DIRECTIONS 4 PARTNERS

RUNNING SQL IN A CONTAINER AND CONNECTING BUSINESS CENTRAL TO IT

Tobias Fenster, COSMO CONSULT

www.directions4partners.com

Tobias Fenster

CTO at COSMO CONSULT Group

Dual Microsoft MVP for Business Applications and Azure

Microsoft Regional Director

@tobiasfenster

tobiasfenster.io

tobiasfenster

COSMO CONSULT

Business-Software for People

Running SQL in a container

Why

- Same as for most containerizations
  - Easy to “install” and update
  - Easy to separate from other workloads (file system and RAM/CPU)

- Business Central is already containerized
  - Included SQL is Express Edition → max 10G database size
  - Dedicated SQL per Business Central instance unnecessary overhead

Running SQL in a container

How

```
docker run
        -p 1433:1433
        -e accept_eula=y
        -e sa_password=Super5ecret!
        tobiasfenster/mssql-server-dev-unsupported:2019-cu13
```

Running SQL in a container

How

```
docker run
        -p 1433:1433
        -e accept_eula=y
        -e sa_password=Super5ecret!
        tobiasfenster/mssql-server-dev-unsupported:2019-cu13
```

Make available as localhost:1433 - optional

Running SQL in a container

How

```
docker run
        -p 1433:1433
        -e accept_eula=y
        -e sa_password=Super5ecret!
        tobiasfenster/mssql-server-dev-unsupported:2019-cu13
```

Accept End User License Agreement – mandatory

Running SQL in a container

How

```
docker run
       -p 1433:1433
       -e accept_eula=y
       -e sa_password=Super5ecret!
       tobiasfenster/mssql-server-dev-unsupported:2019-cu13
```

Set admin password – mandatory

Running SQL in a container

How

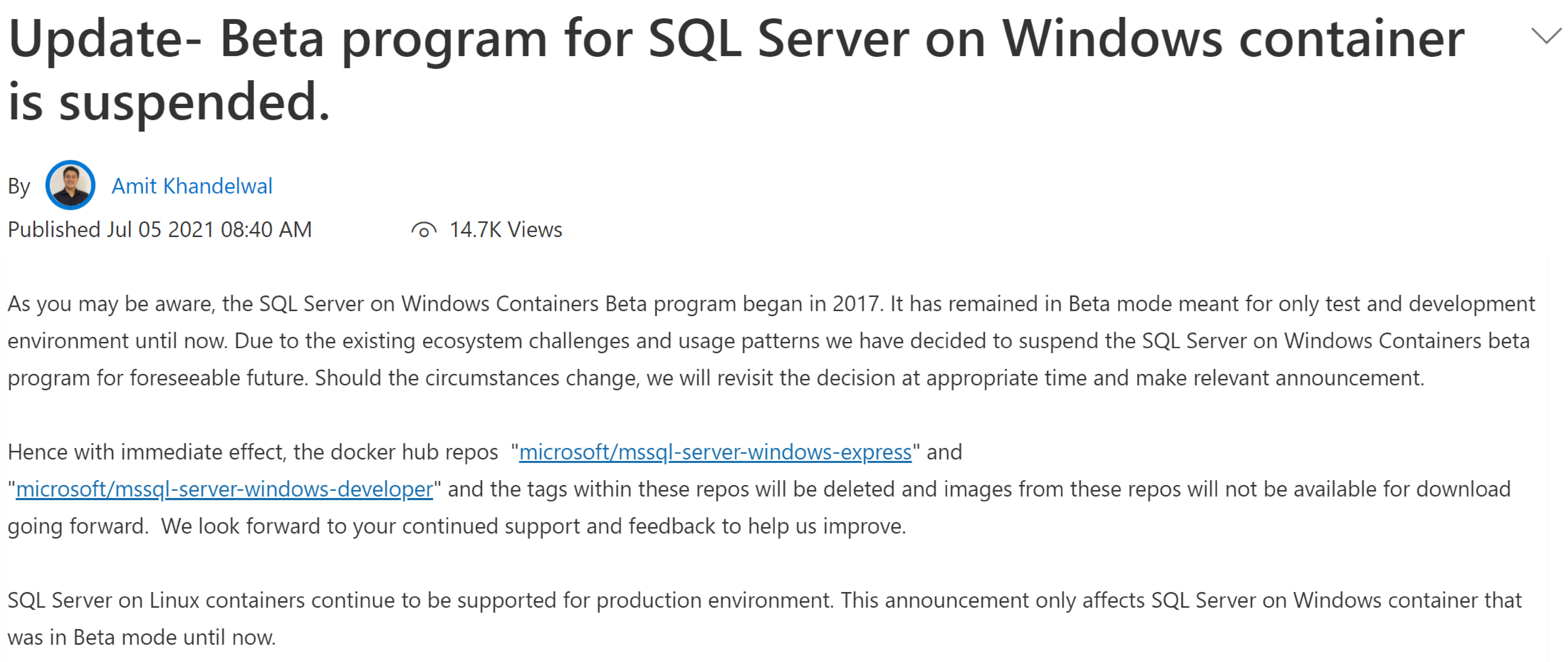
```
docker run
        -p 1433:1433
        -e accept_eula=y
        -e sa_password=Super5ecret!
        tobiasfenster/mssql-server-dev-unsupported:2019-cu13
```

SQL Server image name – mandatory

Why not something like mcr.microsoft.com/mssql/server:2019-cu13?

Running SQL in a container

Which image to use



Update- Beta program for SQL Server on Windows container is suspended.

By Amit Khandelwal

Published Jul 05 2021 08:40 AM 14.7K Views

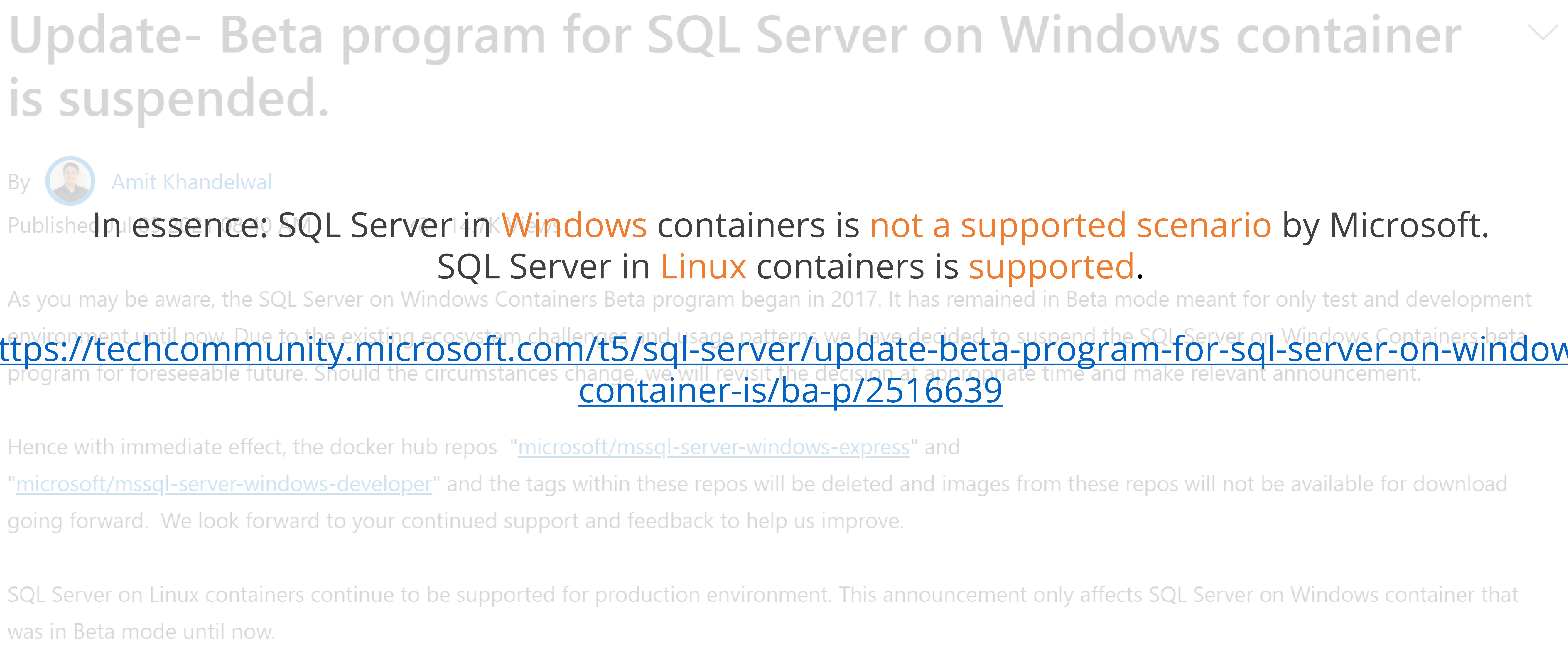
As you may be aware, the SQL Server on Windows Containers Beta program began in 2017. It has remained in Beta mode meant for only test and development environment until now. Due to the existing ecosystem challenges and usage patterns we have decided to suspend the SQL Server on Windows Containers beta program for foreseeable future. Should the circumstances change, we will revisit the decision at appropriate time and make relevant announcement.

Hence with immediate effect, the docker hub repos "microsoft/mssql-server-windows-express" and "microsoft/mssql-server-windows-developer" and the tags within these repos will be deleted and images from these repos will not be available for download going forward. We look forward to your continued support and feedback to help us improve.

SQL Server on Linux containers continue to be supported for production environment. This announcement only affects SQL Server on Windows container that was in Beta mode until now.

Running SQL in a container

Which image to use



Update- Beta program for SQL Server on Windows container is suspended.

By Amit Khandelwal

Published Jul 05 2020 08:40 AM 14.7K Views

As you may be aware, the SQL Server on Windows Containers Beta program began in 2017. It has remained in Beta mode meant for only test and development environment until now. Due to the existing ecosystem challenges and usage patterns we have decided to suspend the SQL Server on Windows Containers beta program for foreseeable future. Should the circumstances change, we will revisit the decision at appropriate time and make relevant announcement.

Hence with immediate effect, the docker hub repos "microsoft/mssql-server-windows-express" and "microsoft/mssql-server-windows-developer" and the tags within these repos will be deleted and images from these repos will not be available for download going forward. We look forward to your continued support and feedback to help us improve.

SQL Server on Linux containers continue to be supported for production environment. This announcement only affects SQL Server on Windows container that was in Beta mode until now.

In essence: SQL Server in Windows containers is not a supported scenario by Microsoft.
SQL Server in Linux containers is supported.

https://techcommunity.microsoft.com/t5/sql-server/update-beta-program-for-sql-server-on-windows-container-is/ba-p/2516639

Running SQL in a container

Which image to use

- Fortunately, actually not that complicated to create the image
  - `tobiasfenster/mssql-server-dev-unsupported:2019-cu13`
  - `tobiasfenster/mssql-server-exp-unsupported:2019-cu13`
- Be aware: No support, no affiliation with Microsoft at all, no guarantee that it works – but it does 😜
- SQL Server 2019 CU 11 and newer
- Windows Server 2019 LTSC (1809), 2004 and 20H2 as multi-arch image, Server 2022 will probably follow

Running SQL in a container

Which image to use

- Base structure:
  - Built on .NET Framework 4.8 to avoid prereq installs
  - Install Developer Edition (from ISO) or Express Edition (from installer) and configure
  - Install CU
  - Run PowerShell script
- github.com/tfenster/mssql-image for the sources
- hub.docker.com/r/tobiasfenster/mssql-server-dev-unsupported for the images
- tobiasfenster.io/ms-sql-server-in-windows-containers for an explanation

Running SQL in a container

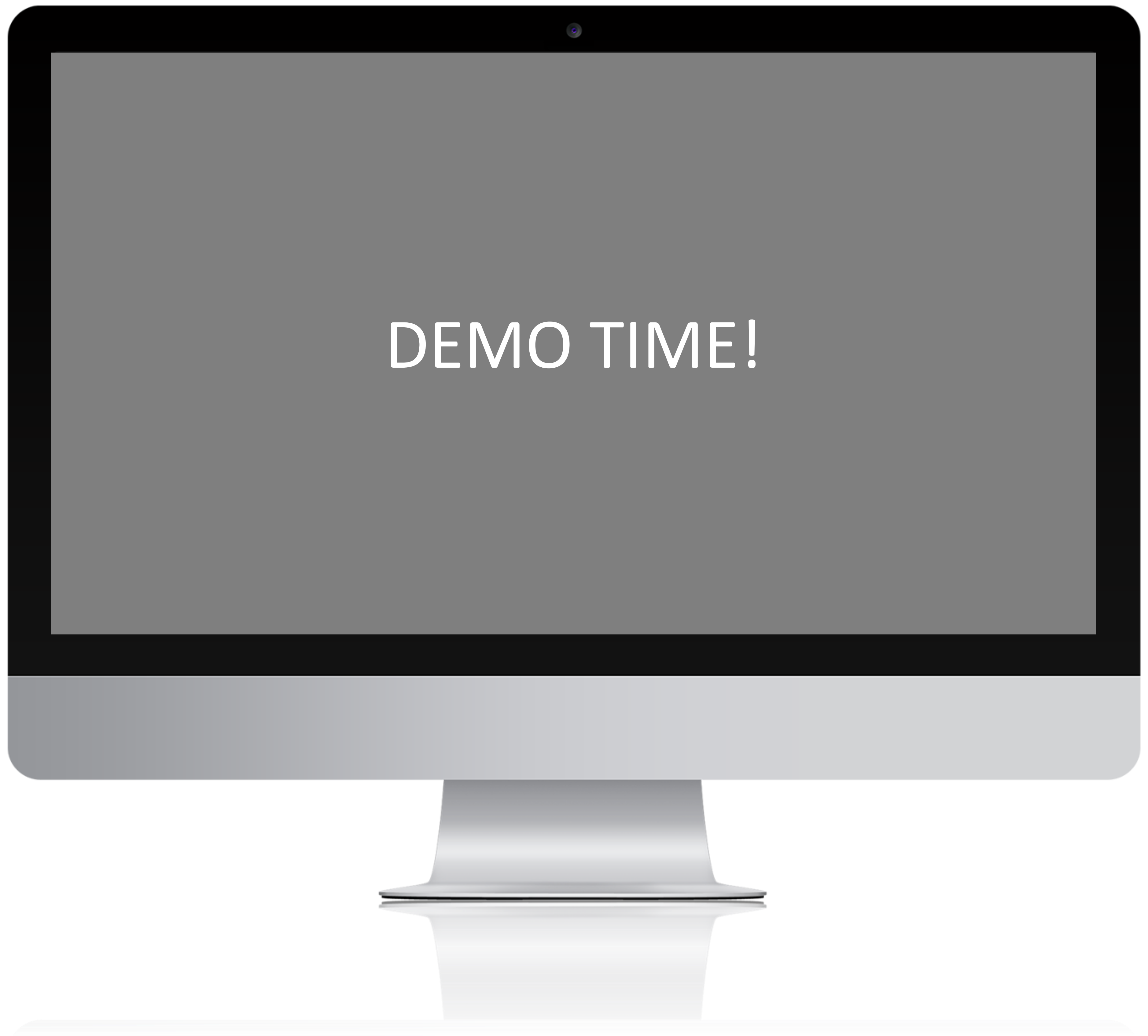
Demo

Run

Run the SQL container

Connect

Connect Azure Data Studio



Restore

Restore a database

Upgrade

Go to the next SQL CU

Connecting Business Central

Database already in place

- Same image as usually, but included SQL Server isn‘t used
- Almost same command, just add SQL connection information
- Make sure database and image version match

Typical BC container start

```
New-BcContainer
        -accept_eula -accept_outdated
        -containerName bc19 -imageName mybc:onprem-19.0.29894.30693-w1
        -Credential (Get-Credential -Message "bc credential")
        -auth NavUserPassword
        -databaseServer localhost -databaseName Cronus19w1
        -databaseCredential (Get-Credential -Message "database credential")
```

Connecting Business Central

Database already in place

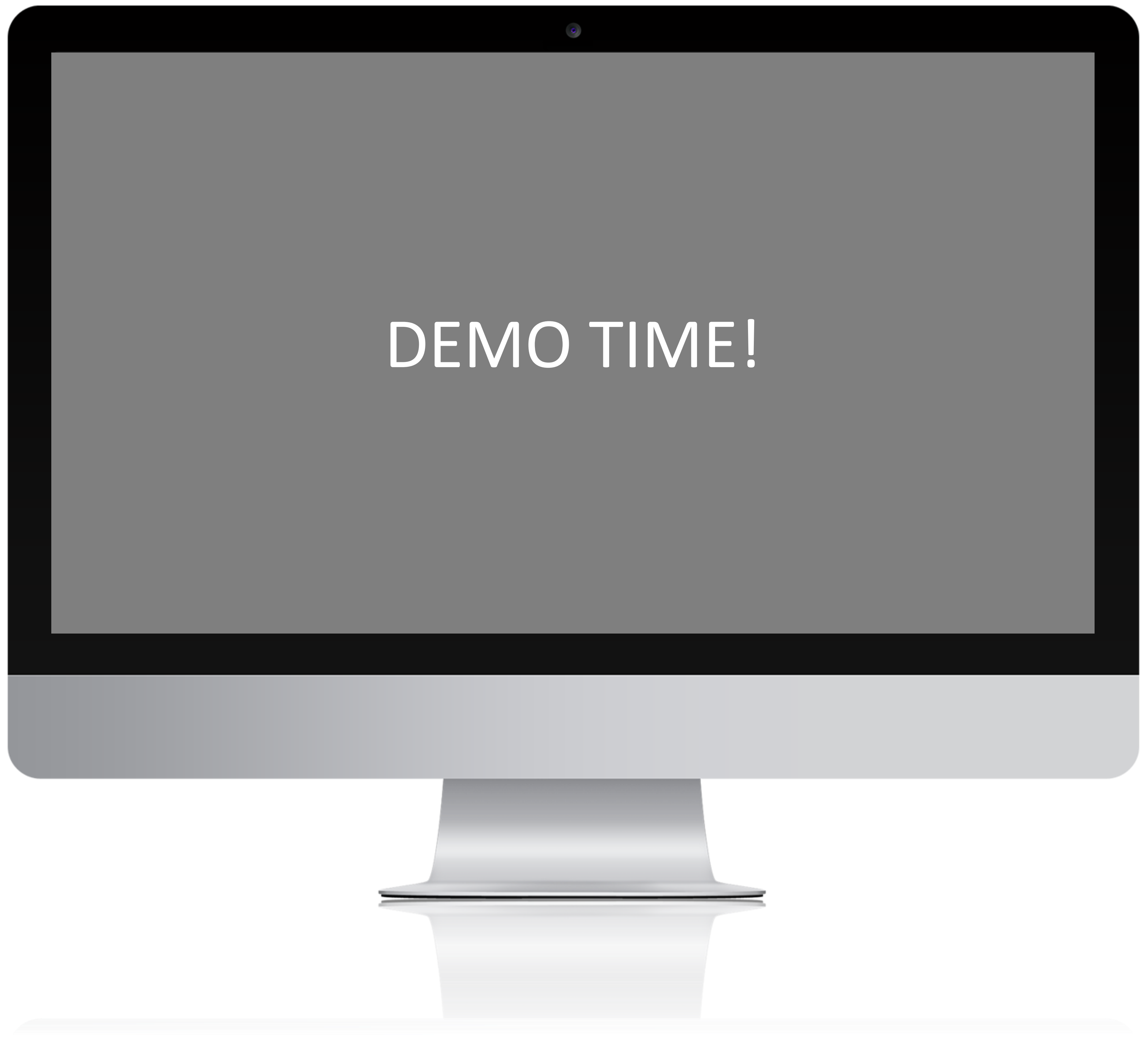
- Same image as usually, but included SQL Server isn‘t used
- Almost same command, just add SQL connection information
- Make sure database and image version match

```
New-BcContainer
        -accept_eula -accept_outdated
        -containerName bc19 -imageName mybc:onprem-19.0.29894.30693-w1
        -Credential (Get-Credential -Message "bc credential")
        -auth NavUserPassword
        -databaseServer localhost -databaseName Cronus19w1
        -databaseCredential (Get-Credential -Message "database credential")
```

Database connection information

Connecting Business Central

Database already in place



Connecting Business Central

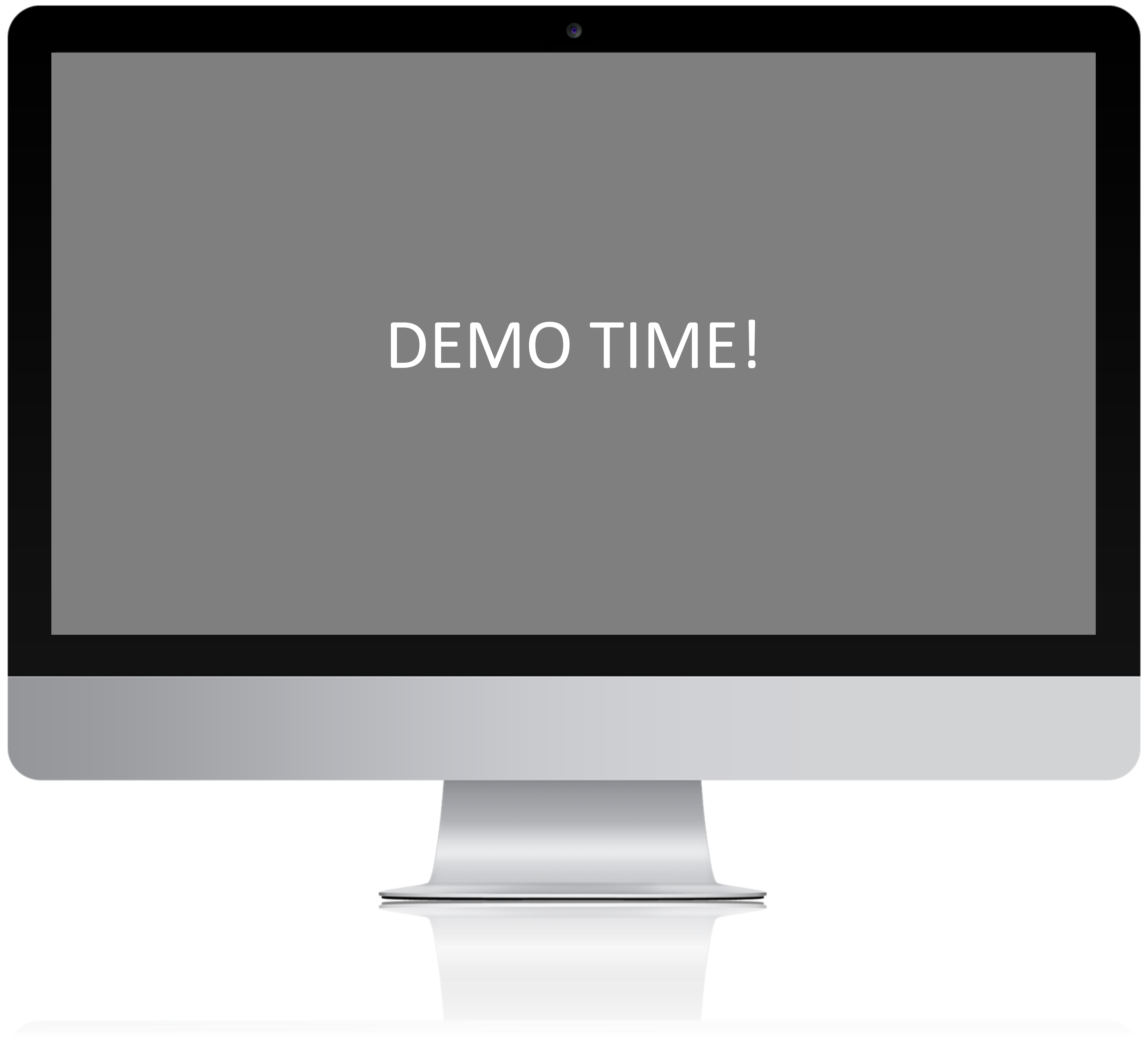
Bring it together

- Declare the SQL Server container and databases together with the BC containers
- Easily make changes and track them through version control
- Start and stop with a single command: docker compose

```
services:
  sql:
    image: tobiasfenster/mssql-server-dev-unsupported:2019-cu13
    …
  bc19:
    image: mybc:onprem-19.0.29894.30693-w1
    …
```

Connecting Business Central

Bring it together



Connecting Business Central

Create database

- Manually, as already seen or via cmdlet in bccontainerhelper (of course… 😊)
- Windows authentication only!

BC artifact to get the database backup from

```
Restore-BcDatabaseFromArtifacts
       -artifactUrl (Get-BCArtifactUrl -type OnPrem -version 19.0 -country w1)
       -databaseServer localhost -databasePrefix cronus -databaseName 19w1
```

- After that: Create container as before

Connecting Business Central

Create database

- Manually, as already seen or via cmdlet in bccontainerhelper (of course… 😊)
- Windows authentication only!

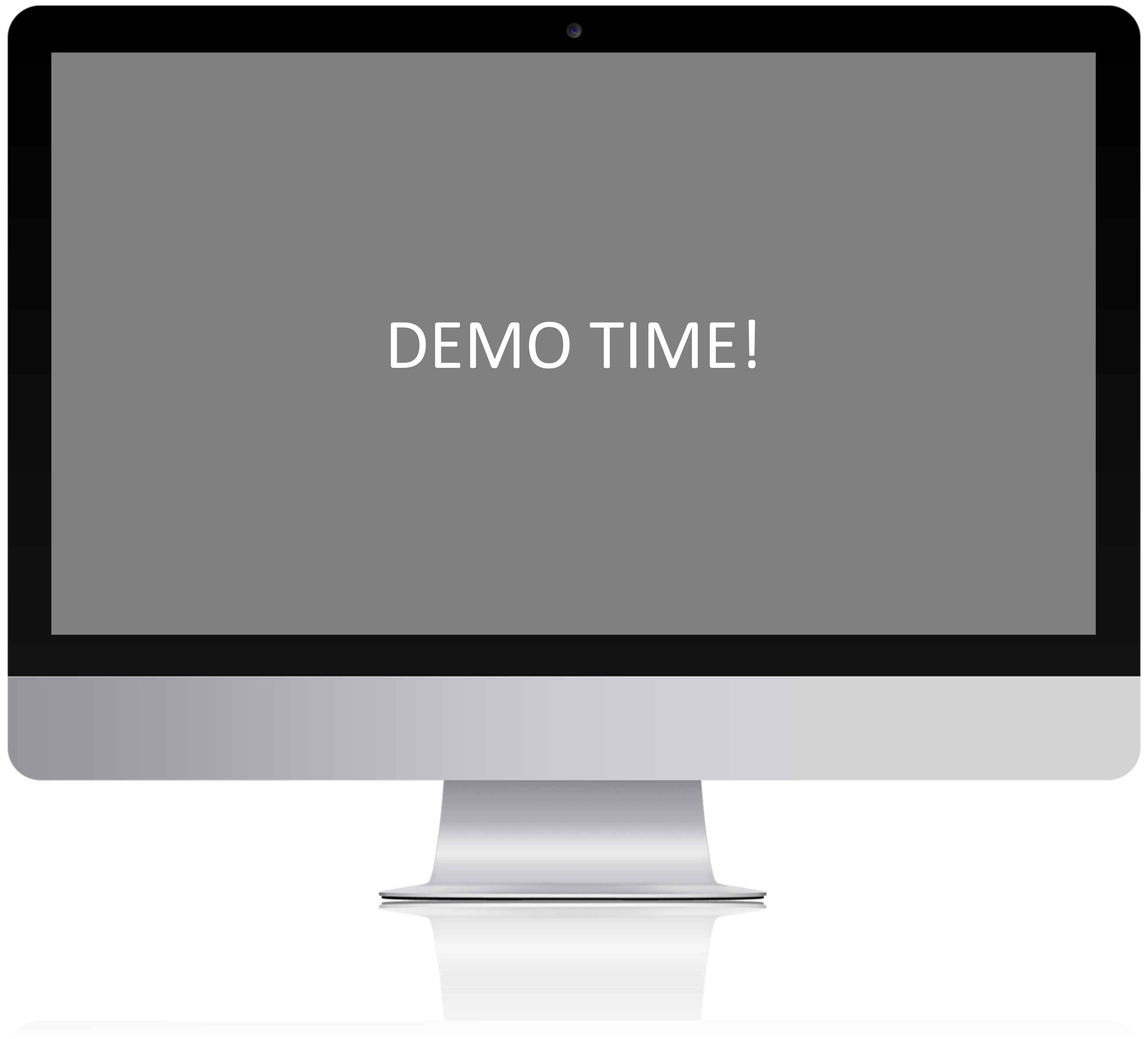
```
Restore-BcDatabaseFromArtifacts
        -artifactUrl (Get-BCArtifactUrl -type OnPrem -version 19.0 -country w1)
        -databaseServer localhost -databasePrefix cronus -databaseName 19w1
```

Database connection information

- After that: Create container as before

Connecting Business Central

Create database



Connecting Business Central

Do it all in one

- Scenario: SQL is in place, you want to use it on the fly when you generate the container

Typical BC container start

```
New-BcContainer
        -accept_eula -accept_outdated -PublishPorts 80
        -containerName bc19 -imageName mybc:onprem-19.0.29894.30693-w1
        -Credential $credentialBC -auth NavUserPassword
        -databaseServer 172.19.112.63 -databasePrefix cronus
        -databaseName 19w1 -databaseCredential $credentialSQL
        -artifactUrl (Get-BCArtifactUrl -type OnPrem -version 19.0 -country w1)
        -replaceExternalDatabases
        -licenseFile "c:\bcartifacts.cache\onprem\19…\w1\database\Cronus.flf"
```

Connecting Business Central

Do it all in one

- Scenario: SQL is in place, you want to use it on the fly when you generate the container

Database information

```
New-BcContainer
        -accept_eula -accept_outdated -PublishPorts 80
        -containerName bc19 -imageName mybc:onprem-19.0.29894.30693-w1
        -Credential $credentialBC -auth NavUserPassword
        -databaseServer 172.19.112.63 -databasePrefix cronus
        -databaseName 19w1 -databaseCredential $credentialSQL
        -artifactUrl (Get-BCArtifactUrl -type OnPrem -version 19.0 -country w1)
        -replaceExternalDatabases
        -licenseFile "c:\bcartifacts.cache\onprem\19…\w1\database\Cronus.flf"
```

Connecting Business Central

Do it all in one

- Scenario: SQL is in place, you want to use it on the fly when you generate the container

```
New-BcContainer
        -accept_eula -accept_outdated -PublishPorts 80
        -containerName bc19 -imageName mybc:onprem-19.0.29894.30693-w1
        -Credential $credentialBC -auth NavUserPassword
        -databaseServer 172.19.112.63 -databasePrefix cronus
        -databaseName 19w1 -databaseCredential $credentialSQL
        -artifactUrl (Get-BCArtifactUrl -type OnPrem -version 19.0 -country w1)
        -replaceExternalDatabases
        -licenseFile "c:\bcartifacts.cache\onprem\19…\w1\database\Cronus.flf"
```

Trigger to create and potentially replace database

Connecting Business Central

Do it all in one

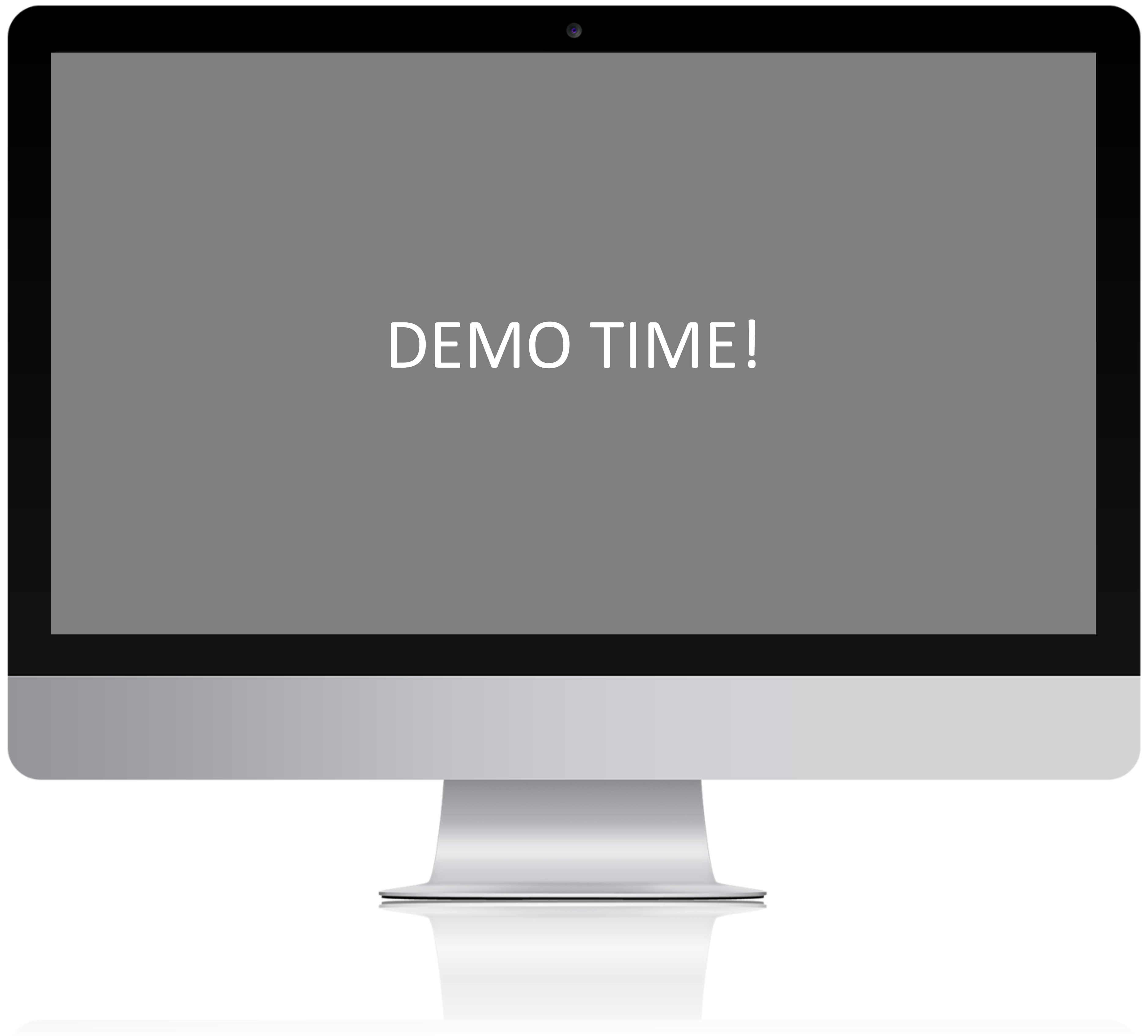
- Scenario: SQL is in place, you want to use it on the fly when you generate the container

```
New-BcContainer
        -accept_eula -accept_outdated -PublishPorts 80
        -containerName bc19 -imageName mybc:onprem-19.0.29894.30693-w1
        -Credential $credentialBC -auth NavUserPassword
        -databaseServer 172.19.112.63 -databasePrefix cronus
        -databaseName 19w1 -databaseCredential $credentialSQL
        -artifactUrl (Get-BCArtifactUrl -type OnPrem -version 19.0 -country w1)
        -replaceExternalDatabases
        -licenseFile "c:\bcartifacts.cache\onprem\19…\w1\database\Cronus.flf"
```

License not in backup

Connecting Business Central

Do it all in one



Running SQL in a container and connecting BC

Recap

- Running SQL in a container (with SQL authentication) and connecting BC to it is straight forward and easy
- bccontainerhelper plays well with it unless you want to use database restore cmdlets
- Database restore cmdlets require Windows auth and that requires some not so nice workarounds

Thanks for your attention

Any questions?

Tobias Fenster
CTO COSMO CONSULT

DIRECTIONS
4 PARTNERS

www.directions4partners.com