



# Corporate Memory Control (cmemc)

v20.12

Manual



# Contents

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduction</b>                            | <b>4</b>  |
| 1.1      | About eccenca Corporate Memory Control (cmemc) | 4         |
| 1.2      | Scope of delivery                              | 5         |
| <b>2</b> | <b>Installation</b>                            | <b>6</b>  |
| 2.1      | Linux installation                             | 6         |
| 2.2      | MacOS installation                             | 7         |
| 2.3      | Windows installation                           | 7         |
| <b>3</b> | <b>Configuration</b>                           | <b>9</b>  |
| <b>4</b> | <b>Reference</b>                               | <b>12</b> |
| 4.1      | Command group: admin                           | 12        |
| 4.1.1    | Command: admin bootstrap                       | 12        |
| 4.1.2    | Command: admin showcase                        | 13        |
| 4.1.3    | Command: admin status                          | 13        |
| 4.2      | Command group: config                          | 14        |
| 4.2.1    | Command: config check                          | 14        |
| 4.2.2    | Command: config edit                           | 15        |
| 4.2.3    | Command: config list                           | 15        |
| 4.3      | Command group: dataset                         | 15        |
| 4.3.1    | Command: dataset create                        | 16        |
| 4.3.2    | Command: dataset delete                        | 17        |
| 4.3.3    | Command: dataset download                      | 17        |
| 4.3.4    | Command: dataset inspect                       | 18        |
| 4.3.5    | Command: dataset list                          | 18        |
| 4.3.6    | Command: dataset upload                        | 19        |
| 4.4      | Command group: graph                           | 19        |
| 4.4.1    | Command: graph count                           | 20        |
| 4.4.2    | Command: graph delete                          | 20        |
| 4.4.3    | Command: graph export                          | 20        |
| 4.4.4    | Command: graph import                          | 21        |
| 4.4.5    | Command: graph list                            | 21        |
| 4.4.6    | Command: graph open                            | 22        |
| 4.5      | Command group: project                         | 22        |
| 4.5.1    | Command: project create                        | 22        |
| 4.5.2    | Command: project delete                        | 23        |

|       |   |    |
|-------|---|----|
| 4.5.3 | Command: project export . . . . .       | 23 |
| 4.5.4 | Command: project import . . . . .       | 24 |
| 4.5.5 | Command: project list . . . . .         | 24 |
| 4.6   | Command group: query . . . . .          | 25 |
| 4.6.1 | Command: query execute . . . . .        | 25 |
| 4.6.2 | Command: query list . . . . .           | 27 |
| 4.6.3 | Command: query open . . . . .           | 27 |
| 4.7   | Command group: vocabulary . . . . .     | 27 |
| 4.7.1 | Command: vocabulary cache . . . . .     | 28 |
| 4.7.2 | Command: vocabulary cache . . . . .     | 28 |
| 4.7.3 | Command: vocabulary install . . . . .   | 29 |
| 4.7.4 | Command: vocabulary list . . . . .      | 29 |
| 4.7.5 | Command: vocabulary open . . . . .      | 29 |
| 4.7.6 | Command: vocabulary uninstall . . . . . | 30 |
| 4.8   | Command group: workflow . . . . .       | 30 |
| 4.8.1 | Command: workflow execute . . . . .     | 31 |
| 4.8.2 | Command: workflow io . . . . .          | 31 |
| 4.8.3 | Command: workflow list . . . . .        | 32 |
| 4.8.4 | Command: workflow open . . . . .        | 32 |
| 4.8.5 | Command: workflow status . . . . .      | 33 |
| 4.9   | Command group: workspace . . . . .      | 33 |
| 4.9.1 | Command: workspace export . . . . .     | 33 |
| 4.9.2 | Command: workspace import . . . . .     | 34 |
| 4.9.3 | Command: workspace reload . . . . .     | 34 |

# 1 Introduction

This manual describes how to install and setup and use *eccenca Corporate Memory Control* (cmemc), the command line client for eccenca Corporate Memory. cmemc is intended for system administrators and Linked Data Expert, who wants to automate / remote control activities on Corporate Memory.

To use this manual, cmemc users should have basic knowledge on command line interfaces, terminal usage and config file creation and editing.

This system manual includes the following parts:

- Installation
- Configuration

This document covers installation and basic usage pattern of cmemc and is not intended to be complete in terms of being a reference for all available options and commands. However, cmemc provides detailed documentation for users via the `--help` option.

The main documentation resource for cmemc <https://eccenca.com/go/cmemc>.

## 1.1 About eccenca Corporate Memory Control (cmemc)

cmemc is the eccenca Corporate Memory Command Line Interface (CLI). It is developed in python and build and delivered as a stand alone single binary for Linux and Windows.

Main features of cmemc include:

- List, edit and check configurations.
- List, create, delete, inspect datasets.
- List, import, export, delete or open graphs.
- List, import, export, create or delete projects.
- List, execute or open local and remote SPARQL queries.
- List, install, uninstall and open vocabularies.
- List, execute, open or inspect workflows.
- Import or export the workspace.

## 1.2 Scope of delivery

The cmemc release package consists of the following files:

- `cmemc` - the Linux ELF 64-bit LSB executable (x86-64, version 1 (SYSV), dynamically linked), tested with Ubuntu
- `cmemc-rhel` - the Linux ELF 64-bit LSB executable (x86-64, version 1 (SYSV), dynamically linked), tested with RHEL
- `cmemc-macosx` - the Mach-O 64-bit executable (x86\_64), tested with macOS Mojave
- `cmemc.exe` - the PE32+ executable (console, x86-64), for Microsoft Windows, tested with Windows 10
- `cmemc_vXX.YY_Manual.pdf` - this document

## 2 Installation

Since cmemc is a stand alone binary, installation is not needed. cmemc can be started from a local path and also from a central binary path, such as `/usr/local/bin`. The only needed installation activity is to copy the binary to a path on your system which is in the `PATH` variable.

### 2.1 Linux installation

As a first step, output the content of the `PATH` variable in order to determine, the path to use:

```
user@ubuntu:/home/user/$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

Then unzip the distribution package and copy the binary to an acceptable path (in this example: `/usr/local/bin`)

**Note:** XXX depends on your actual version

```
user@ubuntu:/home/user/$ unzip cmemc-vXXX.zip
user@ubuntu:/home/user/$ cp cmemc-vXXX/cmemc /usr/local/bin
user@ubuntu:/home/user/$ chmod +x /usr/local/bin/cmemc
```

Finally, test your installation.

**Note:** XXX depends on your actual version

```
user@ubuntu:/home/user/$ cmemc --version
cmemc, version XXX
```

In case you are using bash or zsh as your shell, you can optionally enable tab completion for cmemc. This is documented on the [click framework homepage](https://click.palletsprojects.com/en/7.x/bashcomplete/#activation)<sup>1</sup>.

In order to enable tab completion with **bash** run the following command in your shell:

```
eval "$(_CMEMC_COMPLETE=source cmemc)"
```

In order to enable tab completion with **zsh** run the following command in your shell:

---

<sup>1</sup><https://click.palletsprojects.com/en/7.x/bashcomplete/#activation>

```
eval "$(_CMEMC_COMPLETE=source_zsh cmemc)"
```

You may want to add the corresponding line to your `.bashrc` or `.zshrc` file for your convenience.

## 2.2 MacOS installation

Installation on MacOS is similar to installation on Linux. Please copy the unpacked `cmemc-macosx` binary to a directory in your PATH (renaming it to `cmemc`). Since **zsh** is the default shell on MacOS, use the zsh completion as described above.

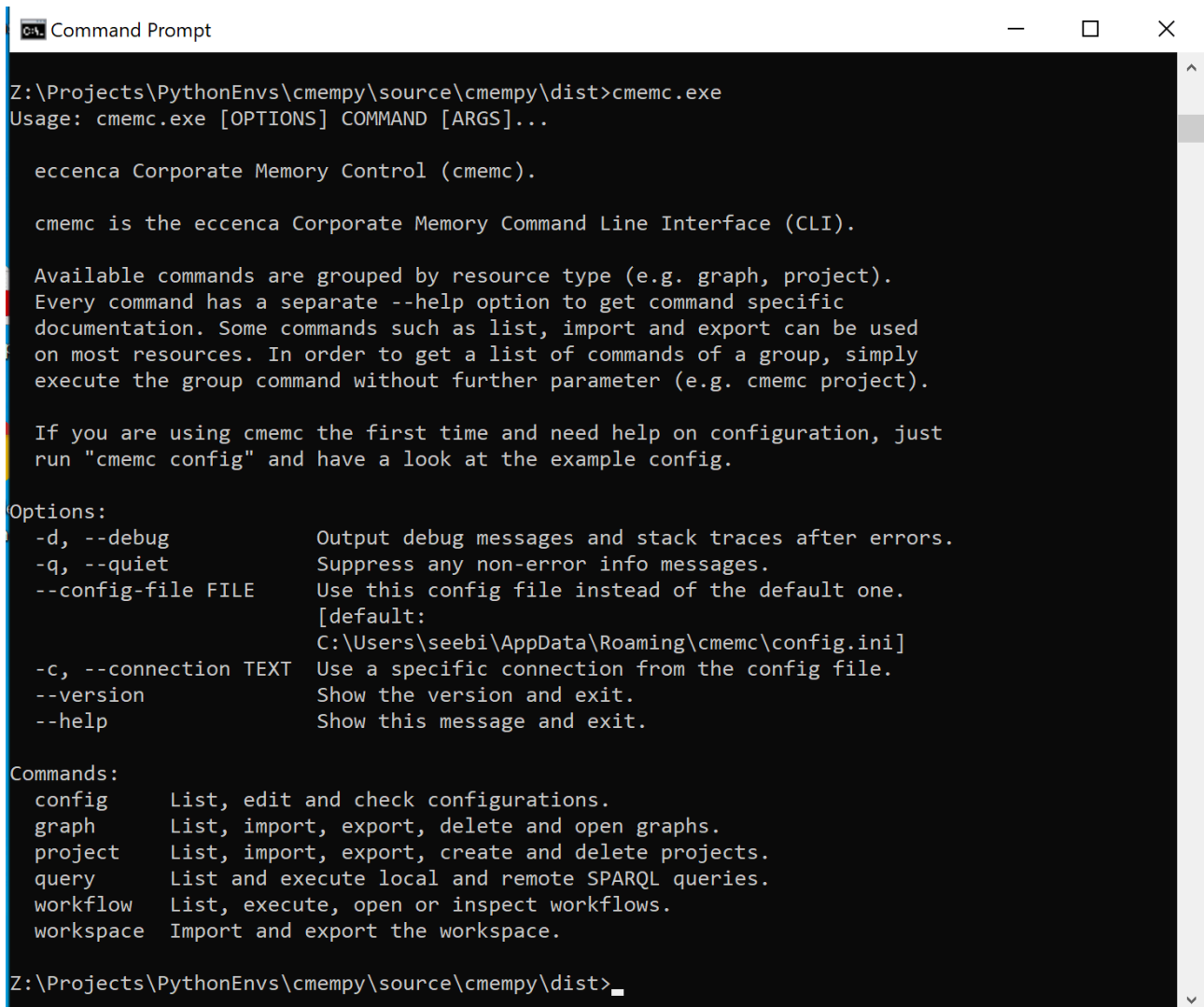
As an alternative installation method, you can use brew:

```
brew install eccenca/main/cmemc
```

## 2.3 Windows installation

The installation for Windows is similar. As a first step, unzip the distribution file ( `cmemc-vXXX.zip` ), then open `cmd.exe` and go to the directory where the cmemc files are extracted. Then you can start `cmemc.exe`.

**Note:** XXX depends on your actual version



```
Command Prompt

Z:\Projects\PythonEnvs\cmempy\source\cmempy\dist>cmemc.exe
Usage: cmemc.exe [OPTIONS] COMMAND [ARGS]...

eccenca Corporate Memory Control (cmemc).

cmemc is the eccenca Corporate Memory Command Line Interface (CLI).

Available commands are grouped by resource type (e.g. graph, project).
Every command has a separate --help option to get command specific
documentation. Some commands such as list, import and export can be used
on most resources. In order to get a list of commands of a group, simply
execute the group command without further parameter (e.g. cmemc project).

If you are using cmemc the first time and need help on configuration, just
run "cmemc config" and have a look at the example config.

Options:
  -d, --debug           Output debug messages and stack traces after errors.
  -q, --quiet           Suppress any non-error info messages.
  --config-file FILE    Use this config file instead of the default one.
                        [default:
                        C:\Users\seebi\AppData\Roaming\cmemc\config.ini]
  -c, --connection TEXT Use a specific connection from the config file.
  --version             Show the version and exit.
  --help               Show this message and exit.

Commands:
  config    List, edit and check configurations.
  graph     List, import, export, delete and open graphs.
  project   List, import, export, create and delete projects.
  query     List and execute local and remote SPARQL queries.
  workflow  List, execute, open or inspect workflows.
  workspace Import and export the workspace.

Z:\Projects\PythonEnvs\cmempy\source\cmempy\dist>
```

Figure 2.1: Example execution of cmemc under Windows



## 3 Configuration

cmemc needs to know where your Corporate Memory is deployed. For this, you need to provide some key variables in a configuration file. Per default, cmemc looks for this configuration file on a reasonable place depending on your operating system.

For Linux, this is `$HOME/.config/cmemc/config.ini`.

For Windows, this is `%APPDATA%\cmemc\config.ini`

**Note:** `USER` is your actual user name.

Once you start cmemc the first time, it will create an empty config file at this location and will output a general introduction. In order to do so, open the terminal application of your choice.

**Note:** All further examples given here are based on Linux commands. For Windows, the output is the same, however, you need to start cmemc as `cmemc.exe`.

```
user@ubuntu:/home/user/$ cmemc
Empty config created: /home/user/.config/cmemc/config.ini
Usage: cmemc [OPTIONS] COMMAND [ARGS]...

eccenca Corporate Memory Control (cmemc).

cmemc is the eccenca Corporate Memory Command Line Interface (CLI).

Available commands are grouped by affecting resource type (such as graph,
project and query). Each command / group has a separate --help screen to
get command / group specific documentation.

In order to see possible commands in a command group, simply execute the
group command without further parameter (e.g. cmemc project). Some
commands such as list, import and export can be executed in most command
groups. Please have a look at the cmemc manual page for more
information:

    https://eccenca.com/go/cmemc

Options:
  -d, --debug          Output debug messages and stack traces after errors.
```

```
-q, --quiet          Suppress any non-error info messages.
--config-file FILE   Use this config file instead of the default one.
                     [default: /Users/seebi/Library/Application
                     Support/cmемc/config.ini]

-c, --connection TEXT Use a specific connection from the config file.
--version            Show the version and exit.
-h, --help           Show this message and exit.
```

#### Commands:

```
config      List, edit and check configurations.
dataset     List, create, delete, inspect datasets.
graph       List, import, export, delete or open graphs.
project     List, import, export, create or delete projects.
query       List, execute or open local and remote SPARQL queries.
vocabulary  List, install, uninstall and open vocabularies.
workflow    List, execute, open or inspect workflows.
workspace   Import or export the workspace.
```

You should now edit your config file and add credentials and URL parameter to your Corporate Memory deployment. You either search the config manually in your home directory or you can use the `config edit` command, which opens the config file in your default text editor.

```
user@ubuntu:/home/user/$ cmemc config edit
Open editor for config file /home/user/.config/cmемc/config.ini
```

The rules for the config file are similar to a Windows INI file and are explained in detail at [docs.python.org](https://docs.python.org/3/library/configparser.html)<sup>1</sup>. Here is an basic example:

```
[my-local]
CMEM_BASE_URI=http://localhost/
OAUTH_GRANT_TYPE=client_credentials
OAUTH_CLIENT_ID=cmem-service-account
OAUTH_CLIENT_SECRET=c9c12831-000c-464b-9b1d-2d8b7e20df6a
```

This basically creates a named section `my-local` which is a connection to a Corporate Memory deployment on `http://localhost`. The authorization will be done with a system account `cmem-service-account` and the given client secret. Using this combination of config parameter is based on a typical installation where all components are available under the same hostname.

However, if you need to fine tune all locations, the following config file parameter can be used in addition to this example:

<sup>1</sup><https://docs.python.org/3/library/configparser.html>

- `DI_API_ENDPOINT` - Data Integration API endpoint, default: `CMEM_BASE_URI/dataintegration`
- `DP_API_ENDPOINT` - Data Platform API endpoint, default: `CMEM_BASE_URI/dataplatform`
- `OAuth_TOKEN_URI` - OAuth 2.0 Token endpoint, default: `CMEM_BASE_URI/auth/realms/cmem/protocol/openid-connect/token`
- `OAuth_GRANT_TYPE` - OAuth 2.0 grant type, default: `client_credentials`
- `OAuth_USER` - Username to retrieve the token, default: `admin`
- `OAuth_PASSWORD` - Password to retrieve the token, default: `admin`
- `OAuth_CLIENT_ID` - OAuth 2.0 client id, default: `cmem-service-account`
- `OAuth_CLIENT_SECRET` - OAuth 2.0 client secret, default: `secret`
- `SSL_VERIFY` - Use SSL verification for requests to DP/DI default: `True`

In order to verify your configuration, you should try to get a list of graphs via cmemc:

```
user@ubuntu:/home/user/$ cmemc -c my-local graph list
https://ns.eccenca.com/example/data/vocabs/
https://vocab.eccenca.com/shacl/
urn:elds-backend-access-conditions-graph
https://ns.eccenca.com/data/queries/
```

If you get a similar list of graphs, you successfully configured cmemc to access your deployment.

## 4 Reference

This section lists the help texts of all commands as a reference and to search for it.

### 4.1 Command group: admin

Usage: cmemc admin [OPTIONS] COMMAND [ARGS]...

Import bootstrap data and create showcase data.

This command group consists of commands for setting up and configuring eccenca Corporate Memory.

Options:

-h, --help Show this message and exit.

Commands:

bootstrap Update/Import bootstrap data.

showcase Create showcase data.

status Output health and version information.

#### 4.1.1 Command: admin bootstrap

Usage: cmemc admin bootstrap [OPTIONS]

Update/Import bootstrap data.

This command imports the bootstrap data needed for managing shapes, access conditions, the query catalog and the vocabulary catalog.

Note: There is currently no deletion mechanism for the bootstrap data, so you need to remove the graphs manually (or just remove all graphs).

Options:

--import Delete existing bootstrap data if present and import bootstrap data which was delivered

-h, --help Show this message and exit.

#### 4.1.2 Command: admin showcase

Usage: cmemc admin showcase [OPTIONS]

Create showcase data.

This command creates a showcase scenario of multiple graphs including integration graphs, shapes, statement annotations etc.

Note: There is currently no deletion mechanism for the showcase data, so you need to remove the showcase graphs manually (or just remove all graphs).

Options:

--scale INTEGER The scale factor provides a way to set the target size of the scenario. A value of 10 results in around 40k triples, a value of 50 in around 350k triples. [default: 10]

--create Delete old showcase data if present and create new showcase databased on the given scale factor.

--delete Delete existing showcase data if present.

-h, --help Show this message and exit.

#### 4.1.3 Command: admin status

Usage: cmemc admin status [OPTIONS]

Output health and version information.

This command outputs version and health information of the selected deployment. To get status information of all configured deployments use this command in combination with xargs:

```
cmemc config list | xargs -i cmemc -c {} admin status
```

Options:

-h, --help Show this message and exit.

## 4.2 Command group: config

Usage: cmemc config [OPTIONS] COMMAND [ARGS]...

List, edit and check configurations.

Configurations are identified by the section identifier in the config file. Each configuration represent a Corporate Memory deployment with its specific access method as well as credentials.

A minimal configuration which uses client credentials has the following entries:

```
[example.org]
CMEM_BASE_URI=https://cmem.example.org/
OAUTH_GRANT_TYPE=client_credentials
OAUTH_CLIENT_ID=cmem-service-account
OAUTH_CLIENT_SECRET=my-secret-account-pass
```

In addition to that, the following config parameters can be used as well:

```
SSL_VERIFY=False    - for ignoring certificate issues (not recommended)
DP_API_ENDPOINT=URL - to point to a non-standard DataPlatform location
DI_API_ENDPOINT=URL - to point to a non-standard DataIntegration location
OAUTH_TOKEN_URI=URL - to point to an external IdentityProvider location
OAUTH_USER=username - username (only if OAUTH_GRANT_TYPE=password)
OAUTH_PASSWORD=pass - password (only if OAUTH_GRANT_TYPE=password)
```

In order to request passwords on start, you can use the following parameter instead the PASSWORD parameter: OAUTH\_PASSWORD\_ENTRY=True  
OAUTH\_CLIENT\_SECRET\_ENTRY=True.

Options:

-h, --help Show this message and exit.

Commands:

```
check  Check the status of deployment.
edit   Edit the user-scope configuration file.
list   List configured CMEM connections.
```

### 4.2.1 Command: config check

Usage: cmemc config check [OPTIONS] [CONFIGS]...

Check the status of deployment.

Options:

-a, --all Check all configured connections.  
-h, --help Show this message and exit.

#### 4.2.2 Command: config edit

Usage: cmemc config edit [OPTIONS]

Edit the user-scope configuration file.

Options:

-h, --help Show this message and exit.

#### 4.2.3 Command: config list

Usage: cmemc config list [OPTIONS]

List configured CMEM connections.

Options:

-h, --help Show this message and exit.

### 4.3 Command group: dataset

Usage: cmemc dataset [OPTIONS] COMMAND [ARGS]...

List, create, delete, inspect datasets.

This command group allows for managing workspace datasets as well as dataset file resources. Datasets can be created and deleted. File resources can be uploaded and downloaded. Details of dataset parameter can be listed with inspect.

Datasets are identified with a combined key of the project ID and the project internal dataset ID (e.g: my-project:my-dataset). To get a list of datasets, use the list command.

Options:

```
-h, --help Show this message and exit.
```

Commands:

```
create    Create a dataset.
delete    Delete datasets.
download  Download the resource file of a dataset.
inspect   Display meta data of a dataset.
list      List available datasets.
upload    Upload a resource file to a dataset.
```

### 4.3.1 Command: dataset create

Usage: cmemc dataset create [OPTIONS] [DATASET\_FILE]

Create a dataset.

Datasets are created in projects and can have associated file resources. Each dataset has a type (such as 'csv') and a list of parameter which can change or specify the dataset behaviour.

To get more information on possible dataset types and parameter on these types, use the '--help-types' and '--help-parameter' options.

Example: cmemc dataset create --project my-project --type csv my-file.csv

Options:

|                                |   |
|--------------------------------|---|
| -t, --type TEXT                | The dataset type of the dataset to create. Example types are 'csv', 'json' and 'eccencaDataPlatform' (-> Knowledge Graph).  |
| --project TEXT                 | The project, where you want to create the dataset in. If there is only one project in the workspace, this option can be omitted.  |
| -p, --parameter <TEXT TEXT>... | A set of key/value pairs. Each dataset type has different parameters (such as charset, arraySeparator, ignoreBadLines, ...). In order to get a list of possible parameter, use the '--help-parameter' option. |
| --replace                      | Replace remote file resources in case there already exists a file with the same name.   |



|                               |   |
|-------------------------------|---|
| <code>--id TEXT</code>        | The dataset ID of the dataset to create. The dataset ID will be automatically created in case it is not present.                      |
| <code>--help-types</code>     | Lists all possible dataset types on given Corporate Memory instance. Note that this option already needs access to the instance.      |
| <code>--help-parameter</code> | Lists all possible (optional and mandatory) parameter for a dataset type. Note that this option already needs access to the instance. |
| <code>-h, --help</code>       | Show this message and exit.   |

### 4.3.2 Command: dataset delete

Usage: `cmemc dataset delete [OPTIONS] [DATASET_IDS]...`

Delete datasets.

This deletes existing datasets in integration projects from Corporate Memory. Datasets will be deleted without prompting! Dataset resources will not be deleted.

Example: `cmemc dataset delete my_project:my_dataset`

Datasets can be listed by using the '`cmemc dataset list`' command.

Options:

|                             |   |
|-----------------------------|---|
| <code>-a, --all</code>      | Delete all datasets. This is a dangerous option, so use it with care.   |
| <code>--project TEXT</code> | In combination with the ' <code>--all</code> ' flag, this option allows for deletion of all datasets of a certain project. The behaviour is similar to the ' <code>dataset list --project</code> ' command. |
| <code>-h, --help</code>     | Show this message and exit.   |

### 4.3.3 Command: dataset download

Usage: `cmemc dataset download [OPTIONS] DATASET_ID OUTPUT_PATH`

Download the resource file of a dataset.

This command downloads the file resource of a dataset to your local file system or to standard out (-). Note that this is not possible for dataset types such as Knowledge Graph (eccencaDataplatform) or SQL endpoint (sqlEndpoint).

Without providing an output path, the output file name will be the same as the remote file resource.

Datasets can be listed by using the 'cmemc dataset list' command.

Options:

- replace Replace existing files. This is a dangerous option, so use it with care.
- h, --help Show this message and exit.

#### 4.3.4 Command: dataset inspect

Usage: cmemc dataset inspect [OPTIONS] DATASET\_ID

Display meta data of a dataset.

Options:

- raw Outputs raw JSON.
- h, --help Show this message and exit.

#### 4.3.5 Command: dataset list

Usage: cmemc dataset list [OPTIONS]

List available datasets.

Outputs a list of datasets IDs which can be used as reference for the dataset create and delete commands.

Options:

- project TEXT The project, from which you want to list the datasets.  
Project IDs can be listed with the 'project list' command.
- raw Outputs raw JSON.

```
--id-only      Lists only dataset identifier and no labels or other meta
                data. This is useful for piping the ids into other cmemc
                commands.

-h, --help      Show this message and exit.
```

#### 4.3.6 Command: dataset upload

Usage: `cmemc dataset upload [OPTIONS] DATASET_ID INPUT_PATH`

Upload a resource file to a dataset.

This command uploads a file to a dataset. The content of the uploaded file replaces the remote file resource. The name of the remote file resource is not changed.

Warning: If the remote file resource is used in more than one dataset, the other datasets are also affected by this command.

Warning: The content of the uploaded file is not tested, so uploading a json file to an xml dataset will result in errors.

Datasets can be listed by using the 'cmemc dataset list' command.

Example: `cmemc dataset upload cmem:my-dataset new-file.csv`

Options:

```
-h, --help  Show this message and exit.
```

### 4.4 Command group: graph

Usage: `cmemc graph [OPTIONS] COMMAND [ARGS]...`

List, import, export, delete or open graphs.

Graphs are identified by an IRI. The get a list of existing graphs, execute the list command or use tab-completion.

Options:

```
-h, --help  Show this message and exit.
```

Commands:

```
count    Count triples in graph(s).
delete   Delete graph(s) from the store.
export   Export graph(s) as NTriples to stdout (-), file or directory.
import   Import graph(s) to the store.
list     List accessible graphs.
open     Open / explore a graph in the browser.
```

#### 4.4.1 Command: graph count

Usage: cmemc graph count [OPTIONS] [IRIS]...

Count triples in graph(s).

This command lists graphs with their triple count. Counts are done without following imported graphs.

Options:

```
-a, --all          Count all graphs
-s, --summarize    Display only a sum of all counted graphs together
-h, --help        Show this message and exit.
```

#### 4.4.2 Command: graph delete

Usage: cmemc graph delete [OPTIONS] [IRIS]...

Delete graph(s) from the store.

Options:

```
-a, --all    Drop all (writeable) graphs
-h, --help   Show this message and exit.
```

#### 4.4.3 Command: graph export

Usage: cmemc graph export [OPTIONS] [IRIS]...

Export graph(s) as NTriples to stdout (-), file or directory.

In case of file export, data from all selected graphs will be concatenated in one file. In case of directory export, .graph and .nt files will be created for each graph.

**Options:**

|   |  |
|---|--|
| <code>-a, --all</code>                    | Export all (readable) graphs   |
| <code>--output-dir DIRECTORY</code>       | Export to this directory   |
| <code>--output-file FILE</code>           | Export to this file [default: -]   |
| <code>-t, --filename-template TEXT</code> | Template for the export file name(s). Used together with <code>--output-dir</code> . Possible placeholders are (Jinja2): <code>{{hash}}</code> - sha256 hash of the graph IRI, <code>{{iriname}}</code> - graph IRI converted to filename, <code>{{connection}}</code> - from the <code>--connection</code> option and <code>{{date}}</code> - the current date as YYYY-MM-DD. The file suffix will be appended. Needed directories will be created. [default: <code>{{hash}}</code> ] |
| <code>-h, --help</code>                   | Show this message and exit.  |

#### 4.4.4 Command: graph import

Usage: `cmemc graph import [OPTIONS] INPUT_PATH [IRI]`

Import graph(s) to the store.

If input is an directory, it scans for file-pairs such as `xxx.ttl` and `xxx.ttl.graph` where `xxx.ttl` is the actual triples file and `xxx.ttl.graph` contains the graph IRI as one string: `"https://mygraph.de/xxx/"`. If input is a file, content will be uploaded to IRI. If `--replace` is set, the data will be overwritten, if not, it will be added.

**Options:**

|                         |  |
|-------------------------|--|
| <code>--replace</code>  | Replace (overwrite) original graph data. |
| <code>-h, --help</code> | Show this message and exit.              |

#### 4.4.5 Command: graph list

Usage: `cmemc graph list [OPTIONS]`

List accessible graphs.

**Options:**

|   |   |
|---|---|
| <code>--raw</code>                        | Outputs raw JSON.                       |
| <code>--filter [readonly writable]</code> | Filter list based on access conditions. |
| <code>-h, --help</code>                   | Show this message and exit.             |

#### 4.4.6 Command: graph open

Usage: cmemc graph open [OPTIONS] IRI

Open / explore a graph in the browser.

Options:

-h, --help Show this message and exit.

### 4.5 Command group: project

Usage: cmemc project [OPTIONS] COMMAND [ARGS]...

List, import, export, create or delete projects.

Projects are identified by an PROJECTID. The get a list of existing projects, execute the list command or use tab-completion.

Options:

-h, --help Show this message and exit.

Commands:

create Create empty new project(s).

delete Delete project(s).

export Export project(s) to file(s).

import Import a project from a file.

list List available projects.

#### 4.5.1 Command: project create

Usage: cmemc project create [OPTIONS] PROJECT\_IDS...

Create empty new project(s).

This creates one or more new projects. Existing projects will not be overwritten.

Example: cmemc project create my\_project

Projects can be listed by using the 'cmemc project list' command.

Options:

`-h, --help` Show this message and exit.

### 4.5.2 Command: project delete

Usage: `cmemc project delete [OPTIONS] [PROJECT_IDS]...`

Delete project(s).

This deletes existing data integration projects from Corporate Memory.  
Projects will be deleted without prompting!

Example: `cmemc project delete my_project`

Projects can be listed by using the '`cmemc project list`' command.

Options:

`-a, --all` Delete all projects. This is a dangerous option, so use it with care.

`-h, --help` Show this message and exit.

### 4.5.3 Command: project export

Usage: `cmemc project export [OPTIONS] [PROJECT_IDS]...`

Export project(s) to file(s).

Projects can be exported with different export formats. The default type is a zip archive which includes meta data as well as dataset resources. If more than one project is exported, a file is created for each project. By default, these files are created in the current directory and with a descriptive name (see `--template` option default).

Example: `cmemc project export my_project`

Available projects can be listed by using the '`cmemc project list`' command.

You can use the template string to create subdirectories as well: `cmemc config list | parallel -I% cmemc -c % project export --all -t "dump/{{connection}}/{{date}}-{{id}}.project"`

**Options:**

|   |   |
|---|---|
| <code>-a, --all</code>                    | Export all projects.  |
| <code>-o, --overwrite</code>              | Overwrite existing files. This is a dangerous option, so use it with care.  |
| <code>--output-dir DIRECTORY</code>       | Directory, where the project files will be created. This directory will be created as well it does not exists. [default: .]   |
| <code>--type TEXT</code>                  | Type of the exported project file(s). [default: xmlZip]   |
| <code>-t, --filename-template TEXT</code> | Template for the export file name(s). Possible placeholders are (Jinja2): <code>{{id}}</code> (the project ID), <code>{{connection}}</code> (from the <code>--connection</code> option) and <code>{{date}}</code> (the current date as YYYY-MM-DD). The file suffix will be appended. Needed directories will be created. [default: <code>{{date}}-{{connection}}-{{id}}.project</code> ] |
| <code>-h, --help</code>                   | Show this message and exit.   |

#### 4.5.4 Command: project import

Usage: `cmemc project import [OPTIONS] FILE PROJECT_ID`

Import a project from a file.

Example: `cmemc project import my_project.zip my_project`

**Options:**

`-h, --help` Show this message and exit.

#### 4.5.5 Command: project list

Usage: `cmemc project list [OPTIONS]`

List available projects.

Outputs a list of project IDs which can be used as reference for the project create, delete, export and import commands.



**Options:**

-h, --help Show this message and exit.

## 4.6 Command group: query

Usage: cmemc query [OPTIONS] COMMAND [ARGS]...

List, execute or open local and remote SPARQL queries.

Queries are identified either by a file path, an URI from the query catalog or a shortened URI (qname, using a default namespace).

In order to get a list of queries from the query catalog, use the list command. One or more queries can be executed one after the other with the execute command. With open command you can jump to the query editor in your browser.

Queries can use a mustache like syntax to specify placeholder for parameter values (e.g. {{resourceUri}}). These parameter values need to be given as well, before the query can be executed (use the -p option).

**Options:**

-h, --help Show this message and exit.

**Commands:**

execute Execute queries which are loaded from files or the query catalog.  
list List available queries from the catalog.  
open Open queries in the editor of the query catalog in your browser.

### 4.6.1 Command: query execute

Usage: cmemc query execute [OPTIONS] QUERIES...

Execute queries which are loaded from files or the query catalog.

Queries are identified either by a file path, an URI from the query catalog, or a shortened URI (qname, using a default namespace).

If multiple queries are executed one after the other, the first failing query stops the whole execution chain.

Limitations: All optional parameters (e.g. accept, base64, ...) are

provided for ALL queries in an execution chain. If you need different parameters for each query in a chain, run cmemc multiple times and use the logical operators && and || of your shell instead.

#### Options:

|   |  |
|---|--|
| <code>--accept TEXT</code>                        | Accept header for the HTTP request(s).<br>Setting this to 'default' means that cmemc uses an appropriate accept header for terminal output (text/csv for tables, text/turtle for graphs, * otherwise). Please refer to the Corporate Memory system manual for a list of accepted mime types.<br>[default: default] |
| <code>--no-imports</code>                         | Graphs which include other graphs (using owl:imports) will be queried as merged overall-graph. This flag disables this default behaviour. The flag has no effect on update queries.  |
| <code>--base64</code>                             | Enables base64 encoding of the query parameter for the SPARQL requests (the response is not touched). This can be useful in case there is an aggressive firewall between cmemc and Corporate Memory.   |
| <code>-p, --parameter &lt;TEXT TEXT&gt;...</code> | In case of a parameterized query (placeholders with the '{{key}}' syntax), this option fills all placeholder with a given value before the query is executed. Pairs of placeholder/value need to be given as a tuple 'KEY VALUE'. A key can be used only once.   |
| <code>--limit INTEGER</code>                      | Override or set the LIMIT in the executed SELECT query. Note that this option will never give you more results than the LIMIT given in the query itself.   |
| <code>--offset INTEGER</code>                     | Override or set the OFFSET in the executed SELECT query.   |
| <code>--distinct</code>                           | Override the SELECT query by make the result set DISTINCT.   |

|                                |  |
|--------------------------------|--|
| <code>--timeout INTEGER</code> | Set max execution time for query evaluation (in milliseconds). |
| <code>-h, --help</code>        | Show this message and exit.                                    |

#### 4.6.2 Command: query list

Usage: `cmemc query list [OPTIONS]`

List available queries from the catalog.

Outputs a list of query URIs which can be used as reference for the query execute command.

Options:

`--id-only` Lists only query identifier and no labels or other meta data.  
This is useful for piping the ids into other cmemc commands.

`-h, --help` Show this message and exit.

#### 4.6.3 Command: query open

Usage: `cmemc query open [OPTIONS] QUERIES...`

Open queries in the editor of the query catalog in your browser.

With this command, you can open (remote) queries from the query catalog in the query editor in your browser (e.g. in order to change them). You can also load local query files into the query editor, in order to import them into the query catalog.

The command accepts multiple query URIs or files which results in opening multiple browser tabs.

Options:

`-h, --help` Show this message and exit.

### 4.7 Command group: vocabulary

```
Usage: cmemc vocabulary [OPTIONS] COMMAND [ARGS]...
```

List, install, uninstall and open vocabularies.

Options:

-h, --help Show this message and exit.

Commands:

```
cache      List und update the vocabulary cache.
install    Install one or more vocabularies from the catalog.
list       Output a list of vocabularies.
open       Open / explore a vocabulary graph in the browser.
uninstall  Uninstall one or more vocabularies.
```

#### 4.7.1 Command: vocabulary cache

```
Usage: cmemc vocabulary cache [OPTIONS] COMMAND [ARGS]...
```

List und update the vocabulary cache.

Options:

-h, --help Show this message and exit.

Commands:

```
list      Output the content of the global vocabulary cache.
update    Reload / updates the data integration cache for a vocabulary.
```

#### 4.7.2 Command: vocabulary cache

```
Usage: cmemc vocabulary cache [OPTIONS] COMMAND [ARGS]...
```

List und update the vocabulary cache.

Options:

-h, --help Show this message and exit.

Commands:

```
list      Output the content of the global vocabulary cache.
update    Reload / updates the data integration cache for a vocabulary.
```

### 4.7.3 Command: vocabulary install

Usage: `cmemc vocabulary install [OPTIONS] [IRIS]...`

Install one or more vocabularies from the catalog.

Vocabularies are identified by their graph IRI. Installable vocabularies can be listed with the "`vocabulary list --filter installable`" command.

Options:

- a, --all Install all vocabularies from the catalog.
- h, --help Show this message and exit.

### 4.7.4 Command: vocabulary list

Usage: `cmemc vocabulary list [OPTIONS]`

Output a list of vocabularies.

Vocabularies are graphs (see '`cmemc graph`' command group) which consists of class and property descriptions.

Options:

- id-only Lists only vocabulary identifier (IRIs) and no labels or other meta data. This is useful for piping the ids into other `cmemc` commands.
- filter [all|installed|installable] Filter list based on status. [default: installed]
- raw Outputs raw JSON.
- h, --help Show this message and exit.

### 4.7.5 Command: vocabulary open

Usage: `cmemc vocabulary open [OPTIONS] IRI`

Open / explore a vocabulary graph in the browser.

Vocabularies are identified by their graph IRI. Installed vocabularies can

be listed with the "vocabulary list" command.

Options:

-h, --help Show this message and exit.

#### 4.7.6 Command: vocabulary uninstall

Usage: cmemc vocabulary uninstall [OPTIONS] [IRIS]...

Uninstall one or more vocabularies.

Vocabularies are identified by their graph IRI. Already installed vocabularies can be listed with the "vocabulary list --filter installed" command.

Options:

-a, --all Uninstall all installed vocabularies.

-h, --help Show this message and exit.

### 4.8 Command group: workflow

Usage: cmemc workflow [OPTIONS] COMMAND [ARGS]...

List, execute, open or inspect workflows.

Workflows are identified by a WORKFLOW\_ID. To get a list of existing workflows, execute the list command or use tab-completion. The WORKFLOW\_ID is a concatenation of an PROJECT\_ID and a TASK\_ID, such as "my-project:my-workflow".

Options:

-h, --help Show this message and exit.

Commands:

execute Execute workflow(s).

io Execute a workflow with file input/output.

list List available workflow ids.

open Open a workflow in your browser.

status Get status information of workflow(s).

### 4.8.1 Command: workflow execute

Usage: cmemc workflow execute [OPTIONS] [WORKFLOW\_IDS]...

Execute workflow(s).

With this command, you can start one or more workflows at the same time or in a sequence, depending on the result of the predecessor.

Executing a workflow can be done in two ways: Without `--wait` just sends the starting signal and does not look for the workflow and its result (fire and forget). Starting workflows in this way, starts all given workflows at the same time.

The optional `--wait` option starts the workflows in the same way, but also polls the status of a workflow until it is finished. In case of an error of a workflow, the next workflow is not started.

Options:

|   |   |
|---|---|
| <code>-a, --all</code>                        | Execute all available workflows.  |
| <code>--wait</code>                           | Wait for one workflow to complete.  |
| <code>--polling-interval INTEGER RANGE</code> | How many seconds to wait between status polls. Status polls are cheap, so a higher polling interval is most likely not needed. [default: 1] |
| <code>-h, --help</code>                       | Show this message and exit.   |

### 4.8.2 Command: workflow io

Usage: cmemc workflow io [OPTIONS] WORKFLOW\_ID

Execute a workflow with file input/output.

With this command, you can execute a workflow that uses variable datasets as input, output or for configuration. Use the input parameter to feed data into the workflow. Likewise use output for retrieval of the workflow result. Workflows without a variable dataset will throw an error.

Options:

|                               |  |
|-------------------------------|--|
| <code>-i, --input FILE</code> | From which file the input is taken: note that the maximum file size to upload is |
|-------------------------------|--|





**Options:**

-h, --help Show this message and exit.

#### 4.8.5 Command: workflow status

Usage: cmemc workflow status [OPTIONS] [WORKFLOW\_IDS]...

Get status information of workflow(s).

**Options:**

|   |  |
|---|--|
| --project TEXT  | The project, from which you want to list the workflows. Project IDs can be listed with the 'project list' command. |
| --raw   | Output raw JSON info.  |
| --filter [Idle Not executed Finished Cancelled Failed Successful Canceling Running Waiting] | Show only workflows of a specific status.  |
| -h, --help  | Show this message and exit.  |

#### 4.9 Command group: workspace

Usage: cmemc workspace [OPTIONS] COMMAND [ARGS]...

Import or export the workspace.

**Options:**

-h, --help Show this message and exit.

**Commands:**

|        |   |
|--------|---|
| export | Export the complete workspace (all projects) to a ZIP file. |
| import | Import the workspace from a file.                           |
| reload | Reload the workspace from the backend.                      |

##### 4.9.1 Command: workspace export

Usage: cmemc workspace export [OPTIONS] [FILE]

Export the complete workspace (all projects) to a ZIP file.

Depending on the requested type, this ZIP contains either a turtle file

for each project (type `rdfTurtle`) or a substructure of resource files and XML descriptions (type `xmlZip`).

The file name is optional and will be generated with by the template if absent.

Options:

|   |   |
|---|---|
| <code>-o, --overwrite</code>              | Overwrite existing files. This is a dangerous option, so use it with care.  |
| <code>--type TEXT</code>                  | Type of the exported workspace file.<br>[default: <code>xmlZip</code> ]   |
| <code>-t, --filename-template TEXT</code> | Template for the export file name. Possible placeholders are (Jinja2): <code>{{connection}}</code> (from the <code>--connection</code> option) and <code>{{date}}</code> (the current date as <code>YYYY-MM-DD</code> ). The file suffix will be appended. Needed directories will be created. [default: <code>{{date}}-{{connection}}.workspace</code> ] |
| <code>-h, --help</code>                   | Show this message and exit.   |

#### 4.9.2 Command: workspace import

Usage: `cmemc workspace import [OPTIONS] FILE`

Import the workspace from a file.

Options:

|                          |  |
|--------------------------|--|
| <code>--type TEXT</code> | Type of the exported workspace file. [default: <code>xmlZip</code> ] |
| <code>-h, --help</code>  | Show this message and exit.  |

#### 4.9.3 Command: workspace reload

Usage: `cmemc workspace reload [OPTIONS]`

Reload the workspace from the backend.

Options:

|                         |                             |
|-------------------------|-----------------------------|
| <code>-h, --help</code> | Show this message and exit. |
|-------------------------|-----------------------------|