DH2i DxEnterprise 21.0 Software: SQL Server Availability Groups for Docker Quick Start Guide

DH2i Company

Support: +1 (800) 380-5405 ext. 2
https://dh2i.com/support/
eFax: +1 970-295-4505
Email: support@dh2i.com
https://www.dh2i.com

SQL Server Availability Groups for Docker

This quick start guide describes how to create and deploy a DxEnterprise + SQL Server container cluster in a Docker environment for Availability Groups. Using this guide, the user will create a DxEnterprise + SQL Server container image and use DxEnterprise to create and configure the Availability Group.

***NOTE:** This guide covers the deployment of three AG replicas. When deploying availability groups, please consider the Microsoft SQL Server quorum requirements for automatic failover described in this KB: Quorum Considerations for SQL Server Availability Groups.*

Prerequisites

- Docker installed on a Linux machine (physical or virtual) with at least 8GB of RAM. For more information about installing Docker, view Docker documentation.
- A valid DxEnterprise license with availability group management features and tunnels enabled. A fully featured developer edition is available for free for non-production use at https://dh2i.com/dxenterprise-dxodyssey-developer-edition. To purchase DxEnterprise software for production workloads, visit https://dh2i.com/store/.

Create the SQL Server + DxEnterprise Container Image

1. Download the dxemssql Dockerfile from DH2i's GitHub to the current working directory.
    - `curl https://raw.githubusercontent.com/dh2i/dxemssql/main/dxemssql.dockerfile -o dxemssql.dockerfile`
2. Build the image using the Dockerfile. The DxEnterprise + SQL Server container image layers DxEnterprise software on top of the base SQL Server image. The Dockerfile contains commands to install .NET Core 3.1, add the DxEnterprise tarball from DH2i's website, and add some additional permissions. Docker will accept many variations for tagging. This guide uses the pattern dh2i/dxemssql, but any tag (e.g., test:1, dh2i:test, mycorp/myrepo:mytag and others) may be used.
    - `docker build -t <tag> -f dxemssql.dockerfile ./`

Create the Containers

1. Create the container using the `docker run` command:
    - `docker run -d --name dxemssql-0 -e ACCEPT_EULA=Y -e SA_PASSWORD=P@ssW0rd -e MSSQL_ENABLE_HADR=1 -h dxemssql-0 <tag>`
2. Repeat the command for the additional containers, replacing the name:
    - `docker run -d --name dxemssql-1 -e ACCEPT_EULA=Y -e SA_PASSWORD=P@ssW0rd -e MSSQL_ENABLE_HADR=1 -h dxemssql-1 <tag>`
    - `docker run -d --name dxemssql-2 -e ACCEPT_EULA=Y -e SA_PASSWORD=P@ssW0rd -e MSSQL_ENABLE_HADR=1 -h dxemssql-2 <tag>`

Configure the First Container

All DxCLI commands can be run from outside the container using the command format `docker exec <container_name> dxcli <command>`. This format will be used for all of the subsequent DxCLI command examples.

1. Activate the DxEnterprise license using the command `dxcli activate-sever`.

   NOTE: *If this command fails, retry a couple more times. Depending on system resources, it may take 30 or more seconds for DxEnterprise and SQL Server to complete their startup.*

   Syntax

   ```
   dxcli activate-server <license_key> [node]
   ```

   Parameters

Name	Description	Required
license_key	The license key.	True
node	The name of the node.	False

   Example

   ```
   docker exec dxemssql-0 dxcli activate-server AAAA-BBBB-CCCC-DDDD
   ```

2. Add a Vhost to the cluster using the command `dxcli cluster-add-vhost`.

   Syntax

   ```
   dxcli cluster-add-vhost <vhost> <vips> <nodes | VHOST:vhosts>
   [autofailback] [1-5] [ilb_ports]
   ```

   Parameters

Name	Description	Required
vhost	The name of the Vhost.	True
vips	The virtual IP(s) for the Vhost (comma separated list for multiples). The use of a loopback address (127.0.0.1) is supported, but must be preceded by an asterisk (*).	True
nodes | VHOST:vhosts	The node(s) or Vhost(s) to add to the Vhost (comma separated list for multiples).	True
autofailback	Set autofailback or leave blank if autofailback is not desired.	False
priority	The priority order of failover between Vhosts (1 is the highest and 5 is the lowest)	False
ilbports	The ports to use for internal load balancer probing (comma separated list for multiples).	False

   Example

   ```
   docker exec dxemssql-0 dxcli cluster-add-vhost vhost1 *127.0.0.1
   dxemssql-0
   ```

3. Encrypt the SQL Server sysadmin password for DxEnterprise using the command `dxcli encrypt-text`. The encrypted password will be used to create the availability group in the next step.

   Syntax

   ```
   dxcli encrypt-text <value>
   ```

   Parameters

Name	Description	Required
value	The text to be encrypted.	True

   Example

   ```
   docker exec dxemssql-0 dxcli encrypt-text p@ssw0rd
   ```

4. Add an availability group to the Vhost using the command `dxcli add-ags`. The SQL Server sysadmin password must be encrypted using the command `dxcli encrypt-text` from the previous step.

   Syntax

   ```
   dxcli add-ags <vhost> <ags_name>
   <node_name|instance_name|sql_login|sql_pass|mirror_port|availability
   _mode|[tunnel_port]>
   ```

   Parameters

Name	Description	Required
vhost	The name of the Vhost.	True
ags_name	The name of the availability group.	True
node_name	The name of the node.	True
instance_name	The name of the instance.	True
sql_login	The user name used to login to SQL Server. The SQL Server login can be any SQL login or Windows domain account that has been assigned the sysadmin role for the instance being managed. When supplying a domain account, the username needs to follow the UPN format (for example: user@domain.com).	True
sql_pass	The password used to login to SQL Server (encrypted using `dxcli encrypt-text`).	True
mirror_port	The mirroring port to use for the availability group (default is 5022).	True
availability_mode	synchronous_commit, asynchronous_commit, or configuration_only.	True
tunnel_port	The port to be used for the tunnel connection. This port should be unique per node.	False

   Example

```
docker exec dxemssql-0 dxcli add-ags vhost1 ags1 "dxemssql-
0|mssqlserver|sa|V1DUmkSp6vNmyLDbzT1+xg==|5022|synchronous_commit|50
221"
```

5. Set a One-Time PassKey (OTPK) using the command `dxcli set-otpk`. The output from this command will be used to join the other nodes to the DxEnterprise cluster.

   Syntax

   ```
   dxcli set-otpk [ttl] [otpk]
   ```

   Parameters

Name	Description	Required
ttl	The time to live.	False
otpk	The one-time passkey in base64	False

   Example

   ```
   docker exec dxemssql-0 dxcli set-otpk
   ```

Join the Second Container to the DxEnterprise Cluster

1. Activate the DxEnterprise license for the second node using the command `dxcli activate-server`.

   Example

   ```
   docker exec dxemssql-1 dxcli activate-server AAAA-BBBB-CCCC-DDDD
   ```

2. Join the second node to the DxEnterprise cluster using the command `dxcli join-cluster-ex`. Use the default NAT proxy of **match.dh2i.com**.

   Syntax

   ```
   dxcli join-cluster-ex <target> <passkey> [do_nat [true|false]]
   ```

   Parameters

Name	Description	Required
target	The IP address of the target server or match.dh2i.com if joining via NAT matchmaking.	True
passkey	The passkey for the target cluster or OTPK if joining via NAT matchmaking.	True
do_nat	Use the NAT matchmaking service instead of a cluster passkey.	False

   Example

   ```
   docker exec dxemssql-1 dxcli join-cluster-ex match.dh2i.com
   331bc8bf-7096-99bc-05e5-0dd097393600 true
   ```

3. Add the second node to the existing availability group using the command `dxcli add-ags-node` command. The SQL Server sysadmin password must be encrypted using the command `dxcli encrypt-text`.

   Syntax

```
dxcli add-ags-node <vhost> <ags_name>
<node_name|instance_name|sql_login|sql_pass|mirror_port|availability
_mode|[tunnel_port]>
```

Parameters

Name	Description	Required
vhost	The name of the Vhost.	True
ags_name	The name of the availability group.	True
node_name	The name of the node.	True
instance_name	The name of the instance.	True
sql_login	The user name used to login to SQL Server. The SQL Server login can be any SQL login or Windows domain account that has been assigned the sysadmin role for the instance being managed. When supplying a domain account, the username needs to follow the UPN format (for example: user@domain.com).	True
sql_pass	The password used to login to SQL Server (encrypted using `dxcli encrypt-text`).	True
mirror_port	The mirroring port to use for the availability group (default is 5022).	True
availability_mode	synchronous_commit, asynchronous_commit, or configuration_only.	True
tunnel_port	The port to be used for the tunnel connection. This port should be unique per node.	False

Example

```
docker exec dxemssql-1 dxcli add-ags-node vhost1 ags1 "dxemssql-
1|mssqlserver|sa|
V1DUmkSp6vNmyLDbzT1+xg==|5022|synchronous_commit|40002"
```

Join the Third Container to the DxEnterprise Cluster

1. Activate the DxEnterprise license for the third node using the command `dxcli activate-server`.

   Example

   ```
   docker exec dxemssql-2 dxcli activate-server AAAA-BBBB-CCCC-DDDD
   ```

2. Join the third node to the DxEnterprise cluster using the command `dxcli join-cluster-ex` with the same parameters from step 2.

   Example

   ```
   docker exec dxemssql-2 dxcli join-cluster-ex match.dh2i.com
   331bc8bf-7096-99bc-05e5-0dd097393600 true
   ```

3. Add the third node to the existing availability group using the command `dxcli add-ags-node`.
   - Example 1: Add the third node as an additional synchronous_commit replica.

Example

```
docker exec dxemssql-2 dxcli add-ags-node vhost1 ags1
"dxemssql-2|mssqlserver|sa|
V1DUmkSp6vNmyLDbzT1+xg==|5022|synchronous_commit|40003"
```

- Example 2: Add the third node as a configuration_only replica.

Example

```
docker exec dxemssql-2 dxcli add-ags-node vhost1 ags1
"dxemssql-2|mssqlserver|sa|
V1DUmkSp6vNmyLDbzT1+xg==|5022|configuration_only|40003"
```

Add the Availability Group Database(s)

1. (optional) If a database is not already available, create it using `sqlcmd` on the primary.
   - `docker exec dxemssql-0 /opt/mssql-tools/bin/sqlcmd -S localhost -U SA -P p@ssw0rd -Q "create database db1"`
2. Add databases to the availability group from the primary using the command `dxcli add-ags-databases`.

Syntax

```
dxcli add-ags-databases <vhost> <ags_name> <db_name>
```

Parameters

Name	Description	Required
vhost	The name of the Vhost.	True
ags_name	The name of the availability group.	True
db_name	The name of the SQL Server database (comma-separated list for multiples).	True

Example

```
docker exec dxemssql-primary-0 dxcli add-ags-databases vhost1 ags1
db1
```

3. Verify the details of the availability group using the command `dxcli get-ags-detail`.

Syntax

```
dxcli get-ags-detail <vhost> <ags_name>
```

Parameters

Name	Description	Required
vhost	The name of the Vhost.	True
ags_name	The name of the availability group.	True

Example

```
docker exec dxemssql-primary-0 dxcli get-ags-detail vhost1 ags1
```

References

- Docker CLI Reference
- DxEnterprise DxCLI Reference
- Quorum Considerations for SQL Server Availability Groups