

Failover operation for PostgreSQL databases



Installation manual for system providers

3/25/2020

Product line neo, version 6.x

The described functions can be used with the following ASC products:

EVOIPneo

EVOLUTIONneo / XXL / eco

EVOflex (country-specific)

Please note that you can always find the most up-to-date technical documentation and product updates in the partner area on our website at <http://www.asctechnologies.com>.

Copyright © 2019 ASC Technologies AG. All rights reserved.

Windows is a registered trademark of Microsoft Corporation. VMware® is a registered trademark of VMware, Inc. All other marks and names mentioned herein may be trademarks of their respective companies.

Contents

1	General information	5
2	Introduction	6
3	Remarks	7
4	Installation of the databases	8
4.1	Primary database	8
4.2	Standby database	10
5	Configure failover operation	14
5.1	Preconditions	14
5.1.1	Network share	14
5.1.2	Start Postgres service within the PostgreSQL account.....	15
5.2	Configure failover operation with Failover Configuration Tool	16
5.2.1	Add additional application server	20
5.3	Configure failover operation manually.....	21
5.3.1	Configuration of the database servers	21
5.3.1.1	Set up primary database	21
5.3.1.2	Create standby database	25
5.3.1.3	Adjust setup.xml file	27
5.3.2	Configuration of the application servers	28
5.3.2.1	Adjust setup.xml file	28
5.3.2.2	Configure database connection	28
5.4	Function test.....	33
5.4.1	Check replication on the primary server.....	33
5.4.2	Check replication on the standby server	33
5.4.3	Check network share	33
5.4.4	Check deletion of the WAL archives	33
6	Failover operation functional description	34
6.1	Failure of the primary database	34
6.2	Failure of the standby database	35
7	Reset failover operation	36
7.1	Reset failover operation manually.....	36
7.1.1	Restore standby server	37
7.2	Reset failover operation with Failover Configuration Tool.....	38
7.2.1	Use failover database as primary database	38
8	Attachment	40
8.1	File pg_hba.conf, primary server.....	40
8.2	File pg_hba.conf, standby server	40
8.3	File postgresql.conf, primary server	41
8.4	File postgresql.conf, standby server	41

8.5	File recovery.conf, standby server	42
8.6	File setup.xml, database server	42
8.7	File setup.xml, application server	42
8.8	File domain.xml, application server	43
	List of figures	45
	List of tables	46
	Glossary	47

General information

In the context of this document ASC represents ASC Technologies AG, its subsidiaries, branch offices, and distributors. An up-to-date overview of the aforementioned entities can be found at <https://www.asctechnologies.com>

ASC assumes no guarantee for the actuality, correctness, integrity or quality of the information provided in the manuals.

ASC regularly checks the content of the released manuals for consistency with the described hardware and software. Nevertheless, deviations cannot be excluded. Necessary revisions are included in subsequent editions.

Some aspects of the ASC technology are described in general terms to protect the ownership and the confidential information or trade secrets of ASC.

The software programs and the manuals of ASC are protected by copyright law. All rights on the manuals are reserved including the rights of reproduction and multiplication of any kind, be it photo mechanical, typographical or on digital data media. This also applies to translations. Copying the manuals, completely or in parts, is only allowed with written authorization of ASC.

Representative, if not defined otherwise, is the technical status at the time of the delivery of the software, the devices and the manuals of ASC. Technical changes without specified announcements are reserved. Previous manuals lose their validity.

The general conditions of sales and delivery of ASC in their latest version apply.

2 Introduction

To secure the access to the recordings in case of a failure of the database, you can set up a failover capability with another database.

This manual describes how to set up a failover concept with two PostgreSQL databases and which steps to take to reset the failover operation once the primary database is available again.

The failover functionality is available for [single-core systems](#) as well as for [multi-core systems](#).

You can carry out the configuration with IPv4 as well as with IPv6 IP addresses.

The two PostgreSQL databases must be installed on different servers. You can choose to install either on application servers ([app servers](#)) or on any other server in the recording system. During the installation of the recording software, you tell the recording system on which servers in your recording system the databases are installed.



For information about the installation of the *neo* software refer to the installation manual for system providers *Installation of the recording software of ASC*.

Once you have installed the two databases and configured the failover operation, the Database Manager module is available to you in the application System Configuration. In this module, you can monitor the failover operation.



For information about the Database Manager module refer to the administration manual for system providers *Database Manager*.



This manual uses the terms primary server and standby server in the following sense:

Primary server = master server on which the primary database is located

Standby server = slave server on which the standby database is located

3

Remarks

This manual describes changes in the configuration which are carried out manually. IP addresses, directories etc. are replaced by placeholders. You on your part have to replace these placeholders with the values of the actual system environment.



Do not only replace the text in the angle brackets but the entire placeholder!
E. g. *Host replication postgres <<IP-ADDRESS-STANDBY-DB>>/<<CIDR>> md5* becomes *host replication postgres 192.168.0.0/32 md5*.

You can carry out the configuration with IPv6 IP addresses.

In general, hexadecimal representation is used, e. g.

FE80::29ED:F6FB:D58D:F92B/128md5



Where the path must be indicated please observe the following spelling

FE80--29ED-F6FB-D58D-F92B.ipv6-literal.net

The exceptions only apply for manual configuration and have been highlighted explicitly.

The following placeholders are used:

Placeholder	Description
<<IP-ADDRESS-PRIMARY-DB>>	IP address of the primary database
<<IP-ADDRESS-STANDBY-DB>>	IP address of the standby database
<<WAL-ARCHIVE>>	Name of the replication directory for depositing the WAL archives (Default: <i>WAL-Archive</i>)
<<WAL-SEGMENTS>>	Number of the WAL segments which are held available by the primary database.
<<CIDR>>	Value for the netmask when using Classless Inter-Domain Routing (CIDR).
<<POSTGRES-INSTALL-FOLDER>>	Drive and path of the PostgreSQL installation directory (Default: <i>C:\Program Files\PostgreSQL\n.m</i> NOTICE! <i>n.m</i> stands for the version number of the PostgreSQL in use, e. g. <i>C:\Program Files\PostgreSQL\9.5</i> .)
<<POSTGRES-DATA-FOLDER>>	Drive and path of the PostgreSQL data directory (Default: <i>D:\ASCDB</i>)
<<POSTGRES-DATA-DRIVE>>	Drive of the PostgreSQL data directory (Default: <i>D:</i>)
<<IP-ADDRESS-NEO-CORE>>	IP address of an neo application server
<<NEO-INSTALL-FOLDER>>	Drive and path of the installation directory of the neo Suite (Default: <i>C:\Program Files (x86)\ASC\ASC Product Suite</i>)
<<DELAY>>	Period of time to pass by before failover to the standby server is initiated in case the connection to the primary server fails. Unit: milliseconds

4 Installation of the databases



Make sure that you have all administrator rights for the installation.



You have to use the same PostgreSQL version for the two databases



For the communication between the primary and the standby database and all neo application servers the following port must be open in the firewall:

- Port 5432



In the operating system of both database servers, file and printer sharing must have been activated.

4.1 Primary database

The primary database can either be installed as local database on a neo application server or as external database on a separate server.



During the installation of multi-server systems, the server on which the database is supposed to run has to be installed first.

When using failover databases, the server on which the primary database is supposed to run has to be installed first.

The installation is carried out during the course of the installation of the ASC software.

1. Insert the installation medium for the ASC software.
2. From the context menu of the file *setup.exe*, select the menu item *Run as Administrator*.
3. Apply the default settings of the installation steps until reaching the installation of the database.
4. Activate the feature *Application Server* if you would like to use all features on this server.
5. Activate the feature *Database* to install an internal PostgreSQL database.

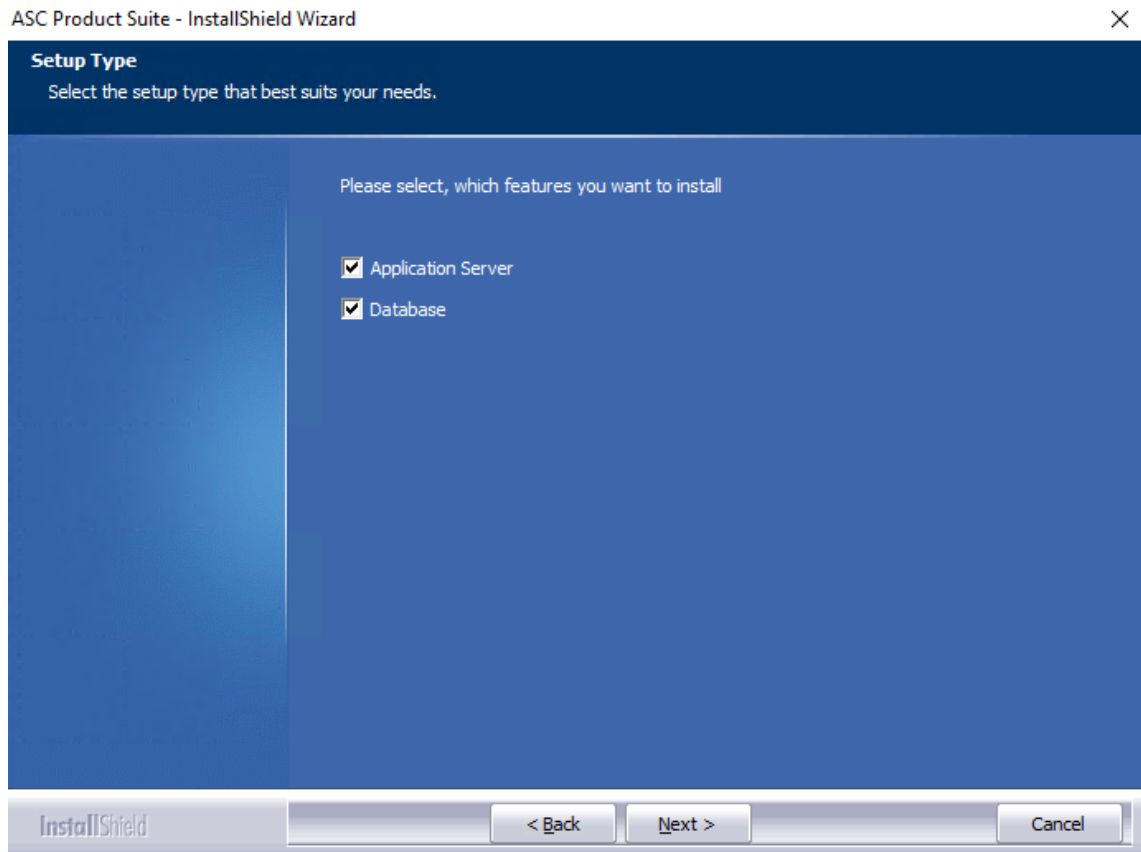


Fig. 1: Select features for the installation

6. Click on the button *Next*.
7. Click on the button *Browse* to select the target drive.

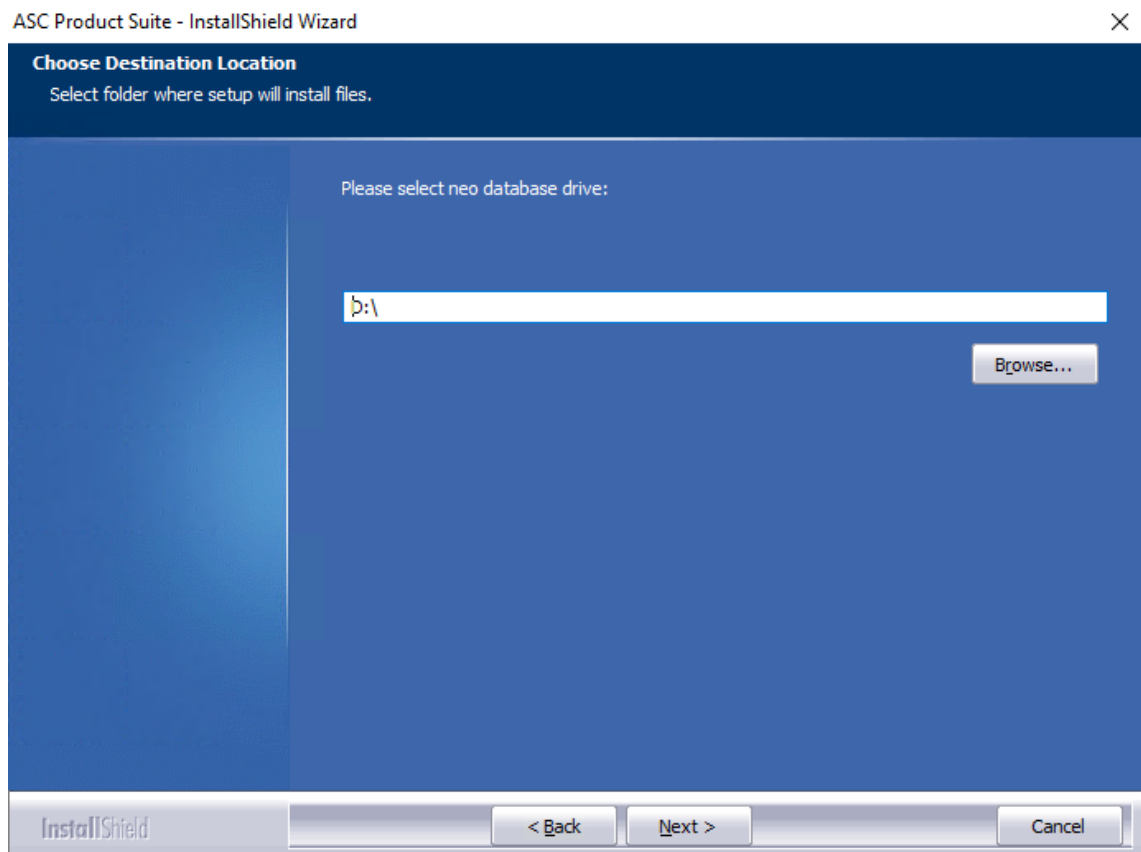


Fig. 2: Select target drive for the internal database

8. Click on the button *Next* to carry out the next steps of the installation.
9. Apply the default settings in all following installation steps until the end of the installation routine.
10. In the last step of the installation routine, select the option *Yes, I want to restart my computer now*.

ASC Product Suite - InstallShield Wizard

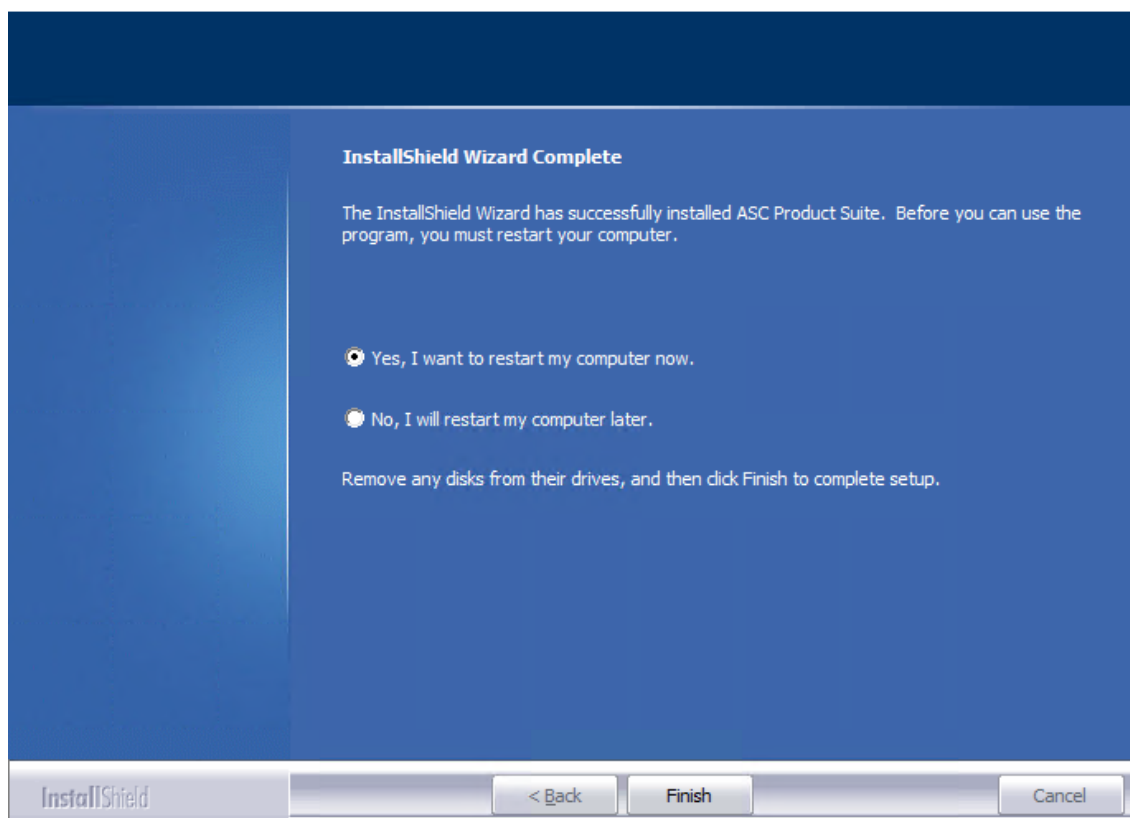


Fig. 3: Finish installation and restart server

11. Complete the installation of the ASC software by clicking on the button *Finish*.
 - ⇒ The installation is finished.
 - ⇒ The server is restarted automatically.
12. Make sure that the following PostgreSQL services have been **activated** on the primary database:
 - postgresql-x64-n.m - PostgreSQL Server n.m
NOTICE! *n.m* stands for the version number of the PostgreSQL in use, e. g. *postgresql-x64-9.5 - PostgreSQL Server 9.5*.
 - PostgreSQL Scheduling Agent - pgAgent

4.2 Standby database

The standby database can either be installed as local database on a [neo application server](#) or as external database on a separate server.

The installation on a [neo application server](#) is carried out during the course of the installation of the ASC software, see installation manual *Installation of the recording software of ASC*.

Since the ASC service ServiceMan has to run on the standby server, the installation of a separate standby server must be carried out via the setup of the ASC software.

To **install a separate standby database**, proceed as follows:

1. Insert the installation medium for the ASC software.

2. From the context menu of the file *setup.exe*, select the menu item *Run as Administrator*.
3. Apply the default settings of the installation steps until reaching the installation of the database.
4. Install an internal PostgreSQL database by selecting the feature *Database*.

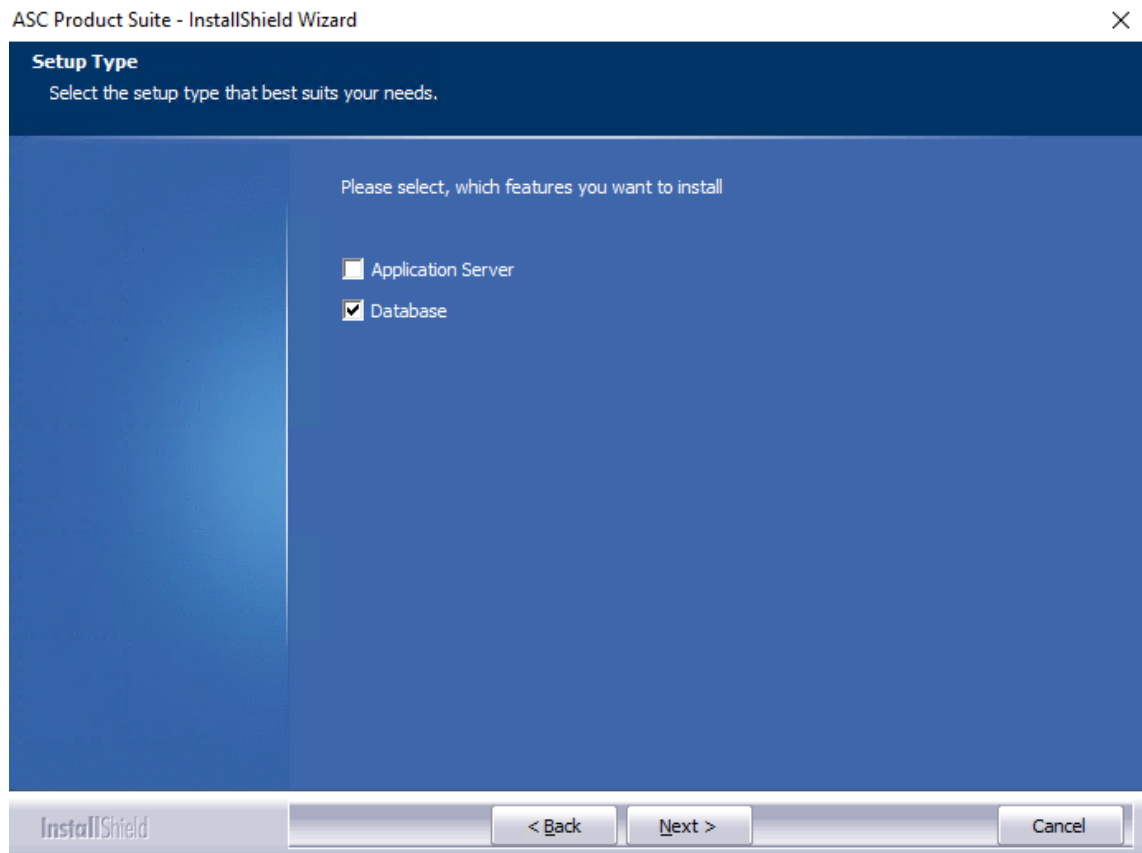


Fig. 4: Select features for the installation

5. Click on the button *Next*.
6. Click on the button *Browse* to select the target drive.

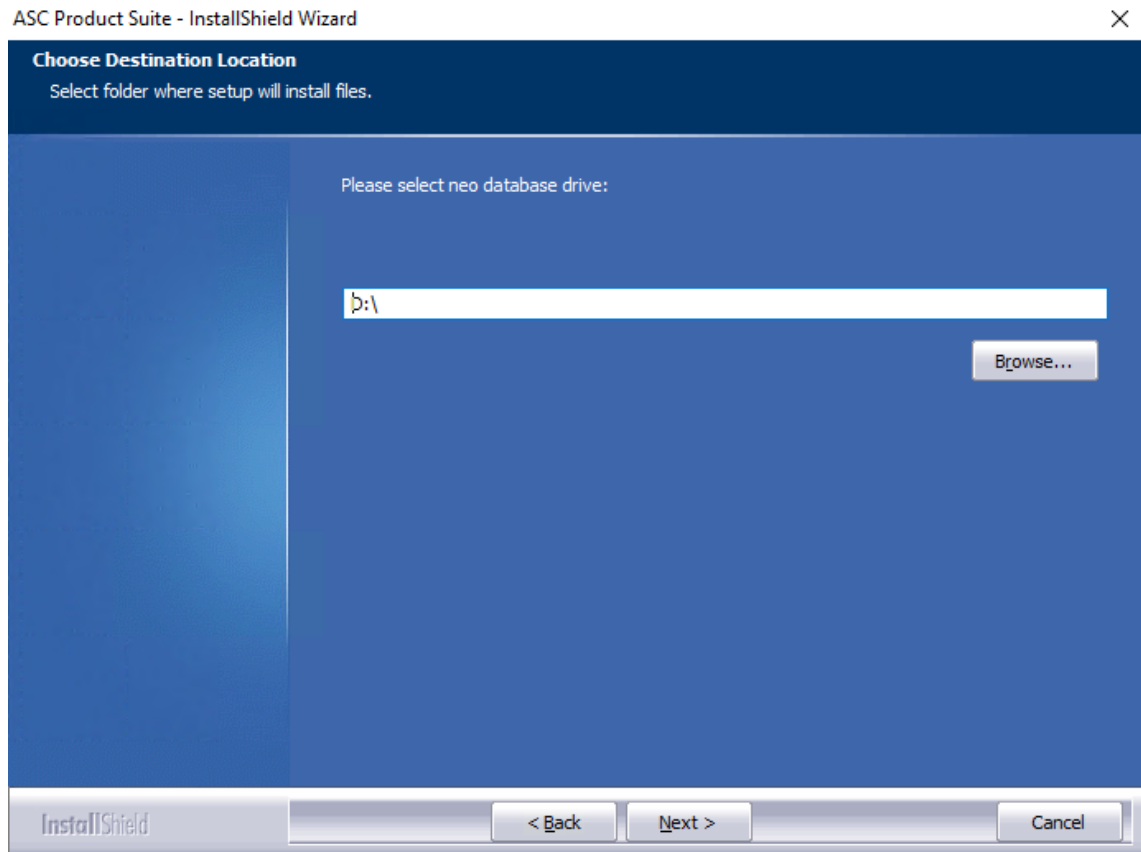


Fig. 5: Select target drive for the internal database

7. Click on the button *Next* to carry out the next steps of the installation.
8. Apply the default settings in all following installation steps until the end of the installation routine.
9. In the last step of the installation routine, select the option *Yes, I want to restart my computer now*.

ASC Product Suite - InstallShield Wizard

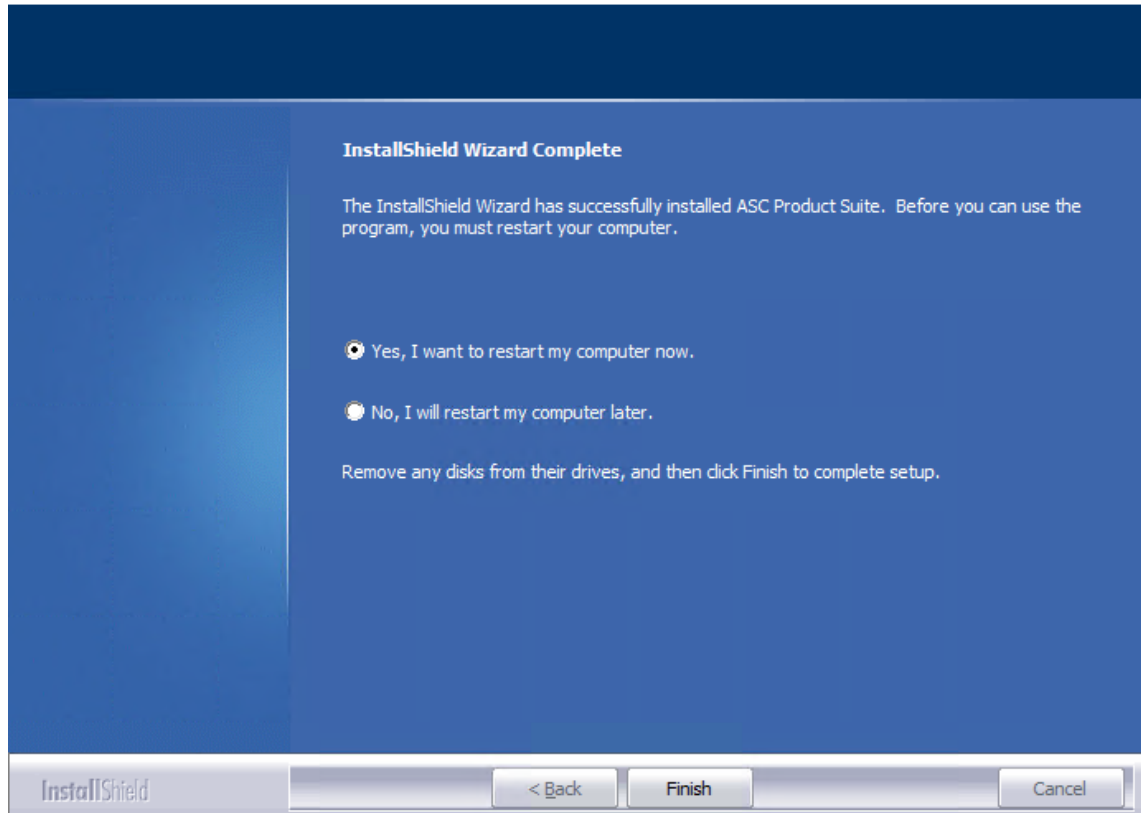


Fig. 6: Finish installation and restart server

10. Complete the installation of the ASC software by clicking on the button *Finish*.
 - ⇒ The installation is finished.
 - ⇒ The server is restarted automatically.
11. Make sure that the following PostgreSQL service has been **activated** on the standby database:
 - postgresql-x64-n.m - PostgreSQL Server n.m

NOTICE! *n.m* stands for the version number of the PostgreSQL in use, e. g. *postgresql-x64-9.5 - PostgreSQL Server 9.5*.

5

Configure failover operation

There are two alternatives to configure failover operation:

- *Configuration with the Failover Configuration Tool*, see [chapter "Configure failover operation with Failover Configuration Tool"](#), p. 16.
- *Manual configuration of the database servers and the application servers*, see [chapter "Configure failover operation manually"](#), p. 21.

5.1

Preconditions

5.1.1

Network share

To ensure that failover operation of the two database servers works properly, the drives of the two servers must be shared and the user *postgres* must be granted full access.

1. Start the Windows Explorer.
2. Right-click on drive F:\.
3. From the context menu, select the menu item *Share with*
4. Click on the button *Advanced Sharing*.
⇒ The window *Advance Sharing* appears.
5. Activate the check box *Share this folder*.
⇒ The settings become active and the share name *F* is suggested.
6. Click on the button *Permissions* to assign permissions to a user or a group.
⇒ The window *Permissions for F* appears.

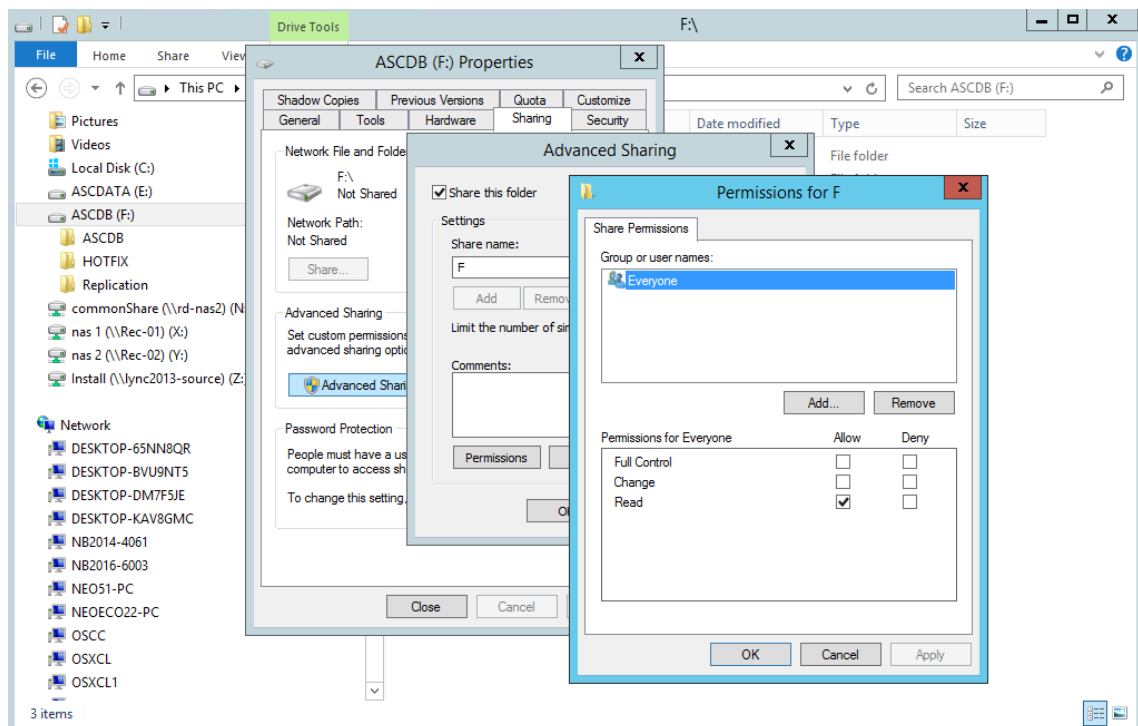


Fig. 7: Share drive

7. Click on the button *Add*.
⇒ The window *Select Users or Groups* appears.

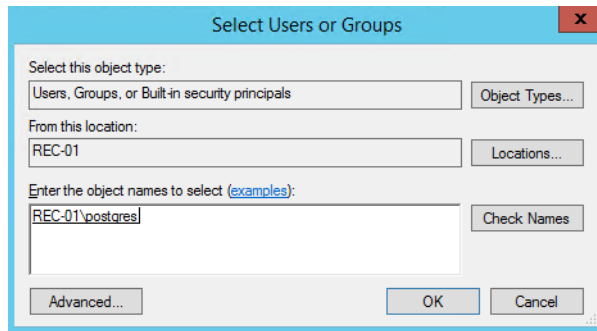


Fig. 8: Share for user *postgres*

8. In the entry field *Enter the object names to select (examples):*, enter the user *postgres*.
9. Click on the button *Check Names*.
10. The user is displayed as **FQDN**.
11. Click on the button *OK* to assign the user and close the window.
12. Activate the check box *Full Control* to grant the user full control.
13. Click on the button *OK* to apply the settings and to close the window.

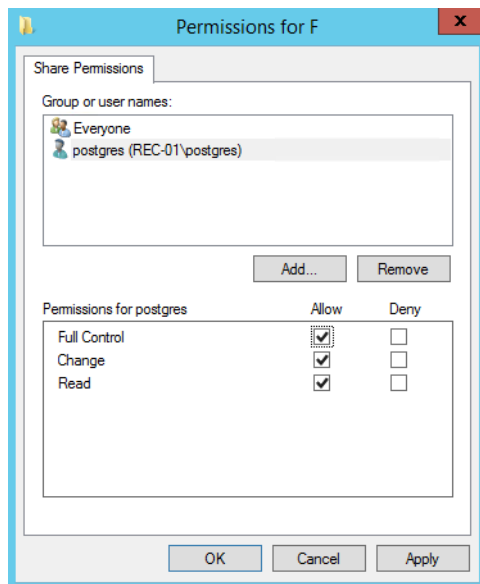


Fig. 9: Grant user *postgres* full control

14. Click on the button *OK* to apply the settings and to close the window.
15. Share the network on the server with the standby database, too.

5.1.2 Start Postgres service within the PostgreSQL account

For the replication, the PostgreSQL service must be started within the context of the local PostgreSQL account for both servers.

1. Select the service *postgresql-x64-9.5* in the overview of services.
2. Select the entry *Properties* from the context menu.
 - ⇒ The following window appears:

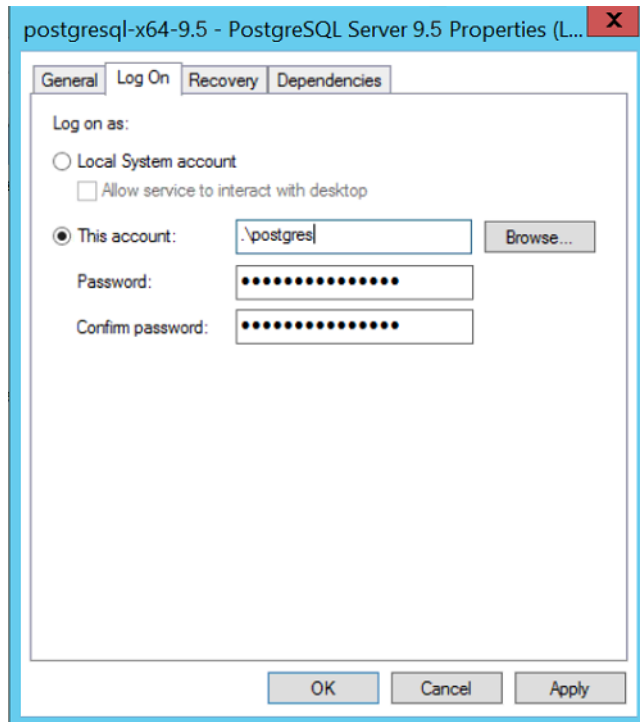


Fig. 10: Configure login for the Postgres service

3. Select the tab *Log On*.
4. Activate the option *This account* and enter the name `.\postgres` and the password for this user.

5.2 Configure failover operation with Failover Configuration Tool

The Failover Configuration Tool is used to automatically configure a database failover for two PostgreSQL databases and any number of [app servers](#).



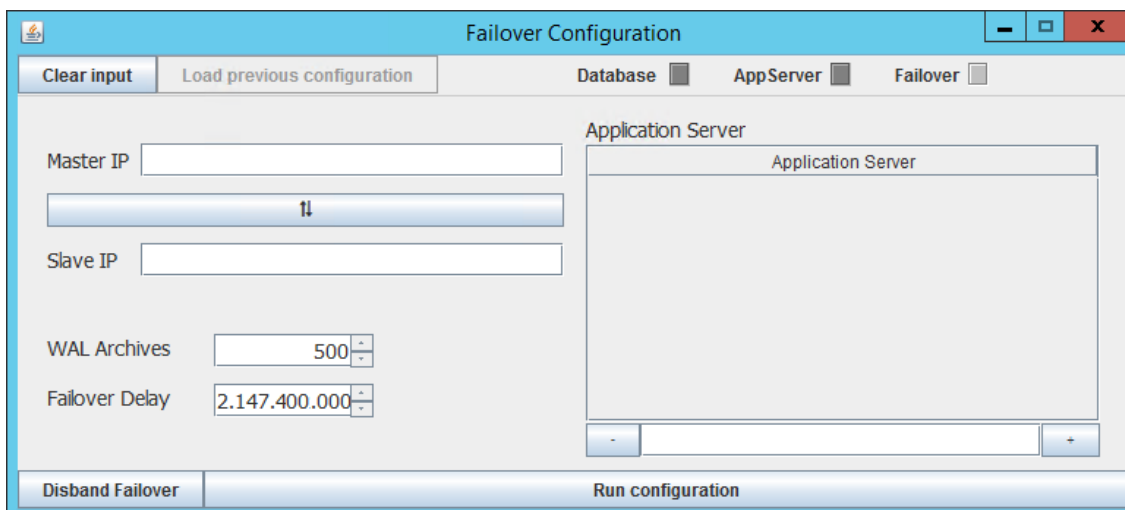
Always use the version of the tool delivered in the respective version.

Always start the tool on the server that is destined as the primary server upon initiation. Otherwise the configuration will fail.



When using the Failover Configuration Tool for the configuration, the databases and the app servers are stopped!

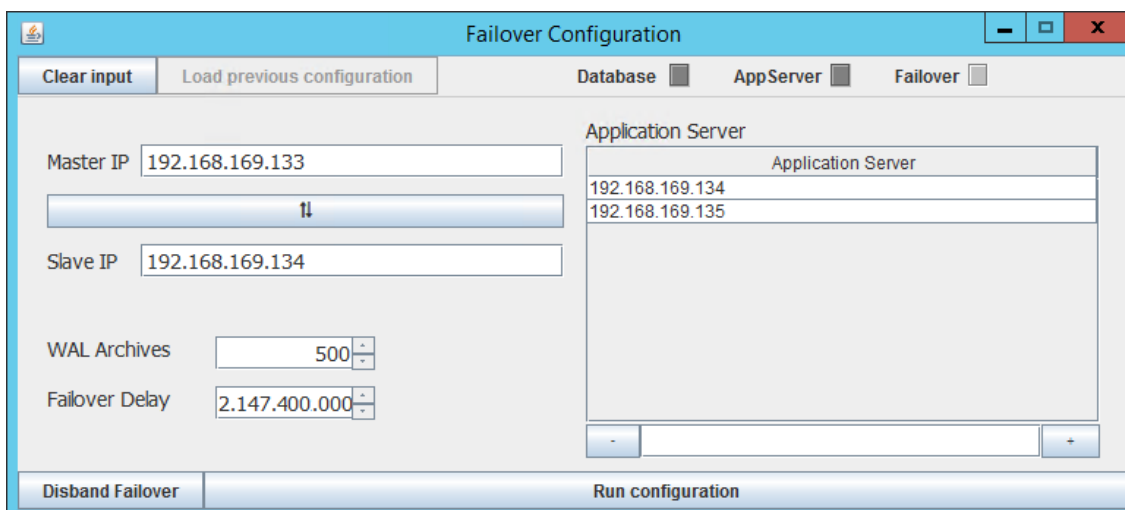
1. Open the Windows Explorer and change to the directory `C:\Program Files (x86)\ASC\ASC Product Suite\scripts`.
2. Open the file `DBFailoverConfig.exe` with a double-click.
 - ⇒ The Failover Configuration Tool appears.



The screenshot shows the 'Failover Configuration' window. It has three tabs: 'Clear input', 'Load previous configuration', and 'Failover'. The 'Failover' tab is selected. On the left, there are input fields for 'Master IP', 'Slave IP', 'WAL Archives' (set to 500), and 'Failover Delay' (set to 2.147.400.000). A double-headed arrow button is between the Master and Slave IP fields. On the right, there is an 'Application Server' section with a list box and a '+' button at the bottom. At the bottom of the window, there are buttons for 'Disband Failover' and 'Run configuration'.

Fig. 11: Failover Configuration Tool - Configure failover

3. Enter the following parameters:



This screenshot shows the same 'Failover Configuration' window as Fig. 11, but with the following values entered: 'Master IP' is 192.168.169.133, 'Slave IP' is 192.168.169.134, 'WAL Archives' is 500, and 'Failover Delay' is 2.147.400.000. In the 'Application Server' list box on the right, two IP addresses are listed: 192.168.169.134 and 192.168.169.135.

Fig. 12: Configure failover components

Master IP	Enter the IP address (IPv4 or IPv6) of the master database. The values of the subnet mask /32 (IPv4) or /128 (IPv6) are added automatically. That way, only the indicated IP address is shared for the database. If you would like to configure an address range for accessing the database, you have to add those values manually.
Slave IP	Enter the IP address (IPv4 or IPv6) of the slave database. The values of the subnet mask /32 (IPv4) or /128 (IPv6) are added automatically. That way, only the indicated IP address is shared for the database. If you would like to configure an address range for accessing the database, you have to add those values manually.
Application server	If an application server is running outside the master IP or slave IP, enter the IP address of the application server in the entry field at the bottom at the right and add it to the list by clicking on the buttons + . Repeat the steps for additional separate application servers. The values of the subnet mask /32 (IPv4) or /128 (IPv6) are added automatically. That way, only the indicated IP address is shared for the database. If you would like to configure an address range for accessing the database, you have to add those values manually.

WAL Archives	<p>By means of the rotating fields, select the number of WAL segments which are supposed to be shared simultaneously. The number affects the duration until the replication is stopped in case the database disconnects.</p> <p>The directory <<POSTGRES-DATA-DRIVE>>\Replication in which the WAL archives are located is created during the installation of the database.</p>
Failover Delay	<p>By means of the rotating fields, select the time in milliseconds that an application server is supposed to wait before triggering the failover and changing to the slave database in case the connection to the master database is lost.</p> <p>Select 2.147.400.000 milliseconds as default value.</p>

4. Click on the button *Run configuration* to apply the configuration.

⇒ The following window appears:

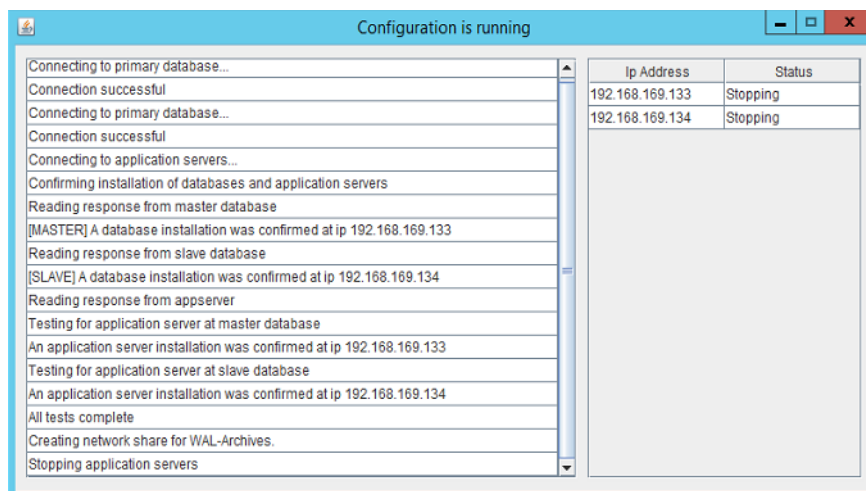


Fig. 13: Watch the progress of the configuration

- ⇒ All relevant information about the process is displayed in the left window. The connections to other systems with their respective connection or operation status are displayed in the right window.
- ⇒ *Warnings are highlighted in yellow.*
- ⇒ *Errors are highlighted in red.*
- ⇒ If the configuration is complete, a success message appears.

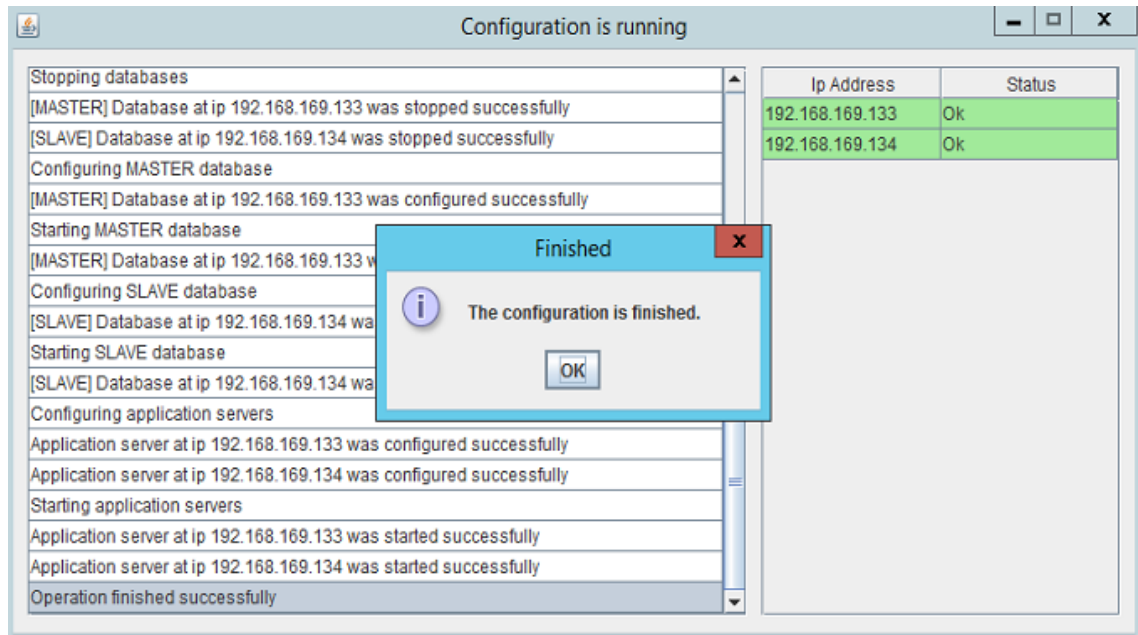


Fig. 14: Success message

NOTICE! Once the process has been completed, the application servers are rebooted automatically. It can take a certain time until the applications are available again.

After the configuration with the Configuration Tool, the status of the local components *Database*, *AppServer* and *Replication* are displayed at the top right. Depending on the status of the components, the indicator lights are displayed in different colors. When hovering the mouse cursor over the indicator, a text summary is displayed.

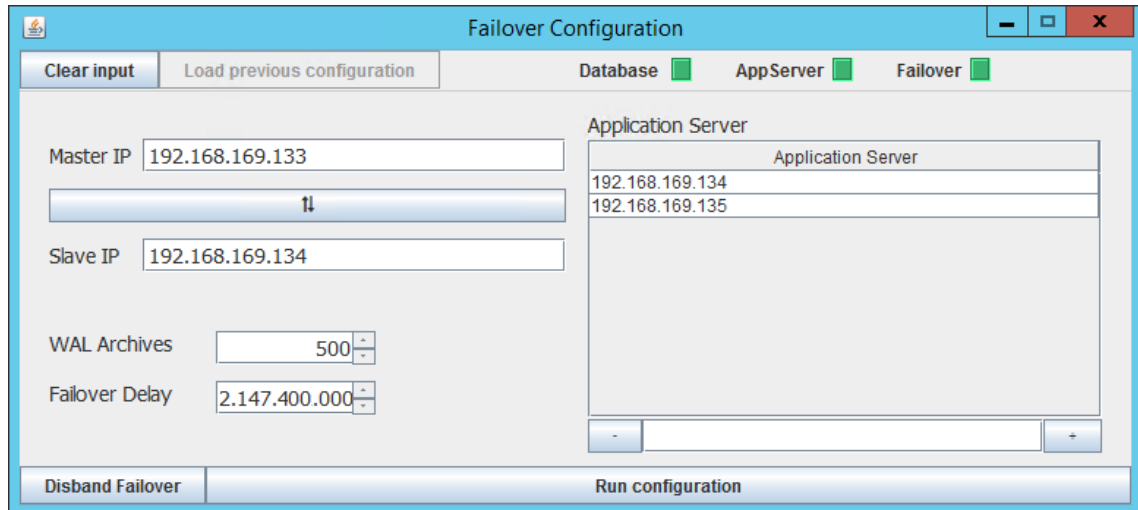


Fig. 15: Status indicator of failover components:

Color legend:














Database 	No local installation found
AppServer 	
Database 	Not configured /
AppServer 	Running in single mode
Database 	Failover configured
AppServer 	
Database 	Unknown /
AppServer 	Error state
Failover 	No Database / AppServer
Failover 	Not configured
Failover 	Failover active
Failover 	Replication stopped (Failover might be triggered)
Failover 	Unknown /
	Error state

Fig. 16: Color legend

5.2.1 Add additional application server

To include another server in the group, you have to call up the Failover Configuration Tool again.

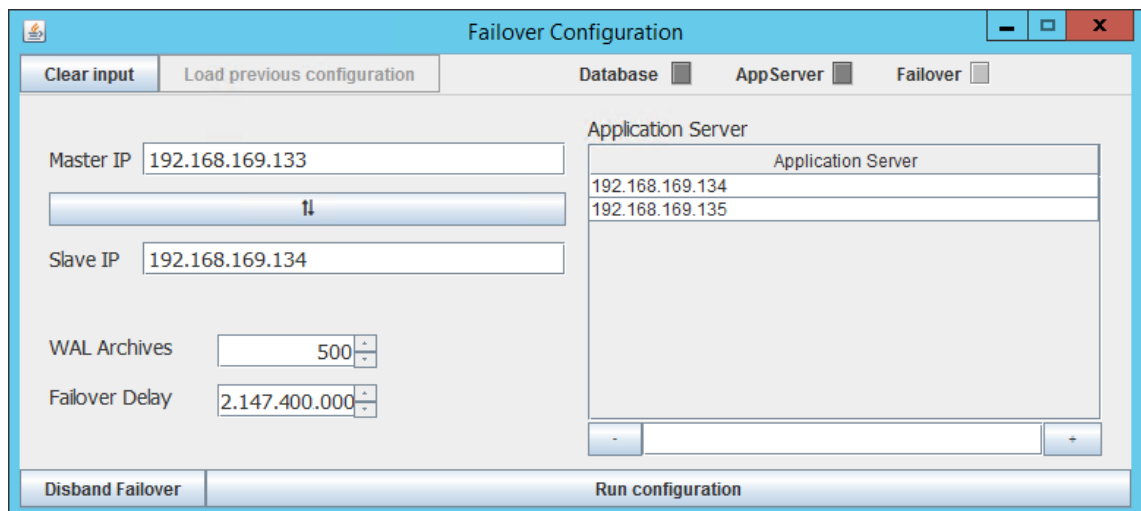


Fig. 17: Call up Failover Configuration Tool again

1. In the menu bar, at the top left, click on the button *Load previous configuration* to load the previous configuration in the dialog window.
2. Add another application server to the list of application servers by clicking on the button + at the bottom at the right.
3. Click on the button *Run Configuration* to carry out the configuration again.

When creating the configuration by means of the Failover Configuration Tool, you can skip the manual configuration and go directly to [chapter "Failover operation functional description"](#), p. 34.

5 Configure failover operation

5.3 Configure failover operation manually

5.3.1 Configuration of the database servers

5.3.1.1 Set up primary database

Proceed in the order of the following sections to set up the failover operation on the primary server.

5.3.1.1.1 Setup replication directory

Create a replication directory. The primary database writes the [WAL](#) archive into this directory. The standby database reads out the [WAL](#) archives from this directory and subsequently deletes the [WAL](#) archives.

1. In parallel to the directory <<PGRES-DATA-FOLDER>> create the directory <<WAL-ARCHIVE>>.
2. Share the directory <<WAL-ARCHIVE>> in the network with the option *Full Access*.

5.3.1.1.2 Network share

To ensure that failover operation of the two database servers works properly, the drives of the two servers must be shared and the user *postgres* must be granted full access.

1. Start the Windows Explorer.
2. Right-click on drive F:\.
3. From the context menu, select the menu item *Share with*
4. Click on the button *Advanced Sharing*.
⇒ The window *Advance Sharing* appears.
5. Activate the check box *Share this folder*.
⇒ The settings become active and the share name *F* is suggested.
6. Click on the button *Permissions* to assign permissions to a user or a group.
⇒ The window *Permissions for F* appears.

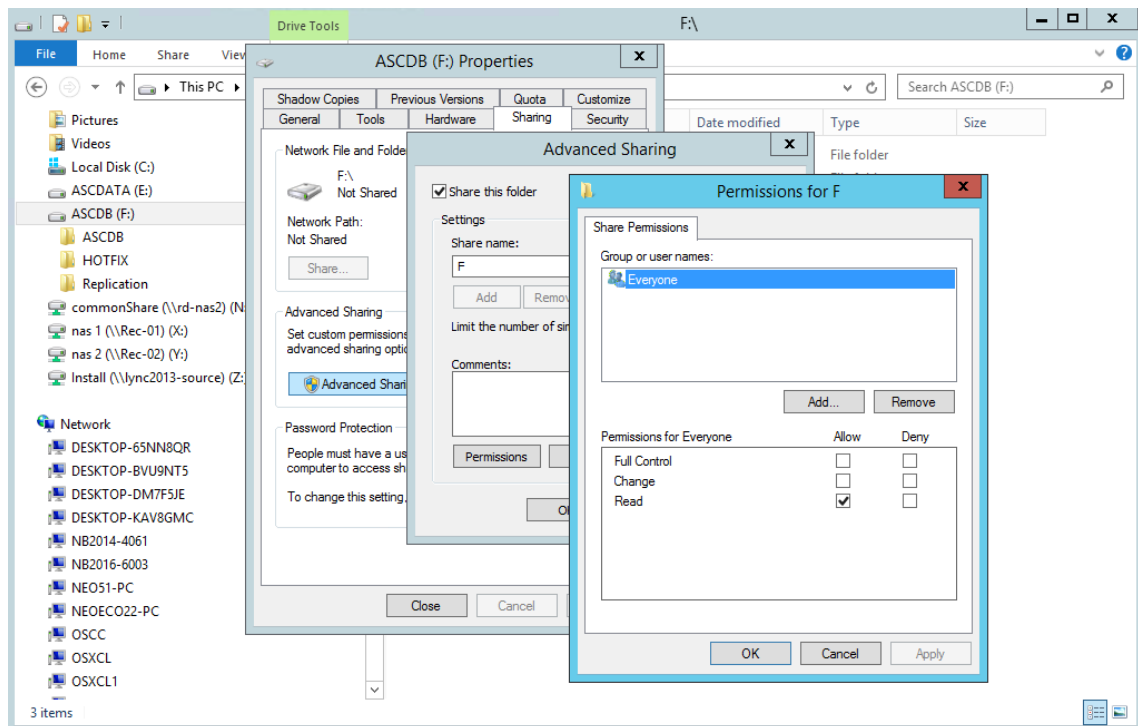


Fig. 18: Share drive

7. Click on the button *Add*.
 ⇒ The window *Select Users or Groups* appears.

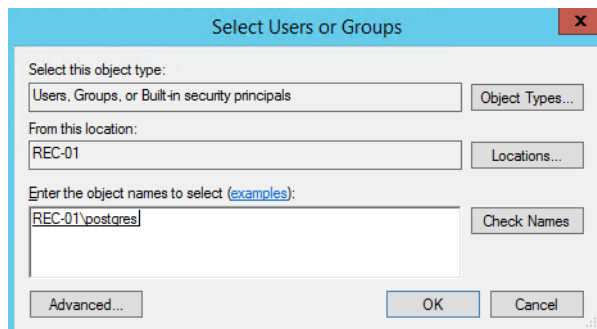


Fig. 19: Share for user *postgres*

8. In the entry field *Enter the object names to select (examples):*, enter the user *postgres*.
9. Click on the button *Check Names*.
10. The user is displayed as **FQDN**.
11. Click on the button *OK* to assign the user and close the window.
12. Activate the check box *Full Control* to grant the user full control.
13. Click on the button *OK* to apply the settings and to close the window.

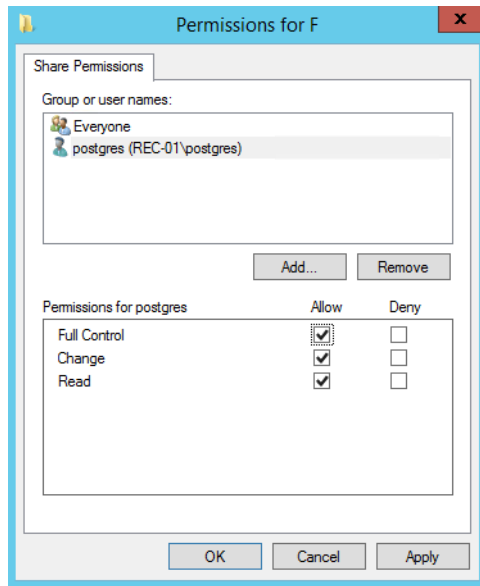


Fig. 20: Grant user *postgres* full control

14. Click on the button **OK** to apply the settings and to close the window.
15. Share the network on the server with the standby database, too.

5.3.1.1.3 Adjust file "pg_hba.conf"

The client access is authenticated by means of the settings in the file *pg_hba.conf*.

1. On the primary server in the directory <<PGRES-DATA-FOLDER>> open the configuration file *pg_hba.conf* with an editor.
2. Add the following entry at the end of the file content, to allow for replication access of the standby database on the data in the primary database:

Type	Database	User	IP address	Method
host	replication	postgres	<<IP-ADDRESS-STANDBY-DB>>/<<CIDR>>	md5

Example:

```
host replication postgres 192.168.171.120/32 md5
```

```
host replication postgres FE80::29ED:F6FB:D58D:F92B/128 md5
```

NOTICE! The subnet masks /32 (IPv4) or /128 (IPv6) apply to the indicated IP address. If you would like to configure an address range for accessing the database, you have to adjust the value to the conditions in the customer's network.

3. In addition, all *neo* application servers must have access to the PostgreSQL instance. Add the following entry for each connected *neo* application server:

Type	Database	User	IP address	Method
host	asc_rs	postgres	<<IP-ADDRESS-NEO-CORE>>/<<CIDR>>	md5
host	postgres	postgres	<<IP-ADDRESS-NEO-CORE>>/<<CIDR>>	md5

Example:

```
host asc_rs postgres 192.168.171.50/32 md5
```

```
host postgres postgres 192.168.171.50/32 md5
```



You have to use a tab character as delimiter between the values type, database, user, IP address, and method.



The value for <<CIDR>> must be provided by the system provider. The values of the subnet masks /32 (IPv4) or /128 (IPv6) apply to the indicated IP address only. If you would like to release an address range for accessing the database, you have to adjust this value.



Exemplary excerpt of an adjusted file:
[chapter "File pg_hba.conf, primary server", p. 40](#)

5.3.1.1.4 Adjust "postgresql.conf" file

The file *postgresql.conf* is the central configuration file for PostgreSQL. Adjust the following settings to enable replication.

1. On the primary server in the directory <<POSTGRES-DATA-FOLDER>>, open the configuration file *postgresql.conf* with an editor.
2. Enter the following values in the file and make sure that the entries are not commented out:
 - `max_wal_senders = 10`
 - `wal_level = hot_standby`
 - `archive_mode = on`
 - `archive_command = 'copy %p \\\\IP-ADDRESS-PRIMARY-DB>>\\<<WAL-ARCHIVE>> \\%f'`
 - `wal_keep_segments = <<WAL-SEGMENTS>>`

Example for IPv4:

```
max_wal_senders = 10
wal_level = hot_standby
archive_mode = on
archive_command = 'copy %p \\\\192.168.171.110\\WAL-ARCHIVE\\%f'
wal_keep_segments = 500
```

NOTICE! This is a path specification and requires the IPv6 IP address to be written in '-' and '.ipv6-literal.net'

Example for IPv6:

```
max_wal_senders = 10
wal_level = hot_standby
archive_mode = on
archive_command = 'copy %p \\\\FE80--29ED-F6FB-D58D-F92B.ipv6-literal.net\\WAL-ARCHIVE\\%f'
wal_keep_segments = 500
```

The number of *wal_keep_segments* is an indicator how long the replication process to the standby database can be interrupted to be resumed later on. The higher the value, the longer this interruption may be without having to manually set up the replication again.



The number of the [WAL](#) segments written per time unit depends on the usage of the database and cannot be stated as a general value. (Default value: 500)

A [WAL](#) segment requires 16 MB storage capacity on the hard disk. For this reason, select the value of the parameter *wal_keep_segments* according to the storage capacity available on the drive <<POSTGRES-DATA-DRIVE>>. Consider that the memory consumption of the database increases over the course of its service life.



Exemplary excerpt of an adjusted file:
[chapter "File postgresql.conf, primary server", p. 41](#)

5.3.1.1.5 Restart PostgreSQL service

Restart the following service:

- postgresql-x64-n.m - PostgreSQL Server n.m

Start type: *Automatic*

NOTICE! *n.m* stands for the version number of the PostgreSQL in use, e. g. *postgresql-x64-9.5 - PostgreSQL Server 9.5*.

5.3.1.1.6 Create database backup

Use the program *PGAdmin* to create a backup of the database *asc_rs*.

5.3.1.2 Create standby database

Proceed in the order of the following sections to set up the failover operation on the standby server.

5.3.1.2.1 Check network share of the standby database to the primary database

1. Open the Windows Explorer on the standby server.
2. Try to access the replication directory on the primary server with the following command \<<IP-ADDRESS-PRIMARY-DB>>\<<WAL-ARCHIVE>>.

Example:

\\192.168.171.110\WAL-ARCHIVE

\\FE80--29ED-F6FB-D58D-F92B.ipv6-literal.net\WAL-ARCHIVE

NOTICE! Contact a network administrator if accessing this directory from the standby server is not possible.

5.3.1.2.2 Install backup of the primary database on the standby database

1. On the standby server, delete the content of the directory <<POSTGRES-DATA-FOLDER>> so that the backup of the primary server can be deposited there.
2. Open a console window in the directory <<POSTGRES-INSTALL-FOLDER>>\bin.
3. Enter the destination path and the command to create the backup:

```
pg_basebackup -R -D <<POSTGRES-DATA-FOLDER>> --host=<<IP-ADDRESS-PRIMARY-DB>> --port=5432 -U postgres --checkpoint=fast --xlog-method=stream -R --verbose --progress
```

Example:

```
pg_basebackup -R -D D:\ASCDB --host=192.168.171.110 --port=5432 -U postgres --checkpoint=fast --xlog-method=stream -R --verbose --progress
```
4. Enter the database password for the user *postgres*.
 ⇒ The primary server creates a backup of the primary database and copies it to the directory <<POSTGRES-DATA-FOLDER>> on the standby server.

5.3.1.2.3 Adjust file "pg_hba.conf"

The client access is authenticated by means of the settings in the file *pg_hba.conf*. Select that all *neo* application servers must have access to the PostgreSQL instance.

1. On the standby server in the directory <<POSTGRES-DATA-FOLDER>>, open the configuration file *pg_hba.conf* with an editor.
2. Add the following entry for each connected *neo* application server at the end of the file content:

Type	Database	User	IP address	Method
host	asc_rs	postgres	<<IP-ADDRESS-NEO-CORE>>/<<CIDR>>	md5

Example:

```
host asc_rs postgres 192.168.171.50/32 md5
```

```
host asc_rs postgres FE80::29ED:F6FB:D58D:F92B/128 md5
```

NOTICE! The subnet masks /32 (IPv4) or /128 (IPv6) apply to the indicated IP address. If you would like to configure an address range for accessing the database, you have to adjust the value to the conditions in the customer's network.



Exemplary excerpt of an adjusted file:

[chapter "File pg_hba.conf, standby server", p. 40](#)



You have to use a tab character as delimiter between the values type, database, user, IP address, and method.



The value for <<CIDR>> must be provided by the system provider. The values of the subnet masks /32 (IPv4) or /128 (IPv6) apply to the indicated IP address only. If you would like to release an address range for accessing the database, you have to adjust this value.

5.3.1.2.4 Adjust "postgresql.conf" file

The file *postgresql.conf* is the central configuration file for PostgreSQL. Adjust the following settings to enable replication.

1. On the standby server in the directory <<POSTGRES-DATA-FOLDER>>, open the configuration file *postgresql.conf* with an editor.
2. Enter the following value in the file and make sure that the entry has not been commented out:
 - hot_standby = on



Exemplary excerpt of an adjusted file:

[chapter "File postgresql.conf, standby server", p. 41](#)

5.3.1.2.5 Adjust the file "recovery.conf"

1. On the standby server in the directory <<POSTGRES-DATA-FOLDER>>, open the configuration file *recovery.conf* with an editor.
2. Add the following entries:
 - restore_command = 'copy \\\<<IP-ADDRESS-PRIMARY-DB>>\<<WAL-ARCHIVE>>\%f %p'
 - trigger_file = '<<POSTGRES-DATA-DRIVE>>:\standby.txt'
 - archive_cleanup_command =
'"<<POSTGRES-INSTALL-FOLDER>>\bin\pg_archivecleanup" \\\<<IP-ADDRESS-PRIMARY-DB>>\<<WAL-ARCHIVE>>\%r'

Example for IPv4:

```
restore_command = 'copy \\192.168.171.110\WAL-ARCHIVE\%f %p'
trigger_file = 'D:\standby.txt'
archive_cleanup_command = '
"C:\Program Files\PostgreSQL\9.5\bin\pg_archivecleanup" \\
192.168.171.110\WAL-ARCHIVE\%r'
```

Example for IPv6:

```
restore_command = 'copy \\FE80::29ED-F6FB-D58D-F92B.ipv6-literal.net\WAL-
ARCHIVE\%f %p'
```

```
trigger_file = 'D:\\standby.txt'
archive_cleanup_command = '
"C:\\Program Files\\PostgreSQL\\9.5\\bin\\pg_archivecleanup" \\\\'FE80--29ED-
F6FB-D58D-F92B.ipv6-literal.net\\WAL-ARCHIVE\\ %r'
```

3. Check whether the correct values for password and host have been set for the parameter *primary_conninfo* in the already existing entry. The values must be identical to the values of the primary server.

host: <<IP-ADDRESS-PRIMARY-DB>>

password: database password of the user *postgres*



Exemplary excerpt of an adjusted file:

[chapter "File recovery.conf, standby server", p. 42](#)

5.3.1.2.6 Restart PostgreSQL service

1. Restart the following service:

- postgresql-x64-n.m - PostgreSQL Server n.m

Start type: *Automatic*

NOTICE! *n.m* stands for the version number of the PostgreSQL in use, e. g. *postgresql-x64-9.5 - PostgreSQL Server 9.5*.

5.3.1.3 Adjust setup.xml file

For failover operation, you have to enter the IP addresses of all *neo* application servers ([app servers](#)) on both database servers in the configuration file *setup.xml* in the AIP list:

1. In the directory <<NEO-INSTALL-FOLDER>>\\Updater\\config, open the file *setup.xml* with an editor.
2. Under **IPv4 IP addresses** search for the entry `<AIPList>127.0.0.1</AIPList>`.
3. In this entry, replace the default IP address *127.0.0.1* with the IP address of the *neo* application server.
If your system has several *neo* application servers, replace the default IP address *127.0.0.1* with the IP addresses of all *neo* application servers separated by a semicolon.
Example:
`<AIPList>192.168.171.50;192.168.171.60;192.168.171.70</AIPList>`
4. Under **IPv6 IP addresses** search for the entry `<AIPList>::1</AIPList>`.
5. In this entry, replace the default IP address `::1` with the IP address of the *neo* application server.
Example:
`<AIPList>FE80::29ED:F6FB:D58D:F92B; FE80::29ED:F6FB:D58D:F82B; FE80::29ED:F6FB:D58D:F72B</AIPList>`
6. Save the changes in the file.
7. Start the service ServiceMan again.



Exemplary excerpt of an adjusted file:

[chapter "File setup.xml, database server", p. 42](#)

5.3.2 Configuration of the application servers

5.3.2.1 Adjust setup.xml file

For failover operation of the databases, you have to add two settings manually on each *neo* application server (*app server*) in the configuration file *setup.xml*.

- `<databaseMode>FAILOVER</databaseMode>`

This setting ensures that the *app server* is operated in failover mode and that the Database Manager module is available to the system provider in the System Configuration.

- `<databaseFailoverDelay><<DELAY>></databaseFailoverDelay>`

This setting ensures that switching to the standby database does not take place before the primary database has been unavailable for a certain time. This avoids switching to the standby database when the network connection is interrupted only for a short period of time. The period of time after which the switchover takes place is entered instead of the placeholder `<<DELAY>>` in milliseconds.

Implement the following steps on all connected *app servers*:

1. In the directory `<<NEO-INSTALL-FOLDER>>\Updater\config`, open the file *setup.xml* with an editor.
2. Add the following two lines before the line `</Settings>`:
`<databaseMode>FAILOVER</databaseMode>`
`<databaseFailoverDelay>60000</databaseFailoverDelay>`

The value of the element `<databaseFailoverDelay>` indicated the period of time in milliseconds to pass by before failover to the standby server is initiated in case the connection to the primary server fails.



If the value is very small, even short interruptions of the network connection can cause failover operation.

If the value is very high, no database will be available for the indicated period of time if the primary server really fails. During this time, the web application of the recording system is not available.

3. Change the settings for the database *dbip* to the IP address of the primary server.
`<dbip><IP-ADDRESS-PRIMARY-DB></dbip>`
4. If the database *dbip* has not been installed on the *neo* application server, replace the following line
`<ascDatabase>...</ascDatabase>`
 with
`<ascDatabase>EXTERNAL</ascDatabase>`
5. Save the changes in the file.



Exemplary excerpt of an adjusted file:
[chapter "File setup.xml, application server", p. 42](#)

5.3.2.2 Configure database connection

All *neo* application servers must be granted access to the primary and the standby database. The database connection is administrated in a configuration file of each *neo* application server.

Implement the following steps on all connected *neo* application servers:

1. On the *neo* application server, open the configuration file *domain.xml* with an editor. You find the file in the following directory:
`<<NEO-INSTALL-FOLDER>>\glassfish4\glassfish\domains\enterprisecore\config\`

2. In this file, search the XML element *jdbc-connection-pool* with the attribute *name="ASCSuitePoolPrimary"*.

The are two equivalent layouts of the configuration of the database connection:

Variant 1 IPv4:

IP address, port, and database name have been entered together in the attribute *value* of the element *property name="URL"*.

Example:

```
<property name="URL" value="jdbc:postgresql://192.168.171.110:5432/asc_rs"></property>
```

Variant 1 IPv6:

IP address, port, and database name have been entered together in the attribute *value* of the element *property name="URL"*.

Example:

```
<property name="URL" value="jdbc:postgresql://[FE80::29ED:F6FB:D58D:F92B]:5432/asc_rs"></property>
```

Variant 2 IPv4:

IP address, ports and database name have been entered in separate *property* elements.

Example:

```
<property name="ServerName" value="postgresql://192.168.171.110"></property>
<property name="databasename" value="asc_rs"></property>
<property name="portname" value="5432"></property>
```

Variant 2 IPv6:

IP address, ports and database name have been entered in separate *property* elements.

Example:

```
<property name="ServerName" value="postgresql://FE80::29ED:F6FB:D58D:F92B"></property>
<property name="databasename" value="asc_rs"></property>
<property name="portname" value="5432"></property>
```

If *domain.xml* variant 2 is used in your file, remove the *property* elements with the names *ServerName*, *databasename*, and *portname* and add the element

```
<property name="URL" value="jdbc:postgresql://localhost:5432/asc_rs"></property>
```

in their place so that the layout equals the layout of variant 1.

3. In the sub-element *property name="URL"* in the attribute *value* enter the IP address, the port, and the name of the primary database.

Example IPv4:

```
<property name="URL" value="jdbc:postgresql://<<IP-ADDRESS-PRIMARY-DB>>:5432/asc_rs"></property>
```

Example:

```
<jdbc-connection-pool max-pool-size="60" fail-all-connections="true"
validate-atmost-once-period-in-seconds="60"
datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
name="ASCSuitePool" validation-table-name="asc_rs.db_connection_control"
is-connection-validation-required="true"
res-type="javax.sql.ConnectionPoolDataSource">
<property name="URL" value="jdbc:postgresql://192.168.171.110:5432/asc_rs">
</property>
<property name="user" value="postgres"></property>
<property name="password" value="{ALIAS=db_password_alias}">
</property>
</jdbc-connection-pool>
```

NOTICE! Observe the following formatting:

```
<jdbc-connection-pool max-pool-size="60" fail-all-connections="true" validate-atmost-once-period-in-seconds="60"
  <property name="URL" value="jdbc:postgresql://192.168.171.110:5432/asc_rs"></property>
  <property name="user" value="postgres"></property>
  <property name="password" value="{ALIAS=db_password_alias}"></property>
</jdbc-connection-pool>
```

Fig. 21: Formatting principle

NOTICE! If entering a port is required when using IPv6 IP addresses, you must enter the IP address in square brackets to differentiate it from the port.

Example IPv6:

```
<property name="URL" value="jdbc:postgresql://<<IP-ADDRESS-PRIMARY-DB>>:5432/asc_rs"></property>
```

Example:

```
<jdbc-connection-pool max-pool-size="60" fail-all-connections="true"
validate-atmost-once-period-in-seconds="60"
datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
name="ASCSuitePool" validation-table-name="asc_rs.db_connection_control"
is-connection-validation-required="true"
res-type="javax.sql.ConnectionPoolDataSource">
<property name="URL" value="jdbc:postgresql://[FE80::29ED:F6FB:D58D:F92B]:5432/asc_rs">
</property>
<property name="user" value="postgres"></property>
<property name="password" value="{ALIAS=db_password_alias}">
</property>
</jdbc-connection-pool>
```

4. In the same file, search the XML element *jdbc-connection-pool* with the attribute *name="ASCSuitePoolStandby"*

For this entry, there are the two layout variant mentioned above, too.



If *domain.xml* variant 2 is used in your file, remove the *property* elements with the names *ServerName*, *databasename*, and *portname* here, too, and add the entry in the layout variant 1 again.

5. Adjust the entry as described in the previous step while using the IP address of the standby server.

Example IPv4:

```
<property name="URL" value="jdbc:postgresql://<<IP-ADDRESS-STANDBY-DB>>:5432/asc_rs"></property>
```

Example:

```
<jdbc-connection-pool max-pool-size="60" fail-all-connections="true"
validate-atmost-once-period-in-seconds="60" steady-pool-size="0"
datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
validation-table-name="asc_rs.db_connection_control"
name="ASCSuitePoolPrimary" is-connection-validation-required="true"
res-type="javax.sql.ConnectionPoolDataSource">
<property name="URL" value="jdbc:postgresql://192.168.171.120:5432/asc_rs">
</property>
<property name="user" value="postgres"></property>
<property name="password" value="{ALIAS=db_password_alias}"></property>
</jdbc-connection-pool>
```

Example IPv6:

```
<property name="URL" value="jdbc:postgresql://<<IP-ADDRESS-STANDBY-DB>>:5432/asc_rs"></property>
```

Example:

```
<jdbc-connection-pool max-pool-size="60" fail-all-connections="true"
validate-atmost-once-period-in-seconds="60" steady-pool-size="0"
datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
validation-table-name="asc_rs.db_connection_control"
name="ASCSuitePoolPrimary" is-connection-validation-required="true"
res-type="javax.sql.ConnectionPoolDataSource">
<property name="URL" value="jdbc:postgresql://[FE80::29ED:F6FB:D58D:F92B]:5432/asc_rs">
</property>
<property name="user" value="postgres"></property>
<property name="password" value="{ALIAS=db_password_alias}"></property>
</jdbc-connection-pool>
```

6. Search the following entry in the file:

```
<resource-ref ref="jms/ConnectionFactory"></resource-ref>
```

7. Add the following line:

```
<resource-ref ref="jdbc/ASCSuitePoolQuartz"></resource-ref>
```

Example:

```
<resource-ref ref="jms/ConnectionFactory"></resource-ref>
<resource-ref ref="jdbc/ASCSuitePoolQuartz"></resource-ref>
```

8. Search the following entry in the file:

```
<jdbc-resource pool-name="ASCSuitePoolStandby"></jdbc-resource>
```

9. Add the following line:

```
<jdbc-resource pool-name="ASCSuitePoolQuartz" jndi-name="jdbc/
ASCSuitePoolQuartz"></jdbc-resource>
```

Example:

```
<jdbc-resource pool-name="ASCSuitePoolStandby"
description="Jdbc Resource for managed connection pool"
jndi-name="jdbc/ASCSuiteStandby"></jdbc-resource>
<jdbc-resource pool-name="ASCSuitePoolQuartz"
jndi-name="jdbc/ASCSuitePoolQuartz"></jdbc-resource>
```

10. Below the XML element `jdbc-connection-pool` with the attribute `name="ASCSuitePoolStandby"`, add the following entry and enter the IP addresses of the primary and the standby database.

```
<jdbc-connection-pool max-pool-size="60" fail-all-connections="true"
validate-atmost-once-period-in-seconds="60"
datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
validation-table-name="asc_rs.db_connection_control"
name="ASCSuitePoolQuartz" is-connection-validation-required="true"
res-type="javax.sql.ConnectionPoolDataSource">
<property name="URL" value="jdbc:postgresql://<<IP-ADDRESS-PRIMARY-DB>>:5432,<<IP-ADDRESS-STANDBY-DB>>:5432/asc_rs"></property>
<property name="user" value="postgres"></property>
<property name="password" value="{ALIAS=db_password_alias}"></property>
</jdbc-connection-pool>
```

Example IPv4:

```
<jdbc-connection-pool max-pool-size="60" fail-all-connections="true"
validate-atmost-once-period-in-seconds="60"
```

```
datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
validation-table-name="asc_rs.db_connection_control"
name="ASCSuitePoolQuartz" is-connection-validation-required="true"
res-type="javax.sql.ConnectionPoolDataSource">
<property name="URL" value="jdbc:postgresql://192.168.171.110:5432,192.168.171.120:5432/
asc_rs"></property>
<property name="user" value="postgres"></property>
<property name="password" value="${ALIAS=db_password_alias}"></property>
</jdbc-connection-pool>
</resources>
```

Example IPv6:

```
<jdbc-connection-pool max-pool-size="60" fail-all-connections="true"
validate-atmost-once-period-in-seconds="60"
datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
validation-table-name="asc_rs.db_connection_control"
name="ASCSuitePoolQuartz" is-connection-validation-required="true"
res-type="javax.sql.ConnectionPoolDataSource">
<property name="URL" value="jdbc:postgresql://[FE80::29ED:F6FB:D58D:F92B]:5432,
[FE80::29ED:F6FB:D58D:F82B]:5432/asc_rs"></property>
<property name="user" value="postgres"></property>
<property name="password" value="${ALIAS=db_password_alias}"></property>
</jdbc-connection-pool>
</resources>
```



Exemplary excerpt of an adjusted file,
see [chapter "File domain.xml, application server", p. 43](#)

11. Stop the service *ServiceMan*.
12. Restart the service *ApplicationServer* or *enterprisecore* and set the startup type of the service to *Automatic*.

ATTENTION!

Possible loss of data

Both databases must be available when starting the *neo* application server. If one of the databases is not available, the start process is canceled and the *neo* application server will not be available. In this case, no recording takes place.

In case, one of the databases is not available for a longer period of time, enter the [IP](#) address of the available database as the [IP](#) address for the two JDBC-Connection pools.

Once the failed database is available again, you can change the [IP](#) address again. Make sure that the replication is operative, see [chapter "Function test", p. 33](#).

If you have to restart the replication manually, proceed as described in [chapter "Reset failover operation", p. 36](#).

13. Wait until the service *ApplicationServer* has been started completely.
Check the functionality of the service by calling up any of the *neo* web applications. If you can start the *neo* web application, this indicates that the service *ApplicationServer* runs.
14. Restart the service *ServiceMan* and set the startup type of the service to *Automatic*.

5.4 Function test

5.4.1 Check replication on the primary server

1. Open the program *PGAdmin* on the primary server and connect with the database *asc_rs*.
2. Execute the following command:

```
select * from pg_stat_replication;
```

 - ⇒ Expected results:
 - Exactly one line is delivered back as result.
 - The value of the attribute *client_addr* is identical to the IP address of the standby server.
 - The value of the attribute *state* is *streaming*.

5.4.2 Check replication on the standby server

1. Open the program *PGAdmin* on the standby server and connect with the database *asc_rs*.
2. Execute the following command:

```
select pg_is_in_recovery(), pg_last_xlog_receive_location(),  
pg_last_xlog_replay_location();
```



- *pg_is_in_recovery* indicates whether the restoration process is running.
- *pg_last_xlog_receive_location* indicates the last [WAL](#) record that the standby server has received from the primary server.
- *pg_last_xlog_replay_location* indicates the last [WAL](#) record which was processed on the standby server.

- ⇒ Expected results:
 - The value of the attribute *pg_is_in_recovery* is *t* (for *true*).
 - The values of the attributes *pg_last_xlog_receive_location* and *pg_last_xlog_replay_location* are displayed hexadecimally and have identical or at least similar numerical values.

5.4.3 Check network share

1. On the standby server, open the file Explorer.
2. Try to access the replication directory on the primary server (`\\<<IP-ADDRESS-PRIMARY-DB>>\\<<WAL-ARCHIVE>>`).
3. Contact a network administrator if accessing this directory from the standby server is not possible.

5.4.4 Check deletion of the WAL archives

Check whether the [WAL](#) archives are deleted on the primary server.

- ✓ The *neo* application servers must be connected to the databases, see [chapter "Configure database connection", p. 28](#).
1. Check in running operation whether older files in the replication directory `\\<<IP-ADDRESS-PRIMARY-DB>>\\<<WAL-ARCHIVE>>` are deleted.
 2. If the number of [WAL](#) archives in the replication directory increases continually, it may be the case that the user on whose account the Postgres service "postgresql-x64-n.m - PostgreSQL Server n.m" runs does not have the right to delete files in the replication directory. In this case, run the Postgres service on the account of the local user *postgres* or on the account of a domain user who has the respective rights.
NOTICE! If PostgreSQL is installed with the setup of the *neo* Suite, then the service runs on a local system account.

6 Failover operation functional description

The replication supports the consistency and the availability of the data. A [partition tolerance](#) is not given, though.

The failover function can be used to cover the following failure scenarios:

- *Failure of the primary database*

Systems with several enterprise cores permanently check whether the primary database is available from all Enterprise Cores. Only when none of the Enterprise Cores accesses the database anymore, the failover function is triggered and switching over to the standby database is initiated. See [chapter "Failure of the primary database", p. 34](#)

- *Failure of the standby database*

If the primary database fails and the standby database is not available, the database cannot be switched. See [chapter "Failure of the standby database", p. 35](#)

- *Failure of an Enterprise Core*

If an Enterprise Core fails, you must trigger the failover function manually. See [Failure of an Enterprise Core](#)

6.1 Failure of the primary database

When the primary database fails, a switch to the standby database takes place automatically.

ATTENTION!

Possible loss of data

Since the replication is interrupted, loss of data can occur. The extent of the loss of data cannot be quantified in general but has to be determined individually, if required.

If the primary database is no longer available, the following processes are started by one of the configured *neo* application servers.

1. The service ServiceMan of the standby server writes the file *standby.txt* into the root directory of the drive <<POSTGRES-DATA-FOLDER>>.
2. This file *standby.txt* is read by the standby database and deleted subsequently.
3. The replication is cancelled.
4. The standby database is restarted automatically with read/write access. It assumes the role of the productive database.
5. If a failure of the primary database is registered, the file *DBSlaveTriggerFile.txt* is created on each *neo* application server. You find the file in the following directory:
<<NEO-INSTALL-FOLDER>>\glassfish4\glassfish\domains\enterprisecore\config\



The file *DBSlaveTriggerFile.txt* makes that the primary database is not used.

If the standby database is available and a failover takes place, the data of the following conversations are saved in the standby database.

Once the error has been remedied, the failover operation must be reset manually. Resetting can be done manually or by means of the Configuration Tool.

- Reset failover operation with the Failover Configuration Tool, see [chapter "Reset failover operation with Failover Configuration Tool", p. 38](#).
- Reset failover operation manually, see [chapter "Reset failover operation manually", p. 36](#).

6.2 Failure of the standby database

A failure of the standby database does not affect the operation of the *neo* Suite as long as the primary database is available during the downtime.

If the standby database fails, the replication is interrupted, though. Depending on the duration of the downtime, the replication can be started automatically again as soon as the standby database is available again. If the replication is not started automatically, you can start the replication manually.

1. Once the standby database is available again, implement the following steps:
 - Check whether the replication works, see [chapter "Function test", p. 33](#).
 - If the replication is **not** executed, you have to restart the replication manually, see [chapter "Restore standby server", p. 37](#).

Reset failover operation

Once the primary database is available again, there are 2 possibilities for further usage.

Depending on the layout of the system architecture, select the corresponding configuration:

1. You can switch back manually from the standby database to the primary database.
See [chapter "Reset failover operation manually"](#), p. 36.
2. You can use the standby database primarily and reverse the replication direction.
See [chapter "Reset failover operation with Failover Configuration Tool"](#), p. 38.

Reset failover operation manually

To restore the original state after a failover, you have to make manual changes in the configuration on the two database servers as well as on the [neo](#) application servers. To do so, proceed in the order of the sections describes below.

ATTENTION!

Possible loss of data

While restoring the original state, the existing data must not be changed, since the two databases will not be available for a short period of time.

To avoid loss of data, a [maintenance window](#) is required during which the databases can be reset from failover operation to their original state.

Stop services on the application servers

Implement the following steps on all [neo](#) application servers:

1. Stop the service ServiceMan and set the startup type for this service to *Deactivated*.
2. Stop the service ApplicationServer and set the startup type for this service to *Deactivated*.
3. In the directory <<NEO-INSTALL-FOLDER>>\glassfish4\glassfish\domains\enterprisecore\config\, delete the file *DBSlaveTriggerFile.txt*, if it exists.
Exemplary directory path:
C:\Program Files (x86)\ASC\ASC Product Suite\glassfish4\glassfish\domains\enterprisecore\config



The file *DBSlaveTriggerFile.txt* does not exist on all [neo](#) application servers.

Create backup of the standby database

1. Use the program *PGAdmin* to create a backup of the database *asc_rs* on the standby server.
2. On the standby server, stop the following services:
 - postgresql-x64-n.m - PostgreSQL Server n.m
Start type: *Automatic*
n.m stands for the version number of the PostgreSQL in use, e. g. *postgresql-x64-9.5 - PostgreSQL Server 9.5*.
 - PostgreSQL Scheduling Agent - pgAgent
Start type: *Automatic*

Restore primary server

1. Depending on the reason why the primary database has failed, it may be required to reinstall the database management system PostgreSQL, see [chapter "Primary database"](#), p. 8.

2. Regardless of whether the database management system has been reinstalled or continues in operation, you definitely have to install the backup of the standby server on the primary server (). To do so, use the program *pgAdmin*.
3. If you had to reinstall the database, implement the changes in the configuration described in [chapter "Set up primary database", p. 21](#).

Restore standby server

1. Restore the replication on the standby server as describes in [chapter "Restore standby server", p. 37](#).

Start services on the application servers

Implement the following steps on all *neo* application servers:

1. Change the startup type of the service ApplicationServer to *Manual*.
 2. Change the startup type of the service ServiceMan to *Automatic*.
 3. Start the service ServiceMan if it has not been started automatically.
- ⇒ The service ServiceMan automatically starts the service ApplicationServer. That is why the service ApplicationServer does not have to be started manually.

Function test

1. Check whether the replication works, see [chapter "Function test", p. 33](#).

7.1.1

Restore standby server

You can check the status of the replication by means of the procedures described under [chapter "Function test", p. 33](#).

In case the replication has been interrupted, e. g. due to a longer downtime of the standby database or a longer interruption of the network connection between primary and standby server, then the following steps are required to restart the replication.



The changes described in the following have to be implemented exclusively on the standby server.

- ✓ The primary database is used as productive database by all *neo* application servers again.
1. Create the directory `<<PGRES-DATA-DRIVE>>\pg_conf_backup`.
 2. Copy the following files into the newly created directory:
 - `<<PGRES-DATA-FOLDER>>\pg_hba.conf`
 - `<<PGRES-DATA-FOLDER>>\postgresql.conf`
 - `<<PGRES-DATA-FOLDER>>\recovery.conf`
 - `<<PGRES-DATA-FOLDER>>\DataBase.conf`
 3. Delete the content of the directory `<<PGRES-DATA-FOLDER>>`.
 4. Open a console window in the directory `<<PGRES-INSTALL-FOLDER>>\bin`.
 5. Enter the destination path and the command to create the backup:


```
pg_basebackup -R -D <<PGRES-DATA-FOLDER>> --host=<<IP-ADDRESS-PRIMARY-DB>> --port=5432 -U postgres --checkpoint=fast --xlog-method=stream -R --verbose --progress
```

Example: `pg_basebackup -R -D D:\ASCDB --host=192.168.171.110 --port=5432 -U postgres --checkpoint=fast --xlog-method=stream -R --verbose --progress`
 6. Enter the database password for the user *postgres*.
 - ⇒ The primary server creates a backup of the primary database and copies it to the directory `<<PGRES-DATA-FOLDER>>` on the standby server.

7. Copy the files from step 2 into the directories <<POSTGRES-DATA-FOLDER>> and overwrite the existing files.
8. Delete the file *standby.txt* in the directory <<POSTGRES-DATA-DRIVE>>, if it exists.
9. Restart the service *postgresql-x64-n.m- PostgreSQL Server n.m.*
NOTICE! *n.m* stands for the version number of the PostgreSQL in use, e. g. *postgresql-x64-9.5 - PostgreSQL Server 9.5.*
10. Check whether the replication works, see [chapter "Function test", p. 33.](#)

7.2 Reset failover operation with Failover Configuration Tool

If you do not want to reset the failover database to become the standby database again but instead want to continue using the failover database as primary database, you can reverse the replication direction.

Upon doing so, the database which had been used as standby database before failover operation will then be used as the active primary database after the failover operation.

7.2.1 Use failover database as primary database

To reverse the replication direction, the Failover Configuration Tool offers a reverse functionality.

1. Open the Windows Explorer and change to the directory *C:\Program Files (x86)\ASC\ASC Product Suite\scripts.*
2. Open the file *DBFailoverConfig.exe* with a double-click.
 ⇒ The Failover Configuration Tool appears.

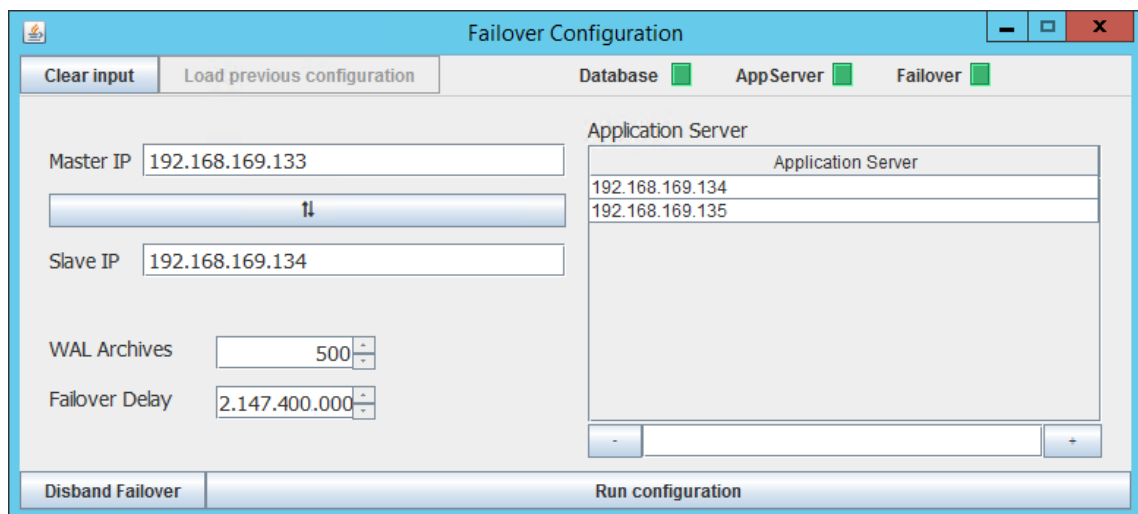
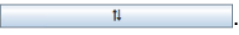


Fig. 22: Failover Configuration Tool - reverse configuration

3. Click on the tab *Load previous configuration* to load the previous configuration.
4. Click on the button .
 ⇒ This switches the entries of *Master-IP* and *Slave-IP*.
5. Click on the button *Run configuration* to initiate a reversal of the configuration.
6. In the progress window, you can see the status of the reversal.
 ⇒ Once the reversal is complete, a success message appears.

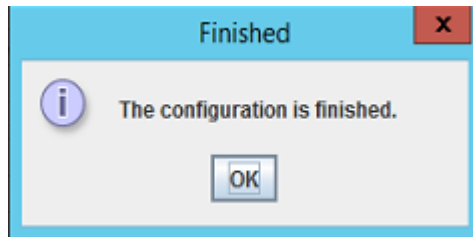


Fig. 23: Success message

7. Check whether the replication works, see [chapter "Function test"](#), p. 33.

NOTICE! The directory <<POSTGRES-DATA-DRIVE>>\Replication in which the WAL archives are located is not deleted during the reversal. Since the directory has been shared in the network, it must be deleted manually.

In the following chapters you find exemplary extracts of files which have to be adjusted for the configuration of the failover operation.

In these extracts, the entries which definitely have to be added for the configuration of the failover operation are highlighted in **bold**.

Depending on the structure of the overall system, additional entries may exist which have been marked by means of the placeholder "...".

For the different servers, the following IP addresses are used in the examples:

- neo application servers: *192.168.171.50*, *192.168.171.60*, *192.168.171.70*
- Primary server: *192.168.171.110*
- Standby server: *192.168.171.120*



Note that in the following line breaks have been inserted for formatting reasons. Do not include the line breaks when entering the commands.

8.1 File pg_hba.conf, primary server

```
...
# TYPE      DATABASE      USER      ADDRESS      METHOD
# IPv4 local connections:
host        all          all        127.0.0.1/32    md5
host        asc_rs        postgres   192.168.171.50/32    md5
host        postgres      postgres   192.168.171.50/32    md5
# IPv6 local connections:
host        all          all        ::1/128        md5
# Allow replication connections from localhost, by a user with the replication privilege.
host        replication    postgres   192.168.171.120/32    md5
...
```

8.2 File pg_hba.conf, standby server

```
...
# TYPE      DATABASE      USER      ADDRESS      METHOD
# IPv4 local connections:
host        all          all        127.0.0.1/32    md5
host        asc_rs        postgres   192.168.171.50/32    md5
# IPv6 local connections:
...
```

8.3 File postgresql.conf, primary server

```
...
#- Sending Server(s) -
#Set these on the master and on any standby that will send replication data.
max_wal_senders = 10          #max number of walsender processes
                              #change requires restart
wal_keep_segments = 500      #in logfile segments, 16MB each; 0 disables

...

#-----
#WRITE AHEAD LOG
#-----
#- Settings -
wal_level = hot_standby      #minimal, archive, hot_standby, or logical
                              #change requires restart
#fsync = on                  #turns forced synchronization on or off

...

#- Archiving -
archive_mode = on            #allows archiving to be done
                              #change requires restart
archive_command = 'copy %p \\\192.168.171.110\WAL-ARCHIVE\\%f'
                              #command to use to archive a logfile segment
                              #placeholders: %p = path of file to archive
                              #%f = file name only
                              #e.g. 'test ! -f /mnt/server/archivedir/%f && cp %p /mnt/server/archivedir/%f'
#archive_timeout = 0         #force a logfile segment switch after this
                              #number of seconds; 0 disables

...
```

8.4 File postgresql.conf, standby server

```
...
#- Standby Server(s) -
#These settings are ignored on a master server.
hot_standby = on             #"on" allows queries during recovery
```

```

                                #change requires restart
#max_standby_archive_delay = 30s    #max delay before canceling queries
                                #when reading WAL from archive
                                #-1 allows indefinite delay

...

```

8.5 File recovery.conf, standby server

```

...
standby_mode = 'on'
primary_conninfo =
'user=postgres password=DatabaseAdmin123# host=192.168.171.110 port=5432 sslmode=prefer sslcompression=1 krbsrv-
name=postgres'
restore_command = 'copy \\\192.168.171.110\\WAL-ARCHIVE\\%f %p'
trigger_file = 'D:\\standby.txt'
archive_cleanup_command = '"C:\\Program Files\\PostgreSQL\\9.4\\bin\\pg_archivecleanup" \\\192.168.171.110\\WAL-AR-
CHIVE\\ %r'
...

```

8.6 File setup.xml, database server

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?><Setup>
  <Settings>
    ...
    <AIPList>192.168.171.50;192.168.171.60;192.168.171.70</AIPList>
    ...
  </Settings>
</Setup>

```

8.7 File setup.xml, application server

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?><Setup>
  <Settings>
    ...
    <dbip>192.168.171.110</dbip>
    ...
    <databaseMode>FAILOVER</databaseMode>
    <databaseFailoverDelay>60000</databaseFailoverDelay>
  </Settings>

```

</Setup>

8.8 File domain.xml, application server



For easier reading, line breaks have been added in the following example. Observe the layout of the existing file *domain.xml* if you would like to follow the examples displayed here.

```
...
<resources>
...
    <jdbc-connection-pool max-pool-size="60" fail-all-connections="true"
        validate-atmost-once-period-in-seconds="60"
        datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
        validation-table-name="asc_rs.db_connection_control"
        name="ASCSuitePoolPrimary"
        is-connection-validation-required="true" res-type="javax.sql.ConnectionPoolDataSource">
        <property name="URL" value="jdbc:postgresql://192.168.171.110:5432/asc_rs"></property>
        <property name="user" value="postgres"></property>
        <property name="password" value="{ALIAS=db_password_alias}"></property>
    </jdbc-connection-pool>
    <jdbc-connection-pool fail-all-connections="true"
        validate-atmost-once-period-in-seconds="60"
        datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
        validation-table-name="asc_rs.db_connection_control"
        name="ASCSuitePoolStandby"
        is-connection-validation-required="true" res-type="javax.sql.ConnectionPoolDataSource">
        <property name="URL" value="jdbc:postgresql://192.168.171.120:5432/asc_rs"></property>
        <property name="user" value="postgres"></property>
        <property name="password" value="{ALIAS=db_password_alias}"></property>
    </jdbc-connection-pool>
    <jdbc-connection-pool max-pool-size="60" fail-all-connections="true"
        validate-atmost-once-period-in-seconds="60"
        datasource-classname="org.postgresql.ds.PGConnectionPoolDataSource"
        validation-table-name="asc_rs.db_connection_control"
        name="ASCSuitePoolQuartz"
        is-connection-validation-required="true" res-type="javax.sql.ConnectionPoolDataSource">
        <property name="URL"
```

```

        value="jdbc:postgresql://192.168.171.110:5432,192.168.171.120:5432/asc_rs"></property>
        <property name="user" value="postgres"></property>
        <property name="password" value="${ALIAS=db_password_alias}"></property>
    </jdbc-connection-pool>
    <jdbc-resource pool-name="ASCSuitePoolPrimary"
jndi-name="jdbc/ASCSuitePrimary"></jdbc-resource>
    <jdbc-resource pool-name="ASCSuitePoolStandby"
jndi-name="jdbc/ASCSuiteStandby"></jdbc-resource>
    <jdbc-resource pool-name="ASCSuitePoolQuartz"
jndi-name="jdbc/ASCSuitePoolQuartz"></jdbc-resource>
</resources>
<servers>
    <server config-ref="server-config" name="server">
        ...
        <resource-ref ref="jdbc/ASCSuitePrimary"></resource-ref>
        <resource-ref ref="jdbc/ASCSuiteStandby"></resource-ref>
        <resource-ref ref="jdbc/ASCSuitePoolQuartz"></resource-ref>
    </server>
</servers>
...

```

List of figures

Fig. 1	Select features for the installation	9
Fig. 2	Select target drive for the internal database	9
Fig. 3	Finish installation and restart server	10
Fig. 4	Select features for the installation	11
Fig. 5	Select target drive for the internal database	12
Fig. 6	Finish installation and restart server	13
Fig. 7	Share drive	14
Fig. 8	Share for user postgres	15
Fig. 9	Grant user postgres full control	15
Fig. 10	Configure login for the Postgres service	16
Fig. 11	Failover Configuration Tool - Configure failover	17
Fig. 12	Configure failover components	17
Fig. 13	Watch the progress of the configuration	18
Fig. 14	Success message	19
Fig. 15	Status indicator of failover components:	19
Fig. 16	Color legend	20
Fig. 17	Call up Failover Configuration Tool again	20
Fig. 18	Share drive	22
Fig. 19	Share for user postgres	22
Fig. 20	Grant user postgres full control	23
Fig. 21	Formatting principle	30
Fig. 22	Failover Configuration Tool - reverse configuration	38
Fig. 23	Success message	39

List of tables

Glossary

App server

Application server or web server. In the system architectures: the server on which the Enterprise Core and the GlassFish software have been installed.

CIDR

Classless Inter-Domain Routing (CIDR) is a method to help slow the rapid exhaustion of Ipv4 addresses. CIDR replaces the firm allocation of an IPv4 address to a network class which used to indicate the prefix length. The prefix length can be selected freely in CIDR and thus has to be indicated when stating an IP subnet. Usually a netmask is used to do so. (Source: Wikipedia 3rd April 2017)

FQDN

Fully Qualified Domain Name

IP

Internet Protocol, basic protocol for Internet communication

Maintenance window

Period of time (time window) during which the maintenance can take place. During this period, the components under maintenance are usually not operated or operation is only possible to a limited extent.

Multi-core system

Recording system in which several application servers (Enterprise Core) are used.

Partition tolerance

Term from the CAP theorem with regards to distributed systems. When it comes to the failover concept of the database, the term signifies that the data pool in both databases (primary and standby) is supposed to remain available AND consistent when the two database servers are separated physically.

Single-core system

Recording system in which only 1 application server (Enterprise Core) is used. It is possible though, that other components (e. g. recording components) have been installed on other servers. Such a system would be a multi-server system/single-core system.

WAL

Write-ahead logging (WAL) is a family of techniques for providing atomicity and durability (two of the ACID properties) in database systems. In a system using WAL, all modifications are written to a log before they are applied. (Source: Wikipedia 22nd February 2017)