Installing a web API server

adlib
an AXIELL product

Axiell ALM Netherlands BV

The information in this document is subject to change without notice and should not be construed as a commitment by Axiell ALM Netherlands BV. Axiell assumes no responsibility for any errors that may appear in this document. The software described in this document is furnished under a licence and may be used or copied only in accordance with the terms of such a licence. While making every effort to ensure the accuracy of this document, products are continually being improved.

As a result of continuous improvements, later versions of the products may vary from those described here. Under no circumstances may this document be regarded as a part of any contractual obligation to supply software, or as a definitive product description.

Contents

Introduction

The Axiell (web) API aka Adlib webopac (wwwopac.exe or wwwopac.ashx), implements the Axiell/Adlib core-software functionality for web applications and is installed as a supplementary program to an existing web server (currently minimally IIS 7.0 for Windows Server 2008 SP2).
But the program has no graphical interface, so therefore a (web) application is needed to call the functions of the API. Such an application can be a website on which visitors are able to search your catalogue via the internet, and maybe place reservations, or take care of other business.
For this, the standard Adlib Internet Server (AIS) web application (which can largely be adjusted to your preferences) is available.

When installing the old Adlib wwwopac.exe CGI extension, you usually also install the Adlib Internet Server web application, although in some cases you may only want to install a wwwopac.exe server.
The later Adlib wwwopac.ashx on the other hand, is a .NET HTTP handler through which the client-side Adlib API (Application Programming Interface) has been implemented. An Adlib Internet Server web application may be part of the package you are installing, but with this HTTP handler, chances are you are building your own (web) application and you probably want to install the wwwopac.ashx server separately.
The latest version of the API uses the ASP.NET Web API framework and is therefore no longer an HTTP handler (so it no longer requires the physical wwwopac.ashx file, although it is still part of the call to the API) and is fully backwards compatible.
This installation guide provides the information you need to just install wwwopac.exe or wwwopac.ashx or the web API on a server. Please see the Adlib Internet Server installation guide for information on how to install a complete Internet Server package.

1 Requirements

You will need the following software on the server to be able to use the Adlib webopac (wwwopac) 6.5.2 or higher:

1.1 wwwopac.ashx/web API

- minimally Windows Windows Server 2008 SP2 (supported as long as Microsoft supports this Windows Server version), although we recommend installing the latest version.
- an Adlib application (with a subfolder \data), although an application is not strictly necessary, in principle just the \data folder is sufficient. A requirement is that the application data directory is actually accessible. If the data directory resides locally on a server, this accessibility shouldn't be a problem. However, should the data directory be located on a different server, then you have to check whether the access rights to the share and the ntfs rights to the relevant folder have been set up properly. For access to the relevant share, by default the account is used under which the application pool is running.
- the package of files that make up the API. If you are using a 32-bit operating system then you must use the (standard) 32-bit version of these programs, while for a 64-bit operating system we recommend to use the 64-bit version. If you decide to use the 32-bit version on a 64-bit OS anyway, you must enable 32-bit applications in your application pool for the relevant .ashx explicitly if your API still contains the physical wwwopac.ashx file.
- http server software must be installed on the servers on which the API will be placed, such as IIS 7.0 for Windows Server 2008. This makes it possible for workstations (client side) to access pages from the server. For the required Windows versions, these services are available: already installed, or included in the Windows installation package.
  Other web servers with scripting support (e.g. Apache and PHP) can also be used.
- The Microsoft .NET Framework (minimally version 4.6.2, but latest version is recommended) must be installed on the server. See http://msdn.microsoft.com/en-us/netframework/aa569263.aspx. (If the .NET Framework still has to be installed, then please take into account that the web server might need rebooting after this installa-

tion.)
On IIS 7, ASP.NET must operate in integrated mode (which is the default configuration). The application pool which we will create for the API server later on in this manual, must run in this mode. It is also a requirement that physical or virtual folders above the API folder do not run in earlier versions of the .NET Framework, so a .NET 2.0 application pool must not contain applications or folders using .NET 4.6.2.

- If Active Directory authentication will be used for access to the database, instead of SQL authentication, then the application pool must be configured to use an account which has access to the SQL Server.
- Use servers with at least a dual-core Intel Pentium processor, and 2 GB RAM or more.

1.2 wwwopac.exe

- minimally Windows Server 2008 SP2.
- an Adlib application (with a subfolder *data*), although an application is not strictly necessary, just the *data* folder is sufficient. A requirement is that the application data directory is actually accessible. If the data directory resides locally on a server, this accessibility is in principle not a problem. However, should the data directory be located on a different server, then you have to check whether the access rights to the share and the ntfs rights to the relevant folder have been set up properly. For access to the relevant share, by default the account is used under which the application pool is running.
- the wwwopac executable, plus accompanying files delivered with the package. Put all dlls in the same folder as wwwopac.exe. Optionally you can place the above mentioned files in the Windows *system32* folder too, instead of in the web application folder. Always, the *system32* folder is searched for these files first, and only when they are not present there the virtual directory from the search request is searched. But in *system32* the Adlib files cannot be kept up-to-date very comfortably.
- an adlib.lic file, containing your product license.
- http server software must be installed on the servers on which wwwopac and the web application will be placed, such as IIS 7.0 for Windows Server 2008 SP2. This makes it possible for workstations (client side) to access pages from the server. For the re-

quired Windows versions, these services are available: already installed, or in the Windows installation package.

- The Microsoft .NET Framework version 4.6.2 is not required on the server for a wwwopac.exe web service, but it is still recommended to create a (.NET) application pool for the web service, for safety reasons.
  For more information about .NET 4.6.2 see:
  http://msdn.microsoft.com/en-us/netframework/aa569263.aspx.
  (If the .NET Framework still has to be installed, then please take into account that the web server might need rebooting after this installation.)
  On IIS 7, ASP.NET must operate in integrated mode (which is the default configuration). The application pool which we will create for the wwwopac.exe server later on in this manual, must run in this mode.
  It is a requirement that physical or virtual folders above the wwwopac.exe folder do not run in earlier versions of the .NET Framework, so .NET 2.0 application pool must not contain applications or folders using .NET 4.6.2.
  If Active Directory authentication will be used for access to the database, instead of SQL authentication, then the application pool must be configured to use an account which has access to the SQL Server.

- Use servers with at least a dual-core Intel Pentium processor, and 2 GB RAM or more.

2 The installation procedure

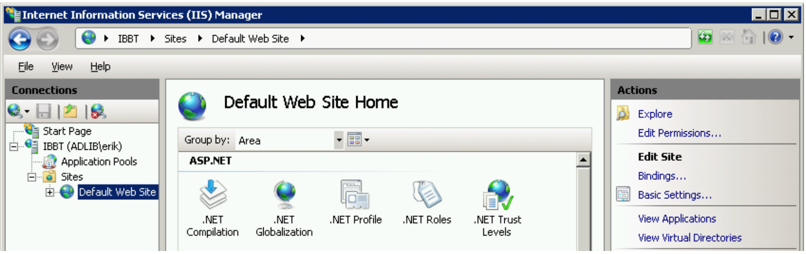
2.1 web API/wwwopac.ashx: IIS 7 setup under Windows Server 2008 SP2

Besides a new subfolder for all Adlib web service files, you need to make at least one so-called *application* in IIS, to secure your server and to create an internet address.

> If the new web service (application) is the only one running on the server, then of course the default .NET 4 application pool can be chosen. However, if multiple web services are present on the server, it's best to configure one application pool per web service, to keep the processes of the web services separated at all times. A new application pool can be created underneath the *Application pools* in IIS.

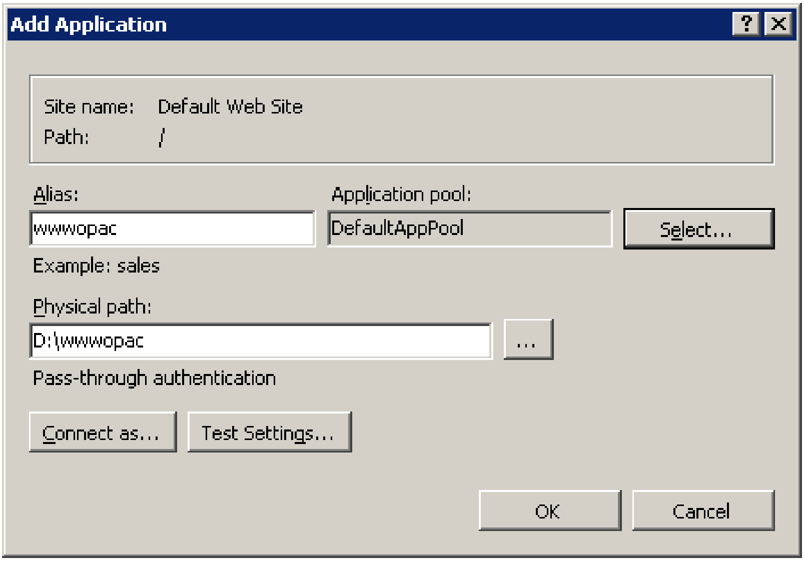
In Windows Server 2008, you create a new application as follows:

1. Start the *Internet Information Services (IIS) Manager* by clicking *Start > All programs > Administrative tools > Internet Information Services (IIS) manager*. In it, open the *Sites* node. Then right-click the site in which you want to accomodate your web service, the *Default Web Site* for instance.



2. In the pop-up menu that opens, choose *Add Application*. The *Add Application* window opens.

3. First, enter the desired *Alias* for the application, for example `AdlibAPI` or `Adlibwwwopac`; choose a clear, descriptive name. Then select the path to the physical folder on your system in which the API files can be found.



Add Application

Site name: Default Web Site
Path: /

Alias: wwwopac

Application pool: DefaultAppPool

Example: sales

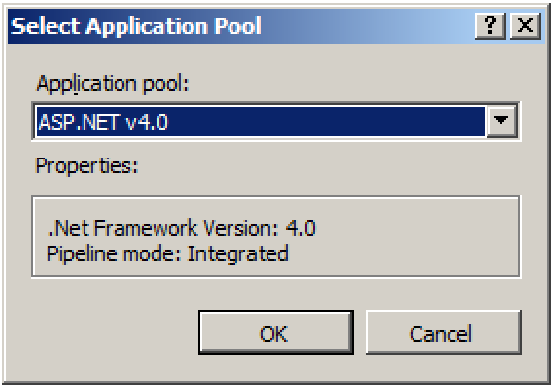
Physical path: D:\wwwopac

Pass-through authentication

The format of the URL for calling the Adlib API web service becomes (even for the latest version of the web API, which doesn't have a physical wwwopac.ashx file):

```
http://<webserver>/<application_alias>/wwwopac.ashx
```

4. Click the *Select* button to set the base application pool. Select *ASP.NET v4.0* or your own .NET 4.0 application pool. Click *OK*.



Select Application Pool

Application pool: ASP.NET v4.0

Properties:

.Net Framework Version: 4.0
Pipeline mode: Integrated

5. Click *OK* in the *Add application* window to create the application.

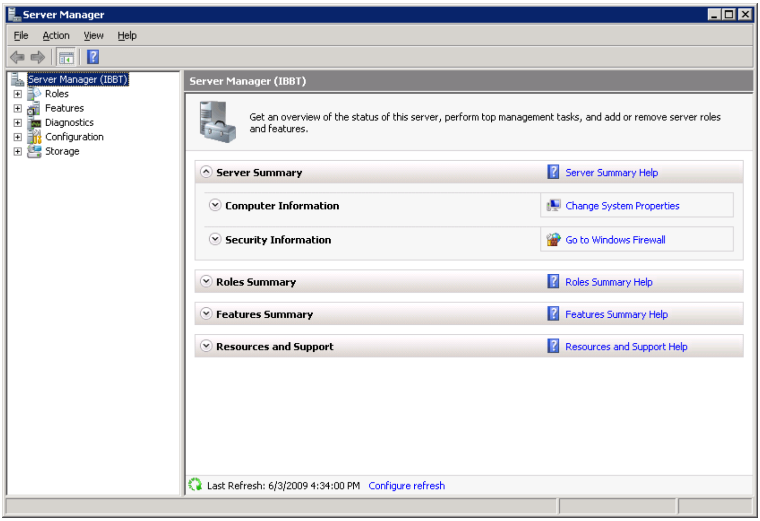
2.2 wwwopac.exe: IIS 7 setup under Windows Server 2008

Besides a new subfolder for all Adlib web service files, it is recommended to make at least one so-called *application* in IIS, to secure your server and to create an Internet address.

> Contrary to wwwopac.ashx, the wwwopac.exe is not dependent on a .NET framework, so it is not mandatory to create a separate application pool for wwwopac.exe, and neither is it mandatory to define the wwwopac folder as a web application. However, we do recommend to create a virtual directory for the folder which contains wwwopac.exe, and then to convert this directory to a so-called *application*.

In Windows Server 2008, the web service setup is as follows:

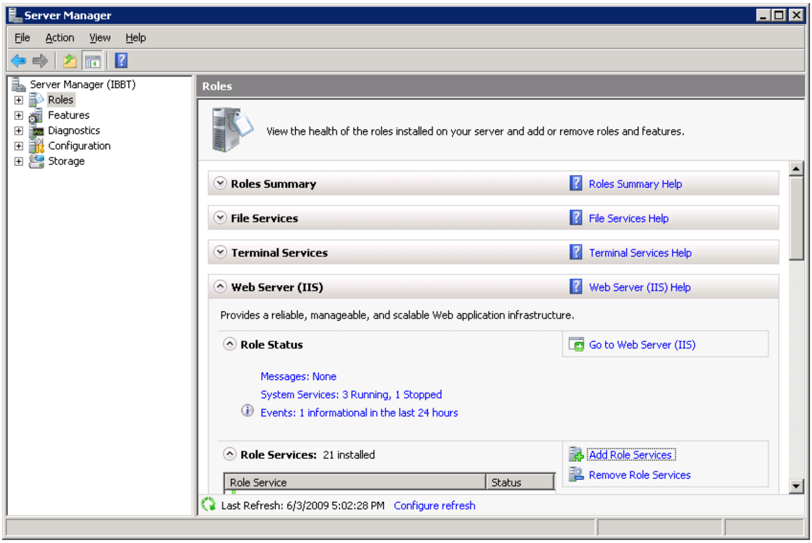
1. Open the Server Manager by clicking *Start* > *All programs* > *Administrative tools* > *Server manager*.



Get an overview of the status of this server, perform top management tasks, and add or remove server roles and features.

Last Refresh: 6/3/2009 4:34:00 PM

2. In the left window pane, click the *Roles* node and then click the *Add role Services* option right next to the *Role services* header in the *Web Server (IIS)* section in the right window pane (fold in sections or scroll down if you do not see the section immediately). The *Add Role Services* window opens. In it, select the *Role services* page and mark the *CGI* checkbox. Click *Next* and then click *Install*.



View the health of the roles installed on your server and add or remove roles and features.

Provides a reliable, manageable, and scalable Web application infrastructure.

Messages: None

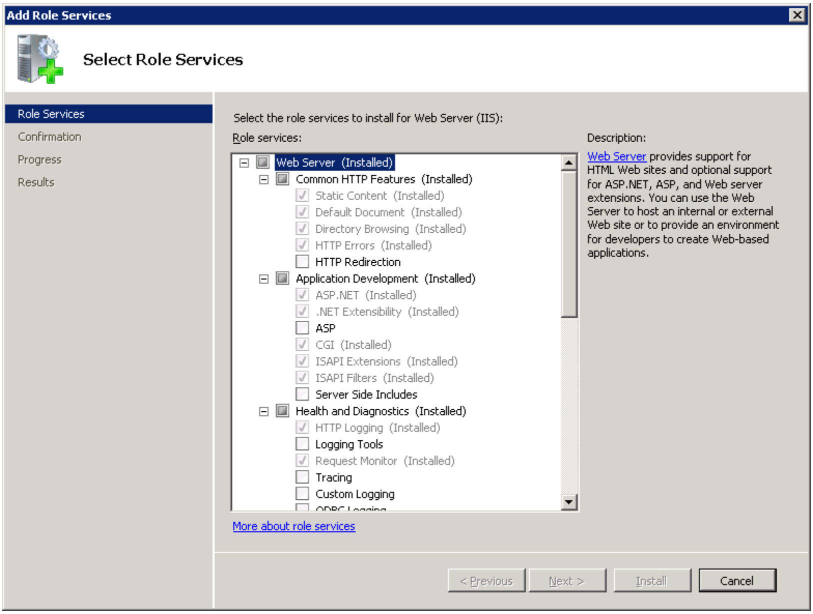
System Services: 3 Running, 1 Stopped

Events: 1 informational in the last 24 hours

Role Services: 21 installed

Last Refresh: 6/3/2009 5:02:28 PM

If CGI has been installed already, the *CGI* checkbox will be displayed checked and greyed out. Close the *Server Manager*.



Select the role services to install for Web Server (IIS):

Role services:

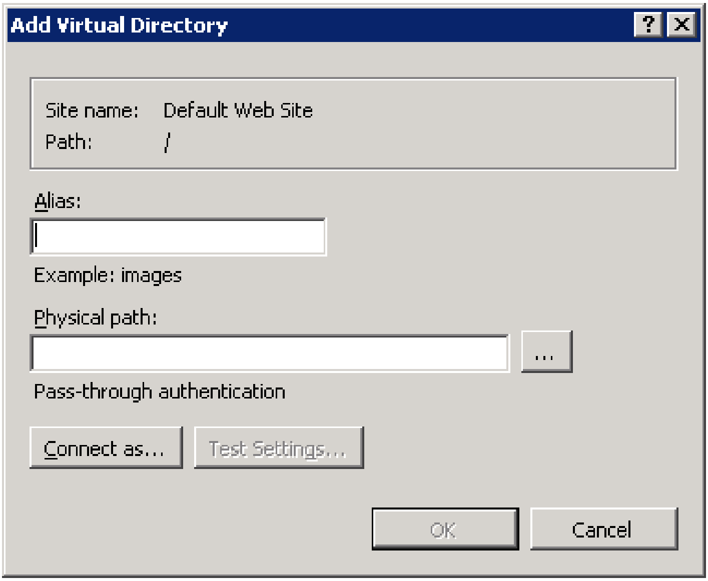
- ■ Web Server (Installed)
  - ■ Common HTTP Features (Installed)
    - ☑ Static Content (Installed)
    - ☑ Default Document (Installed)
    - ☑ Directory Browsing (Installed)
    - ☑ HTTP Errors (Installed)
    - □ HTTP Redirection
  - ■ Application Development (Installed)
    - ☑ ASP.NET (Installed)
    - ☑ .NET Extensibility (Installed)
    - □ ASP
    - ☑ CGI (Installed)
    - ☑ ISAPI Extensions (Installed)
    - ☑ ISAPI Filters (Installed)
    - □ Server Side Includes
  - ■ Health and Diagnostics (Installed)
    - ☑ HTTP Logging (Installed)
    - □ Logging Tools
    - ☑ Request Monitor (Installed)
    - □ Tracing
    - □ Custom Logging
    - □ ODBC Logging

Description:

Web Server provides support for HTML Web sites and optional support for ASP.NET, ASP, and Web server extensions. You can use the Web Server to host an internal or external Web site or to provide an environment for developers to create Web-based applications.

3. Open IIS by clicking *Start > All programs > Administrative tools > Internet Information Services (IIS) manager*.

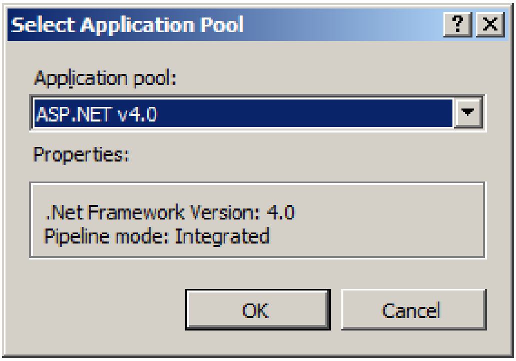
4. Open the *Sites* node and right-click the site in which you want to accomodate your web service, the *Default Web Site* for instance. In the shortcut menu that opens, select *Add virtual Directory*. The *Add virtual directory* window opens.



5. Type a clear and descriptive name for the virtual folder in the *Alias* field, `AdlibWebApp` or `Adlibwwwopac` for example. Then select the path to the physical folder on your system in which wwwopac.exe and its accompanying subfolders and files can be found.
   The format of the URL for calling the Adlib web service becomes:
   `http://<webserver>/<application_alias>/wwwopac.exe`

6. Click the *Connect as* button and select *Application user* in the dialog which opens. Click *OK* in both windows to create the virtual directory.

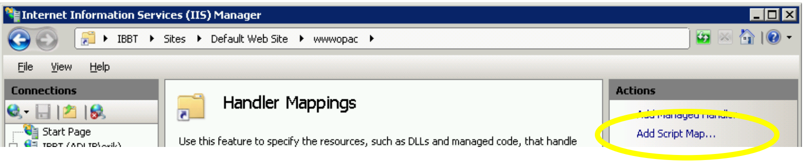
7. Underneath the *Default Web Site*, your new virtual folder now appears. Right-click it and choose *Convert to application* in the pop-up menu. The *Add application* window opens, with virtual directory details already filled in. Click the *Select* button to set the base application pool. Select *ASP.NET v4.0*. Click *OK* in both windows to create the application pool.



Application pool: ASP.NET v4.0

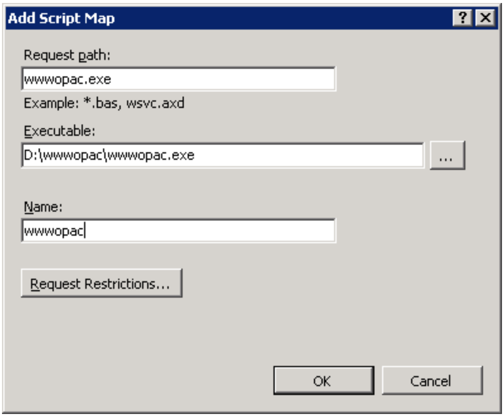
.Net Framework Version: 4.0
Pipeline mode: Integrated

8. To make sure that the wwwopac.exe will be used as CGI handler and won't be downloaded as a file, you'll have to create a handler mapping in IIS.
   Select your virtual directory and double-click the *Handler Mappings* icon in the middle window pane of the *Internet Information Services (IIS) Manager*. When the *Handler Mappings* page is visible, click the *Add Script Map* option underneath *Actions* in the right window pane.



Use this feature to specify the resources, such as DLLs and managed code, that handle

9. The *Add Script Map* window opens. In the first entry field, enter `wwwopac.exe`, and in the second field enter the full path to it, for example like in the figure below. In the last entry field, enter a name of your choice for this executable reference; this name identifies the handler mapping in the mapping list in the IIS Manager main window.
   Make sure that the name of the executable: *wwwopac.exe*, in the first two entry fields both have been entered in lower case, like below, or both in capitals (otherwise you'll get an error message).



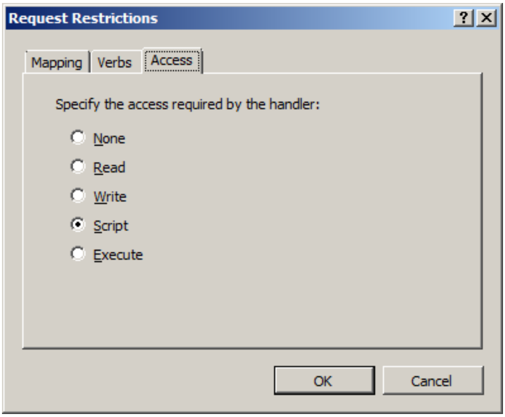
Request path: wwwopac.exe

Example: *.bas, wsvc.axd

Executable: D:\wwwopac\wwwopac.exe

Name: wwwopac

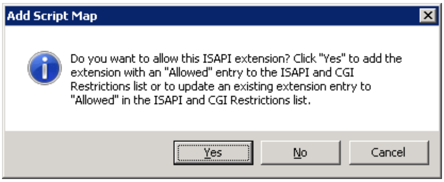
Click the *Request restrictions* button to check if the proper restrictions have been set: this is usually the case, by default. No settings have to be marked on the first two tabs, and on the third at least *Script* access must have been selected. Click *OK* in both windows.



Specify the access required by the handler:

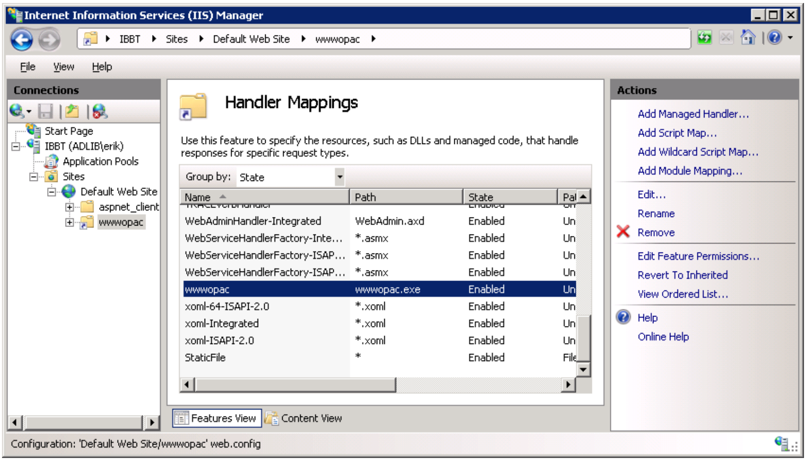
○ None
○ Read
○ Write
● Script
○ Execute

In the *Add Script Map* message which then appears, click *Yes*.



Do you want to allow this ISAPI extension? Click "Yes" to add the extension with an "Allowed" entry to the ISAPI and CGI Restrictions list or to update an existing extension entry to "Allowed" in the ISAPI and CGI Restrictions list.

10. The Script Map for *wwwopac.exe* has now been created and is visible in the *Handler Mappings* list.



Use this feature to specify the resources, such as DLLs and managed code, that handle responses for specific request types.

Group by: State

Name	Path	State	Pat
WebAdminHandler-Integrated	WebAdmin.axd	Enabled	Un
WebServiceHandlerFactory-Inte...	*.asmx	Enabled	Un
WebServiceHandlerFactory-ISAP...	*.asmx	Enabled	Un
WebServiceHandlerFactory-ISAP...	*.asmx	Enabled	Un
wwwopac	wwwopac.exe	Enabled	Un
xoml-64-ISAPI-2.0	*.xoml	Enabled	Un
xoml-Integrated	*.xoml	Enabled	Un
xoml-ISAPI-2.0	*.xoml	Enabled	Un
StaticFile	*	Enabled	File

Configuration: 'Default Web Site/wwwopac' web.config

2.3 Configuring the web service with adlibweb.xml

The *adlibweb.xml* file serves to initialize the API. In here you must at least enter a `<databasepath>` and a `<database>`:

- See chapter 1 in the WWWOPAC reference guide for more information about configuring wwwopac.exe through adlibweb.xml.
- See http://api.adlibsoft.com/site/documentation for more information about configuring the API through adlibweb.xml.

3 User authentication

Adlib applications (Windows applications as well as Axiell Collections and the other web applications) which run on an SQL database can be secured in different ways: some users should only be allowed to retrieve and view data, while others may enter and/or remove data or even get to manage the database itself. Therefore, users must be authenticated before they are allowed to work with Adlib. Authentication of users for access to your SQL database can essentially be set up in two ways:

- **SQL authentication in combination with Adlib access rights** – In this case, the Adlib core software always connects to the server via one and the same general user name and password which are set in the Adlib database structure files (*.inf*). That one "user" must get sufficient permissions in the SQL database, so that in principle the database can be managed in its entirety. The limiting access rights for the actual individual users, must be set in the Adlib application structure files (*.pbk*); see the *Adlib Designer Help* for more information about this. In this setup, those *.pbk* files do need to be located in a secured, e.g. virtual, Adlib folder, to prevent them from being modified by unauthorized persons; see the *Installing Museum, Library and Archive* guide for information about setting up a virtual folder for Adlib structure files.
  The advantage of this authentication method is that the access rights management mainly takes place in Adlib, and can be done by an Adlib application manager. This authentication method is also the easiest method for solving any individual problems with establishing a connection to the SQL database in a multi-server environment; this is because the other authentication method (see below) uses Active Directory, which may sometimes complicate user authentication in a multi-server environment.
  A disadvantage may be that user names and passwords are located in an Adlib *.pbk* file which needs to be secured well. Also, all Adlib users must actually be registered and managed in the *.pbk* file.

- **Windows authentication by means of Active Directory, possibly in combination with Adlib access rights** – With this method, you use the Windows login data (user names and passwords) which has already been registered in Active Directory for your local network. For the benefit of Adlib, those users must, as much as possible, be divided into groups which should be assigned different access rights in SQL Server. So, access of the individual

user to the SQL database depends on the name and the password with which the user is logged onto the local network. Any further refinement of the access rights can be taken care of in Adlib.
An advantage of this method is that user names and passwords are well secured in Active Directory, and that all users of the network are already registered; only for the benefit of Adlib you'll still have to divide the users into groups which can then be assigned certain access rights per group.
A possible disadvantage in a multi-server environment is that each server has its own Active Directory (the servers could have separate domains), and because of this it may sometimes be difficult to streamline user authentication.

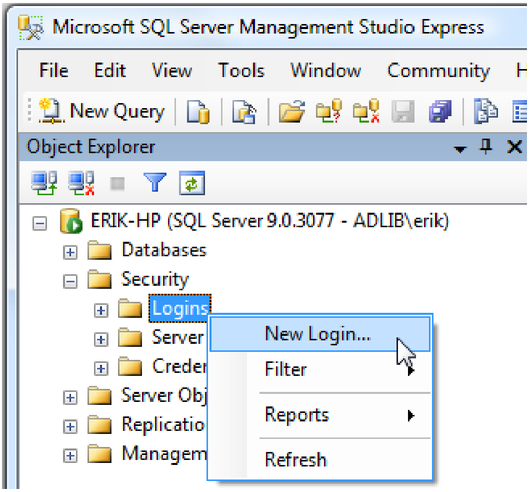
Anonymous internet users who retrieve data from the Adlib SQL database via your web application, enter your network under one and the same IIS account name, by default this is `IUSR_<server>` in which <server name> has been replaced by the actual name of the server on which the web application runs; under Windows 2008, the default account name is just `IUSR`.
If you use SQL authentication, in principle all users have full access rights. You'll have to protect your database from unauthorized writing and deleting by internet users first by limiting access to data sources via the *adlibweb.xml* configuration file. A further refinement of the access rights for anonymous internet users is possible, for instance by specifying access rights on record level in Adlib: the IIS account name for the anonymous users can then be entered as user name in a field to be added to your application for this purpose, which has to be set as *Authorisation user field* with *Exclude* as *Authorisation type* in the database. Every record in which subsequently said account name is stored, is excluded from results of any API search from then on. See the *Use the authorisation functionality* paragraph in the *User authentication and access rights* topic under the *General topics* chapter in the Designer Help for an explanation about setting up this type of access restriction.
If you use Windows authentication, the account name for the anonymous internet user must be set as login name in SQL Server, with just the read-only access rights for example. It is possible that the default account name is not available in SQL Server, in which case you'll have to create a different account name for the anonymous users in IIS yourself. That name should then appear amongst the Active Directory account names of users when you create a login in SQL.
In both cases, the anonymous internet users must have read-only rights to the folder in which the Adlib *.inf* files are located. For this purpose, these files can be copied to a suitable location, if desired.

Which authentication method is to be preferred, depends on the way in which your Adlib system has been installed, on the setup of your local network, and on your own preferences and security policy.

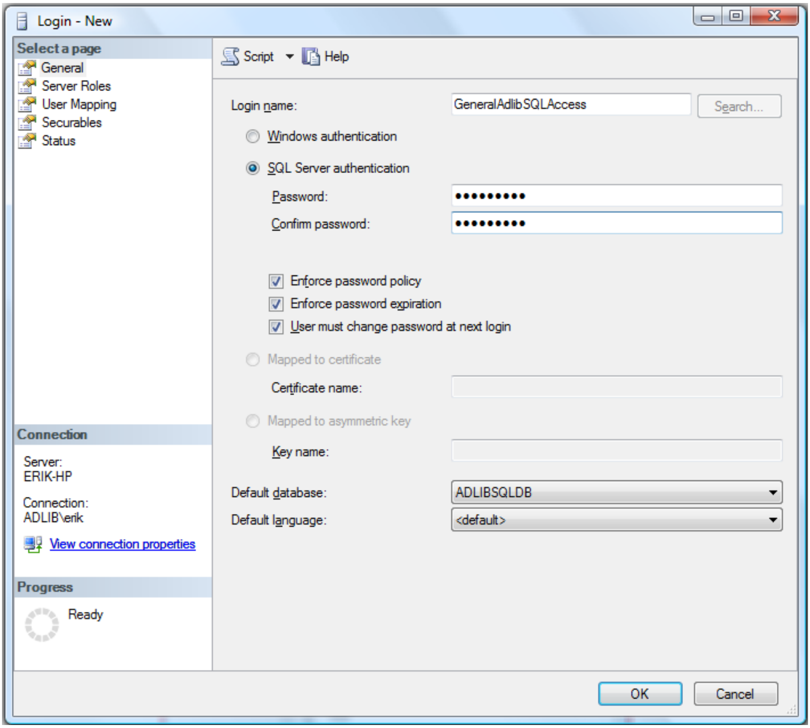
Below, you'll find a step-by-step procedure for setting up either authentication method properly.

3.1 Setting up SQL authentication

1. Start Microsoft SQL Server Management Studio (Express), open the *Security* folder underneath the SQL Server folder and right-click *Logins*. In the pop-up menu which opens, click the *New Login…* option.



2. In the *Login – new* window, choose a sensible login name, mark the *SQL Server authentication* option, provide a password, confirm the password, and choose your Adlib SQL database as the *Default database*. (Do choose a hard to guess and sufficiently long password, otherwise the program produces an error message when you close this window.) Then unmark the *User must change password at next login* checkbox (remove the check).



Login name: GeneralAdlibSQLAccess

○ Windows authentication

● SQL Server authentication

Password: ●●●●●●●●●

Confirm password: ●●●●●●●●●

☑ Enforce password policy

☑ Enforce password expiration

☑ User must change password at next login

○ Mapped to certificate

Certificate name:

○ Mapped to asymmetric key

Key name:

Default database: ADLIBSQLDB

Default language: <default>

Connection

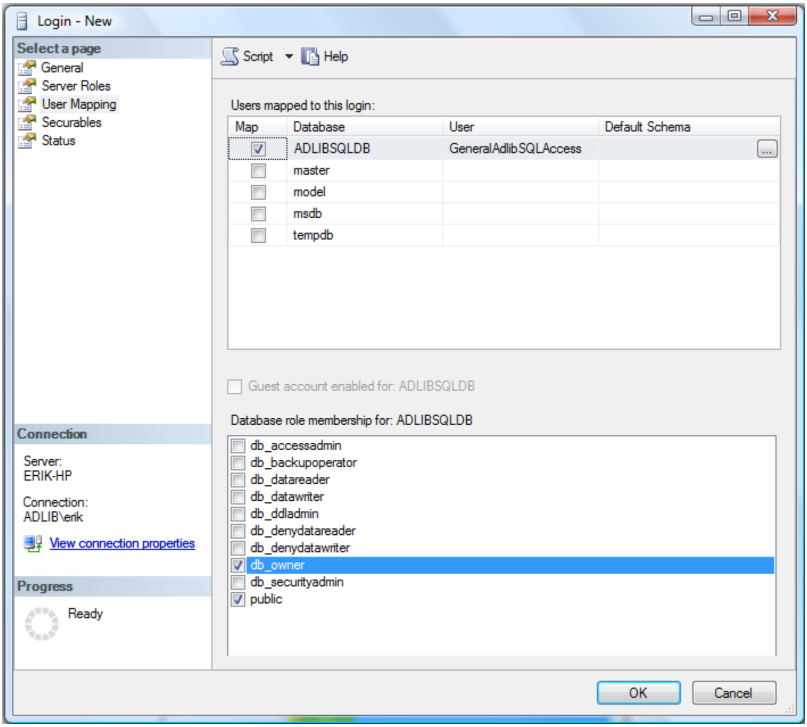
Server:
ERIK-HP

Connection:
ADLIB\erik

Progress

Ready

3. Select the *User mapping* page in the current window. In the list on the right, again mark your Adlib SQL database, and in the list below it, mark the *db_owner* role. (Leave the *public* role marked.)



Server:
ERIK-HP

Connection:
ADLIB\erik

View connection properties

Ready

Users mapped to this login:

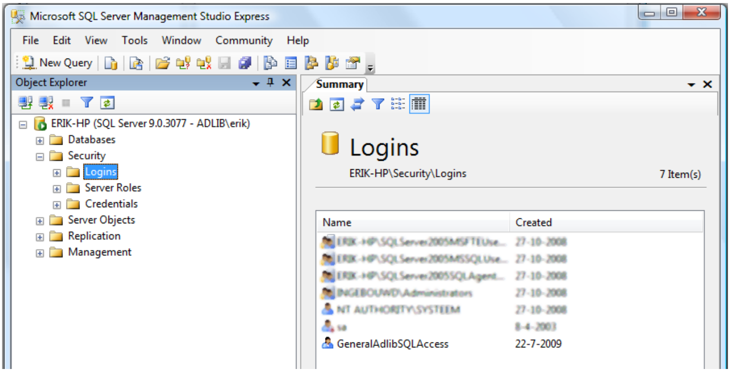
Map	Database	User	Default Schema
☑	ADLIBSQLDB	GeneralAdlibSQLAccess	
□	master		
□	model		
□	msdb		
□	tempdb		

□ Guest account enabled for: ADLIBSQLDB

Database role membership for: ADLIBSQLDB

□ db_accessadmin
□ db_backupoperator
□ db_datareader
□ db_datawriter
□ db_ddladmin
□ db_denydatareader
□ db_denydatawriter
☑ db_owner
□ db_securityadmin
☑ public

Then click *OK*. The new login is now present in the list.



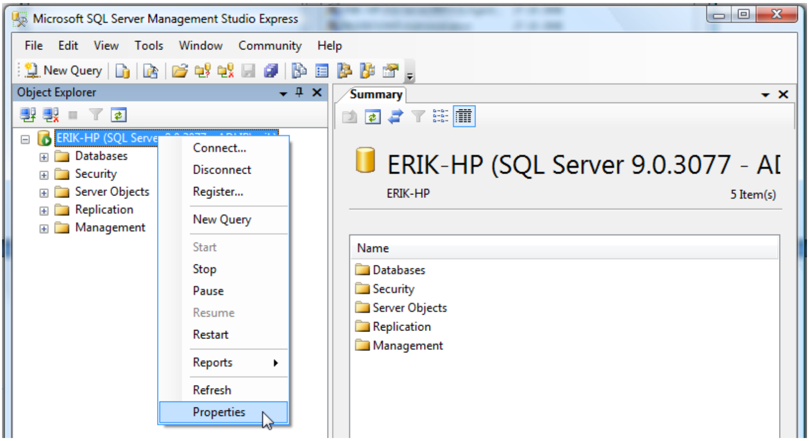
ERIK-HP (SQL Server 9.0.3077 - ADLIB\erik)
Databases
Security
Logins
Server Roles
Credentials
Server Objects
Replication
Management

Logins

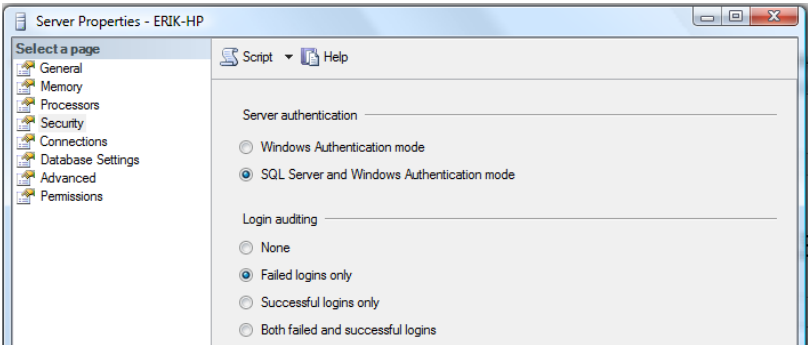
ERIK-HP\Security\Logins 7 Item(s)

Name	Created
ERIK-HP\SQLServer2005MSFTEUse...	27-10-2008
ERIK-HP\SQLServer2005MSSQLUse...	27-10-2008
ERIK-HP\SQLServer2005SQLAgent...	27-10-2008
INGEBOUWD\Administrators	27-10-2008
NT AUTHORITY\SYSTEEM	27-10-2008
sa	8-4-2003
GeneralAdlibSQLAccess	22-7-2009

4. Now check whether the login settings for the SQL Server are correct. In Microsoft SQL Server Management Studio Express, right-click the SQL Server name, and choose *Properties* in the pop-up menu which opens.



5. Select the *Security* page and mark the *SQL Server and Windows Authentication mode* option, if that hasn't been done yet. (This is sometimes also called *mixed mode*.)



Server authentication

○ Windows Authentication mode

● SQL Server and Windows Authentication mode

Login auditing

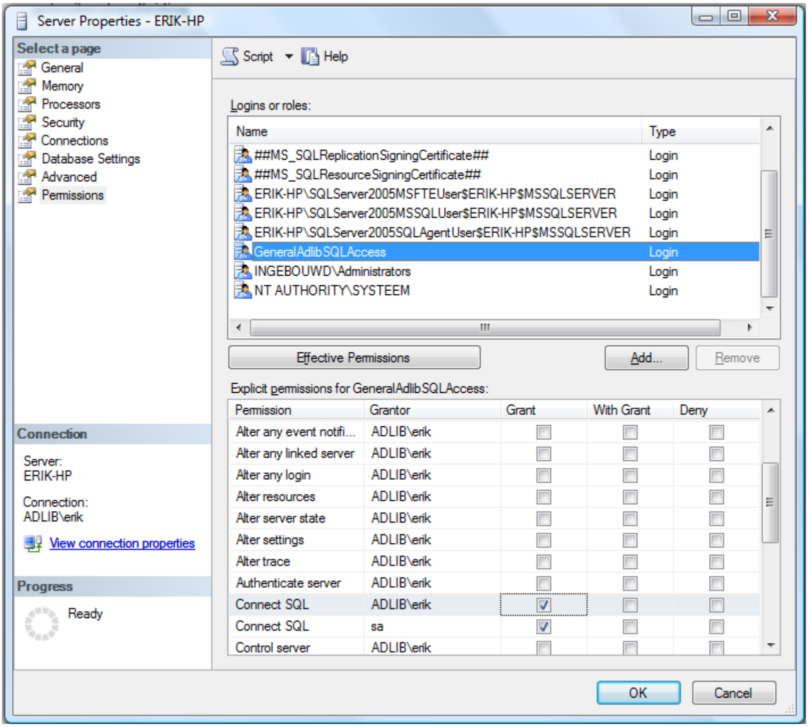
○ None

● Failed logins only

○ Successful logins only

○ Both failed and successful logins

6. Open the *Permissions* page to be able to check the access to the SQL Server for the new login. Select your login in the *Logins* list, and in the list below it mark at least the *Grant* permission for *Connect SQL*, but you may assign more rights if you wish.



Server:
ERIK-HP

Connection:
ADLIB\erik

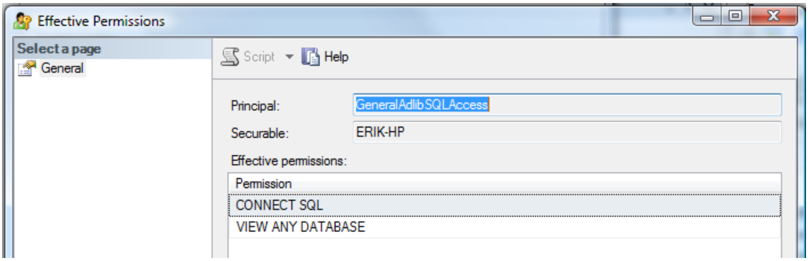
Logins or roles:

Name	Type
##MS_SQLReplicationSigningCertificate##	Login
##MS_SQLResourceSigningCertificate##	Login
ERIK-HP\SQLServer2005MSFTEUser$ERIK-HP$MSSQLSERVER	Login
ERIK-HP\SQLServer2005MSSQLUser$ERIK-HP$MSSQLSERVER	Login
ERIK-HP\SQLServer2005SQLAgentUser$ERIK-HP$MSSQLSERVER	Login
GeneralAdlibSQLAccess	Login
INGEBOUWD\Administrators	Login
NT AUTHORITY\SYSTEEM	Login

Explicit permissions for GeneralAdlibSQLAccess:

Permission	Grantor	Grant	With Grant	Deny
Alter any event notifi...	ADLIB\erik	□	□	□
Alter any linked server	ADLIB\erik	□	□	□
Alter any login	ADLIB\erik	□	□	□
Alter resources	ADLIB\erik	□	□	□
Alter server state	ADLIB\erik	□	□	□
Alter settings	ADLIB\erik	□	□	□
Alter trace	ADLIB\erik	□	□	□
Authenticate server	ADLIB\erik	□	□	□
Connect SQL	ADLIB\erik	☑	□	□
Connect SQL	sa	☑	□	□
Control server	ADLIB\erik	□	□	□

If you click the *Effective permissions* button, you can see which rights users with this login actually have.



Principal: GeneralAdlibSQLAccess

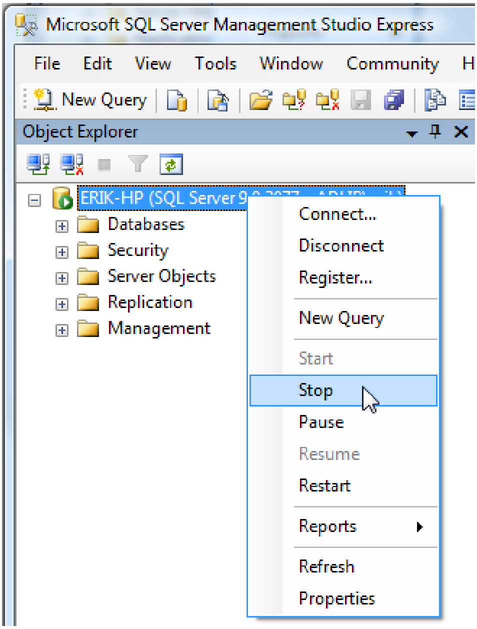
Securable: ERIK-HP

Effective permissions:

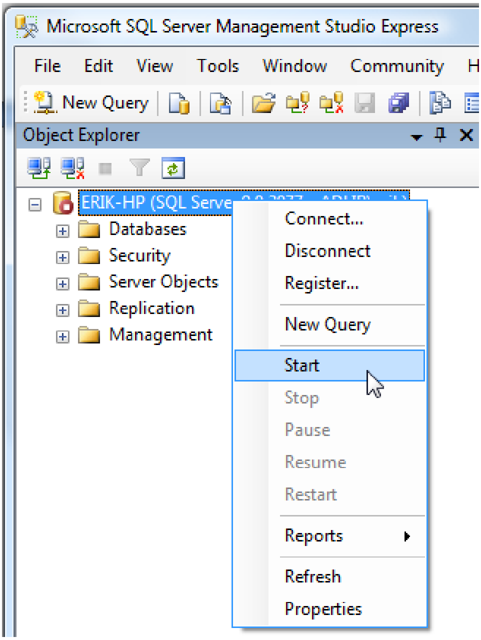
Permission
CONNECT SQL
VIEW ANY DATABASE

Click *OK* to close this window, and click *OK* again in the *Server Properties* window to store the changes.

7. Because you have changed settings for the SQL Server (from *Windows Authentication mode* to *SQL Server and Windows Authentication mode*), the server needs to be restarted. First make sure nobody is currently working in the database. Then stop the server by right-clicking the SQL Server and choosing *Stop* in the pop-up menu.

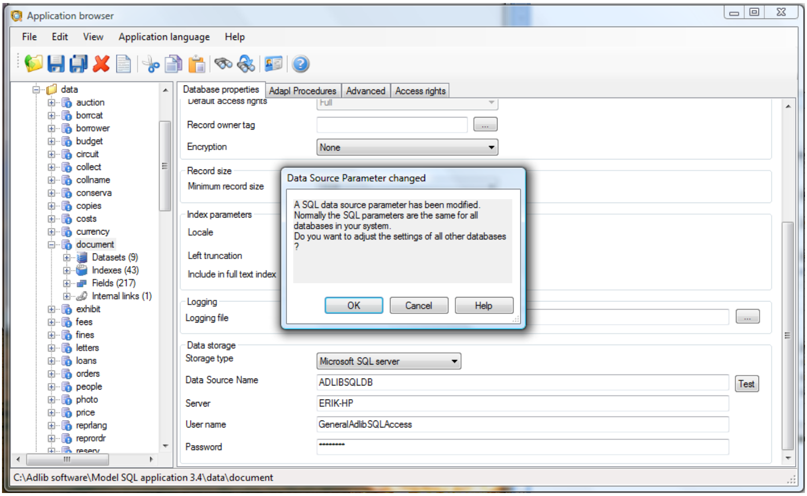


When the server has stopped, this is indicated by a red icon in front of the server name. Right-click the server name again and now choose *Start* in the pop-up menu.



If the icon turns green, it means the server is up and running again. Microsoft SQL Server Management Studio Express can now be closed.

8. Start Adlib Designer, and in the Application browser open the folder in which the *.inf* files of your Adlib SQL database are located. Select a random *.inf* file, for instance that of *DOCUMENT*. For the *User name*, enter the login name which you defined in SQL Server, in our example: `GeneralAdlibSQLAccess`. For the *Password*, enter the password which you provided for this login in SQL Server. Every time you leave one of these two field, Designer will ask you if you want apply this change everywhere; click *OK* in both occassions. This means you don't have to manually repeat your settings for the other *.inf* files. Now, save all changed files.
Note that we assume here that you've already set the *Storage type*, *Data Source Name* and *Server* options on this tab correctly. If not, then do that now (for all Adlib databases).



Default access rights: Full

Record owner tag

Encryption: None

Minimum record size

Locale

Left truncation

Include in full text index

Logging file

Data Source Parameter changed

A SQL data source parameter has been modified. Normally the SQL parameters are the same for all databases in your system.
Do you want to adjust the settings of all other databases ?

Storage type: Microsoft SQL server

Data Source Name: ADLIBSQLDB

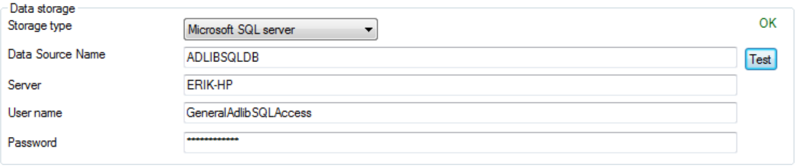
Server: ERIK-HP

User name: GeneralAdlibSQLAccess

Password: ********

C:\Adlib software\Model SQL application 3.4\data\document

Then click the *Test* button behind the *Data Source Name* entry field to test the connection with the SQL Server. If the connection is successful, the text *OK* appears above the button.



Storage type: Microsoft SQL server

Data Source Name: ADLIBSQLDB

Server: ERIK-HP

User name: GeneralAdlibSQLAccess

Password: *************

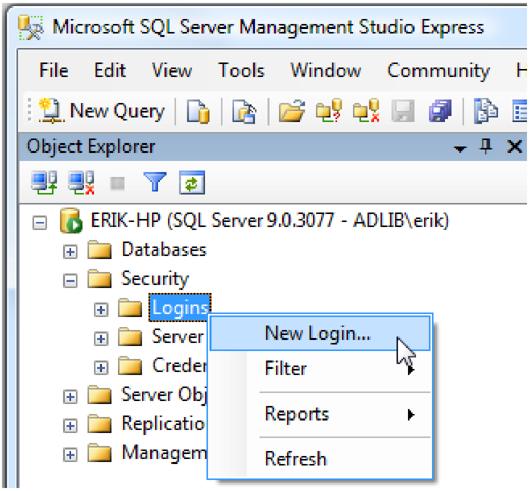
If anything goes wrong, you'll be so notified and the red text *ERR* appears above the *Test* button. Then check your settings on this tab, and the settings in the SQL Server.

All Adlib users, and any other users who know the login name and password, now have full (dbo: database owner) access rights to the SQL database. That is probably undesirable. Therefore, use the *adlibweb.xml* file to configure global access rights, and use the different internal Adlib mechanisms to set access rights for individual users. See the Designer help for more information about the latter.

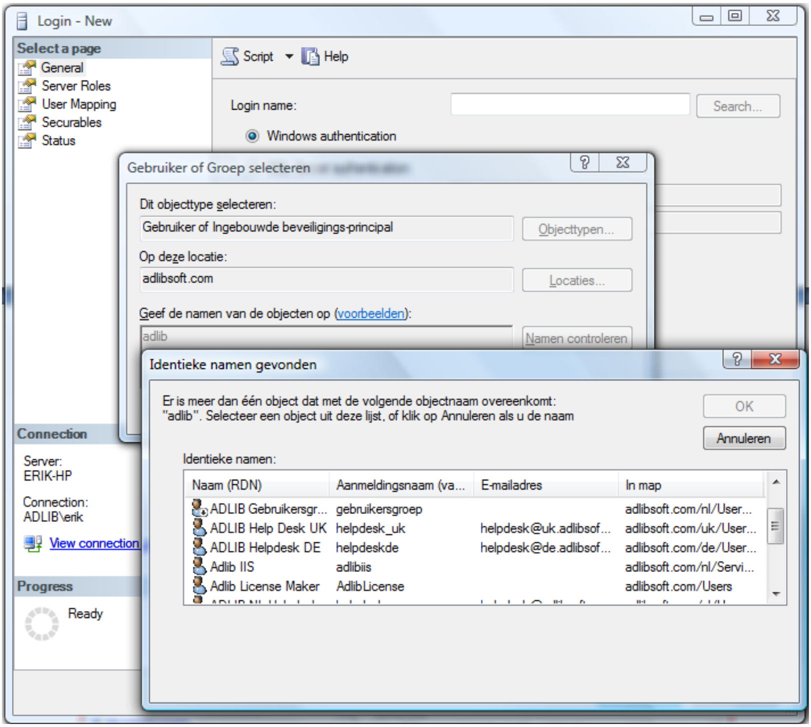
3.2 Windows authentication with Active Directory

1. Divide all Adlib users in Active Directory into groups, so that in SQL Server only groups need to be entered and assigned access rights, instead of having to do that for each individual user. For example, you can put together groups for users who are only allowed to view data (e.g. trainees and visitors), for users who are allowed to view, edit, enter and delete (e.g. registrars and librarians), and for users allowed to manage the database structure.

2. Start Microsoft SQL Server Management Studio Express if that hasn't been done yet, open the *Security* folder underneath the SQL Server folder and right-click *Logins*. In the pop-up menu which opens, click the *New Login…* option.



3. In the *Login – new* window, click the *Search* button to be able to select an Active Directory user group. First, the *Select a user or group* window opens. In it, click the *Locations* button and select the network the Adlib users are part of, *adlibsoft.com* in our example. In the *Enter the names of the objects* field, enter the partial or whole name of a user group which you would like to set as login, and click the *Check names* button. The *Identical names found* window opens if the entered name is not yet correct. In this window, select the desired user group and click *OK*. Also click *OK* in the *Select a user or group* window.



Login name:

● Windows authentication

Server:
ERIK-HP

Connection:
ADLIB\erik

Dit objecttype selecteren: Gebruiker of Ingebouwde beveiligings-principal

Op deze locatie: adlibsoft.com

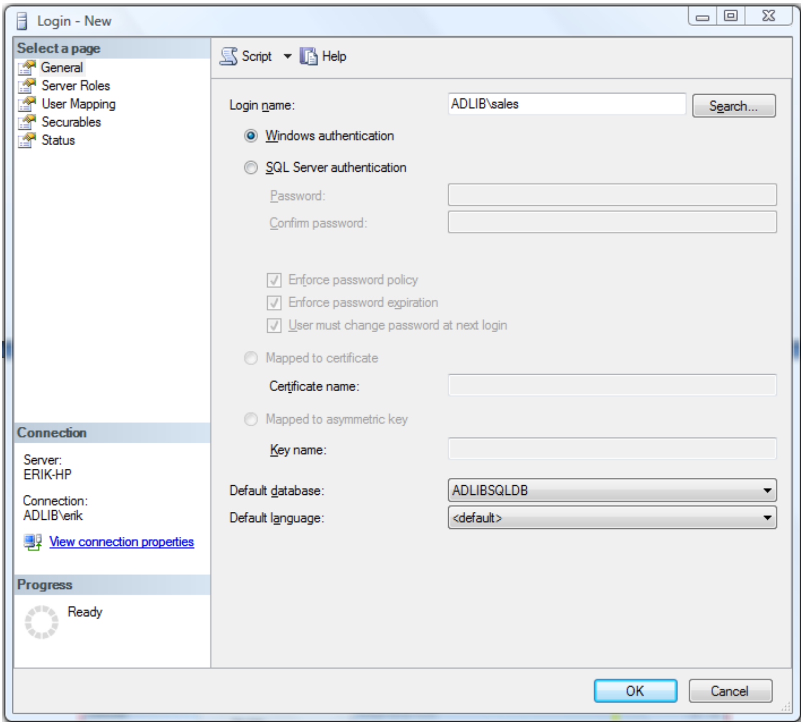
Geef de namen van de objecten op (voorbeelden): adlib

Er is meer dan één object dat met de volgende objectnaam overeenkomt: "adlib". Selecteer een object uit deze lijst, of klik op Annuleren als u de naam

Identieke namen:

Naam (RDN)	Aanmeldingsnaam (va...	E-mailadres	In map
ADLIB Gebruikersgr...	gebruikersgroep		adlibsoft.com/nl/User...
ADLIB Help Desk UK	helpdesk_uk	helpdesk@uk.adlibsof...	adlibsoft.com/uk/User...
ADLIB Helpdesk DE	helpdeskde	helpdesk@de.adlibsof...	adlibsoft.com/de/User...
Adlib IIS	adlibiis		adlibsoft.com/nl/Servi...
Adlib License Maker	AdlibLicense		adlibsoft.com/Users

4. In this example we chose the *ADLIB\sales* user group. We are now creating an SQL Server login with the same name. On this page, choose your Adlib SQL database as the *Default database*, in this example that happens to be *ADLIBSQLDB*.



Login - New

Login name: ADLIB\sales

● Windows authentication

○ SQL Server authentication

Password:

Confirm password:

☑ Enforce password policy

☑ Enforce password expiration

☑ User must change password at next login

○ Mapped to certificate

Certificate name:

○ Mapped to asymmetric key

Key name:

Default database: ADLIBSQLDB

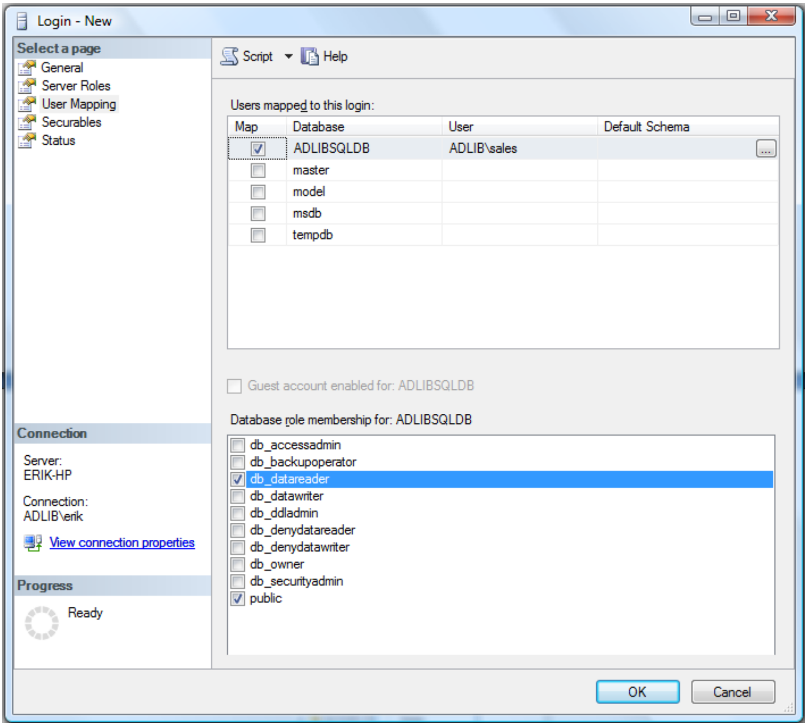
Default language: <default>

Server:
ERIK-HP

Connection:
ADLIB\erik

Progress: Ready

5. Leave the *Server Roles* to *public*, and proceed directly to the *User mapping* page – in the top left of the current window you select a page. On this page in the upper list, mark your Adlib SQL database, and in the list below it, mark the role(s) you want to assign to the current login, in this example: *db_datareader* (so that this user group may only view data, not edit it). Leave the *public* role marked by default. Click *OK* to close the window.



Users mapped to this login:

Map	Database	User	Default Schema
☑	ADLIBSQLDB	ADLIB\sales	
☐	master		
☐	model		
☐	msdb		
☐	tempdb		

☐ Guest account enabled for: ADLIBSQLDB

Database role membership for: ADLIBSQLDB

☐ db_accessadmin
☐ db_backupoperator
☑ db_datareader
☐ db_datawriter
☐ db_ddladmin
☐ db_denydatareader
☐ db_denydatawriter
☐ db_owner
☐ db_securityadmin
☑ public

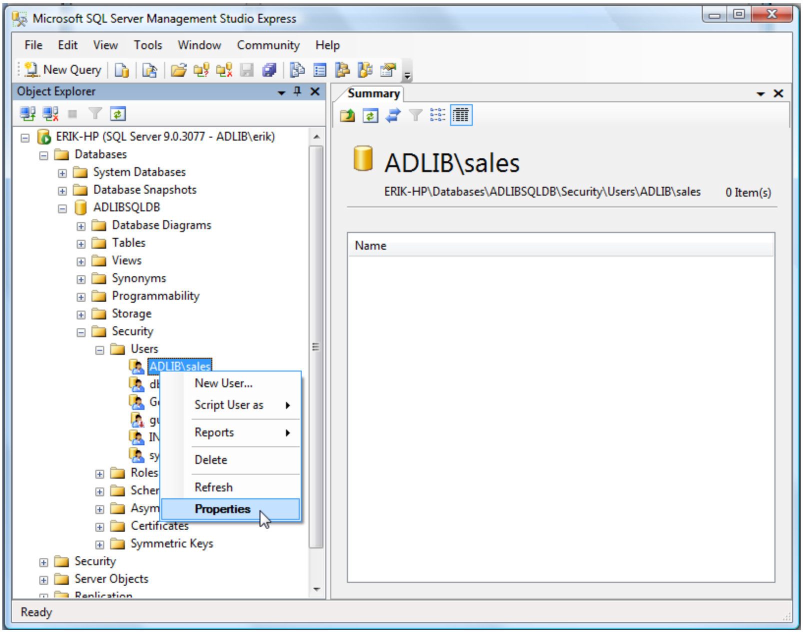
Connection

Server:
ERIK-HP

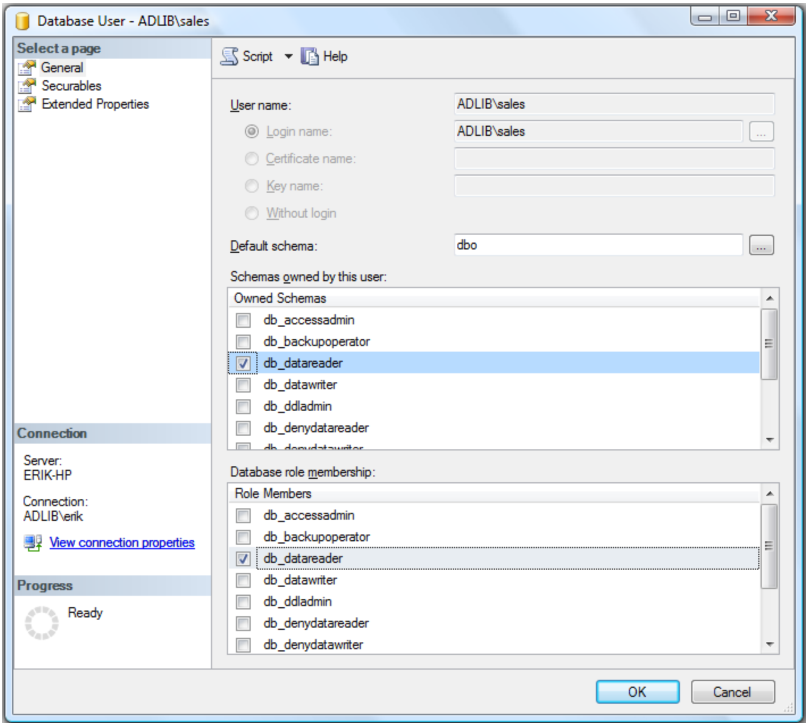
Connection:
ADLIB\erik

Ready

6. In the *Object Explorer*, now open your Adlib SQL database, with in it the *Security* folder and subsequently the *Users* folder. Right-click the user group you just added, *ADLIB\sales* in this example, and choose *Properties* in the pop-up menu which opens.



7. Here, mark the desired schema for this user: it should be the same as the database role(s) marked in the list below it: *db_datareader* in this example. The Active Directory user group has now been added as an SQL Server user, with read-only access rights.



Database User - ADLIB\sales

Server: ERIK-HP

Connection: ADLIB\erik

Ready

User name: ADLIB\sales

● Login name: ADLIB\sales

○ Certificate name:

○ Key name:

○ Without login

Default schema: dbo

Schemas owned by this user:

Owned Schemas
- □ db_accessadmin
- □ db_backupoperator
- ☑ db_datareader
- □ db_datawriter
- □ db_ddladmin
- □ db_denydatareader
- □ db_denydatawriter

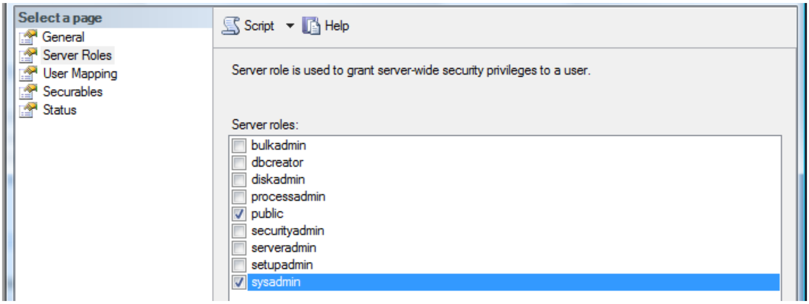
Database role membership:

Role Members
- □ db_accessadmin
- □ db_backupoperator
- ☑ db_datareader
- □ db_datawriter
- □ db_ddladmin
- □ db_denydatareader
- □ db_denydatawriter

8. Repeat this procedure (the steps 2 up to and including 7) for the other Active Directory users or user groups and assign the desired access rights to everyone of them. Note that if you assign the *db_datawriter* role, you should also assign the *db_datareader* role.

9. Also add at least one Active Directory user, probably yourself, who gets the *db_owner* role as SQL Server user. For this user, set the *Default database* in step 4 to *master* (all databases together): this in case there is more than one SQL database which you should be allowed to manage (for instance when a copy of your live Adlib SQL database has been made, for testing purposes). And in step 5 you now do open the *Server Roles* page to assign the database owner the *sysadmin* role as well. So, in the *User Mapping* you not

only mark the *public* role, but the *db_owner* role too; here, you can also select the databases which may actually be managed by this user. In step 7, assign the *db_owner* schema to this user.

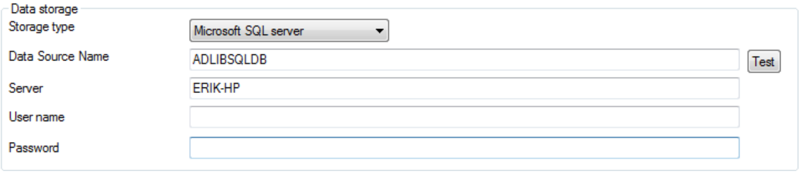


Server role is used to grant server-wide security privileges to a user.

Server roles:

- ☐ bulkadmin
- ☐ dbcreator
- ☐ diskadmin
- ☐ processadmin
- ☑ public
- ☐ securityadmin
- ☐ serveradmin
- ☐ setupadmin
- ☑ sysadmin

9. Now you can close Microsoft SQL Server Management Studio Express. Open Adlib Designer to test the connection between Adlib and the SQL Server. In the Application browser, open the folder in which the *.inf* files of your Adlib SQL database are located, and select a random *.inf* file, for example that of *DOCUMENT*. The *User name* and *Password* can be left empty, because logging onto the SQL Server is now done with the Active Directory login.
Note that we assume here that you've already set the *Storage type*, *Data Source Name* and *Server* options on this tab correctly. If not, then do that now (for all Adlib databases).



Data storage

Storage type	Microsoft SQL server
Data Source Name	ADLIBSQLDB
Server	ERIK-HP
User name	
Password	

Then click the *Test* button behind the *Data Source Name* entry field to test the connection with the SQL Server. If the connection is successful, the text *OK* appears above the button. If anything goes wrong, you'll be so notified and the red text *ERR* appears above the *Test* button. Then check your settings on this tab, and the settings in the SQL Server.

All Adlib users can now access the SQL database with their Active Directory user name and their own Windows password, with the access rights as defined for their login in SQL Server. This probably protects your database enough, but you can always still use the different internal Adlib mechanisms to refine the access rights for individual users. Do make sure that no conflicting access rights are set this way: this can of course lead to unexpected situations and confusion. You could keep an overview of SQL Server rights and Adlib access rights assigned to users. See the Designer Help for more information about access rights on Adlib level.