OPTIMIZE

MERCURY BUSINESS AVAILABILITY CENTER™

Preparing the Database Environment

MERCURY™

BUSINESS TECHNOLOGY OPTIMIZATION

Mercury Business Availability Center

Preparing the Database Environment

Version 6.1

Document Release Date: March 5, 2006

MERCURY™

Mercury Business Availability Center, Version 6.1
Preparing the Database Environment

This manual, and the accompanying software and other documentation, is protected by U.S. and international copyright laws, and may be used only in accordance with the accompanying license agreement. Features of the software, and of other products and services of Mercury Interactive Corporation, may be covered by one or more of the following patents: United States: 5,511,185; 5,657,438; 5,701,139; 5,870,559; 5,958,008; 5,974,572; 6,137,782; 6,138,157; 6,144,962; 6,205,122; 6,237,006; 6,341,310; 6,360,332, 6,449,739; 6,470,383; 6,477,483; 6,549,944; 6,560,564; 6,564,342; 6,587,969; 6,631,408; 6,631,411; 6,633,912; 6,694,288; 6,738,813; 6,738,933; 6,754,701; 6,792,460 and 6,810,494. Australia: 763468 and 762554. Other patents pending. All rights reserved.

Mercury, Mercury Interactive, the Mercury logo, the Mercury Interactive logo, LoadRunner, WinRunner, SiteScope and TestDirector are trademarks of Mercury Interactive Corporation and may be registered in certain jurisdictions. The absence of a trademark from this list does not constitute a waiver of Mercury's intellectual property rights concerning that trademark.

All other company, brand and product names may be trademarks or registered trademarks of their respective holders. Mercury disclaims any responsibility for specifying which marks are owned by which companies or which organizations.

Mercury provides links to external third-party Web sites to help you find supplemental information. Site content and availability may change without notice. Mercury makes no representations or warranties whatsoever as to site content or availability.

Mercury Interactive Corporation
379 North Whisman Road
Mountain View, CA 94043
Tel: (650) 603-5200
Toll Free: (800) TEST-911
Customer Support: (877) TEST-HLP
Fax: (650) 603-5300

© 2005-2006 Mercury Interactive Corporation, All rights reserved

If you have any comments or suggestions regarding this document, please send them by e-mail to documentation@mercury.com.

AMLIB_DB6.1/01

Table of Contents

Welcome to Preparing the Database Environment vii
How This Guide Is Organized vii
Who Should Read This Guide viii
Getting More Information viii

PART I: INTRODUCING THE DATABASE ENVIRONMENT

Chapter 1: Introduction to Preparing the Database Environment 3
Databases Used in Mercury Business Availability Center 3

PART II: DEPLOYING AND MAINTAINING THE MS SQL SERVER DATABASE

Chapter 2: Overview of MS SQL Server Deployment 9
About MS SQL Server Deployment 10
System Requirements 11

Chapter 3: Creating and Configuring MS SQL Server Databases 13
Creating Databases 13
Configuring Databases 22

Chapter 4: Using Windows Authentication to Access MS SQL Databases 29
Enabling Mercury Business Availability Center to Work with Windows Authentication 29
Connecting to MS SQL Databases Using Windows Authentication 32

Chapter 5: Maintaining MS SQL Server Databases 35
Backing Up Databases 35
Database Integrity and Fragmentation 39
Monitoring Databases 50
Database Maintenance – References 55

Chapter 6: MS SQL Server Sizing Guidelines....57
Mercury Business Availability Center Sizing....58
Hardware Sizing....59
Server Configuration Options....60
Data File Property Settings....60
tempdb Database Settings....61

Chapter 7: MS SQL Server Summary Checklists....63
Checklist for Mercury Business Availability Center Support and Certification....63
Verifying and Modifying Server and Database Settings....65

Chapter 8: Installing and Configuring MS SQL Server....69
Installing MS SQL Server....69
Configuring MS SQL Server....78

PART III: DEPLOYING AND MAINTAINING THE ORACLE SERVER DATABASE

Chapter 9: Overview of Oracle Server Deployment....85
About Oracle Server Deployment....85
System Requirements....87
Choosing an Edition of Oracle Server....88

Chapter 10: Setting Up the Mercury Business Availability Center Database Environment91
Mercury Business Availability Center Oracle Tablespaces....92
Management User Schema....95
Profile User Schemas....98
Mercury Business Availability Center Oracle Schema Privileges....100

Chapter 11: Configuring the Oracle Client for Mercury Business Availability Center....103
Oracle Client Versions and Operating System Platforms....104
Oracle Client Installation....104
Oracle Client Configuration....105

Chapter 12: Maintaining an Oracle Server Database....107
Database Maintenance and Tuning....107
Oracle Database Backup and Recovery....120

Chapter 13: Oracle Server Sizing Guidelines....125
Mercury Business Availability Center Sizing....125
Hardware Sizing....127
Oracle Parameter Sizing....127
Oracle File Sizing....132

Chapter 14: Oracle Summary Checklist 137
Checklist for Mercury Business Availability Center Support and Certification 138

PART IV: APPENDIXES

Appendix A: Database Consolidation and Distribution 149
Advantages of Database Consolidation 149
Reasons for Database Distribution 150

Appendix B: Data Storage Recommendations 151
Creating MS SQL Server File Groups 151
Creating Oracle Server Tablespaces 156

Appendix C: Data Partitioning and Purging 159
About Data Partitioning and Purging 159
How the Purging Manager Works 160
Calculating Purging Manager Parameter Values 161
Setting the Purging Manager Parameters 162
Creating Oracle Tablespaces When Using the Purging Manager 165

Appendix D: Database Schema Verification 167
About the Verification Process 167
Running the Verification Process 168
Creating Database Users for the Verify Procedure 173

Appendix E: Managing Data with Loaders 175
Overview of Managing Data with Loaders 176
Troubleshooting Loader Problems 177
Loader Folder Locations 177
Changing the Default Logging Level 179
Changing the Time Frame for Retaining Failed Data 180
The DLQ Cleanup Tool 181

Appendix F: Backing Up and Restoring Monitor Administration Configuration Data 183
Overview of Backing Up and Restoring Configuration Data 184
Basic Backup and Restore 185
Basic Backup on Replicated Server and Restore 186
LDIF Backup and Restore 186
Replication 189
Starting and Stopping OpenLDAP 192
Backup and Restore Examples 194

Index 199

Welcome to Preparing the Database Environment

This guide describes how to deploy and maintain MS SQL Server and Oracle Server databases for use with Mercury Business Availability Center.

Note: This guide is not relevant to Mercury Managed Services customers.

How This Guide Is Organized

The guide contains the following chapters:

Part I Introducing the Database Environment

Describes the types of databases used with Mercury Business Availability Center.

Part II Deploying and Maintaining the MS SQL Server Database

Describes how to install and configure MS SQL Server, how to create and configure databases on MS SQL Server, and how to maintain databases.

Part III Deploying and Maintaining the Oracle Server Database

Describes how to install Oracle Server, how to create and configure databases on Oracle Server, how to set up the Mercury Business Availability Center database environment for Oracle, how to configure the Oracle Client, and how to maintain databases.

Part IV Appendixes

Describes the situations in which you should consolidate all Mercury Business Availability Center data in a single database and the situations in which you should distribute the data among several different databases. Also contains recommendations for creating MS SQL Server file groups and Oracle tablespaces, information on working with the Purging Manager, the procedure for verifying your database schema, information about Mercury Business Availability Center loaders, and the procedures for backing up and restoring LDAP databases for Monitor Administration configuration data.

Who Should Read This Guide

This guide is intended for the following users of Mercury Business Availability Center:

- Mercury Business Availability Center administrators
- Database administrators

Readers of this guide should be knowledgeable and highly skilled in database administration.

Getting More Information

For information on using and updating the Mercury Business Availability Center Documentation Library, reference information on additional documentation resources, typographical conventions used in the Documentation Library, and quick reference information on deploying, administering, and using Mercury Business Availability Center, refer to *Getting Started with Mercury Business Availability Center*.

Part I

Introducing the Database Environment

1

Introduction to Preparing the Database Environment

This chapter contains information on the types of databases used with Mercury Business Availability Center.

This chapter describes:	On page:
Databases Used in Mercury Business Availability Center	3

Databases Used in Mercury Business Availability Center

To work with Mercury Business Availability Center, you must set up the following types of databases:

- **Management database –** for storage of system-wide and management-related metadata for the Mercury Business Availability Center environment. Mercury Business Availability Center requires one management database. You can create this database manually, or by using the set management database utility.
- **Profile database(s) –** for storage of raw and aggregated measurement data obtained from the Mercury Business Availability Center data collectors. Although only one profile database is required, you can store profile data in multiple databases, if required. You can create profile databases manually, or by using the Database Management page, accessible from the Setup and Maintenance tab in Platform Administration.

- **Mercury Universal CMDB** – for storage of configuration information that is gathered from the various Mercury Business Availability Center and third-party applications and tools. This information is used when building Mercury Business Availability Center views. The CMDB also contains the object repositories used to define configuration items and Key Performance Indicator (KPI)s. The CMDB can be shared with Mercury Application Management (for details, see "Sharing the Mercury Universal CMDB Environment" in *Working with the CMDB*).

Note: By default, the CMDB is created as part of the management database, but you can create a CMDB manually, or by using the CMDB Database Management page, accessible from the Setup and Maintenance tab in Platform Administration. Throughout this guide, the CMDB will be considered as part of the management database, unless specifically mentioned otherwise.

You can set up management and profile databases, and the CMDB, on either a Microsoft SQL Server or an Oracle Server, depending on the type of database server used in your organization. If you are working with an MS SQL Server database, refer to Part I, "Deploying and Maintaining the MS SQL Server Database." If you are working with an Oracle Server database, refer to Part II, "Deploying and Maintaining the Oracle Server Database." The appendixes contain additional information that is pertinent to both MS SQL Server and Oracle Server databases.

Note:

- Database servers must be set to the same time zone, daylight savings settings and time as the Mercury Business Availability Center servers.
- To work in a non-English language Mercury Business Availability Center environment, you must use an Oracle Server database. MS SQL Server and MSDE databases do not support non-English language Mercury Business Availability Center environments. For information pertaining to non-English language environments, see *Working in an I18N Environment*.

Part II

Deploying and Maintaining the MS SQL Server Database

2

Overview of MS SQL Server Deployment

You can set up management and profile databases on an MS SQL Server. This chapter describes the following topics related to deploying MS SQL Servers for use with Mercury Business Availability Center:

This chapter describes:	On page:
About MS SQL Server Deployment	10
System Requirements	11

About MS SQL Server Deployment

To deploy MS SQL Server for use with Mercury Business Availability Center, you must perform the following procedures:

➤ **Install and configure MS SQL Server**

For details, see "Installing and Configuring MS SQL Server" on page 69.

➤ **Create databases on MS SQL Server to store management and profile data**

For details, see "Creating Databases" on page 13.

Note: You can allow Mercury Business Availability Center to create databases for you, or you can create databases yourself using the CREATE DATABASE statement or MS SQL Server Enterprise Manager. It is recommended that you create databases yourself.

Information is provided in this section for both recommended and supported MS SQL Server environments. Mercury Business Availability Center recommendation indicates that Mercury quality assurance personnel have rigorously tested the recommended environment/option. A supported environment or option means that Mercury quality assurance personnel have successfully performed basic tests on the environment/option.

System Requirements

The following table describes the system requirements for working with MS SQL Server in conjunction with Mercury Business Availability Center:

Component	Supported		Recommended	
	Version/Edition	**Service Pack**	**Version/Edition**	**Service Pack**
Operating System	Windows 2000 Server / Advanced Server	Service Pack 4	Windows 2003 Server – standard / enterprise	Service Pack 1
MS SQL Server	MS SQL Server 2000 Enterprise	Service Pack 4	MS SQL Server 2000 Enterprise	Service Pack 4
Microsoft Data Access Components (MDAC) on Mercury Business Availability Center servers	2.5; 2.52; 2.61; 2.62; 2.7 SP1 Refresh		2.7 SP1 Refresh – Installed automatically with your Mercury Business Availability Center server installation.	

Note: To check which version of MDAC is installed on your machine, download and run the component checker tool from: http://www.microsoft.com/downloads/details.aspx?FamilyID=8f0a8df6-4a21-4b43-bf53-14332ef092c9&displaylang=en

---

3

Creating and Configuring MS SQL Server Databases

This chapter describes the creation and configuration of Mercury Business Availability Center databases on an MS SQL Server.

This chapter describes:	On page:
Creating Databases	13
Configuring Databases	22

Creating Databases

Once you have installed and configured MS SQL Server, you create a management database for the storage of system-wide and management-related data, and one or more profile databases for the storage of data collected by Mercury Business Availability Center.

You can create databases yourself using the CREATE DATABASE statement or MS SQL Server Enterprise Manager (for more information, see MS SQL Server 2000 Books Online), and the instructions in the following sections:

- Database Permissions – see page 16.
- Database File Layout – see page 17.
- System Databases – see page 21.

Note: This is the preferred method of creating Mercury Business Availability Center databases.

Once you have generated the management and profile schemas (see below), you can connect Mercury Business Availability Center to the existing management database from the set management database utility, and to the existing profile database from the Database Management page in Platform Administration's Setup and Maintenance tab. If you generate another CMDB schema, instead of the default CMDB included in the management database, you can connect to the new CMDB from the CDMB Database Management page in Platform Administration's Setup and Maintenance tab.

Alternatively, you can allow Mercury Business Availability Center to create a management database for you using the set management database utility, and to create profile databases for you using the Database Management page in Platform Administration's Setup and Maintenance tab. If you want to use a different CMDB, instead of the default CMDB included in the management database, you can allow Mercury Business Availability Center to create a CMDB for you using the CMDB Database Management page in Platform Administration's Setup and Maintenance tab.

For information on the set management database utility, see "Setting Management Database Parameters for a Windows Platform" in *Deploying Servers*. For information on creating profile databases in Mercury Business Availability Center, see "Database Management" in *Platform Administration*. For information on creating a CMDB in Mercury Business Availability Center, see "Mercury Universal CMDB Management" in *Platform Administration*.

Note: If you allow Mercury Business Availability Center to create databases for you, it is strongly recommended that you then configure the databases. For more information, see "Configuring Databases" on page 22.

Manually Running Management and Profile Schema Creation Scripts

In addition to creating the database itself manually, according to the recommendations in this section, it is recommended that you manually run the scripts that generate the management and profile schemas (table structures), instead of allowing Mercury Business Availability Center to run them automatically.

Note: If you create a CMDB manually, according the recommendations in this section, you must run the scripts that generate the CMDB user schema as you are unable to connect to an existing CMDB that does not contain the default table structures. For details on creating and connection to CMDB in Mercury Business Availability Center, see "Mercury Universal CMDB Management" in *Platform Administration*.

Note: Mercury Business Availability Center automatically creates two file groups to store management database objects and two file groups to store profile database objects. For more information on creating additional file groups, see Appendix B, "Data Storage Recommendations."

The management and profile schema creation scripts are located in the **<Core Server root directory>\AppServer\webapps\site.war\DataBases\SQL_Svr_DB_Utils** directory.

To create the management schema (including the default CMDB):

Run the following scripts:

- common_sql_dbobjects_create.sql
- management_sql_dbobjects_create.sql
- create_cm_tables_cmdb_ms.sql

To create a CMDB schema:

Run the following scripts:

- common_sql_dbobjects_create.sql
- create_cm_tables_cmdb_ms.sql

To create a profile schema:

Run the following script:

- profile_sql_dbobjects_create.sql

If you revise these scripts—for example, in order to specify user-defined file groups for data/index/large objects—ensure that you make a copy of the scripts, modify the copy, and run it manually using the Query Analyzer or OSQL.

Note: Perform the above procedures only if you are an experienced MS SQL Server database administrator.

It is strongly recommended that you run the database schema verify program after creating the management and/or profile schemas to verify that your database is configured properly. For information on the verification process, see Appendix D, "Database Schema Verification."

Database Permissions

To create a database, you must have CREATE DATABASE permissions. To connect to an existing database, the login account with which you are connecting must be mapped to dbo in the database.

Note: Members of the sysadmin server role automatically have CREATE DATABASE permissions, and are also mapped to dbo in all databases. A database owner is automatically mapped to dbo in the database.

To grant CREATE DATABASE permissions to a user, the user's login must first be mapped to a database user in the master database.

Use one of the following three methods to map a user's login to dbo in the required database:

- Make the user a member of the sysadmin server role (remember that this gives the login the strongest possible permissions for all server activities).
- Make the user a database owner, using EXEC sp_changedbowner 'login'.
- Alias the user login as the dbo in the database, using EXEC sp_addalias 'login', 'dbo'.

To check whether a user is a database owner, run the following:

```
EXEC sp_helpdb <database name>
```

To check whether a user has CREATE DATABASE permissions, log in to the Query Analyzer with the login account of the user whose permissions you want to check, and run the following:

```
USE master
IF PERMISSIONS() & 1 = 1
  PRINT 'User has CREATE DATABASE permissions'
ELSE
  PRINT 'User does not have CREATE DATABASE permissions'
```

To check whether a user is mapped to dbo in the database, log in to the Query Analyzer with the login account of the user whose mapping you want to check. Change the database context to the required database, and run the following:

```
SELECT USER_NAME()
```

Database File Layout

When you create a database, it must consist of at least one data file (with an .mdf extension) and one transaction log file (with an .ldf extension). You can optionally create additional data files (.ndf), as well as additional log files (.ldf).

To enhance performance, you may want to create several data files. MS SQL Server stripes the data among the data files, so that if you do not have RAID controllers that stripe your data, you can spread the data files over several regular physical disks and, in this way, have the data striped. The log, however, is read sequentially, so that there is no performance gain in adding more log files. An additional log file should be created on a different disk when your existing log is out of disk space.

Data and Log Placement

It is recommended that you place the data and log files on separate disk subsystems. Changes are not flushed to the database until they are written to the log, and the log architecture dictates serial writes, so it is advisable that there be as little interference as possible with the log activity. It is usually sufficient to place the log on a RAID 1 system because of the serial writes to the log. If you have processes reading from the log (for example, triggers accessing the inserted and deleted views which are formed from the log records or transactional replication), or several log files for different databases, consider placing the log file(s) on a RAID 0+1 (striped mirror) system.

The data files should be placed on a RAID 0+1 system for optimal performance.

Note: It is recommended not to place data or log files on the same disk that stores the page (swap) file.

File and Database Properties

When you create a database you can specify the following five properties for each file (.mdf, .ndf, .ldf):

- **NAME –** the logical file name which you can use later when you want to alter one of the properties.
- **FILENAME –** the physical file path and name. Make sure the destination directory is not compressed (right-click the directory in Windows Explorer, select **Advanced**, and verify that the compression check box is not selected).

- **SIZE –** the initial file size.
- **MAXSIZE –** determines the maximum size to which the file can grow. If this argument is omitted, or if you specify **Unlimited**, the file can grow until the disk is full.
- **FILEGROWTH –** the automatic growth increment of the file. This argument can be specified as either a percentage of the existing file size, or as a fixed size. For more information, see Chapter 6, "MS SQL Server Sizing Guidelines."

Note: An autogrowth operation invoked by a modification sent by a client that timed out cannot be completed successfully. This means that the next time a client sends a modification, the autogrowth process starts at the beginning and may also time out. To avoid this problem, it is recommended that you either expand the files manually every time the database nearly reaches full capacity (for example, 20 percent free), or set the growth increment to a fixed size that takes less time to be allocated than the client's timeout setting. In general, most systems are able to allocate 100 MB in less than 30 seconds, which is the common client timeout setting.

For more information on this problem, refer to Microsoft Knowledge Base Article - 305635 (http://support.microsoft.com/default.aspx?scid=kb;en-us;Q305635).

File Groups

File groups are logical groupings of data files. Each of the following objects can be placed in its own file group unit:

- A table's data
- A table's large objects (text, ntext, image columns)
- An index

Data is inserted proportionally into all files belonging to the file group in which the object is stored, according to the amount of free space in each file. The **.mdf** file is placed in a file group called **PRIMARY**, which is marked as **Default** when the database is created (the default file group for objects when no file group is specified). If you do not place other data files (**.ndf** files) in their own file groups, they are also placed in the **PRIMARY** file group. Note that you can change the **Default** file group later on.

File groups can be used for performance tuning or maintenance. For details, refer to MS SQL Server 2000 Books Online, as well as the storage and performance tuning-related whitepapers at http://support.microsoft.com/common/canned.aspx?R=d&H=SQL%20Server%202000%20White%20Papers&LL=kbSQLServ2000Search&Sz=kbwhitepaper&CDID=EN-US-KB&LCID=1033.

Following is an example that demonstrates how to use file groups for maintenance:

- **Partial Restoring –** MS SQL Server 2000 does not support the restoration of a single table. Even if you place a single table in a file group, you cannot restore a file group to a point in time earlier than the rest of the data. Instead, you must apply all log file backups in order to synchronize the file group with the rest of the data. MS SQL Server 2000 supports partial restoration to a database with a different name. A partial restoration allows you to restore a single file group, and supports point-in-time restoration. However, you must restore the **PRIMARY** file group because it contains the SYSTEM tables.

  To be able to restore a single table to a point in time if a logical error occurs, you need to design the file groups in your database as follows:

  - Ensure that the **.mdf** file is the only file in the **PRIMARY** file group.
  - Place each large table in its own file group.
  - Place all small tables in a separate file group.

System Databases

The following system databases are especially important for the smooth performance of MS SQL Server:

- **tempdb –** Numerous MS SQL Server activities—such as creating local and global temporary tables, creating work tables behind the scenes to spool intermediate query execution results, and sorting—implicitly or explicitly use the tempdb system database. If your system is not configured properly, the tempdb database can become a performance bottleneck, so it is very important to determine the tempdb database's original size correctly. For more information on setting database sizes, see Chapter 6, "MS SQL Server Sizing Guidelines." To move tempdb's files, use the ALTER DATABASE tempdb MODIFY FILE command, and restart MS SQL Server.
- **master, msdb, model –** These databases, although crucial for the operation of MS SQL Server, are smaller than tempdb because they store only meta data. It is strongly recommended to use a fault tolerant disk—ideally, RAID 1—for these databases.

Note: For Mercury Business Availability Center certification, place system databases on fault tolerant disks.

To check the database's properties, run the following:

```
EXEC sp_helpdb <database name>
```

Configuring Databases

Once you have created the necessary databases, you can add new files to the databases, change some of the existing database file properties, and set the database configuration options appropriately.

Note: If you allowed Mercury Business Availability Center to create management and profile databases for you, it is strongly recommended that you configure them using the configuration instructions in this section.

Database File Configuration

You can change certain database file properties, as well as add or drop files, using the Properties dialog box in the Enterprise Manager or the ALTER DATABASE command (for details, refer to MS SQL Server 2000 Books Online).

Adding Files

Data files can be added to an existing file group in a database, or to a new file group. There are no special restrictions or requirements.

Dropping Files

To drop a file, you must first empty it using the DBCC SHRINKFILE command's EMPTYFILE option, which transmits the file data to all the other files in the file group. Once you empty the file, you can use the ALTER DATABASE <database name> DROP FILE command to drop it.

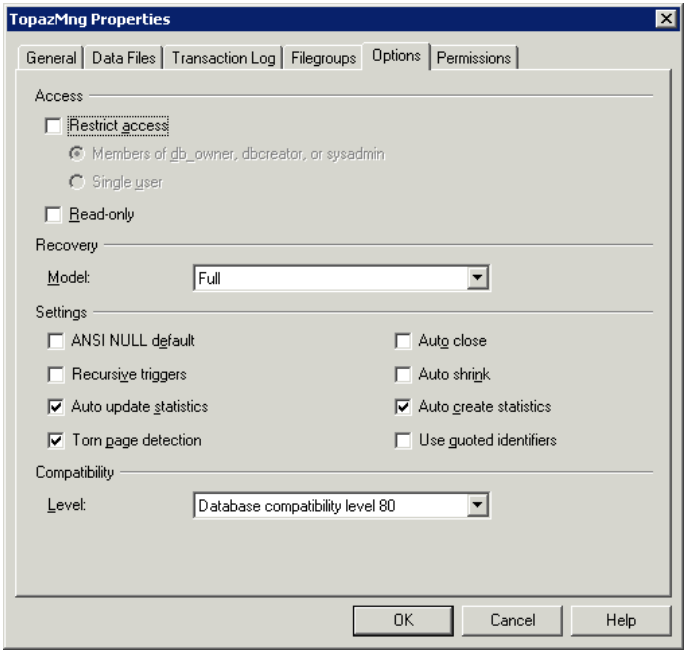
Changing File Properties

You can change the size-related properties for all databases, as well as the filename property for the tempdb database (this takes effect after you restart MS SQL Server). The SIZE, MAXSIZE, and FILEGROWTH properties can be changed using the ALTER DATABASE tempdb MODIFY FILE command. Note that the SIZE property can only be enlarged. To shrink the file, use the DBCC SHRINKFILE command. For details and recommendations concerning file properties, see "Creating Databases" on page 13.

Database Configuration Options

Each database contains a set of configurable options that determine its behavior. You can view or change the database options using:

- The Options tab in the Enterprise Manager's Properties dialog box



Note: Not all of the database configuration options are available in this dialog box.

- The sp_dboptions stored procedure
- The ALTER DATABASE <database name> SET command

The following table lists the default configuration options, as well as the configuration settings required for Mercury Business Availability Center certification:

Configuration Option	Description	Default	Mercury Business Availability Center Certification in MS SQL Server 2000
Restrict access	Only single users or members of the db_owner, dbcreator, or sysadmin groups can access the database.	Not set (MULTI_USER)	MULTI_USER
Read only	Database is read only	Not set (READ_WRITE)	READ_WRITE
Recovery	The database recovery model determines the recovery capabilities by controlling the amount of bulk operation logging (such as Select into, Bulk, Insert, Create index, LOB manipulation). The higher the recovery model, the higher the recovery capabilities. However, the amount of logging also increases, which may affect performance.	Full	Full (unless you are certain that the lower recovery capabilities are sufficient for your system)

Configuration Option	Description	Default	Mercury Business Availability Center Certification in MS SQL Server 2000
Truncate log on checkpoint	Automatically marks inactive portions of log for reuse on checkpoint	Not set	N/A
Select into/bulk copy	Allows the use of minimally logged Select into/bulk copy operations	Not set	N/A
ANSI NULL default (see note below)	Specifies whether the database columns are defined as NULL or NOT NULL, by default	Not set	Not set
Recursive triggers	Specifies whether recursive triggers are supported	Not set	Not set
Auto update statistics	Specifies whether out-of-date statistics required by a query for optimization are built automatically during optimization	Set	Set
Auto create statistics	Specifies whether missing statistics required by a query for optimization are built automatically during optimization	Set	Set
Torn page detection	Specifies whether incomplete pages can be detected	Set	Set

Configuration Option	Description	Default	Mercury Business Availability Center Certification in MS SQL Server 2000
Auto close	Specifies whether the database shuts down after its resources are freed and all users exit	Not set	Not set **Note:** If set, it may take a long time for the database to allocate resources every time a user connects, after the database is closed.
Auto shrink	Specifies whether the database is automatically shrunk every hour, leaving 25% of free space	Not set	Not set **Note:** If set, constant growth/ shrinkage may cause file system fragmentation.

Configuration Option	Description	Default	Mercury Business Availability Center Certification in MS SQL Server 2000
Use quoted identifiers	Specifies whether the MS SQL Server enforces ANSI rules regarding quotation marks. Select this option to specify that double quotation marks be used only for identifiers, such as column and table names. Note that character strings must be enclosed in single quotation marks.	Not set	Not set
Compatibility level	The version of MS SQL Server that the database appears to be (for the application)	80	80

Note: Not all ANSI options can be set using the Enterprise Manager. The ANSI database configuration options include: ANSI_NULLS, QUOTED_IDENTIFIER, ANSI_NULL_DEFAULT, ANSI_PADDING, ANSI_WARNINGS, ARITHABORT, NUMERIC_ROUNDABORT, CONCAT_NULL_YIELDS_NULL. Note that the options you set may not take effect, since these options can also be set at a higher level. For example, if the session option QUOTED_IDENTIFIER was turned on, the equivalent database configuration option is irrelevant. Some tools or database interfaces turn certain session options on or off, so that the database configuration options never take effect.

The following table summarizes the characteristics of each recovery model:

Model/ Support	Allows Log Backup	Allows Point-in-Time/Log Mark Restoration	Allows Backup Log when Data Crashes (Saves changes until the crash point)	Amount of Bulk Operation Logging (can affect the performance of bulk operations)
Simple	No	No	No	Minimal
Bulk Logged	Yes	No	No	Minimal
Full	Yes	Yes	Yes	Full

To check your database's properties, run the following:

```
EXEC sp_helpdb <database name>
```

4

Using Windows Authentication to Access MS SQL Databases

Unless configured otherwise, Mercury Business Availability Center uses MS SQL Server authentication to access MS SQL databases. This chapter describes how to enable Mercury Business Availability Center to use Windows authentication to access MS SQL databases.

This chapter describes:	On page:
Enabling Mercury Business Availability Center to Work with Windows Authentication	29
Connecting to MS SQL Databases Using Windows Authentication	32

Enabling Mercury Business Availability Center to Work with Windows Authentication

You can enable Mercury Business Availability Center to use Windows authentication instead of MS SQL Server authentication to access any of the Mercury Business Availability Center databases (management, profile, and CMDB).

To enable Mercury Business Availability Center to use Windows authentication to access an MS SQL database, you must launch the Mercury Business Availability Center service on all the Mercury Business Availability Center servers with a user that has the necessary permissions to access the specific database. For details on connecting to an MS SQL database using Windows authentication, see "Connecting to MS SQL Databases Using Windows Authentication" on page 32.

Note: By default, the Mercury Business Availability Center service is run as a system service, which does not have the necessary permissions to access the databases.

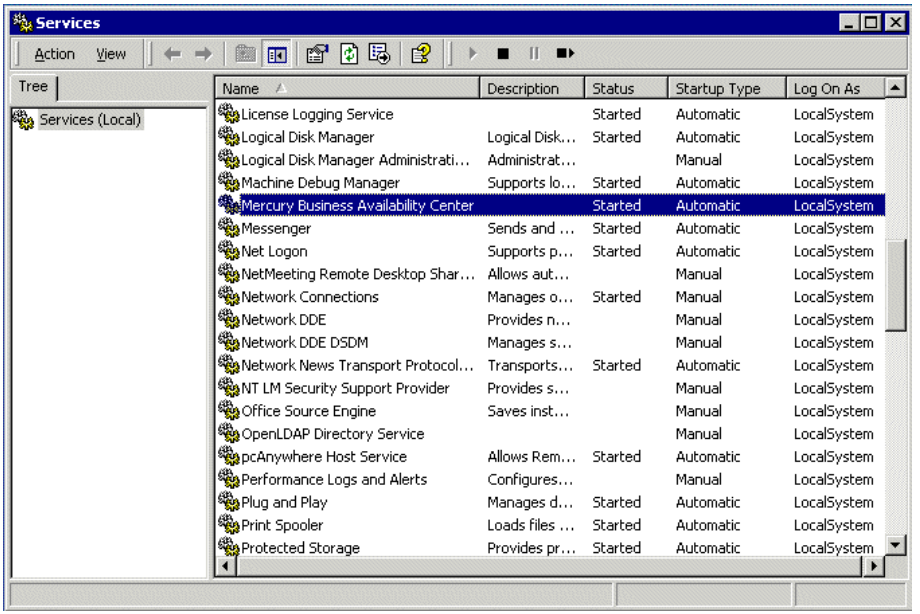
To run the Mercury Business Availability Center service as a specific user:

1 Edit the **<Mercury Business Availability Center home>\bin\ SupervisorStop.bat** file and comment out the last line by adding **rem** at the beginning of the line:

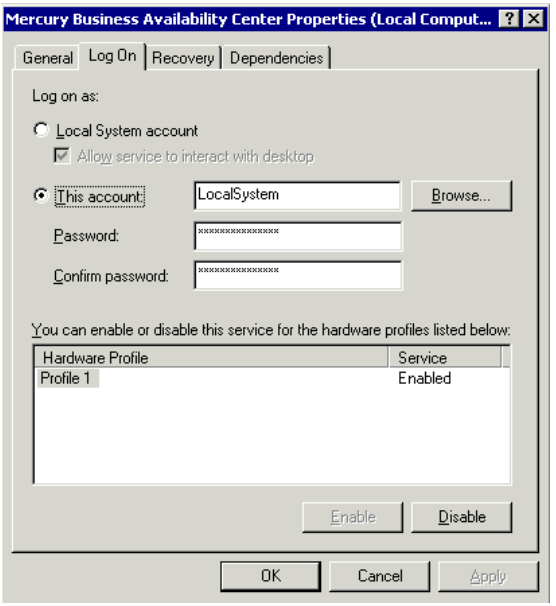
rem magentservice.exe –remove

2 Stop the Mercury Business Availability Center service by selecting **Start > Programs > Mercury Business Availability Center > Administration > Disable Business Availability Center**.

3 Select **Start >Programs >Administrative Tools > Services**. The Services window opens.



4 In the Services window, right click the Mercury Business Availability Center service and select **Properties**. The Properties window opens.



5 Select the **Log On** tab.

6 Select **This account** and enter the user name and password of the user with the permissions to access the database(s).

7 Click **OK**.

8 Start the Mercury Business Availability Center service by selecting **Start** > **Programs** > **Mercury Business Availability Center** > **Administration** > **Enable Business Availability Center**.

9 Repeat these steps for each of the Mercury Business Availability Center servers.

Note: Before uninstalling Mercury Business Availability Center, you must remove the comment marker (**rem**) that was added to the **<Mercury Business Availability Center home>\bin\SupervisorStop.bat** file in step 1, and run the **<Mercury Business Availability Center home>\bin\ SupervisorStop** batch file. For details on uninstalling Mercury Business Availability Center, see "Uninstalling Mercury Business Availability Center Servers" in *Deploying Servers*.

Note: For the Mercury Business Availability Center Purging Manager to work, the user running the Mercury Business Availability Center service must have DLL permissions. For details on the Purging Manager, see "Data Partitioning and Purging" on page 159.

Connecting to MS SQL Databases Using Windows Authentication

Once you have enabled Mercury Business Availability Center to use Windows authentication to access an MS SQL database (for details, see "Enabling Mercury Business Availability Center to Work with Windows Authentication" on page 29), you can connect to any, or all, of the Mercury Business Availability Center databases (management, profile, and CMDB) using Windows authentication instead of MS SQL Server authentication.

Note: You can use a mixture of Windows authentication and MS SQL Server authentication to connect to different databases.

Connecting to the Management Database

You can connect to an existing management database using Windows authentication, or you can create and connect to a new management database using Windows authentication.

To connect to an existing management database using Windows authentication:

1. On the Mercury Business Availability Center server you want to connect to the management database, open a Command window by selecting **Start > Run** and entering **CMD** in the **Open** field.
2. Navigate to the Mercury Business Availability Center **scripts** directory using the command:

   cd <Mercury Business Availability Center home>\scripts
3. Execute the **setmngdb** batch file with the following parameters:

   setmngdb connect MS-SQL "<sql server name>,<management database name>,,,<sql instance port>"

   where

- **sql server name** is the host name or IP address of the MS SQL server on which the management database resides.
- **management database name** is the name of the management database to which you are connecting.
- The next two parameters are for the user name and password and must be left blank for Windows authentication.
- **sql instance port** is the port number on the MS SQL server used to for connecting to the database.

Note: You cannot use the **Connect to Database** option from the Mercury Business Availability Center program menu to connect to an MS SQL database using Windows authentication as you cannot enter blank user name and password data.

To connect to an existing management database using Windows authentication:

Carry out the same steps listed for connecting to an existing management database using Windows authentication, but execute the **setmngdb** batch file using the **create** option instead of the **connect** option:

setmngdb create MS-SQL "<sql server name>,<management database name>,,,<sql instance port>"

Note: You must run the setmngdb batch file as a user that has create database permissions on the target MS SQL server.

Connecting to CMDB

You connect to an existing CMDB, or create and connect to a new CMDB, using Windows authentication on the CMDB Database Management page, accessed from the Setup and Maintenance tab in Platform Administration. For details on CMDB database management, see "Mercury Universal CMDB Management" in *Platform Administration*.

Connecting to a profile database

You connect to an existing profile database, or create and connect to a new profile database, using Windows authentication on the Database Management page, accessed from the Setup and Maintenance tab in Platform Administration. For details on profile database management, see "Database Administration" in *Platform Administration*.

5

Maintaining MS SQL Server Databases

This chapter describes the various maintenance tasks that are recommended for Mercury Business Availability Center databases created on MS SQL Servers, such as backing up databases, checking database integrity and handling fragmentation, and monitoring databases.

This chapter describes:	On page:
Backing Up Databases	35
Database Integrity and Fragmentation	39
Monitoring Databases	50
Database Maintenance – References	55

Backing Up Databases

MS SQL Server supports three main types of database backup: full, differential, and log. It also supports file/file group backup, which is discussed in a separate section below. In order to develop a backup policy that provides the required recovery needs, it is important to thoroughly understand each backup type and the recovery model database configuration option explained in the previous section.

You can automate backup operations using MS SQL Agent jobs. The MS SQL Agent (represented by the SQLServerAgent service) is installed automatically when you install MS SQL Server. Ensure that the MS SQL Agent is configured to autostart in the operating system's Services applet when the server is started.

The following points are applicable to all backup types:

- The backup includes all changes made until the backup is complete.
- The backup can be performed online, but it is recommended to back up the database during periods of low activity, since the backup procedure can negatively impact on your system's performance.
- The following operations should not be performed during a backup procedure:
  - adding or removing files
  - shrinking the database
- The backup target can be a disk device (local or on a shared network that the MS SQL Server service account needs permission to access) or tape (only local).

Full Backup

When you perform a full database backup, all information about the backed up database is contained within the backup, including data, meta data, and file information. A full backup is the basis for differential and log backups. With small databases, it is recommended to perform a full backup every day (for example, system databases that store mainly meta data). For large databases (such as the management and profile databases), it is generally recommended to have longer intervals between full backups (for example, once a week).

The storage requirements for a full backup are about the same as the storage requirements for the occupied data portion of the files. For example, if the total size of the data files is 20 GB, but only 15 GB are used (there are 5 GB of free space), the full backup size of the database should be approximately 15 GB.

Differential Backup

You use a differential backup to back up the extents (blocks of 8 contiguous 8K pages) that were changed since the last full backup. When restoring a database, you need only restore the last differential backup performed after the full backup. Note that the differential backup performance of MS SQL Server 2000 is superior to that of MS SQL Server 7.0. The difference in performance is due to the fact that MS SQL Server 7.0 scans the entire database, reading each extent to check whether it was changed after the last full backup, whereas MS SQL Server 2000 has bitmaps in each file mapping the changed extents, and the differential backup directs the disk arms to read only the relevant extents.

After performing operations that affect large portions of data, such as index rebuilds or defragmentations, it is recommended that you perform a full backup. Otherwise, differential backups can become very large. For more information on index rebuilds and defragmentation, see “Database Integrity and Fragmentation” on page 39.

Differential backup is usually scheduled at intervals between full backups. For example, if you perform a full backup once a week, you may want to perform a differential backup every day, or even several times a day.

The storage requirements for a differential backup are the total size of the extents (64 KB blocks) that were changed since the last full backup.

Log Backup

A log backup—unlike full and differential backups which are mainly based on backing up an image of extents—backs up transactions from the transaction log and replays them upon restoration. In order to perform a log backup, the database must be set to the full or bulk-logged recovery model. If you want to perform a point-in-time or log mark restoration, or back up changes recorded in the log when the data crashes, you must set the database to the full recovery model. Otherwise, all changes made since the last performed backup are lost.

A log backup is incremental in nature, and backs up only the transactions performed since the previous log backup. When restoring a database, you must restore all log backups after the last differential (or full) backup you restored.

A log backup also marks the portion of the log that was backed up as available for reuse. In a database that is set to the full or bulk-logged recovery model, log portions that were not backed up cannot be reused. When the log is full, and MS SQL Server cannot cycle to its beginning to reuse log space, it must expand. The frequency of your log backups, therefore, is a factor in determining the required size of the transaction log. Frequent log backups allow you to keep a smaller transaction log. It is recommended that you back up your log as frequently as possible, for example, every 30 minutes.

File/File Group Backup

Instead of backing up the entire database, you can back up a file or file group. However, when you restore a single file or file group, you must apply all log backups up to and including the point of failure, in order to synchronize (same point-in-time) the file/file group with the rest of the database. This type of backup is generally useful with very large databases, for which you cannot frequently perform a full backup.

Maintenance Plan

In the MS SQL Server Enterprise Manager, under the Management tree view, there is a graphic tool called Database Maintenance Plans. This tool allows you to define and automate common maintenance tasks (full and log backups, integrity checks, index rebuilds, and statistics collection).

Transaction Log Issues

In terms of maintenance, the log is sensitive. When it is full, the log first tries to cycle and reuse inactive backed up log space, but if such space does not exist, the log tries to expand the file. If there is no room for the file to expand, MS SQL Server rejects data modification requests. To avoid log explosion, ensure that the log is large enough and that it is frequently backed up (ideally, by schedule). In addition, note that the active portion of the log starts with the oldest open transaction and continues until the current pointer in the log. The active portion cannot be reused or truncated.

If a transaction remains open for a long time, it will inevitably lead to log explosion at some point, even though the log is backed up. To identify whether such a problem exists, run DBCC OPENTRAN to obtain the transaction that has been open for the longest period of time. To terminate the process running the transaction and roll back the transaction's activity, use the following command: KILL <process id>

Note: In MS SQL Server 2000, the DBCCSHRINKFILE command should always be successful.

Database Integrity and Fragmentation

It is important to periodically check the physical integrity of your database objects, and to handle index fragmentation issues that are the main cause of performance degradation.

Database Integrity

It is recommended that you run DBCC CHECKDB periodically to check the allocation and structural integrity of the objects in the database. You can automate and schedule the DBCC CHECKDB command using MS SQL Agent jobs. Use the following command syntax:

```
DBCC CHECKDB ('database name')
```

Note: You can use the WITH NO_INFOMSGS option to reduce processing and tempdb usage. You can also run a quick physical-only test (page structure and record headers) using the PHYSICAL_ONLY option.

Because the MS SQL Server 2000 database holds only schema locks (which prevent schema changes) and not data changes, the DBCC CHECKDB command can be run online. It is recommended, however, to run the DBCC CHECKDB command during periods of low activity, since it can negatively impact on your system's performance (DBCC CHECKDB is CPU- and disk-intensive, and uses tempdb for sorting).

Understanding File System Fragmentation

File system fragmentation is relevant to all disk files, not just database files. It refers to the scattering of parts of the same disk file over different areas of the disk, as new parts of the file are added and existing parts are deleted. File system fragmentation slows disk access and degrades the overall performance of disk operations, although usually not severely.

To defragment a file system, you rewrite parts of a file to contiguous sectors on a hard disk. This increases the speed of data access and retrieval. In order to avoid fragmentation of your database files, create the files with as large an initial size as possible (so that they can accommodate changes in the future), and manually expand them with large increments as they become full. If you cannot anticipate the future size of a database file, use a large value as the file growth increment in order to avoid small fragmented parts. Do not use too large a value, however, or you will experience client request timeouts when the file autogrows (for more details, see "Creating Databases" on page 13). In addition, avoid using the autoshrink database option, because it increases the chances of fragmentation as the database files continually shrink and grow.

Note: It is recommended that you periodically run a defragmentation utility.

Understanding Internal Fragmentation

Internal fragmentation refers to the percentage of data contained in the pages. In environments such as the Mercury Business Availability Center system, which are characterized by transactions that frequently insert data, internal fragmentation is sometimes initiated in anticipation of new data in indexes and can be a positive occurrence. By leaving a certain percentage of the index pages free, you can avoid page splits for a certain period of time. This is especially significant for clustered indexes, because they contain the actual data pages. You can achieve internal fragmentation by periodically rebuilding your indexes using the CREATE INDEX command, with the DROP_EXISTING and FILLFACTOR options, or the DBCC DBREINDEX command. The FILLFACTOR option specifies the fullness of the leaf level index pages.

Understanding External Fragmentation

As page splits occur in your indexes, new allocated pages are acquired from the database file. Ideally, a page split should yield the allocation of a page contiguous to the one that split. However, in practice, the space contiguous to the split page is usually already occupied. The more page splits that occur, the less the index's linked list reflects the physical layout of the pages on disk, and the greater the amount of external fragmentation.

External fragmentation impacts negatively on the performance of ordered index scans because the disk arm needs to move back and forth in order to retrieve the pages from disk. Ideally, the linked list should reflect the physical layout of the pages on disk so that when an ordered index scan is performed, the disk arm moves in one direction as it retrieves the pages from disk.

You can handle external fragmentation proactively by initiating internal fragmentation and leaving a certain percentage of the leaf level index pages free, thus avoiding page splits for a certain period of time. As mentioned earlier, internal fragmentation can be achieved by periodically rebuilding your indexes using the FILLFACTOR option. You can also handle external fragmentation by checking the external fragmentation status of your indexes, and rebuilding the indexes.

Using DBCC SHOWCONTIG

DBCC SHOWCONTIG allows you to check the levels of both your internal and external fragmentation. The following is the DBCC SHOWCONTIG syntax:

```
DBCC SHOWCONTIG
   [    ( { table_name | table_id | view_name | view_id }
          [ , index_name | index_id ]
      )
   ]
   [ WITH { ALL_INDEXES
            | FAST [ , ALL_INDEXES ]
            | TABLERESULTS [ , { ALL_INDEXES } ]
            [ , { FAST | ALL_LEVELS } ]
          }
   ]
```

Some of the DBCC SHOWCONTIG options are new to MS SQL Server 2000. Whereas in MS SQL Server 7.0 you had to specify table and index IDs, MS SQL Server 2000 allows you to specify table and index names as parameters.

The following is the output of DBCC SHOWCONTING run against the clustered index of the Order Details table in the Northwind sample database:

```
DBCC SHOWCONTIG scanning 'Order Details' table...

Table: 'Order Details' (325576198); index ID: 1, database ID: 6

TABLE level scan performed.

- Pages Scanned.....: 9

- Extents Scanned.....: 6

- Extent Switches.....: 5

- Avg. Pages per Extent.....: 1.5

- Scan Density [Best Count:Actual Count].....: 33.33% [2:6]
```

- Logical Scan Fragmentation .....: 0.00%

- Extent Scan Fragmentation .....: 16.67%

- Avg. Bytes Free per Page.....: 673.2

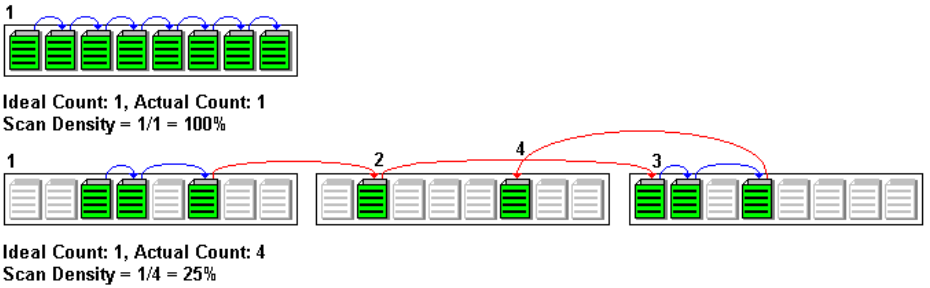
- Avg. Page Density (full).....: 91.68%

DBCC execution completed. If DBCC printed error messages, contact your system administrator.

To determine the level of internal fragmentation, check the value of the Average Page Density, which contains the average percentage of page population. The closer the Average Page Density is to 100 percent, the less internal fragmentation there is.

To determine the level of external fragmentation, check the following three values:

- **Scan Density** – the most important value to check for in order to identify external fragmentation. The scan density is the ratio between the ideal extent changes if everything is contiguously linked, and the actual extent changes. For example, if your index consumes 8000 pages, the ideal count is 1000. As the fragmentation level of your index increases and an ordered index scan switches back and forth between the extents, the actual count increases. Scan density is measured by percentage: 100 percent indicates that there is no external fragmentation, and any number less than a 100 percent indicates the level of fragmentation. The following figure displays an example of how scan density is calculated:



- **Logical Scan Fragmentation –** the percentage of out-of-order pages returned from scanning the leaf pages of an index. An out-of-order page is one for which the next page indicated in an IAM (Index Allocation Map) is a different page than the page pointed to by the next page pointer in the leaf page. The value of logical scan fragmentation should be as low as possible—ideally 0%.

- **Extent Scan Fragmentation –** the percentage of out-of-order extents in scanning the leaf pages of an index. An out-of-order extent is one for which the extent containing the current page for an index is not physically the extent after the one containing the previous page for an index. The value of extent scan fragmentation should also be as low as possible—ideally 0%.

When to Defragment Indexes

It is recommended that you defragment indexes using the techniques in the following section when the Logical Scan Fragmentation values are equal to, or higher than, 10%. When the Logical Scan Fragmentation values are equal to, or higher than, 20%, it is strongly recommended that you defragment indexes.

Handling Fragmentation

MS SQL Server 2000 provides several methods enabling you to handle internal and external fragmentation. You can rebuild a specific index using the DROP_EXISTING option of the CREATE INDEX statement, or you can rebuild all indexes belonging to a table by using the DBCC DBREINDEX statement. When the index is rebuilt, there is no external fragmentation, because all of its pages are laid on disk in the order of the linked list. Note that you can optionally specify a fillfactor to initiate internal fragmentation.

The above two fragmentation handling options are preferable to dropping and recreating your indexes for two reasons. First, you cannot drop an index created by a PRIMARY KEY or UNIQUE constraint. In order to drop this index, you must drop the constraint. Second, when you drop a clustered index, all nonclustered indexes are recreated in order to accommodate the changed row locators. The same thing occurs when you recreate the clustered index. If the clustered index is unique (uses the DROP_EXISTING option of the CREATE INDEX statement or DBCC DBREINDEX), the nonclustered indexes remain untouched when the clustered index is rebuilt.

The drawback of the above two fragmentation handling options is the effect they have on the users that access the data while the index is being rebuilt. When a clustered index is rebuilt, an exclusive lock is acquired on the table so that both the modification and retrieval of data from the table are not allowed. When a nonclustered index is rebuilt, a shared lock is acquired on the table so that data modification is not allowed. Although data retrieval is allowed, the index cannot be used by queries while it is being rebuilt.

If you want to handle external fragmentation and allow data retrieval and modification at the same time, you can use the DBCC INDEXDEFRAG statement, introduced in MS SQL Server 2000. The DBCC INDEXDEFRAG statement uses the fillfactor that was specified in the CREATE INDEX statement; you cannot specify a new fillfactor. Note that DBCC INDEXDEFRAG uses the bubble algorithm to defragment an index, swapping couples of pages until their order from left to right reflects the order of the linked list. However, it does not attempt to restructure the index in order to lay the index pages contiguously on disk. For this reason, it is recommended that you periodically rebuild your indexes using the DROP_EXISTING option or DBCC DBREINDEX.

The following table summarizes the differences between the three defragmentation options:

Option	Create Index with DROP_EXISTING	DBCC DBREINDEX	DBCC INDEXDEFRAG
Locked table	Clustered – exclusive (no modification, no retrieval); Nonclustered – shared (no modification)	Clustered – exclusive (no modification, no retrieval); Nonclustered – shared (no modification)	Online – table is not locked
Granularity	Index	Table (all indexes) or index level	Index
Page sorting	Yes	Yes	Yes

Option	Create Index with DROP_EXISTING	DBCC DBREINDEX	DBCC INDEXDEFRAG
Contiguity handling	Yes	Yes	No
Ability to specify fillfactor	Yes	Yes	No – attempts to apply original

For more information on defragmenting indexes, refer to the Microsoft SQL Server 2000 Index Defragmentation Best Practices site (http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/ss2kidbp.mspx).

Note: It is strongly recommended that you create an automatic index rebuild task for the management database and CMDB, if separate from the management database, as data is frequently changed.

Handling CMDB Index Fragmentation

Since Microsoft SQL Server 2000 maintains indexes to reflect updates to their underlying tables, these indexes can become fragmented. Depending on workload characteristics, this fragmentation can adversely affect workload performance.

Fragmentation exists when indexes have pages in which the logical ordering, based on the key value, does not match the physical ordering inside the data file. All leaf pages of an index contain pointers to the next and previous pages in the index. This forms a doubly linked list of all index/data pages. Ideally, the physical order of the pages in the data file should match the logical order. Overall disk throughput is increased significantly when the physical and logical ordering of the data match. This leads to much better performance for certain types of queries. When the physical ordering does not match the logical ordering, disk throughput can become less efficient because the disk head must move back and forth to gather the index pages, instead of scanning forward in one direction only.

Fragmentation affects I/O performance, but has no effect on performance of queries whose data pages reside in the SQL Server data cache.

When indexes are first built, little or no fragmentation should exist. Over time, as data is inserted, updated, and deleted, fragmentation levels on the underlying indexes may begin to rise.

The main criteria used to determine if indexes are fragmented are:

- **Logical scan fragmentation** – shows the ratio of pages that are out of order. This percentage should be between 0% and 10%, with anything higher indicating external fragmentation.
- **Scan Density** – shows the ratio between the Best Count (the number of pages if they were all full) and the Actual Count of extents. The Actual Count is higher as pages are not full due to fragmentation. This percentage should be as close as possible to 100%.

Mercury Business Availability Center provides two utilities that can be used to detect and rebuild fragmented indexes in the CMDB schema. The utilities use the Logical scan fragmentation and Scan Density criteria to detect, and if instructed to rebuild, fragmented indexes. The operation of listing the fragmented tables has a very small impact on system performance and can be executed on line. The operation of rebuilding the indexes may hinder performance as tables are partially locked during the process, and CPU and I/O are heavily utilized. It is recommended to rebuild the indexes in a maintenance window. The utilities should be run by a systems or database administrator.

The utilities are located in the **<Mercury Business Availability Center root directory>\MercuryAM\CMDB\dbscripts\ms\utils** directory on the Mercury Business Availability Center Modeling Data Processing Server.

Utility to Rebuild All Indexes in Database

The **rebuild_indexes.bat** utility runs through all tables in the database and rebuilds them.

To run the rebuild_indexes.bat utility:

Execute **rebuild_indexes.bat** with the following parameters:

- SQL Server name
- Database name
- SA password

Output from the procedure is located in the **rebuild_indexes.log** file in the same directory.

Utility to Rebuild Indexes Based on the Fragmentation Level of Each Index

The **rebuild_fragmented_indexes.bat** utility has two working modes:

- **List fragmented tables** – in this mode, a list of the fragmented tables (that is, tables with over 30% fragmentation) is returned, together with the commands needed to rebuild the tables at a later time.
- **Rebuild fragmented tables** – in this mode, all fragmented tables (that is, tables with over 30% fragmentation) are rebuilt.

To run the rebuild_fragmented_indexes.bat utility:

Execute **rebuild_fragmented_indexes.bat** with the following parameters:

- SQL Server name
- Database name
- SA password
- Working mode – 0 to provide a rebuild script for later use; 1 to rebuild indexes automatically.

Output from the procedure (a list of fragmented tables and the rebuild commands) is located in the **rebuild_indexes.log** file in the same directory.

Note: If CMDB is part of the management database (which is the default), run the utilities for the management database schema. If you have created a separate CMDB, run the utilities for the CMDB schema.

Distribution Statistics

Microsoft SQL Server 2000 allows statistical information regarding the distribution of values in a column to be created. This statistical information can be used by the query processor to determine the optimal strategy for evaluating a query. When an index is being created, SQL Server automatically stores statistical information regarding the distribution of values in the indexed column(s). The query optimizer in SQL Server uses these statistics to estimate the cost of using the index for a query. As the data in a column changes, index and column statistics can become out of date and cause the query optimizer to make less than optimal decisions on how to process a query.

It is recommended to update index statistics to provide the query optimizer with up to date information about the distribution of data values in the tables. This allows the query optimizer to make better judgments about the best way to access data, as it has more information about the data stored in the database.

By default, the **auto update statistics database** option is enabled, but if this option has been disabled, it is strongly recommended that you create an automatic task to update statistics for the management database and CMDB, if separate from the management database, on a daily basis, as the data is frequently changed. The job should execute the **sp_updatestats** API against the specific database.

Note: This is not necessary for the profile databases, as the data is only being added to or purged, and SQL Server automatically updates this statistical information on a random basis.

In addition to running a daily task to update statistics for the management and CMDB databases if the **auto update statistics database** option is not enabled, it is recommended that you manually refresh statistics for CMDB if major changes to the CMDB schema objects have occurred, usually caused by bulk insert transactions. The following scenarios warrant a manual refresh of CMDB statistics:

- **Automated Discovery tasks** – Discovery Manager is the process responsible for automatically detecting configuration items (CIs) and inserting them into CMDB.
- **Mercury Business Availability Center adapters synchronization** – when expected to load substantial data to CMDB. You synchronize adapters from the Source Manager tab, accessed from Admin > CMDB.

To manually refresh CMDB statistics:

1. In a Web browser, open:
   http://<Centers Server machine name>:8080/jmx-console
2. In the **Topaz** section, select:
   CMDB Dal Services
3. Under **runStatistics**, enter the customer ID. The default customer ID for an individual Mercury Business Availability Center system (that is, one not managed by Mercury Managed Services) is 1.
4. Under **runStatistics**, click **Invoke**. The CMDB statistics are regenerated.

Monitoring Databases

This section describes the recommended system monitor counters for baseline/daily monitoring.

A baseline allows you to track the different counters during normal system activity, and learn what the counter values are when the system is operating well. In problematic situations, you can examine and compare current counter values with the values in the baseline.

The *SQL Server 2000 Operations Guide* (http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/sqlops0.mspx) recommends tracking the following counters to create a baseline and monitor it:

Counter	Description
Memory - Pages/second	The number of pages read from, or written to, disk to resolve hard page faults. (Hard page faults occur when a process requires code or data that is not in its working set or elsewhere in physical memory, and must be retrieved from disk). This counter is designed as a primary indicator of the types of faults that cause system-wide delays, and is the sum of **Memory: Pages Input/sec** and **Memory: Pages Output/sec**. It is measured in numbers of pages, so that it can be compared to other page counts, such as **Memory: Page Faults/sec**, without conversion. It includes pages retrieved to satisfy faults in the file system's non-cached mapped memory files. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.
Network Interface - Total Bytes/second	The number of bytes traveling over the network interface per second. If this rate begins to drop, you should investigate whether network problems are interfering with your application.
PhysicalDisk - Disk Transfers/second	The rate of read and write operations on the disk. You should define a counter for each physical disk on the server.

Counter	Description
Processor - & Processor Time	The percentage of time that the processor is executing a non-idle thread. This counter is designed as a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the idle process in each sample interval, and subtracting this value from 100 percent. (Each processor has an idle thread that consumes cycles when no other threads are ready to run). It can be viewed as the percentage of the sample interval spent doing useful work. This counter displays the average percentage of busy time observed during the sample interval, which is calculated by monitoring the time the service was inactive, and then subtracting that value from 100 percent. After all of the processors devoted to MS SQL Server have reached 100 percent utilization, it is likely that end user requests are being ignored.
SQLServer:Access Methods - Full Scans/second	The number of unrestricted base table or full index scans.
SQLServer:Buffer Manager - Buffer Cache Hit Ratio	The percentage of pages that were found in the buffer pool, without having to incur a read from disk. When this percentage is high, your server is operating at optimal efficiency (as far as disk I/O is concerned).
SQLServer:Databases - Log Growths (run against your application database instance)	The total number of log growths for the selected database.
SQLServer:Databases Application Database - Percent Log Used (run against your application database instance)	The percentage of space in the log that is in use.

Counter	Description
SQLServer:Databases Application Database - Transactions/second (run against your application database instance)	The number of transactions started for the database.
SQLServer:General Statistics - User Connections	The number of users connected to the system. Dramatic shifts in this value should be investigated.
SQLServer:Latches - Average Latch Wait Time	The average latch wait time (in milliseconds) for latch requests that had to wait. If this number is high, your server may be facing contention for its resources.
SQLServer:Locks - Average Wait Time	The average amount of wait time (milliseconds) for each lock request that resulted in a wait.
SQLServer:Locks - Lock Waits/second	The number of lock requests that could not be satisfied immediately and required the caller to wait before being granted the lock.
SQLServer:Locks - Number of Deadlocks/second	The number of lock requests that resulted in a deadlock.
SQLServer:Memory Manager - Memory Grants Pending	The current number of processes waiting for a workspace memory grant.

The following counters can help you identify hardware problems:

Counter	Description
Network Interface()\Packets Outbound Errors	The number of outbound packets that could not be transmitted due to errors.
Network Interface()\Packets Received Errors	The number of inbound packets that contained errors preventing the packets from being delivered to a higher-layer protocol.
Server\Errors System	The number of times that an internal server error was detected. Errors can reflect problems with logon, security, memory allocation, disk operations, transport driver interface operations, communication [such as receipt of unimplemented or unrecognized server message blocks (SMBs)], or I/O request packet stack size for the server. Many of these errors are also written to the system log and security log in the Event Viewer. The server can recover from most the errors displayed by this counter, but they are unexpected and should be reported to Microsoft Product Support Services.

For information on creating a baseline, refer to the system monitor documentation, or to the *SQL Server 2000 Operations Guide* (http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/sqlops0.mspx).

Database Maintenance – References

For extensive information on MS SQL Server performance tuning, refer to the following documents:

- An approach to database optimization: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/sql/reskit/sql2000/part10/c3361.asp
- Troubleshooting application performance with MS SQL Server: http://support.microsoft.com/default.aspx?scid=KB;EN-US;q224587
- Performance counters: http://www.databasejournal.com/features/mssql/article.php/1477311#disk
- How to perform an MS SQL Server performance audit: http://www.sql-server-performance.com/sql_server_performance_audit.asp
- MS SQL Server performance tuning tips: http://www.sql-server-performance.com/best_sql_server_performance_tips.asp
- Operations guide: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/sql/maintain/operate/opsguide/default.asp
- General MS SQL Server performance tuning page with other links: http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/rdbmspft.mspx
- MS SQL Server MSDN performance tuning section: http://msdn.microsoft.com/SQL/sqlperf/ default.aspx

6

MS SQL Server Sizing Guidelines

This chapter contains guidelines for the server and database configuration settings that should be used when working with MS SQL Server and Mercury Business Availability Center. Note that the recommended settings generally differ according to the size of your Mercury Business Availability Center deployment.

This chapter describes:	On page:
Mercury Business Availability Center Sizing	58
Hardware Sizing	59
Server Configuration Options	60
Data File Property Settings	60
tempdb Database Settings	61

Mercury Business Availability Center Sizing

Mercury Business Availability Center database configuration requirements are dependent on the amount of data generated by Mercury Business Availability Center. For example, you would use a small database for data generated over a period of 300 days by three Business Process Monitors, each running five Business Process profiles executing 10 transactions and WebTrace every 15 minutes, with five percent of the transactions returning errors.

You would use a large database for data generated over a period of 250 days by 10 Business Process Monitors, each running 50 SiteScope profiles executing approximately 600 measurements, and 30 Business Process profiles executing 10 transactions and WebTrace every 15 minutes, with five percent of the transactions returning errors.

The following table describes the database size required for each database element, based on the amount of data generated:

Database Element	Mercury Business Availability Center Deployment		Remarks
	Small	Large	
Insert ratio (rows per hour)	6 KB or less	500 KB or more	
Named users (profile database)	fewer than 20	80 or more	Used for memory settings
Named users (management database)	fewer than 30	50 or more	Used for process settings
Database size	approximately 30 GB	approximately 600 GB and greater	

Hardware Sizing

The following table describes the hardware (CPU and memory) requirements for Mercury Business Availability Center MS SQL Server database certification:

Mercury Business Availability Center Deployment	Number of Processors	Physical Memory	Recommended MS SQL Version
Small	1-2	1-2 GB	Microsoft SQL Server 2000 Enterprise, Service Pack 4
Large	8	8 GB or more	Microsoft SQL Server 2000 Enterprise, Service Pack 4

Note: You can find online database sizers for HP/Compaq database servers on the following Web pages:
http://activeanswers.compaq.com/ActiveAnswers/Render/1,1027,5276-6-100-225-1,00.htm
http://activeanswers.compaq.com/ActiveAnswers/Render/1,1027,536-6-100-225-1,00.htm

---

Server Configuration Options

You can reconfigure the following MS SQL Server option:

- **awe enabled –** When MS SQL Server must access more than 4 GB of physical memory, you can use the Microsoft Windows 2000 Address Windowing Extensions (AWE) API to support up to a maximum of 64 GB. For more information, refer to MS SQL Server Books Online and the *SQL Server 2000 Operations Guide* (http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/sqlops0.mspx).

Data File Property Settings

You can configure the following MS SQL Server data file properties:

- **SIZE –** The MS SQL Server allocates the size specified in this parameter when the database is created, and "zeroes" the space allocated (this may take a while). Since MS SQL Server tries to acquire as large a consecutive amount of space as possible, it is recommended that you specify as large a value as possible in order to avoid file system fragmentation caused by future file growth.
- **FILEGROWTH –** The automatic growth increment of the file can be specified as either a percentage of the existing file size, or as a fixed size. Using a small growth increment is not recommended because it increases file system fragmentation. On the other hand, if you have a very large increment, modifications sent by clients might incur connection timeouts while waiting for the automatic expansion to finish. This problem can occur a while after the database was created and come as a surprise when the files are set with the default ten percent growth increment. Note that when the database is small, the automatic growth is completed in a short time, but when the database becomes large (for example, several gigabytes), the default ten percent growth increment may take a while to allocate and initialize.

tempdb Database Settings

The frequent expansion of the tempdb system database can affect the database's performance, especially in large MS SQL Server installations. The size of the tempdb, therefore, should be large enough to avoid the need for early expansion. Its growth increment should be large enough to avoid fragmentation, yet not too large to expand in a reasonable amount of time. Create the tempdb with a minimum, initial size of 500 MB and with a growth increment of 50 MB. The tempdb database should be striped across several disks, ideally on a RAID 0+1 controller. It is recommended to move the tempdb database to its own set of disks.

To ensure that there is enough disk space for the tempdb to grow during times of heavy usage (for example, when aggregating or sorting data), it is recommended that you leave at least 20 GB free disk space on the drive where the tempdb is located.

7

MS SQL Server Summary Checklists

This chapter contains a checklist summarizing the server and database configuration options that are supported and recommended for working with Mercury Business Availability Center. It also contains a table summarizing the procedures for verifying or modifying server and database settings.

This chapter describes:	On page:
Checklist for Mercury Business Availability Center Support and Certification	63
Verifying and Modifying Server and Database Settings	65

Checklist for Mercury Business Availability Center Support and Certification

The following checklist summarizes the server and database configuration options that are supported and certified for working with Mercury Business Availability Center:

Subject	MS SQL Server 2000	
	Supported	Recommended
Instances	Default, Single	
Authentication Mode	Mixed	

Subject	MS SQL Server 2000	
	Supported	**Recommended**
Collation	Case-Insensitive, Accent-Sensitive, Kana-Insensitive, Width-Insensitive	Windows Locale, with the following options selected: Latin1_General, Accent-Sensitive
Network Libraries	**Server:** TCP/IP and Named Pipes **Client:** TCP/IP and Named Pipes	**Server:** TCP/IP **Client:** TCP/IP
Server Configuration Options	Defaults, unless instructed otherwise	
Data File Properties	Manual file growth, or FILEGROWTH less than or equal to 100 MB	FILEGROWTH: ~30-100 MB
Collation Database Property	Server default	
Database Options	Defaults, unless instructed otherwise	
Recovery Model	Any	Full

Verifying and Modifying Server and Database Settings

The following table summarizes the procedures for verifying or modifying server and database settings:

Server/Database Setting	How to Verify/Modify the Setting
Default Instance	In the operating system's Services applet, a default MS SQL Server instance appears as **MSSQLServer**, while a named instance appears as **MSSQL$INSTNAME**. Ensure that MS SQL Server is installed as a default instance.
Authentication Mode	In the MS SQL Server Enterprise Manager, right-click the server, choose **Properties**, and click the **Security** tab. Select **Mixed Mode**.
Collation Settings	Run: sp_helpsort **Required results:** Latin1-General, case-insensitive, accent-sensitive, kanatype-insensitive, width-insensitive
Network Libraries	**Server:** Run **svrnetcn.exe**; **Client:** Run **cliconfg.exe**. **Supported:** TCP/IP and Named Pipes for both the server and client. **Recommended:** Only TCP/IP for both the server and client.

Server/Database Setting	How to Verify/Modify the Setting
View or change server configuration options	➤ To allow the viewing of all options, run: EXEC sp_configure 'show advanced options', 1 reconfigure with override ➤ To view the current values run: EXEC sp_configure ➤ To change a setting, run: EXEC sp_configure '<option>', <value> Some options take effect only after you run reconfigure with override, while others require restarting the MSSQLServer service. See MS SQL Server 2000 Books Online for details.
Check whether the user has CREATE DATABASE permissions	Log in to the Query Analyzer with the login you want to check, and run the following: USE master IF PERMISSIONS() & 1 = 1 PRINT 'User has CREATE DATABASE permissions' ELSE PRINT 'User does not have CREATE DATABASE permissions'
Check whether the user is dbo in the database	Log in to the Query Analyzer with the login you want to check. Change the context of the database to the required database, and run the following: SELECT USER_NAME()
Verify the database owner	Run: EXEC sp_helpdb <database name>
Data and log file destination directory is not compressed (only in NTFS)	Right click the directory, choose **Properties**, and then **Advanced**. Verify that the **Compression** check box is cleared.

Server/Database Setting	How to Verify/Modify the Setting
Database and database file properties (including recovery model and collation properties)	➤ To view the database and database file properties, run: EXEC sp_helpdb <database name> ➤ To change the database properties: ALTER DATABASE <database name> SET <option> <value> ➤ To change the database file properties: ALTER DATABASE <database> MODIFY FILE (name = <filename>, <property> = <value>) You can also view or change these properties from the Database Properties dialog box in the Enterprise Manager.
MS SQL Server service pack version and edition	To verify the version and edition of your MS SQL Server service pack, refer to the following URL: http://support.microsoft.com/default.aspx?scid=kb;en-us;q321185

8

Installing and Configuring MS SQL Server

This chapter contains information on the MS SQL Server installation procedure and configuration settings.

This chapter describes:	On page:
Installing MS SQL Server	69
Configuring MS SQL Server	78

Installing MS SQL Server

You install MS SQL Server by running **setupsql.exe** from the **<MS SQL Server root directory>\x86\setup** directory. Although the installation process is not difficult, it is important that you familiarize yourself with all of the installation details so that you select the appropriate options. Note that selecting the default options, in some cases, may negatively affect the MS SQL Server's performance.

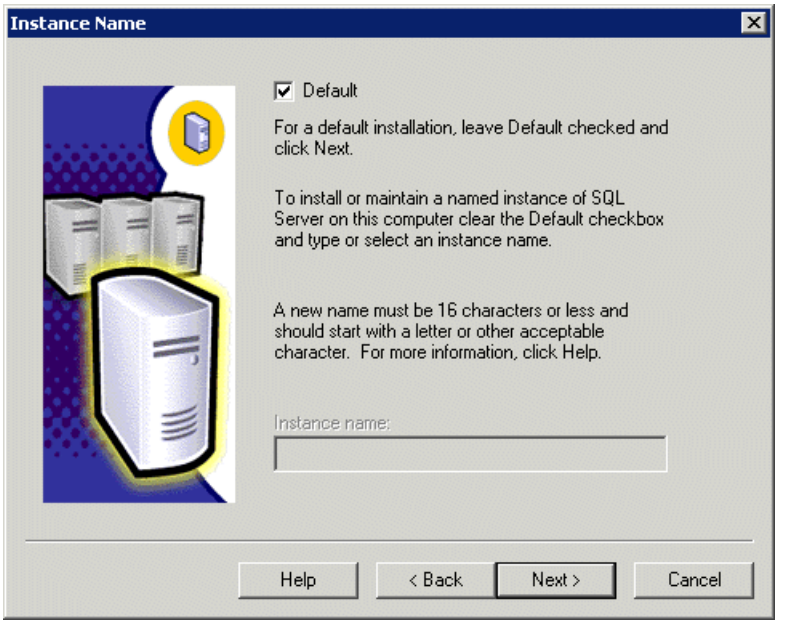
The following sections describe the dialog boxes to which you should pay particular attention during the installation process.

Instance Name Dialog Box

You can install multiple instances of MS SQL Server 2000 on the same machine. Each instance has its own MS SQL Server and MS SQL Agent services, and is completely independent of the other instances.

You can install only one instance as the default instance. All the other instances must be installed as named instances, the names of which you provide during the installation process. You access a default instance by specifying the server name or IP address, and a named instance by specifying the server name or IP address followed by \<instance_name>, for example, **server1\inst1**.

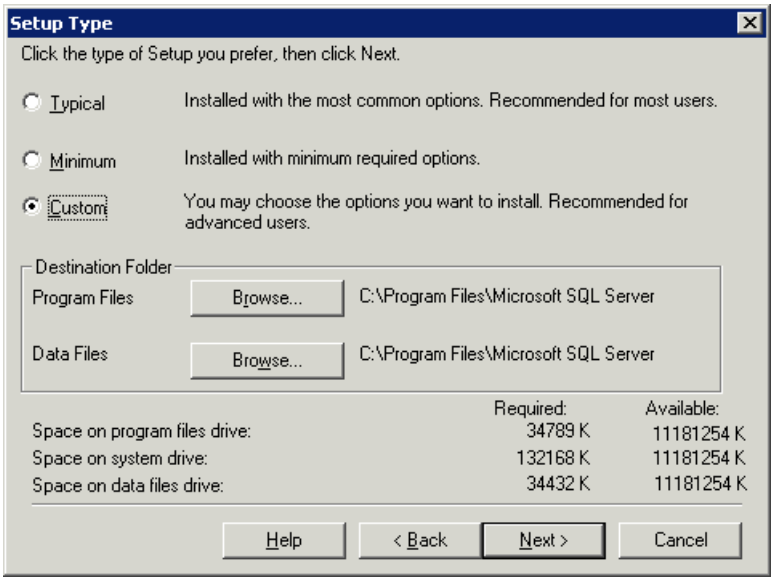
When working with Mercury Business Availability Center, you must install MS SQL Server 2000 as the default instance. You do so by selecting the **Default** check box in the Instance Name dialog box.



For information on verifying this setting after the installation process, see "Verifying and Modifying Server and Database Settings" on page 65.

Setup Type Dialog Box

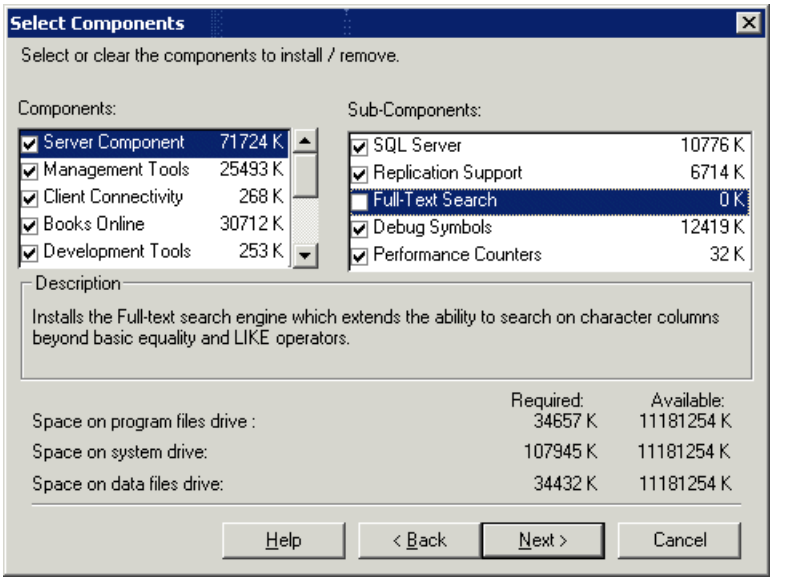
In the Setup Type dialog box, you must select the **Custom** option. If you do not select this option, you will not be able to view the Component Selection, Collation Settings, and Network Libraries dialog boxes, in which you must change certain default options that are not appropriate for working with MS SQL Server in conjunction with Mercury Business Availability Center.



Under Destination Folder, you set the Program Files directory in which MS SQL Server executables are stored, and the Data Files directory, in which system databases, and user databases for which a file location is not specified, are stored. Note that these settings cannot be changed later on. Although you can relocate tempdb files (using the ALTER DATABASE tempdb MODIFY FILE command) and user database files (detach, move, attach) at a later stage, you cannot move master, msdb, and model files. Even though these system databases are fairly small, they are essential for the operation of MS SQL Server. Ensure that the Data Files directory is stored on a fault-tolerant disk system, for example, RAID 1.

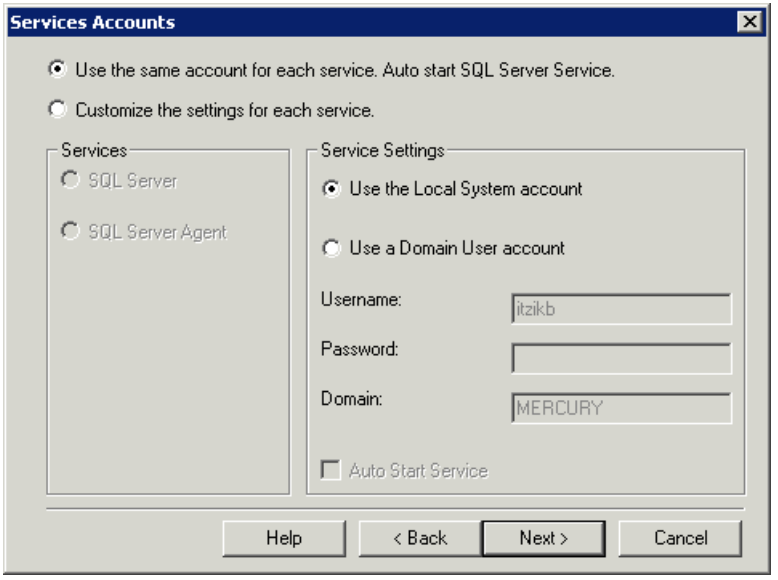
Select Components Dialog Box

In the Select Components dialog box, ensure that the **Full-Text Search** check box is not selected. Mercury Business Availability Center does not require this indexing service.



Services Accounts Dialog Box

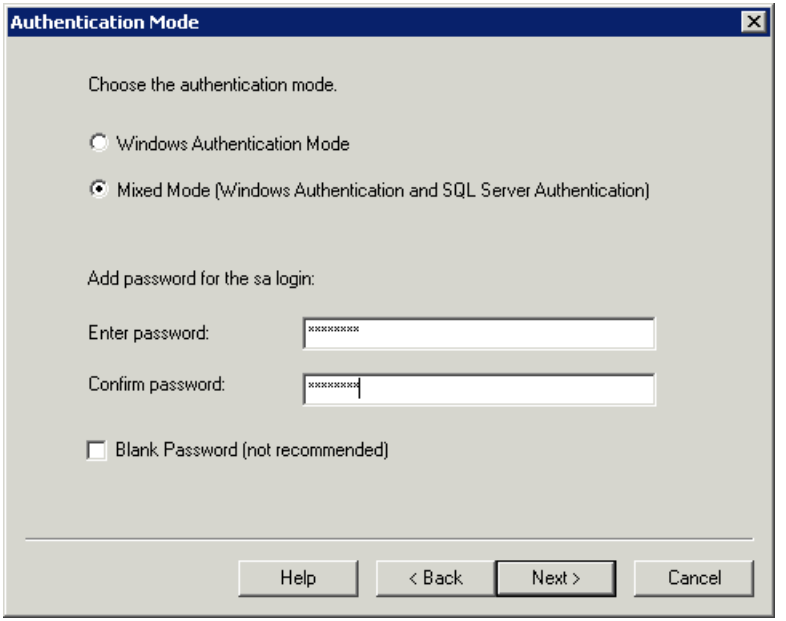
In the Services Accounts dialog box, you select the services and enter the appropriate service settings for your MS SQL Server. If you want the MS SQL Server service (MSSQLServer) to perform activities outside the local machine—for example, backing up files to a shared network directory—select **Use a Domain User account** and specify the user name, password, and domain of a user that is a member of the local machine's administrator group, and that has the appropriate permissions for network resources. If all MS SQL Server activities are limited to the local machine, select **Use the Local System account**, which provides administrative privileges to the MS SQL Server service for the local machine only.



Similarly, if you want the MS SQL Server Agent service (SQLServerAgent) to perform activities that require permissions outside the local machine (for example, replication with other servers, ActiveX script job steps, or CmdExec job steps), select **Use a Domain User account** and specify the user name, password, and domain of a user that is a member of the local machine's administrator group. If all MS SQL Server Agent activities are limited to the local machine, select **Use the Local System account**.

Authentication Mode Dialog Box

In the Authentication Mode dialog box, you select the type of authentication you want MS SQL Server to use. Mercury Business Availability Center works with MS SQL Server authentication, which is disabled by default. To enable MS SQL Server authentication, select **Mixed Mode** and provide a complex password for the **sa** login.



Note: To enhance the security of your MS SQL Server, it is important to enter a password.

The sa user has system administrator privileges and can therefore perform all actions within the MS SQL Server. Similarly, the sa user can perform all operating system/network actions in the context of the MSSQLServer service account, by using the xp_cmdshell extended procedure.

For information on modifying or verifying the authentication mode after installation, see “Verifying and Modifying Server and Database Settings” on page 65.

Collation Settings Dialog Box

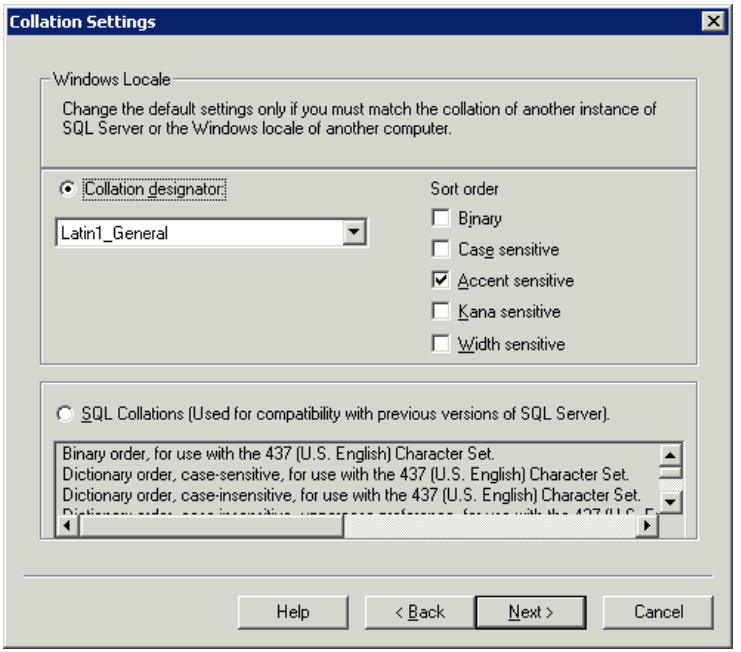
In the Collation Settings dialog box, you set the language, dictionary/binary order, and sensitivity of character data types.

Select one of the following two options:

- **Windows Locale –** Select this option only if you do not need to create backward compatibility with previous versions of MS SQL Server (for example, for replication purposes). If you select this option, set the following for Mercury Business Availability Center certification:
  - **Collation Designator –** the language for regular character data types (char, varchar, text). Select the default option, **Latin1_General**, to support English.
  - **Sort Order –** binary order and sensitivity (case, accent, kana, width) of character data types. Select the **Accent sensitive** option only, and ensure that the other options are cleared.

Note: Mercury Business Availability Center does not support case sensitivity with MS SQL Server 2000.

- **SQL Collations –** Select this option if the current version of MS SQL Server must be compatible with previous versions, for example, if you are replicating data between servers.



The above settings affect only the system databases and serve as the default settings for user databases. Databases can have different collation settings from the server's default settings, and a table column can have different collation settings from the database's default settings. Because of the flexibility in collation management in MS SQL Server 2000, you can restore or attach a database that has different collation settings.

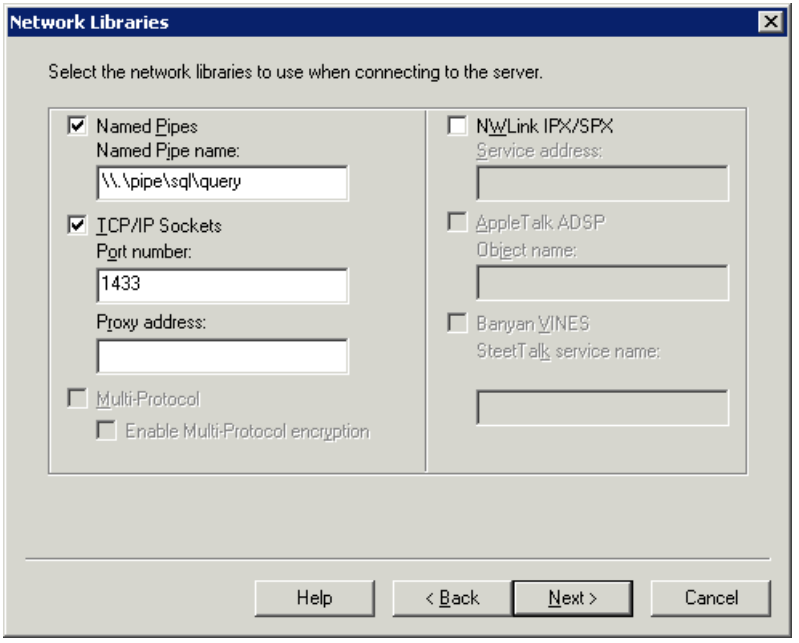
Note: Changing any of the above settings requires scripting all system objects and routines (logins, user defined system messages, master stored procedures, and so forth), reinstalling MS SQL Server (or running the RebuildM.exe utility) with the new settings, recreating all system objects from the saved scripts, and attaching the user databases. It is therefore recommended that you select the appropriate options during the installation process.

For information on verifying the MS SQL Server 2000 collation settings, see "Verifying and Modifying Server and Database Settings" on page 65.

Network Libraries Dialog Box

In the Network Libraries dialog box, you set the session level protocol by which MS SQL Server listens to client connections.

By default, MS SQL Server listens to client connections via both Named Pipes and TCP/IP, which are both supported by Mercury Business Availability Center. For Mercury Business Availability Center certification, however, you should select the **TCP/IP Sockets** option only.



You can configure the client machine to connect to MS SQL Server using both TCP/IP and Named Pipes. However, for Mercury Business Availability Center certification, you should configure the client machine to connect to MS SQL Server using TCP/IP only.

For information on modifying or verifying the above settings after installation, see "Verifying and Modifying Server and Database Settings" on page 65.

Note: All network libraries support Windows authentication and SSL data encryption.

Configuring MS SQL Server

This section describes the service and server options you can configure once you have installed MS SQL Server.

Service Configuration Options

If you installed Full-Text Search, ensure that it is disabled (locate the service in the Services applet using Microsoft Search) so that no resources are unnecessarily wasted.

Unless you are using distributed transactions, ensure that the Distributed Transactions Coordinator service is also disabled or set to manual mode.

Similarly, ensure that all unnecessary services are not set to automatic startup mode.

Server Configuration Options

Most server configuration options are dynamically configured by MS SQL Server (usually set to 0). For Mercury Business Availability Center certification, you should not change the default options unless you are instructed to do so by Mercury Customer Support.

There are specific situations in which you may want to change the default settings. You can change these settings in the sp_configure stored procedure, or in the various dialog boxes in MS SQL Server Enterprise Manager (mainly the Server Properties dialog box).

The following table describes the configuration options available in MS SQL 2000, their default settings, and the settings required for Mercury Business Availability Center certification:

Configuration Option	Default	Mercury Business Availability Center Certification in MS SQL Server 2000
affinity mask	0	Default
allow updates	0	Default
awe enabled (Enterprise Edition)	0	Default, unless the server needs to access 4 to 64 GB of memory
c2 audit mode	0	Default
cost threshold for parallelism	5	Default
cursor threshold	-1	Default
default full-text language	1033	Default
default language	0	Default
fill factor	0	Default
index create memory	0	Default
lightweight pooling	0	Default
locks	0	Default
max async IO	32	N/A
max degree of parallelism	0	Default
max server memory	2,147,483,647	Default
max text repl size	65,536	Default
max worker threads	255	Default
media retention	0	Default
min memory per query	1024	Default
min server memory	0	Default
Using Nested Triggers	1	Default

Configuration Option	Default	Mercury Business Availability Center Certification in MS SQL Server 2000
network packet size	4096	Default
open objects	0	Default
priority boost	0	Default
query governor cost limit	0	Default
query wait	-1	Default
recovery interval	0	Default
remote access	1	Default
remote login timeout	20	Default
remote proc trans	0	Default
remote query timeout	600	Default
scan for startup procs	0	Default
set working set size	0	Default
show advanced options	0	Default
spin counter	0, 10,000	N/A
time slice	100	N/A
two digit year cutoff	2049	Default
user connections	0	Default
user options	0	Default

You can view all of the above configuration options by running the following:

```
EXEC sp_configure 'show advanced options', 1
reconfigure with override
```

To view the current values for each of the options, run: `EXEC sp_configure`

For large installations, you may need to set the **awe enabled** option. For more information, see Chapter 6, "MS SQL Server Sizing Guidelines."

Note: It is strongly recommended that no significant processes other than a single MS SQL Server installation be installed on the server that hosts the Mercury Business Availability Center databases. When MS SQL Server is the sole significant process on the machine, you should not change the default memory settings. You should allow the MS SQL Server to manage memory dynamically (except when you configure **awe enabled** support).

To reconfigure an option, run EXEC sp_configure '<option>', <value>. Note that some options take effect after running reconfigure with override, while others require restarting the MSSQLServer service.

Part III

Deploying and Maintaining the Oracle Server Database

9

Overview of Oracle Server Deployment

You can set up management and profile databases on an Oracle Server. This chapter describes the following topics related to deploying Oracle Servers for use with Mercury Business Availability Center:

This chapter describes:	On page:
About Oracle Server Deployment	85
System Requirements	87
Choosing an Edition of Oracle Server	88

About Oracle Server Deployment

To deploy Oracle Server for use with Mercury Business Availability Center, perform the following procedures:

➤ **Install Oracle Server.**

For details, refer to the installation guide for your specific Oracle platform.

➤ **Build a database on Oracle Server to store management and profile data.**

For details, refer to the installation guide for your specific Oracle platform.

➤ **Create one or more Oracle tablespaces to store management and profile data.**

For details, see "Mercury Business Availability Center Oracle Tablespaces" on page 92.

➤ **Configure the Oracle Client for Mercury Business Availability Center.**

For details, see "Configuring the Oracle Client for Mercury Business Availability Center" on page 103.

➤ **Create an Oracle user schema for management data.**

You can create a management user schema manually, or you can use the set management database utility to have Mercury Business Availability Center create a management user schema for you. For details on creating an Oracle user schema for management data, see "Management User Schema" on page 95.

➤ **Create one or more Oracle user schemas for profile data.**

You can create profile user schemas manually, or you can use the Database Management page, accessible from the Setup and Maintenance tab in Platform Administration, to create profile user schemas for you. For details on creating Oracle user schemas for Mercury Business Availability Center profile data, see "Profile User Schemas" on page 98.

Information is provided in this section for both recommended and supported Oracle environments. Mercury Business Availability Center recommendation indicates that Mercury quality assurance personnel have rigorously tested the recommended environment/option. A supported environment or option means that Mercury quality assurance personnel have successfully performed basic tests on the environment/option.

System Requirements

This section describes the system requirements for working with Oracle Server in conjunction with Mercury Business Availability Center.

Hardware Requirements

For Mercury Business Availability Center hardware sizing guidelines, see Chapter 13, "Oracle Server Sizing Guidelines."

For Oracle hardware requirements, refer to the installation guide for your specific Oracle platform. Additional information is also available in the Oracle software distribution media as well as the online Oracle documentation. For Oracle documentation, refer to: http://otn.oracle.com/documentation/index.html.

Software Requirements

Mercury Business Availability Center supports both UNIX and Windows database servers. The following table describes the subset of the Oracle Server certification matrix that is supported and certified for working with Mercury Business Availability Center.

Component	Supported		Recommended	
	Version/Edition	Service Pack	Version/Edition	Service Pack
Windows Operating System	Windows 2000 Server/Advanced Server	Service Pack 4	Windows 2003 Server standard / enterprise	Service Pack 1
Sun Solaris Operating System	Solaris 9		Solaris 8, 10	
Oracle	Oracle 9.2.0.6		Oracle 10.2.0.1	

Note:

- Solaris environments are 64-bit only.
- Oracle environments for Oracle 9i are 32-bit on all platforms and for Oracle 10g, 32-bit for Windows and 64-bit for UNIX.

For information on Oracle Client software requirements, see "Oracle Client Versions and Operating System Platforms" on page 104.

Oracle Instances

You can install more than one Oracle instance on a machine, using the same Oracle database engine.

For Mercury Business Availability Center certification, do not use more than one Oracle instance. If you do use more than one instance for the Mercury Business Availability Center databases, ensure that all the instances are configured as described in this document and that they all have the same characteristics (such as the same character set).

Choosing an Edition of Oracle Server

You can use one of the following Oracle 9i or 10g editions:

- **Standard Edition** – includes a fully integrated set of management tools, full distribution, replication, and Web features. From single-server environments for small businesses to highly distributed, branch environments, Oracle includes all the facilities necessary to build business-critical applications.
- **Enterprise Edition** – provides efficient, reliable, secure data management for high-end applications, such as high volume on-line transaction processing (OLTP) environments, query-intensive data warehouses, and demanding Internet applications. Oracle Enterprise Edition provides the tools and functionality to meet the availability requirements of today's mission-critical applications.

- **Personal Edition** – supports single user development and deployment that require full compatibility with Oracle Standard and Enterprise Editions.
- **Oracle Lite** – is a small footprint database built from the ground up to deliver enterprise applications to mobile devices, such as laptop computers, handheld computers, and information appliances. With strong management tools support and built in replication to Oracle Standard and Enterprise Editions, Oracle Lite facilitates the effective deployment of mobile solutions. This document does not cover the Oracle features because they are not directly comparable to those in the above editions. For a comparison between Oracle Lite and the above editions, refer to the related Oracle Lite articles.

  Note that there is a difference in cost between the various editions.

Note: Both the Standard and Enterprise editions are supported for Mercury Business Availability Center. However, only the Enterprise edition is recommended for use with Mercury Business Availability Center.

10

Setting Up the Mercury Business Availability Center Database Environment

This chapter describes how to create Oracle tablespaces for Mercury Business Availability Center user schemas and how to build management and profile user schemas. It also contains information on Oracle permissions for Mercury Business Availability Center user schemas.

This chapter describes:	On page:
Mercury Business Availability Center Oracle Tablespaces	92
Management User Schema	95
Profile User Schemas	98
Mercury Business Availability Center Oracle Schema Privileges	100

Note: It is strongly recommended that you run the database schema verify program after creating the management and/or profile schemas to verify that your database schemas are configured properly. For information on the verification process, see Appendix D, "Database Schema Verification."

Mercury Business Availability Center Oracle Tablespaces

An Oracle tablespace is an Oracle object that is a logical container of database objects, for example, tables, indexes, and so forth. When working with Mercury Business Availability Center, you must create one or more dedicated default tablespaces for your Mercury Business Availability Center user schemas. You may also want to create a dedicated temporary tablespace for Mercury Business Availability Center. To create a tablespace, you must provide specific operating system files that physically represent the tablespace, as well as extent parameters.

Note: For information on creating tablespaces when you have enabled the Purging Manager, see "Creating Oracle Tablespaces When Using the Purging Manager" on page 165.

When mapping operating system files, there is an option to make the file auto-extendable. This feature is supported by Mercury Business Availability Center, but not certified for use with Mercury Business Availability Center, since it can cause the system to consume all available disk space.

Locally Managed Tablespaces

A locally managed tablespace is a feature introduced in Oracle8i. Prior to Oracle8i, all tablespaces were dictionary-managed tablespaces. A tablespace that manages its extents locally can have either uniform extent sizes, or variable extent sizes that are determined automatically by the system. When you create the tablespace, the **uniform** or **autoallocate** (system-managed) option specifies the type of allocation.

For system-managed extents, Oracle determines the optimal size of extents, with a minimum extent size of 64 KB. This is the default extent size for permanent tablespaces.

For uniform extents, you can specify an extent size, or use the default size, which is 1 MB. Temporary tablespaces that manage their extents locally can only use this type of allocation.

Note that the NEXT, PCTINCREASE, MINEXTENTS, MAXEXTENTS, and DEFAULT STORAGE storage parameters are not valid for extents that are managed locally.

The following table describes the locally managed tablespace options that are supported and recommended for working with Mercury Business Availability Center:

Option	Supported	Recommended	Remarks
Locally managed temporary tablespace	Yes	Yes	Requires the use of TEMPFILE. Note that TEMPFILE is not always allocated on the disk upon creation.
Locally managed data tablespace	Yes	Yes	

For information on locally managing temporary tablespace using TEMPFILE, see Chapter 13, “Oracle Server Sizing Guidelines”.

Creating a Mercury Business Availability Center Oracle Tablespace

You create a Mercury Business Availability Center Oracle tablespace using the **oracle_tablespace_create.bat** (or **oracle_tablespace_create.sh** in UNIX installation) script.

Note: This script is a basic script for tablespace creation that contains one data file. You can edit this file according to the size of your Mercury Business Availability Center installation. For more information, see Chapter 13, “Oracle Server Sizing Guidelines.”

To create a Mercury Business Availability Center Oracle tablespace:

Run the following command from the directory in which the **oracle_tablespace_create.bat** or **oracle_tablespace_create.sh** script is located:

oracle_tablespace_create [admin_user] [admin_password] [tns_entry_name] [tablespace_name] [file_name] [file_size]

- [admin_user] – name of user with administrative permissions on Oracle Server
- [admin_password] – password of specified user
- [tns_entry_name] – the TNS name specified in the **tnsnames.ora** file on the local Oracle Client
- [tablespace_name] – name of tablespace to create
- [file_name] – file name to be created, including full path to file
- [file_size] – file size; use M for MB and K for KB
  - For a management user schema, specify a file size of 500 MB.
  - For a standalone CMDB user schema, specify a file size of 500 MB.
  - For a profile user schema, specify a file size of at least 500 MB. Note that the actual size needed will vary depending on the amount of data Mercury Business Availability Center generates and the amount of time you keep historical data.

Management User Schema

The management user schema contains the Mercury Business Availability Center configuration data and is usually a small database. The management user schema can be created manually (the recommended method – see below), or automatically using the set management database utility. For more information on the set management database utility, see "Setting Management Database Parameters for a Windows Platform" in *Deploying Servers*.

Note: If you create a CMDB manually, according the recommendations in this section, you must run the scripts that generate the CMDB user schema as you are unable to connect to an existing CMDB that does not contain the default table structures. For details on creating and connection to CMDB in Mercury Business Availability Center, see "Mercury Universal CMDB Management" in *Platform Administration*.

Note: To work with the set management database utility, you must first install and configure the Oracle Client (**tnsnames.ora** file) on all of the Mercury Business Availability Center servers. For more information, see "Configuring the Oracle Client for Mercury Business Availability Center" on page 103.

The set management database utility prompts you to supply the following parameters:

- **Administrator user name –** the name of a user with administrative permissions on Oracle Server that is used to create the user schema (by default, the Oracle administrative user is **system**).
- **Administrator password –** the specified user's password (by default, the Oracle administrative user's password is **manager**).

- **Management user name –** a name for the management user schema that you want to create. Note that you must specify a unique user name for each user schema that you create for Mercury Business Availability Center.
- **Management password –** a password enabling you to access the management user schema. You use this password, for example, when specifying management user schema settings in the set management database utility.
- **tnsnames entry name –** the TNS name specified in the **tnsnames.ora** file on the Oracle Client machine.
- **Default tablespace –** the name of the dedicated default tablespace you created for the management user schema. For more information, see "Mercury Business Availability Center Oracle Tablespaces" on page 92.
- **Temporary tablespace –** the name of the dedicated default temporary tablespace that you created for the management user schema. If you did not create a dedicated temporary tablespace, specify an alternative temporary tablespace (the default Oracle temporary tablespace is called **temp**).
- **Hostname –** the name of the host machine on which the Oracle server is installed.
- **SID –** an Oracle parameter that identifies the specific Oracle instance on the above host machine.

Note: If the connection parameters to a database server change, you must update the **tnsnames.ora** file on the Oracle Client installed on all three Mercury Business Availability Center servers. You must then restart the Mercury Business Availability Center service on all machines.

Creating a Management User Schema Manually

It is recommended that a certified database administrator create the management user schema manually (using your organization's database characteristic methodology). You then connect to the management user schema later in the set management database utility. For more information on the set management database utility, see "Setting Management Database Parameters for a Windows Platform" in *Deploying Servers*.

Note: Manually configuring a management user schema may be required if you cannot submit database administrator connection parameters over a non-secure connection.

You create the management user schema manually using the **management_ora_dbobjects_create.bat** (or **management_ora_dbobjects_create.sh** in UNIX installation) script, located in the <**Core Server root directory**>**AppServer****webapps****site.war****DataBases****ORA_DB_Utils** directory.

To create a management user schema manually (including the default CMDB):

Run the following command (see page 95 for an explanation of the parameters to be used) from the <**Core Server root directory**>**AppServer****webapps****site.war****DataBases****ORA_DB_Utils** directory:

management_ora_dbobjects_create [admin_user] [admin_password] [application_management_user] [application_management_password] [tns_entry_name] [default_tablespace] [temporary_tablespace]

To create a CMDB user schema manually:

From the tablespace in which you want to create the objects, run the following scripts that are located in the <**Core Server root directory**>**AppServer****webapps****site.war****DataBases****ORA_DB_Utils** directory:

- common_ora_dbobjects_create.sql
- create_cm_tables_cmdb.sql

Profile User Schemas

Profile user schemas contain performance data collected by Mercury Business Availability Center, and are generally large databases. For information on the size of the profile databases, see Chapter 13, "Oracle Server Sizing Guidelines".

Profile user schemas can be created manually (the recommended method – see below), or automatically using the Database Management page in Platform Administration's Setup and Maintenance tab. For information on creating profile databases automatically, see "Database Management" in *Platform Administration*.

When creating profile user schemas, you must supply the following parameters:

- **Administrator user name –** the name of a user with administrative permissions on Oracle Server that is used to create the user schema (by default, the Oracle administrative user is **system**).
- **Administrator password –** the specified user's password (by default, the Oracle administrative user's password is **manager**).
- **Profile user name –** a name for the profile user schema that you want to create. Note that you must specify a unique user name for each user schema that you create for Mercury Business Availability Center.
- **Profile user password –** a password enabling you to access the profile user schema. You use this password when connecting to a predefined profile user schema from the Database Management page in Platform Administration's Setup and Maintenance tab.
- **tnsnames entry name –** the TNS name specified in the **tnsnames.ora** file on the Oracle Client machine.
- **Default tablespace –** the name of the dedicated default tablespace you created for the profile user schema. For more information, see "Mercury Business Availability Center Oracle Tablespaces" on page 92.
- **Temporary tablespace –** the name of the dedicated default temporary tablespace that you created for the profile user schema. If you did not create a dedicated temporary tablespace, specify an alternative temporary tablespace (the default Oracle temporary tablespace is called **temp**).

Note: If the connection parameters to a database server change, you must update the **tnsnames.ora** file on the Oracle Client installed on all three Mercury Business Availability Center servers. You must then restart the Mercury Business Availability Center service on all machines.

Creating Profile User Schemas Manually

It is recommended that a certified database administrator create profile user schemas manually (using your organization's database characteristic methodology). You then connect to these profile user schemas later from the Database Management page in Platform Administration's Setup and Maintenance tab. For more information on connecting to profile user schemas from Platform Administration, see "Database Management" in *Platform Administration*.

Note: Manually configuring profile user schemas may be required if you cannot submit database administrator connection parameters over a non-secure connection.

You create the profile user schema manually using the **profile_ora_dbobjects_create.bat** (or **profile _ora_dbobjects_create.sh** in UNIX installation) script, located in the <**Core Server root directory**>**AppServer\webapps\site.war\DataBases\ORA_DB_Utils** directory.

To create a profile user schema manually:

Run the following command (see page 98 for an explanation of the parameters to be used) from the <**Core Server root directory**>**AppServer\webapps\site.war\DataBases\ORA_DB_Utils** directory:

```
profile_ora_dbobjects_create [admin_user] [admin_password]
[application_management_user] [application_management_password]
[tns_entry_name] [default_tablespace] [temporary_tablespace]
```

Mercury Business Availability Center Oracle Schema Privileges

When using an Oracle user schema, you can perform any database action that is allowed by the Oracle permission granted to the user. The following list (taken from the **oracle_user_create.sql** script located in the **<Core Server root directory>\AppServer\webapps\site.war\DataBases\ORA_DB_Utils** directory) describes the required database permissions that must be granted to the Mercury Business Availability Center user. These permissions are also used by the Mercury Business Availability Center platform components to create a new Oracle user:

- GRANT "CONNECT" TO <Mercury Business Availability Center Oracle user schema>;
- GRANT CREATE ANY INDEX TO <Mercury Business Availability Center Oracle user schema>;
- GRANT CREATE ANY SEQUENCE TO <Mercury Business Availability Center Oracle user schema>;
- GRANT CREATE ANY TABLE TO <Mercury Business Availability Center Oracle user schema>;
- GRANT CREATE ANY TRIGGER TO <Mercury Business Availability Center Oracle user schema>;
- GRANT UNLIMITED TABLESPACE TO <Mercury Business Availability Center Oracle user schema>;
- GRANT CREATE ANY VIEW TO <Mercury Business Availability Center Oracle user schema>;
- GRANT CREATE ANY PROCEDURE TO <Mercury Business Availability Center Oracle user schema>;
- ALTER USER <Mercury Business Availability Center Oracle user schema> DEFAULT ROLE ALL;

Note: Mercury Business Availability Center supports any user with higher permissions. For Mercury Business Availability Center certification, use an Oracle user that has the exact Oracle permissions described above.

11

Configuring the Oracle Client for Mercury Business Availability Center

This chapter describes how to configure the Oracle Client for Mercury Business Availability Center. Note that in order to work with Mercury Business Availability Center, you must install and configure the Oracle Client software on all three Mercury Business Availability Center servers.

This chapter describes:	On page:
Oracle Client Versions and Operating System Platforms	104
Oracle Client Installation	104
Oracle Client Configuration	105

Oracle Client Versions and Operating System Platforms

The following table describes the Oracle Client versions and operating system platforms that are supported and recommended for working with Mercury Business Availability Center.

Component	Supported		Recommended	
	Version/Edition	**Service Pack**	**Version/Edition**	**Service Pack**
Windows Operating System	Windows 2000 Server/Advanced Server	Service Pack 4	Windows 2003 Server standard / enterprise	Service Pack 1
Sun Solaris Operating System	Solaris 9		Solaris 8, 10	
Oracle	Oracle 9.2.0.6		Oracle 10.2.0.1	

Oracle Client Installation

To install the Oracle Client, refer to the Oracle documentation.

If you choose the custom installation option during the installation process, ensure that you install the following components (under **Oracle Client**):

- Oracle Net (Including TCP/IP Adaptor)
- Oracle Database Utilities
- SQL*Plus
- Oracle Call Interface (OCI)

Oracle Client Configuration

In order to work with Mercury Business Availability Center, you must configure the **tnsnames.ora** file that is located in the **<ORACLE_HOME>\network\admin** directory. Make sure to specify the name or IP of the Oracle Server host machine, the Oracle Server listener port (by default, usually 1521), and the SID (by default, ORCL) or service_name. The following is an example of a **tnsnames.ora** file.

```
# TNSNAMES.ORA Network Configuration File: D:\oracle\ora81\network\admin\tnsnames.ora
# Generated by Oracle configuration tools.

LONDON.MERCURY.CO.IL =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = london)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SID = london)
    )
  )
```

It is recommended that you use the Oracle Net Configuration Assistant Oracle tool built to configure the **tnsnames.ora** file. For more information, refer to the Oracle documentation.

Ensure that the Oracle Client configuration, such as SID and port settings, matches the Oracle Server configuration. To test the connection from the Oracle Client machine to the Oracle Server machine, use the tnsping utility.

Note: The Mercury Business Availability Center servers access the Oracle Server using the JDBC thin driver. The JDBC thin driver does not support a firewall connection that is net*8/9 compliant, and therefore allows for SQL data transmission only.

MDAC for Oracle Client

Mercury Business Availability Center uses Microsoft MDAC components to connect to the database. MDAC is installed by default on a Windows 2000 operating system.

Note: MDAC versions 2.5, 2.52, 2.61, 2.62, and 2.7 SP1 Refresh are all supported by Mercury Business Availability Center. For Mercury Business Availability Center certification, ensure that MDAC version 2.7 SP1 Refresh is installed on the Oracle Client machine. Note that this version of MDAC is installed automatically with Mercury Business Availability Center server installation.

12

Maintaining an Oracle Server Database

This chapter describes the various maintenance and tuning procedures that are recommended for Mercury Business Availability Center databases created on Oracle Servers, as well as the available database backup and recovery methods.

This chapter describes:	On page:
Database Maintenance and Tuning	107
Oracle Database Backup and Recovery	120

Database Maintenance and Tuning

Poor database performance can be caused by the faulty configuration of the instance and database, or by abnormal resource consumption of an Oracle transaction, user, or process. It is essential for the database administrator to proactively monitor resource consumption, and correct any abnormalities before performance is affected.

Note: Memory, CPU, and I/O are the three most common system resources consumed by Oracle.

There are a number of third-party tools that you can use to monitor database behavior and assist you in identifying bottlenecks in your system. Use the following guidelines to help you.

System Global Area (SGA)

Always configure your SGA to fit physical memory and avoid using swapping. It is recommended that you not set the SGA for more than 70 percent of system physical memory, leaving enough memory for additional system and client processes.

Database Load Behavior

Run **ustlbstat/utlestat** (or STATSPACK) regularly to monitor the database behavior. In Oracle10g, an AWR snapshot is created regularly (every hour by default) and reserved for 7 days. An AWR report can be generated and used in the same way as a STATSPACK report. For additional information on running and interpreting the output you receive, refer to *Oracle Metalink Note 62161.1: BSTAT/ESTAT* (http://metalink.oracle.com/metalink/plsql/showdoc?db=Not&id=62161.1) or *Oracle Metalink Note 94224.1: STATSPACK FAQ* (http://metalink.oracle.com/metalink/plsql/showdoc?db=Not&id=94224.1).

It is also recommended that you monitor I/O load on the system to identify I/O contention. Once you determine which disk is most loaded, you can use the utlbstat/utlestat output to determine which particular Oracle data file is the cause of the contention and consider the relocation of data files.

CPU and I/O

It is recommended that you monitor the CPU and file system, which are the main resources consumed by the database server. CPU usage should not exceed 70 percent and the I/O wait should not be higher than 10 percent.

You can use **perfmon** on Windows, or **top** in UNIX, to monitor the above resources.

Oracle Alert File

Oracle registers abnormal events in the **alert.log** file, whose location is defined by the BACKGROUND_DUMP_DEST parameter.

It is recommended that you check this file regularly to identify abnormalities that should be corrected, for example, ORA-XXXXX errors.

Archive Log – File System

When using the archivelog mode, monitor your ARCHIVE_DUMP_DEST location for disk usage. These files should be backed up and deleted regularly to leave sufficient disk space for new archive files.

The archive file is usually the same size as the redo log file. To determine the size of a redo log file, use the operating system command or the following query:

```
SQL> select GROUP#, BYTES
      from V$LOG;
```

To determine the number of archive files generated over a period of time, for example, a day, you can use the following query after the system is stable:

```
SQL> alter session set NLS_DATE_FORMAT = 'DD-MON-YYYY';
SQL> select TO_DATE(TO_CHAR(FIRST_TIME,'DD-MON-YYYY')) as “Day”,
      COUNT(*) as "Number of files"
      from V$LOG_HISTORY
      group by TO_CHAR(FIRST_TIME,'DD-MON-YYYY')
      order by 1 asc;
```

Tablespace Storage Space

To avoid space errors caused by data growth, monitor your tablespace usage regularly.

If you run out of space in one of your tablespaces, you can add one or more data files to it by using the ALTER TABLESPACE <tablespace name> ADD DATAFILE… command.

Dictionary-Managed Tablespace Coalescing

Free space in Oracle tablespaces is composed of newly created extents, or extents that were used and freed. If some of the free space in a tablespace is composed of extents that were used and freed, your tablespace may become temporarily fragmented. To repair fragmentation, two extents that reside next to one another can be coalesced in order to create one large extent.

To check for fragmentation, run the following query using SQL*Plus (using the system administrator account):

```
SELECT A.TABLESPACE_NAME, COUNT(*) BLOCK_CASES
      FROM DBA_FREE_SPACE A, DBA_FREE_SPACE B
      WHERE A.TABLESPACE_NAME = B.TABLESPACE_NAME
      AND A.FILE_ID = B.FILE_ID
      AND A.BLOCK_ID+A.BLOCKS = B.BLOCK_ID
GROUP BY A.TABLESPACE_NAME
/
```

This query returns a list of tablespaces that require coalescing. Although the Oracle SMON process automatically performs coalescing, it may do so infrequently. It is therefore recommended that you coalesce tablespace extents using the ALTER TABLESPACE <tablespace name> COALESCE; command.

Collecting Statistics for Databases

The Mercury Business Availability Center platform is planned and built to work with the Oracle Cost Base Optimizer. For the Optimizer to work properly, you must periodically collect statistics for all schema tables.

During the initial phase of Mercury Business Availability Center deployment, it is recommended that you collect statistics for all Mercury Business Availability Center objects (tables and indexes), using the following procedure:

1. Log in to the Mercury Business Availability Center profile schema using SQL*Plus.
2. Run the following command:

   ```
   Exec DBMS_STATS.GATHER_SCHEMA_STATS (ownname  => '<name of Oracle schema>', estimate_percent => 20, cascade => TRUE);
   ```

   Once your Mercury Business Availability Center system is stable, statistics should be collected once a day.

Oracle 10g has an automated job for statistics collection of all database schemas as part of using 10g Scheduler APIs. The automated job is the **GATHER_STATS_JOB** which is owned by the **SYS** super user. The job collects stale (inaccurate) statistics at a predefined time (Maintenance Window). The job only refreshes statistics for objects with empty or stale statistics, thereby avoiding scanning unnecessary data as was the case in Oracle 9i.

The Maintenance Window comprises the WEEKNIGHT_WINDOW (with the job starting at 10:00 PM Monday-Friday), and the WEEKEND_WINDOW (with the job starting at 12:00 AM on Saturday), with no job scheduled for Sunday. If you need to collect statistics at other maintenance times that better suit your system, your database administrator can change the schedule using the Oracle Enterprise Manager console. For an overview of the Oracle Scheduler, refer to the chapter "Overview of Scheduler Concepts" in the Oracle Database Administrator's Guide in the Oracle 10g documentation set.

When working with large Mercury Business Availability Center environments, it is recommended that you collect statistics only for objects for which the amount of data changes significantly during the day, or for new objects that are created (such as new tables and indexes created by the Purging Manager).

To collect statistics for specific schema tables and their indexes:

1. Log in to the schema using SQL*Plus.
2. For each table, collect statistics by running the following command:

```
Exec DBMS_STATS.GATHER_TABLE_STATS (ownname => '<name of Oracle schema>', tabname => '<Name of table for which you want to collect statistics>', estimate_percent => 5, cascade => TRUE);
```

Note: Cascade => True instructs the Oracle database to analyze all the indexes in the table.

Alternatively, you can collect statistics for each table and index using the following procedure:

1. Log in to the profile schema using SQL*Plus.
2. For each table, collect statistics using the ANALYZE TABLE <table_name> ESTIMATE STATISTICS SAMPLE <x> ROWS; query. It is recommended that "x" be approximately five percent of the records in the table.
3. For each index, collect statistics using the ANALYZE INDEX <index name> COMPUTE STATISTICS; query.

Note: Collecting statistics is a resource-consuming operation that can take a long time. It is therefore recommended that you collect statistics during special maintenance hours.

To obtain the list of tables, use the SELECT TABLE_NAME FROM USER_TABLES query.

To obtain the list of schema indexes, use the SELECT INDEX_NAME FROM USER_INDEXES query.

Collecting Statistics for CMDB

Unlike some databases, where queries are predefined and can be tuned according to the expected database size, CMDB database constructs queries dynamically, according to Pattern Views defined against its data model. This necessitates accurate statistics at all times. In addition to running a daily job to update statistics for CMDB, it is recommended that you manually refresh statistics if major changes to the CMDB schema objects have occurred, usually caused by bulk insert transactions. The following scenarios warrant a manual refresh of CMDB statistics:

- **Automated Discovery tasks** – Discovery Manager is the process responsible for automatically detecting configuration items (CIs) and inserting them into CMDB.
- **Mercury Business Availability Center adapters synchronization** – when expected to load substantial data to CMDB. You synchronize adapters from the Source Manager tab, accessed from Admin > CMDB.

Statistics Collection Guidelines for Oracle 9i

The following guidelines are applicable to Oracle 9i.

Running a Daily Job

To create a daily job to run statistics on CMDB, use the **create_statistics_job.bat** script located in the **<Data Processing Server machine root directory>\MercuryAM\CMDB\dbscripts\oracle** directory. A thick database client tool (SQLPLUS) is required to run the script. If you need to move the script to a different machine that has SQLPLUS installed, copy the directory in which the script is located to the required machine. The script is run with the following command:

create_statistics_job.bat <schema> <password> <db alias> <hour>

where:

- **schema** – is the name of the CMDB schema user of the schema for which you are installing the statistics job.
- **password –** is the password for the database schema user.
- **db alias** – is the db alias for connecting to the target database as specified in the **tnsnames.ora** file. Ensure that there is an entry in the tnsnames.ora file for the target server.
- **hour** – is the hour of the day to run the statistics job. Accepted values are 0-23 with a default value of midnight.

Note: If you don't want to use the provided script to generate an automatic, daily job you can execute the following, manual command as needed:

Begin DBMS_STATS.GATHER_SCHEMA_STATS (ownname => '<name of Oracle schema>', cascade => TRUE) ; end;

This API gathers statistics on all tables and their related indexes, even if statistics are the same as the previous call.

The job should be scheduled for a maintenance hour when the system is not heavily loaded (for example, every night at midnight).

Running a Manual Statistics Refresh

When major data changes have been carried out on CMDB, you refresh the statistics by one of the following methods:

- Use the **runStatistics** JMX:
  - In a Web browser, open **http://<Centers Server machine name>:8080/jmx-console**.
  - In the **Topaz** section, select **CMDB Dal Services**.
  - Under **runStatistics**, enter the customer ID. The default customer ID for an individual Mercury Business Availability Center system (that is, one not managed by Mercury Managed Services) is 1.
  - Under **runStatistics**, click **Invoke**. The CMDB statistics are regenerated.
- Manually run the daily job using an Oracle client:
  - Connect to the CMDB schema via SQLPLUS.
  - Run the query **Select job from user_jobs where upper(what) like '%GATHER_SCHEMA_STATS%';**

    The query output is the job number.
  - Run the job with the call **Exec dbms_job.run (<job number>) ;**
- Connect to the CMDB schema and execute the PL/SQL block **begin DBMS_STATS.GATHER_SCHEMA_STATS (ownname => '<name of Oracle schema>' , cascade => TRUE) ; end;**

Statistics Collection Guidelines for Oracle 10g

The following guidelines are applicable to Oracle 10g.

Running a Daily Job

From Oracle 10g and later, there is no need to define a dedicated, daily job against the CMDB schema as Oracle 10g includes an automated job for statistics collection. For details, see "Collecting Statistics for Databases" on page 110.

Running an Hourly Job

In Oracle 10g, object changes are automatically monitored and it is possible to operate the **DBMS_STATS.GATHER_SCHEMA_STATS** API with the **GATHER AUTO** method collecting only missing or stale statistics (that is, when statistics no longer represent the object accurately, usually when the object's data change by 10% or more).

If you do not manually refresh statistics for each major data change, it is recommended to schedule a Scheduler job running every hour for refreshing stale statistics against the CMDB schema. You execute an automatic, hourly refresh of statistics by executing the PL/SQL block **begin DBMS_STATS.GATHER_SCHEMA_STATS (ownname => '<name of Oracle schema>', options => 'GATHER AUTO') ; end;**

Note: From Oracle 10g and on, it is strongly recommended to use Oracle Scheduler job APIs instead of DBMS_JOB APIs for job automation.

Running a Manual Statistics Refresh

When major data changes have been carried out on CMDB, you refresh the statistics by one of the following methods:

- Use the **runStatistics** JMX:
  - In a Web browser, open **http://<Centers Server machine name>:8080/jmx-console**.
  - In the **Topaz** section, select **CMDB Dal Services**.
  - Under **runStatistics**, enter the customer ID. The default customer ID for an individual Mercury Business Availability Center system (that is, one not managed by Mercury Managed Services) is 1.
  - Under **runStatistics**, click **Invoke**. The CMDB statistics are regenerated.

Note: The JMX utility checks the database release and in the case of an Oracle 10g database, runs the statistics in the **GATHER AUTO** method.

➤ Connect to the CMDB schema and execute the PL/SQL block **begin DBMS_STATS.GATHER_SCHEMA_STATS (ownname => '<name of Oracle schema>' , options => GATHER AUTO) ; end;**

FREELISTS Storage Parameter

In a distributed Mercury Business Availability Center environment, it is recommended that you enlarge the FREELISTS storage parameter value in the Mercury Business Availability Center database tables and their indexes from the default value of 1 to a value of 20. Enlarging the FREELISTS storage parameter value prevents data block waits.

Use SQL*Plus to change the FREELISTS parameter to a value of 20.

To change the FREELISTS parameter in a table:

Use the following command:

```
alter table <table name> storage ( freelists 20 );
```

To change the FREELISTS parameter in an index:

Use the following command:

```
alter index <index name> storage ( freelists 20 );
```

When creating large tables in the profile schema, Mercury Business Availability Center automatically sets the FREELISTS parameter to 20. Database objects (tables and indexes) created later on by other components, such as the Purging Manager, are created with the default Oracle FREELISTS value of 1. It is recommended that you track these objects and set their FREELISTS parameter values to 20.

Note: The FREELISTS parameter applies only to objects created in tablespaces whose segment space is manually managed.

CMDB Index Fragmentation

The CMDB schema consists of Oracle B-tree indexes for enhancing searches on table columns.

It is recommended to validate the structure of the CMDB schema indexes periodically (at least every week for active systems), and if necessary to rebuild the indexes found to be fragmented.

The main reasons for indexes becoming fragmented are:

- **Row deletes** – when rows in a table are deleted, Oracle index nodes are not physically deleted, nor are the entries removed from the index. Rather, Oracle logically deletes the index entry and leaves dead nodes in the index tree, where they may be reused if another adjacent entry is required. However, when large numbers of adjacent rows are deleted, it is highly unlikely that Oracle will have an opportunity to reuse the deleted leaf rows. In addition to wasting space, large volumes of deleted leaf nodes cause index scans to take more time.

  Over time, following row deletes from schema tables (both explicitly by users, and implicitly by the CMDB engine), there may be a need to rebuild some of the CMDB schema indexes.

- **Index height** – the height of an index refers to the maximum number of levels encountered within the index. As the number of levels in an index increases, more block reads are needed when searching the index. When a large amount of rows are added to a table, Oracle may create additional levels of an index to accommodate the new rows, thereby causing an index to reach four levels, although only in those areas of the index tree where massive inserts have occurred. While Oracle indexes can support many millions of entries in three levels, any Oracle index that has four or more levels can benefit from rebuilding.

  For CMDB tables, it is recommended to rebuild any index that has more than three levels.

Index Maintenance Utility

Mercury Business Availability Center's index maintenance utility (**maintain_indexes.bat**) can be used to identify and rebuild indexes that have more than three levels, or that have 100,000 values or more with 10% of deleted values.

You can set a flag when running the utility to instruct it to rebuild indexes identified as being fragmented automatically, although it is recommended that you rebuild indexes manually.

When run, the utility produces a log file (**index_stats.log**) that contains the following entries:

- An alphabetical list of indexes that were identified as candidates for rebuilding. For each index listed, statistics are shown such as the height of the index and the percentage of deleted rows.
- Rebuild commands for each index listed that can be used to rebuild the indexes manually.

The utility also creates a table called **TEMP_STATS** in the target schema that contains all the indexes and their related statistics (not only the indexes listed as candidates for rebuilding). The table remains in the schema until it is manually dropped, to enable inspection of the results at a later stage.

Warning: The index maintenance utility is resource intensive, as it analyzes all indexes in the schema. It can also cause locks on database objects, or skip indexes that are locked by other sessions. It is recommended to run the index maintenance utility during maintenance hours only.

To run the index maintenance utility:

1. Copy the following files from the **\MercuryAM\CMDB\dbscripts\oracle\utils** directory on the Mercury Business Availability Center Modeling Data Processing Server to a Windows machine that has Oracle database client installed:
   - **maintain_indexes.bat**
   - **maintain_indexes.sql**

2 On the machine to which you copied the files, open a DOS command window and move to the location in which you copied the files.

3 Run the index maintenance utility with the following command:

maintain_indexes.bat <schema> <password><db alias> (rebuild flag)

where:

- **schema** – is the name of the database schema user of the schema for which you are running the utility.
- **password** – is the password for the database schema user.
- **db alias** –is the db alias for connecting to the target database as specified in the **tnsnames.ora** file. Ensure that there is an entry in the tnsnames.ora file for the target server.
- **rebuild flag** –is the flag to instruct the utility to rebuild indexes automatically. Set the flag to **0** if you don't want the utility to rebuild indexes automatically and to **1** if you do. The default setting is **0**.

When the index maintenance utility has finished running, check the **index_stats.log** file in the directory into which you copied the files in step 1, for the list of indexes that are candidates for rebuilding and the rebuild commands to be used.

Note: If CMDB is part of the management database (which is the default), run the index maintenance utility for the management database schema. If you have created a separate CMDB, run the index maintenance utility for the CMDB schema.

Note: The execution time of the index maintenance utility depends on the size of the indexes and the load on the system when being run.

Oracle Database Backup and Recovery

Your backup strategy is tested when a failure occurs and data is lost. You can lose or corrupt data in several ways, such as a logical application error, an instance failure that prevents Oracle from starting, or a media failure caused by a disk crash. In addition to your scheduled backups, it is important to perform a backup when the database structure changes (for example, when a data file is added to the database), or before you upgrade your software or hardware.

When choosing a backup strategy, consider several factors, such as the system workload, the usage schedule, the importance of the data, and the hardware environment of the database.

Oracle backups can be performed using scripts executing SQL commands combined with operating system commands to copy files, or using Oracle RMAN (Recovery Manager) commands.

It is recommended that you maintain updated records of backups performed on your database so that you can use them for recovery on demand. If you are using RMAN, catalog information is available from the catalog.

Available Backup Methods

This section describes the various backup methods that are available.

Cold Backup

Cold backup, also known as offline backup, is a database level backup. It normally requires that the database be shut down before the backup is started. The amount of downtime is dependent on the database size, the backup media (disk or tape), the backup software, and the hardware in use.

Once the instance is down, all its data files, log files, control files, and configuration files should be copied either to disk or other media. After the copy is complete, the instance can be restarted.

This backup method enables recovery to a the point in time in the past at which the database snapshot was taken.

For more information, refer to the *Oracle Backup and Recovery Guide* (http://otn.oracle.com/pls/db92/db92.show_toc?partno=a96519&remark=drilldown&word=Backup).

Hot Backup

Hot backup, also known as online backup, enables you to run a backup while the instance is running and users are connected to the database. This backup method is a tablespace backup level and requires the database to operate in archivelog mode, which enables Oracle to track changes over time by generating redo log file copies called archive files. The generated archive files are written to the archive destination specified by the LOG_ARCHIVE_DEST (or LOG_ARCHIVE_DEST_NN) parameter in the instance parameter files. Other related archiving parameters are LOG_ARCHIVE_FORMAT and LOG_ARCHIVE_START.

Once you start the backup, all the data files, control files, archive files, and configuration files should be copied either to disk or other media. This method enables recovery to any point in time. Note that working in archivelog mode requires additional disk space to contain incremental archive files, which can influence database performance. During the backup process, Mercury Business Availability Center may also experience some performance degradation due to disk load.

For more information, refer to the *Oracle Backup and Recovery Guide* (http://otn.oracle.com/pls/db92/db92.show_toc?partno=a96519&remark=drilldown&word=Backup).

Export

In addition to the cold and hot physical backup methods, you can use the logical backup method known as export.

The export utility dumps schema structure and contents into an Oracle structured file. This method can be used to transfer data between two schemas in the same database, or between two separate Oracle databases. To load exported data back into the database, use the import utility.

For more information, refer to the Export/Import section of *Oracle Utilities* (http://otn.oracle.com/pls/db92/db92.show_toc?partno=a96652&remark=drilldown&word=Export).

In Oracle 10g, the Oracle Data Pump utility can be used for exporting data. For more information, refer to the Oracle Data Pump page in the Oracle Utilities section of the Oracle Web site (http://download-east.oracle.com/docs/cd/B19306_01/server.102/b14215/part_dp.htm#i436481).

Note: Mercury Business Availability Center does not require you to use a specific backup method; however, it is recommended that your backup method accommodate Mercury Business Availability Center's use of more than one database user schema (management and profile).

Oracle Recovery Manager – RMAN

Recovery Manager (RMAN) is a generic Oracle tool that enables you to back up and restore your target database. When working with RMAN, you can choose to work with the RMAN catalog schema. The catalog is managed within the Oracle schema and stores information on the registered database structure and backups performed using RMAN. It can be queried to produce backup reports and copy availability. A single catalog can manage backup information from one or more target databases.

The RMAN catalog is usually placed on a different database instance than the operational database and has a backup strategy of its own. It need only be available during the backup or recovery process.

The RMAN tool can be used in conjunction with third-party backup software for a complete backup and recovery solution.

The following are some advantages of RMAN:

- Minimizes backed up data by compressing backed up files to exclude empty data blocks, thereby saving time and space.
- Supports incremental backups.
- Supplies the user with backup status reporting ability.
- Supports parallel backup and recovery processes when possible.
- Can be used with a third-party backup media tool.

For more information on RMAN, refer to the Oracle Recovery Manager User's Guide (http://otn.oracle.com/pls/db92/db92.show_toc?partno=a96566&remark=drilldown&word=RMAN) and Oracle Recovery Manager Reference (http://otn.oracle.com/pls/db92/db92.show_toc?partno=a96565&remark=drilldown&word=RMAN) for Oracle 9i; and Backup and Recovery Advanced User's Guide (http://download-east.oracle.com/docs/cd/B19306_01/backup.102/b14191/toc.htm) and Backup and Recovery Reference (http://download-east.oracle.com/docs/cd/B19306_01/backup.102/b14194/toc.htm) for Oracle 10g.

13

Oracle Server Sizing Guidelines

This chapter contains guidelines for the Oracle database configuration settings that should be used when working with Oracle Server and Mercury Business Availability Center. Note that the recommended settings differ according to the size of your Mercury Business Availability Center deployment.

This chapter describes:	On page:
Mercury Business Availability Center Sizing	125
Hardware Sizing	127
Oracle Parameter Sizing	127
Oracle File Sizing	132

Mercury Business Availability Center Sizing

Mercury Business Availability Center database configuration requirements are dependent on the amount of data generated by Mercury Business Availability Center. For example, you would use a small database for data generated over a period of 300 days by 3 Business Process Monitors, each running 5 Business Process profiles executing 10 transactions and WebTrace every 15 minutes, with 5 percent of the transactions returning errors.

You would use a large database for data generated over a period of 250 days by 10 Business Process Monitors, each running 50 SiteScope profiles executing approximately 600 measurements, and 30 Business Process profiles executing 10 transactions and WebTrace every 15 minutes, with 5 percent of the transactions returning errors.

The following table describes the database size required for each database element, based on the amount of data generated:

Database Element	Mercury Business Availability Center Deployment		Remarks
	Small	**Large**	
Insert ratio (rows per hour)	6 KB or less	500 KB or more	
Named users (profile user schema)	fewer than 20	80 or more	Used for memory settings
Named users (management user schema)	fewer than 30	50 or more	Used for process settings
Database size	approximately 30 GB	approximately 600 GB and greater	

Hardware Sizing

The following table describes the hardware (CPU and memory) requirements recommended for the Mercury Business Availability Center Oracle database server:

Mercury Business Availability Center Deployment	Number of Processors	Physical Memory	Minimum Recommended Oracle Version
Small	1	1 GB	9.2.0.6
Large	8	8 GB or more	9.2.0.6

Oracle Parameter Sizing

The following table describes the recommended size for a number of parameters, when working with the Mercury Business Availability Center database server:

Parameter Name	Mercury Business Availability Center Deployment		Remarks
	Small	Large	
DB_BLOCK_SIZE	8192	16,384	See remarks below the table
DB_BLOCK_BUFFERS or DB_CACHE_SIZE	85,000 or 696,320,000	244,000 or 4 GB	See remarks below the table. Note that it is recommended that you use the DB_CACHE_SIZE parameter. For Oracle 10g, it is recommended that you use the SGA_TARGET parameter.

Parameter Name	Mercury Business Availability Center Deployment		Remarks
	Small	**Large**	
DB_CACHE_ADVICE	ON	ON	For gathering statistics when tuning is required
SHARED_POOL_SIZE	**Oracle 9i**: 80 MB **Oracle 10g**: 200 MB	**Oracle 9i**: 112 MB **Oracle 10g**: 252 MB	For Oracle 10g, it is recommended that you use the SGA_TARGET parameter
SGA_TARGET	890 MB	4.25 GB	Available in Oracle 10g only
LOG_BUFFER	512,000 bytes	2,000,000 bytes	
DB_FILE_MULTIBLOCK_READ_COUNT	16	32	
PROCESSES	200	400	Add an additional 100 as a safety net
SESSIONS	225	445	(1.1 * PROCESSES) + 5

Parameter Name	Mercury Business Availability Center Deployment		Remarks
	Small	**Large**	
SORT_AREA_SIZE	1 MB	2 MB	It is recommended that you use the PGA_AGGREGATE_TARGET parameter instead. This parameter is reserved for backward compatibility and shared server mode. In order for it to work, the WORKAREA_SIZE_POLICY parameter must be set to MANUAL.
SORT_AREA_RETAINED_SIZE	Equal to SORT_AREA_SIZE value	Equal to SORT_AREA_SIZE value	See remarks for the SORT_AREA_SIZE parameter.
HASH_AREA_SIZE	3 MB	6 MB	Is equal to 3 times the SORT_AREA_SIZE value. See remarks for the SORT_AREA_SIZE parameter.

Parameter Name	Mercury Business Availability Center Deployment		Remarks
	Small	Large	
WORKAREA_SIZE_ POLICY	AUTO	AUTO	The automatic PGA memory management mode applies only to work areas allocated by dedicated Oracle Servers. The size of work areas allocated by shared Oracle Servers is still controlled by the previous *_AREA_SIZE parameters because these work areas are allocated mainly in SGA and not in PGA.
PGA_AGGREGATE_ TARGET	400 MB	800 MB	
STATISTICS_LEVEL	TYPICAL	TYPICAL	Enables tuning if required.

Note the following:

- **DB_BLOCK_SIZE –** If you are using different database servers for the management and profile databases and want to use a smaller database block size for the management user schema, ensure that the database block size is not less than 4 KB.

  In addition, the DB_BLOCK_SIZE must be a multiple of the operating system block size.

- **DB_BLOCK_BUFFERS** or **DB_CACHE_SIZE** – The above recommendations are for very large implementations. Regardless of Mercury Business Availability Center certification, you must have the following additional hardware requirements in order to configure this parameter:
  - **Solaris hardware –** In 32-bit operating systems, there is a shared memory limit of 1.75 GB for all instances. If you want to use 4 GB of shared memory, you must use a 64-bit operating system and 64-bit Oracle RDBMS.
  - **Windows 2000 –** In order to support and use SGA that is larger than 2 GB, you must enable a specific Windows and Oracle configuration (VLM). For more information, refer to the Microsoft and Oracle online documentation.
- **SGA_TARGET** – Setting this parameter configures Oracle to automatically determine the size of the buffer cache (db_cache_size), shared pool (shared_pool_size), large pool (large_pool_size), java pool (java_pool_size), and streams pool (streams_pool_size).

  The value configured for SGA_TARGET sets the total size of the SGA components.

  When SGA_TARGET is set (that is, it's value is not 0), and one of the above pools is also set to a non-zero value, the pool value is used as the minimum value for that pool.

Oracle File Sizing

This section describes the file sizing guidelines for tablespaces, temporary tablespaces, data tablespaces, redo logs, and rollbacks.

Tablespace Settings

The following table specifies the recommended sizes for tablespaces:

Tablespace	Mercury Business Availability Center Deployment		Remarks
	Small	Large	
Management	500 MB	1 GB	The specified size is a minimum requirement.
Profile	30 GB	600 GB	The specified size is a minimum requirement. If you create more than one profile database, it is recommended that you create each profile database in a separate tablespace.

Note: There are additional size settings for temporary and rollback tablespaces. For more information, see "Temporary Tablespace Settings" on page 133 and "Undo Segment Settings" on page 135.

Temporary Tablespace Settings

The following table specifies the recommended sizes for temporary tablespaces:

Tablespace	Mercury Business Availability Center Deployment		Remarks
	Small	Large	
TEMP	1 GB	5 GB	Use multiple files with large tablespaces.
TEMP storage settings	Uniform allocation: 2 MB	Uniform allocation: 2 MB	➤ Should preferably be locally managed (Uniform allocation, automatic segment space management). ➤ Tablespace should be of a temporary type (use of TEMPFILE).

Note: If the WORKAREA_SIZE_POLICY parameter is set to MANUAL, the initial and next extent temporary tablespace storage space should be a multiple of the SORT_AREA_SIZE parameter (greater than 1).

Data Tablespace Default Storage Settings

The following table specifies the recommended sizes for data tablespace default storage:

Tablespace	Mercury Business Availability Center Deployment		Remarks
	Small	Large	
Management	➤ Locally managed tablespace ➤ Automatic segment space management ➤ Automatic local extent management		
Profile	➤ Locally managed tablespace ➤ Automatic segment space management ➤ Automatic local extent management		

Note: The profile database contains 530 tables and 2400 indexes. When creating profile database tablespaces for a large implementation of Mercury Business Availability Center, it is strongly recommended that you use automatic and not uniform local extent management.

Redo Log Settings

The following table specifies the recommended sizes for redo log files:

Setting	Mercury Business Availability Center Deployment	
	Small	**Large**
Redo log file size	70 MB	200 MB
Minimum number of groups	4	4
Minimum number of members per group	2	2

Undo Segment Settings

The following table specifies the recommended sizes for rollback segments:

Setting	Mercury Business Availability Center System Profile		Remarks
	Small	**Large**	
Undo tablespace size	500 MB	2 GB	The number of segments, the minimum number of extents, and the rollback segment size (initial, next) are all set automatically by Oracle.
UNDO_ MANAGEMENT parameter	AUTO		Oracle default values.
UNDO_ RETENTION parameter	Oracle default value		

14

Oracle Summary Checklist

This chapter contains a checklist summarizing the requirements for Mercury Business Availability Center support and certification.

This chapter describes:	On page:
Checklist for Mercury Business Availability Center Support and Certification	138
Oracle Server Requirements	145
Oracle Client Requirements	146

Note: For more information on the Oracle database configuration settings that should be used when working with Oracle Server and Mercury Business Availability Center, see "Oracle Server Sizing Guidelines" on page 125.

Checklist for Mercury Business Availability Center Support and Certification

The following table summarizes the Oracle database options that are supported and certified for working with Mercury Business Availability Center:

Option	Supported	Recommended	Remarks	For more information, see
Oracle edition	Server & Enterprise	Enterprise		"Choosing an Edition of Oracle Server" on page 88
Dedicated Mercury Business Availability Center server	Not necessary	Necessary		
Use of multiple Oracle instances	Yes	No	The configuration of all instances must match in a certified environment.	"Oracle Instances" on page 88
Use of non-default port	Yes	Yes		
Undo management	Automatic; Manual	Automatic	Set UNDO_MANAGEMENT parameter to AUTO in certified environment	

Option	Supported	Recommended	Remarks	For more information, see
Manual undo management + Public rollback segments	Yes	No		
Oracle multi-threaded server (MTS)	Yes	No	Mercury Business Availability Center uses a connection pool architecture.	
Oracle replication	No full support	No		
Use of SYSTEM, RBS, UNDO, TEMP tablespaces	Not allowed for Mercury Business Availability Center schema	Not allowed for Mercury Business Availability Center schema		
Windows and UNIX file compression	No	No	Not supported by Oracle; Causes abnormal behavior and affects performance	
Database control files required	Greater than or equal to 2	3	Preferable on different disks.	

Option	Supported	Recommended	Remarks	For more information, see
Redo log groups	Greater than or equal to 3	4	Oracle enables the software mirroring of redo log files. This is achieved by creating at least two members of redo log in each group. Members of the same group should reside on different disks.	
Character set	WE8ISO8859P1; UTF8	UTF8		
OPTIMIZER_ INDEX_COST_ ADJ parameter value	100	100	Affects performance	
HASH_JOIN_ ENABLED parameter value	True; False	True	This parameter is obsolete in Oracle 10g	
TIMED_ STATISTICS parameter value	True; False	True		
DB_CACHE_ ADVICE parameter value	Off; On	On		"Oracle Parameter Sizing" on page 127

Option	Supported	Recommended	Remarks	For more information, see
STATISTICS_ LEVEL parameter value	Typical	Typical		"Oracle Parameter Sizing" on page 127
LOG_ CHECKPOINT_ INTERVAL parameter value	Greater than or equal to 0	0		
LOG_ CHECKPOINT_ TIMEOUT parameter value	Greater than or equal to 0	0; or greater than or equal to 1800		
OPTIMIZER_ MODE parameter value	**Oracle 9i**: Choose **Oracle 10g**: ALL_ROWS	**Oracle 9i**: Choose **Oracle 10g**: ALL_ROWS		
PARTITION_ VIEW_ ENABLED parameter value	True; False	True	Performance boost when using the Mercury Business Availability Center Purging Manager. This parameter is obsolete in Oracle 10g.	
CURSOR_ SPACE_FOR_ TIME parameter value	True; False	False		

Option	Supported	Recommended	Remarks	For more information, see
CURSOR_ SHARING parameter value	Exact	Exact		
LOG_BUFFER parameter value	greater than 512,000 bytes	greater than 512,000 bytes		
USE_STORED_ OUTLINES parameter value	No	No		
OPEN_CURSORS parameter value	800	800		
COMPATIBLE parameter value	9.2.06	10.2.0.1		
SQL_TRACE parameter value	True; False	False		
Working in archive log mode	True; False	True		
LOG_ARCHIVE_ START parameter value	True; False	True	Only when working archive log mode is set to “True”	
BLANK_ TRIMMING parameter value	False	False		

Option	Supported	Recommended	Remarks	For more information, see
FIXED_DATE parameter value	Not set	Not set	Mercury Business Availability Center uses the SYSDATE function for generating system time as part of the application process.	
SPIN_COUNT parameter value	Yes	No		
UNDO_ MANAGEMENT parameter value	Manual; Auto	Auto		
UNDO_ RETENTION parameter value	Oracle default value	Oracle default value	Automatic tuning is performed in Oracle 10g	"Undo Segment Settings" on page 135
WORKAREA_ SIZE_POLICY parameter value	Manual; Auto	Auto		"Oracle Parameter Sizing" on page 127. See the SORT_ AREA_SIZE parameter as well.
Autoextend option in tablespace files	Yes	No		"Management User Schema" on page 95

Option	Supported	Recommended	Remarks	For more information, see
Locally managed data tablespace	Yes	Yes		"Locally Managed Tablespaces" on page 92
Tablespace extent management	Dictionary; Local; Local uniform	Local automatic for manage-ment and profile schemas; Local uniform for TEMP tablespace		"Oracle File Sizing" on page 132
RECYCLEBIN	Off	Off		
Automatic Segment Space Management Tablespace	Yes	Yes		
MDAC version	2.5; 2.52; 2.61; 2.62; 2.7 SP1 Refresh	2.7 SP1 Refresh	Set three registry settings according to the Oracle Client installed.	"MDAC for Oracle Client" on page 105

Oracle Server Requirements

The following table describes the Oracle Server versions and operating system platforms that are supported and certified for working with Mercury Business Availability Center.

Component	Supported		Recommended	
	Version/Edition	**Service Pack**	**Version/Edition**	**Service Pack**
Windows Operating System	Windows 2000 Server/Advanced Server	Service Pack 4	Windows 2003 Server standard / enterprise	Service Pack 1
Sun Solaris Operating System	Solaris 9		Solaris 8, 10	
Oracle	Oracle 9.2.0.6		Oracle 10.2.0.1	

Note:

- Solaris environments are 64-bit only.
- Oracle environments for Oracle 9i are 32-bit on all platforms and for Oracle 10g, 32-bit for windows and 64-bit for UNIX.

Oracle Client Requirements

The following table describes the Oracle Client versions and operating system platforms that are supported and certified for working with Mercury Business Availability Center.

Component	Supported		Recommended	
	Version/Edition	**Service Pack**	**Version/Edition**	**Service Pack**
Windows Operating System	Windows 2000 Server/Advanced Server	Service Pack 4	Windows 2003 Server standard / enterprise	Service Pack 1
Sun Solaris Operating System	Solaris 9		Solaris 8, 10	
Oracle	Oracle 9.2.0.6		Oracle 10.2.0.1	

Part IV

Appendixes

A

Database Consolidation and Distribution

This appendix describes the situations in which you should consolidate all Mercury Business Availability Center data in a single database, as opposed to the situations in which you should distribute the data among several different databases.

This chapter describes:	On page:
Advantages of Database Consolidation	149
Reasons for Database Distribution	150

Advantages of Database Consolidation

It is generally recommended that you consolidate all data in a single database. The reasons for data consolidation include:

- **Manageability** – A consolidated database reduces administration workload because there is a single point to manage, maintain, and monitor. When working with multiple databases, you must duplicate all maintenance tasks and monitoring procedures.
- **Easier space management** – With a single database, the data is contained in a single disk or disk array and it is easier to track database growth and disk space problems. With multiple databases and distributed disks or disk arrays, you must track the growth of each database and space management becomes more complex.
- **Report limitations** – Percentile and Standard Deviation reports are only supported on a single database.

Reasons for Database Distribution

The distribution of data among several databases should be considered in the following scenarios:

- **Table size** – Activity performance against small tables (especially insert rates) is generally better than activity performance against large tables. It is recommended that tables be no larger than 1 million rows. To keep table sizes fairly small, you should use partitioning and not distribute the data among different databases. You can fine-tune the partitioning cycle to achieve optimal table sizes. However, if the database is very large (more than 80 GB), keeping table sizes no larger than 1 million rows means that the partitioning cycle would be less than a day. In this case, data for a month would require more than 30 partitions. This scenario should be avoided and you should consider distributing the data among several databases so that the partitioning cycle is not less than a day and table sizes do not exceed 1 million rows.
- **Maintenance granularity** – If you have different maintenance requirements (such as data purging and backup frequency) for different profiles, you may want to use a different database for each group of profiles.

  In addition, if you want to schedule maintenance activities (such as index rebuilds) that require downtime and you do not have a sufficient window for a single maintenance task, you can distribute the data among several databases so that a single maintenance task is completed within the allotted window. Similarly, the unit of recovery is a single database. If there are profiles that you must recover quickly in the case of a system failure, you may want to consider placing these files in a separate database.
- **Physical resources** – If you want to provide different physical resources to different profiles—for example, you want to allocate fast, expensive disks to very active profiles and slower, less expensive disks to profiles with less activity—you can achieve this by distributing the profiles among different databases.

B

Data Storage Recommendations

This appendix contains recommendations for creating MS SQL Server file groups and Oracle tablespaces in which to store Mercury Business Availability Center data. This information applies to advanced users only.

This chapter describes:	On page:
Creating MS SQL Server File Groups	151
Creating Oracle Server Tablespaces	156

Creating MS SQL Server File Groups

By default, Mercury Business Availability Center stores management database objects in two file groups and profile database objects in two file groups (by default, the **PRIMARY** file group for system tables and the **userdata001** file group for user tables). When automatically creating management and profile databases, therefore, Mercury Business Availability Center creates two file groups for the management database (**PRIMARY** and **userdata001**) and two file groups for the profile database (also **PRIMARY** and **userdata001**). If you are setting up a large deployment of Mercury Business Availability Center, it is recommended that you create additional file groups in which to store management and profile database objects.

Benefits of Multiple File Groups

Placing the system catalog in its own file group and creating several user-defined file groups for data storage has the following benefits:

- **Maintenance** – Backing up and restoring databases can be performed on a file group level.
- **Partial restoration** – You can partially restore a database on the file group level; however when you do so, you must restore the **PRIMARY** file group as well, as it contains the system catalog. By keeping only the **.mdf** file in the **PRIMARY** file group and setting another user file group as the default file group, the **PRIMARY** file group remains small and does not pose difficulties for a partial restoration. This suggested file group structure creates a single user file group, which is a step toward multiple file groups that can be created by the database administrator or by Mercury Business Availability Center in the future.
- **Read-only** – A file group can be marked as read-only. Because data cannot be changed, MS SQL Server does not have to acquire shared locks and performance is enhanced.
- **Recovery** – If the data portion of a database crashes, you can save the changes made before the crash even if the **.mdf** file is damaged, provided the log file and master database are undamaged.
- **Performance** – Numerous tuning options are available, such as hot tables on fast disks, the separation of table data and indexes into different file groups, and so forth.
- **Flexibility and Control** – The use of multiple file groups provides the database administrator with greater flexibility and control. In addition, the use of a fixed size for database growth prevents situations in which the 10 percent default autogrowth increment in large databases cannot finish before the timeout setting of Mercury Business Availability Center components and rolls back database activity, causing an infinite loop of inserts.

You can create an additional file group (together with an additional MS SQL Server database) by running the following scripts.

To create a database with default settings:

```
CREATE DATABASE [testdb]
```

To modify the .mdf (catalog) file SIZE parameter:

```
ALTER DATABASE [testdb]
MODIFY FILE
(
        NAME = N'testdb',
        SIZE = 5MB
)
```

To modify the .mdf (catalog) file FILEGROWTH parameter:

```
ALTER DATABASE [testdb]
MODIFY FILE
(
        NAME = N'testdb',
        FILEGROWTH = 5MB
)
```

To modify the .ldf (log) file SIZE parameter:

```
ALTER DATABASE [testdb]
MODIFY FILE
(
        NAME = N'testdb_log',
        SIZE = 10MB
)
```

To modify the .ldf (log) file FILEGROWTH parameter:

```
ALTER DATABASE [testdb]
MODIFY FILE
(
        NAME = N'testdb_log',
        FILEGROWTH = 50MB
)
```

To return the path of the database's default data directory:

```
SELECT LEFT(filename,
        LEN(filename) - CHARINDEX(N'\', REVERSE(filename))) AS path
FROM master.dbo.sysdatabases
WHERE name = N'testdb'
```

Use the '**testdb**' name returned above in the following script.

To create a user file group, add a data file to it, and set it as the default file group:

```
ALTER DATABASE [testdb]
ADD FILEGROUP USERDATA001

ALTER DATABASE [testdb]
ADD FILE
(
        NAME = N'testdb_Data001',
        FILENAME = N'<path to data directory>',
        SIZE = 10MB,
        FILEGROWTH = 50MB
)
TO FILEGROUP USERDATA001
```

To set the userdata001 file group as the default file group:

```
ALTER DATABASE [testdb] MODIFY FILEGROUP USERDATA001 DEFAULT
```

Note the following when creating file groups:

- You should create two separate data files, in two separate file groups: the **.mdf** file in the **PRIMARY** file group and the **.ndf** file in the **userdata001** group.
- The **userdata001** group should be set as the default file group.
- The **.mdf** file should be the only file in the **PRIMARY** file group that contains the system catalog. The **.mdf** file should be created with an initial SIZE parameter value of 5 MB and a FILEGROWTH parameter value of 5 MB. The other data file (**.ndf**) and log file (**.ldf**) should be created with an initial SIZE parameter value of 10 MB and a FILEGROWTH parameter value of 50 MB. In all cases, the FILEGROWTH parameter value should be a fixed size (not a percentage).

The above configuration rules are important for proper file recovery, fault tolerance, performance, and archiving.

Creating Oracle Server Tablespaces

By default, each Oracle user schema has one tablespace defined for it. All Mercury Business Availability Center Oracle database objects (such as tables and indexes) that are created are automatically stored within this tablespace. It is recommended, however, that not all database objects be stored in one tablespace. In fact, you are advised to create the following five additional tablespaces:

Implementation Size	Object Type		
	Table	Index	Lobs
Small	(1)	(2)	(5)
Large	(3)	(4)	(5)

Below are the recommended storage parameters for each tablespace:

Tablespace Number	Tablespace Type	Size	Freelists
(1)	Locally managed tablespace; Automatic allocation of extents	100-500 MB	20
(2)	Locally managed tablespace; Automatic allocation of extents	100-500 MB	20
(3)	Locally managed tablespace; Automatic allocation of extents	15-300 GB	10
(4)	Locally managed tablespace; Automatic allocation of extents	15-300 GB	10
(5)	Locally managed tablespace; Automatic allocation of extents	5-100 GB	1

The following table describes the tablespace type that should be used for each Mercury Business Availability Center table:

Tablespace Number	Table(s)
(5)	Management table: CUSTOM_REPORTS Profile table: SLA_REPORT_INSTANCES
(3) and (4)	Management table: none Profile table: EVENT_METER
(1) and (2)	All the tables mentioned above (and their indexes accordingly)

Transferring Mercury Business Availability Center Data

Once you have created the five tablespaces described above, you can transfer data from the default tablespace to the other tablespaces you created.

To transfer data to the new tablespaces created:

1. If you have not yet done so, create management and profile databases in tablespace (1).
2. Move all tables from tablespace (1) to tablespaces (3) and (5), using the ALTER TABLE <table name> MOVE TABLESPACE <tablespace name> DDL command. If you are using an older version of Oracle that does not support this command, you can recreate the tablespace (1) tables in tablespaces (3) and (5).
3. Move all indexes from tablespace (1) to tablespaces (4) and (5). You can move an index from one tablespace to another by dropping it and recreating it in the required tablespace, using a storage clause.

Note:

- When you create a profile user schema from the Database Management page, in Platform Administration's Setup and Maintenance tab, database objects are created in the Oracle user's default tablespace. It is recommended that the database administrator review the scripts that Mercury Business Availability Center uses to create profile user schemas and edit them so that they meet the needs of your application. These scripts should then be used to manually create a new profile user schema.
- When you upgrade a user schema, all objects that are created after the upgrade are created within the Oracle user's default tablespace. The database administrator can later move these objects to other tablespaces, as required.

C

Data Partitioning and Purging

This appendix describes how to update the profile database table partitioning and purging policy.

This chapter describes:	On page:
About Data Partitioning and Purging	159
How the Purging Manager Works	160
Calculating Purging Manager Parameter Values	161
Setting the Purging Manager Parameters	162
Creating Oracle Tablespaces When Using the Purging Manager	165

About Data Partitioning and Purging

Purging outdated data is an essential part of managing applications into which a large amount of data is constantly streaming. Mercury Business Availability Center uses a method called partitioning to delete outdated data. Mercury Business Availability Center's Purging Manager splits fast-growing tables into partitions at defined time intervals. After a defined amount of time has elapsed, data in a partition is made inaccessible for use in Mercury Business Availability Center reports. After an additional defined amount of time, a partition is purged from the profile database.

You can configure the partitioning and purging policy for the tables in a specific profile database, or in all Mercury Business Availability Center profile databases.

Once you have configured the partitioning and purging policy for the tables in a specific profile database, policy changes made for all profile databases will not take effect on this specific database.

Note: To run the Purging Manager, you must enable it. For details, see "Enabling and Disabling the Purging Manager" in *Platform Administration*.

How the Purging Manager Works

For each table, the Purging Manager uses three sets of parameters to determine when to create new partitions, when to make partitions unavailable and when to delete partitions.

The following are the parameter sets used:

- **SLICE and SLICE_UNITS** – used to define how often the data is divided into new partitions. Data is handled by partition, so the frequency of partition creation determines the frequency of data removal. For example, if slice is defined as 1 week, a new partition is created once a week. If the range is defined as 4 months and lifetime as 6 months, once a week, Mercury Business Availability Center makes the data older than 4 months unavailable, and purges the data older than six months.
- **RANGE and RANGE_UNITS** – used to define the length of time that data is available and can be used in Mercury Business Availability Center reports. For example, if range is defined as 6 months, data remains available to be used in Mercury Business Availability Center reports during this period. After 6 months the partition containing this data is moved to the historical view in the database.

- **LIFETIME and LIFETIME_UNITS** – used to define the length of time that data is retained in the database before being purged. Data that has exceeded its range definition but not its lifetime definition is retained in the database in the historical view, but is unavailable for use in Mercury Business Availability Center reports. For example, if lifetime is defined as 1 year and range is defined as 6 months, data is retained in the database for 1 year, but is only available to be used in Mercury Business Availability Center reports for 6 months. After 1 year the data is purged from the database.

Note: To view the data that no longer appears in Mercury Business Availability Center reports but is still retained in the database, you open the database and access the **<table name>_HIST** view.

Calculating Purging Manager Parameter Values

The Purging Manager calculates the values of the SLICE and LIFETIME parameters of a table based on the values in the RANGE and EPM_VALUE parameters of the table, and the default partition size set by Mercury Business Availability Center.

The EPM_Value parameter defines the number of events per minute that are expected to be entered into the table.

If the value of either the RANGE parameter, or the EPM_VALUE parameter is changed, the values in the SLICE and LIFETIME parameters are recalculated accordingly. For details on changing parameter values, see "Setting the Purging Manager Parameters" on page 162.

The following algorithmic principles are used in calculating SLICE and LIFETIME values:

- SLICE (in hours) = default partition size / (EPM_VALUE * 60)
- The minimum, valid SLICE is 2 hours (that is, SLICE = 2, SLICE_UNITS = Hours)

➤ If the SLICE and RANGE parameter values are equal, LIFETIME will be set to the same value. If they are not equal, the LIFETIME value will be calculated according to an algorithm based on the RANGE value.

Note: If the EPM_VALUE parameter of a table contains a null value, the Purging Manager will use the legacy calculator (from previous versions of Mercury Business Availability Center) to calculate the SLICE, SLICE_UNITS, LIFETIME, and LIFETIME_UNITS parameter values for the table. The legacy calculator does not use the EPM_VALUE parameter in its algorithms.

You can instruct the Purging Manager to work with a specific calculator by setting the **CALC** flag in the **pmmanager.properties** file, located in the **<Core Server>\conf** directory. Valid options for the **CALC** flag are:

➤ 0 – work only with the legacy calculator

➤ 1 – work only with the new calculator

➤ 2 – default. If the EPM_VALUE parameter of a table contains a null value work with the legacy calculator, otherwise work with the new calculator.

Setting the Purging Manager Parameters

Once the initial, default values have been set for the Purging Manager, you can update them by editing the **pmconfig.properties** file and reloading the values, or by changing the value of either the RANGE, or the EPM_VALUE parameters of a table.

Setting Purging Manager Default Values

Mercury Business Availability Center sets default values for the SLICE, RANGE, and LIFETIME sets of parameters, as well as for the EPM_VALUE parameter, for the tables managed by the Purging Manager.

The default values are stored in the **pmconfig.properties** file, located in the **<Core Server>\conf** directory. The first time that the Purging Manager is started, it loads the default values from this file into the **PM_CONFIG** table in the management database.

Using the pmconfig.properties File

You use the **pmconfig.properties** file, located in the **<Core Server>\conf** directory, to modify a table's SLICE, SLICE_UNITS, LIFETIME, and LIFETIME_UNITS parameter values, which you are unable to do via the Purging Manager page in Platform Administration. You can also modify a table's RANGE, RANGE_UNITS, and EPM_VALUE parameter values in this file.

Note: You should not add or delete tables from the **pmconfig.properties** file.

To modify the pmconfig.properties file:

1 Open the **<Core Server>\conf\pmconfig.properties** file.

2 Modify the **SLICE, SLICE_UNITS, RANGE**, **RANGE_UNITS**, **LIFETIME**, **LIFETIME_UNITS,** and **EPM_VALUE** parameters as required. Supported values for SLICE_UNITS, RANGE_UNITS and LIFETIME_UNITS are **hour**, **day**, **week**, **month**, and **year**. The minimum hour value, however, is 2.

Note:

- It is recommended that you do not create more than 52 partitions in the database (beyond this, report generation performance is affected). For example, if slice is defined as 1 week, the maximum value you should define for range is 1 year (since there are 52 weeks in the year).
- If you modify the slice period, your modification will not take effect until the next partition is created. You should therefore ensure that a large slice period (for example, a year) is never specified.

---

3 Save the **pmconfig.properties** file.

4 Double-click the <**Core Server**>**\bin\pmconfig.bat** file to input the updated configuration to the management database. Mercury Business Availability Center opens a Command Prompt window and runs the database update program. All profile databases are updated.

If you want to update a specific profile database only, run the <**Core Server**>**\bin\pmconfig.bat** file using the following command:

pmconfig.bat –id <database ID> [<configuration file>]

where <database ID> is the database ID listed in the SESSIONS table (and <configuration file> is optional).

5 Review the results of the update. If no errors occur, press ENTER. If errors occur, troubleshoot them. You can review errors in the <**Core Server**>**\log\pmanager.log** file. When you have resolved all errors, rerun the **pmconfig.bat** file.

Changing the RANGE Parameter Value

You use the table provided on the Purging Manager page in Platform Administration's Setup and Maintenance tab to specify the length of time that data is saved in each profile database table and can be viewed in Mercury Business Availability Center reports (the RANGE and RANGE_UNITS parameter values).

When you use the Purging Manager page in Platform Administration, the values of the SLICE, SLICE_UNITS, LIFETIME, and LIFETIME_UNITS parameters are recalculated automatically, using the new RANGE and RANGE_UNITS parameter values entered, together with the existing EPM_VALUE parameter values for the specified files.

For detailed instructions regarding the Purging Manager page in Platform Administration, see "Purging Historical Data from Profile Databases" in *Platform Administration*.

Changing the EPM_VALUE Parameter Value

You change the value of the EPM_VALUE parameter for a file by changing it directly in the PM_CONFIG table in the management database.

Note: This should be done by a database administrator.

Once you have changed the value of the EPM_VALUE parameter in the PM_CONFIG table, go to the Purging Manager page in Platform Administration, select the relevant table, and click **Apply**. This recalculates the values of the SLICE, SLICE_UNITS, LIFETIME, and LIFETIME_UNITS parameters using the new EPM_VALUE parameter value entered, and the existing RANGE and RANGE_UNITS parameter values for the table.

Creating Oracle Tablespaces When Using the Purging Manager

The Purging Manager creates partitions in the default Oracle tablespace. These partitions automatically acquire the storage parameters of this tablespace. If you want to enable the Purging Manager to create partitions of fast-growing tables in a tablespace with appropriate storage capabilities, it is recommended that you create two tablespaces—one for slow-growing tables and the other for fast-growing tables. Configure the storage parameters of the two tablespaces as required for each type of table and set the tablespace created to accommodate fast-growing tables as the default tablespace. This enables the Purging Manager to create partitions within a tablespace that contains sufficient storage space.

Note: If you have enabled the Purging Manager and are working with an Oracle database, it is strongly recommended that you set the PARTITION_VIEW_ENABLED parameter in the Oracle initialization file to **True**.

D

Database Schema Verification

This appendix describes how to verify your management and profile schemas to help ensure that your database schemas are configured properly.

This chapter describes:	On page:
About the Verification Process	167
Running the Verification Process	168
Creating Database Users for the Verify Procedure	173

About the Verification Process

You verify the management and profile schemas using the database schema verify program. The verification process does not involve downtime for your system.

The database verify program checks your management and profile database schemas so that they match the correct database schemas for the schema version you specify (the default schema version is that being used, as detected by the verification program).

In addition to verifying the management and profile database schemas, the verification process will also determine if there is a later schema version to which you can upgrade. If a newer version of the database schemas is detected, the verification process will notify you of possible lengthy operations that could slow down a future upgrade.

During the course of the verification process, you will be asked for a user name and password, which is required for certain (read-only) tests to be performed. If you do not want to supply your DBA account user name and password, you can create a user name with the minimum privileges required for the verify program to operate. For details on how to create this user, see "Creating Database Users for the Verify Procedure" on page 173.

Notes and Limitations

The following notes and limitations apply when running the database schema verify program:

- If you are running the database verify program on Oracle 10g schemas and use Oracle datapump utilities to import or export the target schemas, ensure that you do not have any active datapump jobs running against the target schemas.

  If you have datapump tables in the target schemas, they should be dropped prior to running the database schema verification program.

  It is recommended to assign an administrator schema to perform datapump operations and not to use Mercury Business Availability Center schemas as the login. By assigning an administrator schema to perform datapump operations, you do not have to grant additional permissions to Mercury Business Availability Center schemas and the datapump tables will be created in the administrator schema.

- If the same database server is listed in the management database using two different host names (for example, dbserver and dbserver.merc-int.com), you will be prompted for connection separately for each host name. Make sure that you enter the same connection parameters each time.

Running the Verification Process

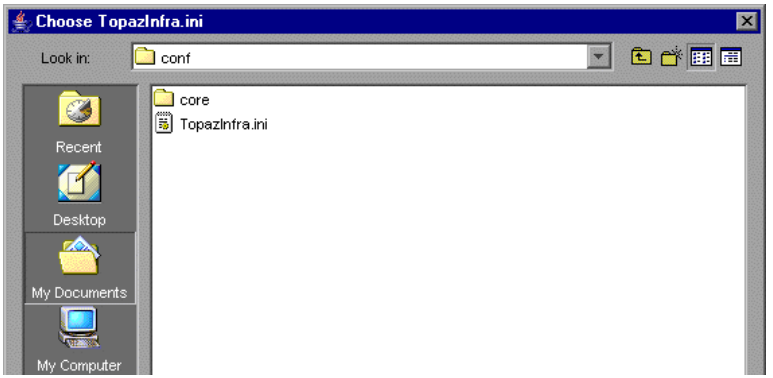
You run the database schema verify program from a Mercury Business Availability Center server machine.

To verify a database schema:

1 Go to the **<Mercury Business Availability Center server root directory>\dbverify\bin** directory and run the **run_db_verify.bat** file.

The database verify program starts.

2 In the Choose TopazInfra.ini dialog box, browse to the **<Mercury Business Availability Center server root directory>\conf\TopazInfra.ini**.

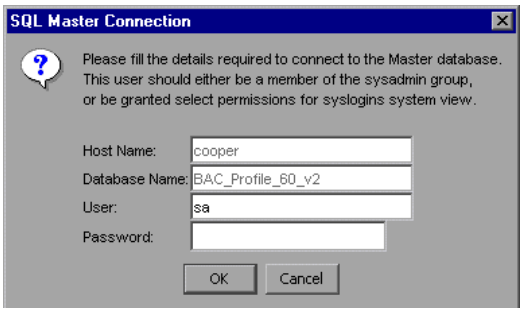


Note: If you have already run the dbverify process and your database settings have not changed, you can select the **<Mercury Business Availability Center server root directory>\dbverify\conf\TopazInfra.ini** default that is displayed.

---

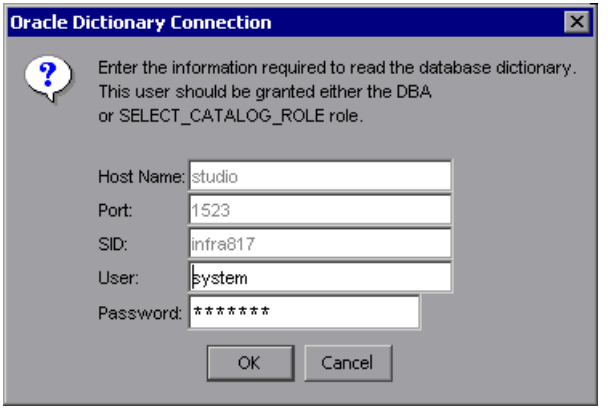
3 Specify the details required to connect to the appropriate database:

➤ In the SQL Master Connection dialog box, specify the details required to connect to the Master database.



In the **User** and **Password** boxes, type the user name and password of a user with permissions for the database. (The User box displays the default MS SQL Server administrator user name, **sa**; by default, there is no password.) Click **OK**.

➤ In the Oracle Dictionary Connection dialog box, specify the details required to connect to the Oracle database.



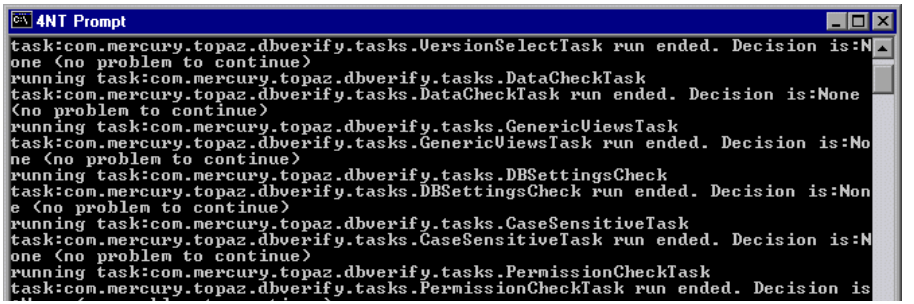
In the **User** and **Password** boxes, type the user name and password of a user with permissions for the database, and click **OK**.

Note: You will be prompted for connection data for each different server on which your management and profile databases reside.

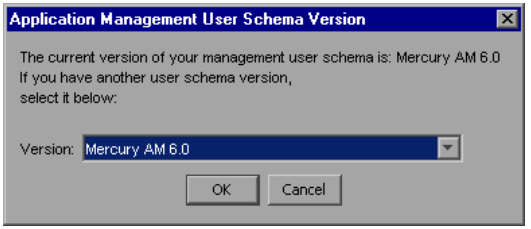
Note: If you do not want to supply your database administrator account user name and password, you can create a user name with the minimum privileges required for the verify program to operate. For details on how to create this user, see “Creating Database Users for the Verify Procedure” on page 173.

---

4 The database verify program performs database verification. You can view the progress of the verify process in a command prompt window.

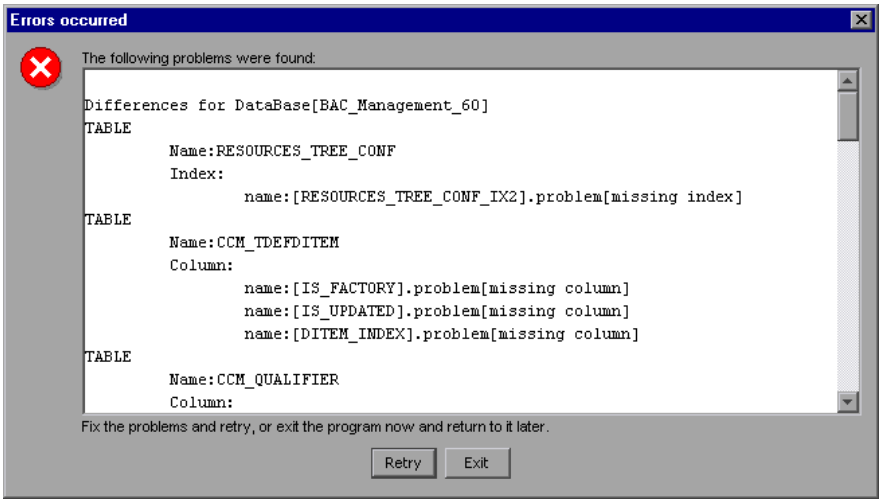


5 Check that your schema version is displayed in the Application Management User Schema Version dialog box.



Click **OK**.

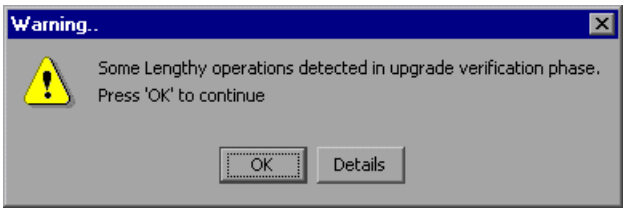
6 If problems occur during the database verification, a dialog box is displayed listing the errors.



Either fix the problems found and click **Retry**, or click **Exit** and rerun the database schema verify program at a later date. If you are unable to fix the problems, contact Mercury Customer Support for assistance.

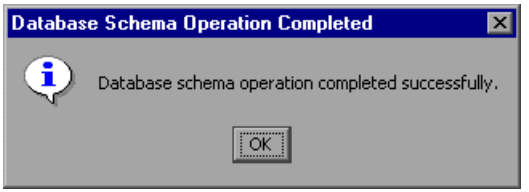
You can view a log file of the errors located in the **<Mercury Business Availability Center server root directory>\dbverify\log** directory.

7 If the verification process detected a later version of the database schemas to which you can upgrade, and encountered lengthy operations that could slow down the upgrade process, resulting in longer downtime, a dialog box is displayed.



Click **OK** to continue, or **Details** to display the estimated times that the lengthy operations detected could take during an upgrade.

8 If the database verification is successful, the following confirmation message is displayed.



9 Click **OK**.

Note: For details on database verification troubleshooting issues, see the Mercury Customer Support Knowledge Base (http://support.mercury.com/cgi-bin/portal/CSO/kbBrowse.jsp).

Creating Database Users for the Verify Procedure

When running the database schema verify and upgrade utility, you are prompted to supply a user name and password that can access the master database. You can create users with minimum privileges by running one of the following scripts.

For MS SQL Server

```
set nocount on
use master
GO
sp_addlogin @loginame ='dbv_read',@passwd = '<pass>'
GO
sp_adduser @loginame = 'dbv_read', @name_in_db = 'dbv_read'
go
grant select on syslogins to dbv_read
go
set nocount off
```

Note: You must run this script as an **sa** user.

For Oracle Server

```
CREATE USER dbv_read IDENTIFIED BY admin;
GRANT SELECT_CATALOG_ROLE TO dbv_read;
GRANT CONNECT TO dbv_read;
```

Note: You must run this script as a system user.

E

Managing Data with Loaders

This appendix explains how Mercury Business Availability Center uses loaders to manage the data streaming from data collectors to the Core Server and on to the database. The loader system enables you to verify that data collected by Mercury Business Availability Center is being reported to the database. Mercury Customer Support also uses loader information when helping to troubleshoot data problems.

This chapter describes:	On page:
Overview of Managing Data with Loaders	176
Troubleshooting Loader Problems	177
Loader Folder Locations	177
Changing the Default Logging Level	179
Changing the Time Frame for Retaining Failed Data	180
The DLQ Cleanup Tool	181

Overview of Managing Data with Loaders

Mercury Business Availability Center loaders enable efficient and persistent reporting of data from data collectors to the Core Server and on to the database.

Data streams, via HTTP requests, from a data collector (Business Process Monitor, Client Monitor, SiteScope, and so forth) to the loader folders on the Core Server. Every 60 seconds, the loaders automatically transfer data from the loader persistency folders to the database. If the database is unavailable, the loaders transfer the data to the loader persistency recovery folders. If the data is not successfully transferred to the database, for example, because the data is corrupt, the loaders transfer the data to the DLQ ("dead letter queue") folders where it is stored. For the locations of the loader folders, see "Loader Folder Locations" on page 177.

You can use the DLQ Cleanup tool to transfer the data that is stored in the DLQ folders to the database. For details, see "The DLQ Cleanup Tool" on page 181.

A primary indication of data transfer problems is the persistent increase in the number of files in the loader folders. For details on how to recognize this problem, see the next section.

The loader logs store information about loader issues. This information is useful when debugging data transfer problems, at which time you will need to change the default logging level. For details, see "Changing the Default Logging Level" on page 179. The following three log files are used:

- **collector_log** – the main loader log file containing details of all the actions performed by the loader.
- **dbloader_statistics** – containing statistical information about the loader's performance.
- **dbloader_samples** – containing copies of all the data samples that pass through the loader.

Troubleshooting Loader Problems

You can troubleshoot problems involving the transfer of data to the database by checking the loader log files and by checking the number of files in each loader directory.

- Check the log files for error messages. If the level of logging is set to **flow** (or **info**), you will see all **fatal**, **error**, and **warning** messages in the log files. If requested by Mercury Customer Support, you can change the default logging level. For details, see "Changing the Default Logging Level" on page 179. For a definition of Mercury Business Availability Center log levels, see "Log Severity Levels" in *Reference Information*.
- Check whether there is an increase in the number of files in a directory. This is an indication of a persistency problem. You should contact Mercury Customer Support. For a description of the loader directories, see the next section.

Loader Folder Locations

Each profile database has its own directory that contains the loader data, as follows:

Loader persistency directories – include data that is to be transferred to the database

Loader persistency recovery directories – include data that was not transferred to the database because the database was not available. Used to troubleshoot loader problems.

DLQ directory – includes data that was not transferred to the database because data transference was unsuccessful

The loaders automatically transfer data from the loader persistency directories to the database. If the database is unavailable, the loaders transfer the data to the loader persistency recovery directories. If the data is not successfully transferred to the database, for example, because the data is corrupt, the loader transfers the data to the DLQ directory.

Example of Folder Locations

In this example, the following names are used:

- the Core Server is running on a machine named **coreserv**
- the database name is **BAC_6**

The paths to the data files are as follows:

Loader persistency directories:

.persist_dir\lnch_persistent\coreserv_project_topaz
\guarantee\+coreserv_project_topaz_coreserv_topaz_collector\.msgs

Loader persistency recovery directories:

.persist_dir\lnch_persistent\coreserv_project_topaz
\guarantee\+coreserv_project_topaz_coreserv_BAC_6\.msgs

DLQ directory:

.persist_dir\lnch_persistent\coreserv_project_topaz
\guarantee\+coreserv_project_topaz_coreserv_dl_queue_host_BAC_6\.msgs

The loader log files are located at:

<Core Server root directory>\MercuryAM\log\collector_log.txt

<Core Server root directory>\MercuryAM\log\dbloader_statistics.txt

<Core Server root directory>\MercuryAM\log\dbloader_samples.txt

The loader configuration file is located at:

<Core Server root directory>\MercuryAM\conf\collector.ini

Changing the Default Logging Level

As the loaders work automatically, you will generally not need to make any changes to the loader files. You can change the level of logging, however, if requested by Mercury Customer Support, from a reporting level to a debug level.

To change the level of logging:

1 Access the file **<Core Server root directory>\MercuryAM\conf\collector.ini**.

2 In the section for the log file whose logging level you want to change, locate the entry

```
;; Verbosity level: error|warning|flow|debug1-debug5
LogLevel="flow"
```

Note: By default, the LogLevel parameter is set to **flow** for the collector_log and dbloader_statistics logs, and **error** for the dbloader_samples log.

---

3 Change the LogLevel parameter setting (**error** or **flow)** to one of the debug levels: each debug level adds more detailed information; debug5 contains the most information.

4 Save the file.

If you have multiple Core Server machines, repeat this procedure on every machine.

Changing the Time Frame for Retaining Failed Data

If the loader cannot send data to the database, for example, because of primary key violations or check constraint violations, the loader writes the data to a DLQ directory (for details, see "Loader Folder Locations" on page 177).

Mercury Business Availability Center uses the following parameters to determine how long to keep data in the DLQ directory:

- **DLQConnectionsMessageLifeTime** – sets the amount of time a message should be kept, from the time it was written to the DLQ directory. The default setting for this parameter is 1440 hours (60 days.)
- **DaysRange** – sets the period of time during which a message can be inserted into the Profile database, from the time the message was created in Mercury Business Availability Center. Messages that are older than this time period will remain in the DLQ directory until they are deleted according to the DLQConnection MessageLifeTime parameter setting. The default setting for the DaysRange parameter is 60 days.

  You can change the settings for these parameters, but it is not recommended to shorten the time periods for which data is kept. If necessary, you can lengthen them, for example, if you want to keep data until Mercury Customer Support has dealt with a question.

To change the time period that Mercury Business Availability Center keeps failed data:

1. Access the file **<Core Server root directory>\MercuryAM\conf\collector.ini**.
2. Locate the entry for the parameter you are changing:

```
;; Defines the expiration time of samples in the DLQ.
;; If the sample life time is greater than this value (in hours) the bus will not pass the sample from dlq and will throw it.
DLQConnectionsMessageLifeTime=1440
```

or

```
;; highest accepted delay in days
DaysRange="60"
```

3 Change the time period. The DLQConnectionsMessageLifeTime parameter is set in hours, and the DaysRange parameter is set in days.

Note: The values in both parameters should be equal, so that the number of hours in the DLQConnectionsMessageLifeTime parameter is equivalent to the number of days in the DaysRange parameter times 24.

---

4 Save the file.

If you have multiple Core Server machines, repeat this procedure on every machine.

The DLQ Cleanup Tool

The DLQ Cleanup tool takes the data stored in the DLQ directory and transfers it to the database.

If data transfer problems still occur, and some of the data does not reach the database, the data is returned to the DLQ directory. You can run the tool as often as necessary until all the data reaches the database. Any data remaining in the DLQ directory for more than 60 days is deleted.

Running the DLQ Cleanup Tool

You run the DLQ Cleanup tool as an administrator on a Windows platform, or as the Mercury Business Availability Center user defined during installation on a Unix platform You run the DLQ Cleanup tool using the command line.

To run the DLQ Cleanup tool:

1 From a command window on the Core Server, launch the DLQ Cleanup batch file according to your operating platform:

- On a Windows platform by typing the following, and press **Enter**:
  <Mercury Business Availability Center root directory>\tools\DLQ_cleanup\dlq_cleanup.bat <machine name>
- On a Unix platform type the following and press **Enter**:
  <Mercury Business Availability Center root directory>/tools/DLQ_cleanup/dlq_cleanup.sh <machine name>

where <machine name> is the machine name as written in the URL of the Core Server on which the loader is installed. For example, if the Core Server is called CoreServer1 and is part of a domain called mydomain.com, then <machine name> could be either CoreServer1, or CoreServer1.mydomain.com.

2 When the DLQ Cleanup tool has successfully ended, the confirmation message **done** is displayed.

Note: In a distributed system where there is more than one Core Server, the DLQ Cleanup tool must be run for each Core Server in the system.

F

Backing Up and Restoring Monitor Administration Configuration Data

This appendix describes the backup and restore procedures for Monitor Administration configuration data. Establishing dedicated backup and restore processes is a fundamental task in a high-availability implementation.

This chapter describes:	On page:
Overview of Backing Up and Restoring Configuration Data	184
Basic Backup and Restore	185
Basic Backup on Replicated Server and Restore	186
LDIF Backup and Restore	186
Replication	189
Starting and Stopping OpenLDAP	192
Backup and Restore Examples	194

Note: For details on setting up automatic replication and failover procedures for the LDAP server, refer to Mercury Customer Support Knowledge Base article 42154.

Overview of Backing Up and Restoring Configuration Data

Monitor Administration configuration data is stored in an LDAP database (Lightweight Directory Access Protocol). By default this database resides in **<Mercury Business Availability Center server root directory>\openldap\bdb\id*.***. Any backup and restore processes should be performed on this directory.

This section explains about backup and restore methods and about LDAP database locations.

Backup and Restore Methods

Choose between the following methods for backing up and restoring your Monitor Administration data files:

- Basic Backup and Restore
- Basic Backup on Replicated Server and Restore
- LDIF Backup and Restore

LDAP Database Location

To verify the location of the LDAP database that is used for storing the Monitor Administration data:

1. Access **Admin** > **Platform** > **Setup and Maintenance** > **Infrastructure Settings**.
2. Select the **Foundations** context.
3. Choose **Monitor Administration** from the list.
4. Locate the **Monitor Administration Data Storage Location** entry.

Basic Backup and Restore

This is the basic backup and restore method and is the preferred method. The LDAP service must not be running during the physical copying of the configuration database, to prevent the possibility of cached data and file locking.

Backup

It is good practice to back up the data at least once daily. Store the data on a high-availability server where the files are stored in directories with a date–time stamp name such that multiple historical archives can be preserved. Another technique is to back up to a specific high-availability location using a rotating day of the week, for example, **..\\backups\Sunday**, **..\\backups\Monday**, and so forth.

It is also considered good practice to back up before and after any major upgrade or delete operation.

To manually back up your LDAP database:

1. Stop the LDAP service by stopping Mercury Business Availability Center (**Start** > **Programs** > **Mercury Business Availability Center** > **Administration** > **Disable Business Availability Center**).
2. Copy the contents of the directory **<Mercury Business Availability Center server root directory>\openldap\bdb** to the backup media. You can do this manually or by running a batch file. For an example of a backup file, see "backup.bat" on page 194.
3. Start the LDAP service by re-starting Mercury Business Availability Center (**Start** > **Programs** > **Mercury Business Availability Center** > **Administration** > **Enable Business Availability Center**).

Restore

To manually restore your LDAP database:

1. Stop the LDAP service by stopping Mercury Business Availability Center (**Start** > **Programs** > **Mercury Business Availability Center** > **Administration** > **Disable Business Availability Center**).

2 Delete all the files in the **<Mercury Business Availability Center server root directory>\openldap\bdb** directory. (It is good practice to copy these files to a temporary location before deleting them.)

3 Copy the files from the appropriate source to the appropriate target media. You can do this manually or by running a batch file. For an example of a restore file, see "restore.bat" on page 195.

4 Start the LDAP service by re-starting Mercury Business Availability Center (**Start > Programs > Mercury Business Availability Center > Administration > Enable Business Availability Center**).

Basic Backup on Replicated Server and Restore

This method is identical to the Basic Backup and Restore procedure, except that the Basic Backup process is performed on the replicated server file system. It is the most highly recommended solution for high-availability.

The advantage is that the primary service (including Mercury Business Availability Center) does not need to be shut down for the duration of the operation. Furthermore, the replicated server acts as a cold-backup in the event of a failure of the primary server.

LDIF Backup and Restore

This method uses the LDAP Data Interchange Format (LDIF) representation to extract the contents of an LDAP database to an external, textual, representation.

Note that, with OpenLDAP 2.1.4 or higher, using the back-bdb backend (the default) you can use slapcat.exe and db_dump.exe while slapd.exe is running. This is because the back-bdb backend has page level locking.

Backup

To back up the data to LDIF format:

Note: If you are running OpenLDAP 2.1.4 or higher, you can skip steps 1 and 2 below.

---

1. Stop the LDAP service by stopping Mercury Business Availability Center (**Start** > **Programs** > **Mercury Business Availability Center** > **Administration** > **Disable Business Availability Center**).
2. From a command window (with sufficient rights to read the files), change the working directory to the <**Mercury Business Availability Center server root directory**>/**openldap** directory.
3. Invoke the following command:

```
slapcat –f slapd.conf –b “E=SSEnterprise” –f filename
```

Where:

filename is the target LDIF file, for example, **JUL0404.LDIF**.

For an example of a backup file, see “backupLDIF.bat” on page 196.

4. Start the LDAP service by re-starting Mercury Business Availability Center (**Start** > **Programs** > **Mercury Business Availability Center** > **Administration** > **Enable Business Availability Center**).

Restore

It is excellent practice to test the following process on a staging or test LDAP server.

To restore the data to a new LDAP database:

Note: If you are running OpenLDAP 2.1.4 or later, you can skip steps 1 and 2 below.

---

1. Stop the LDAP service by stopping Mercury Business Availability Center (**Start** > **Programs** > **Mercury Business Availability Center** > **Administration** > **Disable Business Availability Center**).
2. From a command window (with sufficient rights to read the files), change the working directory to the <**Mercury Business Availability Center server root directory**>**/openldap** directory.
3. Delete all the files in the **bdb** directory (it is good practice to copy these to a temporary location before deleting).
4. Invoke the following command:

```
slapadd –f slapd.conf –b “E=SSEnterprise” –f filename
```

Where:

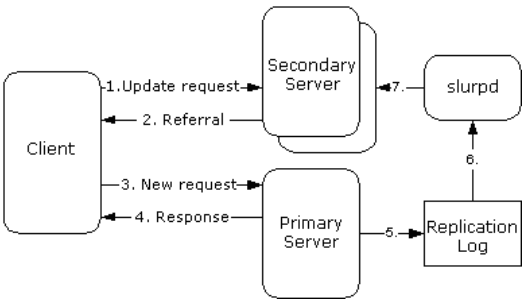
filename is the target LDIF file, for example, **JUL0404.LDIF**.

For an example of a restore file, see “restoreLDIF.bat” on page 197.

5. Start the LDAP service by re-starting Mercury Business Availability Center (**Start** > **Programs** > **Mercury Business Availability Center** > **Administration** > **Enable Business Availability Center**).

Replication

OpenLDAP provides replication services (see slurpd at http://www.openldap.org/software/man.cgi?query=slurpd). You can use these services to maintain a cold-backup of all information on your primary server (the host machine) replicated to a secondary machine. You should consult with the Mercury Business Availability Center system administrator before using these services.



This section contains the following topics:

- Establishing Replication
- Modifying the Nanny Supervisor Process for Replicated Environments

Establishing Replication

The following procedure assumes that Mercury Business Availability Center is running on a primary machine that also has an LDAP server installed on it, and that a secondary machine is running the secondary LDAP server.

To establish a replicated OpenLDAP service:

1. Stop Mercury Business Availability Center.
2. Zip the **openldap** directory (<**Mercury Business Availability Center server root directory**>**openldap**).
3. Transfer the **openldap.zip** file to the secondary machine and unzip it.
4. Add a registry entry as needed.

   For details on adding a registry entry (when running openldap as a service on a non-standard port), see the next section.

5 Install the **slapd** service on the secondary LDAP server:

```
cd c:\openldap
slapd install
```

6 Edit the **slapd.conf** file running on the primary LDAP server (located in the **<Mercury Business Availability Center server root directory>\openldap** directory) and add or uncomment the following lines:

```
# Replication MASTER
Replica uri=ldap://slaveHost:9389 binddn="E=SSEnterprise"
bindmethod=simple credentials=fl1pp3r
replogfile ./replicate.log
```

Note: slaveHost is the name of the LDAP server running on the secondary machine.

---

7 Edit the **slapd.conf** file running on the secondary LDAP server (located in the **openldap** directory) and add or uncomment the following line:

```
#Replication SLAVE
updatedn "E=SSEnterprise"
updateref ldap://mercuryAMhostname:9389 minServer installation
Change
```

Note: mercuryAMhostname is the name of the Mercury Business Availability Center primary machine.

---

8 Locate the file **replica.log** in the **<Mercury Business Availability Center server root directory>\openldap** directory on the primary machine. If one is not present, create the file.

9 Install **slurpd** on the primary machine:

```
cd c:\mercuryAM\openldap
slurpd install
```

10 Start Mercury Business Availability Center.

11 Start the LDAP server running on the secondary machine.

12 Verify that replication is successful:

- Add a template container to the enterprise.
- Connect to the secondary machine with an LDAP browser on port 9389, using the credentials: e=SSEnterprise password: fl1pp3r.
- Verify that the template container you added is present in the LDAP server on the secondary machine. There should be an LDAP node on the LDAP server for the object you created.

Modifying the Nanny Supervisor Process for Replicated Environments

On Mercury Business Availability Center machines running both **slapd** and the **slurpd** replication service, you should modify the files which control the Nanny supervisor process.

To modify the files controlling the Nanny supervisor process:

1 Modify the file **<Mercury Business Availability Center server root directory>\openldap\openldap_service_recover.bat**: add the following line:

```
net start openldap-slurpd
```

2 Create the batch file **openldap_stop**, add the following lines, and save it in the **openldap** directory:

```
net stop openldap-slapd
net stop openldap-slurpd
```

3 Modify the file **<Mercury Business Availability Center server root directory>\launch_service\dat\nanny\a_openldap.nanny**: Change the **stop_nt** line to run the **openldap_stop** batch file, to stop the OpenLDAP services. For example,

```
stop_nt="d:\MercuryAM/openldap/openldap_stop"
```

Starting and Stopping OpenLDAP

By default, OpenLDAP starts each time Mercury Business Availability Center is started. Furthermore, OpenLDAP performs a data integrity test automatically at startup. To prevent Mercury Business Availability Center from starting OpenLDAP automatically, you must remove the **a_openldap.nanny** file in the **<Mercury Business Availability Center server root directory>\launch_services\dat** directory, or you must remove its .nanny extension.

To stop and start OpenLDAP on a Windows platform, you perform the following procedure.

OpenLDAP Running on a Windows Platform

OpenLDAP is run as an NT service on Windows server machines. Even if Mercury Business Availability Center is stopped, OpenLDAP continues to run as a service, as it may be serving multiple Centers Server machines in the configuration. If the Centers Server is restarted and the service is already running, the service continues to run.

You can assess if the OpenLDAP service is running and other properties of the service by using the services management tool (**Start** > **Control Panel** > **Administrative Tools** > **Services**).

To stop and start OpenLDAP running on a Windows platform:

Issue the command:

```
net stop openldap-slapd
net start openldap-slapd
```

You can manually remove OpenLDAP as a service by changing directory:

```
cd \<Mercury Business Availability Center home directory>\openldap
```

and issuing the command:

```
slapd remove
```

The service can be manually reinstalled by entering:

```
slapd install
```

A registry key is required (and is automatically created at installation) to run OpenLDAP as a service on a non-standard (9389) port:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\OpenLDAP\Parameters]
"Urls"="ldap://:9389"
```

If the port is changed, the Mercury Business Availability Center configuration needs to be updated to reflect the change.

Backup and Restore Examples

This section contains the following examples:

- backup.bat
- restore.bat
- backupLDIF.bat
- restoreLDIF.bat

backup.bat

The following batch file can be run on a Windows system to back up a Monitor Administration LDAP database:

```
@echo on
rem back.bat
rem
rem This batch file will back up a Monitor Administration database
rem
if "%1"=="" goto usage
if not exist %1 md %1
xcopy bdb\*.* %1
goto done
:usage
echo backup [DATAFILE]
echo Where:
echo [DATAFILE] is the path of the file to back up to
echo Example: backup \\backups\mar0404\
:done
```

restore.bat

The following batch file can be run on a Windows system to restore a Monitor Administration LDAP database:

```
@echo on
rem restore.bat
rem
rem This batch file will copy a Monitor Administration database from a
rem previous backup.bat database copy
rem
if "%1"=="" goto usage
cd bdb
if not exist hold md hold
cd hold
echo Y | del *.*
cd ..
xcopy *.* hold
echo Y | del *.*
cd ..
xcopy %1 bdb
goto done
:usage
echo restore [DATAFILE]
echo Where:
echo [DATAFILE] is the path of the file to restore
echo Example: restore \\backups\mar0404.ldif
:done
```

backupLDIF.bat

The following batch file can be run on a Windows system to back up a Monitor Administration database to an LDAP Interchange Format textual file:

```
@echo off
rem back.bat
rem
rem This batch file will back up a Monitor Administration database to an
rem LDAP Interchange Format textual file
rem
if "%1"=="" goto usage
slapcat -f slapd.conf -b "E=SSEnterprise" -l %1
goto done
:usage
echo backup [DATAFILE]
echo Where:
echo [DATAFILE] is the path of the file to back up to
echo Example: backup \\backups\mar0404.ldif
:done
```

restoreLDIF.bat

The following batch file can be run on a Windows system to restore a Monitor Administration database from a previously created LDIF file:

```
@echo on
rem restore.bat
rem
rem This batch file will restore a Monitor Administration database from a
rem previous backup.bat created LDIF file
rem
if "%1"=="" goto usage
cd *ldbm
if not exist hold md hold
cd hold
echo Y | del *.*
cd ..
xcopy *.* hold
echo Y | del *.*
cd ..
slapadd -f slapd.conf -b "E=SSEnterprise" -l %1
goto done
:usage
echo restore [DATAFILE]
echo Where:
echo [DATAFILE] is the path of the file to restore
echo Example: restore \\backups\mar0404.ldif
:done
```

Index

A

alert file, Oracle Server 108
archive file, Oracle Server 109
authentication
    connecting to MS SQL databases using Windows authentication 32
    enabling Mercury Business Availability Center for Windows authentication 29

B

backing up databases
    MS SQL Server 35
    Oracle Server 120

C

checklist for support and certification
    MS SQL Server 63
    Oracle Server 138
CMDB database fragmentation, MS SQL Server 46
CMDB index fragmentation
    Oracle Server 117
CPU, Oracle Server 108

D

data loaders 175
    changing default logging level 179
    changing time frame for failed data 180
    DLQ cleanup utility 181
    folder locations 177
    managing data 175
    troubleshooting 177
data management 175
data partitioning 159
data placement, MS SQL Server 18
data purging 159
    parameter value calculations 161
    purging manager 160
data purging, setting parameter values 162
data storage 151
database configuration, MS SQL Server 22
database consolidation, advantages 149
database creation
    MS SQL Server 13
database distribution 150
database fragmentation, MS SQL Server 40
database integrity, MS SQL Server 39
database load behavior, Oracle Server 108
database maintenance
    MS SQL Server 35
    Oracle Server 107
database monitoring, MS SQL Server 50
database permissions, MS SQL Server 16
database properties, MS SQL Server 18
database requirements 3
database schema, verification 167
database users, creating 173
databases
    purging data from 159
    setting on Oracle Server 91
DBCC SHOWCONTIG 42
dbverify tool 168
deployment
    MS SQL Server 10
    Oracle Server 85
distribution of databases 150
distribution statistics, MS SQL Server 49

F

file groups, MS SQL Server 19, 151
file properties, MS SQL Server 18
file sizing, Oracle Server 132
fragmentation, MS SQL Server CMDB database 46
fragmentation, MS SQL Server database 40

H

hardware sizing
  MS SQL Server 59
  Oracle Server 127

I

index fragmentation, CDMB
  Oracle Server 117
installation
  MS SQL Server 69
  Oracle Client 104
instance, Oracle Server 88

L

LDAP
  backing up and restoring monitor administration configuration data 183
  backup and restore examples 194
  backup and restore methods 184
  backup.bat 194
  backupldif.bat 196
  basic backup and restore 185
  basic backup on replicated server and restore 186
  database location 184
  modifying the nanny supervisor process for replicated environments 191
  replication 189
  restore.bat 195
  restoreldif.bat 197
  starting and stopping openLDAP 192
LDIF, backup and restore 186
loaders. *See* data loaders
locally managed tablespace, Oracle Server 92
log level
  changing default for data loaders 179
log placement, MS SQL Server 18

M

management database
  creating on MS SQL Server 13
  definition 3
management user schema 95
MDAC
  MS SQL Server 11
  Oracle Client 105
  Oracle Server 105
Mercury Business Availability Center
  enabling for Windows authentication 29
Mercury Business Availability Center sizing
  MS SQL Server 58
  Oracle Server 125
Mercury Universal CMDB
  definition 4
MS SQL
  connecting to databases using Windows authentication 32
MS SQL Server
  adding data files 22
  backing up databases 35
  changing data file properties 22
  checklist for support and certification 63
  CMDB database fragmentation 46
  collecting statistics 38
  configuring 78
  configuring databases 22
  creating databases 13
  data placement 18
  data storage 151
  database fragmentation 40
  database integrity 39
  database maintenance 35
  database monitoring 50
  database permissions 16
  database properties 18
  deployment overview 10

distribution statistics 49
dropping data files 22
file groups 19
file properties 18
installing 69
log placement 18
modifying settings 65
permissions 16
sizing guidelines 57
system databases 21
system requirements 11
verifying settings 65
MS SQL Server installation
Authentication Mode dialog box 74
Collation Settings dialog box 75
Instance Name dialog box 69
Network Libraries dialog box 77
Select Components dialog box 72
Services Accounts dialog box 73
Setup Type dialog box 71

O

openLDAP
running on a Windows platform 192
Oracle
database environment 91
file sizing 132
instances 88
optimizing query performance 110, 112
Oracle alert file 108
Oracle Client
configuring for Application Management 105
installation 104
MDAC 105
system requirements 104
Oracle Recovery Manager (RMAN) 122
Oracle schema
privileges for Application Management 100
Oracle Server
backing up databases 120
checklist for support and certification 138
CMDB index fragmentation 117
collecting statistics for CMDB 112
collecting statistics for profile databases 110
CPU 108
data storage 151
database maintenance 107
deployment overview 85
editions 88
Input/Output 108
instances 88
parameter sizing 127
sizing guidelines 125
system requirements 87
tablespace maintenance 109
tablespaces 92

P

parameter sizing, Oracle Server 127
permissions, MS SQL Server 16
privileges, Application Management Oracle schema 100
profile databases
creating on MS SQL Server 13
definition 3
profile user schemas 98
purging data from database 159
Purging Manager 159

Q

query performance on Oracle Server, optimizing 110, 112

R

RMAN
Oracle Recovery Manager 122

S

server configuration options, MS SQL Server 60, 78
service configuration options, MS SQL Server 78
set management database utility 14, 95

SID 96, 105
sizing guidelines
MS SQL Server 57
Oracle Server 125
statistics
collecting for CMDB in Oracle Server 112
collecting for profile databases in Oracle Server 110
collecting in MS SQL Server 38
system databases, MS SQL Server 21
System Global Area, Oracle Server 108
system requirements
MS SQL Server 11
Oracle Client 104
Oracle Server 87

T

tablespaces, Oracle Server 92
creating when using the Purging Manager 165
creation 156
maintenance 109
tempdb database, MS SQL Server 21, 61
tnsnames.ora file 105
transaction log 17, 37, 38
troubleshooting, data loaders 177

U

user schema
management 95
profile 98
users, creating for database schema verification 173

V

verification
database schemas 167
MS SQL Server settings 65

W

Windows authentication
connecting to MS SQL databases 32
enabling Mercury Business Availability Center 29