Operations Manager i

Software Version: 10.60

OMi Database Guide

Document Release Date: January 2017
Software Release Date: January 2017

Hewlett Packard Enterprise

Legal Notices

Warranty

The only warranties for Hewlett Packard Enterprise products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Hewlett Packard Enterprise shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from Hewlett Packard Enterprise required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe® and Acrobat® are trademarks of Adobe Systems Incorporated.

AMD, the AMD Arrow symbol and ATI are trademarks of Advanced Micro Devices, Inc.

Citrix® and XenDesktop® are registered trademarks of Citrix Systems, Inc. and/or one more of its subsidiaries, and may be registered in the United States Patent and Trademark Office and in other countries.

Google™ and Google Maps™ are trademarks of Google Inc.

Intel®, Itanium®, Pentium®, and Intel® Xeon® are trademarks of Intel Corporation in the U.S. and other countries.

iPad® and iPhone® are trademarks of Apple Inc.

Java is a registered trademark of Oracle and/or its affiliates.

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

Microsoft®, Windows®, Lync®, Windows NT®, Windows® XP, Windows Vista® and Windows Server® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

NVIDIA® is a trademark and/or registered trademark of NVIDIA Corporation in the U.S. and other countries.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates.

Red Hat® is a registered trademark of Red Hat, Inc. in the United States and other countries.

SAP® is the trademark or registered trademark of SAP SE in Germany and in several other countries.

UNIX® is a registered trademark of The Open Group.

Documentation Updates

To check for recent updates or to verify that you are using the most recent edition of a document, go to: https://softwaresupport.hpe.com/.

This site requires that you register for an HPE Passport and to sign in. To register for an HPE Passport ID, click **Register** on the HPE Software Support site or click **Create an Account** on the HPE Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HPE sales representative for details.

Support

Visit the HPE Software Support site at: https://softwaresupport.hpe.com/.

Most of the support areas require that you register as an HPE Passport user and to sign in. Many also require a support contract. To register for an HPE Passport ID, click **Register** on the HPE Support site or click **Create an Account** on the HPE Passport login page.

To find more information about access levels, go to: https://softwaresupport.hpe.com/web/softwaresupport/access-levels.

HPE Software Solutions Now accesses the Solution and Integration Portal website. This site enables you to explore HPE product solutions to meet your business needs, includes a full list of integrations between HPE products, as well as a listing of ITIL processes. The URL for this website is https://softwaresupport.hpe.com/km/KM01702731.

Contents

Part I: Introducing the database environment

Chapter 1: Introduction to preparing the database environment

This chapter contains information on the types of databases used with HPE Operations Manager i (OMi).

Database overview

To work with OMi, you must set up the following databases:

- **Management.** For storage of system-wide and management-related metadata for the OMi environment. OMi requires one management database. You can create this database manually, or by using the configuration wizard.
- **Event.** For storage of events and related data, such as annotations, as well as for storage of configuration data, such as event correlation rules.
- **Run-time Service Model (RTSM).** For storage of configuration information that is gathered from the various OMi and third-party applications and tools. This information is used when building OMi views.

You can set up OMi databases on a Microsoft SQL Server, an Oracle Server, or a PostgreSQL Server, depending on the type of database server used in your organization:

- If you are working with a Microsoft SQL Server database, see "Deploying and maintaining the Microsoft SQL Server databases" on page 14.
- If you are working with an Oracle Server database, see "Deploying and Maintaining the Oracle Server database" on page 56.
- If you are working with a PostgreSQL Server database, see "Deploying and maintaining the PostgreSQL Server databases" on page 93.

> **Note:** Database servers must be set to the same time zone, daylight savings settings, and time as OMi servers.

System requirements

The following table describes the hardware (CPU and memory) requirements recommended for the OMi database server:

Deployment size	Number of processors	Physical memory
Small (up to 2,000 nodes)	2 CPU cores	Minimum: 2 GB RAM Recommended: 4 GB RAM
Medium (up to 5,000 nodes)	2 CPU cores	Minimum: 2 GB RAM Recommended: 6 GB RAM
Large (more than 5,000 nodes)	Minimum 4 CPU cores	Minimum: 4 GB RAM Recommended: 8 GB RAM and up

For the database hardware requirements, see the relevant documentation available at:

- Microsoft SQL: https://msdn.microsoft.com/en-us/library/mt590198
- Oracle: http://otn.oracle.com/documentation/index.html
- PostgreSQL: http://www.postgresql.org/docs/manuals

For software requirements, see Support Matrices for Operations Center products.

Database character set configuration

When creating a database and specifying a character set, which determines what languages can be stored and represented in the database and OMi, make sure that you choose the one that will support the characters you need in your OMi environment. You must take special care when using languages with non-Latin characters or working in a multilingual environment that requires storing and processing data in several languages simultaneously.

For successful communication, both OMi and the database must use a compatible character set. Otherwise, data is not stored correctly and errors occur. In addition, using the same character set increases the performance because the need for character conversion is reduced or eliminated. For detailed information about considerations and current limitations when using non-Latin characters, see the *OMi Administration Guide*.

> **Note:** You can use the Oracle and PostgreSQL server databases for storing multilingual data in OMi. However, with the current implementation, it is not possible for OMi to store multilingual data in the Microsoft SQL Server database—only one character set can be used at a time.

Depending on the type of database server, see one of the following sections:

- "Character set support in Microsoft SQL Server" on page 15
- "Character set support in Oracle" on page 58
- "Character set support in PostgreSQL" on page 96

Setting up TLS for database connections

To set up TLS for database connections, you must perform the following tasks:

- Task 1: "Checking requirements" below
- Task 2: "Configuring the database for TLS" on the next page
- Task 3: "Securing the connection from OMi to the database" on the next page

Checking requirements

Before you start setting up TLS for database connections, consider the following requirements:

- The database endpoint must provide a valid X.509 server certificate for establishing a TLS connection.

  > **Note:** The client does not use the server hostname to check the server's identity.

- *Microsoft SQL Server only:* If you want to use TLS 1.2 to connect to a server that is running Microsoft SQL Server 2014 or 2012, make sure to use one of the following updates of SQL Server:
  - SQL Server 2014 SP1 CU1
  - SQL Server 2014 RTM CU8
  - SQL Server 2012 SP3 CU1
- *FIPS mode only:* When using HTTPS communication between OMi and the database, you cannot use the Retrieve certificate option of the OMi configuration wizard. You can import the database

server certificate only from the file by browsing for it. For details about OMi in FIPS mode, see the *OMi FIPS Configuration Guide*.

Configuring the database for TLS

For details on configuring the database for TLS, see the corresponding database documentation.

Securing the connection from OMi to the database

The OMi configuration wizard enables you to configure OMi to establish a secure connection to the database.

After you select the database that you want to use with OMi and provide the required information related to the database setup in the Database Settings page of the OMi configuration wizard, select the **Use TLS** check box to secure the connection from OMi to the database.

> **Caution:** When the Use TLS check box is selected, the port for the database does not change automatically. Therefore, make sure to set the value of the Port field to the TLS port number.

Click **Next** in the Database Settings page. You are prompted to either retrieve the database server certificate automatically or to browse for it. After you retrieve or upload the certificate, the certificate information is displayed. When you click Next, it is assumed that you trust the certificate.

In addition, you can also test the connection to the database by clicking the **Test Connection** button.

Part II: Deploying and maintaining the Microsoft SQL Server databases

Chapter 2: Microsoft SQL Server deployment overview

You can use Microsoft SQL Server for deploying OMi databases. This chapter describes the following topics related to deploying Microsoft SQL Servers for use with OMi:

- "About Microsoft SQL Server deployment" below
- "Character set support in Microsoft SQL Server" below

About Microsoft SQL Server deployment

To deploy Microsoft SQL Server for use with OMi, perform the following procedures:

- **Install and configure Microsoft SQL Server.**

  For details on installing and configuring Microsoft SQL Server, see "Installing and configuring Microsoft SQL Server" on page 21.

- **Create databases on Microsoft SQL Server.**

  You can create OMi databases manually or you can use the configuration wizard to create the databases.

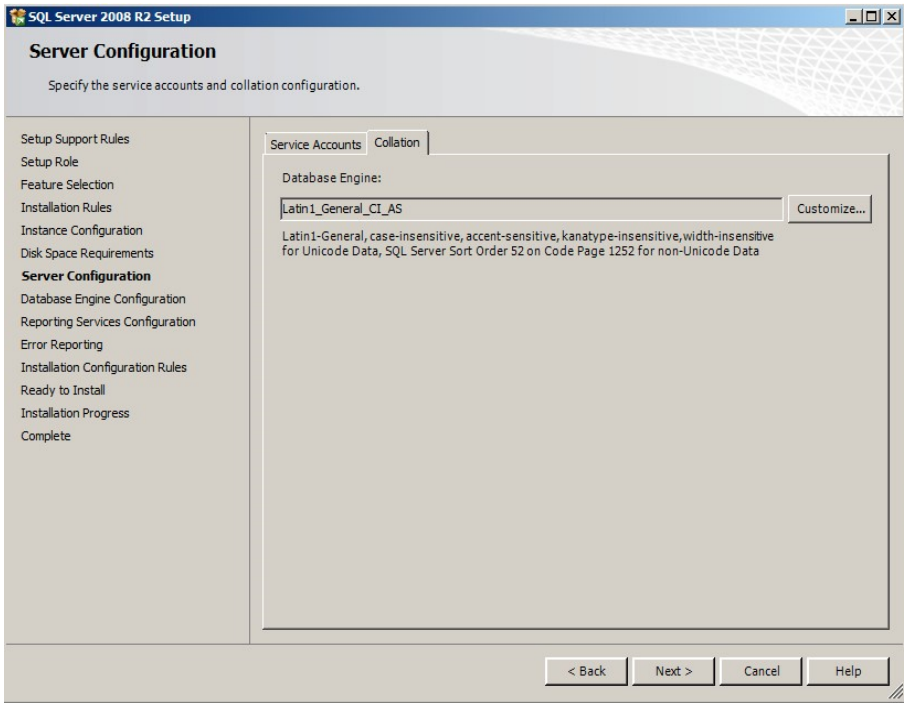
  For details on manually creating databases for OMi, see "Manually creating the OMi databases" on page 17.

  For details on creating Microsoft SQL database, see "Creating and configuring Microsoft SQL Server databases" on page 31.

Character set support in Microsoft SQL Server

Because Microsoft SQL Server does not support UTF-8 encoding (although data can be stored internally in the `NVARCHAR` fields as UCS or UTF-16), OMi cannot store multilingual data if a Microsoft SQL Server database is used. It is still, however, possible to use one or more languages with non-Latin characters, but they must share the same character set. For this, you must specify a collation that is in accordance with the OMi language and machine's regional settings.

During the database installation, the setup tool detects the current regional settings and suggests the most suitable collation.



The following table lists languages and their default collations:

Language	Default Collation
English	Latin1_General_CI_AS
French	French_CI_AS
Spanish	Modern_Spanish_CI_AS
German	Latin1_General_CI_AS
Russian	Cyrillic_General_CI_AS
Japanese	Japanese_CI_AS
Korean	Korean_Wansung_CI_AS
Simplified Chinese	Chinese_PRC_CI_AS

Chapter 3: Manually creating the Microsoft SQL Server database

This chapter describes the procedure for manually creating OMi databases on a Microsoft SQL Server.

This chapter includes:

Microsoft SQL Server database overview

OMi uses the Management, RTSM, and Event databases.

During the OMi setup, new databases can be set up automatically by the setup procedure, or already existing databases can be used. Existing databases can either be created manually in advance (for example, due to organization security restrictions), or can be created by a previous installation of the same release of OMi.

For details about installing OMi, see the *OMi Installation and Upgrade Guide*.

Manually creating the OMi databases

After you install and configure Microsoft SQL Server, you create OMi databases for the storage of data collected by OMi.

You can also allow OMi to create OMi databases for you by using the configuration wizard. If you created the databases in advance, you can connect OMi to the existing databases from the configuration wizard. For information on the configuration wizard, see the *OMi Installation and Upgrade Guide*.

This section includes the following topics:

> **Note:** In a distributed environment, make sure to run all generation scripts from the Data Processing Server.

Creating databases

The database administrator should create the following Microsoft SQL Server databases: one for Management, one for RTSM, and one for Event.

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.

For details on creating the databases manually, see "Creating and configuring Microsoft SQL Server databases" on page 31.

> **Note:** When the OMi setup procedure automatically creates the databases, it uses two file groups for each database, one to hold the system tables and one for the application data.

Creating objects

After creating each OMi database, you create objects by running SQL scripts.

For the Management database, the SQL script is dynamically created by the `opr-generate-sql-script` utility:

`<OMi_HOME>/bin/opr-generate-sql-script.[sh|bat]`

Creating Management objects

To create the OMi Management objects, follow the procedure below to generate and run the Management database SQL script:

To generate the Management database SQL script:

Run the following script file located in `<OMi_HOME>/bin` directory:

`opr-generate-sql-script.[sh|bat]`

The following parameters should be used for the Management database:

- `--create`: Generate a database creation script.
- `--db_type`: `SQLServer`
- `--db_schema`: `Management`
- `--out_file`: The name of the SQL file to generate, for example `sqlsvr.sql`

Example:

```
/opt/HP/BSM/bin/opr-generate-sql-script.sh --create --db_type SqlServer
--db_schema Management --out_file sqlsvr.sql
```

To run the Management database SQL script:

1. Create the Management database. For details, see "Creating databases" on the previous page.
2. Connect to the Management database and run the script generated in the previous section to deploy the Management objects.

Creating RTSM objects

Create one database for storing the RTSM objects. Then connect to the RTSM database and run the following script:

`<OMi_HOME>\conf\sql\sqlserver\rtsm\rtsm_set_db_parameter.sql`

Creating Event objects

Create the Event database for storing Event objects. Follow the procedure below to create Event objects:

To generate the Event database SQL script:

Run one of the following script files:

For Windows:

`<OMi_HOME>\bin\opr-generate-sql-script.bat`

For Linux:

`<OMi_HOME>/bin/opr-generate-sql-script.sh`

Use the following parameters for the Event database:

- `--create`: Generate a database creation script.
- `--db_type`: `SQLServer`
- `--db_schema`: `Event`
- `--out_file`: The name of the SQL file to generate, for example `sqlsvr.sql`

Example:

```
/opt/HP/BSM/bin/opr-generate-sql-script.sh --create --db_type SqlServer
--db_schema Event --out_file sqlsvr.sql
```

To run the Event database SQL script:

1. Create the Event database. For details, see "Creating databases" on page 18.
2. Connect to the Event database and run the script generated in the previous section to deploy the Operations objects.

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

Chapter 4: Installing and configuring Microsoft SQL Server

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

This chapter includes:

Workflow for Microsoft SQL Server deployment

To deploy Microsoft SQL Server for use with OMi, perform the following steps:

1. **Review Microsoft SQL Server sizing guidelines**

   For details, see "Database file layout" on page 34.

2. **Review installation prerequisites**

   For details, see "Installation prerequisites" on the next page.

3. **Review server and database checklists**

   These checklists summarize the server and database configuration options supported and recommended for OMi.

   For details, see "Checklist for support and certification" on page 23.

4. **Install Microsoft SQL Server Standard or Enterprise edition**

   For details, see "Installing Microsoft SQL Server" on page 24.

5. **Configure Microsoft SQL Server**

   For details, see "Configuring Microsoft SQL Server" on page 26.

6. **Create OMi databases on Microsoft SQL Server**

   For details, see "Creating databases" on page 18.

7. **Configure OMi databases**

   For details, see "Configuring databases" on page 37.

8. **Verify Microsoft SQL Server and databases**

   For details, see "Verifying and modifying server and database settings" on page 27.

9. **Set up Windows authentication**

   For details, see "Using Windows authentication to access Microsoft SQL Server databases" on page 53. This step is only relevant if you have selected Windows authentication instead of SQL Server authentication.

Installation prerequisites

Make sure that the following installation prerequisites are met:

- **System Requirements**

  For a list of supported operating systems, see "System requirements" on page 11.

- **Software**

Software	Version needed
Microsoft Windows Installer	4.5 or later

- **Database file placement**

  For improved performance, it is recommended to spread out OMi databases among several disks or RAID systems.

- **Memory**

  There must be at least 2 GB of RAM.

- **User Accounts**

  - If you plan to perform activities outside the local machine (such as file copying to or from different servers, backing up to a shared remote repository, and so forth), you must have a domain account available for Microsoft SQL Server services.
  - If you want to install OMi by using the installation wizard, you must provide a user account that has database creator privileges. If you are going to create databases manually, supply OMi with a user account that is a part of **db_datareader**, **db_datawriter** and **db_ddladmin** in each OMi database.

- **Miscellaneous**
  - Verify that the disk where Microsoft SQL Server is to be installed is not compressed.
  - Disable antivirus software and any other applications that may interfere while installing Microsoft SQL Server.

Notes and limitations

- Microsoft SQL Server must be installed on a dedicated machine to host the OMi databases.
- Do not change the default memory settings. Allow the Microsoft SQL Server to manage memory dynamically, except when you configure **awe enabled** support. The AWE feature was removed from SQL Server 2012, and is not recommended for use starting SQL Server 2008 R2.
- Microsoft SQL Server 64-bit versions include support for extended systems, also known as Windows on Windows (WOW64). WOW64 is a feature of 64-bit editions of Microsoft Windows that allows 32-bit applications to execute natively in 32-bit mode.
- Applications function in 32-bit mode even though the underlying operating system is running on the 64-bit platform.
- When working with Microsoft SQL Server, the RTSM collation and the SQL Server collation must be the same in the old server and new server for an upgrade to run properly. For more information, see "Database character set configuration" on page 11.

Checklist for support and certification

The information provided in this section is for both supported and certified Microsoft SQL Server options. The certified options are recommended for working with OMi.

Subject	Microsoft SQL Server	
	Supported	**Recommended**
Editions	Standard, Enterprise	
Instances	Default, Single	
Authentication Mode	Mixed, Windows	
Collation	Case-Insensitive. OMi does not support binary sort order and case sensitivity. Only case-insensitive order with a combination of accent, kana, or width settings is supported.	Use the Collation Settings dialog box to select the collation. Do not select the binary check box. Accent, kana, and width sensitivity should be selected according to the relevant data language requirements. The selected language must be the same as the OS Windows regional settings language.
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

Installing Microsoft SQL Server

It is important that you familiarize yourself with all of the installation details so that you select the appropriate options. Selecting the default options may, in some cases, negatively affect the Microsoft SQL Server's performance.

Select the following options in the installation dialog boxes:

- **Feature Selection dialog box configuration.**
  - Remove Full Text Search from the list since OMi does not use this indexing search feature.
  - Under Destination Folder, make sure that the Data Files directory is stored on a fault-tolerant disk system, for example, RAID 1. Even though these system databases are fairly small, they are essential for the operation of Microsoft SQL Server.
- **Instance Name dialog box configuration.** For details on named instances, see "Port allocation options for named instances" on page 29. You access a default instance by specifying the server name or IP address.
- **Service Account dialog box configuration.**
  - If all Microsoft SQL Server activities are outside the local machine (for example, file copying to or from a different server, backing up to shared remote repository, replication with other servers, ActiveX script job steps, CmdExec job steps, and so forth), choose **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 Microsoft SQL Server activities are limited to the local machine, choose **Use the built-in System account** and select **Local system**. This selection provides SQL Server administrative privileges on the local machine only.
- **Authentication Mode dialog box.** OMi works with both Windows authentication and with Microsoft SQL Server authentication (recommended). To enable Microsoft SQL Server authentication, do the following:

  a. Choose **Mixed Mode (Windows Authentication and SQL Server Authentication)**.

  b. Enter the password for user **sa**.

  > **Note:** To further secure your Microsoft SQL Server, it is important to enter a password.

- **Collation Settings dialog box.** Follow the recommendations found in the checklist in "Checklist for support and certification" on page 23.

  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, but this is not recommended for dedicated OMi databases. Because of the flexibility in collation management in Microsoft SQL Server, 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 Microsoft 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.

- **Data Directories dialog box.** Specify the destination folders for the data and log files of the user databases and temporary databases, as well as the backup directory. It is recommended to store data and log files on high-performance storage systems.

Install the latest service pack for Microsoft SQL Server available at the Microsoft Download Center site:

http://www.microsoft.com/downloads/details.aspx?FamilyID=cb6c71ea-d649-47ff-9176-e7cac58fd4bc&DisplayLang=en

Configuring Microsoft SQL Server

This section describes the service and server options you can configure once you have installed Microsoft SQL Server and includes the following topics:

- "Service configuration options" below
- "Server configuration options" on the next page

Service configuration options

If you installed **Full-Text Search**, ensure that it is disabled and set to manual mode (locate the service in the Services applet by using Microsoft Search) so that no resources are wasted.

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

If the dynamic port option is used for Microsoft SQL Server instances, make sure the SQL Server browser service, as well as all the SQL Server instance services, are in automatic mode.

Server configuration options

Most server configuration options are dynamically configured by Microsoft SQL Server. For OMi certification, you may not change the default options unless you are instructed to do so by HPE Software 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 Microsoft SQL Management Studio (mainly the Server Properties dialog box).

The default values should be used except for the following items:

- Agent XPs (A) = 1
- awe enabled (A) = 1. This should only be set when more than 4GB is required on a 32Bit machine. Note that this feature is deprecated in Microsoft SQL Server 2012.

For a full list of Microsoft SQL Server configuration options, see "Microsoft SQL Server configuration options" on page 115.

Verifying and modifying server and database settings

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

Server or database setting	How to verify or modify the setting
Default Instance	In the operating system's Services applet, a default Microsoft SQL Server instance and a named instance appear as SQL Server (Instance_Name).
Authentication Mode	In the Microsoft SQL Server Management Studio, right-click the server, choose **Properties**, and click the **Security** tab. Select **SQL Server and Windows Authentication mode (Mixed Mode)**.
Collation Settings	Run the following command: `sp_helpsort`
Network Libraries	On the Server, select **Start > Programs > Microsoft SQL**

Server or database setting	How to verify or modify the setting
	Server <release number> > Configuration Tools > Configuration Manager. At **SQL Native Client**, choose **client protocols** and ensure the selected protocol is in the enabled state. • Supported Shared memory, TCP/IP ,and Named Pipes for both the server and client • Recommended Only TCP/IP for both the server and client
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 Microsoft SQL Server service. For more details, see Microsoft SQL Server documentation.

Check whether the user who is going to create OMi databases has CREATE DATABASE permissions	Log in to the Microsoft SQL Server Management Studio 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 OMi database user has enough permissions in the database	• Log in to the Microsoft SQL Server Management Studio with the user name you want to check.
	
	• Change the context of the database to the required database.
	
	• Open a new query and do the following in each database:
	
	`select case when IS_MEMBER ('db_owner')=1`
	`or IS_SRVROLEMEMBER ('sysadmin')=1`
	`or (IS_MEMBER ('db_ddladmin') = 1 and`
	`IS_MEMBER ('db_datareader')=1 and`
	`IS_MEMBER ('db_datawriter')=1 and`
	`IS_MEMBER ('db_denydatareader')=0 and`
	`IS_MEMBER ('db_denydatawriter')=0 )`
	`then 'User has enough permissions'`

Server or database setting	How to verify or modify the setting
	`else 'User does not have enough permissions'` `end`
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.

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, run:
	`ALTER DATABASE <database name> SET <option> <value>`
	• To change the database file properties, run:
	`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.

Microsoft SQL Server service pack version and edition	Log into Management Studio and make the following query: `select @@version`.

Port allocation options for named instances

When working with a named instance in SQL Server, there are two options for the instance's port allocation: dynamic port and static port.

- **Dynamic Port.** When working with a dynamic port, a new port is assigned to the instance each time the instance is started. To enable clients to know the port when connecting to the instance, a service called SQL Server Browser needs to be started. The browser service listens to port 1434 and directs clients to the correct port according to the desired instance name.

  You access the named instance by specifying the server name or IP address followed by **\<instance_name>, for example, server1\inst1**.

- **Static Port.** When working with a static port, a specific port is assigned to the instance, and SQL Server Browser service is not needed in order to connect to the instance (although you can still use it).

The default for SQL Server is a dynamic port.

To work with a static port, follow the following steps:

1. Select **Start > Programs > Microsoft SQL Server > Configuration Tools > SQL Server Configuration Manager**.
2. Expand Protocols for <instance name>, and double-click **TCP/IP**.
3. In the TCP/IP Properties dialog box, on the IP Addresses tab, several IP addresses appear, in the format IP1, IP2, up to IPAll. For each address:
   a. If the TCP Dynamic Ports dialog box contains 0, indicating that the database engine is listening on dynamic ports, delete 0.
   b. In the TCP Port box, enter the port number on which you want this IP address to listen, and click **OK**.
4. In the console pane, click SQL Server Services.
5. In the details pane, right-click SQL Server (<instance name>) and click **Restart** to stop and restart SQL Server.

You access the named instance by specifying the server name or IP address followed by the port number, for example, **server1\1435**.

Chapter 5: Creating and configuring Microsoft SQL Server databases

This chapter describes the creation and configuration of OMi databases on a Microsoft SQL Server.

This chapter includes:

Creating databases

This section includes the following topics:

Creating Microsoft SQL server named instances

Before creating Microsoft SQL Server Named Instances, Microsoft SQL Server must be already installed on the system. For more information, see "Installing Microsoft SQL Server".

To create a Microsoft SQL Server Named Instance, follow these steps:

1. Log in as an administrator.
2. Run the SQL Server installation files from the SQL server installation media (DVD).
3. Select **Installation**.
4. Select **New SQL Server stand-alone installation or add features to an existing installation**.
5. Click **OK**. The installation will search for the latest updates. You may choose to skip and check for updates later.

6. Click **Next**. An error message may appear stating that the Windows firewall is active.

   You can enable the ports for remote database access later.

7. Select **Perform a new installation of SQL Server 2012** and click **Next**.
8. Type the product key and click **Next**.
9. Accept the license agreement and click **Next**.
10. Select all features with defaults and click **Next**.
11. In the Feature Selection page, remove Full Text Search from the list as OMi does not use the indexing search feature. Click **Next**.
12. In the Installation Rules page, click **Next**.
13. Select the following:

    Named Instance,

    Instance name: <any name>

    Instance ID

    > **Note:** The Instance ID creates automatically when you click the field.

    Instance root directory: <path>

14. Click **Next**

    > **Note:** Do not make any changes in the disk space requirements and the server configuration pages.

15. In the Database Engine Configuration page, select Mixed Mode (SQL Server authentication and Windows authentication) and specify the following:
    a. Specify the password for the SA user.
    b. Specify SQL Server Administrators.
16. Click **Next**
17. In the Analysis Service Configuration page, specify the Administrator with required privileges for Analysis Services.
18. In the Reporting Services Configuration page, select the following:
    a. Reporting Services Native Mode
    b. Install only (reporting services configuration is out of scope)

c. Reporting Services Sharepoint: Keep the default selection

19. Click **Next**.
20. Specify the administrators with privileges for the Distributed Replay Controller. Click **Next**
21. In the Distributed Replay Client page, type the name of the controller that the client computer will communicate with for the Distributed Replay client service.

> **Note:** The name must be a fully qualified domain name (FQDN). For example, a host called server1 in the products hierarchy at Microsoft may have an FQDN of **server1.products.microsoft.com**. If you have already set up the controller, specify the name of the controller while configuring each client.

> **Note:** If you have not set up the controller, you do not have to specify the controller name. However, you must manually specify the controller name in the client configuration file.

For the Working Directory and Result Directory fields, you can either keep the default path or choose a new path.

22. In the Error Reporting page, select the option if you want to send error reports to Microsoft.
23. In the Installation Configuration Rules page, click **Next**.
24. In the Ready to Install page, review the chosen configuration. After a few seconds, the summary page appears with the complete message. Click **Close**
25. Close the SQL Server installation center.

Database permissions

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

> **Note:** OMi log-in accounts should be mapped to **dbo** in the database. 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 check whether a user has CREATE DATABASE permissions, log in to Management Studio with the log-in 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.'
```

To check whether a user has enough permissions in the database, log in to Management Studio with the log-in account of the user whose mapping you want to check. Change the database context to the required database, and run the following command:

```
select case when IS_MEMBER ('db_owner')=1
or IS_SRVROLEMEMBER ('sysadmin')=1
or (IS_MEMBER ('db_ddladmin') = 1 and
IS_MEMBER ('db_datareader')=1 and
IS_MEMBER ('db_datawriter')=1 and
IS_MEMBER ('db_denydatareader')=0 and
IS_MEMBER ('db_denydatawriter')=0 )
then 'User has enough permissions'
else 'User does not have enough permissions'
end
```

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 can create several data files. Microsoft 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

> **Note:** It is recommended that you place data and log files on separate disk subsystems, and not on the same disk that stores the page (swap) file.

- **Log files.** Changes are not flushed to the database until they are written to the log, and the log architecture dictates serial writes. Therefore, 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 1+0 (striped mirror) system.
- **Data files.** Data files should be placed on a RAID 1+0 system for optimal performance.

File and database properties

When you create a database you can specify the following 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.

  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. Using a small growth increment is not recommended because it increases file system fragmentation. On the other hand, if you use a very large increment, modifications sent by clients might incur connection timeouts while waiting for the automatic expansion to finish. For large databases, a percentage growth increment can lead to exponential growth of the database and should be avoided.

tempdb database settings

The frequent expansion of the tempdb system database can affect the database's performance, especially in large Microsoft SQL Server installations. The size of the tempdb, therefore, should be large enough to avoid the need for early expansion. The growth increment of the tempdb 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 1 GB and with a growth increment of 50 MB. The tempdb database should be striped across several disks, ideally on a RAID 1+0 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.

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, see Microsoft SQL Server Books Online at http://www.microsoft.com/downloads.

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

- **Partial Restoring.** Microsoft SQL Server 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. Microsoft SQL Server 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 restore a single table to a point in time if a logical error occurs, you must 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.

For additional data on creating file groups, see "Microsoft SQL Server data storage recommendation" on page 112.

System databases

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

- **tempdb.** Numerous Microsoft 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 "tempdb database settings" on page 35.

  To move tempdb's files, use the **ALTER DATABASE** tempdb **MODIFY FILE** command, and restart Microsoft SQL Server.

- **master, msdb, model.** These databases, although crucial for the operation of Microsoft 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 OMi certification, place system databases on fault tolerant disks. It is recommended to use RAID 1 disks.

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

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

Configuring databases

After 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.

This section includes:

- "Database file configuration" on the next page
- "Database configuration options" on page 39

> **Note:** If you enabled OMi to create OMi databases for you, it is strongly recommended that you configure them following the configuration instructions in this section.

Database file configuration

You can change certain database file properties, as well as add or drop files by using either of the following methods:

- Use the Properties dialog box in Management Studio.
- Use the `ALTER DATABASE` command (for details, see Microsoft SQL Server 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 by 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 Microsoft SQL Server). The `SIZE`, `MAXSIZE`, and `FILEGROWTH` properties can be changed by 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 31.

Database configuration options

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

- The Options tab in the Management Studio's Properties dialog box
- The `EXEC sp_dboptions` stored procedure
- The `ALTER DATABASE <database name> SET` command

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

The following table lists, in alphabetical order, the default configuration options, as well as the configuration settings required for OMi certification:

Configuration option	Description	Default	OMi certification in Microsoft SQL Server
ANSI NULL default (see note below)	Specifies whether the database columns are defined as NULL or NOT NULL, by default.	Not set	Not set
ANSI PADDING	Controls the way the column stores values shorter than the defined size of the column and the way the column stores values that have trailing blanks in char, varchar, binary, and varbinary data.	OFF	ON
Auto close	Specifies whether the database shuts down after its resources are freed and all users exit.	Not set	Not set 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 create statistics	Specifies whether missing statistics required by a query for optimization are built automatically during optimization.	Set	Set
Auto shrink	Specifies whether the database is automatically shrunk every hour, leaving 25% of free space.	Not set	Not set **Note:** If set, constant

Configuration option	Description	Default	OMi certification in Microsoft SQL Server
			growth/ shrinkage may cause file system fragmentation.
Auto update statistics	Specifies whether out-of-date statistics required by a query for optimization are built automatically during optimization.	Set	Set
Compatibility level	The version of Microsoft SQL Server that the database appears to be (for the application).	The same version as the release installed	The same version as the release installed
Read only	The 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)
Recursive triggers	Specifies whether recursive triggers are supported.	Not set	Not set
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
Page verify	Specifies whether incomplete pages can be detected.	Set	Set
Quoted identifiers enabled	Specifies whether the Microsoft 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

> **Note:** Not all ANSI options can be set by using Management Studio. The ANSI database configuration options include: ANSI_NULLS, ANSI_NULL_DEFAULT, ANSI_PADDING, ANSI_WARNINGS, ARITHABORT, CONCAT_NULL_YIELDS_NULL, NUMERIC_ROUNDABORT, and QUOTED_IDENTIFIER.
>
> 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 command:

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

For information on SQL databases, see Microsoft SQL Server Books Online.

Chapter 6: Maintaining Microsoft SQL Server databases

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

This chapter includes:

Backing up databases

Microsoft SQL Server supports three main types of database backup: full, differential, and log. It also supports backup of files and file groups, 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 by using Microsoft SQL Agent jobs. The Microsoft SQL Agent (represented by the SQLServerAgent service) is installed automatically when you install Microsoft SQL Server. Ensure that the Microsoft 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 may 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 Microsoft SQL Server service account needs permission to access) or tape (only local).

This section describes:

Full backup

When you perform a full database backup, all information about the backed up database is contained within the backup, including data, metadata, 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, 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 eight 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.

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 45.

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 Microsoft 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 and 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 Microsoft SQL Server Management Studio, 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, Microsoft 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 inevitably leads 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 command:

```
KILL <process id>
```

> **Note:** In Microsoft SQL Server, 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.

This section describes:

- "Collecting statistics for RTSM" on page 51
- "Utility to refresh statistics" on page 51

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 by using Microsoft 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) by using the `PHYSICAL_ONLY` option.

Because the Microsoft SQL Server 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. 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, to avoid small, fragmented parts, use a large value as the file growth increment. Do not use too large a value, however, because this leads to client request timeouts when the file autogrows (for more details, see "Database permissions" on page 33). 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 on the database.

Understanding internal fragmentation

Internal fragmentation refers to the percentage of data contained in the pages. In environments such as the OMi 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 `ALTER INDEX REBUILD` command (online or offline) and `FILLFACTOR` option. 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 by using the `FILLFACTOR` option. You can also handle external fragmentation by checking the external fragmentation status of your indexes, and rebuilding the indexes.

Detecting and handling index fragmentation

Using sys.dm_db_index_physical_stats to detect fragmentation

The dynamic management function **sys.dm_db_index_physical_stats** is used to determine the degree of fragmentation of an index. You can detect fragmentation in a specific index, in all indexes on a table or indexed view, in all indexes in a specific database, or in all indexes in all databases. For partitioned indexes, **sys.dm_db_index_physical_stats** also provides fragmentation information for each partition.

Fragmentation of a table occurs through the process of data modifications (`INSERT`, `UPDATE`, and `DELETE` statements) that are made against the table and to the indexes defined on the table. Because these modifications are not ordinarily distributed equally among the rows of the table and indexes, the fullness of each page can vary over time. For queries that scan part or all of the indexes of a table, this kind of fragmentation can cause queries to return slower.

The table syntax of `sys.dm_db_index_physical_stats` is:

```
sys.dm_db_index_physical_stats (
{ database_id | NULL }
, { object_id | NULL }
, { index_id | NULL | 0 }
, { partition_number | NULL }
, { mode | NULL | DEFAULT }
)
```

Run this command as regular select statement from table, for example:

```
select * from sys.dm_db_index_physical_stats (DB_ID('<OMi_database>'),
object_id('<Table_Name>'), NULL, NULL, 'SAMPLED')
```

The **Avg_fragmentation_in_percent** column returned by sys.dm_db_index_physical_stats is the logical and extent fragmentation of the index.

For more information about **sys.dm_db_index_physical_stats**, see the Microsoft SQL Server documentation.

Handling fragmentation

The table column **Avg_fragmentation_in_percent** returned by **sys.dm_db_index_physical_stats** reflects the degree of fragmentation per index. Based on this value, you can determine whether to

handle the fragmentation and which method to use in handling it.

Use the following rough guidelines to determine the best method to correct the fragmentation:

- **Between 5% and 30%.** Use the `ALTER INDEX REORGANIZE` command to reorganize the index. Index reorganization is always executed online.
- **Greater than 30%.** Use the `ALTER INDEX REBUILD` command to rebuild the index. Index rebuilding can be executed online or offline. Rebuild the indexes online to achieve the availability similar to that of the Reorganize option.

Very low levels of fragmentation (less than 5%) may not be addressed by either of these commands due to the negligible benefit gained from removing such a small amount of fragmentation.

For more information on defragmenting indexes, see "Reorganizing and Rebuilding Indexes" under Designing and Implementing Structured Storage in SQL Server Books Online: http://technet.microsoft.com/en-us/library/ms189858.aspx.

> **Note:** It is strongly recommended that you create an automatic index rebuild task for each of the OMi databases. In the RTSM and event databases, run this task at least once a day.

Supplied utilities for monitoring and rebuilding indexes

OMi provides two utilities that can be used to detect and rebuild fragmented indexes. The **rebuild_fragmented_indexes.bat** utility uses 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 usually hinders 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 database administrator.

The utilities for Microsoft SQL Server are located in the `<OMi_HOME>\opr\support\database\sqlserver` directory on the OMi server.

Utility to rebuild all indexes in a database

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

To run the rebuild_indexes.bat utility:

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

- SQL Server name
- Database name
- SA password

Example:

```
rebuild_indexes.bat SQL_SRVR_3 OMi_DB_3 ad%min52.
```

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.

Example:

```
rebuild_fragmented_indexes.bat SQL_SRVR_3 OMi_DB_3 ad%min52 1
```

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.

Distribution statistics

Microsoft SQL Server 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 columns. 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 daily 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.

Whether the auto update statistics database option is enabled or disabled, it is strongly recommended that you create an automatic task to update statistics for each of the OMi databases on a daily basis, as the data is frequently changed. The job should execute the sp_updatestats API against the specific database.

Collecting statistics for RTSM

Unlike some databases where queries are predefined and can be modified according to the expected database size, the RTSM database constructs queries dynamically, according to the pattern views defined against its data model. This necessitates accurate statistics at all times. In addition to running a daily job to update statistics, it is recommended that you manually refresh statistics if major changes to the RTSM schema objects have occurred, usually as a result of bulk insert transactions.

An example of a scenario that warrants a manual refresh of the RTSM statistics is automated Data Flow Management (DFM) jobs. DFM is the process responsible for automatically detecting configuration items and inserting them into the RTSM.

Utility to refresh statistics

The `update_statistics.bat` utility has two working modes:

- List tables with out of date statistics. In this mode, a list of the tables is returned, together with the commands needed to update statistics the tables at a later time.
- Update statistics on the tables. In this mode, all tables with outdated statistics are being updated.

To run the `update_statistics.bat` utility, from `<OMi_HOME>\opr\support\database\sqlserver`, execute `update_statistics.bat` with the following parameters:

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

Example:

```
update_statistics.bat SQL_SRVR_3 RTSM_DB_3 ad%min52 1
```

Output from the procedure (a list of tables and the update commands) is located in the `update_statistics.log` file in the same directory.

Chapter 7: Using Windows authentication to access Microsoft SQL Server databases

Unless configured otherwise, OMi uses Microsoft SQL Server authentication to access Microsoft SQL Server databases. However, Windows authentication can also be used.

This chapter describes how to enable OMi to use Windows authentication to access Microsoft SQL Server databases.

This chapter includes:

"Enabling OMi to work with Windows authentication" below

Enabling OMi to work with Windows authentication

You can enable OMi to use Windows authentication instead of Microsoft SQL Server authentication to access any of the OMi databases.

To enable OMi to use Windows authentication to access a Microsoft SQL database, you must:

- Configure the Microsoft SQL Server to use Windows authentication.
- Launch the OMi Server service on all the OMi servers with a Windows user that has the necessary permissions to access the Microsoft SQL database.
- Use the configuration wizard to create or connect to a Microsoft SQL database and specify the use of Windows authentication.

This section includes the following topics:

Configuring Microsoft SQL Server to use Windows authentication

In the SQL Server Enterprise Manager, select **Security > Logins**, right-click and choose **New Login**. Enter the desired domain account, including the domain name, in the following pattern: `DOMAIN\USER` (for example, `MY_DOMAIN\MY_ACCOUNT`).

In the Server Roles tab, select **System Administrators** and click **OK**.

Launching the OMi service with a Windows user

By default, the OMi service is run as a system service. If you have configured your Microsoft SQL Server to use Windows authentication, you must change the user running the OMi service to the same Windows user you defined for the Microsoft SQL Server in order to enable the service user to access the database.

The user you assign to run the service must have the following permissions:

- Sufficient database permissions (as defined by the database administrator)
- Sufficient network permissions
- Administrator permissions on the local server

To change the OMi service user, follow these steps:

1. Disable OMi (**Start > Programs > HPE Operations Manager i > Administration > Disable Operations Manager i**).
2. In Microsoft's Services window, double-click **HPE Operations Manager i**. The HPE Operations Manager i Properties (Local Computer) dialog box opens.
3. Click the **Log On** tab.
4. Select **This account** and browse to choose the user you previously defined for your Microsoft SQL Server.
5. Enter the selected user's Windows password and confirm this password.
6. Click **Apply** to save your settings and **OK** to close the dialog box.

7. Enable OMi (**Start > Programs > HPE Operations Manager i > Administration > Enable Operations Manager i**).

Creating or connecting to a Microsoft SQL database by using Windows authentication

You create or connect to a database by using the configuration wizard. To create or connect to a Microsoft SQL database by using Windows authentication, you must select this option within the configuration wizard. For details on using the configuration wizard, see the *OMi Installation and Upgrade Guide*.

Part III: Deploying and Maintaining the Oracle Server database

Chapter 8: Oracle Server deployment overview

You can set up OMi schemas on an Oracle Server. This chapter describes the following topics related to deploying Oracle Servers for use with OMi:

About Oracle Server deployment

To deploy Oracle Server for use with OMi, perform the following procedures:

- **Install Oracle Server.**

  For details on Oracle software installation, see the installation guide in the documentation for your specific Oracle platform. For software installation options, see "Oracle Server configuration and sizing guidelines" on page 66 and "Oracle summary checklist" on page 89.

  > **Note:** If you are installing Oracle 12c, in the Typical Installation step, it is recommended that you clear the **Create as Container database** option. This ensures that the database is created without CDB and will be the same as in Oracle 11.
  >
  > For a *silent* or *response file* setup, use the **createDatabase** operation (with the corresponding configuration section) instead of **createPluggableDatabase** in your DBCA response file.
  >
  > ```
  > OPERATION_TYPE = "createDatabase"
  > [CREATEDATABASE]
  > ```

- **Build a database on Oracle Server to store OMi data.**

  For instance configuration and sizing guidelines, see "Oracle Server configuration and sizing guidelines" on page 66. For details on database instance installation, see the installation guide in the documentation for your specific Oracle platform.

- **Create one or more Oracle tablespaces to store OMi data.**

  For details, see "Oracle tablespaces" on page 69.

- **Create an Oracle user schema for OMi schemas.**

You can create OMi user schemas manually, or you can use the configuration wizard to create the schemas for you. For details on creating an Oracle user schema for OMi, see "Manually creating OMi schemas" on page 61.

Oracle instances

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

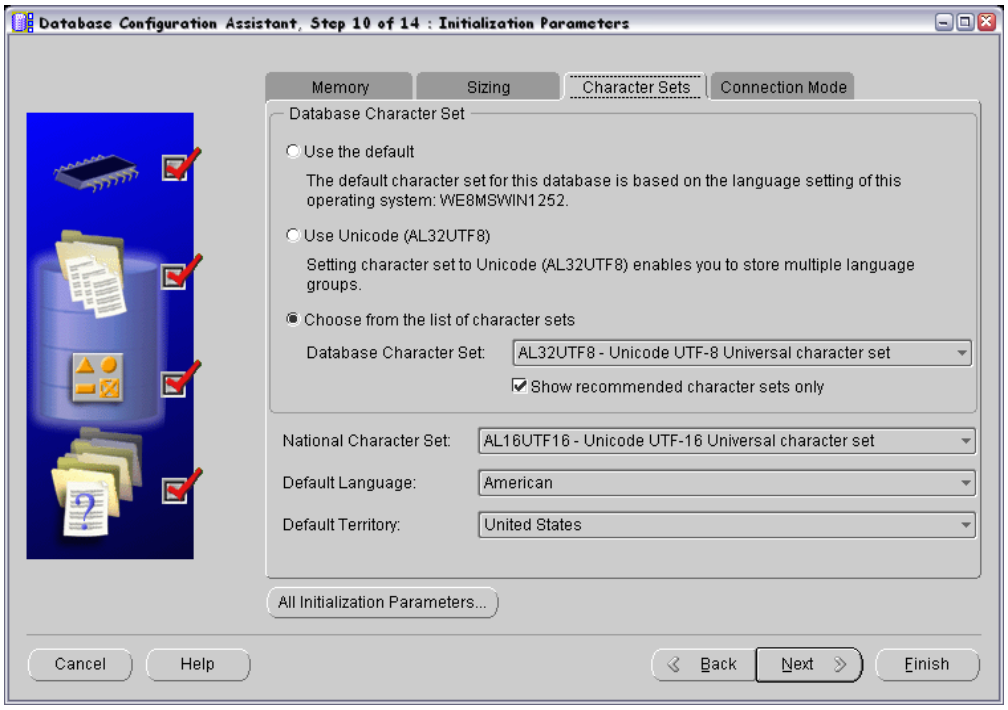
For OMi certification, do not use more than one Oracle instance. If you do use more than one instance for the OMi databases, ensure that all the instances are configured as described in this guide and that they all have the same characteristics (such as the same character set).

> **Note:** It is possible (although not recommended) to share the Oracle instance used by OMi with other applications. The prerequisites are:
>
> - making sure that the requirements for OMi, for example memory, are still fulfilled
> - the schema used by OMi must not be used by another application or another OMi (the Oracle users must be specific to one OMi instance)

Character set support in Oracle

When you create a new Oracle instance in an Oracle database, you must specify the character set for the instance. All character data, including the data in the data dictionary, is stored in the character set of the instance.

During the database creation, you can choose from the list of available character sets.



A UTF-8 character set is required for Oracle instances. There are two Oracle character sets that implement UTF-8 encoding, `UTF8` and `AL32UTF8`. Additionally, you can convert the character set `WE8ISO8859P1` to `UTF8`or `AL32UTF8`. For details, see "How to convert a WE8ISO8859P1 to a UTF-8 character set" on the next page.

It is recommended to set the client locale by setting the `NLS_LANG` environment variables to appropriate values, so that database client applications can communicate with the database server.

Language	`NLS_LANG` value
English	`american_america.AL32UTF8`
French	`french_france.AL32UTF8`
Spanish	`spanish_spain.AL32UTF8`
German	`german_germany.AL32UTF8`
Russian	`russian_russia.AL32UTF8`
Japanese	`japanese_japan.AL32UTF8`
Korean	`korean_korea.AL32UTF8`
Simplified Chinese	`simplified chinese_china.AL32UTF8`

How to convert a WE8ISO8859P1 to a UTF-8 character set

If your database is encoded in WE8ISO8859P1, you can convert it to a UTF-8 encoded character set by doing the following:

1. Export the database by using the Data Pump Export. For details, see the Oracle documentation.
2. Import the export dump file into a UTF-8 encoded Oracle database by using the Data Pump Import. The target database should use UTF8 or AL32UTF8. For details, see the Oracle documentation.

The character set is converted automatically during the data pump import.

Chapter 9: Manually creating Oracle schemas

This chapter describes how to manually create and populate OMi user schemas. It also contains information on Oracle permissions for OMi user schemas.

This chapter includes:

Oracle Server database overview

OMi uses the Management, RTSM, and Event databases.

During the OMi setup, new databases can be set up automatically by the setup procedure, or already existing databases can be used. Existing databases can either be created manually in advance (for example, due to organization security restrictions), or can be created by a previous installation of the same release of OMi.

For details about installing OMi, see the *OMi Installation and Upgrade Guide*.

Manually creating OMi schemas

The OMi user schemas can be created manually or automatically by using the configuration wizard. For more information on the configuration wizard, see the *OMi Installation and Upgrade Guide*.

> **Note:** You must use the same password for all schemas, which you then enter in the Database Settings page of the configuration wizard when connecting OMi to the database.

The configuration wizard prompts you to supply the following:

- **Host name.** The name of the host machine on which Oracle Server is installed.
- **Port.** The Oracle listener port. OMi automatically displays the default port, **1521**.
- **Admin user name and password.** (to connect as an administrator) The name and password of a

user with administrative permissions on Oracle Server (for example, a System user).

- **Password.** The password of a user that is used by OMi to connect to and retrieve data from the OMi databases.
- **SID.** The Oracle instance name that uniquely identifies the Oracle database instance being used by OMi.
- **Use TLS.** If selected, OMi will use an encrypted connection to communicate with the database server. For details on setting up TLS for database connections, see "Setting up TLS for database connections" on page 12.
- **Database Names (Management, RTSM, and Event).** The names of the existing user schemas, or the names that you choose for the new user schemas (for example, OMi_Management).
- **Default tablespace.** The name of the dedicated default tablespace you created for the user schemas. For details on creating a dedicated OMi tablespace, see "Oracle tablespaces" on page 69.
- **Temporary tablespace.** The name of the temporary tablespace you assigned to the user schemas. The default Oracle temporary tablespace is **temp**.

> **Note:** If you are using the SQL*Plus Oracle client, make sure you are aware of its limitations. For example, a database creation command may exceed the maximum command-line length supported by SQL*Plus. For information about the limitations of SQL*Plus, see the Oracle documentation.

This section includes the following topics:

> **Note:** In a distributed environment, make sure to run all generation scripts from the Data Processing Server.

Manually creating a Management user schema

It is recommended that a certified database administrator creates the management user schema manually (using your organization's database characteristic methodology). You then connect to the

management user schema later in the configuration wizard. For more information on the configuration wizard, see the *OMi Installation and Upgrade Guide*.

> **Note:** If you cannot submit database administrator connection parameters over a non-secure connection, you may need to configure a management user schema manually.

To generate and run the Management schema SQL script:

1. Create the user schema:

   Create a user schema with the permissions described in "OMi Oracle schema privileges" on page 65.

2. Generate the Management schema SQL script:

   Run the following script file located in the `<OMi_HOME>/bin` directory:

   `opr-generate-sql-script.[sh|bat]`

   The following parameters should be used for the Management schema:

   - `--create`: Generate a database creation script.
   - `--db_type`: `Oracle`
   - `--db_schema`: `Management`
   - `--out_file`: The name of the SQL file to generate, for example `mgmt.sql`

   Example:

   ```
   /opt/HP/BSM/bin/opr-generate-sql-script.sh --create --db_type Oracle
   --db_schema Management --out_file mgmt.sql
   ```

3. Run the Management schema SQL script:

   a. Connect to the Management schema and run the script generated in step 2 to deploy the Management objects.

   b. When the script is finished running, commit the change.

Manually creating an RTSM user schema

It is recommended that a certified database administrator creates the RTSM user schema manually (using your organization's database characteristic methodology). You then connect to the RTSM user schema later in the configuration wizard. For more information on the configuration wizard, see the *OMi Installation and Upgrade Guide*.

To create a separate RTSM user schema manually:

Create a user schema with the permissions described in "OMi Oracle schema privileges" on the next page.

Manually creating an Event schema

Create an Event schema with the permissions described in "OMi Oracle schema privileges" on the next page.

To generate the Event schema scripts:

Run the following script file:

`<OMi_HOME>\bin\opr-generate-sql-script.[sh|bat]`

The following parameters should be used for the Event schema:

- `--create`: Generate a database creation script.
- `--db_type`: `Oracle`
- `--db_schema`: `Event`
- `--out_file`: The name of the SQL file to generate, for example `event.sql`

Example:

```
/opt/HP/BSM/bin/opr-generate-sql-script.sh --create --db_type Oracle
--db_schema Event --out_file event.sql
```

To run the Event database SQL script:

1. Create the Event database.
2. Connect to the Event database and run the script generated in the previous section to deploy the Operations objects.

> **Note:** Perform the above procedures only if you are an experienced Oracle Database administrator.

OMi 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-db-creation.sql` script located in the `<OMi_HOME>/conf/sql/oracle` directory) describes the required database permissions that must be granted to the OMi user. These permissions are also used by the OMi platform components to create a new Oracle user:

- `GRANT CONNECT TO <OMi Oracle user schema>`
- `GRANT CREATE SEQUENCE TO <OMi Oracle user schema>`
- `GRANT CREATE TABLE TO <OMi Oracle user schema>`
- `GRANT CREATE TRIGGER TO <OMi Oracle user schema>`
- `GRANT UNLIMITED TABLESPACE TO <OMi Oracle user schema>`
- `GRANT CREATE VIEW TO <OMi Oracle user schema>`
- `GRANT CREATE PROCEDURE TO <OMi Oracle user schema>`
- `ALTER USER <OMi Oracle user schema> DEFAULT ROLE ALL`

If the schema is being used for the RTSM database, then the following permission is also needed:

```
GRANT CREATE TYPE TO <OMi Oracle user schema for RTSM>
```

> **Note:** OMi supports any user with higher permissions. For OMi certification, use an Oracle user that has the exact Oracle permissions described above.

Chapter 10: Oracle Server configuration and sizing guidelines

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

This chapter includes:

Oracle parameter settings

The table below describes the recommended values for a number of Oracle database initialization parameters, when working with the OMi database server.

> **Note:** If any of the database initialization parameters listed below is deprecated in your Oracle version, you can safely ignore it.

Parameter name	OMi deployment		Remarks
	Small	Large	
DB_BLOCK_SIZE	8K	8K-16K	Should be a multiple of the operating system block size.
DB_CACHE_ADVICE	ON	ON	For gathering statistics when tuning is required.
SGA_TARGET	1 GB	Minimum 4 GB	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_**

Parameter name	OMi deployment		Remarks
	Small	Large	
			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, its 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.
MEMORY_TARGET	2 GB	Minimum 5 GB	Automatic Memory Management enables the entire instance memory to be automatically managed and tuned by the instance. The instance memory contains the System Global Area (SGA) and the Program Global Area (PGA). MEMORY_ TARGET is the only required memory parameter to set. However, it is recommended to set SGA_TARGET or PGA_ AGGREGATE_TARGET as well to avoid frequent resizing of the SGA and PGA components. The values entered for SGA_ TARGET and PGA_AGGREGATE_TARGET serve as minimum values.
LOG_BUFFER	1 MB	5 MB	
DB_FILE_ MULTIBLOCK_READ_ COUNT	Oracle default value	Oracle default value	
PROCESSES	200	500	
SESSIONS	225	445	(1.1 * PROCESSES) + 5
OPTIMIZER_ INDEX_ COST_ ADJ	100	100	When set to 100, the optimizer evaluates the index access path at the regular cost. When set to any other value (for example, 50), the access path is evaluated at that percentage of the regular cost.
TIMED_ STATISTICS	True	True	
LOG_ CHECKPOINT_ INTERVAL	0	0	

Parameter name	OMi deployment		Remarks
	Small	Large	
LOG_ CHECKPOINT_ TIMEOUT	0 or ≥ 1800 (greater than or equal to 1800)	0 or ≥ 1800 (greater than or equal to 1800)	
OPTIMIZER_ MODE	ALL_ ROWS	ALL_ ROWS	
CURSOR_ SHARING	Exact	Exact	
OPEN_CURSORS	2000	2000	
COMPATIBLE	The same as the Oracle Server release installed	The same as the Oracle Server release installed	For example, 11.2.0 for Oracle 11g R2 or 12.0.0 for Oracle 12c.
UNDO_ MANAGEMENT	Auto	Auto	
UNDO_ RETENTION	Oracle default value	Oracle default value	Automatic tuning is performed.
RECYCLEBIN	Off	Off	
NLS_LENGTH_ SEMANTICS	BYTE	BYTE	This parameter controls the length definition of varchar columns.
WORKAREA_SIZE_ POLICY	AUTO	AUTO	
PGA_AGGREGATE_ TARGET	400 MB	Minimum 1 GB	
STATISTICS_LEVEL	TYPICAL	TYPICAL	Enables tuning if required.
OPTIMIZER_ CAPTURE_SQL_PLAN_ BASELINES	FALSE	FALSE	Controls Automatic Plan Capture as part of Oracle SQL Management Base (SMB).
AUDIT_TRAIL	NONE	NONE	Out-of-the-box database auditing is written to the SYS.AUD$ audit trail table. It is advisable to change this value to none to avoid growth of the system tablespace.
BLANK_ TRIMMING	False	False	

Parameter name	OMi deployment		Remarks
	Small	Large	
FIXED_DATE	*Do not set this value.*	*Do not set this value.*	OMi uses the SYSDATE function for generating system time as part of the application process.
_PARTITION_LARGE_ EXTENTS	FALSE	FALSE	Relevant for Oracle 11.2.0.2 and Oracle 12c. When this hidden parameter is set to TRUE, it affects the size of partitions in native partitioned tables. The initial extent allocated for each partition is very large, thus causing unwanted growth of database data files.
DEFERRED_ SEGMENT_CREATION	FALSE	FALSE	Relevant for Oracle 11.2.0.* only.

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 OMi, you must create one or more dedicated default tablespaces for your OMi user schemas. You may also want to create a dedicated temporary tablespace for OMi. To create a tablespace, you must provide specific operating system files that physically represent the tablespace, as well as extent parameters.

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

> **Note:** The following terms all see the identical database: Event schema = OPR Schema = Event database.

This section includes:

Locally managed tablespaces

Locally managed tablespaces 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.

All data and temporary tablespaces should be locally managed when working with OMi.

For information on locally managing temporary tablespace by using `TEMPFILE`, see "Temporary tablespace settings" on page 73.

Creating an Oracle tablespace

You create an OMi Oracle tablespace by using the `oracle_tablespace_create.bat` (for a Windows installation) or `oracle_tablespace_create.sh` script (for a Linux installation). Both scripts can be found in *<OMi_HOME>*`/opr/support/database/oracle`.

> **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 OMi installation. For more information, see "Data tablespace settings" on the next page.

To create an OMi 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>*

For example:

```
oracle_tablespace_create system manager topaz omi_tablespace
h:\oracle\oradata\topaz01.ora 50M
```

Parameter	Description
<admin_user>	Name of user with administrative permissions on Oracle Server.
<admin_password>	Password of specified user.
<tns_entry_name>	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. Note that the actual size needed varies depending on the amount of data OMi generates and the amount of time you keep historical data.

Caution: OMi does not support the table compression feature of Oracle Database. Therefore, when creating Oracle tablespaces, make sure table compression is disabled.

Oracle tablespace settings

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

This section includes the following topics:

Data tablespace settings

The following table specifies the recommended sizes for OMi tablespaces:

Tablespace	OMi deployment		Remarks
	Small	Large	
Management	5 GB	10 GB	The specified size is a minimum requirement.
RTSM	1 GB	20 GB	The specified size is a minimum requirement.
Event	5 GB	20 GB	The specified size is a minimum requirement.

Data tablespace default storage settings

The storage settings for data tablespaces should be:

- Locally managed tablespace
- Automatic segment space management
- Automatic local extent management

System tablespace settings

The following table specifies the recommended settings for system tablespaces:

Tablespace	OMi deployment	
	Small	Large
SYSTEM	2 GB	5 GB
SYSAUX	2 GB	5 GB

Note: Make sure you do not use the SYSTEM and SYSAUX tablespaces for the OMi schemas.

The System Tablespace storage default settings should be:

- Locally managed tablespace
- Segment space management:
  - **SYSAUX:** Automatic
  - **SYSTEM:** Manual
- Automatic local extent management

Temporary tablespace settings

The following table specifies the recommended settings for temporary tablespaces:

Tablespace	OMi deployment		Remarks
	Small	Large	
TEMP	1 GB	10 GB	Use multiple files with large tablespaces.
TEMP storage settings	Uniform allocation: 2 MB	Uniform allocation: 2 MB	• Should be locally managed (Uniform allocation). • Tablespaces should be of a temporary type (use of TEMPFILE). • Segment space management in temporary tablespaces is manual.

Redo log settings

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

Setting	OMi deployment	
	Small	Large
Redo log file size	200 MB	500 MB - 1 GB
Minimum number of groups	4	4
Minimum number of members per group	2	2

Undo segment settings

The following table specifies the recommended Undo settings:

Setting	OMi system profile		Remarks
	Small	Large	
Undo tablespace size	1 GB	10 GB	The number of segments, the minimum number

Setting	OMi system profile		Remarks
	Small	Large	
			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		

The Undo Tablespace storage default settings should be:

- Locally managed tablespace
- Automatic local extent management
- Segment space management in undo tablespaces is manual

Using RAID configuration

The use of RAID is transparent to Oracle. All the features specific to RAID configurations are handled by the operating system and not by Oracle.

The use of RAID devices differs according to the Oracle file type. Data files and archive logs can be placed on RAID devices, since they are accessed randomly. Redo logs may not be put on RAID devices, since they are accessed sequentially and performance is enhanced by having the disk drive head near the last write location. However, mirroring of redo log files is strongly recommended by Oracle.

RAID is much easier to use than the Oracle techniques for data placement and striping.

Note the following RAID configuration recommendations:

- RAID usually impacts write operations more than read operations. This is especially true where parity needs to be calculated (RAID 3, RAID 5, and so forth).
- You can place online or archived redo log files on RAID 1 devices. Do not use RAID 5. In addition, place TEMP tablespace data files on RAID 1 devices, instead of RAID 5, because the streamed write performance of distributed parity (RAID 5) is not as good as that of simple mirroring (RAID 1).
- Swap space can be used on RAID devices without affecting Oracle.

The following table describes the RAID devices and types to be used with each Oracle file type:

RAID	Type of RAID	Control file	Database file	Redo log file/temporary	Archive file
0	Striping	Avoid	OK	Avoid	Avoid
1	Shadowing	Recommended	OK	Recommended	Recommended
1+0	Striping and Shadowing	OK	Recommended	Avoid	Avoid
3	Striping with static parity	OK	OK	Avoid	Avoid
5	Striping with rotating parity	OK	Recommended if RAID 1+0 is not available.	Avoid	Avoid

Note:

- RAID 0 does not provide protection against failures. It requires a strong backup strategy.
- RAID 1+0 is recommended for database files because it avoids hot spots and provides the best possible performance during a disk failure. The disadvantage of RAID 1+0 is its costly configuration.
- Use the highest RPM disks for temporary/redo logs. Use as many controllers as you can in the array, and ensure that you place the redo log groups on different controllers.

ASM storage

In addition to the traditional file system option, Oracle Automatic Storage Management (ASM) can be used as a storage option for Oracle database files.

For details on configuring and administrating ASM, see the *Oracle Database Storage Administrator's Guide*.

For an ASM Overview, best practices, and technical papers, see:

http://www.oracle.com/technology/products/database/asm/index.html

Pay attention to the following tasks:

- Preparing and mapping of the ASM disks, configuring the ASM disk groups, and setting the correct redundancy.

- Usage of the Oracle ASMLIB driver for improved discovery and management of ASM disks and for enhanced I/O processing (Linux only).
- Proper setting of ASM_DISKSTRING according to your disks discovery path.
- Rebalance operation manual and automatic control options.

Chapter 11: Maintaining an Oracle Server database

This chapter describes the various maintenance and tuning procedures that are recommended for OMi databases created on Oracle Servers, as well as some of the available database backup and recovery methods.

This chapter includes:

- "Database maintenance and tuning" below
- "Oracle database backup and recovery" on page 83

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.

This section includes the following topics:

- "System Global Area (SGA)" on the next page
- "Database load behavior" on the next page
- "CPU and I/O" on the next page
- "Oracle alert file" on page 79
- "Archive log - file system" on page 79
- "Tablespace storage space" on page 79
- "Space management" on page 80

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. The parameters `MEMORY_TARGET` or `MEMORY_MAX_TARGET` cannot be greater than the shared memory file system (**/dev/shm**) on your operating system.

Database load behavior

Oracle AWR (Automatic Workload Repository) reports can be used to monitor performance bottlenecks and database behavior. For details, see Oracle Metalink Note 276103.1: PERFORMANCE TUNING USING ADVISORS AND MANAGEABILITY FEATURES.

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 AWR output to determine which particular Oracle data file is the cause of the contention and consider changing the I/O storage configuration.

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 may not exceed 70 percent and the I/O wait may not be higher than 10 percent.

To monitor the above resources, you can use **perfmon** on Windows or **top** on Linux, and the storage system management tools.

Oracle alert file

Oracle registers abnormal events in the **alert.log** file, whose location is defined by the BACKGROUND_DUMP_DEST parameter. As of Oracle 11g, the BACKGROUND_DUMP_DEST parameter is replaced by DIAGNOSTIC_DEST as part of Oracle Automatic Diagnostic Repository (ADR).

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 FIRST_TIME as "Day",
        COUNT(*) as "Number of files"
        from V$LOG_HISTORY
        group by FIRST_TIME
        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.

Space management

Free space in Oracle tablespaces is composed of newly created extents, or space that was used and freed by operations such as update and delete. If some of the free space in a tablespace is composed of extents that were used and freed, your tablespace may become fragmented. You can use the Oracle Segment Advisor to determine whether objects have unused space that can be released. Objects that are found to be fragmented can be reorganized to reclaim wasted space and compact the segment. For an overview on reclaiming wasted space, see the *Oracle Database Administrator's Guide* for your Oracle release.

Collecting statistics for databases

The OMi 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 OMi deployment, it is recommended that you collect statistics for all OMi objects (tables and indexes).

When working with large OMi 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 the tables and indexes created by the Partitioning and Purging Manager). Once your OMi system is stable, you should collect statistics once a day.

Oracle automated statistics collection

The Automatic Optimizer Statistics Collection job, GATHER_STATS_JOB, is managed by the Automatic Maintenance Tasks framework instead of the Oracle Scheduler framework. Automatic Maintenance Tasks are predefined tasks that perform maintenance operations on the database. These tasks run in a maintenance window, which is a predefined time interval that is intended to occur during a period of low system load. You can manually customize maintenance windows based on the resource usage patterns of your database, or disable certain default windows from running.

According to the Oracle documentation, the preferred way to manage or change configuration of the Automatic Optimizer Statistics Collection job is to use the Automatic Maintenance Tasks screens in Enterprise Manager Database Control and Grid Control. For an overview of the Automatic Maintenance Tasks, see the *Oracle Database Administrator's Guide*.

To manually collect statistics for all OMi objects on a database schema:

1. Log in to the OMi relevant schema by using SQL*Plus (a third party Oracle client).
2. Run the following command:

```
Exec DBMS_STATS.GATHER_SCHEMA_STATS (ownname  => '<name of Oracle schema>',
options => ‘GATHER AUTO’);
```

To collect statistics for specific schema tables and their indexes:

1. Log in to the schema by 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.
> - 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.

Collection statistics for RTSM

Unlike some databases, where queries are predefined and can be tuned according to the expected database size, RTSM 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 RTSM, it is recommended that you manually refresh statistics if major changes to the RTSM schema objects have occurred, such as bulk insert transactions triggered by automated DFM jobs. DFM is the process responsible for automatically detecting configuration items (CIs) and inserting them into RTSM.

Index fragmentation

OMi schemas consist of Oracle B-tree indexes for enhancing searches on table columns.

It is recommended to validate the structure of the 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 is able 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, there may be a need to rebuild some of the 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 OMi tables, it is recommended to rebuild any index that has more than three levels.

Index maintenance utility

OMi'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, follow these steps:

1. Copy the following files from the *<OMi_HOME>*\opr\support\database\oracle directory 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]

Parameter	Description
<schema>	Name of the database schema user of the schema for which you are running the utility.
<password>	Password for the database schema user.
<db alias>	Database 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]	Flag to instruct the utility to rebuild indexes automatically. Set the flag to 0 if you do not 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:** 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 by using scripts executing SQL commands combined with operating system commands to copy files, or by 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.

This section describes:

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, see the *Oracle Database Backup and Recovery User's Guide* for your Oracle release.

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.

> **Note:** If the Oracle Flash Recovery Area is used, then the default destination for archiving is the value set in the **db_ recovery_file_dest** parameter.

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, OMi may also experience some performance degradation due to disk load.

For more information, see the *Oracle Database Backup and Recovery User's Guide* for your Oracle release.

Data Pump Export and Import utilities

In addition to the cold and hot physical backup methods, you can use the backup method known as Data Pump. Data Pump APIs for moving data and metadata between databases and schemas replaced Oracle's original export/import utilities.

The Oracle Data Pump export utility dumps schema structure and contents into one or more Oracle binary files called a dump 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 Data Pump import utility. For more information on Oracle Data Pump utilities, see Oracle Database Utilities in the Oracle documentation for your release.

> **Note:** OMi does not require you to use a specific backup method; however, it is recommended that your backup method accommodate OMi's use of more than one database user schema.

Oracle Recovery Manager

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 by 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, see the *Oracle Database Backup and Recovery User's Guide* for your release.

Chapter 12: Configuring the Oracle Client for OMi

This chapter describes how to configure the Oracle Client for OMi database offline utilities. Installing an Oracle client on the application server is not a requirement. However, it is recommended to have one to operate the various offline database scripts such as manually creating users or collecting statistics against OMi databases.

Oracle Client versions and operating system platforms

The Oracle Client versions and operating system platforms that are supported and recommended for working with OMi are identical to the Oracle Server versions and platforms. For details, see Support Matrices for Operations Center products.

Oracle Client installation

To install the Oracle Client, see 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 OMi, 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.

MY.COMPANY.CO =
  (DESCRIPTON =
```

```
    (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, see 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.

If you are using a RAC deployment, you can configure your tnsnames entry to work with the RAC. For an example of a RAC entry, see "Support for Oracle Real Application Cluster" on page 122.

> **Note:**
>
> - The OMi servers access the Oracle Server by 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.
> - The Easy Connect Naming method eliminates the need to look up names in the `tnsnames.ora` files for TCP/IP environments. For details, see the *Oracle Database Net Services Administrator's Guide* in the Oracle documentation set of your release.

Chapter 13: Oracle summary checklist

This chapter contains a checklist summarizing the requirements for OMi support and certification.

For more information on the Oracle database configuration settings that should be used when working with Oracle Server and OMi, see "Oracle Server configuration and sizing guidelines" on page 66.

Checklist for OMi support and certification

The information provided in this section is for both supported and certified Oracle options. The certified options are recommended for working with OMi.

Option	Supported	Recommended	Remarks	For more information, see
Oracle edition	Enterprise	Enterprise		
Dedicated OMi server	Not necessary	Not necessary It is recommended to dedicate an instance for OMi.		
Use of multiple Oracle instances	Yes	No	The configuration of all instances must match in a certified environment.	"Oracle instances" on page 58
Use of non-default port	Yes	Yes		
Undo management	Automatic; Manual	Automatic	Set UNDO_ MANAGEMENT parameter to AUTO in a certified environment.	
Oracle Shared Servers connection method	Yes	No	OMi uses a connection pool architecture. Use the dedicated server connection method in a certified environment.	

Option	Supported	Recommended	Remarks	For more information, see
Oracle replication	No full support	No		
Operating System 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.	
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	UTF8; AL32UTF8	AL32UTF8		
OPEN_ CURSORS	2000	2000		
Working in archive log mode	True; False	True		
Autoextend option in tablespace files	Yes	No		
Locally managed data tablespace	Yes	Yes		"Locally managed tablespaces" on page 70
Tablespace extent management	Local automatic for application schemas; Local uniform for TEMP tablespace	Local automatic for application data tablespaces; Local uniform for TEMP tablespace		"Oracle tablespace settings" on page 71

Option	Supported	Recommended	Remarks	For more information, see
Automatic Segment Space Management Tablespace (ASSM)	Yes	Yes		

Oracle Client Configuration

In order to work with OMi, 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.

MY.COMPANY.CO =
  (DESCRIPTON =
    (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, see 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.

If you are using a RAC deployment, you can configure your tnsnames entry to work with the RAC. For an example of a RAC entry, see "Support for Oracle Real Application Cluster" on page 122.

> **Note:**
>
> - The OMi 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.

- The Easy Connect Naming method eliminates the need to look up names in the tnsnames.ora files for TCP/IP environments. For details, see the *Oracle Database Net Services Administrator's Guide* in the Oracle documentation set of your release.

Oracle Server and Client requirements

For the Oracle Server and Client versions and operating system platforms that are supported for working with OMi, see Support Matrices for Operations Center products.

Setting Oracle initialization parameters

For a list of recommended settings for Oracle server initialization parameters, see "Oracle parameter settings" on page 66.

Part IV: Deploying and maintaining the PostgreSQL Server databases

Chapter 14: PostgreSQL deployment overview

You can use PostgreSQL for deploying OMi databases. This chapter describes the following topics related to deploying PostgreSQL for use with OMi:

When configuring OMi, you can choose between an external PostgreSQL database (**Postgres**) and an embedded PostgreSQL database (**PostgresEmbedded**). You can always migrate from your embedded database to an external one. For details, see "Migrating from an embedded PostgreSQL database to an external PostgreSQL database" on page 109.

If you use the embedded PostgreSQL database, you can set up only one Data Processing Server. No additional Data Processing Servers can be added to an OMi deployment.

> **Note:** The location of the embedded PostgreSQL database files is as follows:
>
> - **Windows:** `%OvDataDir%\databases\omidb\`
> - **Linux:** `/var/opt/OV/databases/omidb/`

About PostgreSQL deployment

To deploy PostgreSQL for use with OMi, perform the following procedures:

- **Install and Configure PostgreSQL.**

  For details on PostgreSQL software installation, see the installation guide in the documentation for your specific PostgreSQL version.

  > **Caution:** Make sure you configure the `pg_hba.conf` file on the PostgreSQL server to accept remote connections. Otherwise, the connection to the PostgreSQL database cannot be established when running the OMi configuration wizard.

- **Create Databases on the PostgreSQL Server.**

You can create OMi databases manually, or you can use the configuration wizard to create the databases for you.

The configuration wizard prompts you to supply the following:

- **Host name.** The name of the host machine on which PostgreSQL is installed.
- **Port.** The PostgreSQL listening port. OMi automatically displays the default port for the external PostgreSQL database, 5432.

  > **Note:** If you use the embedded PostgreSQL database, OMi uses port 5433 to connect to the database.

- **Admin user name and password.** (*Only when creating a new database.*) The name and password of a user with administrative permissions on the PostgreSQL server.
- **User name and password.** The name and password of a user that OMi uses to connect to and retrieve data from the OMi databases.
- **Use TLS.** If selected, OMi will use an encrypted connection to communicate with the database server. For details on setting up TLS for database connections, see "Setting up TLS for database connections" on page 12.
- **Database names (Management, RTSM, and Event).** The names of the databases (for example, OMi_Management).

> **Note:** If you use the embedded PostgreSQL database, you are asked to provide only the password of a user that OMi uses to connect and retrieve data from the OMi databases.

For details on creating the databases for OMi manually, see "Manually creating the OMi databases" on page 97. For more information on the configuration wizard, see the *OMi Installation and Upgrade Guide*.

PostgreSQL parameter settings

The table below describes the recommended values for a number of PostgreSQL database initialization parameters. However, depending on how large your OMi deployment is, you might need to increase some of the values (for example, when using several Gateway Servers, you might need to increase the maximum number of client connections allowed).

Parameter name	Recommended value
`shared_buffers`	512 MB

Parameter name	Recommended value
work_mem	10 MB
maintenance_work_mem	64 MB
effective_cache_size	512 MB
max_connections	200
commit_delay	500
checkpoint_segments	32
checkpoint_timeout	10 min
checkpoint_completion_target	0.9

Character set support in PostgreSQL

When you create a new PostgreSQL cluster in a PostgreSQL database, you can specify the character set during the cluster creation by using the `-E` parameter with the `initdb` command:

```
initdb -E UTF8
```

To create a database inside a cluster with a different locale, use one of the following commands:

- From a terminal: `createdb -E UTF8 <dbname>`
- In SQL: `CREATE DATABASE <dbname> WITH ENCODING 'UTF8';`

> **Note:** To check the encoding of the database server, run the `psql -l` command or type `\l` inside the `psql` terminal.

When you use the PostgreSQL client application or any other client application to manually connect to the database, it is recommended that you specify the correct character set with the following SQL command:

```
SET NAMES "UTF8";
```

Chapter 15: Manually creating the OMi databases

This chapter describes the procedures for creating OMi databases on a PostgreSQL server manually.

This chapter includes:

PostgreSQL database overview

OMi uses the Management, RTSM, and Event databases.

During the OMi configuration, new databases can be created automatically by the configuration wizard, or already existing databases can be used. Existing databases can either be created manually in advance (for example, due to organization security restrictions), or can be created by a previous installation of the same release of OMi.

For details about installing the OMi server, see the *OMi Installation and Upgrade Guide*.

Manually creating OMi databases

This section describes how to create each of the OMi databases manually:

It is recommended that a certified database administrator creates the required database in compliance with the organization's database strategy.

> **Note:** In a distributed environment, make sure to run all generation scripts from the Data Processing Server.

Manually creating a Management database

To create a Management database manually, follow these steps:

1. Create the Management database for storing Management objects.
2. Create a database user that has full database access and superuser permissions. This user profile will be used by OMi to access the database. For information on granting permission, see "Granting access permissions" on page 100.
3. Create the Management objects as follows:
   a. Generate the Management database SQL script by using the following script file:
      - **For Windows:**

        `<OMi_HOME>\bin\opr-generate-sql-script.bat`
      - **For Linux:**

        `/opt/HP/BSM/bin/opr-generate-sql-script.sh`

      Use the following parameters for the Management database:
      - `--create`: Generate a database creation script.
      - `--db_type`: `Postgres`
      - `--db_schema`: `Management`
      - `--out_file`: The name of the SQL file to generate, for example `mgmt.sql`

      Example:

      ```
      /opt/HP/BSM/bin/opr-generate-sql-script.sh --create
      --db_type Postgres --db_schema Management --out_file mgmt.sql
      ```
   b. Connect to the Management database, and then run the generated script to deploy the Management objects.

Manually creating an RTSM database

Create a database for storing RTSM objects.

> **Note:** Unlike the Management and Event databases, the RTSM database does not require running any scripts to create the objects to populate it.

Manually creating an Event database

To create an Event database manually, follow these steps:

1. Create the Event database for storing Event objects.
2. If you haven't already, create a database user that has full database access and superuser permissions. This user profile is used by OMi to access the database. For information on granting permission, see "Granting access permissions" on the next page.
3. Create the Event objects as follows:
   a. Generate the Event database SQL script by using the following script file:
      - **For Windows:**

        `<OMi_HOME>\bin\opr-generate-sql-script.bat`
      - **For Linux:**

        `/opt/HP/BSM/bin/opr-generate-sql-script.sh`

      Use the following parameters for the Event database:
      - `--create`: Generate a database creation script.
      - `--db_type`: `Postgres`
      - `--db_schema`: Event
      - `--out_file`: The name of the SQL file to generate, for example `event.sql`

      Example:

      ```
      /opt/HP/BSM/bin/opr-generate-sql-script.sh --create
      --db_type Postgres --db_schema Event --out_file event.sql
      ```
   b. Connect to the Event database, and then run the generated script to deploy the Event objects.

Parameters for Creating OMi Databases

To create OMi databases, use the `CREATE DATABASE` SQL command. The syntax of this command is as follows:

```
CREATE DATABASE name
    [ [ WITH ] [ OWNER [=] user_name ]
           [ TEMPLATE [=] template ]
```

```
[ ENCODING [=] encoding ]
[ LC_COLLATE [=] lc_collate ]
[ LC_CTYPE [=] lc_ctype ]
[ TABLESPACE [=] tablespace ]
[ CONNECTION LIMIT [=] connlimit ] ]
```

Important: To create a database, the PostgreSQL server must be up and running.

Example:

This is an example of creating OMi databases named `event3`, `mgmt3`, and `rtsm3`:

```
CREATE DATABASE event3
 WITH OWNER = hpbsm
   ENCODING = 'UTF8'
   TABLESPACE = pg_default
   LC_COLLATE = 'English_United States.1252'
   LC_CTYPE = 'English_United States.1252'
   CONNECTION LIMIT = -1;

CREATE DATABASE mgmt3
 WITH OWNER = hpbsm
   ENCODING = 'UTF8'
   TABLESPACE = pg_default
   LC_COLLATE = 'English_United States.1252'
   LC_CTYPE = 'English_United States.1252'
   CONNECTION LIMIT = -1;

CREATE DATABASE rtsm3
 WITH OWNER = hpbsm
   ENCODING = 'UTF8'
   TABLESPACE = pg_default
   LC_COLLATE = 'English_United States.1252'
   LC_CTYPE = 'English_United States.1252'
   CONNECTION LIMIT = -1;
```

For details on how to grant access permissions, see "Granting access permissions" below.

Granting access permissions

When using a PostgreSQL database, you can perform any database action that is allowed by the PostgreSQL permissions granted to the OMi user. The following are the required database permissions that must be granted to the OMi user for all OMi databases (the Management, RTSM, and Event databases):

```
GRANT ALL ON DATABASE <mgmt db name> TO hpbsm;
GRANT ALL ON DATABASE <event db name> TO hpbsm;
GRANT ALL ON DATABASE <RTSM db name> TO hpbsm;

ALTER USER hpbsm WITH SUPERUSER;
```

> **Note:** The superuser permissions are required for the registration of the `uuid-ossp` extension, which is used during database creation. The superuser permissions can be removed after the databases have been created.

Chapter 16: Maintaining a PostgreSQL database

This chapter describes the maintenance procedures that are recommended for OMi databases created on PostgreSQL servers.

This chapter includes:

Database maintenance

Achieving optimum database performance requires that certain maintenance tasks are performed regularly. One of the required PostgreSQL database maintenance tasks is called **vacuuming** and it is performed by the autovacuum daemon, a built-in PostgreSQL tool that is enabled in OMi and executes the following recurring maintenance tasks:

- Recovering disk space by running VACUUM commands
- Updating planner statistics

You can also perform database vacuuming manually. This is especially useful after running a large amount of `INSERT`, `UPDATE`, or `DELETE` statements. For this purpose, use the built-in PostgreSQL tool, `vacuumdb`, which you can find at the following location:

`<OMi_HOME>/pgsql/bin/`

> **Note:** The built-in tool may not be fully compatible with the version of the external PostgreSQL database server. Therefore, if you use an external PostgreSQL database, make sure to use the `vacuumdb` tool from the database server instead.

Although indexes are maintained regularly by PostgreSQL, sometimes a manual rebuild of an index can be done to improve query performance (for example, after running a large amount of `INSERT`, `UPDATE`, or `DELETE` statements). To do so, use the built-in tool for rebuilding indexes in a PostgreSQL database, `reindexdb`.

> **Note:** Rebuilding indexes by using `reindexdb` may block table access. Therefore, run this tool only when the database server is under low load.

PostgreSQL database backup and restore

A well-defined backup and restore strategy is crucial for every database because you can lose or corrupt data due to various reasons. Whatever might be the cause for data loss or corruption, this is when your backup strategy, the key to successfully recovering your data, is put to the test. In addition to your scheduled backups, it is important to perform a backup whenever the database structure changes or before you upgrade your software or hardware.

Depending on whether you use an embedded or external PostgreSQL database, see one of the following sections about how to back up and restore your data:

- "Backing up and restoring an embedded PostgreSQL database" below
- "Backing up and restoring an external PostgreSQL database" on page 105

Backing up and restoring an embedded PostgreSQL database

OMi provides you with the `opr-pgctl` tool that you can use to manage, back up, and restore your embedded PostgreSQL database. You can find this tool at the following location:

- **Windows:** `<OMi_HOME>\bin\opr-pgctl.bat`
- **Linux:** `<OMi_HOME>/bin/opr-pgctl.sh`

The `opr-pgctl` tool is available only on the Data Processing Server.

Embedded PostgreSQL database backup

To perform a full backup of an embedded PostgreSQL database, run the following command:

```
opr-pgctl -backup <destination> [-z] [-u] <user_name> [-p] <password>
```

`<destination>`

The name of the backup output file. Make sure that it does not exist already.

`-z`

Optional. Compresses the output file by using `gzip`.

`-u <user_name>`

Optional. The user name of the administrative database user. If not specified, the user name `postgres` is used.

`-p <password>`

Optional. The password of the administrative database user. If not specified, the OMi database user's password is used.

The file that is created by the backup operation includes the information for recreating all databases, user roles and groups, access permissions, and tablespaces of the embedded PostgreSQL database cluster. It does not, however, include the information about the configuration file (that is, `postgresql.conf`). If you want to back up the complete cluster data directory, including all the configuration information of the database, you can use the continuous archiving backup method. For details, see the PostgreSQL documentation available at:

http://www.postgresql.org/docs/manuals/

> **Note:** Backing up the embedded PostgreSQL database requires no OMi downtime.

Embedded PostgreSQL database restore

To restore an embedded PostgreSQL database by using the backup file that you created during the backup operation, run the following command:

`opr-pgctl -restore <source> [-z] [-u] <user_name> [-p] <password>`

`<source>`

The name of the backup file. Make sure that you provide the `-z` parameter if you compressed the backup file by using `gzip`.

`-z`

Optional. Decompresses the output file by using `gzip`.

`-u <user_name>`

Optional. The user name of the administrative database user. If not specified, the user name `postgres` is used.

`-p <password>`

Optional. The password of the administrative database user. If not specified, the OMi database user's password is used.

Restoring the backup might fail if the database password of the backup is different than the current database password. To restore an embedded PostgreSQL backup although the password was changed, make sure to specify the changed administrative database user's credentials.

When you specify those credentials, the user name and password are synced to the restored database. You can disable this behavior by adding `--no-user-sync` to the command.

Restoring the embedded PostgreSQL database is performed in the following order:

- Existing cluster data directory of the embedded PostgreSQL database is removed.
- Cluster data directory is reinitialized using the credentials entered in the configuration wizard.
- Databases are recreated by using the specified backup file.

Caution: You must stop OMi before restoring the embedded PostgreSQL database.

Backing up and restoring an external PostgreSQL database

When backing up an external PostgreSQL database, you can choose among the following backup methods:

- SQL dump
- File system level
- Continuous archiving

This section includes a brief description of all three methods. For more details as well as for advantages and disadvantages of each of these backup methods, see the PostgreSQL documentation.

SQL dump

With the SQL dump method, a file containing the necessary SQL commands to recreate the database is generated. You can choose between the following utility programs to generate the backup file:

- `pg_dump`

  The backup output file contains the information about tables of a single database. It does not, however, include the information about all databases, user roles and groups, access permissions, and tablespaces, which means that you must recreate this information manually before you restore your database.

- `pg_dumpall`

  The backup output file contains all the information inside a database cluster, including the information about all databases, user roles and groups, access permissions, and tablespaces.

> **Note:** The configuration files are not backed up with the SQL dump backup method. Therefore, you must recreate them or back them up manually.

File system level

With the file system level backup method, you directly copy the files that your PostgreSQL database uses for storing the data. For this purpose, you can use tools such as `copy` or `tar`.

> **Caution:** Make sure that you shut down the database server before copying the files used for storing the data.

Continuous archiving

This method involves the continuous archiving of the write ahead logs (WALs) that are used to record all changes made to the data files inside a PostgreSQL database. By default, these log files are periodically overwritten with new information. However, you can configure PostgreSQL to keep copies of the old WALs, so that it is possible to restore a database to a specific point in time by replaying the log entries.

Resetting or changing the user password for the embedded PostgreSQL database

To reset a lost user password or to change a user password for the embedded PostgreSQL database, on the Data Processing Server, perform the following steps:

1. Stop OMi.
2. Temporarily disable the password requirement for the administrative PostgreSQL user `postgres`. To do so, edit the `pg_hba.conf` file by adding the following line or lines at the beginning of the file:
   - **Windows:**

     ```
     host all postgres ::1/128 trust

     host all postgres 127.0.0.1/32 trust
     ```

   - **Linux:**

     ```
     local all postgres trust
     ```

   > **Note:** You can find the `pg_hba.conf` file at the following location:
   > - **Windows:** `%OvDataDir%\databases\omidb\`
   > - **Linux:** `/var/opt/OV/databases/omidb/`

3. Start the embedded PostgreSQL database by running the following command:

   ```
   <OMi_HOME>/bin/opr-pgctl -start
   ```

4. Connect to the database by running the following command:
   - **Windows:**

     ```
     <OMi_HOME>\pgsql\bin\psql.exe -U postgres -p 5433
     ```

   - **Linux:**

     ```
     <OMi_HOME>/pgsql/bin/psql -U postgres -p 5433
     ```

   > **Note:** You are not prompted to provide a password because the password requirement was temporarily disabled for the `postgres` user.

5. Change the password of the `postgres` and `hpbsm` database users (both users must share the same password) by running the following SQL commands in an open PostgreSQL client session:

   ```
   \password postgres

   \password hpbsm
   ```

   You must enter a new password twice for each command.

6. Terminate the PostgreSQL client session by entering `\q`.
7. Stop the embedded PostgreSQL database by running the following command:

```
<OMi_HOME>/bin/opr-pgctl -stop
```

8. Enable the password requirement for the `postgres` user by removing the following from the `pg_hba.conf` file:
   - **Windows:**

     ```
     host all postgres ::1/128 trust

     host all postgres 127.0.0.1/32 trust
     ```

   - **Linux:**

     ```
     local all postgres trust
     ```

After resetting the lost user password or changing the user password, you must reconfigure OMi to use the new password. To do so, on all Data Processing Servers and Gateway Servers, perform the following steps:

1. Run the OMi configuration wizard. For details, see the *OMi Installation and Upgrade Guide*.
2. In the **Database Settings** page, select **Connect to an existing database or user schema**.
3. Select the **Embedded Postgres** database type.
4. Enter and confirm your new password.

Chapter 17: Migrating from an embedded PostgreSQL database to an external PostgreSQL database

This chapter describes how to migrate your embedded PostgreSQL database to an external PostgreSQL database.

This chapter includes:

Reasons for migrating your embedded PostgreSQL database

The embedded PostgreSQL database server supplied with OMi allows for a quick and simple configuration procedure and it is sufficient for small environments. However, as the managed environment grows in complexity, it may be necessary for you to move to a dedicated external PostgreSQL database server.

The external PostgreSQL database is more suitable for advanced environments because the administrator has increased control over the configuration of the database server so that the database adjusts better to the needs of the OMi environment. With the external database, it is also possible to set up replication and load balancing, which highly increases the performance and availability of the database server, and helps reduce downtime in case of failure.

> **Note:** With the external PostgreSQL database, it is the database administrator's responsibility to keep the database up and running all the time, as well as to ensure that it is responsive and performs well.

Procedure for migrating from an embedded PostgreSQL database to an external PostgreSQL database

To migrate from an embedded PostgreSQL database to an external PostgreSQL database, follow these steps:

1. Create a backup of the existing embedded PostgreSQL database as described in "Backing up and restoring an embedded PostgreSQL database" on page 103.
2. Recreate the OMi databases on the external PostgreSQL database cluster by running the following command:

   `psql -f <backup file> -h <db host> -p <db port> -U <admin user>`

   In this instance, `<backup file>` is the name of the backup file that you created during the backup procedure. If you used the `-z` parameter with the `opr-pgctl` command, make sure to unzip the backup file manually by using tools such as 7-Zip or `gzip`.

   > **Note:** The `psql` tool is available with OMi and you can find it in the `<OMi_HOME>/pgsql/bin` directory. However, if you use an external PostgreSQL database, it is highly recommended to use the tool from the database server instead.

3. Reconfigure OMi to connect to the already existing PostgreSQL database.

The old OMi databases are removed and the following databases are created on the specified external PostgreSQL database cluster:

- `mgmt`
- `rtsm`
- `event`

Part V: Appendixes

Appendix A: Microsoft SQL Server data storage recommendation

This appendix contains recommendations for creating Microsoft SQL Server file groups in which to store OMi data. This information applies to advanced users only.

This appendix includes:

"Creating Microsoft SQL Server file groups" below

Creating Microsoft SQL Server file groups

When OMi databases are created automatically by the configuration wizard, OMi stores database objects in two file groups: the **PRIMARY** file group for system tables and the **userdata001** file group for user tables. If you are setting up a large deployment of OMi, it is recommended that you create additional file groups in which to store OMi 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 OMi in the future.
- **Read-only.** A file group can be marked as read-only. Because data cannot be changed, Microsoft 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 OMi components and rolls back database activity, causing an infinite loop of inserts.

You can create an additional file group (together with an additional Microsoft 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:**
>
> When creating file groups, the following configuration rules are important for proper file recovery, fault tolerance, performance, and archiving:
>
> - 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).

Appendix B: Microsoft SQL Server configuration options

Most server configuration options are dynamically configured by Microsoft SQL Server. For OMi certification, you may not change the default options unless you are instructed to do so by HPE Software 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 Microsoft SQL Management Studio (mainly the Server Properties dialog box).

The default values should be used except for the following items:

- Agent XPs (A) = 1
- awe enabled (A) = 1. This should only be set when more than 4GB is required on a 32Bit machine. Note that this feature is deprecated in Microsoft SQL Server 2012.

Configuration option	Default value for SQL Server 2012 and 2014	Default value for SQL Server 2008 and 2008 R2	OMi certification
access check cache bucket count (A)	0	0	Default
access check cache quota (A)	0	0	Default
ad hoc distributed queries (A)	0	0	Default
affinity I/O mask (A, RR)	0	0	Default
affinity64 I/O mask (A, only available on 64-bit version of SQL Server)	0	0	Default
affinity mask (A)	0	0	Default
affinity64 mask (A, RR), only available on 64-bit version of SQL Server	0	0	Default
Agent XPs (A)	0 (Changes to 1 when SQL Server	0 (Changes to 1 when SQL Server	1

Configuration option	Default value for SQL Server 2012 and 2014	Default value for SQL Server 2008 and 2008 R2	OMi certification
	Agent is started. Default value is 0 if SQL Server Agent is set to automatic start during Setup)	Agent is started. Default value is 0 if SQL Server Agent is set to automatic start during Setup)	
allow updates (Obsolete. Do not use. Will cause an error during reconfigure.)	0	0	Default
awe enabled (A, RR)		0	Default, Unless more than 4GB memory is required on a 32Bit machine. This feature was deprecated in Microsoft SQL Server 2012.
backup compression default	0	0	Default
blocked process threshold (A)	0	0	Default
c2 audit mode (A, RR)	0	0	Default
clr enabled	0	0	Default
common criteria compliance enabled (A, RR)	0	0	Default
contained database authentication	0	N/A	Default
cost threshold for parallelism (A)	5	5	Default
cross db ownership chaining	0	0	Default
cursor threshold (A)	-1	-1	Default
Database Mail XPs (A)	0	0	Default
default full-text language(A)	1033	1033	Default
default language	0	0	Default
default trace enabled (A)	1	1	Default

Configuration option	Default value for SQL Server 2012 and 2014	Default value for SQL Server 2008 and 2008 R2	OMi certification
disallow results from triggers (A)	0	0	Default
EKM provider enabled	0	0	Default
filestream_access_level	0	0	Default
fill factor (A, RR)	0	0	Default
ft crawl bandwidth (max), see ft crawl bandwidth(A)	100	100	Default
ft crawl bandwidth (min), see ft crawl bandwidth(A)	0	0	Default
ft notify bandwidth (max), see ft notify bandwidth(A)	100	100	Default
ft notify bandwidth (min), see ft notify bandwidth(A)	0	0	Default
index create memory (A, SC)	0	0	Default
in-doubt xact resolution (A)	0	0	Default
lightweight pooling (A, RR)	0	0	Default
locks (A, RR, SC)	0	0	Default
max degree of parallelism (A)	0	0	Default
max full-text crawl range(A)	4	4	Default
max server memory (A, SC)	2147483647	2147483647	Default
max text repl size	65536	65536	Default
max worker threads (A)	0 Zero auto-configures the number of max worker threads depending on the number of processors, using the formula (256+ (<processors>-4)	0 Zero auto-configures the number of max worker threads depending on the number of processors, using the formula (256+ (<processors>-4)	Default

Configuration option	Default value for SQL Server 2012 and 2014	Default value for SQL Server 2008 and 2008 R2	OMi certification
	* 8) for 32-bit SQL Server and twice that for 64-bit SQL Server.	* 8) for 32-bit SQL Server and twice that for 64-bit SQL Server.	
media retention (A, RR)	0	0	Default
min memory per query (A)	1024	1024	Default
min server memory (A, SC)	0	0	Default
nested triggers	1	1	Default
network packet size (A)	4096	4096	Default
Ole Automation Procedures (A)	0	0	Default
open objects (A, RR, obsolete)	0	0	Default
optimize for ad hoc workloads (A)	0	0	Default
PH_timeout (A)	60	60	Default
precompute rank (A)	0	0	Default
priority boost (A, RR)	0	0	Default
query governor cost limit(A)	0	0	Default
query wait (A)	-1	-1	Default
recovery interval (A, SC)	0	0	Default
remote access (RR)	1	1	Default
remote admin connections	0	0	Default
remote login timeout	10	20	Default
remote proc trans	0	0	Default
remote query timeout	600	600	Default
Replication XPs Option (A)	0	0	Default
scan for startup procs (A, RR)	0	0	Default

Configuration option	Default value for SQL Server 2012 and 2014	Default value for SQL Server 2008 and 2008 R2	OMi certification
server trigger recursion	1	1	Default
set working set size (A, RR, obsolete)	0	0	Default
show advanced options	0	0	Default
SMO and DMO XPs (A)	1	1	Default
SQL Mail XPs (A)	N/A	0	Default
transform noise words (A)	0	0	Default
two digit year cutoff (A)	2049	2049	Default
user connections (A, RR, SC)	0	0	Default
User Instance Timeout (A, only appears in SQL Server 2008 Express.)	N/A	60	Default
user instances enabled (A, only appears in SQL Server 2008 Express.)	N/A	0	Default
user options	0	0	Default
xp_cmdshell (A)	0	0	Default

Appendix C: Changing the database host

This section describes the procedure to migrate your OMi database schema to a different server.

1. Stop all OMi servers. This is required to prevent data loss and schema structure mismatch in the target environment.
2. Duplicate the OMi schemas you wish to migrate.

   This can be performed by an offline database backup or a database level export. For details, see one of the following sections, depending on your database type:

   - Microsoft SQL: "Backing up databases" on page 42
   - Oracle: "Oracle database backup and recovery" on page 83
   - PostgreSQL: "PostgreSQL database backup and restore" on page 103
3. Modify sessions table in your management database according to your new configuration.
4. Run the configuration wizard, specifying the location of the new database host in the Database Settings page.
5. Start all OMi servers.

Appendix D: Database clustering

With OMi, the following database clustering solutions are supported:

For more information about each database clustering solution, see the corresponding section.

SQL Server AlwaysOn

SQL Server AlwaysOn offers two distinct solutions to provide high-availability and disaster recovery on server as well as on database level: Failover Cluster and Availability Groups.

- To configure a Failover Cluster for OMi, enter the cluster server name as the host name in the OMi configuration wizard.
- To configure Availability Groups for OMi, enter the Availability Group Listener name as the host name and the port of the Availability Group Listener in the OMi configuration wizard. The group listener must have a static IP address.

For details about how to configure a Microsoft SQL Server Failover Cluster and Availability Groups, see the Microsoft SQL Server documentation.

Failover Cluster and Availability Groups are supported for all OMi databases (the Management, RTSM, and Event databases).

> **Note:** Database mirroring is not supported with OMi.

SQL Server Availability Group Recommendations

- It is recommended that you use a Certificate Authority (CA) certificate on all Availability Group nodes.

Alternatively, you can use database server specific certificates, but please make sure that their SubjectAlternativeName (SAN) contain the Fully Qualified Domain Names (FQDN) of all other nodes in the Availability Group.

For details about how to configure a Microsoft SQL Server Availability Group, see the Microsoft SQL Server documentation.

Support for Oracle Real Application Cluster

This section contains the configuration that needs to be done for OMi to work with Oracle Real Application Cluster. This information applies to advanced users only.

This section includes:

About Oracle Real Application Cluster

A cluster is a collection of interconnected servers that appear as one server to the end user and to applications. Oracle Real Application Cluster (RAC) is Oracle's solution for high availability, scalability, and fault tolerance. It is based on clustered servers that share the same storage.

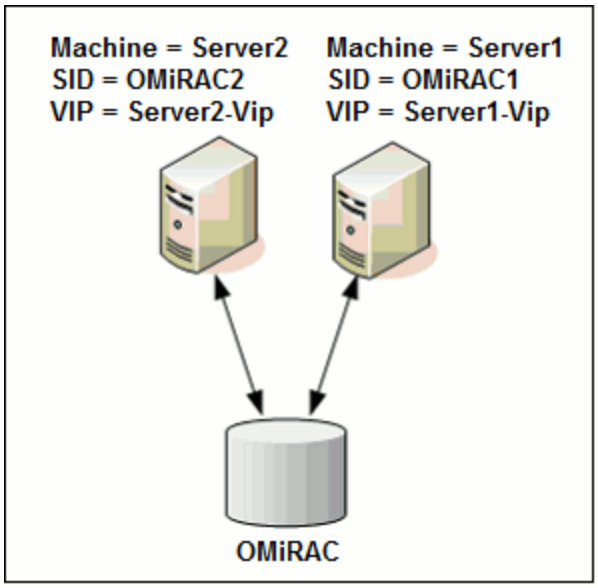
Oracle RAC is a single Oracle database installed on a cluster of hardware servers. Each server runs an instance of the database and all the instances share the same database files.

For more details about Oracle RAC, see the *Oracle Clusterware Administration and Deployment Guide* and the *Oracle Real Application Clusters Administration and Deployment Guide* in the Oracle documentation set of your release.

In this section, the following Oracle RAC example is used:

- Oracle RAC database name: OMiRAC
- Machine names: Server1, Server2
- On each machine there is an Oracle instance of OMiRAC:
  - SID on Server1: OMiRAC1
  - SID on Server2: OMiRAC2
- On each machine there is a virtual IP (Server1-Vip and Server2-Vip):
  - Server1-Vip is assigned to Server1
  - Server2-Vip is assigned to Server2

  The virtual IP is in addition to the static IP assigned to the machine.
- The listeners on both servers are listening on the default port 1521 and support the database service OMiRAC.



Single Client Access Name

In release 11g, Oracle introduced the Single Client Access Name (SCAN), as a preferred access method for clients connecting to the RAC. In this method clients are not required to configure individual nodes in the RAC, rather they use a single virtual IP known as the SCAN or the SCAN VIP.

The SCAN is a single network name defined for the cluster either in your organization's Domain Name Server (DNS) or in the Grid Naming Service (GNS) that rotates between several IP addresses

reflecting multiple listeners in the cluster. The SCAN eliminates the need to change clients when nodes are added to or removed from the cluster.

The SCAN and its associated IP addresses provide a stable name for clients to use for connections, independent of the nodes that make up the cluster. Database server SCAN addresses, virtual IP addresses, and public IP addresses must all be on the same subnet.

> **Note:** When using Oracle 11g RAC, it is recommended to use the SCAN method.

Client-side configuration for Oracle RAC

OMi uses DataDirect's JDBC driver to connect to regular Oracle databases and to Oracle RAC databases.

> **Note:** OMi also supports Oracle Data Guard.

Make the following changes in OMi's configuration files before you create the management database or connect to an existing one on Oracle RAC:

1. **On all servers, create the file <*OMi_HOME*>\conf\omi-tnsnames.ora.**

   The format of **omi-tnsnames.ora** is the same as **tnsnames.ora**:

   ```
   <DB service name> =
   (DESCRIPTION =
     (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = <first instance virtual ip> )
      (PORT = <first instance's listener port>))
      (ADDRESS = (PROTOCOL = TCP)(HOST = <second instance virtual ip> )(PORT =
   <second instance's listener port>))
      (... entry for each instance...)
      (LOAD_BALANCE = on)
      (FAILOVER = on)
      )
      (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = <DB service name>)
      )
    )
   ```

   where:

   - **<DB service name>** is the name of a service the listeners support. It is the same one used in the CONNECT_DATA part.

- **ADDRESS_LIST** contains an address entry for each node in the RAC environment. In the case of Oracle using SCAN, it contains only the SCAN virtual IP. The address contains all the details needed for connecting to the node:
  - **HOST** contains the virtual-IP for that instance. It is important to use the virtual IP and not the static IP of the node for faster failure detection.
  - **PORT** is the port on which the listener is configured to listen on that given node.
- **FAILOVER** set to **on** allows the driver to try to connect to another node after failing to connect to one of the nodes. The connection attempts continue until a connection is successfully established to one of the nodes or until none of the nodes can be reached.

  An attempt to connect to another node takes place only if the connection to the current node fails. If the connection is refused by the node (for example, communication to the node was established, but the connection was rejected), no attempt is made to connect to another node.

  > **Note:** Failover is for connection attempts only. If a connection fails during a database transaction, there is no failover of the transaction to another machine to continue the transaction.

- **LOAD_BALANCE** set to on instructs the driver to distribute connection requests between the nodes to prevent overloading any single node. The order in which the driver accesses the nodes is random.
- **SERVER** is the connection mode you want to use. There are two possible connection modes: **dedicated** and **shared**. Configure this according the Server configuration you support.
- **SERVICE_NAME** is the name of a service that the listeners support. It is the same one you gave in **<DB service name>**.

  > **Note:** Keep in mind that when connecting to a single Oracle database system, you can specify either a service name or a SID.

  In the above example, **omi-tnsnames.ora** would be configured as:

```
OMiRAC =
(DESCRIPTION =
 (ADDRESS_LIST =
  (ADDRESS = (PROTOCOL = TCP)(HOST = Server1-Vip)(PORT = 1521))
  (ADDRESS = (PROTOCOL = TCP)(HOST = Server2-Vip)(PORT = 1521))
  (LOAD_BALANCE = on)
  (FAILOVER = on)
 )
(CONNECT_DATA =
```

```
(SERVER = DEDICATED)
(SERVICE_NAME = OMiRAC)
)
)
```

> **Note:** If OMi schemas are deployed in more than one Oracle database or service, omi-tnsnames.ora should contain one entry for each database or database service. When all schemas are under the same RAC service, only one entry is required.

2. **On all servers, edit the file <*OMi_HOME*>\conf\jdbc.drivers.properties.**

   a. Find the section named **#ddoracle**. In this section, there is a line for **ddoracle.url** similar to:

   ```
   ddoracle.url=jdbc:mercury:oracle://${host}:${port};sid=${sid}
   ```

   b. Replace this line with either of the following lines (depending on the operating system):

   - **Windows.**

     ```
     ddoracle.url=jdbc:mercury:oracle:TNSNamesFile=<OMi_HOME>\\conf\\omi-tnsnames.ora;TNSServerName=${sid}
     ```

     where **${sid}** correlates to the service name entered in the Service and Database Configuration step. This parameter is saved per OMi schema; therefore, if there is more than one service name in the system, make sure that **omi-tnsnames.ora** includes an entry for each service name.

     Note that each back slash (\) in the path of <*OMi_HOME*> is doubled.

   - **Linux.**

     ```
     ddoracle.url=jdbc:mercury:oracle:TNSNamesFile=<OMi_HOME>/conf/omi-tnsnames.ora;TNSServerName=${sid}
     ```

3. **On all servers, open the directory <*OMi_HOME*>\odb\conf. Locate the jdbc.properties file.**

   a. Find the line starting with **cmdb.url**.

   b. Replace this with the following line:

   ```
   cmdb.url = jdbc:mercury:oracle:TNSNamesFile=<OMi_HOME>\\conf\\omi-tnsnames.ora;TNSServerName=<SERVICE NAME>
   ```

   where <SERVICE NAME> is the entry in **omi-tnsnames.ora**, equivalent to the RAC service name.

   c. If the file does not exist, create an empty **jdbc.properties** file under the above folder and add the following entry:

   ```
   Oracle = ddoracle
   cmdb.url =
   ```

```
jdbc:mercury:oracle:TNSNamesFile=<OMi_HOME>\\conf\\omi-
tnsnames.ora;TNSServerName=<SERVICE NAME>
```

where `<SERVICE NAME>` is the entry in **omi-tnsnames.ora**, equivalent to the RAC service name.

d. If your server is running Linux OS, replace all the double back slashes with single slashes.

Server-side configuration

In Oracle RAC, the Oracle listeners are always balancing the connection between the nodes according to either of the following algorithms:

- **Load Based (Default).** The listener redirects the connection according to the run queue length on the nodes. The least loaded node, CPU-wise, is connected.
- **Session based.** The listener balances the number of sessions between the nodes.

The first algorithm is optimized for short-lived connections, and less optimized for long-lived connections like the ones used with connection pools.

The entire load balancing is done during the connection time and not after it. This means that an established connection does not move to another node once connected.

Recommendation for an OMi database

It is recommended to use the session based algorithm since OMi uses connection pools. One connection can be used for different purposes and has a long lifetime.

To make the Oracle listener use the session based algorithm, a listener parameter must be added to the listener.ora parameter file of every listener (each node has its own listener, so this change must be done on all listeners):

```
PREFER_LEAST_LOADED_NODE_<LISTENER_NAME> =OFF
```

where:

`<LISTENER_NAME>` is the name of the listener. The default is `LISTENER_<node name>`.

For example:

The default listener name on **Server1** is **LISTENER_SERVER1**. In this case, you must add the following to Server1's listener.ora file:

```
PREFER_LEAST_LOADED_NODE_LISTENER_SERVER1=OFF
```

Create or connect to one of the OMi databases

When creating a new database schema or connecting to an existing schema in the configuration wizard, fill all the parameters as you usually would except for the following:

Parameter	Needed value
Oracle Machine	One of the Virtual IPs. The SCAN virtual IP can be used.
Port	The local listener port on the Oracle machine or the port of the SCAN listener.
SID	The service name of the database. **Note:** This must be the same as `<SERVICE_NAME>` entered in **omi-tnsnames.ora**.

In the example, the parameters would be:

Parameter	Value
Oracle Machine	Server1-Vip
Port	1521
SID	OMiRAC

PostgreSQL warm standby or log shipping

PostgreSQL provides the possibility to configure a warm standby server. The warm standby or log shipping high-availability feature allows one or more standby servers to take over operations if the primary server fails.

For OMi to be able to connect to an external PostgreSQL clustered environment, the cluster must be configured in such a way that it is accessible through a single virtual host name. This can be done by using a middleware (for example, pgpool-II) that maps the virtual host name to the currently active database node. The virtual host name can then be entered in the OMi configuration wizard.

For details about how to configure a warm standby server, see the PostgreSQL documentation.

Send documentation feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on OMi Database Guide (Operations Manager i 10.60)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to ovdoc-asm@hpe.com.

We appreciate your feedback!

Go OMi!