MICROSOFT
TRAINING
AND CERTIFICATION

Microsoft Official Curriculum

Module 5: Performing Administrative Tasks

Contents

Overview	1
Configuration Tasks	2
Lab A: Configuring SQL Server	14
Routine SQL Server Administrative Tasks	20
Automating Routine Maintenance Tasks	21
Lab B: Creating Jobs and Operators	33
Creating Alerts	41
Troubleshooting SQL Server Automation	51
Lab C: Creating Alerts	54
Automating Multiserver Jobs	66
Recommended Practices	69
Review	70

Trainer Materials for Microsoft Certified Trainer Use Only

Microsoft

Information in this document is subject to change without notice. The names of companies, products, people, characters, and/or data mentioned herein are fictitious and are in no way intended to represent any real individual, company, product, or event, unless otherwise noted. Complying with all applicable copyright laws is the responsibility of the user. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Microsoft Corporation. If, however, your only means of access is electronic, permission to print one copy is hereby granted.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2000 Microsoft Corporation. All rights reserved.

Microsoft, Active Directory, ActiveX, BackOffice, FrontPage, Jscript, Outlook, PowerPoint, Visual Basic, Visual Studio, Windows, Windows Media, and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the U.S.A. and/or other countries.

Other product and company names mentioned herein may be the trademarks of their respective owners.

Development Lead: Xandria Eykel
Technical Lead: Rick Byham
Instructional Designers: Cheryl Hoople, Lin Joyner (Content Master Ltd), Marilyn McGill (Independent Contractor), Gordon Ritchie (Content Master Ltd.),
Subject Matter Experts: Karl Dehmer, Mike Galos, Graeme Malcolm (Content Master), Mary Neville (Content Master Ltd), and Carl Rabeler (Shadow Mountain Computers),
Classroom Automation: Lorrin Smith-Bates
Graphic Artist: Kimberly Jackson (Independent Contractor)
Editing Manager: Lynette Skinner
Editor: Wendy Cleary
Copy Editor: Bill Jones (S&T Consulting)
Production Manager: Miracle Davis
Production Coordinator: Jenny Boe
Production Support: Ed Casper (S&T Consulting), Theano Petersen (S&T Consulting)
Test Manager: Sid Benavente
Courseware Testing: Testing Testing 123
Creative Director, Media/Sim Services: David Mahlmann
Web Development Lead: Lisa Pease
CD Build Specialist: Julie Challenger
Online Support: David Myka (S&T Consulting)
Localization Manager: Rick Terek
Operations Coordinator: John Williams
Manufacturing Support: Laura King; Kathy Hershey
Lead Product Manager, Release Management: Bo Galford
Lead Product Manager, Database Management: Margo Crandall
Group Manager, Courseware Infrastructure: David Bramble
Group Product Manager, Content Development: Dean Murray
General Manager: Robert Stewart

Microsoft Materials for Microsoft Certified Trainer Use Only

Instructor Notes

Presentation:
120 minutes

Labs:
135 minutes

This module provides students with details on performing configuration and routine administrative tasks. It describes how to automate tasks by creating jobs, operators, and alerts. The module also discusses automating tasks in a multiserver environment.

This module has three labs. In the first lab, students will configure SQL Server Agent, SQLAgentMail, and SQL Mail. In the second lab, students will create operators and jobs that consist of multiple job steps. In the third lab, students will create alerts.

After completing this module, the student will be able to:

- Perform common Microsoft® SQL Server™ 2000 configuration tasks.
- Describe routine database administrative tasks.
- Automate routine maintenance tasks by creating and scheduling jobs.
- Create alerts for SQL Server errors, user defined errors, or performance conditions, and notify operators when they occur.
- Troubleshoot automated jobs, alerts, or notifications.
- Automate administrative jobs within a multiserver environment.

Materials and Preparation

This section provides the materials and preparation tasks that you need to teach this module.

Required Materials

To teach this module, you need the Microsoft PowerPoint® file 2072A_05.ppt.

Preparation Tasks

To prepare for this module, you should:

- Read all of the materials for this module.
- Complete the labs.
- Practice the presentation, including the animated slide and multimedia presentation.
- Review any relevant white papers located on the Trainer Materials compact disc.

Trainer Materials for Microsoft Certified Trainer Use Only

Section and Lab Timing

When you practice the presentation, use the following section and lab timings as guidelines.

Section/Lab	Timing (minutes)
Configuration Tasks	30
Lab A: Configuring SQL Server	30
Routine SQL Server Administrative Tasks	45
Automating Routine Administrative Tasks	
Automating SQL Server Administration	
Lab B: Creating Jobs and Operators	60
Creating Alerts	30
Troubleshooting SQL Server Automation	
Lab C: Creating Alerts	45
Automating Multiserver Jobs	15
Creating a Master Job	

Instructor Setup for a Lab

This section provides setup instructions that are required to prepare the instructor computer or classroom configuration for a lab.

Lab A: Configuring SQL Server

► **To prepare for the lab**

1. Ensure that you have prepared your computer according to the steps presented in the Instructor Preparation for the Module section of these instructor notes.
2. Discuss the scenario to explain the multiple mail profiles on each computer.
3. The purpose of the exercises is to configure SQL Server Agent to send messages by using Microsoft Exchange.

Lab B: Creating Jobs and Operators

► **To prepare for the lab**

1. Ensure that you have prepared your computer according to the steps presented in the Instructor Preparation for the Module section of these instructor notes.
2. Discuss the scenario in the lab to explain the dependencies on the previous labs.
3. The purpose of this exercise is to configure a mail profile for SQL Server Agent and create jobs and operators.

Trainer Material for Microsoft Certified Trainer Use Only

Lab C: Creating Alerts

► **To prepare for the lab**

1. Ensure that you have prepared your computer according to the steps presented in the Instructor Preparation for the Module section of these instructor notes.
2. Discuss the scenario in the lab to explain the dependencies on the previous labs.
3. The purpose of this Exercise is to create user-defined error messages and performance condition alerts.

Instructor Preparation for the Module

Microsoft Outlook® and Microsoft Exchange Server are installed during classroom setup.

► **To prepare the instructor computer for this module**

In this procedure, you will start the Exchange Server services and verify that the mail features were properly configured during classroom setup.

1. Log in to the instructor computer as **Administrator**, with a password of **password**.
2. Open SQL Server Enterprise Manager.
3. Expand your server, and then expand **Management**.
4. Right-click **SQL Server Agent**, and then click **Properties**.
5. On the **General** tab, verify that the mail session has a profile name of MS Exchange Settings, and then click **Test**.

   You should receive a message that the test was successful. If not, or if the profile name is blank, proceed to the following procedures on troubleshooting and configuring.
6. If you received a message that the test was successful, open Outlook to verify that the **Administrator** domain user account can send e-mail messages to itself.

Trainer Materials for Microsoft Certified Trainer Use Only

Note If the previous procedure was successful, you are ready to proceed with the module; however, if the procedure failed, you must perform the following procedures.

► To troubleshoot problems sending and receiving e-mail

In this procedure, you will verify that the Exchange Server services are started and that the **Administrator** domain user account is able to send e-mail messages.

1. Verify that the following Exchange Server services were started by clicking **Services** in the Administrative Tools group:
   - Microsoft Exchange IMAP4
   - Microsoft Exchange Information Store
   - Microsoft Exchange MTA Stacks
   - Microsoft Exchange POP3
   - Microsoft Exchange Routing Engine
   - Exchange System Attendant
2. Verify that the SQL Server Agent service is running.
3. Open Outlook and verify that the **Administrator** domain user account can send e-mail messages to itself.

► To configure and test an Exchange profile for the Administrator domain user account if the Administrator could not send and receive e-mail

In this procedure, you will delete existing Exchange profiles and use Outlook to configure an Exchange profile for the domain user account, **Administrator**. Skip this procedure if **Administrator** is able to send e-mail messages to itself.

1. On the desktop, right-click **Microsoft Outlook**, and then click **Properties**.
2. If any information services are set up in this profile, click **Show Profiles**, and then remove all profiles that exist.
3. When complete, click **Close**.
4. On the desktop, double-click **Microsoft Outlook**.
5. Use the information in the following table to configure Outlook.

Option	Value
E-mail upgrade options	None of the above
E-mail service options	Corporate or workgroup
Name	Administrator
Initials	A
Information service	Microsoft Exchange Server
Microsoft Exchange Server	London
Mailbox	Administrator
Do you travel with this computer	No

Trainer Materials for Microsoft Certified Trainer Use Only

6. Click **Start using Microsoft Outlook**.
7. Verify that the **Administrator** domain user account can send e-mail messages to itself.
8. Log off Microsoft Windows 2000.

► To configure and test an Exchange profile for the SQL Service Agent domain user account

In this procedure, you will log on to Windows 2000 by using the SQL Service domain user account. You will then delete all existing Exchange profiles, and use Outlook to configure an Exchange profile for the domain user account, SQLService.

1. Log on to the instructor computer as **SQLService**, with a password of **sqlservice**.
2. On the desktop, right-click **Microsoft Outlook**, and then click **Properties**.
3. If any information services are set up in this profile, click **Show Profiles**, and then remove all profiles that exist.
4. When complete, click **Close**.
5. On the desktop, double-click **Microsoft Outlook**.
6. Use the information in the following table to configure Outlook.

Option	Value
Name	SQLService
Initials	S
Information service	Microsoft Exchange Server
Microsoft Exchange Server	London
Mailbox	SQLService
Do you travel with this computer	No

7. Click **Start using Microsoft Outlook**.
8. Verify that the SQLService domain user account can send e-mail messages to itself.
9. Log off Windows 2000.

Trainer Materials for Microsoft Certified Trainer Use Only

► To configure SQL Server Agent to use the SQLService Exchange profile

In this procedure, you will use SQL Server Enterprise Manager to configure SQL Server Agent to use the SQLService Exchange profile that you created in the previous procedure.

1. Log on to the instructor computer as **Administrator**, with a password of **password**.
2. Open SQL Server Enterprise Manager.
3. Expand your server, and then expand **Management**.
4. Right-click **SQL Server Agent**, and then click **Properties**.
5. On the **General** tab, in the **Mail profile** box, select **MS Exchange Settings**, and then click **Test**.

   You should receive a message that the test was successful.
6. Click **OK**, and then close the **SQL Server Agent Properties** dialog box.

Trainer Materials
for Microsoft Certified
Trainer Use Only

Multimedia Presentation

This section provides multimedia presentation procedures that do not fit in the margin notes or are not appropriate for the student notes.

Automating SQL Server Administration

You must play the multimedia presentation in class because the content is not presented anywhere else in the module.

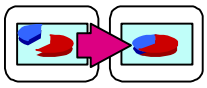
► **To start the multimedia presentation**

- Click the button in the slide to start the multimedia presentation.

  This multimedia presentation introduces the SQL Server automation components that allow the automation of administrative tasks. Then, it presents the SQL Server Agent components (jobs, alerts, and operators) and discusses how these components work together.

Other Activities

You need to become familiar with the following activities:



Displaying the Animated PowerPoint Slides

All animated slides are identified with an icon of links on the lower left corner of the slide.

► **To display the Assigning a Fail-Safe Operator slide**

1. Display the topic slide, which shows that an alert has been fired (error 18204 occurred).

   The alert is defined to notify an operator through e-mail, pager, and **net send** command. Explain the pager schedule.

2. Advance to the next animation, where the pager notification fails and, as a result, a fail-safe operator is notified. Discuss some of the reasons why a pager notification may fail.

Trainer Materials for Microsoft Certified Trainer Use Only

Module Strategy

Use the following strategy to present this module:

- Configuration Tasks

  Discuss the benefits of automation. Explain the configuration issues and the planning required in order to make the ongoing administration and maintenance of SQL Server 2000 much easier.

- Routine SQL Server Administrative Tasks

  Clearly state how to differentiate what can and cannot be automated in order to make clear what the administrator should focus on. Discuss the fact that it is possible to plan for and automate both planned and unplanned tasks.

- Automating Routine Maintenance Tasks

  Discuss creating jobs, which includes verifying permissions, defining one or more steps that make up that job, determining the flow logic of each job step, and scheduling the job.

  Explain that activity information about individual jobs is recorded in the job history table and that the size of the job history table can be configured to maintain sufficient records about each job.

- Creating Alerts

  Emphasize that the benefit of SQL Server Agent is its ability to fire alerts that can respond to potential problems before the problems actually occur.

  Describe the two ways that alerts can be defined: SQL Server error numbers or error severity levels. Discuss the details of creating user-defined error messages. Present the topic on responding to performance conditions and discuss the advantages of doing so. Then, explain that fail-safe operators can be created in the event that an alert that is defined to page an operator.

- Troubleshooting SQL Server Automation

  Focus on some of the guidelines and steps recommended to identify and resolve any issues with jobs, alerts or notifications. Discuss the importance of the error logs in assisting in troubleshooting. .

- Automating Multiserver Jobs

  Highlight the fact that SQL Server Agent allows the flexibility to automate jobs across multiple servers in a network. Describe how a multiserver environment is established and point out the specific activities to perform and issues to consider when students automate jobs in a multiserver environment.

Trainer Materials for Microsoft Certified Trainer Use Only

Customization Information

This section identifies the lab setup requirements for a module and the configuration changes that occur on student computers during the labs. This information is provided to assist you in replicating or customizing Microsoft Official Curriculum (MOC) courseware.

Important The lab in this module is dependent on the classroom configuration that is specified in the Customization Information section at the end of the *Classroom Setup Guide* for course 2072A, *Administering a Microsoft SQL Server 2000 Database*.

Lab Setup

The lab in this module requires that SQL Server 2000 Enterprise Edition has been installed on student computers. To prepare student computers to meet this requirement, perform exercise 1 in lab A, "Installing SQL Server," in module 2, "Planning to Install SQL Server" of course 2072A, *Administering a Microsoft SQL Server 2000 Database*.

Lab Results

Performing the labs in this module introduces the following configuration changes:

- A Microsoft Exchange mail profile is created for the SQLAdmin*x* domain user account.
- An Exchange mail profile is created for the SQLService domain user account.

Trainer Materials for Microsoft Certified Trainer Use Only

Overview

Topic Objective
To provide an overview of the module topics and objectives.

Lead-in
You can automate administrative tasks to simplify database and system maintenance. In this module we will cover…

- **Configuration Tasks**
- **Routine SQL Server Administrative Tasks**
- **Automating Routine Maintenance Tasks**
- **Creating Alerts**
- **Troubleshooting SQL Server Automation**
- **Automating Multiserver Jobs**

This module provides details on performing configuration and routine administration tasks. It describes how to automate tasks by creating jobs, operators, and alerts. The module also discusses automating tasks in a multiserver environment.

After completing this module, you will be able to:

- Perform common Microsoft® SQL Server™ 2000 configuration tasks.
- Describe routine database administrative tasks.
- Automate routine maintenance tasks by creating and scheduling jobs.
- Create alerts for SQL Server errors, user defined errors, or performance conditions, and notify operators when they occur.
- Troubleshoot automated jobs, alerts, or notifications.
- Automate administrative jobs within a multiserver environment.

Trainer Materials Microsoft Certified Trainer Use Only

Configuration Tasks

Topic Objective
To introduce the concepts of SQL Server automation.

Lead-in
Before you automate tasks, it is important to consider the fundamentals of SQL Server automation and how to prepare for automating administrative tasks with SQL Server.

- Configuring SQL Server Agent
- Configuring SQLAgentMail and SQL Mail
- Configuring Linked Servers
- Setting Up Data Source Names
- Configuring SQL Server XML Support in IIS
- Configuring SQL Server to Share Memory Resources with Other Server Applications

SQL Server database administrators have a number of one-time configuration tasks.

This section will examine some of these tasks and describe the necessary configuration steps.

Trainer Materials for Microsoft Certified Trainer Use Only

Configuring SQL Server Agent

Topic Objective
To discuss configuring SQL Server Agent.

Lead-in
In order for SQL Server Agent to execute jobs and fire alerts for SQL Server, it must be running at all times and must have sufficient permissions.

- **SQL Server Agent Must Be Running at All Times**
  - Configure SQL Server Agent to auto start
  - Configure SQL Server and SQL Server Agent services to restart automatically if these services stop unexpectedly
- **The SQL Server Agent Logon Account Must Be Mapped to sysadmin Role**
  - Map this account to the **Administrators** local group
  - Use a Windows domain user account logon account
- **Use Windows Authentication Mode for SQL Server Agent**

SQL Server Agent is the component of SQL Server responsible for automating SQL Server administrative tasks. In order for SQL Server Agent to execute jobs and fire alerts, it must be running at all times and must have sufficient permissions.

Delivery Tip
The **autostart** option is not available on the Windows 95/98 platform.

SQL Server Agent Must Be Running at All Times

On all Windows 2000 and Microsoft Windows NT® operating systems, SQL Server Agent generally runs as a Windows service. This service must be running in order to execute scheduled jobs and fire defined alerts. You should configure the SQL Server Agent service to start automatically whenever Windows 2000 or Windows NT is started. On Microsoft Windows 98, SQL Server Agent runs as an application and cannot be configured to start automatically (although the application could be placed in the Startup group to be launched when a user logs on).

In addition, you can configure the SQL Server Agent service to restart automatically if it stops unexpectedly by using SQL Server Enterprise Manager. In order to restart automatically, the SQL Server Agent service account must be a member of the **Administrators** local group.

Training Materials for Microsoft Certified Partners Only

The SQL Server Agent Logon Account Must Be Mapped to sysadmin Role

When you install SQL Server, you specify a logon account for the SQL Server Agent service. SQL Server Agent will not be able to start unless this login account is mapped to the SQL Server **sysadmin** role:

Delivery Tip
Point out that when using the Local System account, SQL Server Agent can access a network resource that has its own security mechanism, such as FTP or SQL Server.

- *Local System account*. Using the Local System account provides access to the local computer only. This Local System account is automatically a member of the Windows local group **Administrators** and is therefore mapped to the SQL Server **sysadmin** role. When the SQL Server Agent login account uses the Local System account, you generally cannot access network resources.
- *Domain user account*. A domain user account is required in order for SQL Server Agent to have permission to:
  - Communicate with most e-mail systems to send or receive e-mail.
  - Access resources across the network.

  You must map the domain user logon account to the SQL Server **sysadmin** role or add the account to a Windows local group that is mapped to the **sysadmin** role, generally the local Administrator account.

Delivery Tip
Point out that SQL Server Authentication must be selected when running SQL Server Agent on Windows 95/98.

Use Windows Authentication Mode for SQL Server Agent

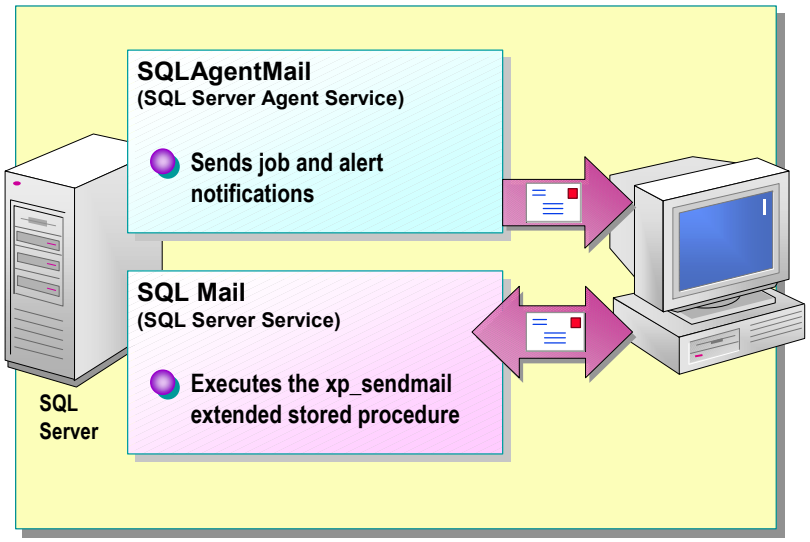
SQL Server Agent can connect to the local SQL Server by using either Windows Authentication or SQL Server Authentication. By default, SQL Server Agent uses Windows Authentication to connect to the local SQL Server by using the login account defined as the SQL Agent Service startup account. Changing the authentication mode for SQL Server Agent can prevent SQL Server Agent from accessing network resources.

Trainer Materials for Microsoft Certified Trainer Use Only

Configuring SQLAgentMail and SQL Mail

Topic Objective
To discuss configuring mail.

Lead-in
SQL Server sends and receives e-mail by using either a SQL Mail or SQLAgentMail session.



SQL Server can send and receive e-mail by establishing a client connection with a messaging server. SQL Server sends and receives e-mail by using either a SQLAgentMail or SQL Mail session.

Using SQLAgentMail

The SQL Server Agent service uses SQLAgentMail to send e-mail messages related to administrative functions. For example, the SQL Server Agent service uses SQLAgentMail to send a message when:

- A scheduled job succeeds or fails.
- An alert is fired.

The SQL Server Agent service attempts to start a mail session every time that SQL Server Agent is started. SQLAgentMail requires that the SQL Server Agent service use a domain user account with a mail profile. You can specify the mail profile to use by using SQL Server Enterprise Manager.

Using SQL Mail

SQL Mail consists of a number of extended stored procedures, which the SQL Server service uses to:

- Process incoming e-mail messages (expected to be only a single query) from the SQL Mail account inbox and return result sets to the message sender.
- Send e-mail messages from your database application, for example, by executing the **xp_sendmail** extended stored procedure from a trigger.

When SQL Mail is started, it starts a mail session by using a mail profile for the SQL Server service domain user account. You can configure SQL Mail to start automatically whenever SQL Server starts. SQL Mail requires that the SQL Server service use a domain user account with a mail profile. You can specify the mail profile to use by using SQL Server Enterprise Manager.

Trainer Materials for Microsoft Certified Trainer Use Only

Configuring SQLAgentMail and SQL Mail

For SQL Mail and SQLAgentMail to start mail sessions successfully, you must:

- Use a messaging server that is MAPI-1-compliant.
- Set up an e-mail client on the computer that is running SQL Server.
- Use a domain user login account for both the SQL Server and SQL Server Agent services. The same or different domain user login accounts may be used.
- Configure a mail profile for each domain user account to use to connect to the messaging server. If both services use the same domain user account, only one mail profile is needed.

  Consider using the same domain user account for both services to reduce the administrative overhead of maintaining multiple user accounts and mail profiles.
- Use SQL Server Enterprise Manager to specify a mail profile for each SQL Mail and SQLAgentMail session.

Delivery Tip
Demonstrate specifying a mail profile for SQL Mail and SQLAgentMail by using SQL Server Enterprise Manager.

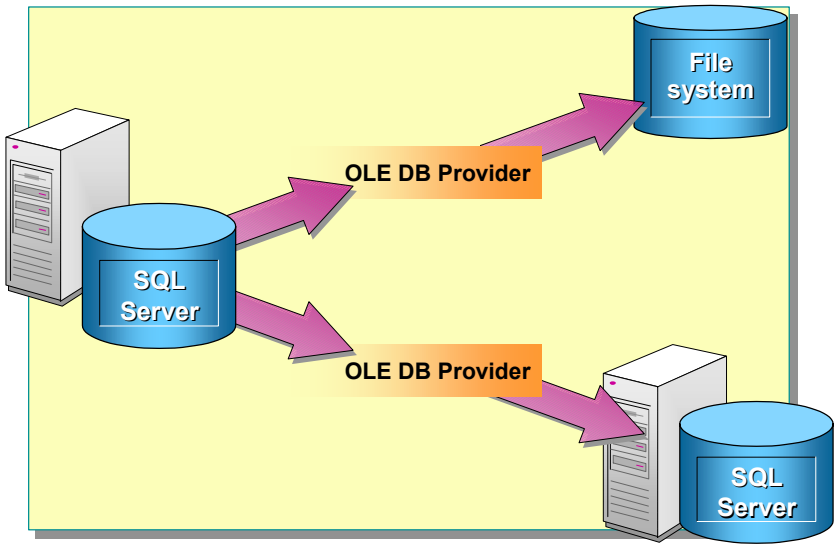
Note If you plan to send pager notifications, you must have a messaging server that is able to communicate with your pagers.

Trainer Materials for Microsoft Certified Trainer Use Only

Configuring Linked Servers

Topic Objective
To discuss the configuration of linked servers.

Lead-in
One of the configuration tasks of a database administrator is to configure linked servers.



One of the configuration tasks of a database administrator is to configure linked servers.

What Is a Linked Server Definition?

A linked server configuration allows SQL Server to execute commands against OLE DB data sources on different servers. A linked server definition specifies an OLE DB provider and an OLE DB data source.

An OLE DB provider is a dynamic-link library (DLL) that manages and interacts with a specific data source. An OLE DB data source identifies the specific database accessible through OLE DB. Although data sources queried through linked server definitions are usually databases, OLE DB providers exist for a wide variety of files and file formats, including text files, spreadsheet data, and the results of full-text content searches.

Managing a Linked Server Definition

When you set up a linked server definition, you register the connection information and data source information of the OLE DB data source with SQL Server. After you complete the registration, that data source can always be referred to with a single logical name.

You can create a linked server definition by using the **sp_addlinkedserver** system stored procedure or SQL Server Enterprise Manager.

Trainer Materials for Microsoft Certified Trainers Only

Establishing Security for Linked Servers

During a linked server connection (for example, when processing a distributed query), the sending server provides a logon name and password to connect to the receiving server on its behalf. You create login mappings between linked servers by using the **sp_addlinkedsrvlogin** system stored procedure or SQL Server Enterprise Manager.

The default mapping for a linked server configuration is to emulate the current security credentials of the logon account. This type of mapping is known as *self mapping*. When you add a linked server by using **sp_addlinkedserver**, a default self mapping is added for all local logon accounts.

If security account delegation is not available on the client or sending server, or the linked server or provider has no concept of Windows Authentication Mode, then self mapping will not work for Windows authenticated logon accounts. Therefore, you need to set up a local login mapping from a Windows authenticated logon account to a specific logon account on the linked server. In this case, the remote logon account will be a SQL Server authenticated logon account if the linked server is an instance of SQL Server.

When security account delegation is available and the linked server supports Windows Authentication, self mapping for the Windows authenticated logon accounts will be supported.

Trainer Materials
for Microsoft Certified
Trainer Use Only

Setting Up Data Source Names

Topic Objective
To discuss setting up data source names.

Lead-in
An ODBC application uses a data source to connect to an instance of SQL Server.

- **A Data Source Name Defines**
  - ODBC driver to use
  - Connection information (including name and location of data source, login account, and password)
  - Driver-specific options for the connection

An ODBC application uses a data source to connect to an instance of SQL Server. A data source is a stored definition that records:

- The ODBC driver to use for connections specifying the data source.
- The information that the ODBC driver uses to connect to a source of data.
- Driver-specific options to use for the connection. For example, a SQL Server ODBC data source can record the SQL-92 options to use, or whether the drivers should record performance statistics.

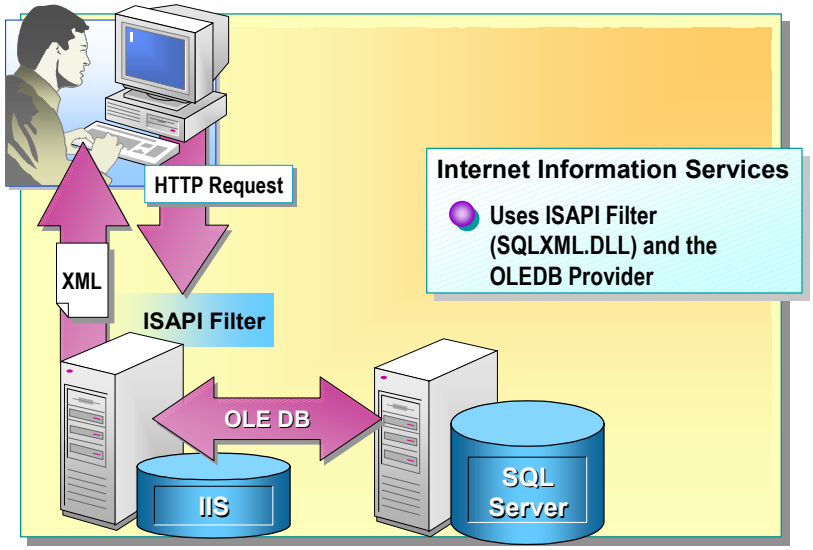
Each ODBC data source on a client has a unique data source name (DSN). An ODBC data source for the SQL Server ODBC driver includes all of the information used to connect to an instance of SQL Server, plus any essential options.

Trainer Materials for Microsoft Certified Trainer Use Only

Configuring SQL Server XML Support in IIS

Topic Objective
To discuss the configuration of IIS for XML access to SQL Server.

Lead-in
You can configure Microsoft Internet Information Services to allow access to SQL Server over IIS.



You can configure Microsoft Internet Information Services (IIS) to allow access to SQL Server over IIS.

Accessing SQL Server over HTTP

You can access SQL Server over Hypertext Transfer Protocol (HTTP) by using a Uniform Resource Locator (URL). You can specify that the URL:

- Directly accesses the database objects, such as tables. For security reasons this is not recommended, however.
- Executes template files. A template is a valid Extensible Markup Language (XML) document consisting of one or more Transact-SQL statements. When a template file is specified in the URL, the Transact-SQL statements stored in the template file are executed. You can specify Transact-SQL queries directly, at the URL, but for security reasons this is not recommended.
- Executes XML Path Language (XPath) queries. XPath queries are executed against annotated mapping schema files specified as part of the URL.

Setting Up a Virtual Directory in IIS

Before accessing a SQL Server database over HTTP, you must set up an appropriate virtual directory. Use the IIS Virtual Directory Management for SQL Server utility to define and register a new virtual directory, also known as the virtual root, on the computer running IIS. This utility instructs IIS to create an association between the new virtual directory and an instance of SQL Server.

Note You cannot use the virtual directory that you use for SQL Server data access to host an HTML/Active Server Pages (ASP) Web application.

Trainer Materials for Microsoft Certified Trainers Use Only

How XML Accesses SQL Server

SQL Server XML integration is made possible by extensions to the Transact-SQL language. In an XML interchange:

1. The IIS server name specified in the URL identifies the IIS server.
2. The IIS server examines the virtual root specified in the URL and determines whether an Internet Server Application Programming Interface (ISAPI) DLL file name extension (Sqlisapi.dll) has been registered for the virtual root that is specified in the URL.
3. The IIS server loads the DLL and passes on the URL request to the DLL.
4. The Sqlisapi.dll file name extension communicates with the OLE DB provider for SQL Server and establishes connection with the instance of SQL Server identified in the virtual root.

Note The XPath queries are handled on the IIS server. The XPath queries are translated into Transact-SQL statements by Sqlxmlx.dll and then passed to SQL Server.

Trainer Materials for Microsoft Certified Trainer Use Only

Configuring SQL Server to Share Memory Resources with Other Server Applications

Topic Objective
To discuss the configuration of SQL Server.

Lead-in
By default, SQL Server can change its memory requirements dynamically, based on available system resources.

- **Configuring the Memory Options**
  - **min server memory**
  - **max server memory**
- **Determining Maximum Amount of Memory**
- **Using Windows 2000 System Monitor to Observe Effects**

SQL Server must have sufficient memory available for the static memory needs (kernel overhead, open objects, locks, and so on). All additional memory is used for the data cache (also called the buffer cache). By default, SQL Server acquires and releases memory for the data cache dynamically, based on available system resources and SQL Server needs. This helps to prevent Windows 2000 paging by allowing the released memory to go on the free list.

This Microsoft Certified Partner Use Only

Configuring the Memory Options

Allowing SQL Server to use memory dynamically is the recommended configuration; however, you can set the memory options manually and override SQL Server's ability to use memory dynamically.

Some applications use whatever memory is available when they start, do not request more if needed, and do not release memory if requested. If an application that behaves like this runs on the same server as your instance of SQL Server, you may need to set the following options manually:

Option	Description
min server memory	This option defines a level below which SQL Server will not release memory.
max server memory	This option prevents SQL Server from using more than the specified amount of memory.

Ideally, you want to allocate as much memory as possible to SQL Server without causing the system to swap pages to disk. The threshold varies depending on your system and the applications with which SQL Server is sharing memory.

Determining Maximum Amount of Memory

You can determine the maximum amount of memory to configure for an instance of SQL Server by subtracting from the total physical memory the memory required for Windows 2000, other instances of SQL Server, and any other system uses or applications, if the computer is not wholly dedicated to SQL Server.

Using Windows 2000 System Monitor to Observe Effects

Use statistics from the Windows 2000 System Monitor to help you evaluate how your server applications are performing and to adjust the memory value, if necessary. Change this value only when you add or remove memory, or when you change how you use your system.

Trainer Materials
for Microsoft Certified
Trainer Use Only

Lab A: Configuring SQL Server

Topic Objective
To prepare students for the lab.

Lead-in
In this lab, you will configure a mail profile for SQL Server Agent.

Explain the lab objectives.

Objectives

After completing this lab, you will be able to:

- Configure a Microsoft Exchange Server profile to use SQLAgentMail for the SQLService domain user account.
- Configure SQL Mail.

Prerequisites

Before working on this lab, you must have script files for this lab. The script files are located in C:\Moc\2072A\Labfiles\L05.

For More Information

If you require help in executing files, search SQL Query Analyzer Help for "Execute a query".

Other resources that you can use include:

- The **Northwind** database schema.
- SQL Server Books Online.

Scenario

The organization of the classroom is meant to simulate that of a worldwide trading firm named Northwind Traders. Its fictitious domain name is nwtraders.msft. The primary DNS server for nwtraders.msft is the instructor computer, which has an Internet Protocol (IP) address of 192.168.*x*.200 (where *x* is the assigned classroom number). The name of the instructor computer is London.

Trainer Materials
Microsoft Certified
Trainer Use Only

The following table provides the user name, computer name, and the IP address for each student computer in the fictitious nwtraders.msft domain. Find the user name for your computer and make a note of it.

User name	Computer name	IP address
SQLAdmin1	Vancouver	192.168.x.1
SQLAdmin2	Denver	192.168.x.2
SQLAdmin3	Perth	192.168.x.3
SQLAdmin4	Brisbane	192.168.x.4
SQLAdmin5	Lisbon	192.168.x.5
SQLAdmin6	Bonn	192.168.x.6
SQLAdmin7	Lima	192.168.x.7
SQLAdmin8	Santiago	192.168.x.8
SQLAdmin9	Bangalore	192.168.x.9
SQLAdmin10	Singapore	192.168.x.10
SQLAdmin11	Casablanca	192.168.x.11
SQLAdmin12	Tunis	192.168.x.12
SQLAdmin13	Acapulco	192.168.x.13
SQLAdmin14	Miami	192.168.x.14
SQLAdmin15	Auckland	192.168.x.15
SQLAdmin16	Suva	192.168.x.16
SQLAdmin17	Stockholm	192.168.x.17
SQLAdmin18	Moscow	192.168.x.18
SQLAdmin19	Caracas	192.168.x.19
SQLAdmin20	Montevideo	192.168.x.20
SQLAdmin21	Manila	192.168.x.21
SQLAdmin22	Tokyo	192.168.x.22
SQLAdmin23	Khartoum	192.168.x.23
SQLAdmin24	Nairobi	192.168.x.24

Trainer Materials for Microsoft Certified Trainer Use Only

Estimated time to complete this lab: 30 minutes

Exercise 1
Configuring SQL Server Agent to Send Messages

In this exercise, you will use Microsoft Outlook® to configure an Exchange Server profile for the SQLService domain user account to enable SQL Server Agent to send and receive messages. You will then configure an Exchange profile for your domain user account, **SQLAdmin*x***, to send and receive messages with Outlook. Finally, you will use SQL Server Enterprise Manager to configure SQL Server Agent to use the mail profile that is configured for the SQLService domain user account.

► To configure an Exchange profile for your administrative account

In this procedure, you will use Outlook to configure an Exchange profile for your domain user account, **SQLAdmin*x***.

1. Log on to the **NWTraders** classroom domain with the information in the following table.

Option	Value
User name	**SQLAdmin*x*** (where *x* corresponds to your computer name as designated in the nwtraders.msft classroom domain)
Password	**password**

2. On the desktop, double-click **Microsoft Outlook**.

   The **Outlook 2000 Startup** dialog box appears.

3. Configure Outlook by using the information in the following table. Accept the defaults for any options that are not listed.

Option	Value
E-mail Service Options	Corporate or workgroup
Information service	Microsoft Exchange Server
Microsoft Exchange Server	**London**
Mailbox	**SQLAdmin*x*** (where *x* corresponds to your computer name as designated in the nwtraders.msft classroom domain)

4. Click **Yes** to make Outlook your default manager for Mail, News, and Contacts.
5. Compose and send an e-mail message to your e-mail alias.
6. Verify that the message appears in the Inbox.
7. Close Outlook.
8. Log off Microsoft Windows 2000.

Trainer Materials for Microsoft Certified Trainer Use Only

► To configure an Exchange profile for the SQLService domain user account

In this procedure, you will use Outlook to configure an Exchange profile for the domain user account, SQLService.

1. Log on to the **NWTraders** classroom domain with the information in the following table.

Option	Value
User name	**Sqlservice**
Password	**Sqlservice**

2. On the desktop, double-click **Microsoft Outlook**. Use the information in the following table to configure Outlook. Accept the defaults for any options that are not listed.

Option	Value
Information service	Microsoft Exchange Server
Microsoft Exchange Server	**London**
Mailbox	SQLService

► To test and verify that the SQLService domain user account received an e-mail message

In this procedure, you will send an e-mail message to the SQLService domain user account and verify its receipt.

1. Compose and send a message in Outlook to **SQLService**.
2. Verify that the message appears in the Inbox.

Trainer Materials for Microsoft Certified Trainer Use Only

► To verify the profile name

In this procedure, you will verify that the MS Exchange Settings profile name was created for the SQLService domain user account.

1. On the **Tools** menu, click **Options**, and then click the **Mail Services** tab.
2. Under **Startup Settings**, in the **Always use this profile** box, verify that **MS Exchange Settings** is displayed.
3. Close the **Options** dialog box, and then close Outlook.
4. Log off Windows 2000.

► To configure SQL Server Agent to use the mail profile

In this procedure, you will use SQL Server Enterprise Manager to configure SQL Server Agent to start a mail session that uses the MS Exchange Settings profile.

1. Log on to the **NWTraders** classroom domain with the information in the following table.

Option	Value
User name	**SQLAdmin*x*** (where *x* corresponds to your computer name as designated in the nwtraders.msft classroom domain)
Password	**password**

2. Open SQL Server Enterprise Manager.
3. In the console tree, expand **Microsoft SQL Servers**, expand **SQL Server Group**, expand your server, and then expand **Management**.
4. Verify that SQL Server Agent has started.
5. Right-click **SQL Server Agent**, and then click **Properties**.
6. On the **General** tab, in the **Mail profile** list, select **MS Exchange Settings**.
7. Click **Test**, and then click **OK** to acknowledge the message indicating that a mail session was successfully started (and stopped) with this profile.
8. Click **OK** to close the **SQL Server Agent Properties** dialog box.
9. Click **Yes** to stop and restart SQL Server Agent, and then click **OK** when SQL Server Agent has been successfully restarted.
10. In the console tree, right-click **SQL Server Agent**, and then click **Display Error Log**.
11. In the **Type** list, click **All Types**.
12. Verify that a mail session was started when SQL Server Agent was started.
13. Close the **SQL Server Agent Error Log** dialog box.

Trainer Materials for Microsoft Certified Trainer Use Only

Exercise 2
Using SQL Mail

In this exercise, you will configure SQL Mail and execute the **xp_sendmail** extended stored procedure to send the result set of a query to your e-mail account in Outlook.

► To configure SQL Mail

In this procedure, you will configure SQL Mail by specifying a mail profile name for SQL Mail.

1. In the SQL Server Enterprise Manager console tree, expand **Support Services**.
2. Right-click **SQL Mail**, and then click **Properties**.
3. In the **Profile name** list, select **MS Exchange Settings**, and then click **Test**.

   The mail session of SQL Server Agent also uses this profile, because both the SQL Server Agent and the SQL Server services use the SQLService Windows 2000 domain user account.
4. When the test completes successfully, click **OK**.
5. Click **OK** to close the **SQL Mail Configuration** dialog box.

► To send a query result by using SQL Mail

In this procedure, you will execute the **xp_sendmail** extended stored procedure to send the result set of a query to yourself to verify that the SQL Mail session works as expected.

1. Open SQL Query Analyzer and, if requested, log in to the (local) server with Windows Authentication.

   You have permission to log in to and administer SQL Server because you are logged as **SQLAdmin*x***, which is a member of the Windows 2000 local group, Administrators. All members of this group are automatically mapped to the SQL Server **sysadmin** role.
2. Open C:\Moc\2072A\Labfiles\L05\Sqlmail.sql, and review its contents.
3. Modify the script to specify your SQLAdmin*x* (where *x* corresponds to your computer name as designated in the nwtraders.msft classroom domain) e-mail account, and then execute the script.

   The result pane indicates that mail was sent.
4. Open Outlook and review your Inbox to ensure that you received an e-mail message with your query result.

   What was the query result?

   77

Trainer Materials for Microsoft Certified Trainer Use Only

Routine SQL Server Administrative Tasks

Topic Objective
To discuss the routine tasks of a SQL Server administrator.

Lead-in
Automating routine maintenance tasks on your local server or in a multiserver environment allows you to spend time on other database administrative functions.

- **Performing Regularly Scheduled Tasks**
  - Back up databases
  - Import and export data
- **Recognizing and Responding to Potential Problems**
  - Monitor database and log space
  - Monitor performance

Delivery Tip
Present automation in terms of planned versus unplanned maintenance tasks.

One of the primary functions of a database administrator is to maintain SQL Server and its databases. These include tasks that must be performed on a scheduled basis, and additional tasks to anticipate problems before they occur.

Performing Regularly Scheduled Tasks

You perform certain maintenance and administrative tasks routinely in SQL Server, such as backing up databases or importing and exporting data. For example, you might back up a database every Sunday and back up the transaction log every Tuesday and Friday. In addition, you might import data from another data source on a weekly or monthly basis.

Recognizing and Responding to Potential Problems

You need to monitor the databases and transaction logs to make sure that they do not run out of space. For example, if a transaction log is getting full, you can back up and truncate it.

You need to monitor performance conditions. For example, you need to monitor locks to determine whether blocking locks are preventing users from modifying data.

Trainer Materials
Microsoft Certified
Trainer Use Only

Automating Routine Maintenance Tasks

Topic Objective
To introduce the steps that are involved in automating routine maintenance jobs.

Lead-in
The job of an administrator entails various duties that do not change from day to day and can be tedious.

- Automating SQL Server Administration
- Creating Jobs
- Verifying Permissions
- Defining Job Steps
- Determining Action Flow Logic for Each Job Step
- Scheduling Jobs
- Creating Operators to Notify
- Reviewing and Configuring Job History

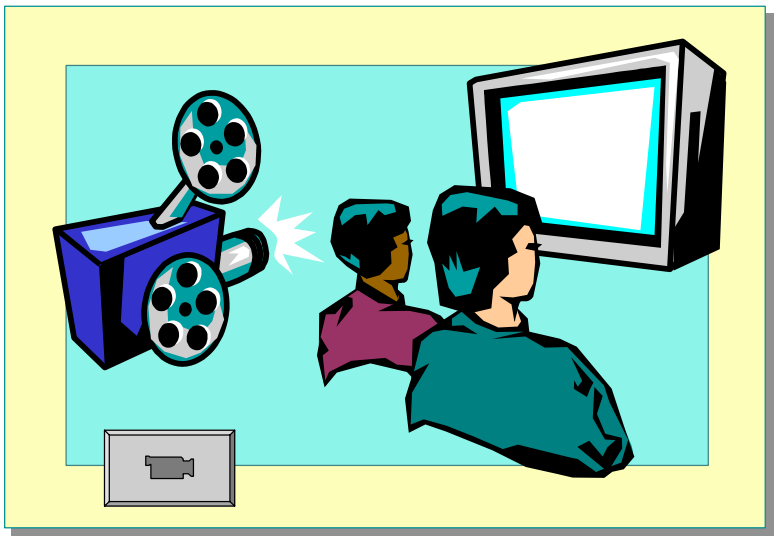
The job of an administrator entails various administrative duties that do not change from day to day and can be tedious. You can automate these routine tasks and configure SQL Server to monitor for certain types of problems before they occur. When you automate routine maintenance tasks, you typically use SQL Server Enterprise Manager to create jobs and operators.

Trainer Materials for Microsoft Certified Trainer Use Only

Multimedia Presentation: Automating SQL Server Administration

Topic Objective
To introduce automating SQL Server administration.

Lead-in
Let's watch a multimedia presentation on automating SQL Server administration.



The multimedia presentation presents the following concepts.

Services Used to Automate SQL Server

SQL Server automation components include the SQL Server Agent, SQL Server, and Microsoft Windows Event Viewer services. These services work together to allow automated administration.

SQL Server Agent Components

The components of SQL Server Agent that enable automation are alerts, jobs, and operators.

Combining Jobs and Alerts

Jobs and alerts are defined separately and can be executed or fired independently. You can combine jobs and alerts to provide additional functionality. For example, if a job fails because of a system error, an alert defined to respond to that error number could execute another job to resolve the problem.

Trainer Materials
Microsoft Certified
Trainer Use Only

Creating Jobs

Topic Objective
To discuss creating jobs.

Lead-in
To automate routine tasks, you must first define the overall characteristics of the job.

- **Ensure That the Job Is Enabled**
- **Specify the Job Owner**
- **Determine Where the Job Will Execute**
- **Create a Job Category**

Delivery Tip
Demonstrate defining a job to back up the **master** database.

You can use SQL Server Enterprise Manager or execute the **sp_add_job** system stored procedure to define a new job. The job definition is stored in the **msdb..sysjobs** system table. This table is maintained in cache to improve performance.

When you define jobs, you should:

Delivery Tip
If you query the **msdb..sysobjects** system table, the **sysjobs** table (and similar tables discussed in this module) is defined as type "U". However, because the **msdb** database is installed as part of SQL Server, these tables contain the sys prefix and are referred to as system tables.

- Ensure that a job is enabled.

  Jobs are enabled by default. If a job is disabled, it cannot execute as scheduled. However, a user can still execute a disabled job manually by starting it in SQL Server Enterprise Manager.
- Specify the owner who is responsible for performing the job.

  By default, the owner is the Windows or SQL Server user logon account creating the job.
- Define whether the job executes on a local server or on multiple remote servers.
- Create job categories to help you organize, filter, and manage many jobs.

  For example, you can create job categories that correspond to the departments in your organization.

Note SQL Server Enterprise Manager can also help you define a core set of automated jobs by using the Database Maintenance Plan Wizard.

Training Materials for Microsoft Certified Trainer Use Only

Verifying Permissions

Topic Objective
To discuss permissions when creating jobs.

Lead-in
If a logon account that is not a member of the **sysadmin** role owns a job, you must verify that the job owner has appropriate permissions to execute the job steps.

- **Executing Transact-SQL Jobs**
  - Execute in the context of the job owner or a specific user
- **Executing Operating System Commands or ActiveX Script Jobs**
  - Members of the **sysadmin** role use the SQL Server Agent login account
  - Job owners that are not members of the **sysadmin** role use a defined domain user account called a proxy account

If a logon account that is not a member of the **sysadmin** role owns a job, you must verify that the job owner has appropriate permissions to execute the job steps.

Executing Transact-SQL Jobs

- All users can initiate Transact-SQL jobs, which operate in the security context of the job owner or a specified user.

Executing Operating System Commands or ActiveX Script Jobs

- For operating system and Microsoft ActiveX® script jobs, steps owned by users who are members of the **sysadmin** role execute in the security context of the SQL Server Service login account. If a logon account that is not a member of the **sysadmin** role owns a job, SQL Server Agent must verify that the job owner has appropriate permissions to execute the job steps.
- By default, users who are not in the **sysadmin** role are not allowed to run operating system commands or ActiveX script jobs.
- However, an administrator can allow users who are not in the **sysadmin** role to execute operating system commands or ActiveX script jobs. In this case, job steps execute in the security context of a defined domain user account called a *proxy account*. You can define this proxy account in SQL Server Enterprise Manager or by using the **xp_sqlagent_proxy_account** extended stored procedure.

Note The domain user login account that the SQL Server service uses must be a member of the Windows local group, **Administrators**, to execute operating system command and ActiveX script job steps when a user who is not a member of the **sysadmin** role owns a job. This allows SQL Server Agent to access and use the proxy account.

Trainer Materials Microsoft Certified Trainer Use Only

Defining Job Steps

Topic Objective
To discuss the types of job steps that can be defined.

Lead-in
You can define job steps to execute Transact-SQL statements, operating system commands, ActiveX scripts, or replication.

- Using Transact-SQL Statements
- Using Operating System Commands
- Using ActiveX Scripts
- Using Replication

You can use SQL Server Enterprise Manager or execute the **sp_add_jobstep** system stored procedure to define each job step. The job step definitions are stored in the **msdb..sysjobsteps** system table.

Delivery Tip
Point out that you can only specify one execution type for each job step.

You can define job steps to execute Transact-SQL statements, operating system commands, ActiveX scripts, or SQL Server replication tasks. However, you can only specify one execution type for each job step.

Trainer Materials for Microsoft Certified Trainer Use Only

Using Transact-SQL Statements

When you define job steps to execute Transact-SQL statements, stored procedures, or extended stored procedures, consider the following guidelines:

- You must identify the database to be used.
- You must include required variables and parameters in the job step.
- You can send the result set of a job step to an output file.

  Output files are often used in troubleshooting to capture any error messages that may have occurred while the statement was executing. You cannot use an output file from one job step as input into a succeeding step.

Using Operating System Commands

When you define a job step to execute an application or operating system command (identified by .exe, .bat, .cmd, or .com file name extensions), you should:

- Identify a process exit code to indicate that the command was successful.
- Include the full path to the executable application in the **Command** text box. The path is required to help SQL Server Agent find the application source.

Using ActiveX Scripts

You can write job steps using ActiveX scripts with languages, such as Microsoft Visual Basic® Scripting Edition (VBScript) or Microsoft Jscript®. You can also use other languages if the language library for the scripting language is installed.

When you create an ActiveX script job step, you must:

- Identify the scripting language in which the job step is written.
- Write or open the active script.

  You can use the **SQLActiveScriptHost** object to print output to the job step history or create objects.

Alternatively, you can externally compile ActiveX scripts (for example, by using Visual Basic) and then run them as operating system commands.

Using Replication

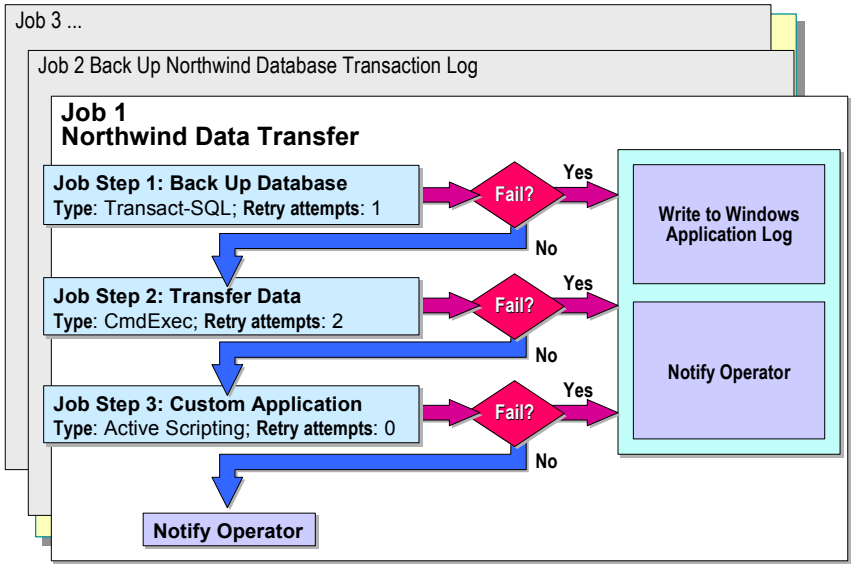
Replication processes are called Agents and are implemented as SQL Server Agent jobs.

Trainer Materials
for Microsoft Certified
Trainer Use Only

Determining Action Flow Logic for Each Job Step

Topic Objective
To discuss action flow logic for each job step.

Lead-in
You can determine the action that SQL Server should take upon the success or failure of each job step.



Delivery Tip
Discuss the flow logic for each job step, including the number of retry attempts.

When creating jobs, a database administrator will want to specify that SQL Server should take an appropriate action if a failure occurs during job execution.

You can accomplish this by determining the action that SQL Server should take upon the success or failure of each job step:

- By default, SQL Server advances to the next job step upon success and stops upon failure of a job step.

  However, job steps can go to any step defined in the job upon success or failure.
- You can specify the number of times that SQL Server should attempt to retry execution of a job step if the step fails. You also can specify the retry intervals (in minutes).

  For example, if the job step requires a connection to a remote server, you could define several retry attempts in case the connection fails.

Delivery Tip
Point out that jobs can be defined to delete themselves after they complete.

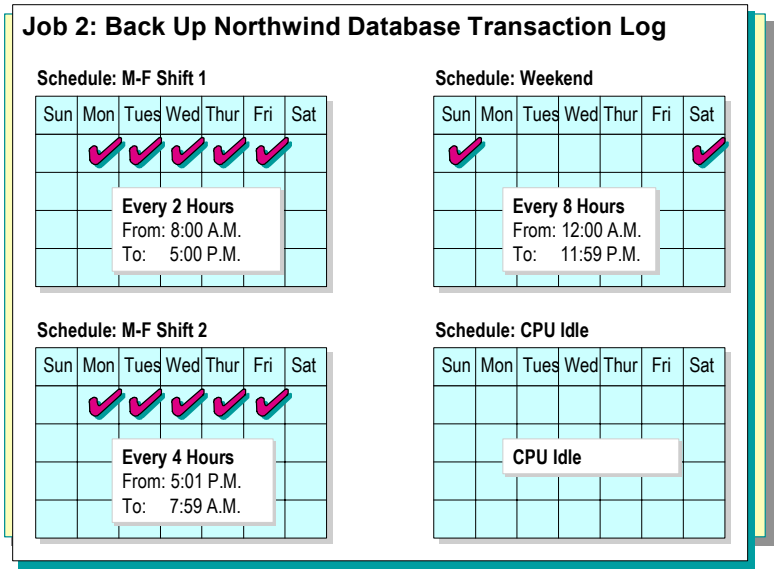
Additionally, if you define a job to run only once, you can specify that the job be deleted when it completes.

Trainer Materials for Microsoft Certified Trainer Use Only

Scheduling Jobs

Topic Objective
To discuss scheduling jobs.

Lead-in
Each job can have multiple schedules.



You can use SQL Server Enterprise Manager or execute the **sp_add_jobschedule** system stored procedure to define each job schedule. The job schedules are stored in the **msdb..sysjobschedules** system table.

Jobs execute according to their defined schedules or in response to alerts. If you are in a multiserver environment, you can define the job to execute on multiple target servers.

A job only executes as scheduled when the specific schedule is enabled. You can schedule jobs to start automatically:

- When SQL Server Agent is started.
- At a specific time (one time only).
- On a recurring basis (daily, weekly, or monthly).
- When the CPU is idle.

Note The domain user account that the SQL Server Agent service uses must be a member of the Windows local group **Administrators** to execute a job when the CPU is idle.

Trainer Materials for Microsoft Certified Trainer Use Only

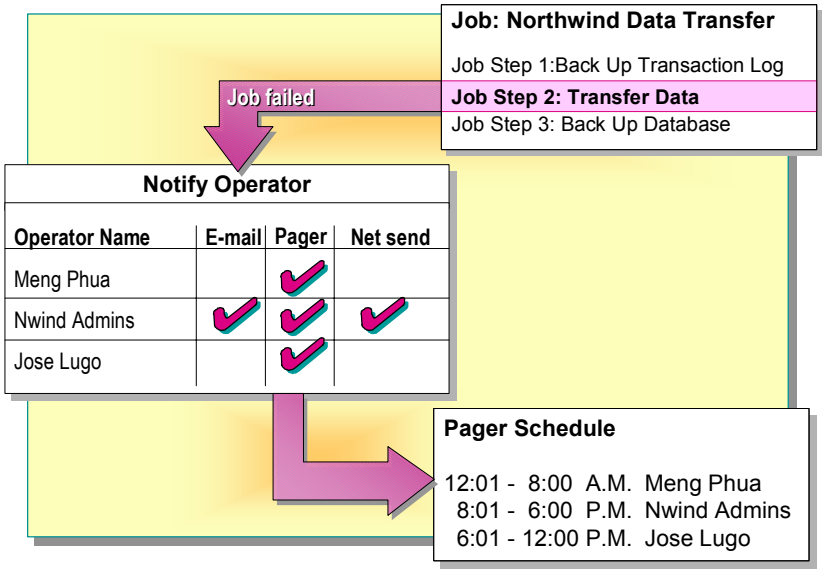
Multiple Schedules

A job can have multiple schedules. For example, you can schedule a job to back up a database transaction log on Monday through Friday, every two hours during peak business hours (8:00 A.M. to 5:00 P.M.), and you can define another schedule to execute the job every four hours during non-peak hours.

Creating Operators to Notify

Topic Objective
To discuss the operators to notify.

Lead-in
You have several options to choose from when you notify operators about a specific event.



Delivery Tip
The **net send** command (illustrated in the slide) is available for users and servers running Windows 2000 or Windows NT only.

You can use SQL Server Enterprise Manager or execute the **sp_add_operator** system stored procedure to define a new operator. The operator definition is stored in the **msdb..sysoperators** system table. When the job completes, or in the event that any of the job steps fail to execute, you can write the event to the Windows application log, delete the job, or notify an operator by pager, e-mail, or a **net send** command.

Trainer Materials for Microsoft Certified Partner Use Only

Creating Operators

When you create operators, you should:

- Use a group e-mail alias to notify more than one individual to respond to potential problems.
- Test each notification method that is used to notify the operator to ensure that the operator is able to receive messages.
- Specify a work schedule for each operator to be notified by pager. If a job that is defined to notify an operator by pager conflicts with the operator's work schedule, the notification fails.
- Use a **net send** command to send messages to network operators and servers running Windows 2000 or Windows NT.

Specifying E-mail Names

When specifying the e-mail name of an operator, you should use the fully qualified e-mail address (smithj@microsoft.com) to avoid potential conflicts when operators with similar names are created at later dates. SQL Server cannot resolve e-mail aliases or display names where the initial characters are identical. For example the e-mail aliases smithj and smithje cannot be resolved.

Troubleshooting Operator Notifications

For each operator, the date and time of the most recent attempts that are sent by each type of notification (e-mail, pager, and **net send** command) is maintained in the **sysoperators** system table. If an operator is not receiving notifications, you should:

- Ensure that the operator is available to receive notifications.
- Ensure that the Messenger service is running on the computer of the operator to be notified by a **net send** command.
- Review the most recent notification attempts to determine the date and time of the last notification.
- Test individual notification methods outside of SQL Server by verifying that you can send e-mail messages, page an operator, or successfully execute a **net send** command.

Trainer Materials for Microsoft Certified Trainer Use Only

Reviewing and Configuring Job History

Topic Objective
To discuss the job history log.

Lead-in
SQL Server Agent captures all job activities and stores the information in the **sysjobhistory** table.

- **Reviewing Individual Job History**
  - Job step result—success or failure
  - Execution duration
  - Errors and messages
- **Configuring the Job History Size**
  - Retain information about each job
  - History overwritten when maximum size is reached

SQL Server Agent captures job step execution status and stores the information in the **msdb..sysjobhistory** system table. You can view history information on individual jobs, as well as configure the size of the job history, by using SQL Server Enterprise Manager.

Reviewing Individual Job History

If a job fails, you can view the job history log to obtain information about each job step, the cause for the failure, and solutions to resolve the problem. Specifically, the job history records:

- The date and time that the job step occurred.
- Whether the job step failed or succeeded.
- The operator who was notified and the notification method.
- The duration of the job step.
- Errors or messages.

Trainer Materials for Microsoft Certified Trainer Use Only

Delivery Tip
Demonstrate changing the job history configuration.

Also demonstrate the history per job and point out the difference when the **Show step details** check box is cleared.

Configuring the Job History Size

If you want to retain information about each job, you should increase the maximum row size for the job history (**sysjobhistory** system table). The job history is overwritten automatically when the row size is reached.

When you configure the size of the job history, consider the following facts:

- By default, the **msdb** database file properties are set to auto grow, and the truncate log on checkpoint database option is turned on.
- By default, the maximum job history size is set at 1,000 rows.
- By default, the maximum job history size for each job is set at 100 rows.
- Rows will be removed from the **sysjobhistory** system table in a first-in, first-out (FIFO) manner when the limits are reached.

Note If SQL Server or SQL Server Agent is shut down while a Transact-SQL statement is executing, the job history contains information about the job step that was in progress. You can specify the maximum number of seconds that SQL Server Agent will wait for a job to finish executing before the service is shut down.

Trainer Materials
for Microsoft Certified
Trainer Use Only

Lab B: Creating Jobs and Operators

Topic Objective
To prepare students for the lab.

Lead-in
In this lab, you will configure a mail profile for SQL Server Agent and create jobs and operators.

Explain the lab objectives.

Objectives

After completing this lab, you will be able to:

- Create operators for notification.
- Create a job with the Create Job Wizard.
- Create a job that consists of multiple job steps.

Prerequisites

Before working on this lab, you must have script files for this lab. These script files are located in C:\Moc\2072A\Labfiles\L05.

This lab requires that you have completed lab A, "Configuring SQL Server."

For More Information

If you require help in executing files, search SQL Query Analyzer Help for "Execute a query".

Other resources that you can use include:

- The **Northwind** database schema.
- SQL Server Books Online.

Scenario

The organization of the classroom is meant to simulate that of a worldwide trading firm named Northwind Traders. Its fictitious domain name is nwtraders.msft. The primary DNS server for nwtraders.msft is the instructor computer, which has an Internet Protocol (IP) address of 192.168.*x*.200 (where *x* is the assigned classroom number). The name of the instructor computer is London.

Trainer Materials for Microsoft Certified Trainer Use Only

The following table provides the user name, computer name, and the IP address for each student computer in the fictitious nwtraders.msft domain. Find the user name for your computer and make a note of it.

User name	Computer name	IP address
SQLAdmin1	Vancouver	192.168.x.1
SQLAdmin2	Denver	192.168.x.2
SQLAdmin3	Perth	192.168.x.3
SQLAdmin4	Brisbane	192.168.x.4
SQLAdmin5	Lisbon	192.168.x.5
SQLAdmin6	Bonn	192.168.x.6
SQLAdmin7	Lima	192.168.x.7
SQLAdmin8	Santiago	192.168.x.8
SQLAdmin9	Bangalore	192.168.x.9
SQLAdmin10	Singapore	192.168.x.10
SQLAdmin11	Casablanca	192.168.x.11
SQLAdmin12	Tunis	192.168.x.12
SQLAdmin13	Acapulco	192.168.x.13
SQLAdmin14	Miami	192.168.x.14
SQLAdmin15	Auckland	192.168.x.15
SQLAdmin16	Suva	192.168.x.16
SQLAdmin17	Stockholm	192.168.x.17
SQLAdmin18	Moscow	192.168.x.18
SQLAdmin19	Caracas	192.168.x.19
SQLAdmin20	Montevideo	192.168.x.20
SQLAdmin21	Manila	192.168.x.21
SQLAdmin22	Tokyo	192.168.x.22
SQLAdmin23	Khartoum	192.168.x.23
SQLAdmin24	Nairobi	192.168.x.24

Trainer Materials for Microsoft Certified Trainer Use Only

Estimated time to complete this lab: 60 minutes

Exercise 1
Creating Operators

In this exercise, you will create an operator to receive notifications from SQL Server Agent.

► To create operators

In this procedure, you will create an operator and verify that a message was received.

1. Log on to the **NWTraders** classroom domain by using the information in the following table.

Option	Value
User name	**SQLAdmin*x*** (where *x* corresponds to your computer name as designated in the nwtraders.msft classroom domain)
Password	**password**

2. Open SQL Server Enterprise Manager.
3. In the console tree, expand **Microsoft SQL Servers**, expand **SQL Server Group**, expand your server, expand **Management**, and then expand **SQL Server Agent**.
4. Right-click **Operators**, and then click **New Operator**.
5. Use the information in the following table to create an operator to receive messages.

Option	Value
Name	Your name
E-mail name	**SQLAdmin*x*@NWTraders.msft** (where *x* maps to your computer name as designated in the NWTraders classroom domain)
net send address	***servername*** (where *servername* maps to your computer name as designated in the NWTraders classroom domain)

6. Click **Test** for both e-mail and **net send** command.
7. Verify that the operator received the messages through e-mail and **net send** command.
8. Click **OK** to close the **New Operator Properties** dialog box.

Trainer Materials for Microsoft Certified Trainer Use Only

Exercise 2
Using the Create Job Wizard

In this exercise, you will use the Create Job Wizard to create a job that is scheduled to back up the **master** database every Monday at 5:00 P.M.

► To use the Create Job wizard

In this procedure, you will use the Create Job wizard to define a job that backs up the **master** database every week on Monday at 5:00 P.M.

1. In SQL Server Enterprise Manager, on the **Tools** menu, click **Wizards**.
2. Expand **Management**, and then double-click **Create Job Wizard**.
3. Use the information in the following table to create a job that is scheduled to back up the **master** database every Monday at 5:00 P.M. Accept the defaults for any options that are not listed.

Option	Value
Type	Transact-SQL command
Database	**master**
Transact-SQL statement	`BACKUP DATABASE master TO DISK = 'C:\Program Files\Microsoft SQL Server\Mssql\Backup\MasterDB.bak' WITH INIT`
Schedule	Recurring, every Monday at 5:00 P.M.
Notifications	E-mail message to yourself **net send** command to yourself
Job name	Master DB Backup

► To execute the job manually

In this procedure, you will verify that your job was created successfully and that notifications were sent.

1. In the console tree, expand **SQL Server Agent**, and then click **Jobs** to display all jobs in the details pane.
2. In the details pane, right-click **Master DB Backup**, and then click **Start Job**.

   This executes the job manually. It will take a few minutes.
3. Click **OK** when the **net send** message appears on your computer screen informing you that the backup job completed successfully.

Trainer Materials for Microsoft Certified Trainer Use Only

4. In the details pane, right-click **Master DB Backup**, and then click **View Job History** to verify that the job was completed successfully.

   What information is displayed in the job history when the **Show step details** checkbox is selected? What information is displayed when this option is cleared?

   Show step details provides information (error messages and notifications) on each job step and the job outcome. When the option is not selected, only the job outcome is displayed.

   _____

   _____

5. Close the **Job History** dialog box.
6. Switch to Outlook to verify that you received an e-mail message.
7. Minimize Outlook.

Trainer Materials for Microsoft Certified Trainer Use Only

Exercise 3
Creating a Job with Multiple Steps

In this exercise, you will use SQL Server Enterprise Manager to create a job that consists of multiple job steps to import data by using an operating system command, and then back up the database by using Transact-SQL.

► To create a job with multiple job steps

In this procedure, you will create a job that imports a text file into the **Products** table in the **Northwind** database, and then backs up the database after the data import successfully completes.

1. In the SQL Server Enterprise Manager console tree, right-click **Jobs**, and then click **New Job**.
2. In the **Name** box, type **Nwind Monthly Data Import**
3. Click the **Steps** tab, click **New**, and then use the information in the following table to create the first job step.

Note Operating system commands must be written on one line, as if you were typing the command at a command prompt.

Option	Value
Step name	Copy new product data
Type	Operating system command (CmdExec)
Command	`C:\Moc\2072A\Labfiles\L05\Import.cmd`
On success action	Go to the next step
Retry attempts	One attempt, at one minute intervals
Output file	C:\Prodcopy.out (overwrite)

4. Click **New**, and then use the information in the following table to create the second job step.

Option	Value
Job step name	Back up Northwind DB
Type	Transact-SQL
Database	**Northwind**
Command	`BACKUP DATABASE Northwind TO DISK = 'C:\Program Files\Microsoft SQL Server\Mssql\Backup\Nwind.bak' WITH INIT`
On success action	Quit the job reporting success
Output file	C:\Prodcopy.out (append)

Trainer Materials for Microsoft Certified Trainer Use Only

5. Use the information in the following table to schedule the Nwind Monthly Data Import job and specify notifications. Accept the defaults for any options that are not listed.

Option	Value
Schedule name	First day of the month
Schedule	Recurring, monthly on the first day at 1:00 A.M.
Notifications	E-mail message to yourself When the job fails
	net send command sent to yourself When the job completes
	Windows application log When the job fails

6. Start the Nwind Monthly Data Import job.

   After a few moments, a **net send** message appears, indicating that the job succeeded.

7. Click **OK** to close the **Messenger Service** dialog box.

► To simulate a failure and verify that a job step failed

In this procedure, you will rename the data import file to simulate a failure that causes the first job step to fail.

1. Open Microsoft Windows Explorer.
2. Using Notepad, open C:\Prodcopy.out, review its contents, and then close the file.
3. Rename the C:\Moc\2072A\Labfiles\L05\Prods.txt file as **Prodsnew.txt**
4. Switch to SQL Server Enterprise Manager.
5. Start the Nwind Monthly Data Import job.
6. When you receive the **net send** message indicating that the job failed, click **OK** to close the **Messenger Service** dialog box.
7. Review the job history, looking at the step details.

   What did you notice in the history?

   The job failed after the first job step failed. The job executed again according to the retry interval. The error message stated that the bcp host data file could not be opened.

Trainer Materials for Microsoft Certified Trainer Use Only

8. Open Event Viewer and review the application log to confirm that the job failure was logged.

   What information is displayed in the log?

   SQL Server Agent is the event source. The event category is Job Engine. The description includes the name of the job, the user who invoked the job, the job failure message, and the step number of the last step to run.

   _____

   _____

9. Close Event Viewer.
10. Switch to Outlook to confirm that you received an e-mail message notifying you that the job failed.
11. Open Notepad and review the contents of C:\Prodcopy.out.

    What information is displayed in the output file?

    The SQLState error number, native error number, and error message "Unable to open BCP host data-file."

    _____

    _____

12. Close the file.
13. Switch to SQL Server Enterprise Manager and disable the Nwind Monthly Data Import Job.

Trainer Materials for Microsoft Certified Trainer Use Only

Creating Alerts

Topic Objective
To introduce creating alerts.

Lead-in
When you create alerts, SQL Server allows you to respond to potential problems.

- Using Alerts to Respond to Potential Problems
- Writing Events to the Application Log
- Creating Alerts to Respond to SQL Server Errors
- Creating Alerts on a User-defined Error
- Responding to Performance Condition Alerts
- Assigning a Fail-Safe Operator

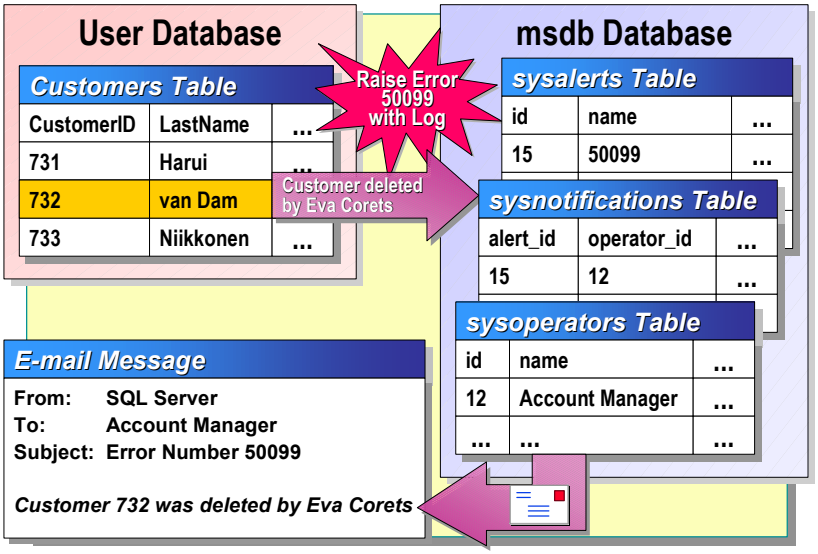
SQL Server allows you to respond to potential problems by creating alerts to respond to SQL Server errors, user-defined errors, or performance conditions. You can also create a fail-safe operator in the event that a pager notification fails to contact an operator.

Trainer Materials for Microsoft Certified Trainer Use Only

Using Alerts to Respond to Potential Problems

Topic Objective
To describe how alerts are processed.

Lead-in
Alerts respond to SQL Server errors or user-defined errors that have been written to the Windows application log.



Alerts respond to SQL Server or user-defined errors (events) that have been written to the Windows application log. SQL Server errors are raised in response to predefined problems, such as insufficient user permissions to modify a table or the transaction log becoming full. To raise user-defined messages, the database application (typically, a stored procedure or trigger) must call the RAISERROR statement.

Scenario Example

An account manager wants to be notified by e-mail any time that a customer is removed from the database. She also wants to know the name of the employee who deleted the customer in the event that subsequent action is necessary.

Alert Process

The following steps illustrate the sequence of events that fires an alert to respond to the account manager's request:

1. Eva Corets, a customer service representative, removes customer Van Dam from the **Customers** table. The **RemoveCustomer** stored procedure is executed, which raises error number 50099.
2. The error (event) is written to the Windows application log.
3. The Windows application log notifies SQL Server Agent that an event has occurred.
4. SQL Server Agent then compares the error to defined alerts in the **msdb..sysalerts** system table, which is maintained in cache.
5. SQL Server Agent processes the alert response by:
   a. Reviewing the **msdb..sysnotifications** system table to send an e-mail message.
   b. Reviewing the **msdb..sysoperators** system table that identifies to whom to send the notification.

Trainer Material for Microsoft Certified Trainer Use Only

Writing Events to the Application Log

Topic Objective
To discuss writing events to the application log.

Lead-in
SQL Server writes events to the Windows application log when any of the following occurs.

- **SQL Server Errors Severity Levels Between 19 and 25**
- **sp_addmessage or sp_altermessage System Stored Procedures**
- **RAISERROR WITH LOG Statement**
- **xp_logevent Extended Stored Procedure**

Delivery Tip
In Windows 98, the role of the Event Viewer is substituted by a SQL Server Profiler trace.

When the SQL Server Agent service starts, it registers with the Event Viewer service and connects to the SQL Server service. This allows SQL Server Agent to be informed when SQL Server events are written to the Windows application log. SQL Server Agent then compares the events with the cached jobs and alerts to determine whether a defined action must be performed.

SQL Server writes events to the Windows application log when:

- SQL Server errors with severity levels between 19 and 25 occur.
- Error messages are defined to be written to the Windows application log with the **sp_addmessage** or **sp_altermessage** system stored procedure.
- The RAISERROR WITH LOG statement is executed.
- The **xp_logevent** extended stored procedure is executed.

Trainer Materials for Microsoft Certified Trainer Use Only

Creating Alerts to Respond to SQL Server Errors

Topic Objective
To discuss creating alerts to respond to SQL Server errors.

Lead-in
You can use alerts to respond to SQL Server errors.

- **Defining Alerts on SQL Server Error Numbers**
  - Must be written to the Windows application log
  - System-supplied or user-defined
- **Defining Alerts on Error Severity Levels**
  - Severity levels 19 through 25 are automatically logged
  - Configure event forwarding

When you create an alert to fire a response when a SQL Server error occurs, you can specify a single error number, such as 9002, or all errors of a specific severity level, such as 17.

Note At most one alert will fire for a given event. SQL Server Agent will fire the most specific alert that is defined. For example, if you have an alert defined on severity level 17 errors in all databases, and you also have an alert defined on error 9002 (which is also severity 17), error 9002 will fire.

Delivery Tip
The **sp_add_alert** system stored procedure includes a parameter, @raise_snmp_trap. This is included for backward compatibility only.

Defining Alerts on SQL Server Error Numbers

You can use SQL Server Enterprise Manager or execute the **sp_add_alert** system stored procedure to define a new alert. The alert definition is stored in the **msdb..sysalerts** system table. This table is maintained in cache to improve performance. When you define alerts on a SQL Server error number, consider the following facts and guidelines:

- The error numbers must be written to the Windows application log.
- You can define alerts on any SQL Server system-supplied or user-defined error number that is stored in the **master..sysmessages** system table.
- You can define more than one alert for a SQL Server error number. However, each alert must be limited to a specific database or must apply to all databases.

  For example, to respond to error number 9002 in both the **Payroll** and **Customer** databases, you could create two separate alerts. Or, you could create one alert to respond to error number 9002 in all databases.

Trainer Materials
Microsoft Certified
Trainer Use Only

- When you create an alert that applies to all databases, ensure that the error message provides a sufficiently detailed explanation. Substitution parameters that provide the database name, for example, typically accomplish this.
- You can define a specific string in the error message text in addition to the error number.

  For example, you can create an alert to notify you when someone is attempting to log in as the system administrator. To create this alert, specify error number 18456 (Login failed for user '%s') and the message string "sa".

Delivery Tip
Suggest that students use predefined alerts for these severity levels.

Demonstrate how to edit the predefined alerts by specifying an operator to notify and then modifying the name of the alert to remove the word demo.

Defining Alerts on Error Severity Levels

When you define alerts on error severity levels as a condition, consider the following facts and guidelines:

- SQL Server errors with severity levels 19 through 25 are automatically written to the Windows application log.
- Severity levels 20 through 25 are fatal errors. You always should define an operator to be notified when SQL Server errors with these severity levels occur.
- SQL Server provides predefined alerts that you can use. You should modify the predefined alerts on fatal errors by defining an operator to notify and renaming the alert to remove the word demo.
- You can create an alert to be fired when an error of a specific severity level occurs on all databases or on a particular database.
- You can define a specific string in the error message text in addition to the severity level. For example, you can create an alert to notify you when severity level 17 occurs in any database with the message string "disk space".

Delivery Tip
Event forwarding is only available when SQL Server is installed on Windows 2000 or Windows NT.

Configure Event Forwarding

You can configure SQL Server Agent to designate a server to receive all or only unhandled event messages that meet or exceed a specified error severity level. You may want to forward events to a server that has less traffic than other servers that are in the domain.

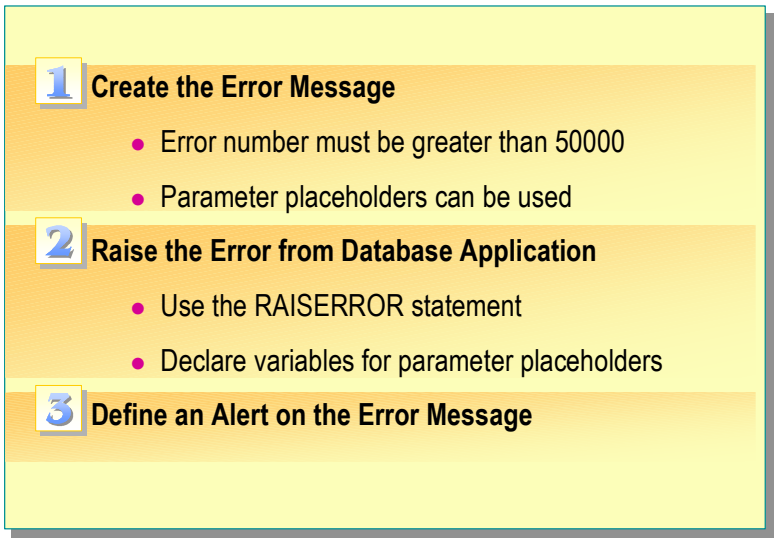
For example, you can configure errors with severity levels 18 or above to be forwarded to the Accounting server. If an error with severity level 19 occurs on your server, the event is automatically forwarded to the Accounting server to address the problem.

Trainer Materials
Microsoft Certified
Trainer Use Only

Creating Alerts on a User-defined Error

Topic Objective
To discuss creating user-defined error messages for alerts.

Lead-in
When you create alerts, you also can specify user-defined (customized) error messages for individual database applications.



Delivery Tip
Demonstrate creating a user-defined message with SQL Server Enterprise Manager.

When you create alerts, you also can specify user-defined (customized) error messages for individual database applications that allow you to define solutions to avoid potential problems before they occur.

For example, you could create a user-defined error message to be raised from an update trigger on the **Inventory** table. When a column in the **Inventory** table is updated, indicating that inventory levels have fallen below 25 percent for a particular product, the trigger will raise your user-defined error. You then could define an alert on the error message that executes a job to reorder inventory and sends an e-mail message to the purchasing agent.

To create an alert on a user-defined error, you first must create the error message. Then you must raise the error from your database application and define an alert on the error message.

Trainer Materials for Microsoft Certified Trainer Use Only

Create the Error Message

To create user-defined errors, you can use SQL Server Enterprise Manager or the **sp_addmessage** system stored procedure. When you create user-defined errors, consider the following facts:

- User-defined errors numbers must be greater than 50000. Error numbers less than 50000 are reserved for predefined SQL Server system errors.
- All user-defined errors are stored in the **sysmessages** system table in the **master** database.
- Error messages can include parameters to capture specific details so that adequate information is provided to detail status or to troubleshoot the problem.

- SQL Server error messages are displayed in the language that is selected during setup. If you administer a multiple language SQL Server environment, you can also create user-defined messages for other languages.
- You must write the error message to the Windows application log if you plan to fire an alert on the message.

Example 1

This example creates a user-defined error message (number 50099) that is logged in the Windows application log (specified by **true** in the syntax below) when the error occurs.

The *%d* parameter is replaced with the customer number that is deleted, and the *%s* parameter is replaced with the user name that executed a DELETE statement on the **Customers** table.

Delivery Tip
The **true** option specifies that the message should be written to the Windows application log.

```
EXEC sp_addmessage 50099, 16, 'Customer %d was deleted by %s',
'us_english','true'
```

Raise the Error from Database Application

Use the RAISERROR statement in a stored procedure or trigger to raise an error. The RAISERROR statement returns a user-defined error message and sets a system flag (in the **@@error** system function) to record that an error has occurred.

Example 2

This example illustrates the partial script that is used to create a stored procedure that deletes a customer from the **Customers** table. The stored procedure raises error number 50099 (which was created above), with the RAISERROR statement, which substitutes the appropriate values for the deleted customer number and user name that executed the stored procedure.

Partial Syntax

```
CREATE PROCEDURE removecustomer
@CustomerID varchar(5) = NULL
AS
.
DECLARE @username varchar(60)
SET @username = suser_sname()

BEGIN TRANSACTION
  DELETE Customers
  WHERE CustomerID = @CustomerID
  RAISERROR (50099, 16, 1, @CustomerID, @username)
COMMIT TRANSACTION
```

A client application will call the **RemoveCustomer** stored procedure and pass the customer number variable that the user entered.

Trainer Materials Microsoft Certified Trainer Use Only

Define an Alert on the Error Message

An alert is created on error 50099 to send an e-mail message that includes the text of the error message to the account manager.

When a user executes the **RemoveCustomer** stored procedure, error 50099 is raised and recorded in the Windows application log. The alert on the error number is fired and sends an e-mail message that includes the text of the error message to the account manager.

Result

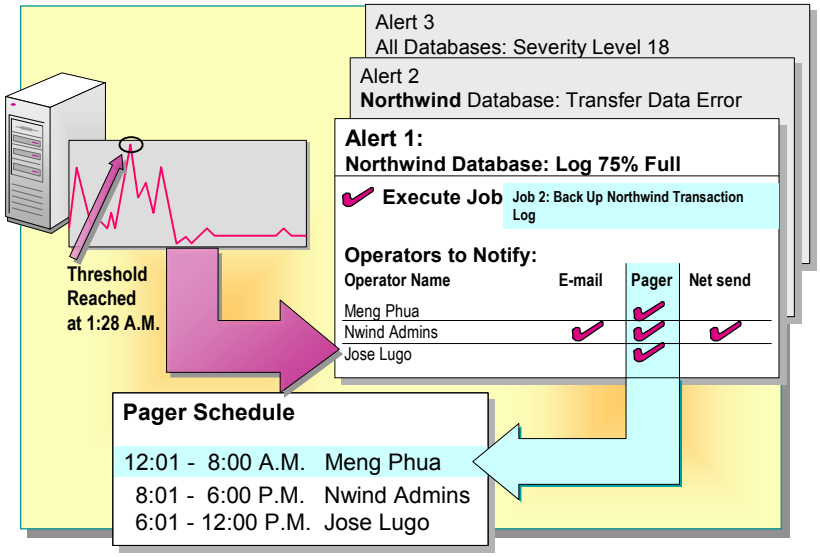
```
Error: 50099, Severity: 16, State 1
Customer 732 was deleted by ACCOUNTING\evacorets
```

Trainer Materials for Microsoft Certified Trainer Use Only

Responding to Performance Condition Alerts

Topic Objective
To explain defining alerts that respond to Windows System Monitor thresholds.

Lead-in
In addition to using alerts to respond to SQL Server errors, you can use them to respond to SQL Server performance conditions.



In addition to using alerts to respond to SQL Server errors, you can use them to respond to SQL Server performance conditions such as those that are defined in the Windows System Monitor. When the condition value is exceeded, an alert is fired.

For example, you can create a performance condition alert that fires when the transaction log in the **Northwind** database has exceeded 75 percent capacity. The alert response could execute a job to back up the transaction log and notify the database administrator.

Note The Windows System Monitor does not need to be running on your SQL Server for you to use performance condition alerts.

You can create performance condition alerts on SQL Server resources that include some of the following objects:

- Access Methods
- Buffer Manager
- Cache Manager
- Databases
- Locks
- SQL Statistics

Reducing the Delay Between Responses

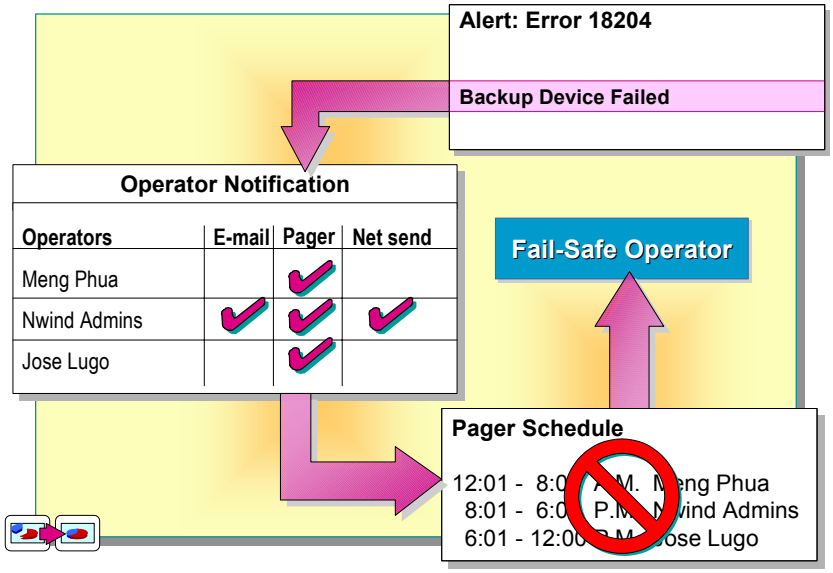
Performance data is sampled periodically (a few times each minute), which can cause a few seconds delay between the threshold being reached and the performance alert firing. Therefore, you may need to reduce the time delay between responses, or modify the threshold of your performance condition when the alert response is required to correct the condition quickly.

Trainer Materials for Microsoft Certified Trainer Use Only

Assigning a Fail-Safe Operator

Topic Objective
To discuss creating a fail-safe operator

Lead-in
In the event that pager notifications for defined alerts fail, you can notify a fail-safe operator.



Delivery Tip
This slide is animated. Refer to the Instructor Notes if you require help in navigating through this slide.

Note that the information in the illustration is fictitious and is provided as a teaching example.

You can assign a fail-safe operator to respond to an alert when pager notifications to defined operators fail. For example, if Meng Phua is off-duty on Thursday when the alert for error 18204 is fired at 1:30 A.M., the fail-safe operator will be contacted.

A fail-safe operator will be notified when all of the following occurs:

- The alert has pager notifications defined for the response.
- All the operators to be paged are not on-duty.
- A fail-safe operator is defined.
- The SQL Server Agent mail session is started.

  The SQL Server Agent mail session is required for the fail-safe operator to be notified by e-mail or pager.

When you assign a fail-safe operator, consider the following facts:

- The fail-safe operator information is cached so it is not dependent upon the connection to the MSDB database.
- You can have only one fail-safe operator.
- You cannot delete an operator that has been designated as the fail-safe operator. However, you can remove the fail-safe operator assignment and then delete the operator.

Trainer Materials for Microsoft Certified Trainer Use Only

Troubleshooting SQL Server Automation

Topic Objective
To discuss troubleshooting automated tasks.

Lead-in
Use these guidelines to help you troubleshoot potential problems with your automated administrative jobs and alerts.

- Verify That SQL Server Agent Is Started
- Verify That the Job, Schedule, Alert, or Operator Is Enabled
- Ensure That the Proxy Account Is Enabled
- Review the Error Logs
- Review the History
- Verify That the Mail Client Is Working Properly

If your automated jobs, alerts or notifications are not working properly, use the following guidelines to help isolate and solve the problem.

Verify That SQL Server Agent Is Started

If SQL Server Agent has stopped for any reason, it is not able to execute your defined jobs and alerts.

- Verify that the login account for SQL Server Agent is mapped to the SQL Server **sysadmin** role.
- Verify the login account password.

Verify That the Job, Schedule, Alert, or Operator Is Enabled

If jobs are not executing as scheduled, alerts are not being fired, or operators are not receiving notifications, verify that the job, schedule, alert, or operator is enabled.

For example, if you disable an alert, the alert is inactive but maintains its properties. You can reuse this alert if necessary by enabling it. However, if you delete an alert, all of the associated notifications for that alert are also removed.

Ensure That the Proxy Account Is Enabled

Job owners who are not members of the **sysadmin** role execute operating system and ActiveX script job steps in the security context of the defined Windows domain user account. If the operating system command or ActiveX script job steps fail to execute, verify that:

- Users have permission to run these types of jobs.
- The domain user account used as the proxy account has permission to run jobs.

Trainer Materials for Microsoft Certified Trainer Use Only

Review the Error Logs

Review error messages in the Windows application log and the SQL Server Agent and SQL Server error logs to troubleshoot the source of your problem.

- SQL Server Agent error log records SQL Server Agent errors. By default, all SQL Server Agent errors and warnings are recorded in the error log. You also can record informational execution trace messages:
  - Each time that SQL Server Agent is stopped and restarted, a new error log is created.
  - You can view the current error log by using SQL Server Enterprise Manager or any text editor. Up to nine previous versions of the error log are saved in the C:\Program Files\Microsoft SQL Server\Mssql\Log folder.
  - You can define an error message pop-up recipient to send a net send command when errors are logged into the SQL Server Agent error log.
- The Windows application log can show no events existing for SQL Server to process if the maximum size is too small or defined to be overwritten frequently. To avoid losing event information about SQL Server, increase the maximum log size for the Windows application log.
- SQL Server error log should be reviewed. By comparing the dates and times for events among the SQL Server error log, the SQL Server Agent error log, and the Windows application log, you can narrow down the list of probable causes.

Review the History

After you review the SQL Server Agent error log, you also may want to review history information on an alert, operator, or a job. Alert and operator history captures only the most recent activity.

Ensure that the **msdb** database file size and growth size match the number of rows maintained in the **sysjobhistory** system table. If the file and growth sizes are not configured correctly, you can run out of disk space or database space, which will prevent jobs from executing.

Verify That the Mail Client Is Working Properly

If e-mail or pager notifications are not working, verify that the mail client is working properly. To do so, log in to the mail client by using the SQL Server Agent domain user account and send an e-mail or pager notification to an operator.

Training Materials for Microsoft Certified Partners - Only

Troubleshooting Alerts

Topic Objective
To discuss looping alerts and how to prevent them.

Lead-in
Because SQL Server Agent depends on and monitors SQL Server events, SQL Server Agent can become caught in an endless loop of firing the same alert repeatedly.

- **Factors That Cause an Alert Processing Backlog**
  - Windows application log is full
  - CPU use is unusually high
  - Number of alert responses is high
- **Resolving Alert Processing Backlog**
  - Disable the alert temporarily
  - Increase delay between responses
  - Correct global resource problem
  - Clear the Windows application log

Because SQL Server Agent depends on and monitors SQL Server events, SQL Server Agent can become caught in an endless loop of firing the same alert repeatedly. This generally occurs when SQL Server runs out of an essential global resource and an alert has been defined on this event.

Factors That Cause an Alert Processing Backlog

An alert processing backlog may occur if the Windows application log rapidly fills up with the same error, if the processor use is unusually high, or if the number of alert responses is high. Under these conditions, the delay increases between when the event appears in the Windows application log and when SQL Server Agent responds to that event.

Resolving Alert Processing Backlog

When the number of alerts exceeds the SQL Server Agent alert processing rate, a backlog is created. To resolve the backlog and prevent it from recurring:

- Disable the alert temporarily.
- Increase the amount of time between responses for each alert.
- Correct the global resource problem to prevent recurring alerts from using all of your resources.
- Clear the backlog of events from the Windows application log. Clearing the Windows application log deletes all events from the error log, including those unrelated to SQL Server.
- Configure an error to be non-alert generating so that SQL Server Agent will not fire the alert when the error occurs.

Delivery Tip
Global resources include locks and open objects. These are dynamically configured but could have been manually configured.

Important You can configure an error not to generate an alert within the Windows registry only. You should use this solution only if it is your last remaining option.

Trainer Materials for Microsoft Certified Trainer Use Only

Lab C: Creating Alerts

Topic Objective
To prepare students for the lab.

Lead-in
In this lab, you will create user-defined error messages and performance condition alerts.

Explain the lab objectives.

Objectives

After completing this lab, you will be able to:

- Create an alert by using the Create Alert Wizard.
- Create an alert on a user-defined message.
- Create a performance condition alert.

Prerequisites

Before working on this lab, you must have script files for this lab. These script files are located in C:\Moc\2072A\Labfiles\L05.

This lab requires that you have completed lab A, "Configuring SQL Server" and lab B, "Creating Jobs and Operators."

For More Information

If you require help in executing files, search SQL Query Analyzer Help for "Execute a query".

Other resources that you can use include:

- The **Northwind** database schema.
- SQL Server Books Online.

Scenario

The organization of the classroom is meant to simulate that of a worldwide trading firm named Northwind Traders. Its fictitious domain name is nwtraders.msft. The primary DNS server for nwtraders.msft is the instructor computer, which has an Internet Protocol (IP) address of 192.168.*x*.200 (where *x* is the assigned classroom number). The name of the instructor computer is London.

Trainer Materials for Microsoft Certified Trainer Use Only

The following table provides the user name, computer name, and the IP address for each student computer in the fictitious nwtraders.msft domain. Find the user name for your computer and make a note of it.

User name	Computer name	IP address
SQLAdmin1	Vancouver	192.168.x.1
SQLAdmin2	Denver	192.168.x.2
SQLAdmin3	Perth	192.168.x.3
SQLAdmin4	Brisbane	192.168.x.4
SQLAdmin5	Lisbon	192.168.x.5
SQLAdmin6	Bonn	192.168.x.6
SQLAdmin7	Lima	192.168.x.7
SQLAdmin8	Santiago	192.168.x.8
SQLAdmin9	Bangalore	192.168.x.9
SQLAdmin10	Singapore	192.168.x.10
SQLAdmin11	Casablanca	192.168.x.11
SQLAdmin12	Tunis	192.168.x.12
SQLAdmin13	Acapulco	192.168.x.13
SQLAdmin14	Miami	192.168.x.14
SQLAdmin15	Auckland	192.168.x.15
SQLAdmin16	Suva	192.168.x.16
SQLAdmin17	Stockholm	192.168.x.17
SQLAdmin18	Moscow	192.168.x.18
SQLAdmin19	Caracas	192.168.x.19
SQLAdmin20	Montevideo	192.168.x.20
SQLAdmin21	Manila	192.168.x.21
SQLAdmin22	Tokyo	192.168.x.22
SQLAdmin23	Khartoum	192.168.x.23
SQLAdmin24	Nairobi	192.168.x.24

Trainer Materials for Microsoft Certified Trainer Use Only

Estimated time to complete this lab: 45 minutes

Exercise 1
Using the Create Alert Wizard

In this exercise, you will create an alert to respond to errors with severity level 18 by using the Create Alert Wizard.

► To create an alert by using the Create Alert Wizard

In this procedure, you will use the Create Alert Wizard to create an alert that is defined to send an e-mail message and a **net send** command to yourself when an error with severity level 18 occurs.

1. Log on to the **NWTraders** classroom domain with the information in the following table.

Option	Value
User name	**SQLAdmin*x*** (where *x* corresponds to your computer name as designated in the nwtraders.msft classroom domain)
Password	**password**

2. Open SQL Server Enterprise Manager, expand **Microsoft SQL Servers**, expand **SQL Server Group**, and then expand your server.
3. On the **Tools** menu, click **Wizards**.
4. Expand **Management**, and then double-click **Create Alert Wizard**.
5. Create an alert with the information in the following table. Accept the defaults for any items that are not listed.

Option	Value
For any error severity	018 – NonFatal Internal Error
Database name	All databases
Notify operators	Select yourself as the operator to receive notification by e-mail and **net send** command
Include error message text in:	E-mail and Net Send
Alert name	Severity level 18 errors

6. In the console tree, expand **Management**, expand **SQL Server Agent**, and then click **Alerts**.
7. In the details pane, verify that the alert was created.
8. Right-click the Severity level 18 errors alert, and then click **Properties** to review the alert definition.
9. Click **OK** to close the **Severity Level 18 Errors Properties** dialog box.

Trainer Materials for Microsoft Certified Trainer Use Only

Exercise 2
Creating an Alert on a User-defined Error Message

In this exercise, you will create a user-defined (custom) error message that indicates when the stock units reach the reorder level for any product.

► To create a user-defined error message

In this procedure, you will use SQL Server Enterprise Manager to create a user-defined error message.

1. In the console tree, right-click your server, point to **All Tasks**, and then click **Manage SQL Server Messages**.
2. On the **Messages** tab, click **New**.
3. Use the information in the following table to create a user-defined message that indicates that the number of stock units for a particular product has reached the reorder level. Accept the defaults for any items that are not listed.

Option	Value
Error number	50001
Severity	010 – Information
Message text	The units in stock for %s has reached %d. Please reorder
Always write to Windows eventlog	Select this check box

► To create an alert for a user-defined error message

In this procedure, you will create an alert called Reorder Inventory that sends an e-mail message to the warehouse manager when error number 50001 occurs.

1. In the console tree, expand **Management**, expand **SQL Server Agent**, right-click **Alerts**, and then click **New Alert**.
2. Create an alert with the information in the following table. Accept the defaults for any items that are not listed.

Option	Value
Alert name	Reorder Inventory
Error number	50001
Database	**Northwind**
Operators to notify	Select yourself as the operator to receive notification by E-mail and Net Send
Include alert error text in:	E-mail and Net Send

3. Verify that the alert was created.

   The Reorder Inventory alert should appear in the details pane.

Trainer Materials for Microsoft Certified Trainers Use Only

► To raise a user-defined error message

In this procedure, you will use SQL Query Analyzer to create and execute a stored procedure that will raise error 50001 to test that the Reorder Inventory alert works as expected.

1. Open SQL Query Analyzer and, if requested, log in to the (local) server with Microsoft Windows® authentication.

   You have permission to log on to and administer SQL Server because you are logged as **SQLAdmin*x***, which is a member of the Windows 2000 local group, Administrators. All members of this group are automatically mapped to the SQL Server **sysadmin** role.

2. Open C:\Moc\2072A\Labfiles\L05\Reorder.sql, review its contents, and then execute it.

   This script creates a stored procedure named reorder.

3. Open a new query window and execute the stored procedure by supplying any valid product ID value. For example:

   ```
   USE Northwind
   EXEC ReOrder
   @prodid = 2
   ```

   Did you receive the defined notifications in response to the alert?

   Yes. A net send command and an e-mail message were received.

Trainer Materials for Microsoft Certified Trainer Use Only

Exercise 3
Creating Performance Condition Alerts

In this exercise, you will execute a script that creates a job to back up the **Northwind** database. You will then execute a script that alters the size of the **Northwind** database transaction log. Next, you will create a performance condition alert in SQL Server Enterprise Manager on the Percent Log Used counter that is defined to notify you when the **Northwind** database transaction log has reached 60 percent capacity. Finally, you will verify that the condition alert works as expected.

► To execute a script that creates a job to back up the Northwind transaction log

In this procedure, you will execute a script that creates a job that backs up the **Northwind** transaction log every four hours.

1. Open C:\Moc\2072A\Labfiles\L05\Nwlog.sql, review its contents, and then execute it.
2. Switch to SQL Server Enterprise Manager.
3. Right-click **Jobs**, and then click **Refresh**.
4. Click **Jobs**, and then, in the details pane, verify that the Backup Northwind Log job was created.
5. Edit the job properties to specify that you should be notified by e-mail and **net send** command whenever the job completes and write the event to the Windows application log when the job fails.

► To disable auto grow, change the database recovery model, and back up the Northwind database

In this procedure, you will alter the **Northwind** database so that the transaction log does not auto grow, set the database recovery model to full, and then back up the **Northwind** database.

1. In the console tree, expand **Databases**, right-click **Northwind**, and then click **Properties**.
2. On the **Transaction Log** tab, clear the **Automatically grow file** check box.
3. On the **Options** tab, change the database recovery model to full, and then click **OK** to the close the **Northwind Properties** dialog box.
4. Switch to SQL Query Analyzer.
5. Open C:\Moc\2072A\Labfiles\L05\NWBackup.sql, review its contents, and then execute it.

Trainer Materials for Microsoft Certified Trainer Use Only

► To create a performance condition alert

In this procedure, you will create a performance condition alert when the Percent Log Used counter rises above 60 percent for the **Northwind** database.

1. In the SQL Server Enterprise Manager console tree, expand **Management**, and then expand **SQL Server Agent**.
2. Right-click **Alerts**, and then create a new alert by using the information in the following table.

Option	Value
Alert name	Northwind Log 60% Full
Alert type	SQL Server performance condition alert
Object	SQLServer:Databases
Counter	Percent Log Used
Instance	**Northwind**
Alert if counter	Rises above 60
Execute job	Select the **Execute job** check box, and click **Backup Northwind Log**
Operators to notify	Yourself by e-mail and Net Send
Delay between responses	0 minutes, 0 seconds

3. Review the schedule properties of the Backup Northwind Log job.

   What schedule exists?

   Two schedules exist. One schedule runs every four hours on a recurring schedule, and the other schedule runs in response to the performance condition alert, Northwind Log 60% Full.

   _____

   _____

Trainer Materials for Microsoft Certified Trainer Use Only

► To monitor the percent log used with Windows System Monitor

In this procedure, you will create a new chart in Windows System Monitor to monitor the percent log used.

1. Click **Start**, point to **Programs**, point to **Administrative Tools**, and then click **Performance**.
2. In the console tree, verify that System Monitor is selected.
3. On the toolbar, click **Add**, click the **SQLServer:Databases** object, click the **Percent Log Used** counter, click the **Northwind** instance, and then click **Add**.
4. Click **Close** to close the **Add Counters** dialog box.

► To test the performance condition alert

In this procedure, you will review and execute a script that generates activity in the **Northwind** database to fill up the transaction log. Then, you will verify that SQL Server Agent fired the alert and executed the job as defined.

1. Switch to SQL Query Analyzer.
2. Open C:\Moc\2072A\Labfiles\L05\Fulltlog.sql, review its contents, and then execute it.

   This script generates activity in the **Northwind** database and fills up the transaction log.
3. Switch to Windows System Monitor to observe the transaction log activity.
4. When the alert is fired, you will receive a **net send** message.

   Notice that when the job executes, backing up the transaction log, and reduces the percent of the transaction log that is used.
5. Close the **Messenger Service** dialog boxes, and then switch to SQL Query Analyzer.
6. On the toolbar, click **Cancel Query** to stop the execution of the script.
7. Open Outlook and review your Inbox to ensure that you received one or more e-mail messages.
8. Switch to SQL Server Enterprise Manager.
9. Review the job history on Backup Northwind Log.
10. Review the alert history on Northwind Log 60% Full.

    Refresh the alert to show the most recent date and time that the alert was fired.
11. In the console tree, expand **Databases**, right-click **Northwind**, and then click **Properties**.
12. On the **Transaction Log** tab, select the **Automatically grow file** check box.
13. On the **Options** tab, change the recovery model to simple, and then click **OK** to close the **Northwind Properties** dialog box.

Trainer Materials for Microsoft Certified Trainer Use Only

If Time Permits
Assigning a Fail-Safe Operator

In this exercise, you will define yourself as the fail-safe operator for your server. You will also execute a script that creates an alert on error 9002 (transaction log is full) and a multistep job that addresses the alert. Next, you will modify the **Northwind** database properties and execute a script that will fill the transaction log. Finally, you will verify that the alert executed the job to clear the transaction log and that the fail-safe operator was notified.

► To assign a fail-safe operator

In this procedure, you will designate yourself as a fail-safe operator for your server.

1. In the console tree, expand **Management**, right-click **SQL Server Agent**, and then click **Properties**.
2. In the **SQL Server Agent Properties** dialog box, click the **Alert System** tab.
3. In the **Failsafe Operator** list, define yourself as the fail-safe operator for your server to be notified by e-mail and **net send** command.
4. Click **OK** to close the **SQL Server Agent Properties** dialog box.

► To execute a script that creates an alert

In this procedure, you will execute a script that creates an alert on error 9002 and designates the Northwind backup log job to execute in response to the alert firing.

1. Switch to SQL Query Analyzer.
2. Open C:\Moc\2072A\Labfiles\L05\Alert.sql, review its contents, and then execute it.

   You will receive notification that non-existing steps were referenced. This is expected behavior.
3. Switch to SQL Server Enterprise Manager.
4. In the console tree, refresh **Alerts** to verify that the Northwind 9002 Log Full alert was created.

Trainer Materials for Microsoft Certified Trainer Use Only

► To disable auto grow, change the database recovery model, and the Northwind database

In this procedure, you will alter the **Northwind** database so that the transaction log does not auto grow, set the database recovery model to full, and then back up the **Northwind** database.

1. In the console tree, expand **Databases**, right-click **Northwind**, and then click **Properties**.
2. On the **Transaction Log** tab, clear the **Automatically grow file** check box.
3. On the **Options** tab, change the database recovery model to full, and then click **OK** to close the **Northwind Properties** dialog box.

4. Switch to SQL Query Analyzer.
5. Open C:\Moc\2072A\Labfiles\L05\ NWBackup.sql, review its contents, and then execute it.

► To raise an alert that will notify the fail-safe operator

In this procedure, you will create an operator and an on duty pager schedule specifying the times that the new operator is on duty. You will specify that this new operator be notified by pager whenever the alert that you created in the previous procedure is activated. You will then disable the performance condition alert that you created in the previous exercise so that it is not activated. Finally, you will execute a script to fill the transaction log, which raises error 9002 with severity level 17.

1. Switch to SQL Server Enterprise Manager.
2. In the console tree, expand **Management**, expand **SQL Server Agent**, and then click **Operators** to display all operators in the details pane.
3. Create the Administrator as a new operator by using the information in the following table:

Option	Value
Name	Administrator
E-mail name	Administrator@NWTraders.msft
Pager e-mail name	Administrator@NWTraders.msft
net send address	London
Pager on duty schedule	Saturday only

4. On the **Notifications** tab, select the **Pager** check box for the Northwind 9002 Full Log alert, and then click **OK**.
5. In the console tree, click **Alerts**.
6. In the details pane, right-click **Northwind Log 60% Full**, and then click **Properties**.
7. Clear the **Enabled** check box, and then close the **Northwind Log 60% Full Properties** dialog box.

   This will prevent this alert from firing and executing the associated job.
8. Switch to SQL Query Analyzer and open a new query window.

Trainer Materials for Microsoft Certified Trainer Use Only

9. Open C:\Moc\2072A\Labfiles\L05\Fulltlog.sql, review its contents, and then execute it.

   This will fill the **Northwind** database transaction log and raise error 9002 with severity level 17. The error is also written to the Windows application log.

   What occurred when the **Northwind** database transaction log became full?

   The script stopped executing when error 9002 was raised. The Northwind 9002 Full Log alert fired, which executed the Backup Northwind Log job in response to the alert. The job backed up the transaction log.

   You were notified as the fail-safe operator by e-mail and a net send command.

   _____

   _____

► **To verify that the fail-safe operator is working correctly**

In this procedure, you will verify that the fail-safe operator is contacted when an alert is fired when the operator who is designated to receive the notification is off-duty.

1. Switch to SQL Server Enterprise Manager and review the alert history for the Northwind 9002 Full Log alert. You may have to refresh the alert.

   How many times has the alert occurred? What was the date and time of the last occurrence?

   The alert should have occurred one time. The date and time will vary.

   _____

   _____

2. Review the operator notification attempts for the Northwind 9002 Full Log alert.

   One pager notification will have occurred.

3. Verify that you also received an e-mail message as the fail-safe operator.

   How can you distinguish this message from other alert system messages?

   [Fail-safe] is specified at the beginning of the subject line.

   _____

   _____

Trainer Materials for Microsoft Certified Trainer Use Only

4. Open Event Viewer and review the application log. List the events related to the error and the failed pager notification.

   Error message: "The log file for database 'Northwind' is full. Back up the transaction log for the database to free up some log space."

   Warning message: "The 1 pager operator(s) responsible for alert 'Northwind 9002 Log Full' are currently off duty so cannot be paged."

   _____

   _____

5. Close Event Viewer.
6. In the SQL Server Enterprise Manager console tree, expand **Databases**, right-click **Northwind**, and then click **Properties**.
7. On the **Transaction Log** tab, select the **Automatically grow file** check box.
8. On the **Options Log** tab, change the recovery model to simple, and then click **OK** to close the **Northwind Properties** dialog box.

Trainer Materials
for Microsoft Certified
Trainer Use Only

Automating Multiserver Jobs

Topic Objective
To introduce automating multiserver jobs.

Lead-in
You can automate jobs across multiple servers in a network. Automating multiserver jobs consists of defining a master server and one or more target servers.

- **Defining a Master Server**
  - Creates MSXOperator on master server and all target servers
  - Represents a primary department or business unit
- **Defining Target Servers**
  - Are assigned to one master server
  - Reside in the same domain as the master server

You can automate jobs across multiple servers in a network. Automating multiserver jobs consists of defining a master server and one or more target servers. A master server distributes jobs to and receives events from networked target servers.

Having a multiserver administration configuration allows you to:

- Group multiple servers into logical functioning business units.
- Manage multiple servers from one location.

For example, if a subset of your customer database is maintained at each branch office, you can create a job at the corporate headquarters to back up the customer databases in each branch office.

Trainer Materials for Microsoft Certified Trainer Use Only

Defining a Master Server

Delivery Tip
Enlisting the first target server defines the server being enlisted into it as a master server. The process of enlisting results in a row being inserted into the **systargetservers** system table on the master server. The existence of at least one row in this table designates the server as a master server.

To automate jobs on multiple servers, you must first define a master server and one or more target servers. A master server defines, schedules, and manages the jobs on all target servers. A master server can be defined on a computer running Windows 2000 or Windows NT Server *only*, because of the higher connection load that a master server requires.

You can use SQL Server Enterprise Manager or execute the **sp_msx_enlist** system stored procedure to define the master server. Consider the following facts and guidelines about master servers:

- When you define a master server you also enlist at least one target server. This inserts a row into the **systargetservers** system table on the master server. The existence of rows in this table designates the master server.

- A SQL Server logon account and password is automatically created for each target server with the suffix, _msx_probe.

  To use this account, you must configure SQL Server to use mixed mode security, and then restart the SQL Server service. This account accesses the **msdb** database to enable target servers to download jobs from the master server.

  Note If you modify the password for this account, the target server will no longer be able to access the master server. You must defect and reenlist the target server.

- The wizard creates an MSXOperator on the master server and on each target server.
- The master server usually represents a primary department or business unit server. In smaller organizations, one master server can serve the entire organization.
- You may also designate the master server as the events-forwarding server.

  If the master server is not performing database production functions, the load of managing events that are forwarded from target servers will not affect database application performance.

Defining Target Servers

You can execute the **sp_msx_enlist** system stored procedure or use the Make Target Server Wizard in SQL Server Enterprise Manager to define any additional target servers. The target server definition is stored in the **msdb..systargetservers** system table. Target servers:

- Are assigned to only one master server.
- Must reside in the same Windows domain as the master server or in a trusted Windows domain.
- Cannot be members of other master servers until they defect from their current master servers.

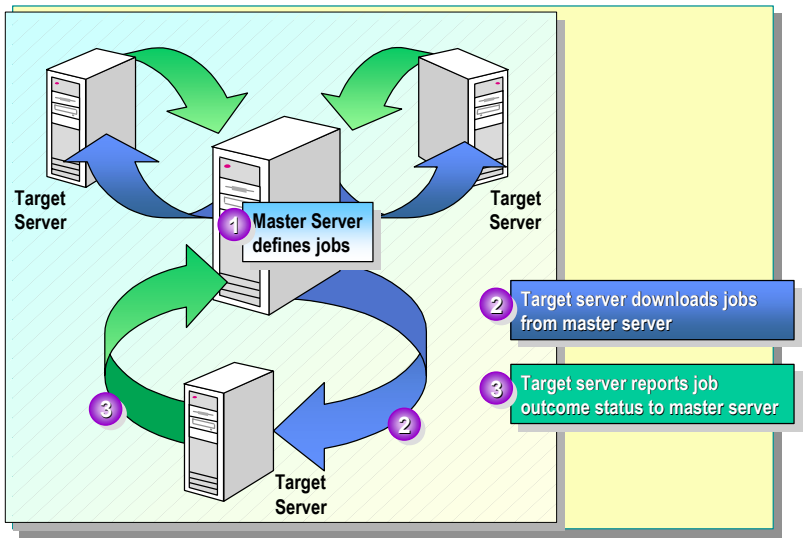
Note If you are configuring a multiserver environment with a named instance of SQL Server 2000 and a server running SQL Server version 7.0, the SQL Server 7.0 server will not work with the named instance.

Trainer Materials for Microsoft Certified Trainer Use Only

Defining Multiserver Jobs

Topic Objective
To illustrate how jobs are processed in a multiserver environment.

Lead-in
You can create jobs on the master server to occur at a target server.



After your master and target servers have been defined, you can create jobs on the master server to execute on one or more target servers. SQL Server goes through the following steps to process jobs in a multiserver environment:

1. The master server posts jobs for the target servers in a download list in the **msdb..sysdownloadlist** system table.
2. The target servers periodically connect to the master server to determine whether any new or updated jobs have been posted for download.
3. The target server uploads the job outcome status to the master server when the job completes.

Caution The paths and syntax in the job must be identical on all target servers; otherwise, the job will not work on all servers.

Trainer Materials for Microsoft Certified Trainer Use Only

Modifying Multiserver Job Definitions

The master server stores the master copy of job definitions and schedules. When you make any changes to jobs in a multiserver environment, consider the following facts and guidelines:

- Job definitions cannot be modified at the target server; they must be modified on the master server.
- SQL Server Enterprise Manager automatically posts the necessary instructions to the download list.

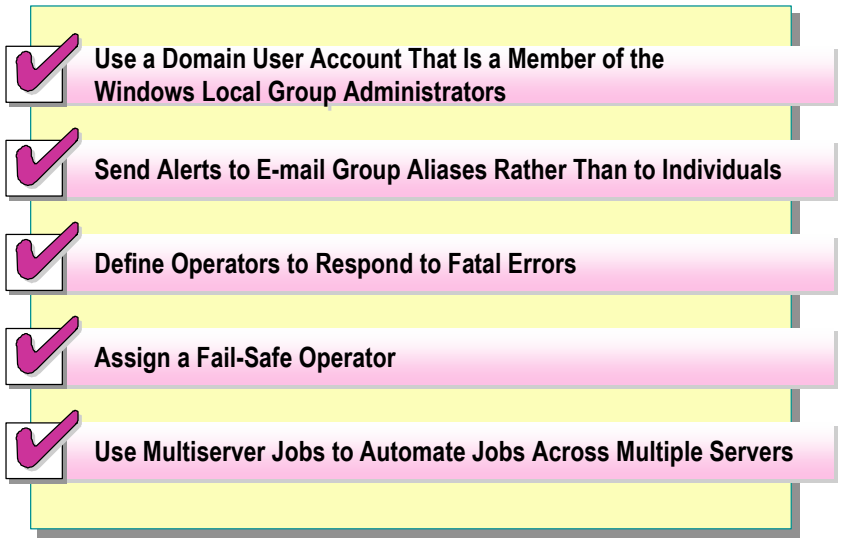
Reviewing Job History

The master server records the job outcome information from the target servers in the **msdb..sysjobservers** system table. This is in addition to the normal job history information that is recorded in the **msdb..sysjobhistory** system table of each local target server.

Recommended Practices

Topic Objective
To list the recommended practices for automating administrative tasks.

Lead-in
To get the most out of automating administrative tasks, consider the following recommended practices.



When you automate administrative tasks, you should:

- Use a domain user account that is a member of the Windows local group **Administrators**. This allows SQL Server Agent to:
  - Execute jobs when the CPU is idle.
  - Restart SQL Server automatically, if SQL Server fails unexpectedly.
  - Execute operating system command and ActiveX script job steps for jobs owned by users who are not members of the **sysadmin** role.
- Send alerts to e-mail group aliases rather than to individuals in order to ensure adequate notification for errors with severity levels greater than 18.
- Define an alert to notify an operator when fatal errors occur. SQL Server provides predefined demo alerts for error severity levels 19 through 25. You should modify and rename the predefined demo alerts.
- Create a fail-safe operator to respond to an alert when all pager notifications to other assigned operators have failed.
- Use multiserver jobs to automate jobs across multiple servers. Having a multiserver administration configuration allows you to group multiple servers into logical functioning business units and manage multiple servers from one location.

Trainer Materials for Microsoft Certified Trainer Use Only

Additional information on the following topics is available in SQL Server Books Online.

Topic	Search on
Configuring SQL Server Agent	"SQL server agent"
Configuring SQL Server to share memory resources with other server applications	"server memory options"
Creating alerts on a user-defined error	raiserror

Review

Topic Objective
To reinforce module objectives by reviewing key points.

Lead-in
The review questions cover some of the key concepts taught in the module.

- Configuration Tasks
- Routine SQL Server Administrative Tasks
- Automating Routine Maintenance Tasks
- Creating Alerts
- Troubleshooting SQL Server Automation
- Automating Multiserver Jobs

1. You want to back up the transaction log of your production database every hour during peak business hours (8:00 A.M. to 6:00 P.M.) and every three hours during non-peak hours (6:00 P.M. to 8:00 A.M.). What is the most efficient method for automating these tasks?

   Create one job to back up the transaction log, and specify two schedules.

2. The customer account manager has asked to be notified whenever a customer's credit limit is changed (increased or decreased). In addition, she wants the name of the customer representative who updated the customer's account, as well as any remarks about why the change was made. How would you accomplish this task?

   The first step is to create a custom error message that specifies the customer account name, credit limit, remarks (assuming that the column already exists in the database), and the name of the customer representative who made the update.

   The next step is to modify the stored procedure or trigger that changes customer credit limits to fire the custom error with the RAISERROR statement.

   Then, you would create the customer account manager as an operator.

   Finally, you would create an alert on the custom error message that sends an e-mail message to the customer account manager when the alert is fired.

Trainer Materials for Microsoft Certified Trainers Use Only