want to manage.
5. Take one of the following actions:
 - To disable PMI entirely, clear **Enable Performance Monitoring Infrastructure (PMI)**.
 - To disable just the URIRequestCount and URIServiceTime counters:
 - a. Select **Custom**.
 - b. Select **Web Applications** from the tree listing.
 - c. Select the check box next to **URIConcurrentRequests**.
 - d. Select the check box next to **URIRequestCount**.
 - e. Select the check box next to **URIServiceTime**.
 - f. Click **Disable** at the top of the pane.
6. Save the settings to the master configuration.
7. Repeat this procedure for each IBM Tivoli Identity Manager application server.
8. Restart all application servers for the changes to take effect.

Configuring WebSphere JDBC connections

IBM Tivoli Identity Manager server uses JDBC connections from WebSphere Application Server to communicate with the database.

About this task

The JMS architecture in WebSphere, version 6.x, provides IBM Tivoli Identity Manager, versions 5.0 and 5.1, an additional JDBC Data Source for the JMS cluster members database connectivity. This Bus Data Source requires database connections in addition to those required for the application cluster members.

The number of connections from the application server to the database depends on the needs of the application. The maximum connection values are set independently on each application server. Typically, you do not need to increase the maximum connection values from the following default values.

- 30 (IBM Tivoli Identity Manager Bus Data Source)
- 30 (IBM Tivoli Identity Manager Bus Shared Data Source)
- 50 (IBM Tivoli Identity Manager Data Source)

Decrease the number of connections if the database cannot service all the concurrent requests due to resource limitations.

Use the following parameters to configure JDBC connections:

bus_data_source_size

Specifies the maximum JDBC pool size of the IBM Tivoli Identity Manager Bus Data Source. Initial value: 30.

bus_shared_data_source_size

Specifies the maximum JDBC pool size of the IBM Tivoli Identity Manager Bus Shared Data Source. This number is allocated by each cluster member. Initial value: 30.

data_source_size

Specifies the maximum JDBC pool size of the IBM Tivoli Identity Manager Data Source. This number is allocated by each cluster member. Initial value: 50.

Procedure

1. Open the WebSphere Integrated Solutions Console.
2. Expand **Resources**.
3. Expand the **JDBC** list.
4. Select **Data sources**.
5. Select the Data Source to update.
6. Select **Connection pool properties** from the **Additional Properties** pane.
7. Set the **Maximum connections** to the corresponding value for the Data Source selected.
8. Click **OK**.
9. Save the settings to the master configuration.
10. Repeat this procedure for each Data Source you want to change.
11. Restart all application servers for the changes to take effect.

Performance implications for Java 2 Security

Java 2 Security can degrade system performance on specific WebSphere Application Server versions.

WebSphere Application Server versions before version 6.1.0.9 had a significant performance penalty when Java 2 Security was enabled. For version 6.1.0.9 and later, you can enable Java 2 Security with minimal performance impact after setting some system properties. If you do not need Java 2 Security, disable it.

See APAR PK43270 for information about the system properties necessary for good performance.

Related information




PK43270: IMPROVE J2SE AND J2EE SECURITY PERFORMANCE.

Chapter 5. Tuning IBM HTTP Server

Small and medium configurations can typically use default configuration parameters for the IBM HTTP Server. Increase certain parameters for environments with a large number of concurrent users. The WebSphere Application Server uses the IBM HTTP Server as a front-end server in a single-server installation. It uses the IBM HTTP Server as a load balancer between nodes in a cluster installation.

Related information

 [IBM HTTP Server product documentation](#)

Optimizing IBM HTTP Server connections

You can set the number of connections that the IBM HTTP Server accepts at one time. The default value might be too small if the servers experience a large number of concurrent users.

About this task

The IBM HTTP Server supports the HTTP/1.1 KeepAlive request that allows a client to make multiple HTTP requests through a single persistent connection. A single connection can accept only a limited number of KeepAlive requests. After reaching this limit is, the connection closes, and another connection must be established. The default value might be too small for some external provisioning processes, such as a Java Naming and Directory Interface (JNDI) feed.

Use the following parameters to optimize server connections:

ibmhttp_home

Specifies the home directory of the IBM HTTP Server, such as /usr/IBMHttpServer.

max_connections

Specifies the maximum number of connections that can be made to the HTTP server at one time. Set this parameter to the maximum number of concurrent users you expect on your system.

max_keepalive

Specifies the maximum number of requests for a single connection.

Tip: The MaxClients parameter on Windows is called ThreadsPerChild. You might need to adjust the ServerLimit, ThreadLimit, and ThreadsPerChild parameters on UNIX systems when adjusting the MaxClients parameter. See the IBM HTTP Server documentation for more information.

Procedure

1. Edit the *ibmhttp_home/conf/httpd.conf* file and update the following entries:
MaxClients *max_connections*
MaxKeepAliveRequests *max_keepalive*
2. Stop and restart the IBM HTTP Server for these changes to take effect.

Enabling content compression for the IBM HTTP Server

The IBM HTTP Server shipped with WebSphere Application Server includes the `mod_deflate` plug-in. Use this plug-in to compress pages before sending them to the client.

About this task

Typically, the `mod_deflate` plug-in yields better results for Administrative Console users, particularly if they are not on the same LAN as the IBM Tivoli Identity Manager server.

Enabling the `mod_deflate` plugin for the Self-Service interface is not necessary due to the smaller size of returned pages. Enabling it can increase page response times.

Use these parameters to compress pages:

ibmhttp_home

Specifies the home directory of the IBM HTTP Server, such as `/usr/IBMHttpServer`.

itim_console_location

Specifies the base URL of the IBM Tivoli Identity Manager Console application, such as `/itim/console`.

Procedure

1. Edit the `ibmhttp_home/conf/httpd.conf` file and add the following lines:

Note: The `<Location itim_console_location>` stanza must include the `itim_console_location` value.

```
LoadModule deflate_module modules/mod_deflate.so

# Compress content for ITIM Administrative Console interface.
#
# Requires modules:
#   LoadModule deflate_module modules/mod_deflate.so
<Location itim_console_location>
  # All ITIM supported web browsers correctly declare their support for
  # compressed content via Accept-Encoding so we don't Vary the compression
  # based on User-Agent.

  # Insert filter for compression
  SetOutputFilter DEFLATE

  # Don't compress images
  SetEnvIfNoCase Request_URI \.(?:gif|jpe?g|png|ico)$ no-gzip
</Location>
```

2. Stop and restart the IBM HTTP Server for these changes to take effect.

Improving the caching of static content served from the IBM HTTP Server

You can use the IBM HTTP Server to improve the caching of static content both in the browser and in any intermediate caching proxies.

About this task

You can improve the use of caches for static content between the end-user and the WebSphere Application Server instance. Adjust the Expire and Vary headings from the IBM HTTP Server.

If you set the Expire header to the distant future, caches can store and serve the unchanging static content without refreshing it from the WebSphere Application Server. Removing the Vary header from images instructs caching proxies to serve the images from the cache, irrespective of the browser User-Agent. Removing the Vary header increases cache hits and improves overall interface performance.

Use these parameters in the following procedure:

ibmhttp_home

Specifies the home directory of the IBM HTTP Server, such as /usr/IBMHttpServer.

itim_location

Specifies the base URL of the IBM Tivoli Identity Manager application, such as /itim.

Procedure

1. Edit the *ibmhttp_home/conf/httpd.conf* file and add the following lines:

Note: The <Location *itim_location*> stanza must include the *itim_location* value.

```
LoadModule headers_module modules/mod_headers.so
LoadModule expires_module modules/mod_expires.so
```

```
# Ensure static content is cached by the browser and intermediate proxies
# as efficiently as possible. This applies both to the Administrative Console
# as well as the Self-Service interface.
# Static content includes images (gif/jpeg/png/ico), stylesheets (css) and
# Javascript files (js).
```

```
#
# Requires modules:
#   LoadModule headers_module modules/mod_headers.so
#   LoadModule expires_module modules/mod_expires.so
```

```
<Location itim_location>
# Set the Expires header for static content to +1 month
ExpiresActive On
ExpiresByType image/gif "access plus 1 month"
ExpiresByType image/jpeg "access plus 1 month"
ExpiresByType image/png "access plus 1 month"
ExpiresByType image/x-icon "access plus 1 month"
ExpiresByType text/css "access plus 1 month"
ExpiresByType application/x-javascript "access plus 1 month"
```

```
# Don't Vary image content at all. This allows caching proxies to cache
# the images once and serve it to all browsers. Note we can't include
# css/js content in case some browsers request compressed content and
# some don't. The mod_deflate plugin will automatically set a Vary
# header against Accept-Encoding, we just need to not override it.
# Caching proxies will still cache css/js content but will cache
# two or more copies and serve them accordingly.
```

```
SetEnvIfNoCase Request_URI \.(?:gif|jpe?g|png|ico)$ dont-vary
Header unset Vary env=dont-vary
```

```
</Location>
```

2. Stop and restart the IBM HTTP Server for these changes to take effect.

Edge Side Include caching

The IBM HTTP plug-in interfaces with single-server WebSphere environments and balances GUI requests in clustered WebSphere environments. The plug-in has built-in support for Edge Side Include (ESI), which does page- and fragment-level caching.

Edge Side Include does more than caching static content, but IBM Tivoli Identity Manager primarily uses it to cache images, Javascript, and CSS files.

The HTTP plug-in enables Edge Side Include by default with a cache size of 1024 KB and a cache timeout value of 300 seconds (5 minutes).

Configuring the Edge Side Include cache size

If both the Administrative Console and the Self-Service GUI are heavily used, increase the size of the Edge Side Include cache.

About this task

The approximate size of the static content from the Administrative Console is 825 KB and for the Self-Service GUI is 550 KB. Static content includes images, Javascript, and CSS. If the bulk of traffic is from one or the other GUI, the default 1024 KB cache is adequate to completely cache the static content. If you have both kinds of traffic, you can increase the cache size to 1536 KB or 2048 KB. An example of both kinds of traffic is users servicing their own requests for password changes in addition to help desk personnel doing account maintenance.

Important: Use caution when increasing this value. There is one ESI cache per HTTP process. The total memory used by the ESI cache is `cache_size * num_HTTP_processes`.

ibmhttp_home

Specifies the home directory of the IBM HTTP Server, such as `/usr/IBMHttpServer`.

pluginxml_file

Specifies the name of the configuration file for the HTTP plug-in. You can find the name in the `WebSpherePluginConfig` parameter in the IBM HTTP Server configuration file. The configuration file is at `ibmhttp_home/conf/httpd.conf`.

cache_size

Specifies the size (in kilobytes) of the ESI cache. If you use both the Administrative Console and the Self-Service GUI, set the value to 2048. Default value: 1024.

Procedure

1. Edit *pluginxml_file*.
2. Locate the line for the `ESIMaxCacheSize` value. For example:

```
<Property Name="ESIMaxCacheSize" Value="1024"/>
```
3. If the `ESIMaxCacheSize` line does not exist, add it.
4. Set the value to *cache_size*:

```
<Property Name="ESIMaxCacheSize" Value="cache_size"/>
```
5. Save and exit the file.
6. Stop and restart the IBM HTTP Server for these changes to take effect.

Configuring the Edge Side Include cache timeout

The Edge Side Include cache timeout controls how long an entry can exist in the cache before it expires. Adjusting length of this timeout might improve performance under load.

About this task

By default entries are valid in the Edge Side Include cache for 5 minutes. After 5 minutes, the data expires, and a subsequent request is passed back to WebSphere. A busy environment in which new users access IBM Tivoli Identity Manager every few minutes results in requests to WebSphere every 5 minutes.

IBM Tivoli Identity Manager static content does not change every 5 minutes. You can change the timeout value to higher value, like one hour. Add the following parameter to the JVM command line for the application servers running IBM Tivoli Identity Manager.

cache_timeout

Specifies the Edge Side Include cache timeout value in seconds. To set it for one hour, use 3600. Default value: 300.

Procedure

1. Open the WebSphere Integrated Solutions Console.
2. Expand the **Servers** list.
3. Select **Application Servers**.
4. Select the application server you want to manage. Do not select the JMS server.
5. Expand the **Java and Process Management** list under the **Server Infrastructure** pane.
6. Select **Process Definition**
7. Select **Java Virtual Machine** from the **Additional Properties** pane.
8. Add the following line to the **Generic JVM arguments**:
`-Dcom.ibm.servlet.file.esi.timeOut=cache_timeout`
9. Click **OK**.
10. Save the settings to the master configuration.
11. Repeat this procedure for each IBM Tivoli Identity Manager server.
12. Restart all application servers for the changes to take effect.

Chapter 6. Tuning IBM Tivoli Identity Manager

Tune IBM Tivoli Identity Manager by adjusting values in configuration files and through the Administrative Console.

IBM Tivoli Identity Manager includes several configuration files for tuning various parts of the application performance. These files are in the `data/` directory in the IBM Tivoli Identity Manager home directory.

Related information

 [IBM Tivoli Identity Manager product documentation](#)

 [IBM Tivoli Identity Manager wiki](#)

Configuring LDAP connection pooling

IBM Tivoli Identity Manager can reuse LDAP connections to the directory server to decrease the performance impact of establishing new connections.

About this task

IBM Tivoli Identity Manager can use LDAP connection pooling to communicate with the LDAP data store. A single connection consists of a bind; an operation, such as a search, add, modify, or delete; and an unbind. Connection pooling improves performance by allowing multiple LDAP operations to reuse a single connection with the same bind credentials. The single connection eliminates the performance impact of bind and unbind.

You can enable connection pooling for non-SSL connections (plain), SSL connections, or both. By default IBM Tivoli Identity Manager is configured to pool only plain connections. Establishing SSL connections can cause a significant performance impact. For environments using SSL to connect to the directory server, configure the server to pool the SSL connections.

Procedure

1. Edit the `enRole.properties` file and change the following property:
`enrole.connectionpool.protocol=plain ssl`
2. In a clustered environment, edit the `enRole.properties` file on each node.
3. Restart the IBM Tivoli Identity Manager application for this value to take effect.

Configuring list controls

The `ui.properties` file has several parameters. These parameters control how many entries, such as viewing the people in an organizational unit, and how many pages are in a list.

About this task

Setting the following values too high can result in Java `OutOfMemory` errors due to heap fragmentation.

itim_home

Specifies the home directory for IBM Tivoli Identity Manager, such as /opt/IBM/itim

page_size

Specifies the number of entries to show on a page. Default value: 50.

page_link_max

Specifies the number of pages a user can access for a single search. Default value: 10.

max_search_results

Specifies the maximum number of results to return from a search. Increasing this value can result in heap fragmentation issues. Always make this value equal to or greater than (page_size * page_link_max). If (page_size * page_link_max) is larger than 1000, decrease one of the two parameters until the product is less than 1000. Default value: 1000.

Change the values on all nodes in a clustered environment.

Procedure

1. Edit the *itim_home/data/ui.properties* file and change the following properties:
enrole.ui.pageSize=*page_size*
enrole.ui.pageLinkMax=*page_link_max*
enrole.ui.maxSearchResults=*max_search_results*
2. In a clustered environment, edit the *itim_home/data/ui.properties* file on each node.
3. Restart the IBM Tivoli Identity Manager application for these values to take effect.

Configuring report data synchronization

You must synchronize data before you can generate reports against IBM Tivoli Identity Manager. Synchronizing report data pulls configuration and user information from the configured LDAP and uses it to populate the database.

About this task

This task synchronizes the data and the related Access Control Information (ACIs).

If only IBM Tivoli Identity Manager administrators create reports, populating the ACI data when synchronizing report data is not required. ACI data is not applied to reports generated by administrators. Similarly, if you use IBM Tivoli Common Reporting to generate reports, populating the ACI data when synchronizing report data is not required. IBM Tivoli Common Reporting does not enforce ACIs during report generation.

Disabling ACI synchronization can improve report data synchronization performance by an order of magnitude. Improvement depends on the number and complexity of the configured ACIs and the structure of your organizational tree. The `availableForNonAdministrators` parameter in the `adhocreporting.properties` file controls enabling and disabling ACI synchronization. Setting the parameter to true synchronizes ACIs.

After disabling report data synchronization with ACI data, non-administrators cannot generate reports from IBM Tivoli Identity Manager.

Disabling ACI synchronization for IBM Tivoli Identity Manager, version 5.0, requires IF23 or later.

IBM Tivoli Identity Manager, version 5.0, IF17 reduced the completion time for ACI data synchronization by half for most systems. Make sure that you are at this level or higher if the population of ACIs data is necessary.

synchronize_ACIs

Specify true to enable ACI synchronization and false to disable it. Default value: true. These values are case-sensitive.

Procedure

1. Edit the `adhocreporting.properties` file and change the following property:
`availableForNonAdministrators=synchronize_ACIs`
2. In a clustered environment, edit the `adhocreporting.properties` file on each node.
3. Restart the IBM Tivoli Identity Manager application for this change to take effect.

Configuring report batch sizes

Adjust the CSV report batch size to improve report scalability.

About this task

Generating large CSV reports can require adjusting values in the `adhocreporting.properties` file to avoid Java `OutOfMemory` errors for large reports.

batch_size

Specifies the number of items requested at a time from the reporting tables. If you do not set a value or comment out the line, all items are fetched. Try setting this value to 10000.

Procedure

1. Edit the `adhocreporting.properties` file and change the following property:
`reportBatchSize=batch_size`
2. Verify that the line is not commented out.
3. In a clustered environment, edit the `adhocreporting.properties` file on each node.
4. Restart the IBM Tivoli Identity Manager application for these values to take effect.

Configuring e-mail notifications

Configuring the system to send e-mail notifications when no e-mail addresses exist can slow down provisioning actions.

About this task

When you configure the system to send an e-mail for an action, the software checks if the user has an e-mail address on the person record. For example, creating an account is an action. If the software finds no e-mail address, it checks the manager of the user. If the manager does not have an e-mail address or the user does not have a manager, the software sends an e-mail to the system administrators.

For large populations, the LDAP search for system administrators can take a while and might slow down provisioning actions. Ensure that user records have e-mail addresses if you use e-mail notifications. If you do not want e-mail notifications, disable them to avoid the lookup.

Procedure

1. Access IBM Tivoli Identity Manager as a system administrator.
2. Expand **Configure System**.
3. Select **Workflow Notification Properties**.
4. In **E-mail Notification Templates**, locate the notification you want to disable.
5. In the **Status** column, hover and select **Disable**.
6. Click **OK**.

Results

The change takes effect immediately.

Using the recycle bin

When you enable the recycle bin and then delete objects from IBM Tivoli Identity Manager, the software moves them to the recycle bin.

When you delete objects from IBM Tivoli Identity Manager, they are moved to the recycle bin in the LDAP directory. Deleting objects does not remove them from the underlying directory server. You can delete objects either from the graphical user interface or the application programming interface. Examples of objects include people, accounts, roles, and provisioning policies.

Using the recycle bin can have negative performance impacts. . You might use this feature for a business policy that prohibits reusing a deleted user ID. You can, however, use custom code to enforce the policy and then disable the recycle bin.

The recycle bin is implemented as the following LDAP container:

```
ou=recycleBin, ou=itim, ou=<tenant>, <suffix>
```

When you delete objects, the following process occurs:

1. The software moves the LDAP entries under this DN after you delete them.
2. The software sets the `erIsDeleted` attribute to `Y`.
3. The `Y` value tells IBM Tivoli Identity Manager not to display these objects to users or act on them.

Important: The default behavior of the recycle bin changed with IBM Tivoli Identity Manager, version 5.0. Previously, the recycle bin was enabled by default. With version 5.0 and later, it is disabled by default. If you are upgrading from version 4.6, disable the recycle bin unless your environment requires it.

Disabling the recycle bin

Disable the recycle bin to avoid performance degradation.

About this task

Disable the recycle bin to avoid performance degradation under the following circumstances:

- You upgraded from a previous version of IBM Tivoli Identity Manager. The default behavior of the recycle bin changed with IBM Tivoli Identity Manager, version 5.0. Previously, the recycle bin was enabled by default. With version 5.0 and later, it is disabled by default.
- Your business policy does not prohibit reusing a deleted user ID. You can, however, use custom code to enforce the policy and then disable the recycle bin.

Procedure

1. Edit `enRole.properties`.
2. Set the property `enrole.recyclebin.enable` to `false`. If the `enrole.recyclebin.enable` property does not exist, add it to the end of the file with the value of `false`.
3. Stop all IBM Tivoli Identity Manager nodes.
4. Empty the recycle bin.
5. Restart all IBM Tivoli Identity Manager nodes.

Emptying the recycle bin

Keep the size of the recycle bin as small as possible for optimum performance.

About this task

Use the `ldapClean` script that is included with IBM Tivoli Identity Manager to remove items from the recycle bin. This script does not delete workflow records that are in the recycle bin but that are still used by outstanding activities. Use the following parameters for this task:

itim_home

Specifies the home directory for IBM Tivoli Identity Manager, such as `/opt/IBM/itim`

script_dir

Specifies the location of the `ldapClean` script. It is in `itim_home/bin/os` where *os* is `unix` for UNIX or Linux systems or `win` for Windows systems.

Procedure

1. Edit `enRole.properties`.
2. Set the property `enrole.ldapservers.agelimit` to `-1`.
3. Run the `ldapClean` script:
`script_dir/ldapClean`

What to do next

For an IBM Tivoli Directory Server, run `runstats` to instruct IBM DB2 pick up the changes.

Related tasks

“Updating Tivoli Directory Server database statistics” on page 74
DB2 requires information about the number of rows in the tables and what indexes are available so that it can efficiently fulfill queries. If Tivoli Directory Server database is running DB2, version 9, you can set RUNSTATS to run automatically. Version 9 is the default for Tivoli Directory Server, version 6.1. RUNSTATS eliminates the need for running it manually.

Working with reconciliations

Reconciliations are resource-intensive operations. Reconciliations for services with a large account population can affect performance.

You can improve reconciliation performance by limiting the number of attributes returned by the adapter and processed by IBM Tivoli Identity Manager.

Large reconciliations can exceed the default `Max Duration`, but you can increase the value. Larger reconciliations can also benefit from using paged searches.

Note: The default value for the `enrole.reconciliation.accountcachesize` parameter in `enRole.properties` file has been optimized. Do not change the value of this parameter unless instructed by IBM Support. Increasing this value can decrease reconciliation performance.

Related tasks

“Configuring paged searches” on page 26
IBM Tivoli Identity Manager, version 5.0 and later, incorporates LDAP paged searches to alleviate `JavaOutOfMemory` errors in large environments.

Limiting attributes returned from the adapter

Limiting attributes returned from the adapter can reduce the amount of work required by the adapter. It can also reduce the amount of data sent to IBM Tivoli Identity Manager.

Some adapters (such as the adapter for Microsoft Active Directory) can limit the attributes that are returned to the IBM Tivoli Identity Manager server during reconciliations. Consult the adapter documentation for information specific to that adapter.

Related tasks

“Configuring attributes returned during an Active Directory reconciliation” on page 31
Removing calculated attributes returned from an Active Directory reconciliation can improve performance.

Reducing policy enforcements

You can reduce the number of policy enforcement by limiting the attributes being evaluated during the reconciliation, You can also ensure that provisioning policies do not specify mandatory enforcement for attributes that are not reconciled.

The reconciliation process updates any changed attributes in the IBM Tivoli Identity Manager directory server. Before this update takes place, the process evaluates the new value against the provisioning policy that governs the account. The validation ensures that the policy permits the change. If not, a policy enforcement is triggered. Any change to the account triggers the policy evaluation for that account regardless if the change invalidates the policy.

Limiting attributes evaluated during reconciliation

To reduce the number of policy evaluations, limit the attributes that are evaluated during reconciliation.

About this task

Some endpoints (such as Microsoft Active Directory) contain attributes that change frequently but are seldom used to enforce policy. An example of this type of attribute is last logon time. If these attributes are required, consider

- Setting up a second reconciliation to reconcile them on a more infrequent schedule.
- Remove them from the more frequently running reconciliations.
- If possible, reconcile only those attributes that are required for policy evaluation.

Use the following parameter:

excluded_attributes

Specifies the list of attributes that are returned from the adapter to exclude from processing in IBM Tivoli Identity Manager. Ideally, you exclude all attributes except those that are required for policy evaluation.

Procedure

1. Access IBM Tivoli Identity Manager as a user with sufficient privileges to edit the service you want to reconcile.
2. Select **Manage Services**.
3. Search for the service you want to reconcile.
4. Select **Set up reconciliation**.
5. Select the reconciliation schedule to modify.
6. Select the **Query** tab.
7. Select all *excluded_attributes*.
8. Click **Remove**.
9. Click **OK**.

Optimizing entitlement enforcement

To reduce unnecessary policy enforcement, set entitlement parameter enforcement to mandatory only for attributes that are returned during reconciliation for that service type.

Some attributes can be provisioned to a service, but they are not included during a reconcile for that service type. If mandatory enforcement is configured for not-returned attributes, Tivoli Identify Manger updates their value on the endpoint during a reconciliation whether or not the value actually changed. This process causes unnecessary provisioning actions on the endpoint and increases the load on IBM Tivoli Identity Manager.

See the individual adapter documentation for information on which attributes are returned during reconciliation.

Configuring reconciliation threads

Each reconciliation process creates additional threads to process the accounts returned from the adapter. Decreasing the number of threads can decrease resource usage while maintaining reconciliation performance.

About this task

The `enrole.reconciliation.threadcount` parameter controls the number of reconciliation threads that are started for a single reconciliation process. The default number is 8. A single thread can process accounts faster than most adapters can return them. Reducing the number of threads decreases the number of idle threads and the JVM resources required to create, track, and destroy them. For adapters that return accounts faster than a single thread can process them, decreasing the number of threads can decrease CPU utilization caused by thread contention while maintaining the reconciliation throughput. Use the following variable when specifying the number of threads.

recon_threads

Specifies the number of threads a single reconciliation process starts.

Default value: 8. Typical value: 2 to 4.

Procedure

1. Edit the `enRole.properties` file and change the following property:
`enrole.reconciliation.threadcount=recon_threads`
2. In a clustered environment, edit the `enRole.properties` file on each node.
3. Restart the IBM Tivoli Identity Manager application for these values to take effect.

Configuring the maximum duration of a reconciliation

Large reconciliations sometimes exceed the default maximum duration specified in the reconciliation schedule. When this limit is reached, the reconciliation halts.

About this task

Increase the limit to allow longer-running reconciliations to complete using the following variable:

max_duration

Specifies the number for minutes that the reconciliation runs. To calculate this value, do an initial run with a very large duration and measure the time. Consider setting the maximum duration to 10% above this time.

Default value: 600.

Procedure

1. Access IBM Tivoli Identity Manager as a user with sufficient privileges to edit the you want to reconcile service.
2. Select **Manage Services**.
3. Search for the service you want to reconcile.
4. Select **Set up reconciliation**.
5. Select the reconciliation schedule you want to modify.
6. Set the **Maximum duration** to *max_duration*.
7. Click **OK**.

Configuring paged searches

IBM Tivoli Identity Manager, version 5.0 and later, incorporates LDAP paged searches to alleviate `JavaOutOfMemory` errors in large environments.

About this task

Note: Paged searches are useful only for directory servers that support them, such as the IBM Tivoli Directory Server. The Sun ONE Directory Server does not support paged searches; enabling paging has no effect.

Paged searches are used in areas that potentially result in large data sets, including:

- Reconciliations
- Provisioning policy creation, modification, deletion, and preview
- Service enforcement changes
- Dynamic role creation, modification, and deletion
- Report data synchronization

Paged searches are disabled by default. They place an additional load onto the LDAP server; some LDAP servers have a limit on the number of concurrent paged searches. When enabling this parameter, configure the underlying LDAP server to accept as least as many paged search requests as the concurrent activities from the from the previous list.

Use paged searches for the following large (500,000 or more) data sets:

- Accounts
- People in an organizational tree node
- People in a single role

Tip: A related parameter governs the enabling of server-side sorting. Do not enable server-side sorting. See “Enabling server-side sorting” on page 28.

Use the following properties to set up paged searches:

paging_enabled

Enables LDAP paging for searches that support it. Valid values: true or false. Default value: false.

paging_size

Specifies the size of the paging request to the LDAP server. If you set this value too high, the LDAP server might ignore the paging request. Do not set this value larger than 128. Default value: 128

Procedure

1. Access the `enRole.properties` file
2. Change or add the following properties:
`enrole.search.paging.enable=paging_enabled`
`enrole.search.paging.pagesize=paging_size`
3. In a clustered environment, edit the `enRole.properties` file on each node.
4. Restart the IBM Tivoli Identity Manager application for these values to take effect.

Related concepts

“Enabling server-side sorting”

Enabling the server-side sorting property can have a negative impact when viewing large organizational units. Typically, this option remains disabled.

Enabling server-side sorting

Enabling the server-side sorting property can have a negative impact when viewing large organizational units. Typically, this option remains disabled.

When retrieving lists of objects from LDAP to display in the interface, IBM Tivoli Identity Manager sorts the results before presenting them to the user. When you enable paged searches, IBM Tivoli Identity Manager also supports the LDAP server sorting the results. Enabling server-side sorting (through the `enrole.search.sss.enabled` property in `enRole.properties`) can have a negative impact when viewing large organizational units. Do not enable this option for most environments.

Related tasks

“Configuring paged searches” on page 26

IBM Tivoli Identity Manager, version 5.0 and later, incorporates LDAP paged searches to alleviate `JavaOutOfMemory` errors in large environments.

Configuring the ACI cache

Adjusting the time between ACI refreshes and the size of the ACI cache can improve performance in some cases.

About this task

The following properties control the ACI cache. They can improve performance or reduce memory requirements.

refresh_interval

Specifies the number of minutes between ACI cache refreshes. Increasing this value can result in better ACI performance, but ACI changes might take longer to be enforced. Default value: 5.

user_cache_size

Specifies the maximum number of ACI evaluation results to cache per user. Increasing this value can result in better performance for systems with many ACIs. Increasing the value requires more memory from the JVM. Default value: 50.

cache_size

Specifies the maximum size of the ACI cache. Increasing this value can result in better performance. Increasing the value requires more memory from the JVM. Default value: 1000.

Procedure

1. Edit the `enRole.properties` file and change or add the following properties:
`enrole.accesscontrollist.refreshInterval=refresh_interval`
`enrole.userACICache.maxSize=user_cache_size`
`enrole.accesscontrollist.maxSize=cache_size`
2. In a clustered environment, edit the `enRole.properties` file on each node.
3. Restart the IBM Tivoli Identity Manager application for these values to take effect.

Controlling the size of the database

To maintain optimum performance, use the DBPurge utility included with IBM Tivoli Identity Manager to automate removing entries over a certain age from the database.

About this task

The IBM Tivoli Identity Manager database stores data for

- In-progress system transactions.
- Completed system transactions.
- Auditing information.

The database has no growth boundaries. For best performance, keep as little data as necessary in the live database. Use database backups for older data sets.

The DBPurge utility works with all supported databases. It processes all time-based data, including transaction, audit, and reconciliation records. Use the following variables with this utility:

itim_home

Specifies the home directory for IBM Tivoli Identity Manager, such as `/opt/IBM/itim`.

os_type

Specifies the operating system time of the IBM Tivoli Identity Manager server. Use either `win` (for Windows) or `unix` (for UNIX).

days_to_retain

Specifies the number of days of data to retain records. The utility removes any records in the database older than this value.

purge_trans

Specifies whether to remove transactional data older than *days_to_retain*. Default value: `true`.

purge_audit

Specifies whether to remove the audit data older than *days_to_retain* during the purge. Default value: `true`.

purge_recon

Specifies whether to remove reconciliation data older than *days_to_retain* during the purge. Default value: `true`.

Procedure

Run the following command on one line:

```
itim_home/bin/os_type/DBpurge  
-age days_to_retain  
-workflow purge_trans  
-audit purge_audit  
-recon purge_recon
```

Chapter 7. IBM Tivoli Identity Manager adapters

Sometimes you must tune IBM Tivoli Identity Manager adapters when doing large provisioning changes or reconciliations.

This information supplements, rather than supersedes, the documentation provided for each adapter.

Tuning the Microsoft Active Directory adapter

Changing parameters on the Microsoft Active Directory adapter can improve reconciliation and provisioning performance.

Related information



Tivoli Identity Manager V5.1 Active Directory Adapter Installation and Configuration Guide (PDF)



Tivoli Identity Manager V5.0 Active Directory Adapter Installation and Configuration Guide (PDF)

Configuring attributes returned during an Active Directory reconciliation

Removing calculated attributes returned from an Active Directory reconciliation can improve performance.

About this task

During a reconciliation, the Microsoft Active Directory adapter returns attributes to IBM Tivoli Identity Manager that are not directly retrieved from Active Directory. These attributes are calculated from other Windows sources. Querying these external sources can slow down Active Directory reconciliations. You can disable the query if these attributes are not needed.

Working with Windows Terminal Services attributes can also slow down provisioning and reconciliation.

To disable calculated attributes, review and adjust the keys in the adapter registry.

Procedure

- Set `ReconHomeDirSecurity` and `ReconMailboxPermissions` to `FALSE` if the Home Directory Security and Mailbox Permissions attributes are not required.

Retrieving this information requires looking up the appropriate access control entry, which can slow down reconciliation. Disabling these attributes improves throughput.

- Set `ReconPrimaryGroup` to `FALSE`.

Disabling this attribute can significantly improve Active Directory reconciliation performance.

- Set `WtsEnabled` to `FALSE`.

This key controls adapter access to Windows Terminal Services attributes. If you set this value to `TRUE`, the adapter can provision and reconcile the attributes. If

you set this key to FALSE, the adapter cannot provision the attributes if requested or return them during reconciliation. Default value: FALSE.

- Set `WtsDisableSearch` to TRUE.

This key only applies if the `WtsEnabled` key is set to TRUE. It controls whether the adapter returns Windows Terminal Services attributes during a reconciliation, which is a search from an adapter perspective. If set to TRUE, a reconciliation does not return the attributes, but it updates the attributes in account provisions. If this key is set to FALSE, the reconciliation returns the attributes. Default value: TRUE.

Related information



Tivoli Identity Manager V5.1 Active Directory Adapter Installation and Configuration Guide (PDF)



Tivoli Identity Manager V5.0 Active Directory Adapter Installation and Configuration Guide (PDF)

Configuring the number of threads for the Active Directory adapter

Increasing the number of threads allocated to provisioning actions can increase the provisioning throughput of the Microsoft Active Directory adapter.

About this task

By default, each provisioning action is configured to use three threads. Doubling the number of threads from 3 to 6 can improve the account provisioning throughput by approximately 100%. Increasing these values too much can result in directory service is busy error messages in the adapter log. These messages indicate that Active Directory cannot accept the configured number of concurrent threads from the adapter.

You can specify the number of threads that are dedicated for each provisioning action.

Procedure

In the adapter configuration **Advanced Settings** menu, change the appropriate parameter or parameters: `ADD`, `MODIFY`, `DELETE`, or `SEARCH`

What to do next

If you receive directory service is busy messages, decrease the number of threads until the error goes away.

Related information



Tivoli Identity Manager V5.1 Active Directory Adapter Installation and Configuration Guide (PDF)



Tivoli Identity Manager V5.0 Active Directory Adapter Installation and Configuration Guide (PDF)

Tuning the LDAP adapter

Reconciling large LDAP directories using the IBM Tivoli Directory Integrator-based LDAP adapter might require enabling the LDAP paging control on the adapter. It might also require increasing the amount of memory available to the IBM Tivoli Directory Integrator JVM.

About this task

Servers that support the LDAP paging control include IBM Tivoli Directory Server and Microsoft Active Directory. The Sun ONE Directory Server does not support the paging control.

Procedure

- If you use the LDAP adapter with a server that supports the paging control and enable it, the adapter can fetch data from the LDAP server in distinct chunks. See *Directory Integrator-Based LDAP Adapter Installation and Configuration Guide* for more information about enabling the paging control.
- If you use a server that does not support the LDAP paging control:
 - Increase the size of the IBM Tivoli Directory Integrator JVM so that larger reconciliations can process successfully. See *IBM Tivoli Directory Integrator User's Guide* for information about increasing the JVM size.
 - Use reconciliation filters to pull back only a subset of the entries at one time to decrease the amount of JVM memory required.

Related information



Tivoli Identity Manager V5.1 Directory Integrator-Based LDAP Adapter Installation and Configuration Guide (PDF)



Tivoli Identity Manager V5.0 Directory Integrator-Based LDAP Adapter Installation and Configuration Guide (PDF)

Tuning the RACF adapter

You can adjust RACF adapter reconciliation performance with the `PDU_ENTRY_LIMIT` environment variable.

About this task

By default this value is not set and reverts to 3000. This value provides good reconciliation performance for most IBM Tivoli Identity Manager environments.

Procedure

- Sometimes the IBM Tivoli Identity Manager server running the reconciliation cannot process entries as quickly as they are streamed back from the RACF adapter. The RACF adapter can continue to use up memory as it buffers the requests being sent. This process can negatively affect other workloads on the

computer due to paging. If this problem occurs, set the size of the PDU_ENTRY_LIMIT environment variable to a lower number, such as 1000 or 500.

- If the IBM Tivoli Identity Manager server can process data faster than the RACF adapter can stream back the accounts, increase the PDU_ENTRY_LIMIT environment variable to decrease total reconciliation time. For example, you might use 4000 or 5000.

Related information



Tivoli Identity Manager V5.1 RACF Adapter Installation and Configuration Guide (PDF)



Tivoli Identity Manager V5.0 RACF Adapter Installation and Configuration Guide (PDF)

Chapter 8. Tuning Tivoli Directory Integrator

Tivoli Directory Integrator is often used in a IBM Tivoli Identity Manager environment both for adapters shipped with the product and for creating custom adapters.

Related information



IBM Tivoli Directory Integrator 7.0: Users Guide



IBM Tivoli Directory Integrator 6.1.1: Users Guide (PDF)

Configuring logging levels for Tivoli Directory Integrator

The default logging level for Tivoli Directory Integrator is INFO. You can change the logging level to WARN or ERROR to prevent security and administrative issues in production environments.

About this task

The default logging level for IBM Tivoli Directory Integrator is INFO. At the INFO level, the software writes informational messages to the log file. For production systems, this setting has potential security and administrative issues.

Security

At the INFO level, entity attributes and their corresponding values are printed to the log. Password values are not included in the log. IDs and other attributes are included, which might create privacy issues.

Administrative

In busy environments, the log file can grow quickly and might fill up the disk.

You can change the logging level to WARN or ERROR using the following variables:

itim_solution_dir

Specifies the name of the IBM Tivoli Identity Manager solution directory. It is located underneath the home directory for Tivoli Directory Integrator.

log_level

Specifies the logging level to use, such as WARN or ERROR. Default value: INFO.

Procedure

1. Stop IBM Tivoli Directory Integrator.
2. Edit *itim_solution_dir/etc/log4j.properties*.
3. Change *log4j.rootCategory* to the level you want.
`log4j.rootCategory=log_level`
4. Restart IBM Tivoli Directory Integrator for these changes to take effect.

Related information



Tivoli Identity Manager V5.1 Directory Integrator RMI Dispatcher Installation and Configuration Guide (PDF)



Tivoli Identity Manager V5.0 Directory Integrator RMI Dispatcher Installation and Configuration Guide (PDF)

Using the DSML connector with Tivoli Directory Integrator

You can use the DSML connector to create custom agents for returning information to IBM Tivoli Identity Manager.

The DSML connector can return information as a single unit or in smaller units using the chunked encoding mechanism. Each method has advantages and disadvantages.

Chunked encoding

- Applies to all responses to IBM Tivoli Identity Manager, although it is most relevant for reconciliations.
- Prevents the DSML file from being created in-memory in Tivoli Directory Integrator.
- Begins processing IBM Tivoli Identity Manager account reconciliations sooner. The adapter starts streaming accounts back to the server after collecting enough accounts to populate the first chunk.

Without chunked encoding, the DSML file is created in-memory in the IBM Tivoli Directory Integrator. Large reconciliations can cause `OutOfMemory` errors.

Enable chunked encoding in Tivoli Directory Integrator for all DSML-feed-based adapters.

Tuning the RMI Dispatcher

The IBM Tivoli Identity Manager RMI Dispatcher services requests for RMI-based adapters in Tivoli Directory Integrator.

Configuring timeouts for large reconciliations

The Dispatcher uses timeout values to remove assembly lines that are no longer needed.

About this task

For large reconciliations, the default value of timeouts, such as `SearchALUnusedTimeout`, can be too small. Small values can result in removing the assembly line before all results have been returned. Use the following variables to configure timeouts:

itdi_home

Specifies the home directory for IBM Tivoli Directory Integrator.

searchal_timeout

Specifies the number of seconds before unused assembly lines are cleaned up. Default value: 600.

Procedure

1. Stop the RMI Dispatcher.
2. In *itdi_home/itim_listener.properties*, update the following option:
`SearchALUnusedTimeout=searchal_timeout`
3. Restart the RMI Dispatcher for this change to take effect.

Configuring the number of concurrently running assembly lines

Using RMI Dispatcher controls, you can decrease the number of concurrently running assembly lines to prevent an `OutOfMemory` condition.

About this task

The `GlobalRunALCount` parameter controls the maximum number of assembly lines that can run concurrently. The default is 100. If the RMI Dispatcher receives a request that exceeds this limit, it places the request in the wait queue. The request stays in the wait queue until the number of running assembly lines is less than the specified limit. The `MaxWaitingALcount` parameter controls the number of waiting assembly lines. The default is 0 or no limit.

Decreasing the number of concurrently running assembly lines might prevent an `OutOfMemory` condition in Tivoli Directory Integrator. This condition shows up as a `Failed to fork OS thread` message in either a `javacore` or the `ibmdi.log` file.

Use the following variables to set limits:

itdi_home

Specifies the home directory for IBM Tivoli Directory Integrator.

max_running_ALs

Specifies the maximum number of assembly lines that can run concurrently. Zero indicates no limit. Default value: 100.

max_waiting_ALs

Specifies the maximum number of assembly lines that can wait at one time if the maximum number of running assembly lines is reached. Zero indicates no limit. Default value: 0.

Procedure

1. Stop the RMI Dispatcher.
2. In *itdi_home/itim_listener.properties*, update the following configuration options:
`GlobalRunALCount=max_running_ALs`
`MaxWaitingALcount=max_waiting_ALs`
3. Restart the RMI Dispatcher for this change to take effect.

Configuring assembly line caching

The RMI Dispatcher caches assembly lines, one per service instance, to improve performance for repeated requests to service instances. You can change the default value that controls the number of lines to prevent reusing stale connections.

About this task

The cached assembly lines improve performance by

- Retaining ready-to-use copies of the assembly lines in memory.
- Holding open connections to remote endpoints.

The size of the assembly line cache might require downward adjustments to compensate for memory constraints on the system. If the assembly line cache is too large, connections to remote endpoints might time out before they are reused. A timeout causes a failure of provisioning actions that use these stale connections.

The `ALCacheSize` configuration parameter (Dispatcher version 5.010) controls the number of cached assembly lines. In environments managing many service instances, change this value from 100 (the default) to 1 to prevent the reuse of stale connections. Use the following variables:

itdi_home

Specifies the home directory for IBM Tivoli Directory Integrator.

num_cached_ALs

Specifies the number of assembly lines to cache. Default value: 100.

Procedure

1. Stop the RMI Dispatcher.
2. In `itdi_home/itim_listener.properties`, update or add the following option:
`ALCacheSize=num_cached_ALs`
3. Restart the RMI Dispatcher for this change to take effect.

Chapter 9. Database servers used with IBM Tivoli Identity Manager

IBM Tivoli Identity Manager supports the following databases: DB2, Oracle Database, and Microsoft SQL Server. Each database requires slightly different tuning. Tuning the database is one of the most important tuning procedures for IBM Tivoli Identity Manager.

Each database server requires at least one processor and 1 GB of RAM. The database can be on a single-processor server by itself or share a multiprocessor server with other applications. The database server requires a minimum of 1 GB of RAM per processor.

Related information

Database server requirements

See the information about supported database products and versions.

Tuning IBM DB2

IBM Tivoli Identity Manager, version 5.0 and later, works with DB2 for Linux, UNIX, and Windows starting with Version 9. Version 9 has auto-tuning mechanisms that can reduce administrative and maintenance tasks.

About this task

Tuning DB2 to run with Tivoli Identity Manager includes:

- Adjusting the buffer pools.
- Modifying the number of connections.
- Modifying internal database values.
- Adding table space.
- Adjusting logs.
- Indexing.
- Updating statistics.

Related information

 Recommended fixes for Tivoli Identity Manager

See the information about supported versions of IBM DB2.

Database server requirements

See the information about supported database products and versions.

Enabling the self-tuning memory manager

The self-tuning memory manager removes the guesswork in determining the memory values for areas such as buffer pools, the sort heap, and the package heap. With self-tuning memory enabled, DB2 can move memory between areas based on system need. DB2, version 9, databases have the self-tuning memory manager enabled by default.

About this task

The `DATABASE_MEMORY` parameter determines the total amount of memory available for database-level memory areas. The memory setting depends on the operating system and the installer.

| Installer | Operating System | Setting |
|---|---------------------------|---|
| IBM Tivoli Identity Manager regular installer | AIX and Microsoft Windows | Self tuning with the AUTOMATIC value. The database memory grows or shrinks as needed, based on free operating system memory. |
| IBM Tivoli Identity Manager regular installer | Linux and Sun Solaris | Self tuning with the COMPUTED value. COMPUTED allocates a calculated value on database activation and releases memory on database deactivation. |
| IBM Tivoli Identity Manager Launchpad Single-server installer | Not applicable | 40000 pages or 164 MB |

Typically, the actual value determined by DB2 with either the AUTOMATIC or COMPUTED setting is sufficient. You can manually raise or lower the value when DB2 shares a system with other components or databases.

The amount of memory available to the global database pool depends on a number of factors, including the following ones:

- The amount of system memory.
- The memory used by other components on the system.
- The number of active database connections.

IBM Tivoli Identity Manager enables its buffer pools for automatic sizing and retains the default value of AUTOMATIC for the sort heap and package cache. If you upgrade from a previous version, the previous values for these settings are retained. You must set the value to AUTOMATIC to enable self-tuning.

Use the following variables when enabling self-tuning:

itim_database_name

Specifies the name of the IBM Tivoli Identity Manager database, such as `itimdb`.

db_mem_size

Specifies the amount of memory, measured in 4 KB pages, that DB2 uses for self-tuning. You can use the self-tuning value of AUTOMATIC for Windows and AIX platforms and COMPUTED for Linux and Solaris platforms.

Procedure

1. Optional: Determine the current value of `DATABASE_MEMORY` on your system.
 - a. Connect to the database.
 - b. Run the following commands:

```
db2 get db cfg for itim_database_name show detail
```

The commands display the current value for DATABASE_MEMORY. On Linux or Solaris, it also displays the computed future value.

- c. Multiply the value from step 1b on page 40 by 4096 for the number of bytes.
2. Update the database configuration to use the self-tuning memory manager:

```
db2 update db cfg for itim_database_name using SELF_TUNING_MEM ON
```
3. Set the amount of memory available to the self-tuning memory manager:

```
db2 update db cfg using DATABASE_MEMORY db_mem_size
```
4. Restart IBM DB2 for this change to take effect.

What to do next

If you upgraded from version 8 to version 9, enable self-tuning of the sort heap and package heaps. Enter the following commands:

```
db2 update db cfg using sortheap AUTOMATIC
db2 update db cfg using sheapthres_shr AUTOMATIC
db2 update db cfg using pckcachesz AUTOMATIC
```

If self-tuning is active, these settings take effect immediately and do not require an additional restart.

Related tasks

“Configuring database connections for DB2 databases” on page 42

DB2 requires enough memory for all possible JDBC connections to run statements without using swap space. If the system does not have sufficient memory, consider decreasing the maximum sizes for the JDBC Data Sources connection pools.

Related information

 Self-tuning memory (DB2 V9.7 information center)

Starting in DB2 Version 9, a memory-tuning feature simplifies the task of memory configuration by automatically setting values for several memory configuration parameters. When enabled, the memory tuner dynamically distributes available memory resources among the following memory consumers: buffer pools, locking memory, package cache, and sort memory.

Configuring row-level compression

Row-level compression decreases the on-disk footprint of the database. It also improves performance by decreasing I/O wait. It improves buffer pool usage even with the additional CPU usage required by compression.

About this task

Attention: Row-level compression is included with the DB2 Storage Optimization, which is a separately purchasable feature. The DB2 license that is included with IBM Tivoli Identity Manager and Tivoli Directory Server prohibits installing any separately purchasable features. To use row-level compression, you must purchase fully licensed versions of both DB2 Enterprise Server Edition and DB2 Storage Optimization.

DB2 version 9 can estimate how well a table can compress. Use the following variables and steps to determine if a specific table is a good compression candidate and to enable row-level compression.

tablename

Specifies the name of the table for which you want to estimate compression savings.

instancename

Specifies the name of the instance to which the tables belong.

Procedure

1. Evaluate your tables:
 - a. Run the DB2 INSPECT command to determine if the table is a good candidate for row-level compression.

```
db2 inspect rowcompestimate table name tablename  
schema instancename results keep tablename.inspect
```

This command creates the `sqllib/db2dump/tablename.inspect` binary file.
 - b. Format `sqllib/db2dump/tablename.inspect` into a readable format. Enter the following command:

```
db2inspf tablename.inspect tablename.inspect_out
```
 - c. Review the results in the `tablename.inspect_out` file
The report shows the percent of pages and space saved by compressing the table. If compression reduces the number of pages by at least 50%, the table is a good candidate for compression.
2. For each table that is a good candidate for compression, enable compression.
 - a. Turn off IBM Tivoli Identity Manager.
 - b. Connect to the database as an administrator.
 - c. Enter the following commands, each on a separate line:

```
db2 alter table instancename.tablename compress yes  
db2 reorg table instancename.tablename  
db2 reorg indexes all for table instancename.tablename
```
 - d. Run RUNSTATS on the table.

Configuring database connections for DB2 databases

DB2 requires enough memory for all possible JDBC connections to run statements without using swap space. If the system does not have sufficient memory, consider decreasing the maximum sizes for the JDBC Data Sources connection pools.

About this task

The default value for MAXAPPLS is AUTOMATIC, which is sufficient for most environments. If you require an explicit value set MAXAPPLS to five more than the total maximum number of connections.

When determining memory allocation for DB2, you must consider the number of active connections. Each connection is assigned a DB2 agent that is allocated its own private agent memory (`applheapsz`). DB2 allocates additional memory when running a statement (`stmtheap`). To calculate the amount of memory in megabytes required for a single connection, use this formula:

$$per_connection_memory = (applheapsz + stmtheap) * 4.096 / 1000$$

Use the following variables in the configuration procedure:

itim_database_name

Specifies the name of your IBM Tivoli Identity Manager database, such as `itimdb`.

num_connections

Specifies the maximum number of connections.

Procedure

1. Connect to the database as the database administrator.
2. Run the following command:

```
db2 update db cfg for itim_database_name using maxappls num_connections
```

Related tasks

“Configuring WebSphere JDBC connections” on page 11

IBM Tivoli Identity Manager server uses JDBC connections from WebSphere Application Server to communicate with the database.

Configuring buffer pools for the IBM Tivoli Identity Manager database

DB2 buffer pools must be large enough so that most table searches can read directly from memory instead of the disk. You can measure this value by looking at the hit ratio for the buffer pools.

About this task

The IBM Tivoli Identity Manager database has the following buffer pools:

IBMDEFAULTBP

Used as a buffer for table spaces with small extent sizes (4 KB).

ENROLEBP

Used as a buffer for table spaces with large extent sizes (32 KB). Most IBM Tivoli Identity Manager database tables use the table space with a large extent size.

If the buffer pools are not set to AUTOMATIC, use a 1:3 memory ratio between the IBMDEFAULTBP and ENROLEBP buffer pools. Use the following variables in the configuration procedure:

mem_for_itimdb

Specifies the amount of memory in bytes to allocate to the IBM Tivoli Identity Manager database buffer pools. Make this value small enough to be in physical memory so that it is not swapped out to disk. Suggested value: 500000000 (500 MB) or greater.

Procedure

1. Connect to the database as the database administrator.
2. Optional: View the current buffer pool sizes. Enter the following command at a command prompt:

```
db2 select bpname, npages, pagesize from syscat.bufferpools
```

An npages value of -1 indicates that the buffer pools are sized according to the BUFFPAGE parameter. A value of -2 indicates that the buffer pools use automatic sizing.

3. Calculate the optimum size, measured in pages, for the buffer pools:

$$ibmdefaultbp_npages = (mem_for_itimdb / 4096) * 0.25$$
$$enrolebp_npages = (mem_for_itimdb / 32768) * 0.75$$

4. Alter the buffer pool sizes for the database by running the following commands on separate lines:

```
db2 alter bufferpool ibmdefaultbp size ibmdefaultbp_npages
```

```
db2 alter bufferpool enrolebp size enrolebp_npages
```

Related tasks

“Calculating the buffer pool hit ratio” on page 101

The buffer pool hit ratio gives a good indication of how many data reads come from the buffer pool and how many from the disk. The larger the hit ratio, the less disk I/O used. Calculate the buffer pool hit ratio by enabling buffer pool monitoring and taking a database snapshot.

“Enabling the self-tuning memory manager” on page 39

The self-tuning memory manager removes the guesswork in determining the memory values for areas such as buffer pools, the sort heap, and the package heap. With self-tuning memory enabled, DB2 can move memory between areas based on system need. DB2, version 9, databases have the self-tuning memory manager enabled by default.

Configuring table spaces for IBM DB2 databases

IBM Tivoli Identity Manager uses a database managed space (DMS) table space to store data. This type of table space performs better than system managed space (SMS) table spaces, but you must preallocate disk space for the database to use. The tables spaces created by the installer have `autoresize` enabled and grow as needed.

About this task

You might need to define additional table space containers, depending on your specific environment, disk restrictions, and table space layouts.

Adding additional table space containers

DB2 performs better if a table space has multiple containers on multiple drives. You can add more containers to a table space to increase the amount of space available to tables.

About this task

Add more table spaces with the DB2 `alter tablespace` command. If possible, add files that reside on another physical drive. The creation and adoption of altered table spaces is not immediate. Examine the output of the `alter tablespace` command as it executes and rerun the command if the database is busy altering a table space.

Use the following variables in the procedure:

tablespace_name

Specifies the name of the table space for which you want to add containers. IBM Tivoli Identity Manager table space names are `ENROLE_DATA`, `ENROLE_INDEXES`, and `TEMP_DATA`.

database_home

Specifies the home directory of your database administrator, such as `/home/db2inst1`.

instance

Specifies the name of the instance, such as `db2inst1`. This is a subdirectory in *database_home*.

container

Specifies the name of the file you want to create to hold the table space container, such as `enrole_data2`.

num_pages

Specifies the number of 32 KB pages you want to add to the table space. To calculate the number of pages from the amount of disk space, divide the size in megabytes by 0.032768. A 512 MB table space is 15625 pages.

Note: This container might grow if you set the table space to autoresize.

Procedure

1. As the database administrator, connect to the database.
2. Run the following command for each table space.

```
db2 "ALTER TABLESPACE tablespace_name
    ADD ( FILE '/database_home/instance/NODE0000/SQL00001/container'
        num_pages)"
```

Enabling automatic resizing of table spaces

Enable automatic resizing so that containers for a table space can grow automatically if they become full.

About this task

Automatic resizing can decrease the administrative workload for DMS table spaces; however, make sure that the disks on which the containers reside do not become full.

Tip: All table spaces created for IBM Tivoli Identity Manager, version 6, have automatic resizing enabled by default. If you migrate from versions 4.5.1 or 4.6, you can benefit from enabling automatic resizing on existing table spaces.

Use the DB2 alter tablespace command and the following variables to enable automatic resizing on both new and existing table spaces.

database_name

Specifies the name of the database, such as db2inst1.

tablespace_name

Specifies the name of the table space on which you want to enable autoresize. IBM Tivoli Identity Managertable space names are ENROLE_DATA, ENROLE_INDEXES, and TEMP_DATA.

Procedure

1. Connect to the database as the database administrator.
2. Optional: View the current status of automatic resizing for the table space.
 - a. Enter the following command at a command prompt:

```
db2 get snapshot for tablespaces on database_name
```
 - b. In the stanza describing the table space look for the line:

```
Auto-resize enabled =
```

3. Turn on automatic resizing by running the following command,

```
db2 ALTER TABLESPACE tablespace_name AUTORESIZE YES
```

Setting the table space prefetch size

The default prefetch sizes of the ENROLE_DATA, ENROLE_INDEXES, and TEMP_DATA table spaces are not optimal. Change the prefetch size to AUTOMATIC so DB2 can control this parameter.

About this task

Use the following variable to set the table prefetch size:

tablespace_name

Specifies the name of the table space for which to set the prefetch size. IBM Tivoli Identity Manager table space names are ENROLE_DATA, ENROLE_INDEXES, and TEMP_DATA.

Procedure

1. As the database administrator, connect to the database.
2. Run the following command for each table space.

```
db2 ALTER TABLESPACE tablespace_name PREFETCHSIZE AUTOMATIC
```

Updating table space overhead and transfer rate

The DB2 overhead and transfer rate parameters used by IBM Tivoli Identity Manager table spaces might not be optimal for upgraded databases.

About this task

The optimizer uses DB2 overhead and transfer rate parameters to calculate query plan costs. IBM Tivoli Identity Manager table spaces use the version 8 default values for these parameters. In DB2, version 9, the default values changed to account for faster I/O subsystems. The following table shows the default values for both versions.

Table 1. Overhead and Transfer rate values

| DB2 version | Overhead rate parameter | Transfer rate parameter |
|--------------------------------|-------------------------|-------------------------|
| Version 8 | 24.1 | 0.9 |
| Version 9 (migrated databases) | 12.67 | 0.18 |
| Version 9 (new databases) | 7.5 | 0.06 |

You can determine the actual overhead and transfer rate values for your subsystem by using the formulas in the DB2 version 9 *Performance Guide*. If you cannot determine the value for your hardware, use the version 9 migration values for older hardware or new values for new hardware.

Use the following variables when updating the overhead and transfer rates:

tablespace_name

Specifies the name of the table space for which to set the prefetch size. The IBM Tivoli Identity Manager table space names are ENROLE_DATA, ENROLE_INDEXES, and TEMP_DATA.

overhead

Specifies the number of milliseconds required by the container before reading any data into memory. Suggested value: The calculated value for your hardware. Use the new databases values if you are running on new hardware. Use the migrated database values if you are running on old hardware. See Table 1.

transferrate

Specifies the number of milliseconds required to read one page of data into memory. Recommended value: the calculated value for your hardware, if

possible. Use the new databases values if you are running on new hardware. Use the migrated database values if you are running on old hardware. see Table 1 on page 46.

Procedure

1. Connect to the database as the database administrator.
2. Optional: View the current overhead and transfer rates. Enter the following command:

```
db2 select tbspace, overhead, transferrate from syscat.tablespace
```
3. Run the following command for each table space:

```
db2 ALTER TABLESPACE tablespace_name OVERHEAD overhead TRANSFERRATE transferrate
```

Disabling file system caching

IBM Tivoli Identity Manager table spaces are created with file system caching enabled. If the buffer pools are adequately sized, the file system cache is not necessary and can reduce performance due to double-buffering.

About this task

When the file system cache is disabled on a table space, DB2 uses Direct I/O (DIO) and bypasses the file system cache. DB2 can use Concurrent I/O (CIO) on some platforms with some file systems increasing I/O performance when file system caching is disabled.

If you tuned the buffer pools so that the buffer pool hit ratio is above 95%, disable file system caching. Use the following variables for this procedure:

database_name

Specifies the name of the database, such as db2inst1.

tablespace_name

Specifies the name of the table space for which you want to disable file system caching. IBM Tivoli Identity Manager table space names are ENROLE_DATA, ENROLE_INDEXES, and TEMP_DATA.

Procedure

1. Connect to the database as the database administrator.
2. Optional: View the current caching status.
 - a. Enter the following command:

```
db2 get snapshot for tablespaces on database_name
```
 - b. In the stanza describing the desired table space look for the following line:

```
File system caching = Yes
```
3. Run the following command for each table space:

```
db2 ALTER TABLESPACE tablespace_name NO FILE SYSTEM CACHING
```
4. Stop IBM Tivoli Identity Manager.
5. Stop and restart DB2.
The new caching policy becomes effective when DB2 restarts.

Table compression candidates for the IBM Tivoli Identity Manager database

The IBM Tivoli Identity Manager database can use row-level compression, which was introduced in DB2, version 9.

Typically, the following tables are good compression candidates:

- activity
- process
- processdata
- audit_event
- audit_mgmt_provisioning
- audit_mgmt_target
- audit_mgmt_delegate

Because building a compression dictionary requires the tables to have data, compression is not enabled by default.

Related tasks

“Configuring row-level compression” on page 41

Row-level compression decreases the on-disk footprint of the database. It also improves performance by decreasing I/O wait. It improves buffer pool usage even with the additional CPU usage required by compression.

Configuring transaction logs for DB2 databases

DB2 keeps logs during transaction processing. During large transactions, the default log number and sizes might be too small and cause transaction rollbacks. Increase the size and number of log files to resolve this issue.

About this task

Tip: For best performance, move transaction logs to a different physical drive than the one where the database is located. Intelligent data storage devices might not require a different physical drive.

The IBM Tivoli Identity Manager Middleware Configuration utility increases the size of the transaction logs to 10000 and updates the number of secondary logs to 12.

DB2 has the following types of transaction log files:

Primary logs

Allocated when the database is started. They remain allocated until the database is stopped.

Secondary logs

Allocated as needed after the primary logs are full. They are released when they are no longer needed.

Increase the number of secondary logs in preparation for large transactions. The default size of log files is 1000 4 KB pages or 4 MB. Increase this value to 10000 4 KB pages, or 40 MB. The following procedure, which uses these variables, increases the size of primary and secondary log files.

itim_database

Specifies the name of the IBM Tivoli Identity Manager database, such as *itim*.

logs_secondary

Specifies the number of secondary logs. Suggested value: 12.

logs_size

Specifies the size of the primary and secondary logs in 4 KB pages.
Suggested value: 10000.

log_path

Specifies the path where you want to put the transaction logs.

Procedure

1. Connect to the database as the database administrator.
2. Update the database configuration by running the following commands on separate lines.

```
db2 update db cfg for itim_database using logsecond logs_secondary  
db2 update db cfg for itim_database using logfilsiz logs_size  
db2 update db cfg for itim_database using newlogpath log_path
```

3. Stop and restart the database instance. The changes take effect when the database instance restarts.

Configuring database application heaps

Some of the queries that the IBM Tivoli Identity Manager application submits to the DB2 server result in complex SQL statements. If you see transaction rollback errors in the trace.log file, increase the values of the heaps in increments of 256 until the errors stop.

About this task

The IBM Tivoli Identity Manager Middleware Configuration utility increases the application heap size to 2048 and the application control heap size to 1024.

IBM Tivoli Identity Manager adjusts the values of the following parameters from their default state. Appropriate tuning requires additional adjustments.

itim_database

Specifies the name of the IBM Tivoli Identity Manager database, such as *itim*.

applheap_size

Specifies the value of *applheapsz* in 4 KB pages. Initial value: 2048.

appctl_size

Specifies the value of *appctl_heap_sz* in 4 KB pages. Initial value: 1024.

Procedure

1. Connect to the database as the database administrator.
2. Update the database configuration. Run the following commands separate lines:

```
db2 update db cfg for itim_database using applheapsz applheap_size  
db2 update db cfg for itim_database using appctl_heap_sz appctl_size
```

3. Stop and restart the database instance. The changes take effect when the database instance restarts.

Configuring automatic statistics collection for the IBM Tivoli Identity Manager database

Administrators can configure automatic statistics collection so that DB2 automatically updates database statistics. Automatic collection eliminates the necessity of manually running a periodic statistics collection against the database.

About this task

Automatic statistics collection is not enabled by default. For WebSphere, version 6.0, and later JMS implementation to operate properly, exclude the SIBOWNER table from the automatic statistics collection. To improve performance, exclude the SCHEDULED_MESSAGE and PROCESSDATA tables.

Important: Enabling automatic statistics collection without excluding the SIBOWNER table results in database lockups.

For newly created databases, run manual statistics collection (RUNSTATS) after a small data load, even if automatic collection is enabled. RUNSTATS provides statistics for good performance until DB2 initiates the first automatic collection.

Use the following variable in the procedure:

itim_database_name

Specifies the name of your IBM Tivoli Identity Manager database, such as itimdb.

Tip: If the database server does not have the DB2 Control Center, perform this task from a remote system by connecting to the IBM Tivoli Identity Manager database.

Procedure

1. Use DB2 Control Center to update the DB2 maintenance policies.
 - a. Start the DB2 Control Center.
 - b. Connect to your database with database administrator authority.

Note: If you do not see your database in Control Center, add it to the catalog before you can continue.

- c. Browse to *itim_database_name*.
 - d. Right-click *itim_database_name*.
 - e. Click **Configure Automatic Maintenance**.
 - f. Click **Next** until you access **Activities**.
 - g. Select **Optimize data access (RUNSTATS)**.
 - h. Click **Configure Settings**.
 - i. Click **Selected tables**.
 - j. Select **Use the custom filter**.
 - k. In the **Conditions** field, type:
TABNAME NOT IN ('SIBOWNER', 'SCHEDULED_MESSAGE', 'PROCESSDATA', 'PROCESS', 'ACTIVITY')
 - l. Click **Refresh Resulting Tables**.
 - m. Confirm that **Resulting tables (SCHEMA.NAME)** is populated with all tables except for the ones that you specified in step 1k.
 - n. Click **OK**.
 - o. Click **Finish**.
 - p. Confirm the message that no errors were encountered.
 - q. Quit Control Center.
2. Enable automatic statistics collection:
 - a. As the database administrator, connect to the database at the command prompt.

- b. Run the following command:

```
db2 update db cfg for itim_database_name using auto_runstats on
```

Related information



Control Center overview

See the information about using IBM DB2 Control Center.



RUNSTATS command

See the information about using the RUNSTATS command.

Updating IBM Tivoli Identity Manager database statistics for DB2 databases

DB2 requires statistics on the number of rows in the tables and available indexes to efficiently execute queries. DB2 version 9 can update the statistics automatically, or you can manually update the statistics.

About this task

If enabling automatic statistics collection is not feasible, you must run the RUNSTATS command manually. Update table and index statistics after large Directory Server Markup Language (DSML) loads, HR feeds, and reconciliations.

Note: DB2 REORGCHK does not update index statistics and is not a replacement for RUNSTATS.

If you experience high processor usage or poor DB2 performance, run RUNSTATS on all of the tables in the database. To update index statistics, run the RUNSTATS command on each table individually. IBM Tivoli Identity Manager performance tuning scripts (`perftune_runstats.sh` and `perftune_runstats.bat`) detect the version of DB2 and run the RUNSTATS command against all tables for a specific schema in a database.

If you run the RUNSTATS command in a working environment, make sure that the connected applications can continue to write to the database. Use the `allow write access` option so users can write to a database while RUNSTATS runs.

Use RUNSTATS on an idle or lightly used database because it requires update locking on the system statistics table to update the database statistics. The system acquires locks on the tables that are used by the database optimizer to fulfill queries. The locks might cause transaction rollbacks on a database with a heavy load.

In addition to running RUNSTATS on all tables in the database, you must manually update the statistics table for the `ACTIVITY`, `PROCESS`, `PROCESSDATA`, and `SCHEDULED_MESSAGE` table. Updating the statistic tables ensures a minimum cardinality. Setting a minimum cardinality on these tables helps the DB2 query optimizer and can decrease locking issues in the database.

The following procedure runs RUNSTATS on every table in the `ITIMUSER` schema.

Procedure

1. Connect to the database as the database administrator.
2. Generate a listing of all tables in the schema by running the following command:

```
db2 list tables for all | grep ITIMUSER
```

3. For each table in the ITIMUSER schema, run the following command on a single line:

```
db2 runstats on table ITIMUSER.table_name
   on all columns with distribution
   and detailed indexes all allow write access
```

4. Manually update the database statistics table for the workflow tables by running the following commands on separate lines:

```
db2 update sysstat.tables
   set card = 50000
   where tabname = 'ACTIVITY' and card < 50000
db2 update sysstat.tables
   set card = 50000
   where tabname = 'PROCESS' and card < 50000
db2 update sysstat.tables
   set card = 50000
   where tabname = 'PROCESSDATA' and card < 50000
db2 update sysstat.tables
   set card = 50000
   where tabname = 'SCHEDULED_MESSAGE' and card < 50000
```

Related tasks

“Configuring automatic statistics collection for the IBM Tivoli Identity Manager database” on page 49

Administrators can configure automatic statistics collection so that DB2 automatically updates database statistics. Automatic collection eliminates the necessity of manually running a periodic statistics collection against the database.

Related information



ITIM performance tuning scripts

Download performance tuning scripts for IBM Tivoli Identity Manager.

Changing the maximum number of open files

To work well with other applications running on the system, DB2 sets a limit on the number of files it keeps open with the `maxfilop` setting. You can adjust this number to meet the needs of your environment.

About this task

After reaching the specified limit, DB2 closes a currently open file to open the new one. This process can cause a performance loss on systems that do not require a restriction on the number of open files. The IBM Tivoli Identity Manager installation raises the default value. If database snapshots show that database files have been closed, increase this value in increments of 64.

The IBM Tivoli Identity Manager Middleware Configuration utility increases the maximum number of open files to 256.

IBM Tivoli Identity Manager adjusts the values of the following parameters from their default state. Further adjustment might be required.

itds_database

Specifies the name of the IBM Tivoli Identity Manager database, such as `itimdb`.

max_files_open

Specifies the maximum number of files DB2 has open at any one time.
Initial value: 256.

Procedure

1. Connect to the database as the database administrator.
2. Run the following command on a single line:

```
db2 update db cfg for itim_database using maxfilop max_files_open
```

Adjusting lock list and maximum locks

The default settings for the DB2 lock list (`locklist`) and maximum locks (`maxlocks`) are adequate for most environments.

Increase these values if the local DB2 administrator tells you to do so.

Changing the lock timeout

The default lock timeout value (`locktimeout`) in the IBM Tivoli Identity Manager database is infinity. You can adjust this value if locking problems occur.

If you see locking problems, you can change this value from infinity, represented by `-1`. The configured value must be greater than or equal to the WebSphere total transaction timeout value, which has a default value of 1200. Setting this value to less than the WebSphere total transaction timeout is unsupported. It can cause transaction rollback errors because not all components recover from a lock timeout.

Disabling the EXTSHM parameter on AIX

If the EXTSHM environment parameter is enabled, it can degrade database performance for large AIX systems.

About this task

IBM Tivoli Identity Manager versions 4.5.1 and 4.6 required EXTSHM to be enabled for the Type 2 JDBC driver. Versions 5.0 and later use the Type 4 JDBC driver, which eliminates the requirement.

Disable EXTSHM under the following circumstances:

- If you upgraded from an earlier version of IBM Tivoli Identity Manager.
- If you used the Middleware Configuration utility to configure the IBM Tivoli Identity Manager database. Some versions of the utility enable EXTSHM for the IBM Tivoli Identity Manager database instance.

Procedure

1. Connect to the database as the database administrator.
2. Determine if EXTSHM parameter is enabled by running the following command:

```
db2set
```

 - If the line `DB2ENVLIST=EXTSHM` is present, the parameter is enabled. Perform the remaining steps to disable it.
 - If the line is not present, the parameter is not enabled. No additional action is required.
3. Update the database configuration by running the following command:

```
db2set DB2ENVLIST=
```
4. Stop and restart the database instance. The changes take effect when the database instance restarts.

Improving disk I/O performance

Disk I/O performance depends upon the drive types, layout, and configuration. You can change some registry variables to improve performance on some systems.

About this task

The following DB2 registry variables might improve performance.

| Systems | Parameter | Value |
|---|----------------------------------|-------|
| All systems | DB2_USE_ALTERNATE_PAGE_CLEANSING | ON |
| Systems with SAN, RAID, or other advanced disk subsystem: | DB2_PARALLEL_IO | * |

Related information



DB2 registry and environment variables

See the information about using DB2 registry and environment variables.

Tuning Oracle

IBM Tivoli Identity Manager supports Oracle databases starting with version 10g on some operating systems.

About this task

Tuning Oracle to run with IBM Tivoli Identity Manager requires configuring table spaces, indexing, and updating statistics.

Related information

Database server requirements

See the information about supported database products and versions.

Configuring the init.ora configuration file

The default Oracle configuration uses the small settings in the `init.ora` file for the database. Using the middle or large values can provide faster performance.

Consult with an Oracle DBA or administrator or the Oracle documentation for more information about tuning the Oracle server.

Configuring database connections for Oracle databases

The Oracle `SESSIONS` parameter controls the number of database connections. By default this value is derived from the `PROCESSES` parameter. You might need to increase the default value.

About this task

Increase the `PROCESSES` parameter if the default derived value is not high enough. If an explicit value is needed, set `PROCESSES` to 5 more than the total maximum number of connections.

If the `SESSIONS` parameter is not derived from the `PROCESSES` parameter, it might be necessary to update both parameters.

Use the following variable to configure the number of database connections:

num_connections

Specifies the maximum number of connections.

Procedure

1. As the database administrator, connect to the database.
2. Run the following command:

```
alter system set processes=num_connections scope=spfile;
```
3. Stop and restart the database instance. Changes take effect when the instance is restarted.

Related tasks

“Configuring WebSphere JDBC connections” on page 11
IBM Tivoli Identity Manager server uses JDBC connections from WebSphere Application Server to communicate with the database.

Enabling XA recovery operations

You must enable XA recovery operations after installing IBM Tivoli Identity Manager on Oracle.

About this task

Failure to enable XA recovery can result in a WTRN0037 message that indicates that the transaction service encountered an error on an `xa_recover` operation. Use the following variable to enable XA recovery operations:

itim_db_user

Specifies the user that owns the IBM Tivoli Identity Manager database, such as `itimuser`.

Procedure

1. As the database administrator, connect to the database.
2. Run the following commands on separate lines:

```
grant select on pending_trans$ to public;  
grant select on dba_2pc_pending to public;  
grant select on dba_pending_transactions to public;  
grant execute on dbms_system to itim_db_user;
```
3. Stop and restart the database instance.

Related information

 WTRN messages

See the information about WTRN messages that are issued by WebSphere Application Server.

Configuring open cursors

IBM Tivoli Identity Manager uses prepared statements through the WebSphere Application Server JDBC interface. Each prepared statement requires an open cursor in Oracle. If you receive an error message about too many open cursors, you can increase the maximum number of open cursors.

About this task

The message `ORA-01000` from the Oracle server indicates that too many open cursors exist. You must either increase the number of `OPEN_CURSORS` in Oracle or

decrease the maximum number of JDBC connections in WebSphere Application Server. To increase the number of OPEN_CURSORS, use the following variable:

num_open_cursors

Specifies the maximum number of open cursors. Default value: 50.
Suggested value: 1000.

Procedure

1. As the database administrator, connect to the database.
2. Run the following command:

```
alter system set open_cursors=num_open_cursors scope=both;
```
3. Stop and restart the database instance. Changes take effect when the instance is restarted.

Configuring table spaces for Oracle databases

During database configuration, IBM Tivoli Identity Manager creates several small table spaces that can automatically extend as necessary. You can add additional data files.

Additionally, consider spreading the data across multiple physical disks either during the initial database configuration or afterward by adding additional table space containers.

Spreading database data across multiple disks

Spreading the database files across multiple disks decreases the I/O contention in the database and improves Oracle performance. When you create the Oracle database, consider spreading the table space files across multiple disks.

About this task

Use the following variable in the procedure.

disk# Specifies the disk onto which you want to spread the table spaces.

See *itim_home/config/rdbms/oracle/enrole_admin_template.sql* for information about the default table space definitions.

The *itim_home/config/rdbms/oracle/create_rollbackSegment.sql* script puts the rollback segment table space on ORACLE_HOME (because it specifies only the file name). Consider using a different disk if your environment supports it.

Important: The following SQL statements are for illustrative purposes only. They are environment-specific for both the file system and the size that are allocated to each table space. Consult your Oracle DBA, and tailor the statements to your environment before you apply them.

Use the following procedure to distribute the Oracle database across four hard disk drives: *disk1* through *disk4* with Oracle on *disk1*.

Procedure

1. Create the TEMP table space on *disk2*

```
create temporary tablespace TEMP
  tempfile '/disk2/oradata/temp01.dbf'
  size 1000m
  reuse
  autoextend on next 32m
  maxsize unlimited;
```


2. Create the ENROLE_DATA table space on *disk3*

```
create tablespace ENROLE_DATA
  datafile '/disk3/oradata/enrole_data_01.dbf'
  size 64m
  autoextend on next 64m
  maxsize unlimited;
```

3. Create the ENROLE_INDEXES table space on *disk4*

```
create tablespace ENROLE_INDEXES
  datafile '/disk4/oradata/enrole_indexes_01.dbf'
  size 32m
  autoextend on next 32m
  maxsize unlimited;
```

Related information

 [IBM Tivoli Identity Manager Server Installation and Configuration Guide](#)
See the information about installing and configuring IBM Tivoli Identity Manager servers.

Adding table space data files

It might be necessary to define additional data files on separate physical devices to provide enough disk space for large deployments.

About this task

The initial data files are created with `autoextend on` and `maxsize unlimited`. Use the following variables when adding additional data files to a table space:

tablespace_name

Specifies the name of the IBM Tivoli Identity Manager table space to alter, such as ENROLE_DATA, ENROLE_INDEXES, or ITIML000_DATA.

datafile_name

Specifies the name of the file to use when adding additional data files to a table space or modifying an existing data file. Example value:
/data/ou1/app/oracle/oradata/itimdb/enrole_data2.dbf.

initial_size

Specifies the initial size of the data file. Example value: 512m.

maxsize_string

Specifies the string used to set the maximum size of the data file. Use UNLIMITED if you want the data file to grow unbounded. Use `maxsize <number>` to limit it to a specific size. Example: `maxsize 2048M`.

Procedure

1. As the database administrator, connect to the database.
2. Add data files to a table space. Run the following command:

```
alter tablespace tablespace_name
  add datafile 'datafile_name'
  size initial_size
  autoextend on maxsize_string;
```

3. Optional: To alter the maximum size of an existing table space. Run the following command:

```
alter database datafile 'datafile_name'  
autoextend on maxsize_string;
```

Configuring IBM Tivoli Identity Manager indexes for Oracle databases

Adding an index to a heavily used table can greatly increase performance. Without indexes, Oracle must scan every row of the table until it finds the specified data. With an index, it uses a more efficient search method.

About this task

Operational database queries require the following indexes:

- ACCT_CHANGE (POLICY_ANALYSIS_ID ASC, OPERATION_TYPE ASC, REASON ASC)
- ACTIVITY_LOCK (PROCESS_ID ASC)
- ACTIVITY (PROCESS_ID DESC)
- BULK_DATA_INDEX (DATAOBJECTID DESC)
- BULK_DATA_INDEX (STOREID DESC)
- BULK_DATA_STORE (SERVICEID DESC)
- POLICY_ANALYSIS (LAST_ACCESSED ASC, ANALYSIS_ID)
- PROCESS (PARENT_ACTIVITY_ID ASC, ID DESC)
- PROCESS (REQUESTER ASC, PARENT_ID ASC, TENANT ASC)
- PROCESSLOG (ACTIVITY_ID ASC)
- PROCESSLOG (PROCESS_ID ASC)
- RECONCILIATION_INFO (ACCOUNTID ASC, RECONID DESC)
- RESOURCE_PROVIDERS (RESOURCE_STATUS ASC, RESTART_TIME ASC, PROVIDER_ID ASC)
- TASKS_VIEWABLE (VIEW_ID ASC, VIEWABLE ASC, TASK_ID ASC)

DBPurge performance improves with the creation of the following indexes:

- AUDIT_EVENT (WORKFLOW_PROCESS_ID ASC, ID DESC)
- AUDIT_MGMT_DELEGATE (EVENT_ID ASC)
- AUDIT_MGMT_PROVISIONING (EVENT_ID ASC)
- AUDIT_MGMT_TARGET (EVENT_ID ASC)
- LCR_INPROGRESS_TABLE (CHILD_ID ASC)
- RECONCILIATION (COMPLETED ASC)
- RECONCILIATION_INFO (RECONID ASC, OPERATION ASC)
- WORKFLOW_CALLBACK (PROCESS_ID ASC)

The preceding indexes are defined in the `itimIndexes/itim5.0_indexes_for_oracle.sql` file that is included with IBM Tivoli Identity Manager performance tuning scripts. Use this file to apply the indexes to ensure consistent naming. Indexes that apply to specific conditions, such as viewing completed requests sorted by completion time, are included in the file but commented out. You can edit the file to uncomment these indexes.

Procedure

1. Enter `sqlplus` at a command prompt.
2. Connect to the database as the system user.
3. In the SQLPLUS interface, run the following command:

@ itim5.0_indexes_for_oracle.sql

What to do next

Update database statistics.

Related tasks

“Updating IBM Tivoli Identity Manager database statistics for Oracle databases”

You must gather and update database statistics at regular intervals. Intervals can be one week to one month on a production IBM Tivoli Identity Manager system or after processing a large amount of data.

Related information

 ITIM performance tuning scripts

Download performance tuning scripts for IBM Tivoli Identity Manager.

Updating IBM Tivoli Identity Manager database statistics for Oracle databases

You must gather and update database statistics at regular intervals. Intervals can be one week to one month on a production IBM Tivoli Identity Manager system or after processing a large amount of data.

Before you begin

If you have not already done so, install the DBMS_STAT package.

About this task

Oracle uses statistics to make query decisions on locating information that impact how fast Oracle can return requests. Use the following variable with the Oracle DBMS_STAT commands:

database_instance

Specifies the name of the database instance, such as enrole.

Tip: Generate statistics during off-peak times. Generating statistics can take from several minutes to several hours for a large database.

Procedure

1. Create a file named `Oracle_dbms.stat_cmds.txt`.
2. Edit the file and insert the following text:

```
exec dbms_stats.gather_schema_stats(ownname => 'database_instance',
  cascade => true);
```
3. Enter `sqlplus` at a command prompt.
4. Connect to the database as the system user.
5. In the SQLPlus interface, run the following command:

```
@ Oracle_dbms.stat_cmds.txt
```

Tuning Microsoft SQL Server

Tivoli Identity Manager supports certain versions Microsoft SQL Server databases.

About this task

Tuning Microsoft SQL Server to run with the IBM Tivoli Identity Manager product requires indexing.

Related information

Database server requirements

See the information about supported database products and versions.

Configuring indexes on Microsoft SQL Server databases

Adding an index to a heavily used table can greatly increase performance. Without indexes, Microsoft SQL Server must scan every row of the table until it finds the specified data. With an index, it uses a more efficient search method.

About this task

Operational database queries require the following indexes:

- ACCT_CHANGE (POLICY_ANALYSIS_ID ASC, OPERATION_TYPE ASC, REASON ASC)
- ACTIVITY_LOCK (PROCESS_ID ASC)
- ACTIVITY (PROCESS_ID DESC)
- POLICY_ANALYSIS (LAST_ACCESSED ASC, ANALYSIS_ID)
- PROCESS (ID ASC, STATE, ASC)
- PROCESS (PARENT_ACTIVITY_ID ASC, ID DESC)
- PROCESS (SUBMITTED DESC, PARENT_ID, ASC)
- PROCESS (REQUESTER ASC, PARENT_ID ASC, TENANT ASC)
- PROCESSDATA (PROCESS_ID ASC, DEF_ID ASC, VALUE_LAST_MODIFIED ASC)
- PROCESSLOG (ACTIVITY_ID ASC)
- PROCESSLOG (PROCESS_ID ASC)
- RECONCILIATION_INFO (ACCOUNTID ASC, RECONID DESC)
- RESOURCE_PROVIDERS (RESOURCE_STATUS ASC, RESTART_TIME ASC, PROVIDER_ID ASC)
- SCHEDULED_MESSAGE (SERVER ASC)
- TASKS_VIEWABLE (VIEW_ID ASC, VIEWABLE ASC, TASK_ID ASC)

DBPurge performance can improve with the creation of the following indexes:


- AUDIT_EVENT (WORKFLOW_PROCESS_ID ASC, ID DESC)
- AUDIT_MGMT_DELEGATE (EVENT_ID ASC)
- AUDIT_MGMT_PROVISIONING (EVENT_ID ASC)
- AUDIT_MGMT_TARGET (EVENT_ID ASC)
- LCR_INPROGRESS_TABLE (CHILD_ID ASC)
- RECONCILIATION (COMPLETED ASC)
- RECONCILIATION_INFO (RECONID ASC, OPERATION ASC)
- WORKFLOW_CALLBACK (PROCESS_ID ASC)

The preceding indexes are defined in the `itimIndexes/itim5.0_indexes_for_mssql.sql` file included with the IBM Tivoli Identity Manager performance tuning Scripts. Use this file to apply the indexes to ensure consistent naming. Indexes that apply to specific conditions, such as viewing completed requests sorted by completion time, are included in the file but commented out. You can edit the file to uncomment these indexes.

Procedure

1. Start Microsoft SQL Server Management Studio.
2. Log in as the IBM Tivoli Identity Manager database administrator.
3. Open the `itim5.0_indexes_for_mssql.sql` file.
4. Select the IBM Tivoli Identity Manager database from the database selection list.
5. Click **Execute** to create the indexes in the file.

Related information

 [ITIM performance tuning scripts](#)
Download performance tuning scripts for IBM Tivoli Identity Manager.

Chapter 10. Directory servers supported by IBM Tivoli Identity Manager

IBM Tivoli Identity Manager supports two different directory servers: Tivoli Directory Server and Sun ONE Directory Server.

Tuning Tivoli Directory Server

When tuning IBM Tivoli Directory Server, it is important to understand the interaction between the IBM Tivoli Directory Server process and DB2.

In a well-tuned environment, the Tivoli Directory Server process and the DB2 processes use approximately the same amount of CPU cycles. DB2 can max out the CPU usage trying to fulfill queries in a poor manner.

Both Tivoli Directory Server and DB2 have caches that speed up data retrieval. Optimizing available memory is the key to tuning IBM Tivoli Directory Server. When a read request comes in to Tivoli Directory Server, it checks the filter cache to see if it saw that search filter previously. If it has, it pulls the results from the cache, otherwise the query goes to DB2. After evaluating the search filter, Tivoli Directory Server pulls the entries that match the search filter from the entry cache. If the values are not in the entry cache, it queries DB2. For each request, DB2 checks to see if the data is in a buffer pool. If not, it reads the value from the disk. Ideally, all requests to the directory server register a Tivoli Directory Server cache hit or a DB2 buffer pool hit for the quickest response. Queries that require disk access can be slow.

Related information

“Tivoli Directory Server outages” on page 91


Incorrect system or product configuration can cause Tivoli Directory Server to fail, hang, or disappear due to resource restrictions.

“Tivoli Directory Server slow queries” on page 91

Slow queries from IBM Tivoli Directory Server can degrade overall system performance.

 [IBM Tivoli Directory Server V6.2 Performance Tuning and Capacity Planning Guide](#)

See product information about tuning the IBM Tivoli Directory Server V6.2.

 [IBM Tivoli Directory Server V6.1 Performance Tuning and Capacity Planning Guide](#)

See product information about tuning the IBM Tivoli Directory Server V6.1.

 [IBM Tivoli Directory Server V6.0 Performance Tuning Guide](#)

See product information about tuning the IBM Tivoli Directory Server V6.0.

Configuring cache sizes

You can configure the Tivoli Directory Server caches to increase performance and meet the needs of your environment.

About this task

IBM Tivoli Directory Server has the following caches:

Access control list (ACL) cache

Because IBM Tivoli Identity Manager server binds as an authoritative user, this cache is used only for internal processes. The allocated size can be small, and the memory can be used, which increases Tivoli Directory Server performance.

Filter cache

This cache helps programs that issue more read requests than write or update requests, because the entire filter cache is invalidated at every write. IBM Tivoli Identity Manager frequently updates the directory server, so it is not beneficial to allocate a large filter cache. Enable the filter cache, but keep it small.

Entry cache

You can control how many entries the entry cache can store. You cannot restrict the size of the cache. The size of each entry is based on the number and the size of attributes that a specific LDAP entry has.

Typically, many entries are users and their accounts, which have a fairly constant size. When setting the value for the entry cache, calculate the size of the average entry. Divide the size of the average memory into the amount of memory used by the Tivoli Directory Server process.

Users with few attributes can generate entry sizes that are approximately 4 KB. Users with more attributes can generate entry sizes around 9 KB. See the *IBM Tivoli Directory Server Performance Tuning Guide* for the procedure to determine the average entry size.

Do not set the entry cache size larger than available physical memory. If the Tivoli Directory Server process size exceeds the amount of available memory, page swapping causes significant performance degradation. When increasing the cache size, make sure the amount of memory required does not exceed the maximum amount a process can allocate. For example, the maximum amount is 2 GB for most 32-bit processes.

Example: For an average entry cache size of 9 KB, setting the entry cache size to 75,000 would require 675 MB ($75,000 * 9 \text{ KB} = 675,000 \text{ KB} = 675 \text{ MB}$) of physical RAM for the entry cache. The requirement does not include the 128 MB for the server process.

Attribute cache

Performance metrics suggest that the attribute cache available in Tivoli Directory Server, version 5.0 and later, does not provide a significant performance boost. You can allocate the memory elsewhere.

Use the following variables for configuring the cache sizes:

acl_cache

Specifies whether the ACL cache is used. Suggested value: TRUE (enabled).

acl_cache_size

Specifies the size of the ACL cache. Suggested value: 100.

filter_cache_size

Specifies the size of the filter cache. Suggested value: 100.

entry_cache_size

The size of the entry cache. Suggested value: *max_users* * (*average_accounts* + 1)

For example if you have 25,000 users with two accounts each: 25,000 * (2+1) = 75,000. This value is bounded by the amount of memory allocated to the Tivoli Directory Server process minus the size of the process itself (about 128 MB).

Procedure

1. Stop IBM Tivoli Directory Server.
2. Update the following configuration options in `ibmslapd.conf`:

```
ibm-slapdACLCache: acl_cache
ibm-slapdACLCacheSize: acl_cache_size
ibm-slapdFilterCacheSize: filter_cache_size
ibm-slapdEntryCacheSize: entry_cache_size
```
3. Restart IBM Tivoli Directory Server for these changes to take effect.

What to do next

Caches are only one part of tuning the IBM Tivoli Directory Server. Tuning the underlying IBM DB2 database has equal or greater performance impact than tuning the caches. Do not skip the DB2 tuning.


Related information

“Tivoli Directory Server outages” on page 91


Incorrect system or product configuration can cause Tivoli Directory Server to fail, hang, or disappear due to resource restrictions.

“Tivoli Directory Server slow queries” on page 91

Slow queries from IBM Tivoli Directory Server can degrade overall system performance.

 [IBM Tivoli Directory Server V6.2 Performance Tuning and Capacity Planning Guide](#)

See product information about tuning the IBM Tivoli Directory Server V6.2.

 [IBM Tivoli Directory Server V6.1 Performance Tuning and Capacity Planning Guide](#)

See product information about tuning the IBM Tivoli Directory Server V6.1.

 [IBM Tivoli Directory Server V6.0 Performance Tuning Guide](#)

See product information about tuning the IBM Tivoli Directory Server V6.0.

Configuring paging parameters

Tivoli Directory Server supports returning search results in pages so that the client has more control over receiving the data. If you enable paged searches in IBM Tivoli Identity Manager, you must set the paged search parameters correctly for optimum performance.

About this task

Because paged searches require more resources on the directory server, you can specify whether non-administrative users can perform paged searches. You can also specify the number of concurrent paged searches.

Use the following variables when configuring paging parameters:

allow_non_admin

Specifies if non-administrative users can request paged searches. Suggested value: TRUE, if IBM Tivoli Identity Manager is binding as a non-administrative user. If not, specify FALSE. Default value: TRUE

concurrent_paged_searches

Specifies the maximum number of concurrent paged searches. Set this value to 1 more than the maximum expected number of paged searches. Default value: 3

When increasing the number of concurrent paged searches, monitor resource utilization on the directory server to ensure that overall performance does not degrade. Also ensure that the number of backend database connections is larger than the total number of paged searches.

Procedure

1. Stop IBM Tivoli Directory Server.
2. In `ibmslapd.conf`, update the following configuration options:

```
ibm-slapdPagedResAllowNonAdmin: allow_non_admin
ibm-slapdPagedResLmt: concurrent_paged_searches
```
3. Restart IBM Tivoli Directory Server for these changes to take effect.

Related tasks

“Configuring paged searches” on page 26

IBM Tivoli Identity Manager, version 5.0 and later, incorporates LDAP paged searches to alleviate JavaOutOfMemory errors in large environments.

Configuring database buffer pools for the Tivoli Directory Server database

DB2 buffer pools are the secondary buffer for Tivoli Directory Server. These buffer pools must be large enough so that most table searches can be read directly from memory instead of using the disk.

About this task

Tivoli Directory Server database has the following buffer pools:

IBMDEFAULTBP

Used as a buffer for table spaces with small extent sizes (4 KB). Most of the tables in the database have table spaces with a small extent size and use IBMDEFAULTBP.

LDAPBP Used as a buffer for table spaces with large extent sizes (32 KB)

DB2, version 9, is the default for Tivoli Directory Server, version 6.1. If you use version 9, set the buffer pools to AUTOMATIC so that the self-tuning memory manager can adjust the memory settings for your workload.

If the buffer pools are not set to AUTOMATIC, use a 3:1 memory ratio between IBMDEFAULTBP and LDAPBP. Allocate enough memory to the DB2 buffer pools so the buffer pool hit ratio is greater than 95%. Allocate the remaining memory to the Tivoli Directory Server process and the caches.

Use the following variables to configure buffer pools:

ldap_database

Specifies the name of the IBM Tivoli Directory Server database, such as `ldapdb2`.

mem_for_ldapdb2_bps

Specifies the amount of memory in bytes to allocate to the `ldapdb2` buffer pools. Make this value small enough so that it is in physical memory and is not swapped out to disk. Suggested value: 500000000 (500 MB) or greater.

Procedure

1. Connect to the database as the database administrator.
2. Optional: View the current buffer pool sizes by entering the following command at a command prompt:

```
db2 select bpname, npages, pagesize from syscat.bufferpools
```

An `npages` value of -1 indicates that the buffer pools are sized according to the `BUFFPAGE` database configuration parameter. A value of -2 indicates that the buffer pools use automatic sizing.
3. Calculate the size for the buffer pools, measured in pages:
$$ibmdefaultbp_npages = (mem_for_ldapdb2_bps / 4096) * 0.75$$
$$ldapbp_npages = (mem_for_ldapdb2_bps / 32768) * 0.25$$
4. Alter the buffer pool sizes for the database by running the following commands on separate lines:

```
db2 alter bufferpool ibmdefaultbp size ibmdefaultbp_npages
db2 alter bufferpool ldapbp size ldapbp_npages
```

Related tasks

“Calculating the buffer pool hit ratio” on page 101

The buffer pool hit ratio gives a good indication of how many data reads come from the buffer pool and how many from the disk. The larger the hit ratio, the less disk I/O used. Calculate the buffer pool hit ratio by enabling buffer pool monitoring and taking a database snapshot.

“Enabling the self-tuning memory manager” on page 39

The self-tuning memory manager removes the guesswork in determining the memory values for areas such as buffer pools, the sort heap, and the package heap. With self-tuning memory enabled, DB2 can move memory between areas based on system need. DB2, version 9, databases have the self-tuning memory manager enabled by default.

Disabling file system caching

Both Tivoli Directory Server table spaces (`LDAPSPACE` and `USERSPACE1`) are created with file system caching enabled. If the buffer pools are adequately sized, the file system cache is unnecessary and can reduce performance due to double-buffering.

About this task

When the file system cache is disabled on a table space, DB2 uses Direct I/O (DIO) and bypasses the file system cache. DB2 can use Concurrent I/O (CIO) on some platforms. Some file systems increase I/O performance when file system caching is disabled.

If you tuned the buffer pools so that their hit ratio is over 95%, disable file system caching.

Tivoli Directory Server, version 6.2, disables file system caching by default.

Use the following variable when disabling the file caching system:

tablespace_name

Specifies the name of the table space for which you want to disable file system caching. IBM Tivoli Identity Manager table space names are LDAPSPACE and USERSPACE1.

Procedure

1. Connect to the database as the database administrator.
2. Run the following command for each table space:

```
db2 ALTER TABLESPACE tablespace_name NO FILE SYSTEM CACHING
```
3. Stop the IBM Tivoli Directory Server.
4. Stop and restart IBM Tivoli Directory Server database. The new caching policy becomes effective after you disconnect all database connections.
5. Start the IBM Tivoli Directory Server.

Related information



Creating table spaces without file system caching

See the list of I/O methods used when file system caching is disabled for IBM DB2 table spaces.

Table compression candidates for the IBM Tivoli Directory Server database

IBM Tivoli Directory Server database can use row-level compression with DB2, version 9.

Because building a compression dictionary requires the tables to have data, compression is not enabled by default.

The following tables are good compression candidates for IBM Tivoli Identity Manager:

- ldap_entry
- objectclass
- erparent
- erservice
- erroles
- owner
- manager
- secretary

Because each LDAP attribute is stored in a separate table, there are many possible candidates. A good candidate depends on the composition of person and account objects in your environment. Other attributes to consider:

- street
- l (location)
- st (state)
- title
- description
- erlostpasswordanswer
- erchangepwdrequired

- `mobile`, `telephonenumber`, and `facsimileTelephoneNumber` (if your users have a common set of area codes or prefixes)

Tivoli Directory Server, version 6.2, includes the `idsdbmaint` command. This command automatically evaluates tables for compression and compresses good candidates.

Related tasks

“Configuring row-level compression” on page 41

Row-level compression decreases the on-disk footprint of the database. It also improves performance by decreasing I/O wait. It improves buffer pool usage even with the additional CPU usage required by compression.

Configuring transaction logs for the Tivoli Directory Server database

DB2 keeps logs during transaction processing. During large transactions, the default log number and sizes might be too small and cause transaction rollbacks. Increase the size and number of log files available to DB2.

About this task

DB2 has the following types of log files:

Primary logs

Are allocated when the database is started and remain allocated until the database is stopped.

Secondary logs

Are allocated as needed when the primary logs are full and released when they are no longer needed.

For best performance, move the transaction logs to a different physical drive than the database. Intelligent data storage devices might not require a different physical drive.

Increase the number of secondary logs to prepare for large transactions. The default size of the log files is 1000 4 KB pages (4 MB). Increase the size to 10000 4 KB pages (40 MB). Increasing the default changes the size of both primary and secondary log files.

Use the following variables when configuring logs:

ldap_database

Specifies the name of the Tivoli Directory Server database, such as `ldapdb2`.

logs_secondary

Specifies the number of secondary logs. Suggested value: 12.

logs_size

Specifies the size of the primary and secondary logs in 4 KB pages. Suggested value: 10000.

log_path

Specifies the path to where the transaction logs are located.

Procedure

1. Connect to the database as the database administrator.
2. Run the following commands on separate lines:

```
db2 update db cfg for ldap_database using logsecond logs_secondary
db2 update db cfg for ldap_database using logfilsiz logs_size
db2 update db cfg for ldap_database using newlogpath log_path
```

3. Stop and restart the database instance. The changes take effect when the database instance restarts.

Configuring database statement heaps

You can increase the size of the DB2 statement heap (*stmtheap*) to eliminate errors caused by long queries.

About this task

IBM Tivoli Identity Manager can submit long LDAP queries to the Tivoli Directory Server. Some queries might not fit in the DB2 statement heap (*stmtheap*). Tivoli Directory Server returns an error to IBM Tivoli Identity Manager.

The statement heap is allocated per agent (connection). Increasing this value can dramatically increase the memory used by the DB2 server.

Use the following variables to configure database statement heaps:

ldap_database

Specifies the name of the Tivoli Directory Server database, such as *ldapdb2*.

stmtheap_size

Specifies the value of *stmtheap* in 4 KB pages. Default value: 2048.
Suggested value: 4096.

Procedure

1. Connect to the database as the database administrator.
2. Run the following command:

```
db2 update db cfg for ldap_database using stmtheap stmtheap_size
```
3. Stop IBM Tivoli Directory Server
4. Stop and restart the IBM Tivoli Directory Server database.
5. Start IBM Tivoli Directory Server.

Configuring system limits

System limits (*ulimits*) might prevent the Tivoli Directory Server process from accessing enough real or virtual memory. To avoid memory dumps or stopping without indication, increase the *ulimits*.

About this task

Use the following variables to configure system limits:

process_data_size

Specifies the maximum data segment size for the process. Minimum value: 256000 (256 MB). Suggested value: unlimited.

virtual_mem_size

Specifies the maximum virtual memory size for the process. Minimum value: 256000 (256 MB). Suggested value: unlimited.

Procedure

1. Connect to the database as the user who starts the Tivoli Directory Server process.
2. (AIX only) Update the `/etc/security/limits` file:
 - a. In `/etc/security/limits`, locate the stanza for the user who starts the Tivoli Directory Server process.
 - b. If the stanza does not exist, add it.
 - c. Change the data limit to `process_data_size` or to `-1` for unlimited. If the limit setting is not there, add it.
 - d. Change the rss limit to `virtual_mem_size`, or to `-1` for unlimited. If the limit setting is not there, add it.
 - e. Log out of the current session and log back in for the changes to take effect.
3. (Solaris only): Run the following commands before starting `ibmslapd` or place them into the shell startup files for the user:

```
ulimit -d process_data_size
ulimit -v virtual_mem_size
```

Related information



Adjusting user process resource limits for ITDS

See more information about setting the `ulimit` correctly for IBM Tivoli Directory Server.

Configuring attribute indexes for Tivoli Directory Server

Indexing the attributes on which applications search increases Tivoli Directory Server performance. Tivoli Directory Server indexes automatically translate into DB2 indexes when you update the Tivoli Directory Server schema for those attributes.

About this task

Index those attributes on which you intend to search, if you extend the LDAP schema in Tivoli Directory Server to include additional attributes. Any filter in IBM Tivoli Identity Manager (such as with dynamic roles) is translated into a search string for the Tivoli Directory Server.

Tivoli Directory Server reports messages in the `ibmslapd.log` file for attributes on which a search was run and the attributes were not indexed. Consider indexing attributes that have more than 100 searches.

IBM Tivoli Identity Manager provides the `perfanalyze_indexes.pl` script in its performance scripts. You can use it to find attributes that were searched, but not indexed. The script can generate an LDIF that you can use to index the attributes. See the documentation that comes with the performance scripts for detailed information about using `perfanalyze_indexes.pl`.

Use the following variables when configuring attribute indexes:

`root_dn`

Specifies the root DN of the IBM Tivoli Directory Server server.

`root_password`

Specifies the password for the root DN.

Procedure

1. Use the `perfanalyze_indexes.pl` script to create an LDIF to index the attributes. For example: `perfanalyze_indexes.pl -i audit.log -d /home/idsinst/idsslapd-idsinst/etc -l indexes.ldif`
2. Edit the resulting `indexes.ldif` file to remove any stanzas for attributes you do not want to index.

Tip: Indexes add additional overhead for update events. Not every attribute needs an index.

3. Run the following command to import the LDIF into IBM Tivoli Directory Server: `ldapmodify -D root_dn -w root_password -f indexes.ldif`
4. After updating the LDAP schema, run `RUNSTATS` on the database to update the statistics for the newly created indexes.

Related information



ITIM performance tuning scripts

Download performance tuning scripts for IBM Tivoli Identity Manager.

Configuring DB2 indexes

Adding indexes directly to specific tables in the underlying DB2 database can improve performance for some Tivoli Directory Server queries.

About this task

The following indexes improve search performance for some queries and have been included in later versions of IBM Tivoli Directory Server.

- LDAP_DESC (AEID ASC, DEID ASC)
- OBJECTCLASS (EID ASC, OBJECTCLASS ASC)
- OBJECTCLASS (OBJECTCLASS ASC, EID ASC)
- LDAP_ENTRY (PEID ASC, EID ASC)

Use the following variable when configuring DB2 indexes:

schema_name

Specifies the schema for IBM Tivoli Directory Server tables.

Procedure

1. Connect to the database as the database administrator.
2. Run the following commands on separate lines:

```
db2 'create index schema_name.LDAP_DESC_DEID
on schema_name.LDAP_DESC ("AEID" ASC, "DEID" ASC)
MINPCTUSED 10 ALLOW REVERSE SCANS'

db2 'create index schema_name.OBJECTCLASS_EOC
on schema_name.OBJECTCLASS ("EID" ASC, "OBJECTCLASS" ASC)
MINPCTUSED 10 ALLOW REVERSE SCANS'

db2 'create index schema_name.OBJECTCLASS2
on schema_name.OBJECTCLASS ("OBJECTCLASS" ASC, "EID" ASC)
MINPCTUSED 10 ALLOW REVERSE SCANS'

db2 'create index schema_name.LDAP_ENTRY_PEID3
on schema_name.LDAP_ENTRY ("PEID" ASC, "EID" ASC)'
```

Some of these indexes might exist, possibly with different names. If there are "unable to create index" errors, you can ignore them as duplicates.

Configuring automatic statistics collection for the Tivoli Directory Server database

Administrators can use automatic statistics collection so that DB2 automatically updates the necessary database statistics. Automatic collection eliminates the necessity of manually running a periodic statistics collection against the database.

Before you begin

Enabling automatic statistics collection for Tivoli Directory Server database requires the creation of a DB2 administration server on the system to connect to it using the DB2 Control Center.

About this task

Automatic statistics collection is not enabled by default. For Tivoli Directory Server to operate properly, you must exclude the `LDAP_DESC`, `LDAP_ENTRY`, and `ERPARENT` tables from the automatic statistics collection. You must also exclude any other tables with artificial cardinalities.

For newly created databases, run manual statistics collection (`RUNSTATS`) after a small data load, even if automatic collection is enabled. `RUNSTATS` provides statistics for good performance until DB2 initiates the first automatic collection.

Use the following variable when configuring and enabling automatic statistics collection:

tds_database_name

Specifies the name of the Tivoli Directory Server database, such as `itimldap`.


Procedure


1. Use DB2 Control Center to update the DB2 maintenance policies:
 - a. Start the DB2 Control Center on a remote machine.
 - b. Connect to your database with database administrator authority. If you do not see your database in Control Center, then add it to the catalog before you continue.
 - c. Browse to *tds_database_name*.
 - d. Right-click *tds_database_name*.
 - e. Click **Configure Automatic Maintenance**.
 - f. Click **Next** until you reach **Activities**.
 - g. Select **Optimize data access (RUNSTATS)**.
 - h. Click **Configure Settings**.
 - i. Click **Selected tables**.
 - j. Select **Use the custom filter**.
 - k. At **Conditions**, type:
`TABNAME NOT IN ('LDAP_DESC', 'LDAP_ENTRY', 'ERPARENT')`
 - l. Click **Refresh Resulting Tables**.
 - m. Confirm that **Resulting tables (SCHEMA.NAME)** is populated with all tables other than the ones that you specified.
 - n. Click **OK** to accept the configuration.
 - o. Click **Finish** to complete the wizard.

- p. Confirm the message that no errors were encountered executing the command.
 - q. Quit Control Center.
2. Enable automatic statistics collection:
 - a. As the database administrator, connect to the database at the command prompt.
 - b. Run the following command:


```
db2 update db cfg for tds_database_name using auto_runstats on
```

Related information

 [Control Center overview](#)
See the information about using IBM DB2 Control Center.

 [RUNSTATS command](#)
See the information about using the RUNSTATS command.

Updating Tivoli Directory Server database statistics

DB2 requires information about the number of rows in the tables and what indexes are available so that it can efficiently fulfill queries. If Tivoli Directory Server database is running DB2, version 9, you can set RUNSTATS to run automatically. Version 9 is the default for Tivoli Directory Server, version 6.1. RUNSTATS eliminates the need for running it manually.

About this task

Note: DB2 REORGCHK does not update index statistics and is not a replacement for RUNSTATS.

If enabling automatic statistics collection is not feasible, you must run RUNSTATS manually. It is important to update table and index statistics after large Directory Server Markup Language (DSML) loads, HR feeds, and reconciliations.

If you experience high CPU usage or poor DB2 performance, run RUNSTATS on all of the tables in the database. To update index statistics, run the RUNSTATS command on each table individually. IBM Tivoli Identity Manager performance tuning scripts (`perftune_runstats.sh` and `perftune_runstats.bat`) detect the version of DB2 and run the RUNSTATS command against all tables for a specific schema in a database.

If you run the RUNSTATS command in a working environment, make sure that the connected applications can continue to write to the database. Use the `allow write access` option so users can write to a database while RUNSTATS runs.

Use RUNSTATS on an idle or lightly used database because it requires update locking on the system statistics table to update the database statistics. The system acquires locks on the tables that are used by the database optimizer to fulfill queries. The locks might cause transaction rollbacks on a database with a heavy load.

In addition to running RUNSTATS on all tables in the database, you must manually update the statistics table for the LDAP_DESC, LDAP_ENTRY, and ERPARENT tables. The choices DB2 makes about when to use these tables for fulfilling queries for the Tivoli Directory Server are not ideal for IBM Tivoli Identity Manager. Manually adjusting the statistics table helps DB2 make better choices and use these tables at the end of the access plan instead of the beginning.

The following procedure uses RUNSTATS on every table in the ITIMLDAP schema

Procedure

1. Connect to the database as the database administrator.
2. Generate a listing of all tables in the schema by running the following command:

```
db2 list tables for all | grep ITIMLDAP
```

3. For each table in the ITIMLDAP schema, run the following command on a single line:

```
db2 runstats on table ITIMLDAP.table_name  
and indexes all allow write access
```

4. Manually update the database statistics table for the LDAP_DESC, LDAP_ENTRY, and ERPARENT tables. Run the following commands on separate lines:

```
db2 update sysstat.tables  
set card = 9E18  
where tabname = 'LDAP_DESC' and card <> 9E18  
db2 update sysstat.tables  
set card = 9E18  
where tabname = 'LDAP_ENTRY' and card <> 9E18  
db2 update sysstat.tables  
set card = 9E10  
where tabname = 'ERPARENT' and card <> 9E10
```

Related tasks

“Configuring automatic statistics collection for the Tivoli Directory Server database” on page 73

Administrators can use automatic statistics collection so that DB2 automatically updates the necessary database statistics. Automatic collection eliminates the necessity of manually running a periodic statistics collection against the database.

Related information

 ITIM performance tuning scripts

Download performance tuning scripts for IBM Tivoli Identity Manager.

Configuring the maximum open files

To work well with other applications running on the system, DB2 sets a limit on the number of files it keeps open with the `maxfilop` setting. You can adjust this number to meet the needs of your environment.

About this task

After reaching the specified limit, DB2 closes a currently open file to open the new one. This process can cause a performance loss on systems that do not require a restriction on the number of open files. The default value is often too small, particularly for larger directories.

Increasing this value is important for the SMS table spaces that IBM Tivoli Directory Server uses.

Use the following variables when configuring the maximum open files:

itds_database

Specifies the name of the Tivoli Directory Server database, such as `ldapdb2`.

max_files_open

Specifies the maximum number of files DB2 can have open at any one time. Initial value: 64. Suggested value: 256.

Procedure

1. Connect to the database as the database administrator.
2. Run the following command:

```
db2 update db cfg for itds_database using maxfilop max_files_open
```

Disabling hash joins

The DB2 optimizer can use several different join types when determining the most efficient means to fulfill a query.

About this task

For Tivoli Directory Server, a hash join is seldom, if ever, the correct approach. This procedure describes how to disable hash joins.

Procedure

1. As the database administrator, run the following command:

```
db2set DB2_HASH_JOIN=NO
```
2. Stop and restart the IBM Tivoli Directory Server database. The changes take effect when the database restarts.

Improving disk I/O performance

Disk I/O performance is highly dependent upon the drive types, layout, and configuration.

About this task


The following DB2 registry variables might improve performance on some systems. See the DB2 documentation to find out if the setting applies to your environment.

| System | Setting |
|--|------------------------------------|
| Systems with SAN, RAID, or other advanced disk subsystem | DB2_PARALLEL_IO=* |
| All systems | DB2_USE_ALTERNATE_PAGE_CLEANING=ON |

Tuning Sun ONE Directory Server

With the Sun ONE Directory Server, you can improve performance by tuning the following areas: indexing, the All IDs Threshold value, and cache sizes.

Related information

 [Sun ONE Directory Server V5.2 documentation](#)
See product information about tuning the IBM Tivoli Directory Server.

Configuring the All IDs Threshold value

The Sun ONE Directory Server sets an upper boundary on the number of index entries for each indexed attribute with the All IDs Threshold value. Tuning this value is a trade-off between update performance and search performance.

About this task

If the specified boundary value is exceeded, the index is invalidated, no longer maintained, and not used during searches. This behavior prevents the server from including all entries in an index and improves update performance at the expense of search performance.

If you set the value too low:

- Update performance is good. There is less index maintenance.
- Search performance might suffer for queries against attributes whose indexes are no longer maintained.

If you set the value too high:

- Search performance is good. Attribute indexes can be used to fulfill the query.
- Update performance might suffer because of index maintenance.

The *Sun ONE Directory Server Installation and Tuning Guide* suggests setting this value to 5% of your directory size. For optimum search performance in IBM Tivoli Identity Manager, increase this value. Use the value that includes largest number of users, groups, services, or accounts on a specific service type in the directory. IBM Tivoli Identity Manager does LDAP searches against `objectclass` for these objects.

Set the value based on the estimated size of your directory in its final form. If the current size of your directory is smaller than the expected final size, use the larger value to calculate the All IDs Threshold value. See the *Sun ONE Directory Server Installation and Tuning Guide* for more information about adjusting the All IDs Threshold value.

Changing this value can disrupt a production system, because you must:

- Take the system offline.
- Export the data.
- Import the data.
- Bring the system online.

Important: Data loss or corruption can occur if you perform this procedure incorrectly. Back up the information in your directory before setting this value.

Use the following variables when configuring the All IDs Threshold value:

directory_home

Specifies the Sun ONE Directory Server home directory, such as `/usr/iplanet/servers/slaped-gso-2DS`.

number_of_entries

Specifies the number of entries in your directory. To estimate this number, multiply the number of IBM Tivoli Identity Manager users by the average number of accounts for each user.

suffix_name

Specifies the name of the root suffix, such as `dc=com`.

database_name

The name of the directory server database, such as `itim5145`.

temp_directory

Specifies the name of a temporary directory, such as `/tmp`. This directory

must have enough free space to store an LDIF of your entire directory. Allocate 1 KB per IBM Tivoli Identity Manager user.

Increasing the size of the All IDs Threshold cause the indexes to grow. Because the indexes are stored in the database cache, you might need to increase the cache size.

Procedure

1. Access the *directory_home/config/dse.ldif* file.
2. Find the following stanza:
`dn: cn=config,cn=ldbm database,cn=plugins,cn=config`
3. Change the following attribute:
`nsslapd-allidsthreshold: number_of_entries`
4. Export your data:
`db2ldif -n database_name -a temp_directory/export.ldif -s "suffix_name"`
5. Stop the directory server.
6. Import your data:
`ldif2db -n database_name -i temp_directory/export.ldif`
7. Start the directory server.

Related tasks

“Configuring cache sizes” on page 79

You must tune both the database and the entry caches for the Sun ONE Directory Server, version 5.2.

Configuring attribute indexes for Sun ONE Directory Server

You can increase Sun ONE Directory Server performance by indexing the attributes on which applications search.

About this task

Index those attributes on which you want to search if you extend the LDAP schema to include additional attributes. Any filter in the IBM Tivoli Identity Manager application (such as with dynamic roles) is translated into a search string for the LDAP server.

You can use the LDAPConfig tool to configure IBM Tivoli Identity Manager with a subsuffix (such as `dc=oak,dc=com`) of an existing root suffix (such as `dc=com`). The indexes added by LDAPConfig are applied to the root suffix, not the subsuffixes. When using subsuffixes, you must manually apply all indexes that LDAPConfig typically creates. See *itim_home/config/ldap/iplanet/er-indexes.conf* for the indexes that LDAPConfig creates.

Procedure

1. Open the Directory Server Console.
2. Select **Configuration**.
3. Expand the **Data** node.
4. Expand the suffix of your database and select the database.
5. Select **Indexes**.
6. Click **Add attribute**.
7. Select the attribute you want to index and click **OK**.
8. If the **Equality** and **Presence** options are not selected for the attribute, select them.

9. Click **Save**. The software displays a progress dialog.
10. Click **Close**.

Configuring cache sizes

You must tune both the database and the entry caches for the Sun ONE Directory Server, version 5.2.

About this task

The Sun ONEDirectory Server has the following caches:

Database cache

Stores database-level entries and indexes. It is used for both subtree and one-level searches. It is also used for base-level searches if the results are not found in the entry cache.

Entry cache

Stores formatted entries. It is used only for base-level searches and provides no benefit for subtree or one-level searches.

A typical IBM Tivoli Identity Manager system does about 50% base searches and 50% one-level and subtree searches. For this reason, allocate no more than 50% of your total cache memory to the entry cache. For larger directories, you might see better performance by using more memory in the database cache than in the entry cache.

Use the following variables when configuring cache sizes:

db_cache_size

Specifies the size of the database cache. This variable is the `nsslapd-dbcachesize` parameter in the Sun ONE documentation. Suggested value: 512000000 (512 MB) or higher if you have the available memory.

entry_cache_size

Specifies the size of the entry cache. This variable is the `nsslapd-cachememsize` parameter in the Sun ONE documentation. Suggested value: 512000000 (512 MB) or higher if you have the available memory.

Setting these values greater than the amount of physical RAM available causes performance degradation because the system swaps the pages out to disk. Consult the Sun ONE documentation before increasing the parameters beyond the suggested values.

Procedure

1. Open the Directory Server Console.
2. Select **Configuration**.
3. Select **Performance**.
4. Select **Caching**.
5. Set the **Database Cache size** to *db_cache_size*.
6. Select the suffix of your database.
7. In **Entry cache**, set the **Size** parameter to *entry_cache_size*. Leave the maximum number of entries at -1.
8. Click **Save**.
9. Restart the server.

Configuring the referential integrity plug-in

You can improve update and delete performance by tuning the referential integrity plug-in parameters and associated attributes.

Before you begin

Make sure that all attributes used by the referential integrity plug-in are indexed for equality.

About this task

By default, referential integrity enforcement occurs immediately after every update or delete. You can improve performance by changing enforcement to every few seconds, but you risk having stale data during this timeframe. If you change from the default value of 0 (immediate), keep the interval small (60 seconds or less).

Use the following variable when configuring the referential integrity plug-in

enforcement_delay

Specifies how frequently referential integrity enforcement occurs. Default value: 0 (immediately following an update or delete). Suggested value: 60 seconds or less.

Procedure

1. Open the Directory Server Console.
2. Select **Configuration**.
3. Expand **Plugins**.
4. Select the **Referential integrity postoperation** plug-in.
5. Set **Argument 1** to *enforcement_delay*.
6. Click **Save**.
7. Restart the server.

Related tasks

“Configuring attribute indexes for Sun ONE Directory Server” on page 78

You can increase Sun ONE Directory Server performance by indexing the attributes on which applications search.

Chapter 11. Improving operating system performance

You can improve performance on some systems by making some operating system-specific changes. This information serves only as a guideline. Consult the documentation for your middleware, and apply any required operating system tuning.

AIX

You might improve performance by tuning the virtual memory-management (VMM) settings such as `minperm` and `maxperm`. Consult the AIX documentation for more information.

Ensure that there is at least as much swap space as there is physical RAM on the system. Insufficient swap space can result in out of memory messages due to how the operating system handles memory allocations.

Enable Large File support for all file systems using Journaled File Systems (JFS). Large File support is not required for file systems using Enhanced Journaled File Systems (JFS2) as JFS2 supports large files natively.

Solaris

Ensure that there is at least as much swap space as there is physical RAM on the system. Insufficient swap space can result in out of memory messages due to how the operating system handles memory allocations.

Chapter 12. Best practices

You can set up and configure IBM Tivoli Identity Manager in many ways. Use this information to determine the best configuration for your environment.

Table 2. Hardware best practices

| Consideration | Best practice |
|--|--|
| Database and directory activity can be CPU- and memory-intensive. | Allocate each application at least 1 processor and 2 GB of RAM. More processors are better. For optimal performance, do not have all IBM Tivoli Identity Manager components on the same system. |
| In general, network latency is not a major performance bottleneck, but components can degrade performance. Components include the IBM Tivoli Identity Manager server components, the directory server, database server, agents, and agent endpoints. | Try to have as few hops as possible between components. If possible, install all components on the same subnet or no more than one hop away. Put components on a 100 megabit or faster network. |
| Allocation of CPU resources in an LPAR can affect system performance. | Suggested actions listed in order of potential benefit. <ul style="list-style-type: none"> • Disable SMT for IBM Tivoli Identity Manager nodes. • Use the most current version of the WebSphere Application Server JVM. SR6 improves performance on LPARs. • Give at least one physical CPU to each LPAR. • Use dedicated processors rather than virtual ones. |
| Disk bottlenecks can negatively affect performance. | Use multiple disks rather than a single large disk. <ul style="list-style-type: none"> • IBM DB2 and Oracle can use multiple disks, but you must configure them to do so. • High-end I/O backplanes or other advanced storage systems can balance the I/O load across multiple disks automatically. |
| Do not use physical disks in a SAN failover environment across multiple data stores. For example: If the database for each LDAP server is on the same physical devices, I/O performance problems are likely to develop. | Use separate physical devices in the SAN for the underlying data store of each failover. |

Table 3. Software best practices

| Consideration | Best practice |
|--|--|
| Each agent modifies the LDAP schema by adding new attributes to support a new service. These attributes are created without indexes | For services that manage thousands of users, you realize significant benefit by adding indexes to attributes that have many members. |
| Complicated provisioning policies can result in complicated directory and database queries with poor performance. | Policies with small numbers of roles and services perform best. |
| Provisioning policies without account approval work flows perform better than policies with account approval workflows due to optimizations for the former case. Provisioning policies created by the system when a service is created use a Default Account Request Workflow. | If it is not needed, remove the default account workflow from the provisioning policy to improve performance. |

Table 3. Software best practices (continued)

| Consideration | Best practice |
|---|--|
| <p>Dynamic roles affect people in a given scope, either one-level or subtree. When a person object in that scope is modified or added, the system must reevaluate that role. This process is true for every dynamic role in the system. For example, if there are three dynamic roles with subtree scope and a person object in that scope is updated, the system must reexamine all three dynamic roles.</p> | <ul style="list-style-type: none"> • Limit the number of dynamic roles, either by number or by scope, that affect person objects that are modified frequently. It does not matter if the dynamic role ultimately enrolls the person or not: the evaluation affects the performance. • When creating dynamic roles that apply to all people within an organizational unit, place the dynamic role inside the organizational unit and use the filter (objectclass=*). This filter yields better performance from the directory server than a filter like (cn=*). |
| <p>When creating a role hierarchy, the order in which the hierarchy is created can affect performance due to any associated provisioning policies being enforced.</p> | <p>When adding multiple new roles to an existing role hierarchy, create the parent-child relationship between all new roles first. Then, create the parent-child relationship between the new role and the existing ones. This process limits the number of policy evaluations. If possible, create the entire hierarchy before adding any of the involved roles to a provisioning policy.</p> |
| <p>Evaluation of ACIs affects performance.</p> | <ul style="list-style-type: none"> • Limiting the scope (through placement within the organizational tree) and number of ACIs increases performance by requiring fewer evaluations. • When doing a person search through the APIs, limit the scope of your search to be as narrow as possible. Limiting the scope prevents unnecessary evaluations. |
| <p>When updating a person object, the system must reevaluate all provisioning policies in which the user to see if the update changes a provisioning action. The guideline applies to both manual or automated methods such as an HR feed or JNDI update.</p> | <ul style="list-style-type: none"> • Store only the person information that is needed for policy evaluation and account management in IBM Tivoli Identity Manager to reduce attribute updates that are not used for policy enforcement. • Minimize person object updates when possible. |
| <p>IBM Tivoli Identity Manager includes searches on the O, OU, and L attributes for the organizational chart. This search can slow down if large numbers of users have these attributes. This consideration is important for large user populations.</p> | <p>When loading users in bulk, do not include O, OU, or L attributes on the person records. Follow this guideline when using a DSML file or an IDI Feed.</p> |
| <p>Numeric erGlobalIDs allows the application to make more efficient use of memory when processing reconciliations.</p> | <p>When loading objects such as people or accounts directly into the IBM Tivoli Identity Manager directory server, such as during an initial LDIF data load, use all numeric values for the erGlobalID , not an alphanumeric value.</p> |
| <p>Having the same value for the family name (sn) attribute for all users in a test environment results in poor performance. This attribute is used by the default identity policy to determine a unique UID for an account. Therefore, performance degrades when the identity policy creates an ID for a service due to the resulting LDAP lookups.</p> | <p>When loading a test environment, make sure that users have unique IDs for their family name (sn) attribute.</p> |
| <p>Often administrative accounts on target systems are mapped to a single person object. This mapping can result in one person with possibly thousands of accounts. It can degrade performance or result in Java OutOfMemory errors.</p> | <p>Limit the number of accounts that person objects have, ideally to no more than 1200 accounts. The ability to scale beyond this threshold depends on the system configuration and hardware.</p> |
| <p>Complex workflows (operational, account request, and access request workflows) can degrade performance.</p> | <p>Keep frequently used workflows, such as the modify person operational workflow, as simple as possible.</p> |

Table 3. Software best practices (continued)

| Consideration | Best practice |
|--|--|
| In a workflow, each transition results in a message being placed on the JMS queue. Transitions also serialize and deserialize data from the database. | Design workflows so that they have the fewest number of transitions from start to finish as possible. Consider reducing the number of nodes by: <ul style="list-style-type: none"> • Combining adjacent script nodes • Combining a non-script node followed by a script node by moving the script node contents into the PostScript for a non-script node, • Creating an initial node at the beginning of a long workflow to jump to a specific node later to shortcut unnecessary transitions for common paths |
| Doing several poorly performing non-cached LDAP lookups within a workflow can negatively affect performance. | Store the results of redundant lookups as relevant data items for reuse in later nodes. |
| Relevant data must be serialized and deserialized from the database for each node transition. | Keep the quantity and size of workflow relevant data objects as small as possible. |
| Each call to process.auditEvent() adds more data that must be written to the database. | Minimize the amount of logging done in workflow nodes. Consider using process.auditEvent() calls during development and testing but comment out these lines before promoting the code into production. |
| HR feeds for IBM Tivoli Directory Integrator use the following types: Push feed Tivoli Directory Integrator uses the IBM Tivoli Identity Manager JNDI unsolicited notification feature to push records into IBM Tivoli Identity Manager. The push method is single-threaded. It requires that IBM Tivoli Identity Manager confirm the JNDI operation completed successfully before proceeding to the next object. Pull feeds IBM Tivoli Identity Manager requests all available records from a Tivoli Directory Integrator DSML version 2 Event Handler assembly line through a reconciliation. This method streams all objects directly into IBM Tivoli Identity Manager. Any available cluster member can then act on the update operations. | To ensure optimal HR feed performance, use the pull method wherever possible. Use the push method for asynchronous updates or updates that are not performance-sensitive. |
| Environment stability can be compromised by having other applications deployed in the same WebSphere JVM where IBM Tivoli Identity Manager is deployed. | When installing into a shared WebSphere environment, install IBM Tivoli Identity Manager into an existing cell or node but on a separate application server. You can tune the application server without affecting other applications. |
| Using an existing instance when installing into a shared DB2 environment might limit tuning possibilities and negatively affect other databases in the instance. This limitation includes the instance for Tivoli Directory Server. | Use a separate instance for the IBM Tivoli Identity Manager database to yield the best performance. |

Related information



IBM Tivoli Identity Manager best practices wiki

See the information about best practices for IBM Tivoli Identity Manager.

Chapter 13. Planning a maintenance schedule

Perform regular maintenance to maintain optimal performance for IBM Tivoli Identity Manager environment.

About this task

You can find the latest up-to-date best practices on the IBM Tivoli Identity Manager wiki at <http://www.ibm.com/developerworks/wikis/display/tivoliim/Regular+Maintenance>. These suggestions are from that wiki. To maintain your environment, perform the following tasks at appropriate intervals:

Procedure

- Clean out the recycle bin

If enabled, regularly empty the IBM Tivoli Identity Manager recycle bin. As the number of objects in the recycle bin increase, LDAP performance can degrade. The frequency with which you empty the recycle bin depends on how frequently deletes occur in the system. Disable the recycle bin for systems that do not need it.
- Update database statistics

Update database statistics after many updates or on a weekly basis for most environments. Updating database statistics in the underlying databases can significantly improve performance. This maintenance task applies to the DB2 or Oracle Database used by IBM Tivoli Identity Manager. It also applies to the DB2 database used by the Tivoli Directory Server. Consider enabling automatic RUNSTATS if your environment meets the software and configuration criteria.
- Clean out the IBM Tivoli Identity Manager database

Keep the IBM Tivoli Identity Manager database as small as possible for efficient database access. Cleaning involves regular purging of database records that are no longer required for auditing or transactional purposes.
- Evaluate and apply fixes

IBM releases software updates on a regular basis for IBM Tivoli Identity Manager and its supported middleware. Check for updates on a quarterly basis to ensure that your environment is up to date. Test fixes thoroughly in a test environment before applying them to the production environment. Consider the information IBM provides about compatibility before applying updates.
- Access the latest version of the tuning guide

Just like software updates, the tuning guide is updated on a regular basis with new tuning information to improve performance. Check the IBM website every quarter to see if a new tuning guide has been released. Always test a new tuning change in a test environment, including load testing, before applying it to the production environment. Look for the latest tuning scripts, which also undergo revisions, from IBM.
- Take regular backups

Backups do not contribute towards performance of the IBM Tivoli Identity Manager environment, but perform them as part of regular maintenance.

Related concepts

“Using the recycle bin” on page 22

When you enable the recycle bin and then delete objects from IBM Tivoli Identity Manager, the software moves them to the recycle bin.

Related tasks

“Updating IBM Tivoli Identity Manager database statistics for DB2 databases” on page 51

DB2 requires statistics on the number of rows in the tables and available indexes to efficiently execute queries. DB2 version 9 can update the statistics automatically, or you can manually update the statistics.

“Updating Tivoli Directory Server database statistics” on page 74

DB2 requires information about the number of rows in the tables and what indexes are available so that it can efficiently fulfill queries. If Tivoli Directory Server database is running DB2, version 9, you can set RUNSTATS to run automatically. Version 9 is the default for Tivoli Directory Server, version 6.1. RUNSTATS eliminates the need for running it manually.

“Updating IBM Tivoli Identity Manager database statistics for Oracle databases” on page 59

You must gather and update database statistics at regular intervals. Intervals can be one week to one month on a production IBM Tivoli Identity Manager system or after processing a large amount of data.

“Configuring automatic statistics collection for the IBM Tivoli Identity Manager database” on page 49

Administrators can configure automatic statistics collection so that DB2 automatically updates database statistics. Automatic collection eliminates the necessity of manually running a periodic statistics collection against the database.

“Configuring automatic statistics collection for the Tivoli Directory Server database” on page 73

Administrators can use automatic statistics collection so that DB2 automatically updates the necessary database statistics. Automatic collection eliminates the necessity of manually running a periodic statistics collection against the database.

“Controlling the size of the database” on page 29

To maintain optimum performance, use the DBPurge utility included with IBM Tivoli Identity Manager to automate removing entries over a certain age from the database.

Chapter 14. Troubleshooting IBM Tivoli Identity Manager

Middleware dependencies can complicate the task of finding performance problems with IBM Tivoli Identity Manager. For example, a slow DSML feed with account provisioning might be caused by a slow directory server, database locking, or insufficient worker threads.

This information is designed to assist you in identifying problem areas and provide some pointers on fixing them. Information is provided with the assumption that you have read and applied the tuning.

Sun ONE Directory Server slow query performance

Slow queries from Sun ONE Directory Server can degrade overall system performance.

Symptoms

Poor search performance when using Sun ONE Directory Server.

Causes

You might see poor performance when:

- Long-running queries need indexes.
- The All IDs Threshold value is too small.
- The database cache is too small.

Diagnosing the problem

Determine the specific cause or causes for poor search performance:

Long-running queries

Queries typically take less than one second. Queries taking longer than a second might be searching on attributes that are not indexed.

Follow these steps to find queries taking longer than a second.

1. Search the Sun ONE Directory server access logs for `etime=X` where `X` is an integer greater than 1. It represents the number of seconds that the query took to run.
2. Look for queries running 2 seconds or longer.

Example:

```
[26/Mar/2004:12:54:25] conn=4236 op=1 msgId=2 -  
RESULT err=0 tag=101 nentries=1 etime=2
```

Identify the query associated with the search time. Search the same file for the matching connection and operation number, which in the preceding example is `conn=4236 op=1`. The query for this example was:

```
[26/Mar/2004:12:54:24] conn=4236 op=1 msgId=2 -  
SRCH base="ou=ibm,dc=com" scope=2 filter="(&(!(erIsDeleted=y))  
(&(erEnabled=true)(erPolicyMembership=2;*)))  
(objectClass=erProvisioningPolicy)" attrs=ALL
```

3. Examine the LDAP query string and identify the attributes being searched.

In the example, they are `erIsDeleted`, `erEnabled`, `erPolicyMembership`, and `objectClass`.

All IDs Threshold value

If the All IDs Threshold value is too small, Sun ONE Directory Server might not be using the indexes for certain attributes.

Determine if queries are searching on these attributes by searching for `notes=U` in the Sun ONE Directory Server access logs.

The presence of this flag indicates that one or more attributes being searched on is over the All IDs Threshold. Consider increasing the threshold.

Database cache

If the Sun ONE Directory Server database cache is too small, the server must access the disk for the information. Accessing the disk results in a low database cache hit ratio. Identify the hit ratio for the database by viewing the Monitor tab for that database. An optimum hit ratio is 95% or higher.

Resolving the problem

Take the appropriate action or actions to improve query performance.

Long-running queries

Index all attributes being searched.

The `perfanalyze_audit.pl` script included with the performance scripts can analyze a Sun ONE access log. It also reports which queries are taking the longest.

All IDs Threshold value

Increase the value.

Database cache

If the hit ratio is lower than 95%, increase the size of your database cache.

Related tasks

“Configuring attribute indexes for Sun ONE Directory Server” on page 78
You can increase Sun ONE Directory Server performance by indexing the attributes on which applications search.

“Configuring the All IDs Threshold value” on page 76

The Sun ONE Directory Server sets an upper boundary on the number of index entries for each indexed attribute with the All IDs Threshold value. Tuning this value is a trade-off between update performance and search performance.

“Configuring cache sizes” on page 79

You must tune both the database and the entry caches for the Sun ONE Directory Server, version 5.2.

Related information



ITIM performance tuning scripts

Download performance tuning scripts for IBM Tivoli Identity Manager.

Tivoli Directory Server outages

Incorrect system or product configuration can cause Tivoli Directory Server to fail, hang, or disappear due to resource restrictions.

Symptoms

The directory server fails or hangs for no obvious reason.

Diagnosing the problem

Check the size of your entry cache. If the entry cache size causes the Tivoli Directory Server process to grow beyond what is supported by your operating system memory model it can fail. A typical system memory model is 2 GB on 32-bit operating systems.

The `ibmslapd` process might be hitting an artificial system limit, such as a `ulimit`.

Resolving the problem

Decrease the size of the Tivoli Directory Server entry cache, or increase the `ulimits` for the process.

Related tasks

“Configuring system limits” on page 70

System limits (`ulimits`) might prevent the Tivoli Directory Server process from accessing enough real or virtual memory. To avoid memory dumps or stopping without indication, increase the `ulimits`.

Tivoli Directory Server slow queries

Slow queries from IBM Tivoli Directory Server can degrade overall system performance.

Symptoms

Poor search performance when using IBM Tivoli Directory Server

Causes

You might see poor performance due to:

- Long-running queries that need indexes.
- Low buffer pool hit ratio.

Diagnosing the problem

Determine the specific cause or causes for poor search performance.

Long-running queries

Check for long-running queries that need indexes. The Tivoli Directory Server uses DB2 to process LDAP queries. By checking DB2 for long-running queries, you can discover what attributes need indexing.

1. To find how long each query takes, turn on statement cache monitoring in DB2.

```
db2 update dbm cfg using DFT_MON_STMT ON
```

2. Stop the directory server, restart the database, and restart the directory server.

3. After monitoring is turned on, duplicate the suspected action in IBM Tivoli Identity Manager.

4. Get a snapshot of the statement cache:

```
db2 get snapshot for dynamic sql on database_name
```

Example: The snapshot contains stanzas like this one:

```
Number of executions           = 1
Number of compilations         = 1
Worst preparation time (ms)    = 3
Best preparation time (ms)     = 3
Internal rows deleted          = 0
Internal rows inserted         = 0
Rows read                      = 10024
Internal rows updated          = 0
Rows written                   = 0
Statement sorts                 = 0
Total execution time (sec.ms) = 136.000663
Total user cpu time (sec.ms)   = 62.010000
Total system cpu time (sec.ms) = 10.000000
Statement text                  =
```

```
SELECT distinct E.EID
FROM LDAPDB2.LDAP_ENTRY AS E, LDAPDB2.LDAP_ENTRY as pchild
WHERE E.EID=pchild.EID AND pchild.PEID=?
AND E.EID IN (SELECT EID FROM LDAPDB2.OU WHERE OU = ?)
```

5. Calculate the average execution time per query. Divide the total execution time by the number of executions: total execution time / number of executions.

In the preceding example: $136 / 12 = 11.33$ seconds per execution.

Queries typically take one second or less. Queries that take longer might be searching on columns that are not indexed by DB2. If they are not indexed in DB2, they are not indexed in the Tivoli Directory Server.

Another symptom of this problem is a high average number of rows read, which is calculated by dividing the rows read by the number of executions. In the preceding example, the column OU is probably not indexed. IBM

Tivoli Identity Manager tuning scripts provide the `perfanalyze_dynamicsql.pl` script that calculates the time per execution for all stanzas and sorts the results.

Low buffer pool hit ratio

Tivoli Directory Server uses DB2 to process LDAP queries. Tivoli Directory Server database requires a high (greater than 95%) hit ratio. If the buffer pools are not large enough, DB2 must read more information from the disk. Reading the disk can result in high I/O wait.

See “Calculating the buffer pool hit ratio” on page 101.

IBM Tivoli Identity Manager tuning scripts provide the `perfanalyze_bufferpools.pl` script that calculates the hit ratio for all buffer pools.

Resolving the problem

Take the appropriate action or actions to improve query performance.

Long-running queries

Index any attribute in the Tivoli Directory Server that is not indexed. See “Configuring attribute indexes for Tivoli Directory Server” on page 71.

Low buffer pool hit ratio

Increase the memory allocated to the buffer pools. See “Configuring database buffer pools for the Tivoli Directory Server database” on page 66.

Related tasks

“Configuring attribute indexes for Tivoli Directory Server” on page 71

Indexing the attributes on which applications search increases Tivoli Directory Server performance. Tivoli Directory Server indexes automatically translate into DB2 indexes when you update the Tivoli Directory Server schema for those attributes.

“Configuring database buffer pools for the Tivoli Directory Server database” on page 66

DB2 buffer pools are the secondary buffer for Tivoli Directory Server. These buffer pools must be large enough so that most table searches can be read directly from memory instead of using the disk.

Related information



ITIM performance tuning scripts

Download performance tuning scripts for IBM Tivoli Identity Manager.

Governing policy search errors

Searches for governing policies can fail due to statement heap constraints.

Symptoms

The `trace.log` file contains the Error searching for governing policies message.

Causes

The statement heap for the Tivoli Directory Server database is too small, which causes large LDAP queries to fail.

Resolving the problem

Increase the statement heap.

Related tasks

“Configuring database statement heaps” on page 70

You can increase the size of the DB2 statement heap (stmtheap) to eliminate errors caused by long queries.

Java OutOfMemory errors

OutOfMemory errors can occur if the Java virtual machine heap is too small.

Symptoms

The trace.log file contains Java OutOfMemory errors.

Causes

The message is from WebSphere Application Server. The Java virtual machine (JVM) ran out of heap size.

Resolving the problem

Increase the maximum heap size if possible, and restart the application server. If the heap size is already at the limit, break up transactions. For example, you might use fewer services or roles in a provisioning policy.

Related concepts

“Using the DSML connector with Tivoli Directory Integrator” on page 36

You can use the DSML connector to create custom agents for returning information to IBM Tivoli Identity Manager.

Related tasks

“Adjusting the Java virtual machine size” on page 9

IBM Tivoli Identity Manager, version 5.0 and 5.1, runs on 64-bit JVMs on supported platforms. Using a 64-bit JVM, you can allocate 2 GB or more of memory. You might need to allocate more memory for very large (more than 6 million accounts) reconciliations.

“Configuring paged searches” on page 26

IBM Tivoli Identity Manager, version 5.0 and later, incorporates LDAP paged searches to alleviate JavaOutOfMemory errors in large environments.

Transaction rollback errors

Transaction rollback errors can occur due to database resource constraints.

Symptoms

The trace.log file contains transaction rollback errors.

Causes

Transaction rollbacks can occur for several different reasons, most of them database-related. An error message in the trace.log file can provide more information about what went wrong. Some areas to check when you get a transaction rollback:

- Lack of database storage space.
- Database locking issues.
- Database memory issues.

Diagnosing the problem

Determine the specific cause or causes for rollback errors.

Storage space

If the database runs out of storage space for the table spaces, a transaction rollback error can occur.

Locking

If the database encounters extreme locking issues, it might cause a transaction rollback error.

Memory

If there is not enough memory available to database structures to fulfill the requested query, a transaction rollback error might occur. The JNDI error in the trace.log file can indicate which database heap to increase.

Resolving the problem

Take the appropriate action or actions.

Storage space

Increase the amount of disk space allocated to the table spaces.

Locking

Confirm that the locks are tuned appropriately. Update the table statistics.

Memory

Increase the appropriate heap for the specific middleware.

Related concepts

“Adjusting lock list and maximum locks” on page 53

The default settings for the DB2 lock list (`locklist`) and maximum locks (`maxlocks`) are adequate for most environments.

“Changing the lock timeout” on page 53

The default lock timeout value (`locktimeout`) in the IBM Tivoli Identity Manager database is infinity. You can adjust this value if locking problems occur.

Related tasks

“Configuring table spaces for IBM DB2 databases” on page 44

IBM Tivoli Identity Manager uses a database managed space (DMS) table space to store data. This type of table space performs better than system managed space (SMS) table spaces, but you must preallocate disk space for the database to use. The tables spaces created by the installer have `autoresize` enabled and grow as needed.

“Configuring table spaces for Oracle databases” on page 56

During database configuration, IBM Tivoli Identity Manager creates several small table spaces that can automatically extend as necessary. You can add additional data files.

“Updating IBM Tivoli Identity Manager database statistics for DB2 databases” on page 51

DB2 requires statistics on the number of rows in the tables and available indexes to efficiently execute queries. DB2 version 9 can update the statistics automatically, or you can manually update the statistics.

“Updating IBM Tivoli Identity Manager database statistics for Oracle databases” on page 59

You must gather and update database statistics at regular intervals. Intervals can be one week to one month on a production IBM Tivoli Identity Manager system or after processing a large amount of data.

“Configuring database application heaps” on page 49

Some of the queries that the IBM Tivoli Identity Manager application submits to the DB2 server result in complex SQL statements. If you see transaction rollback errors in the `trace.log` file, increase the values of the heaps in increments of 256 until the errors stop.

Chapter 15. Identifying performance bottlenecks

Multiple middleware dependencies can complicate finding performance problems with IBM Tivoli Identity Manager. Identifying the performance bottleneck requires a step-wise approach.

The following guidelines can help you identify performance problems.

- Monitor the processor and disk usage of every server to see which server is most heavily used. The servers include IBM Tivoli Identity Manager nodes, directory, and database. Based on this information, review the monitoring and tuning steps specific to that component.
- Either the database or the directory server might be a bottleneck during heavy usage or large provisioning changes.

IBM Tivoli Identity Manager makes intense usage of its database and directory server. The database is an information staging area and audit trail for provisioning actions. The directory server is a permanent storage location that can be heavily queried when evaluating provisioning policies.

- An incorrectly tuned directory server can become the bottleneck as the IBM Tivoli Identity Manager server waits for the result set before starting the required provisioning action.

During an action that evaluates a large provisioning policy that affects many users, the affected users must be queried from the directory server. Examples of a large evaluation include adding a new policy or modifying an existing policy. The directory server evaluates the query and returns the matching users. Make sure that the directory server fulfills the requested queries as quickly and efficiently as possible to minimize this behavior.

- After the result set is returned, the IBM Tivoli Identity Manager server begins enforcing the provisioning policy for each user. This process is multithreaded and benefits from multiple processors. If it seems that the processors on the server are not fully used, check for a bottleneck on the LDAP or database server.
- Enforcement actions can cause access contention and locking in the database. The database stores any enforcement action required for a user (account addition, modification, or deletion) as a workflow item. When a thread becomes available, it queries the database for the next workflow item that requires processing and then acts on that item. Appropriate indexes and access plan statistics can minimize the number of required locks for filling these requests.

Related concepts

“Tuning Tivoli Directory Server” on page 63

When tuning IBM Tivoli Directory Server, it is important to understand the interaction between the IBM Tivoli Directory Server process and DB2.

“Tuning Sun ONE Directory Server” on page 76

With the Sun ONE Directory Server, you can improve performance by tuning the following areas: indexing, the All IDs Threshold value, and cache sizes.

Related tasks

“Tuning IBM DB2” on page 39

IBM Tivoli Identity Manager, version 5.0 and later, works with DB2 for Linux, UNIX, and Windows starting with Version 9. Version 9 has auto-tuning mechanisms that can reduce administrative and maintenance tasks.

“Tuning Oracle” on page 54

IBM Tivoli Identity Manager supports Oracle databases starting with version 10g on some operating systems.

“Tuning Microsoft SQL Server” on page 59

Tivoli Identity Manager supports certain versions Microsoft SQL Server databases.

Chapter 16. Monitoring system resources

Tuning an IBM Tivoli Identity Manager system requires monitoring system resources to determine environment bottlenecks.

Using IBM Tivoli Monitoring scripts

IBM released a monitoring solution for several IBM Tivoli software products.

About this task

The solution uses IBM Tivoli Monitoring on the Open Process Automation Library (OPAL). The scripts are user-extensible for specific customizing or additional monitoring.

- IBM Tivoli Identity Manager
- IBM Tivoli Directory Server
- IBM Tivoli Access Manager

Enabling DB2 monitoring

To gather performance information, turn on the DB2 monitoring flags.

About this task

Do not enable the table monitor. IBM Tivoli Identity Manager does not need it. It has a slight performance impact when enabled,

Procedure

1. As the database administrator, connect to the database and run the following commands for each database:

```
db2 update dbm cfg using DFT_MON_STMT ON
db2 update dbm cfg using DFT_MON_BUFPOOL ON
db2 update dbm cfg using DFT_MON_LOCK ON
db2 update dbm cfg using DFT_MON_SORT ON
db2 update dbm cfg using DFT_MON_TIMESTAMP ON
db2 update dbm cfg using DFT_MON_UOW ON
```
2. Stop and restart the database instance for the monitoring to take effect.

Collecting DB2 snapshots

Use snapshots to view the internal state of various IBM DB2 components.

Procedure

- To access specific IBM DB2 snapshots:

```
db2 get snapshot for database on database_name
db2 get snapshot for dynamic sql on database_name
db2 get snapshot for bufferpools on database_name
db2 get snapshot for tables on database_name
db2 get snapshot for locks on database_name
```
- To gather all snapshots:

```
db2 get snapshot for all on database_name
```

Configuring the DB2 statement monitor

Use the statement monitor to examine what is occurring for each request sent to the database.

About this task

The monitor collects a large amount of information. Activate it only for a short time to gather requests.

Procedure

1. Enter the following command to create the monitor dstatement writing to /tmp/dstatements:

```
db2 "create event monitor dstatement for statements  
write to file '/tmp/dstatements'"
```
2. If it does not exist, create the directory /tmp/dstatements.

```
mkdir /tmp/dstatements
```
3. The first time you generate an explain plan on this database, set up the explain tables with the following command.

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

What to do next

Monitor the database statements.

Using the DB2 statement monitor

After it has been enabled, the statement monitor collects detailed information about each request sent to the database.

Before you begin

Configure the statement monitor.

About this task

When you enable the statement monitor, it records each SQL request. You can examine the query results for missing indexes, execution time, preparation time, database scans, and index scans. Activate the monitor only for a short time to gather requests, because it collects a great deal of information.

Tip: The `otherTools/do_statement_monitoring.sh` script in the *Tuning Guide* scripts package automates this process. You can customize the script for your system. You can use the `explainSQL.sh` script to have DB2 explain how the optimizer processes a particular query, including any index usage.

Procedure

1. Connect to the database, clear out any previous data, and turn on the monitor by entering the following commands.

```
db2 connect to ldapdb2  
rm -f /tmp/dstatements/*  
db2 "set event monitor dstatement state 1"
```
2. Run the query or the action that you want to monitor.
3. Turn off the monitor with the following command.

```
db2 "set event monitor dstatement state 0"
```

4. Convert the data so that you can read it.

```
db2evmon -path /tmp/dstatements > /tmp/dstate.out
```

Related tasks

“Configuring the DB2 statement monitor” on page 100

Use the statement monitor to examine what is occurring for each request sent to the database.

Related information



ITIM performance tuning scripts

Download performance tuning scripts for IBM Tivoli Identity Manager.

Calculating the buffer pool hit ratio

The buffer pool hit ratio gives a good indication of how many data reads come from the buffer pool and how many from the disk. The larger the hit ratio, the less disk I/O used. Calculate the buffer pool hit ratio by enabling buffer pool monitoring and taking a database snapshot.

About this task

Use the following formula to calculate the buffer pool hit ratio:

P = buffer pool data physical reads + buffer pool index physical reads

L = buffer pool data logical reads + buffer pool index logical reads

Hit ratio = $(1 - (P/L)) * 100\%$

Related tasks

“Collecting DB2 snapshots” on page 99

Use snapshots to view the internal state of various IBM DB2 components.

Notices

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

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

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

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law :

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

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

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

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

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
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to

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

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information in softcopy form, the photographs and color illustrations might not be displayed.

Trademarks

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

Adobe, Acrobat, PostScript and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

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

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

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

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

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

Cell Broadband Engine and Cell/B.E. are trademarks of Sony Computer Entertainment, Inc., in the United States, other countries, or both and is used under license therefrom.



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

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

Index

A

- access control information
 - synchronization 20
- accessibility vii
- ACI cache
 - improving performance 28
 - reducing memory requirements 28
- ACL cache
 - directory server 64
- Active Directory 31
 - attributes 31
 - threads 32
- adapter
 - RACF 33
- adapters
 - Active Directory 31, 32
 - LDAP 33
 - Tivoli Identity Manager 31
- AIX
 - EXTSHM 53
 - parameters 53
 - performance 81
- allocating memory 5
- allocating resources
 - disk space 5
 - memory 5
 - processor usage 5
- allocating threads
 - Active Directory 32
- app_ctl_heap_sz
 - DB2 application heaps 49
- applheapsz
 - DB2 application heaps 49
 - DB2 connections 42
- application heaps
 - DB2 49
- assembly line caching
 - directory integrator 38
 - RMI dispatcher 38
- assembly line configuring
 - directory integrator 37
 - RMI dispatcher 37
 - running concurrently 37
- attribute
 - reconciliation 25
- attribute cache
 - directory server 64
- attributes
 - paging 65
 - returned during reconciliation 24
- automatic collection
 - statistics 50

B

- batch size
 - reports 21
- best practices
 - hardware 83
 - software 83

- books
 - See publications
- bottlenecks
 - identifying 97
- buffer pool
 - hit ratio 101
- buffer pools
 - DB2 43
 - DB2 caching 47
 - directory server 66
 - ENROLEBP 43
 - IBMDEFAULTBP 43, 66
 - LDAPBP 66

C

- cache
 - ACI 28
 - file system 47
- cache size
 - HTTP server 16
 - Sun ONE 79
- cache sizes
 - directory server 64
- cache timeout
 - HTTP server 17
- caching
 - assembly line 38
 - edge side include 16
- collecting statistics
 - DB2 50
- compressing database tables 48
- compression
 - HTTP server 14
 - tables 68
- configuration
 - table space 44
- configuring
 - e-mail notifications 21
 - list controls 19
 - paged search 27
- configuring attributes
 - Active Directory 31
- configuring connections
 - Oracle 54
- configuring indexes
 - SQL server 60
- configuring logging levels
 - directory integrator 35
- connection pooling
 - LDAP 19
- connections
 - DB2 42
 - HTTP server 13
 - JDBC 11
 - Oracle database server 54
- connector
 - Directory Integrator 36
 - DSML 36
- containers
 - table space 44

- controlling database size 29
- conventions
 - typeface viii
- CPU usage 6
- customized monitoring
 - scripts 99
- CVS report 21

D

- database
 - application heaps 49
 - buffer pools 43
 - compression 41
 - connections 42
 - DB2 tuning 39, 42, 43, 44, 50
 - directory server tuning 66
 - requirements 39
 - rollback error 49
 - row level compression 41
 - statistics 50
 - table compression 48
- database connections
 - Oracle 54
- database settings
 - lock list 53
 - lock timeout 53
 - maximum locks 53
- database size
 - controlling 29
 - DBPurge 29
 - SecurityIdentity Manager 29
- database tuning
 - indexes 58
 - init.ora configuration 54
 - open cursor 55
 - Oracle 54, 55
 - Oracle multiple disks 56
 - SQL 60
 - statement monitor 100
 - statistics 59
 - table space 56
- databases 39
- DB2
 - application heaps 49
 - buffer pools 43
 - compression 41
 - configuring table space 44
 - connections 42
 - maximum open files 52
 - memory manager 40
 - monitoring 99
 - multiple drivers 44
 - prefetch size 46
 - registry variables 54
 - snapshots 99
 - statement monitor 100
 - statistics 51
 - statistics collection 50
 - table space overhead 46
 - transaction logs 48

- DB2 (*continued*)
 - transfer rate 46
 - tuning 39, 44
 - variables 42, 43, 44, 45, 47, 49, 50, 52
- DB2 application heaps
 - app_ctl_heap_szS 49
 - applheapsz 49
- DB2 connections
 - applheapsz 42
 - MAXAPPLS 42
- DB2 indexes
 - directory server 72
- DB2 optimizer
 - hash joins 76
- DB2_PARALLEL_IO 54, 76
- DBPurge
 - variables 29
- directory integrator
 - assembly line caching 38
 - concurrent assembly line caching 37
 - logging levels 35
 - RMI dispatcher 36, 37, 38
 - timeouts 36
 - tuning 36
- directory integrator tuning
 - assembly line caching 38
 - concurrent assembly lines 37
 - removing assembly lines 36
- directory names, notation x
- directory server
 - buffer pools 66, 67
 - cache sizes 64
 - compressing tables 68
 - DB2 indexes 72
 - DB2 maximum open files 75
 - DB2 registry variables 76
 - DB2 variables 75
 - file system caching 67
 - hash joins 76
 - outages 91
 - paging 65
 - searches 65
 - slow query performance 89, 91
 - statement heap 70
 - statistics 74
 - system limits 70
 - table space 67
 - transaction logs 69
- directory server cache sizes
 - ACL 64
 - attribute 64
 - entry 64
 - filter 64
- directory server indexes
 - DB2 72
- directory server tuning
 - automatic statistic collection 73
 - cache size 79
 - index attributes 78
 - indexes 71, 77
 - referential integrity plug-in 80
 - Sun ONE 76
 - threshold value 77
 - Tivoli Directory sever 63
- directory servers 63
- disabling the recycle bin 22
- disapbe
 - EXTSHM2 53
 - file system caching 47
- disk
 - parameters 54
 - parameters for directory server 76
- disk performance
 - directory server 76
 - input 54, 76
 - output 54, 76
- disk space
 - storage allocation 6
- disk subsystems
 - RAID 54, 76
 - SAN 54, 76
- DMS
 - table space 45
- DSML
 - Tivoli Directory Integrator 36
- E**
 - e-mail notifications
 - configuring 21
 - edge side caching 16
 - edge side include
 - cache size 16
 - cache timeout 17
 - HTTP 16, 17
 - education
 - See* Tivoli technical training
 - emptying the recycle bin 23
 - enable
 - automatic resizing 45
 - enforcing
 - policy attributes 24
 - ENROLEBP
 - DB2 buffer pools 43
 - entitlement
 - parameters 25
 - entry cache
 - directory server 64, 72
 - environment variables, notation x
 - errors
 - rollback 94
 - ESI
 - HTTP 16, 17
 - EXTSHM
 - AIX 53
- F**
 - file system
 - caching 47
 - file system caching
 - directory server 67
 - disabling 67
 - filter cache
 - directory server 64
- H**
 - hardware
 - best practices 83
 - hash joins
 - DB2 optimizer 76
 - hash joins (*continued*)
 - directory server 76
 - disabling 76
 - heap size
 - JVM 9
 - WebSphere 94
 - hit ratio
 - buffer pool 101
 - HTTP server
 - cache size 16
 - cache timeout 17
 - compression 14
 - connections 13
 - static content 15
 - tuning 13
- I**
 - I/O
 - performance 54, 76
 - IBMDEFAULTBP
 - DB2 buffer pools 43
 - directory server buffer pools 66
 - index attributes
 - Sun ONE 78
 - index configuration
 - SQL server 60
 - indexes
 - directory server attributes 71
 - Oracle database server 58
 - Sun ONE 77, 78
 - init.ora
 - Oracle database server 54
 - initial tuning 3
- J**
 - Java
 - out of memory 21, 27
 - out of memory errors 94
 - Java 2 Security
 - system performance 12
 - JAVA Naming and Directory Interface 13
 - Java virtual machine
 - size 9
 - JDBC 55
 - JDBC connections
 - JDBC 11
 - JDBC driver
 - type 2 53
 - type 4 53
 - JNDI 13, 14
 - JVM
 - ACI cache 28
 - size 9
- L**
 - LDAP
 - connection pooling 19
 - paged search 27
 - paging control 33
 - reconciling 33
 - recycle bin 22

- LDAP search
 - e-mail 21
- LDAPBP
 - directory server buffer pools 66
- ldapClean 23
- list control
 - configuring 19
 - parameters 19
- load balancing
 - HTTP 13
- lock list 53
- lock timeout 53
- logs
 - transaction 48, 69

M

- machine size
 - JVM 9
- maintenance schedules
 - Tivoli Identity Manager 87
- manuals
 - See* publications
- max duration
 - reconciliation 24
- MAXAPPLS
 - DB2 connections 42
- maximum duration
 - reconciliation 26
- maximum locks 53
- memory allocation 5
- memory manager
 - buffer pools 40
 - DB2 40
 - package heap 40
 - self-tuning 40
 - sort heap 40
- memory space 5
- Microsoft SQL server
 - index configuration 60
 - tuning 60
- mod_deflate plug-in
 - HTTP 14
- monitoring
 - DB2 99
 - scripts 99
 - snapshots 99
 - system resources 99
- multiple disks
 - Oracle database server 56
- multiple drivers
 - DB2 44

N

- notation, environment variables
 - path names x
 - typeface x

O

- online publications
 - accessing vi
- open cursor
 - Oracle database server 55

- open files
 - maximum 52, 75
- open process automation library 99
- operating systems 81
- Oracle database server
 - connections 54
 - indexes 58
 - init.ora file 54
 - multiple disks 56
 - open cursor 55
 - statistics 59
 - table space 56, 57
 - tuning 54
 - XA recovery 55
- ordering publications vi
- out of memory
 - Java 94
 - Java error 21, 27
- outages
 - directory server 91

P

- paged search
 - configuring 27
- paging
 - configuring attributes 65
 - directory server 65
 - LDAP 33
 - searches 65
- parameters
 - AIX 53
 - list control 19
- path names, notation x
- PDU_ENTRY_LIMIT
 - RACF adapter 33
- performance 81
 - AIX 81
 - bottlenecks 97
 - high-yield improvements 1
 - maintenance schedule 87
 - reconciliation 24
 - Solaris 81
- performance monitoring infrastructure
 - WebSphere 10
- plug-ins
 - mod_deflate 14
- PMI
 - WebSphere 10
- policy
 - governing 93
- policy enforcement
 - optimizing 25
 - reconciliation 25
 - reducing 24
- pooling
 - connection 19
 - LDAP 19
- prefetch size
 - DB2 46
- problem solving 89
- processor usage
 - allocating 6
- provisioning
 - Active Directory 32
- publications iii
 - accessing online vi

- publications (*continued*)
 - ordering vi

R

- RACF
 - adapter 33
 - PDU_ENTRY_LIMIT 33
- reconciliation
 - Active Directory 31
 - duration 26
 - limiting attributes 25
 - limiting attributes returned 24
 - optimizing policy enforcement 25
 - policy enforcement reduction 24
 - threads 26
- reconciliation performance
 - max duration 24
- reconciling
 - LDAP 33
- recycle bin
 - disabling 22
 - emptying 23
 - LDAP 22
- referential integrity plug-in
 - Sun ONE 80
- REORGCHK 74
- report
 - batch sizes 21
 - data synchronization 20
- resizing
 - table space 45
- resource allocation 5
- resources 12
 - system 99
- RMI dispatcher
 - tuning 36
- rollback errors 94
- RUNSTATS 74

S

- schedules
 - maintenance 87
- scripts
 - monitoring 99
- search errors
 - governing policy 93
- self-tuning memory manager
 - enabling 40
- server connections
 - HTTP 13
- server-side sorting
 - enable 28
- slow query performance
 - Sun ONE 89
 - Tivoli Directory Server 91
- snapshots
 - DB2 99
 - monitoring 99
- software
 - best practices 83
- Solaris
 - performance 81
- sorting
 - server-side 28

- SQL server
 - index configuration 60
 - tuning 60
- statement heap
 - directory server 70
 - LDAP 70
- statement monitor
 - DB2 100
 - using 100
- static content
 - caching 15
 - HTTP server 15
- statistics
 - automatic collection 73
- statistics
 - DB2 51
 - directory server 74
 - Oracle database server 59
 - table 51
- statistics collection
 - database 50
- storage
 - disk space 6
- storage optimization
 - DB2 41
- Sun ONE
 - cache size 79
 - index attributes 78
 - indexes 77
 - referential integrity plug-in 80
 - slow performance 89
 - threshold value 77
 - tuning 76
- Sun ONE directory server 63
- supported databases 39
- synchronization
 - ACI 20
 - report data 20
- system limits
 - configuring 70
 - directory server 70
- system performance
 - Java 2 Security 12
- system resources
 - monitoring 99

T

- table compression
 - database 48
 - directory server database 68
- table monitor 99
- table rows 51
- table space
 - alter tablespace command 45
 - autoresize 45
 - containers 44
 - ENROLE_DATA 45, 47
 - ENROLE_INDEXES 45, 47
 - maxfilop 52, 75
 - Oracle database server 56, 57
 - resizing 45
 - SMS 75
 - TEMP_DATA 45, 47
- table space configuration
 - DB2 44

- table space data files
 - Oracle 57
- table space names
 - LDAPSPACE 67
 - USERSPACE1 67
- table space overhead
 - DB2 46
- threads
 - Active Directory 32
 - reconciliation 26
- threshold value
 - Sun ONE 77
- timeouts
 - directory integrator 36
 - reconciliations 36
 - RMI dispatcher 36
- Tivoli Directory Integrator
 - DSML connector 36
 - tuning 35
- Tivoli Directory server 63
 - indexes 71
 - statistics 73
 - tuning 63, 71, 73
- Tivoli Directory Server
 - outages 91
 - slow performance 91
- Tivoli Documentation Central vi
- Tivoli Identity Manager
 - adapters 31
- Tivoli technical training vii
- Tivoli user groups vii
- training, Tivoli technical vii
- transaction errors 94
- transaction logs
 - DB2 48
 - directory server 69
- transfer rate
 - DB2 46
- troubleshooting 89
 - directory server 89, 91
 - out of memory 94
 - policy search errors 93
 - transaction rollback 94
- tuning
 - ACI cache 28
 - DB2 indexes for directory server 72
 - directory integrator 36
 - directory integrator logging levels 35
 - directory server 63, 65, 70, 71, 76
 - HTTP server 13
 - initial 3
 - Oracle database server 54
 - policy enforcement 25
 - reconciliation 24, 25, 26
 - RMI dispatcher 36
 - SQL server 60
 - Tivoli Directory Integrator 35
 - Tivoli Directory Server 63
 - Tivoli Directory Server indexes 71
 - Tivoli Directory Server statistics 73
 - Tivoli Identity Manager 19
 - ulimits 70
 - WebSphere Application Server 9
- tunint
 - Sun ONE 76
- typeface conventions viii

U

- ulimits
 - directory server 70
- user groups, Tivoli vii

V

- variables, notation for x

W

- WebSphere
 - Java 2 Security 12
 - JDBC connections 11
 - performance monitoring
 - infrastructure 10
 - PMI 10
- WebSphere Application Server
 - tuning 9

X

- XA recovery operations
 - enabling 55
 - Oracle database server 55



Printed in USA

SC23-6594-04

