Denodo Docker container configuration

Revision 20240207

Denodo Inc.: 525 University Avenue, Suite 31, Palo Alto, CA 94301. USA www.denodo.com

CONTENTS

NOTE: This document applies to the Denodo Containers released with Update 20220728 and later.

1 CONTAINER STARTUP

When the Denodo Docker container starts it will execute the `.sh` files under `/container-entrypoint-preinit` before initializing or starting any Denodo Platform components. This folder can be mounted as a volume. This is useful to execute some configuration scripts like `denodo_tls_configurator.sh`, `regenerateMetadata.sh`, etc.

1.1 CONTAINER LAUNCH

To indicate what servers will be started the following arguments can be used:

`--vdpserver|--vqlserver`: to launch the Denodo Virtual DataPort (VDP) Server.
`--datacatalog`: to launch the Data Catalog.
`--designstudio`: to launch the Web Design Studio.
`--dmt`: to launch the Diagnostic & Monitoring. Tool.
`--monitor`: to launch the Denodo Monitor
`--sso|--denodosso`: to launch Denodo SSO.
`--schindex`: to launch the Scheduler Index Server.
`--schserver`: to launch the Scheduler Server.
`--schadmin`: to launch the Scheduler Administration Tool.
`--webpanel`: to launch the Web Panel.
`--lmserver|--licensemanager`: to launch the License Manager.
`--smserver|--solutionmanager`: to launch the Solution Manager server.
`--smadmin|--solutionmanagerwebtool`: To launch Solution Manager Administration Tool.

For example, to launch a Denodo Platform container with VDP, Design Studio and the Data Catalog:

```
$ docker run -d -h denodo-vdpserver \
-p 9999:9999 -p 9997:9997 -p 9996:9996 -p 9995:9995 -p 9090:9090 \
-v <PATH_TO_LOCAL_LICENSE>:/opt/denodo/conf/denodo.lic \
--name denodo-vdpserver \
denodo-platform:latest --vdpserver –-designstudio –-datacatalog
```

To launch a Solution Manager container run:

```
$ docker run -d -h solution-manager \
-p 10091:10091 -p 10090:10090 -p 19090:19090 \
-v <PATH_TO_LOCAL_LICENSE>:/opt/denodo/conf/denodo.lic \
--name solution-manager \
solution-manager:latest --lmserver –-smserver –-smadmin
```

2 INITIALIZING A NEW CONTAINER

When the container is executed for the first time, it will execute the `.sh`, `.vql` and `.zip` files located under `/container-entrypoint-init`. This execution takes place after the execution of the scripts under `/container-entrypoint-preinit`. This folder can be mounted as a volume.
These files are executed against a running Denodo instance for initialization, using the flag `–singleuser`. After the scripts and VQL are executed, the Denodo instance will be stopped.
For the `.zip` files to be executed they must match to the following syntax:

- `dc-metadata-*.zip`: to import the Data Catalog metadata.
- `denodo-scheduler-*.zip`: to import Scheduler metadata.

Any other `zip` files will be ignored.
The VQL files can be associated with a `.properties` file during the import, using the same filename as the associated `.vql`. For instance, if we have `file_to_import.vql` and `file_to_import.properties` located under the same subfolder under `/container-entrypoint-init` they will be imported together.

3 CONTAINER SHUTDOWN

When the Denodo Docker container stops It will execute the `.sh` files located under `/container-entrypoint-prestop` before stopping the Denodo Platform components. This folder can be mounted as a volume.
NOTE: This execution is launched after a graceful shutdown launched on `SIGTERM`. But not if the container is removed or killed.

4 MOUNTING CONFIGURATION FILES

When the Denodo container starts all the files located under the `/denodo/conf` folder of the container will be copied into the `/opt/denodo/conf` folder. This allows loading a static configuration from `ConfigMap` without any permission issues when Denodo writes to internal configuration files because the read-only `ConfigMap` files are mounted into an external location.
If the environment variable `DENODO_MERGE_CONF` is set to true then the configuration `.properties` files are not replaced. The `properties` files will be merged into the final ones.
Once all configuration files have been copied and all the environment variables have been applied, then the `regenerateFiles.sh` script will be executed to make those changes effective.

NOTE: If you have mounted configuration properties files directly under `/opt/denodo/conf/` in case of applying a product update or using a new updated image, the new properties added by the update would be lost because they are replaced with the mounted files.
To avoid this, first try to translate the configuration properties into Environment Variables. If this is not possible:

- Set the environment variable `DENODO_MERGE_CONF` to `true`
- Mount the configuration files but include only the properties that have been modified into the corresponding folder under `/denodo/conf/` instead of the final one under `/opt/denodo/conf/`

Other external files, like libraries, that are not distributed with Denodo can be mounted at:

Mounted path	**Installation path**
`/denodo/conf`	`/opt/denodo/conf`
`/denodo/lib/lib-external`	`/opt/denodo/lib-external`
`/denodo/lib/data-catalog-extensions`	`/opt/denodo/lib/data-catalog-extensions`
`/denodo/lib/scheduler-extensions`	`/opt/denodo/lib/scheduler-extensions`
`/denodo/lib/solution-manager-extensions`	`/opt/denodo/lib/solution-manager-extensions`
`/denodo/extensions/thirdparty`	`/opt/denodo/extensions/thirdparty`

5 MOUNTING VOLUMES TO PERSIST DATA

In some cases we will be interested in persisting data such as Denodo metadata, logs or custom configuration files, so the data remains available once the container dies. To do this, we can mount external volumes on the container so the data is persisted there. We can change the ownership of the mounted folders to the user and/or group using their id (`uid: 999` and `gid:999` by default). For instance using the following:

```
chown -R 999:999 <path_to_mounted_volume>
```

Openshift images use an arbitrary user (which is always a member of the root group) and the previous solution can not be applied. For Openshift images add the root group to the mounted volumes:

```
chgrp -R 0 <path_to_mounted_volume>
```

If the previous solutions do not work, another option will be to grant all permissions into the mounted volumes:

```
chmod -R 777 <path_to_mounted_volume>
```

6 ENVIRONMENT VARIABLES

The following sections list the environment variables available to configure and set up the container.

6.1 CONTAINER STARTUP LOGS

Custom environment variable to enable more details of the container startup:

- `DENODO_DEBUG`: Enable it to retrieve log traces about the container initialization. Uses `false` as default.

6.2 ADMINISTRATOR CREDENTIALS

- `DENODO_USERNAME`: Specifies the administrator username. Uses `admin` by default.
- `DENODO_PASSWORD`: Defines the administrator password. Uses `admin` by default.
- `DENODO_PASSWORD_ENCRYPTED`: Enable it if `DENODO_PASSWORD` is encrypted using the script: `<DENODO_HOME>/bin/encrypt_password.sh 'clear_password'`. Uses false as default.

Example:

```
$ docker run -d -h denodo-vdpserver -p 9999:9999 -p 9997:9997 -p 9996:9996
-p 9995:9995 -p 9090:9090 --name denodo-vdpserver -e DENODO_USERNAME=adm -e
DENODO_PASSWORD=pa$$w0rd1 denodo-platform:latest --vdpserver
```

6.3 DENODO PLATFORM NETWORK CONFIGURATION

- `DENODO_PORT`: Uses 9999 by default.
- `DENODO_SHUTDOWN_PORT`: Uses 9998 by default.
- `DENODO_ODBC_PORT`: Uses 9996 by default.
- `DENODO_REGISTRY_PORT`: Uses 9997 by default.
- `DENODO_FACTORY_PORT`: Uses 9995 by default.
- `DENODO_REGISTRY_HOST`: Uses the environment variable HOSTNAME by default.

6.4 DENODO PLATFORM JVM CONFIGURATION

- `DENODO_JVM_OPTS`: To change the parameters of the Java Virtual Machine (JVM) used to launch the denodo server. Empty by default.

6.5 WEB CONTAINER CONFIGURATION

- `DENODO_WEBCONTAINER_STARTUP`: Enable it to launch the web container during the Denodo startup. For example, set it to true if you need to deploy some webservices. Uses `false` as default.
- `DENODO_WEBCONTAINER_HTTP_PORT`: Port that the web container will listen for incoming requests. It is the same property for HTTP and HTTPS requests. Uses 9090 as default.
- `DENODO_WEBCONTAINER_SHUTDOWN_PORT`: Port that the web container will listen for shutdown requests. Uses 9099 as default.
- `DENODO_WEBCONTAINER_JMX_PORT`: Port that the web container will listen for requests from JMX monitoring tools such as Denodo Monitor, Java VisualVM or JConsole. In addition, it is used by the other modules of the Denodo Platform to send requests to the web container such as deploy or undeploy a Denodo administration tool, a web service, etc. Uses 9098 as default.
- `DENODO_WEBCONTAINER_RMI_PORT`: Auxiliary port used by the web container to communicate with its clients. I.e. JMX monitoring tools and other modules of the Denodo Platform. Uses 9097 as default.
- `DENODO_WEBCONTAINER_RMI_HOST`: Auxiliary host used by the web container to communicate with its clients. Uses `localhost` as default.
- `DENODO_WEBCONTAINER_ENGINE`: To set the name of Service and Engine elements into the web container configuration. Empty as default.
- `DENODO_WEBCONTAINER_OPTS`: To change the parameters of the Java Virtual Machine (JVM) used to launch the web container. Empty as default.
- `DENODO_WEBCONTAINER_PROTO`: Internally used by the container initialization to compose the ping URLs for the web container. If you set TLS configuration remember to change it to `https`. Uses `http` as default.

6.6 TLS CONFIGURATION

The Denodo Platform is configured to launch the Denodo SSL/TLS Configurator Script. The following variables are used in this process:

- `DENODO_TLS_TRUSTSTORE`: Uses `<DENODO_JAVA_DIR>/lib/security/cacerts` by default.
- `DENODO_TLS_TRUSTSTORE_PASSWORD`: Uses `changeit` by default.
- `DENODO_TLS_TRUSTSTORE_PASSWORD_ENCRYPTED`: Uses `false` by default.
- `DENODO_TLS_KEYSTORE`: Path to the JKS keystore that contains the private key to be used. If it does not exist, it will be generated (except when TLS is configured using this JKS as key container). Uses `<DENODO_TMP_DIR>/tls/keystore.jks` by default.
- `DENODO_TLS_KEYSTORE_PASSWORD`: Password of the JKS keystore.
- `DENODO_TLS_KEYSTORE_PASSWORD_ENCRYPTED`: Uses false by default.

6.7 JKS AND CER CERTIFICATE

- `DENODO_TLS_CER_CERT`: Path to a `CER` file with a certificate that will be imported into the selected truststore. This `X.509` certificate must be associated to the private key found in the provided JKS keystore
- `DENODO_TLS_CER_CHAIN`: Path to an optional `CER` chain file. The chain of certificates will be imported into the selected truststore.

6.8 PKCS12 BUNDLE

- `DENODO_TLS_PKCS12`: Path to the `PKCS12` Bundle.
- `DENODO_TLS_PKCS12_PASSWORD`: Password of the `PKCS12` Bundle.
- `DENODO_TLS_PKCS12_PASSWORD_ENCRYPTED`: Uses `false` by default.

6.9 PEM KEY AND CERTIFICATE

- `DENODO_TLS_PEM_KEY`: Path to a file with a PEM-encoded, unencrypted private key that will be used to initialize a keystore in the selected keystore path.
- `DENODO_TLS_PEM_CERT`: Path to a file with a PEM-encoded public `X.509` certificate that will be imported into the selected truststore. This certificate must be associated with the provided private key.
- `DENODO_TLS_PEM_CHAIN`: Optional list of paths to PEM-encoded files with a public certificate chain that will be imported into the selected truststore.

For example to configure with a `PKCS12` bundle, mount `/container-entrypoint-init` with the `p12` bundle:

```
/container-entrypoint-init
└── certificate.p12
```

and start the container with:

```
$ docker run -d -h denodo-vdpserver -p 9999:9999 -p 9997:9997 -p 9996:9996
-p 9995:9995 -p 9443:9443 \
-v /mnt/c/tmp/Denodo/init:/container-entrypoint-init  \
-e DENODO_TLS_KEYSTORE_PASSWORD=G3rRE+hPuhyR90 \
-e DENODO_TLS_KEYSTORE_PASSWORD_ENCRYPTED=true \
-e DENODO_TLS_PKCS12='/container-entrypoint-init/certificate.p12' \
-e DENODO_TLS_PKCS12_PASSWORD=qQKUeQdu7ZT94e957cMSkA \
-e DENODO_TLS_PKCS12_PASSWORD_ENCRYPTED=true \
--name denodo-vdpserver denodo-platform:latest --vdpserver
```

NOTE: The web container port when TLS is configured will change, by default:
- 9443 for the Denodo Platform web container.
- 19443 for the Denodo Solution Manager web container.

NOTE: Encrypted values must be generated with the `<DENODO_HOME>/bin/encrypt_password.sh` script.

6.10 IMPORT CERTIFICATES

- `DENODO_IMPORT_CERTS`: Whitespace separated list of certificates files to be imported. The `alias` will be the filename without extension.

6.11 EXTERNAL METADATA DATABASE

- `DENODO_DATABASE_PROVIDER`: Adapter name (For example: derby, mysql, oracle, postgresql, sqlserver, azure).
- `DENODO_DATABASE_PROVIDER_VERSION`: Adapter version (one of the supported engines).

- DENODO_DATABASE_FILE: External metadata configuration file. Check the official documentation Storing the Metadata on an External Database.
- DENODO_DATABASE_URI: Database access URI.
- DENODO_DATABASE_DRIVER: Name of the Java class of the JDBC adapter to be used.
- DENODO_DATABASE_DRIVER_EXTERNAL: Path to a non default location at Denodo where the driver file is located. This file will be added to the new catalog. If the given path is a directory, then all files inside will be added.
- DENODO_DATABASE_DRIVER_PROPERTIES: Driver properties with JSON format. E.g. `"{"prop1":"value1", "prop2":"value2"}"`.
- DENODO_DATABASE_CLASSPATH: Path for the folder that contains the JAR files with the implementation classes needed by the JDBC adapter.
- DENODO_DATABASE_USER: Database user name.
- DENODO_DATABASE_PASSWORD: If it is not encrypted it will be encrypted using the script: `<DENODO_HOME>/bin/encrypt_password.sh`
- DENODO_DATABASE_PASSWORD_ENCRYPTED: Uses `false` as default.
- DENODO_DATABASE_CATALOG: Database custom catalog.
- DENODO_DATABASE_SCHEMA: Database custom schema.
- DENODO_DATABASE_INITIAL_SIZE: Pool initial size.
- DENODO_DATABASE_MAX_ACTIVE: Pool max active.
- DENODO_DATABASE_TEST: Pool test connections. It requires a validation query value.
- DENODO_DATABASE_QUERY: Pool validation query for connections.
- DENODO_DATABASE_RESET: Reset metadata if exists (disabled by default, reusing metadata if exists).

6.12 GLOBAL LDAP AUTHENTICATION

- DENODO_LDAP_ENABLED: Check to enable LDAP authentication. Allow to retrieve users from a LDAP server and authenticate them with their credentials. Uses `false` by default.
- DENODO_LDAP_DATABASE: `admin` by default.
- DENODO_LDAP_NAME: `ldapds` by default.
- DENODO_LDAP_URI: Uri to the LDAP server. For example in the following format `ldap://ldap-server-hostname:port`
- DENODO_LDAP_USER: Username to access the LDAP server.
- DENODO_LDAP_PASSWORD: User password to access the LDAP server.
- DENODO_LDAP_PASSWORD_ENCRYPTED: Enable it if the DENODO_LDAP_PASSWORD is encrypted by executing the following VQL: ENCRYPT_PASSWORD 'clear_password' . Uses false as default..
- DENODO_LDAP_GSSAPI: Authentication mechanism to connect to the LDAP server with SASL binding with GSSAPI authentication mechanism, instead of “simple binding”. Uses `false` as default.
- DENODO_LDAP_PAGING: Enable this if the LDAP server limits the number of results per query. Uses `false` as default.
- DENODO_LDAP_MAX_PAGE_SIZE: Number of results per page. 1000 as default.
- DENODO_LDAP_ADD_ALLUSERSROLE: If selected, the Server will grant the privileges of the role “`allusers`” to all the users that log in successfully even if this role has not been assigned to the user in the LDAP server. False as default.
- DENODO_LDAP_USER_BASE: Nodes of the LDAP server that are used as scope to search the nodes that represent users. For example to specify two different

nodes:
`"\"CN=Users,DC=dc1,DC=dm,DC=com\" \"CN=Users,DC=dc2,DC=dm,DC=com\""`

- `DENODO_LDAP_USER_ATTRIBUTE=`: Name of the attribute that contains the user name of users, in the nodes that represent users. cn as default
- `DENODO_LDAP_USER_SEARCH`: Pattern used to generate the LDAP queries that will be executed to obtain the nodes that represent the users that try to connect to the Server. For example: `"(&(objectClass=user))"`
- `DENODO_LDAP_ROLE_BASE`: Nodes of the LDAP server that are used as scope to search the nodes that represent Roles. For example to specify two different nodes:
  `"\"CN=Users,DC=dc1,DC=dm,DC=com\" \"CN=Users,DC=dc2,DC=dm,DC=com\""`
- `DENODO_LDAP_ROLE_ATTRIBUTE`: Name of the attribute that contains the name of the role, in the nodes that represent roles. cn as default
- `DENODO_LDAP_ROLE_SEARCH`: Pattern used to generate the LDAP queries that will be executed to obtain the nodes that represent the roles of a user. For example: `"(&(member=@{USERDN})(objectClass=group))"`

6.13 KERBEROS AUTHENTICATION

- `DENODO_KERBEROS_ENABLED`: To enable the kerberos authentication. Uses `false` as default.
- `DENODO_KERBEROS_DEBUG`: `false` by default.
- `DENODO_KERBEROS_PRINCIPAL:`Enter the Service Principal Name (SPN) used to create the keytab file.
- `DENODO_KERBEROS_DISABLE_REFERRALS`: Uses `true` as default.
- `DENODO_KERBEROS_KEYTAB_FILE:` Path to the keytab file.
- `DENODO_KERBEROS_CONF_FILE: `: Path to the kerberos configuration file. (krb5.ini or krb5.conf)
- `DENODO_KERBEROS_KEYTAB_RESOURCE: `: Internal reference to the keytab file as resource. Empty by default.
- `DENODO_KERBEROS_CONF_RESOURCE: `: Internal reference to the kerberos configuration file as resource. Empty by default.
- `DENODO_KERBEROS_ROLE_EXTRACTION`: `GLOBAL_LDAP` by default.
- `DENODO_KERBEROS_ADD_ALLUSERSROLE`: `false` by default.
- `DENODO_KERBEROS_EXCLUDE_DOMAIN_NAME`: To exclude the domain for LDAP Uses `false` as default.

6.14 LICENSE MANAGER

Denodo Platform container:

- `DENODO_LM_PROTO`: Scheme or protocol of the License Manager. Default value: `http`
- `DENODO_LM_HOST`: Hostname of the License Manager.
- `DENODO_LM_PORT`: Port of the License Manager. Uses 10091 as default.

Solution Manager container:

- `DENODO_LM_PORT:` listening port for the License Manager server. Uses 10091 as default.

6.15 DENODO SECURITY TOKEN

- `DENODO_SSO_PROTO`: Scheme or protocol of the Denodo Security Token. Uses `http` as default.
- `DENODO_SSO_HOST`: Hostname of the Denodo Security Token.
- `DENODO_SSO_PORT`: Port of the Denodo Security Token. Empty by default.
- `DENODO_SSO_LOGIN_ENABLED`:Denodo Security Token Single Sign-On enabled for the web applications deployed in this installation. Uses `false` as default value.

6.16 SOLUTION MANAGER

In a Solution Manager installation, these parameters are used for configuring the Solution Manager used by the web tool and for the proper listening port of the solution manager server.

- `DENODO_SM_PROTO`: Scheme or protocol of the Solution Manager. Uses `http` as default.
- `DENODO_SM_HOST`: Hostname of the Solution Manager.
- `DENODO_SM_PORT`: Port of the Solution Manager. Uses 10090 as default.

6.17 SOLUTION MANAGER DATABASE

- `DENODO_SM_DATABASE_PROVIDER`: Possible values are `EMBEDDED_DERBY`, `DERBY`, `ORACLE_11G`, `ORACLE_12C`, `SQL_SERVER`, `MYSQL`, `POSTGRE_SQL`.
- `DENODO_SM_DATABASE_URI`: : JDBC URL to connect to your database server.
- `DENODO_SM_DATABASE_DRIVER`: Class name of the JDBC driver. E.g. for Oracle `oracle.jdbc.OracleDriver`.
- `DENODO_SM_DATABASE_USER`: User credentials to connect to the database. This user account needs privileges to create tables and insert/update/delete rows in these tables.
- `DENODO_SM_DATABASE_PASSWORD`: If it is not encrypted it will be encrypted using the script: `<DENODO_HOME>/bin/encrypt_password.sh`
- `DENODO_SM_DATABASE_PASSWORD_ENCRYPTED`: Uses `false` as default.

NOTE: The database password will be encrypted into the final configuration file.
If the external database requires specific drivers they must be mounted into the correct folder for the Solution Manager.

Mounted path	Installation path
`/denodo/lib/solution-manager-extensions`	`/opt/denodo/lib/solution-manager-extensions`

6.18 SERVER REGISTRATION

- `DENODO_REGISTER_ENVIRONMENT_NAME`: Name of the Standard environment where the cluster is registered.
- `DENODO_REGISTER_ENVIRONMENT_LICENSE`: The license alias assigned to this environment. It must be a valid value from the available scenarios of the current license installed in the Solution Manager. Mandatory when the environment does not exist previously and it will be created.

- `DENODO_REGISTER_CLUSTER_NAME`: Name of the cluster where the server will be registered.
- `DENODO_REGISTER_CLUSTER_VDP_URL`: Optional variable to set the VDP Server Load Balancer URL into the cluster configuration for My Applications Panel.
- `DENODO_REGISTER_CLUSTER_DC_URL`: Optional variable to set the Data Catalog Load Balancer URL into the cluster configuration for My Applications Panel.
- `DENODO_REGISTER_CLUSTER_SCH_URL`: Optional variable to set the Scheduler Server Load Balancer URL into the cluster configuration for My Applications Panel.
- `DENODO_REGISTER_SERVER_NAME`: Name of the server. This name has to be unique. You cannot assign the same name to two servers, even if they are in different environments or clusters. Uses the `HOSTNAME` by default.
- `DENODO_REGISTER_SERVER_PORT`: Port where the server component listens to incoming requests. Uses `DENODO_PORT` by default.
- `DENODO_REGISTER_SERVER_TYPE`: Type of server component. Possible values are `VDP`, `VDP_DATA_CATALOG`, `SCHEDULER`. `VDP` by default. Supports a whitespace separated list.
- `DENODO_REGISTER_SERVER_INSECURE`: To ignore invalid and self-signed certificate checks server registration. Default value false
- `DENODO_REGISTER_SERVER_USERNAME`: Username credentials to connect to the server. Empty by default.
- `DENODO_REGISTER_SERVER_PASSWORD`: Password credentials to connect to the server. Empty by default.
- `DENODO_REGISTER_SERVER_PASSWORD_ENCRYPTED`: If the password to connect to the server is encrypted by the script: `<DENODO_HOME>/bin/encrypt_password.sh`. Uses false as default.
- `DENODO_REGISTER_SERVER_USE_KERBEROS`: Select this to use Kerberos authentication to connect to the server. Uses `false` by default.
- `DENODO_REGISTER_SERVER_USE_PASS_THROUGH`: Select this to create the revisions using the credentials of the user that is logged in the Solution Manager, instead of the credentials specified by `DENODO_REGISTER_SERVER_USERNAME` and `DENODO_REGISTER_SERVER_PASSWORD`. Uses `true` by default.
- `SOLUTION_MANAGER_USERNAME`: Username to authenticate the register request against the Solution Manager. Uses a default username if it is not present.
- `SOLUTION_MANAGER_PASSWORD`: Password to authenticate the register request against the Solution Manager. Uses a default password if it is not present.
- `DENODO_SM_PROTO`: To compose the register request URL. `http` by default.
- `DENODO_SM_HOST`: To compose the register request URL
- `DENODO_SM_PORT`: To compose the register request URL. 10090 by default.

Example of use:

```
docker run -d -h denodo-vdpserver -p 9999:9999 -p 9997:9997 -p 9996:9996 -p
9995:9995 -p 9090:9090 -e DENODO_LM_HOST=host.docker.internal -e
DENODO_SM_HOST=host.docker.internal -e DENODO_SSO_HOST=host.docker.internal
-e DENODO_SSO_PORT=19090 -e DENODO_REGISTER_ENVIRONMENT_NAME=local -e
DENODO_REGISTER_ENVIRONMENT_LICENSE=PRODUCTION -e
DENODO_REGISTER_CLUSTER_NAME=local -e DENODO_REGISTER_SERVER_NAME=local -e
DENODO_REGISTER_SERVER_TYPE=VDP --name denodo-vdpserver denodo-
platform:latest --vdpserver
```

6.19 DATA CATALOG

- `DENODO_DC_PASSWORD`: To change the password of the web local authentication of DataCatalog.
- `DENODO_DC_PASSWORD_ENCRYPTED`: If the Data Catalog password is encrypted by the script: `<DENODO_HOME>/bin/encrypt_password.sh`. Uses false as default.

6.20 DATA CATALOG VDP SERVER

- `DENODO_DC_SERVER_NAME`: The name that will be shown in the login page for the default VDPP server.
- `DENODO_DC_SERVER_HOST`: The connection host of the default VDP server.
- `DENODO_DC_SERVER_PORT`: The connection port of the default VDP server.

6.21 DATA CATALOG DATABASE

- `DENODO_DC_DATABASE_PROVIDER`: Select the database provider you want to use. For example: derby, mysql, oracle, postgresql, sqlserver, azure or aurora postgresql.
- `DENODO_DC_DATABASE_URI`: The connection URL to the database.
- `DENODO_DC_DATABASE_DRIVER`: The name of the Java class of the JDBC driver to be used.
- `DENODO_DC_DATABASE_USER`: Use the credentials of the account used to connect to the database.
- `DENODO_DC_DATABASE_PASSWORD`: If it is not encrypted it will be encrypted using the script: `<DENODO_HOME>/bin/encrypt_password.sh`.
- `DENODO_DC_DATABASE_PASSWORD_ENCRYPTED`: Uses `false` as default.
- `DENODO_DC_DATABASE_MAX_POOL`: The maximum number of actual connections to the database, including both idle and in-use connections.
- `DENODO_DC_DATABASE_MIN_IDLE`: The minimum number of idle connections that the Data Catalog tries to maintain in the pool.
- `DENODO_DC_DATABASE_TIMEOUT`: The maximum number of milliseconds that the Data Catalog will wait for a connection from the pool. If this time is exceeded without a connection becoming available, an error will be thrown.
- `DENODO_DC_DATABASE_QUERY`: The query that will be executed just before using a connection from the pool to validate that it is still alive.

If the external database requires specific drivers, they must be mounted into the correct folder for the Data Catalog.

Mounted path	Installation path
`/denodo/lib/data-catalog-extensions`	`/opt/denodo/lib/data-catalog-extensions`

6.22 SCHEDULER INDEX SERVER

- DENODO_SCHINDEX_PORT: Uses 9000 by default.
- DENODO_SCHINDEX_SHUTDOWN_PORT: Uses 8999 by default.
- DENODO_SCHINDEX_FACTORY_PORT: Uses 8998 by default.
- DENODO_SCHINDEX_REGISTRY_HOST: Uses the environment variable HOSTNAME by default.
- DENODO_SCHINDEX_JVM_OPTS: To change the parameters of the Java Virtual Machine (JVM) used to launch the scheduler index. Empty as default.
- DENODO_SCHINDEX_PASSWORD: To change the password of the local-based authentication of Scheduler Index.
- DENODO_SCHINDEX_PASSWORD_ENCRYPTED: If the Scheduler Index password is encrypted by the script: <DENODO_HOME>/bin/encrypt_password.sh. Uses false as default.

6.23 SCHEDULER SERVER

- DENODO_SCHSERVER_PORT: Uses 8000 by default.
- DENODO_SCHSERVER_SHUTDOWN_PORT: Uses 7999 by default.
- DENODO_SCHSERVER_FACTORY_PORT: Uses 7998 by default.
- DENODO_SCHSERVER_REGISTRY_HOST: Uses the environment variable HOSTNAME by default.
- DENODO_SCHSERVER_JVM_OPTS: To change the parameters of the Java Virtual Machine (JVM) used to launch the scheduler server. Empty by default.
- DENODO_SCHSERVER_PASSWORD: To change the password of the local-based authentication of Scheduler server.
- DENODO_SCHSERVER_PASSWORD_ENCRYPTED: If the Scheduler Server password is encrypted by the script: <DENODO_HOME>/bin/encrypt_password.sh. Uses false as default.

6.24 SCHEDULER VDP SERVER

- DENODO_SCHSERVER_SERVER_HOST: To configure the default VDP server host.
- DENODO_SCHSERVER_SERVER_PORT: To configure the default VDP server port.

6.25 SCHEDULER DATABASE

- DENODO_SCHSERVER_DATABASE_PROVIDER: Adapter name. Possible values are derby, mysql, oracle, postgresql, sqlserver, azure, aurora mysql, aurora postgresql.
- DENODO_SCHSERVER_DATABASE_PROVIDER_VERSION: Adapter version.
- DENODO_SCHSERVER_DATABASE_URI: Database access URI.
- DENODO_SCHSERVER_DATABASE_DRIVER: Name of the Java class of the JDBC adapter to be used.
- DENODO_SCHSERVER_DATABASE_DRIVER_EXTERNAL: Path to a non default location where the driver file is located. This file will be added to the new catalog. If the given path is a directory, then all files inside will be added.
- DENODO_SCHSERVER_DATABASE_CLASSPATH: Path for the folder that contains the JAR files with the implementation classes needed by the JDBC adapter.
- DENODO_SCHSERVER_DATABASE_USER: Database user name.
- DENODO_SCHSERVER_DATABASE_PASSWORD: If it is not encrypted it will be encrypted using the script: <DENODO_HOME>/bin/encrypt_password.sh.

- DENODO_SCHSERVER_DATABASE_PASSWORD_ENCRYPTED: Uses false as default.
- DENODO_SCHSERVER_DATABASE_INITIAL_SIZE: Pool initial size.
- DENODO_SCHSERVER_DATABASE_MAX_ACTIVE: Pool max active connections.

If the external database requires specific drivers, they must be mounted into the correct folder for Scheduler.

Mounted path	Installation path
/denodo/lib/scheduler-extensions	/opt/denodo/lib/scheduler-extensions

6.26 SCHEDULER WEB TOOL

- DENODO_SCHADMIN_PASSWORD: To change the password of the web local authentication of Scheduler Admin Tool.
- DENODO_SCHADMIN_PASSWORD_ENCRYPTED: If the Scheduler Administration Tool password is encrypted by the script: <DENODO_HOME>/bin/encrypt_password.sh. Uses false as default.

6.27 DESIGN STUDIO

- DENODO_DS_PASSWORD: To change the password of the web local authentication of Design Studio.
- DENODO_DS_PASSWORD_ENCRYPTED: If the Web Design Studio password is encrypted by the script: <DENODO_HOME>/bin/encrypt_password.sh. Uses false as default.

7 LOGGING

By default the `entrypoint` script redirects logs files to the standard output using soft links. This behavior can be managed with the environment variable `DENODO_KEEP_LOG_FILES` which is `false` by default. If set to `true` the logs are not redirected to the standard output.
Examples of symbolic links to redirect logs files:

```
   $ ln -sf /dev/stdout "$DENODO_LOG_DIR/vdp/vdp.log"
```

```
$ ln -sf /proc/1/fd/1 "$DENODO_LOG_DIR/design-studio/design-studio-
backend.log"
```

For subprocesses like the web container, the logs will be redirected to `/proc/1/fd/1`, because it is not running on the primary process of the container.
These logs, which are redirected to stdout, are tagged with their origin: `VDP`, `LM`, `SM`, `SMADMIN`, `DC`, `DS`, `DMT`, `SCHINDEX`, `SCHSERVER`, `SCHADMIN`. For example:

```
[VDP] 7200 [main] INFO  2022-05-27T09:44:18.989 server.start [] - TLS is
disabled on incoming connections
```

With this approach the logs are collected by the default flow of containers. It can be retrieved with the following commands:

```
 $ docker logs <container>
```

```
$ kubectl logs <pod>
```

Furthermore, a node-level agent can collect them and forward them into a centralized logging system.