**DMAS Demo-Snowflake**

**Showcase Atlas of Economical Complexity**

|  |  |
| --- | --- |
| Version: | 1.0 |
| Datum: | 06.02.2025 |
| Status: | **in Arbeit** / vorgelegt / freigegeben |
| Autor: | Matthias Balzer |
| Ablage: |  |

# Ziel

Der Showcase demonstriert die allgemeinen und Fähigkeiten des Cloud-DBMS Snowflake Daten zu laden. zu transformieren und zu präsentieren.

Er lehnt sich an den 'Atlas of Economical Complexity' (AoEC) aus dem 'Harvard Dataverse' an (<https://dataverse.harvard.edu/dataverse/atlas>). Die verwendeten Daten wurden aus diesem System bezogen. Es wurde aber keine vollständige oder optisch identische Nachbildung der Vorlage angestrebt. Die vorliegende Arbeit ist vielmehr als Proof of Concept zu verstehen, mit dem die grundlegenden Muster für ähnlich gelagerte Aufgab en erarbeitet sind.

Technisch werden die folgenden Aspekte demonstriert:

* Laden und Speichern großer Datenmengen in Snowflake mit relationalen Datenstrukturen
* Konvertierung von speziellen Datenformaten, hier Stata-Binärdaten.
* Transformation der Daten in Strukturen, die für die Visualisierung optimiert sind.
* Visualisierung der Daten

Darüber hinaus wird eine Architektur vorgestellt, die sowohl

* Plattform unabhängig wie auch
* unabhängig vom Visualisierungstool

ist.

## Offene Punkte

Aus Zeit- und Budgetgründen wurden die folgenden Aspekte nicht in den Showcase aufgenommen:

* Laden der Statistikdaten aus einer Cloud
* Laden der Statistikdaten aus einem SAP-System
* Visualisierung der Daten mit dem Snowflake-Tool Streamlit

# Beschreibung

## Fachliche Problemstellung

Eine fachliche Problembeschreibung findet sich in 'Atlas of Economical Complexity' (AoEC) in dem 'Harvard Dataverse' (<https://dataverse.harvard.edu/dataverse/atlas>).

## Generelles Systembeschreibung

Die Daten wurden aus dem AoEC-System kopiert und auf unterschiedlichen Plattformen abgelegt. Diese Plattformen sind:

* Windows-Filesystem (PC)

Von dort werden sie in das Snowflake-System übertragen und in einer relationalen Tabellenstruktur (Database) abgelegt.

Die Statistikdaten liegen in einem Binärformat vor, das für die Verarbeitung, insbes. Visualisierung, nicht direkt verwendet werden kann. Diese Daten müssen deshalb in ein ladbares Format konvertiert werden.

Nachdem die Daten konvertiert und relational abgelegt sind, werden sie für die Visualisierung vorbereitet und dafür in einen Datamart geschrieben.

Aus diesem Datamart bezieht die Visualisierungskomponente ihre Eingangsdaten und benötigt dann keine weiteren Ergänzungen oder Transformationen mehr.

Ein Bild, das Text, Diagramm, Plan, technische Zeichnung enthält.

Automatisch generierte Beschreibung

Abbildung 1: Grundlegende Möglichkeiten einer Architektur der Datenflusses

Cloud AWS: Dies kann auch ein anderes Cloud-System sein, wie z.B. MS Azure oder Google Clous Service. (Diese Möglichkeit wurde hier nicht betrachtet)

PC: On-premise Rechner, auf dem die Datenfiles verfügbar sind.

Stage (extern): Die Daten aus der Cloud werden nicht physisch abgelegt, sondern es handelt sich nur um Verweise, die das Kopieren aus der Cloud ermöglichen. Hier fallenkeine Kosten für Datenspeicher an.

Stage (intern): Daten, die nicht in einem Clous-System abgelegt sind, müssen zuerst physisch in das Snowflake-System übertragen werden (Upload) , damit sie dort verarbeitbar sind. Nach der Übertragung in die Datenbasis (Elementary Data) können die Daten aber wieder entfernt werden, um Kosten für Speicherplatz zu vermeiden.

Conversion & Load: Üblicherweise werden die binären Stata-Daten für allgemeine System (nicht-Stata) verfügbar gemacht, indem sie in CSV-Files konvertiert werden. In Snowflake kann diese Konvertierung aber ohne explizite physische Ablage erfolgen, d.h. die Daten werden transient konvertiert und danach sofort, ohne dauerhaft gespeichert zu werden, in die Datenbanktabellen geschrieben.

Datamart: Der Datamart stellt eine einheitliche Schnittstelle für unterschiedliche Visualisierungstools dar. Damit können diese ohne besondere Funktionalitäten für eigene Transformationen auskommen. Als Visualisierungstools sind angedacht:

* Streamlit (Snowflake-eigenes Dialog- und Darstellungstool)
* Tableau

## Architektur des Showcase

Ein Bild, das Text, Screenshot, Diagramm, Design enthält.

KI-generierte Inhalte können fehlerhaft sein.

Der Showcase sollte folgendes demonstrieren:

|  |  |  |
| --- | --- | --- |
| (1) | Laden von Daten mit speziellem Format | STATA-Format |
| (2) | Laden der Daten mit Snowflake-Mitteln | * Laden mit PUT-/COPY-Befehlen * *Laden mit Snowflake-Notebook* |
| (3) | Laden der Daten mit externen Programmen | * Laden mit Python-Programm |
| (4) | Aufbereitung der Daten | * Aufbereitung mit Snowflake-SQL |
| (5) | Visualisierung der Daten mit Snowflake-Mitteln | *offen* |
| (6) | Visualisierung der Daten mit externen Mitteln | * Salesforce Tableau |

## Voraussetzungen

Der Showcase wurde mit einem Trial-Account von Snowflake entwickelt. Die Rahmenbedingungen dieses Account enden am 16.04.2025. Die dort abgelegten Daten sind dann zu den Bedingungen des Trail-Accounts grundsätzlich nicht mehr erreichbar, es sei denn, der Account würde in einen regulären, d.h. bezahlten, umgewandelt werden

.

Um den Showcase wieder verfügbar zu machen, werden benötigt:

* Ein neuer Snowflake-Account
* Einige elementare Kenntnisse im Umgang mit Snowflake

Im Anhang Snowflake-Voraussetzungen (Kap. 6.3) findet sich eine kurze Anleitung zum Umgang mit Snowflake.

# Aufbau

## Basisstruktur

Die Basisstruktur für die Datenbank des Showcase in Snowflake besteht aus:

* Einem Snowflake-Account
* in diesem Account eine Engine für Datentransporte (Snowflake Warehouse)
* in diesem Account eine Datenbank 'DMAS' (Snowflake Database)
* in dieser Datenbank Schemata für weitere Snowflake-Objekte
* einem GitHub-Repository [mbasul/DMAS](https://github.com/mbasul/DMAS) für die Skripte des Showcase.

Tabelle 1 Snowflake-Objekte (Account-Level):

| ***#*** | ***Objekt-Name*** | ***Objekttyp*** | ***System*** | ***Erstellung mit*** | ***Anmerkung*** |
| --- | --- | --- | --- | --- | --- |
| 1 | DMAS\_2X | Warehouse | SN | dmas\_setup.DB.sql[[1]](#endnote-1) | WH speziell für Uploads |
| 2 | DMAS | Database | SN | dmas\_setup.DB.sql\* |  |
| 3 | AOEC\_MASTER | Schema | SN | dmas\_setup.DB.sql\* |  |
| 4 | AOEC\_FACTS | Schema | SN | dmas\_setup.DB.sql\* |  |
| 5 | AOEC\_DATAMART | Schema | SN | dmas\_setup.DB.sql\* |  |
| 6 | DMAS | Repository | GitHub | --- | Persistent vorhanden; muss nicht mehr erstellt werden |
|  |  |  |  |  |  |

Tabelle 2: Schema-Objekte in Snowflake

| ***#*** | ***Objekt-Name*** | ***Objekttyp*** | ***Schema*** | ***Erstellung mit*** | ***Anmerkung*** |
| --- | --- | --- | --- | --- | --- |
| 1 | FILES\_MASTERDATA | interner Stage | AOEC\_MASTER | dmas\_setup.DB.sql[[2]](#endnote-2) |  |
| 2 | HS12\_CSV | File Format | AOEC\_MASTER | dmas\_setup.DB.sql\* |  |
| 3 | Tabellen | Tables | AOEC\_MASTER | dmas.setup.Tables.sql | Siehe Kap. 5.2.1 |
| 4 | FILES\_HS12 | interner Stage | AOEC\_FACTS | dmas\_setup.DB.sql\* |  |
| 5 | HS12\_CSV | File Format | AOEC\_FACTS | dmas\_setup.DB.sql\* |  |
| 6 | Tabellen | Tables | AOEC\_FACTS | dmas.setup.Tables.sql | Siehe Kap. 5.3.1 |
| 7 | Tabellen | Tables | AOEC\_DATAMARTS | dmas.setup.Datamart.sql | Siehe Kap. 5.4.1 |

## Datenstrukturen

Die Tabellen wurden entsprechend ihrer Funktion durch ein Database-Schema gruppiert und von Tabellen mit anderer Funktion abgegrenzt:

| ***Funktion*** | ***Schema*** |
| --- | --- |
| Stammdaten | AOEC\_MASTER |
| Bewegungsdaten (Statistikdaten) | AOEC\_FACTS |
| Datamart | AOEC\_DATAMART |

# Laden der Daten

## Voraussetzungen

### Snowflake-Account

Falls kein regulärer Snowflake-Account verfügbar ist, dann kann über [www.snowflake.com/](https://www.snowflake.com/) ein Trial-Account eingerichtet werden. Dieser Account besitzt dann ein Anfangsbudget von 400 USD und ist 30 Tage gültig. Nach Ablauf dieser Frist ist ein Arbeiten mit dem Account nicht mehr möglich, es sei denn, er würde in einen regulären Account umgewandelt werden.

Für den Showcase genügt ein Account mit der Standard-Edition.

### Benötigte Installationen

| ***#*** | ***Item*** | ***Typ*** | ***Version*** | ***Anmerkung*** |
| --- | --- | --- | --- | --- |
|  | snowSQL | Snowflake-Utility |  |  |
|  | AOEC\_FILES\_HS12 | Snowflake Stage | --- |  |
|  | Python |  | 3.9 |  |
|  | pandas | Python Package |  |  |
|  | sqlalchemy | Python Package |  |  |

## Laden der Stammdaten

Die Stammdaten zum Showcase können aus dem Internet-Auftritt des ['Atlas of Economical Complexity'](https://dataverse.harvard.edu/dataverse/atlas) AoEC als .zip-Datei :

datvers\_files.zip

heruntergeladen werden. Die Datei enthält die .csv-Dateien mit Stammdaten:

* location\_country.csv
* location\_group.csv
* location\_group\_member.csv
* product\_hs12.csv
* product\_services\_unilateral.csv
* product\_sitc.csv

### Tabellen der Stammdaten

| ***Tabelle*** | ***Datenquelle*** |
| --- | --- |
| * RAW\_LOCATION\_COUNTRY | location\_country.csv |
| * RAW\_LOCATION\_GROUP | location\_group.csv |
| * RAW\_LOCATION\_GROUP\_MEMBER | location\_group\_member.csv |
| * RAW\_PRODUCT\_HS12 | product\_hs12.csv |
| * RAW\_PRODUCT\_SERVICES\_UNILATERAL | product\_services\_unilateral.csv |
| * RAW\_PRODUCT\_SITC | product\_sitc.csv |

Die Tabellen werden nur beim Ausführen des Python Lade-Programmes automatisch erstellt.

### Laden über Stage

Das Laden der Stammdaten über einen SN-Stage findet in zwei Schritten statt:

1. Hochladen der Daten von einem on-premise Rechner in den Stage FILES\_MASTERRDATA
2. Übertragen der Daten in die SN-Tabellen mit dem FILE\_FORMAT: DMAS.AOEC\_MASTER.HS12\_CSV

Bei keinem dieser Schritte wird eine Tabelle aus den hochgeladenen Daten abgeleitet und erstellt. Die Zieltabellen müssen deshalb explizit erstellt werden. Die benötigten 'CREATE TABLE'-Befehle sind in dem SQL-Skript dmas\_setup.Tables.sql zu finden.

Dieses und alle anderen, im folgenden Ablauf aufgeführten Dateien sind alle [Github-Repository](#_Github-Repository) des Showcase abgelegt.

Tabelle 3: Upload der Masterdaten (CSV-Dateien) nur mit SN-Mitteln

| ***#*** | ***Aktion*** | ***Umgebung*** | ***Skript/Programm*** | ***Anmerkung*** |
| --- | --- | --- | --- | --- |
| 1 | Erstellen der Zieltabellen | SN (Snowsight oder snowsql) | dmas\_setup.Tables.sql |  |
| 2 | Upload der CSV-Dateien mit Masterdaten zu Stage | Windows-Kommandoeingabe | aoec.masterdata.upload.bat | Führt aus:  aoec.masterdata.upload.sql |
| 3 | Übertragen der Masterdaten in Tabellen | SN (Snowsight oder snowsql) | aoec.masterdata.load.sql |  |

### Laden mittels Python-Programm

Das Python-Programm import\_MasterData\_directly.py liest die .csv-Dateien mit den Stammdaten zum AoEC in pandas-Dataframes ein. SQLAlchemy:

* stellt die Verbindung zu Snowflake her
* leitet aus den CSV-Daten die jeweilige Tabellenstruktur ab und legt oder überschreibt die Zieltabelle mit dieser Struktur
* überträgt die Daten aus dem Dataframe in die Snowflake-Tabelle

Tabelle 4: Upload der Masterdaten (CSV-Dateien) mit Python-Programm

| ***#*** | ***Aktion*** | ***Umgebung*** | ***Skript/Programm*** | ***Anmerkung*** |
| --- | --- | --- | --- | --- |
| 1 | Installation benötigter Python-Packages | Windows | --- | Packages:   * pandas * sqlalchemy |
| 2 | Konfiguration des Python-Programms import\_MasterData\_directly.py | Windows | --- | Zu ändernd:   * [Zeile 19] Parameter 'u' von sql.create\_engine: aktueller SN-User-Name * [Zeile 20] Parameter 'p' von sql.create\_engine: Passwort * [Zeile 30] Variable 'csv\_path': Pfad zum Folder der CSV-Dateien mit Master-Daten |
| 3 | Upload der CSV-Dateien mit Masterdaten in Tabellen | Windows-Kommandoeingabe | import\_MasterData\_directly.py | Die Ziel-Tabellen werden beim Ausführen des Lade-Programmes automatisch erstellt. |

## Laden der Statistikdaten

Die Statistikdaten können aus dem Internet-Auftritt des AoEC als .zip-Datei heruntergeladen werden und liegen dann mit dem Namen vor:

datvers\_files.zip

Diese Datei enthält .dat-Files im STATA-Format. Da dieses Format nicht direkt von Snowflake verarbeitet werde kann, muss es entweder in ein CSV-Format oder auf anderem Weg verarbeitbar gemacht werden. In vorliegenden Beispiel wird das mit dem Python-Package pandas gemacht.

Von den unterschiedlichen Möglichkeiten zu laden, wurden drei für den Showcase implementiert:

1. Laden nur mit Funktionen von Snowflake
2. Laden mittels eines externen Python-Programmes
3. Laden mittels eines Snowflake-internen Notebooks

Die Verfahre unterscheiden sich unter anderem auch darin, welche vorbereitenden Schritte ausgeführt werden müssen (e.g. Konvertierung des STATA-Formates in ein CSV-Format) und welche in welche Zwischenstationen die Daten übertragen werden müssen.

Tabelle 5: Eigenschaften der Ladeverfahren

| ***#*** | ***Ladeverfahren*** | ***vorherige Konvertierung .dat 🡪 .csv*** | ***Staging in internem Stage*** | ***Zieltabellen*** | ***Performance*** |
| --- | --- | --- | --- | --- | --- |
| 1 | nur mit Funktionen von Snowflake | erforderlich | ja | müssen vorbereitet sein | hoch |
| 2 | mit externem Python-Programm | wird beim Laden konvertiert | nein | werden automatisch erstellt | gering |
| 3 | mit Snowflake-internem Notebook | wird beim Laden konvertiert | ja | werden automatisch erstellt | gering  Größenbeschränkung |

Ein Bild, das Text, Diagramm, Screenshot, Plan enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung 2: Datenflüsse für Statistikdaten der unterschiedlichen Ladeverfahren

Abbildung 3: Datenfluss-Übersicht

### Tabellen der Statistikdaten

| *Tabelle* | *Datenquelle* |
| --- | --- |
| RAW\_HS12\_COUNTRY\_COUNTRY\_PRODUCT\_YEAR\_1 | hs12\_country\_country\_product\_year\_1.dta hs12\_country\_country\_product\_year\_1.csv |
| RAW\_HS12\_COUNTRY\_COUNTRY\_PRODUCT\_YEAR\_2 | hs12\_country\_country\_product\_year\_2.dta hs12\_country\_country\_product\_year\_2.csv |
| RAW\_HS12\_COUNTRY\_COUNTRY\_PRODUCT\_YEAR\_4\_2012\_2016 | hs12\_country\_country\_product\_year\_4\_2012\_2016.dta hs12\_country\_country\_product\_year\_4\_2012\_2016.csv |
| RAW\_HS12\_COUNTRY\_COUNTRY\_PRODUCT\_YEAR\_4\_2017\_2021 | hs12\_country\_country\_product\_year\_4\_2017\_2021.dta hs12\_country\_country\_product\_year\_4\_2017\_2021.csv |
| RAW\_HS12\_COUNTRY\_COUNTRY\_PRODUCT\_YEAR\_4\_2022 | hs12\_country\_country\_product\_year\_4\_2022.dta hs12\_country\_country\_product\_year\_4\_2022.csv |
| RAW\_HS12\_COUNTRY\_COUNTRY\_PRODUCT\_YEAR\_6\_2012\_2016 | hs12\_country\_country\_product\_year\_6\_2012\_2016.dta hs12\_country\_country\_product\_year\_6\_2012\_2016.csv |
| RAW\_HS12\_COUNTRY\_COUNTRY\_PRODUCT\_YEAR\_6\_2017\_2021 | hs12\_country\_country\_product\_year\_6\_2017\_2021.dta hs12\_country\_country\_product\_year\_6\_2017\_2021.csv |
| RAW\_HS12\_COUNTRY\_COUNTRY\_PRODUCT\_YEAR\_6\_2022 | hs12\_country\_country\_product\_year\_6\_2022.dta hs12\_country\_country\_product\_year\_6\_2022.csv |
| RAW\_HS12\_COUNTRY\_PRODUCT\_YEAR\_1 | hs12\_country\_product\_year\_1.dta hs12\_country\_product\_year\_1.csv |
| RAW\_HS12\_COUNTRY\_PRODUCT\_YEAR\_2 | hs12\_country\_product\_year\_2.dta hs12\_country\_product\_year\_2.csv |
| RAW\_HS12\_COUNTRY\_PRODUCT\_YEAR\_4 | hs12\_country\_product\_year\_4.dta hs12\_country\_product\_year\_4.csv |
| RAW\_HS13\_COUNTRY\_PRODUCT\_YEAR\_6 | hs12\_country\_product\_year\_6.dta hs12\_country\_product\_year\_6.csv |

Die Tabellen werden nur beim Ausführen des Python Lade-Programmes automatisch erstellt.

### Ladeverfahren 1 (Snowflake-Mittel)

* Quelldateien müssen vom STATA-Format in ein CSV-Format umgewandelt werden
* Quelldateien liegen auf einem PC (on-premise)
* Die Struktur der Zieltabellen muss aus den CSV-Dateien abgeleitet werden, entsprechende 'CREATE TABLE'-Befehle müssen erzeugt und ausgeführt werden

Tabelle 6: Ladeablauf mit Snowflake-Mitteln

| ***Step*** | ***Umgebung*** | ***Aktion*** | ***Erläuterungen*** |
| --- | --- | --- | --- |
| 1 | Windows | Konvertieren des STATA-Formates in ein CSV-Format:  Ausführen von  dat2csv.py  für jede zu konvertierende Datei | Datenfluss '1a'  CSV-Dateien sind vorhanden. |
| 2 | Windows | Upload der CSV-Dateien:  Dazu Dateipfade in Skripten:   * hs12.statistics.csv.all.upload.bat * hs12.statistics.csv.all.upload.sql   anpassen.   1. Name des SN-Users in   hs12.statistics.csv.all.upload.bat  einsetzen   1. Passwort für SN-Account in Windows-Variable SNOWSQL\_PWD speichern 2. Skript   hs12.statistics.csv.all.upload.bat  ausführen. | Datenfluss '1b' |
| 3 | Snowflake | Ggf. Datenstrukturen in Snowflake erstellen   1. Stage AOEC\_FILES\_HS12 erstellen:   Kommandos aus  dmas\_setup.DB.sql  ausführen.   1. Tabellen erstellen:   Kommandos aus  dmas\_setup.Tables.sql  ausführen. | Die erstellten DB-Objekte sind:   * Warehouse DMAS\_2X * Database DMAS * Schema AOEC * Stage AOEC\_FILES\_HS12 * File-Format HS12\_CSV   Die 'CREATE TABLE'-Befehle in dmas\_setup.Tables.sql wurden aus den CSV-Dateien abgeleitet. |
| 4 | Snowflake | 1. Daten aus Stage in Tabellen übertragen:   Kommandos aus  hs\_12.statistikDaten.csv.copy.sql  ausführen. | Datenfluss '1c' in Abbildung 2: Datenflüsse der unterschiedlichen Ladeverfahren |

### Ladeverfahren 2 (Python-Programm)

* Quelldateien liegen im STATA-Format vor (.dta-Files)
* Quelldateien liegen auf einem PC (on-premise)
* Die Zieltabellen müssen nicht existieren, sondern werden automatisch aus den STATA-Dateien abgeleitet und generiert

Tabelle 7: Ladeablauf mit Python-Programm

| ***Step*** | ***Umgebung*** | ***Aktion*** | ***Erläuterungen*** |
| --- | --- | --- | --- |
| 1 | Windows | Installation von Python-Modulen:   * pandas * SQLAlchemy |  |
| 2 | Windows | Konfiguration des Python-Programms import\_HS12\_directly.stata.py:   1. Pfad zu den Quelldateien in Python-Programm anpassen (Variable src\_path) 2. Zugangsdaten für SN-Account aktualisieren  * Zeile 23, u='<username>' * Zeile 24, p='<password>' * Zeile 25, a='<account-id>' |  |
| 3 | Windows | Ausführen des Python-Programms import\_HS12\_directly.stata.py | Datenfluss '2' in Abbildung 2: Datenflüsse der unterschiedlichen Ladeverfahren |

### Ladeverfahren 3 (Snowflake-Notebook)

Das Ladeverfahren mit einem Snowflake-Notebook:

* erwartet die Quelldateien in einem Snowflake-Stage
* konvertiert die Quelldateien vom STATA-Format (.dta-Files) in weiter verarbeitbare Formate
* legt automatisch die Zieltabellen an, sofern sie nicht bereits existieren; die Tabellenstruktur wird automatisch aus den STATA-Dateien abgeleitet

#### Notebook IMPORT\_HS12\_INTERNAL\_STAGE\_STATA

Der Code für das Snowflake-Notebok IMPORT\_HS12\_INTERNAL\_STAGE\_STATA ist im GitHub-Repository [MBASUL/DMAS](#_Github-Repository) abgelegt.

Von dort kann er

* exportiert und dann nach Snowflake importiert werden oder
* über einen GitHub-Zugang

in Snowflake verfügbar gemacht werden

##### Import des Notebook via Datei

Ein Bild, das Text, Software, Computersymbol, Webseite enthält.

KI-generierte Inhalte können fehlerhaft sein.Snowsight > Projects > Notebooks: + Notebook >Import .ipynb file

##### Import des Notebooks direkt aus GitHub-Repository

Tabelle 8: Ladeablauf mit SN-Notebook

| ***Step*** | ***Umgebung*** | ***Aktion*** | ***Erläuterungen*** |
| --- | --- | --- | --- |
| 1 | Snowsight | Erstellen der allgemeinen DMAS-Strukturen |  |
| 2 | Snowsight | Erstellen eines GitHub-Stages für das GitHub-Repository MBASUL/DMAS:   1. Ausführen des Kommandos des Skriptes: dmas\_setup.Git.sql | Repository DMAS\PUBLIC\Git Repositories\SNGIT\_REPOS\_GITHUB\_MBASUL\_DMAS ist erstellt. |
| 3 | Snowsight | Starten des Imports von IMPORT\_HS12\_INTERNAL\_STAGE\_STATA aus GitHub:   1. Projects > Notebooks: + Notebook > Create from repository 2. Repository bestimmen: DMAS\PUBLIC\Git Repositories\SNGIT\_REPOS\_GITHUB\_MBASUL\_DMAS 3. Zu Notebook navigieren: DMAS\_AEC\40.DMAS.AEC.Umsetzung\03.Upload\_HS12\_programmatically 4. Notebook importieren: IMPORT\_HS12\_INTERNAL\_STAGE\_STATA: Create notebook (3 Punkte) | Dialog 'Create Notebook from Repository' erscheint. |
| 4 | Snowsight/ Dialog Create Notebook from Repository | Konfiguration des Imports:   * Name: IMPORT\_HS12\_INTERNAL\_STAGE\_STATA * File location in repository: Navigation zu Notebook-Datei in Repository * Notebook location/database: DMAS * Notebook location/schema: AOEC\_FACTS * Python environment: Run on Warehouse * Query warehouse: DMAS\_2X * Notebook Winehouse: COMPUTE\_WH | Pfad zu Notebook: Siehe (a)  Pfad im Repository: Siehe (b) |

1. Pfad zu Notebook: Database: DMAS, Repository: SNGIT\_REPOS\_GITHUB\_MBASUL\_DMAS
2. Pfad im Repository: DMAS.AEC/40.DMAS\_AEC.Umsetzung/03.Upload\_HS12.programmatically/IMPORT\_HS12\_INTERNAL\_STAGE\_STATA/IMPORT\_HS12\_INTERNAL\_STAGE\_STATA\_3.ipynb

#### Durchführung des Ladens

Tabelle 8: Ladeablauf mit SN-Notebook

| ***Step*** | ***Umgebung*** | ***Aktion*** | ***Erläuterungen*** |
| --- | --- | --- | --- |
| 1 | Windows | Installation von Python-Modulen:   * pandas * SQLAlchemy |  |
| 2 | Snowflake | Konfiguration des Notebooks IMPORT\_HS12\_INTERNAL\_STAGE\_STATA:   1. Packages ergänzen aus Anaconda  * pandas * SQLAlchemy  1. In Stage DMAS.PUBLIC.FILES\_UPLOAD hochladen:  * wheel\_loader.py * snowflake\_sqlalchemy-1.7.3-py3-none-any.whl  1. Dateien aus (2) als Packages aus Stage ergänzen | zu (2): Das Anmeldeverfahren von SQLAlchemy funktioniert innerhalb SN nicht. Aus diesem Grund sind die Notebook-Zellen 5 bis erforderlich. |
| 3 | Windows | Upload der Stata-Dateien:   1. Dateipfade in Skripten:   hs12.statistics.dat.all.upload.bat  hs12.statistics.dat.all.upload.sql  anpassen.   1. Name des SN-Users in   hs12.statistics.dat.all.upload.bat  einsetzen   1. Passwort für SN-Account in Windows-Variable SNOWSQL\_PWD speichern 2. Skript   hs12.statistics.dat.all.upload.bat  ausführen. | Datenfluss '3a' in Abbildung 2: Datenflüsse der unterschiedlichen Ladeverfahren |
| 4 | Snowflake | Ausführen des Notebooks IMPORT\_HS12\_INTERNAL\_STAGE\_STATA | Datenfluss '3b' in Abbildung 2: Datenflüsse der unterschiedlichen Ladeverfahren  Die Anmeldung für die Snowflake-Session wird von Notebook als Snowflake-Zugang verwendet. |

#### Bewertung des Ladens mit dem Snowflake-Notebook

Mit der aktuell beschriebenen Auswahl von Snowflake-Mitteln lassen sich allerdings alle STATA-Dateien mit dem Snowflake-Notebook laden. Eine Größenbeschränkung von 250 MB bei internen temporären Dateien lässt es nicht zu, dass z.B. hs12\_country\_country\_product\_year\_6\_2012\_2016.dta so geladen werden kann.

Die Performance des Ansatzes mit Notebooks ist außerdem deutlich schlechter als mit dem PUT/COPY-Ansatz.

## Laden des Datamarts

### Tabellen und Views des Datamart

Die Objekte des Datamarts gehören alle zum Database-Schema AOEC\_DMART.

| ***Tabelle/View*** | ***Typ*** | ***Beschreibung*** |
| --- | --- | --- |
| HS12\_PROD\_FLOW\_TOTALS | Table | Zusammenfassung der geladenen Statistikdaten für HS12. |
| HS12\_FLOW\_TOTALS | Table | Zusammenfassung über alle Produkte |
| HS12\_FLOW\_DENORM | Table | Für Tableau-Bericht aoec.twb aufbereitete Zusammenfassung über alle Produkte |
| HS12\_FLOW\_TABLEAU | View | View zur Entkoppelung des Tableau-Berichts aoec.twb von der Tabelle HS12\_FLOW\_DENORM |

Sie werden durch Ausführen der SQL-Kommandos des Skriptes dmas\_setup.Datamart.sql erstellt.

### Laden der Tabellen des Datamart

Tabelle 8: Ablauf des Ladens des Datamart AOEC\_DMART

| ***Step*** | ***Umgebung*** | ***Aktion*** | ***Erläuterungen*** |
| --- | --- | --- | --- |
| 1 | Snowflake | Ausführen der Befehle des SQL-Skriptes:  dmas.Datamart.etl.sql |  |

# Visualisierung

Unter den möglichen Ansätzen, Geo-Daten zu visualisieren sind diese für DMAS ausgewählt worden:

| ***Tool-Klasse*** | ***Tool*** |
| --- | --- |
| aus der Familie der Snowflake-Produkte | Streamlit |
| 3d-Party | [Tableau](#_Tableau) |

## Tableau

### Voraussetzungen

#### Tableau

Der Bericht wurde mit Tableau Desktop Professional Edition 2024.3.3 entwickelt.

#### Snowflake

Um Daten aus einer Snowflake-Datenbank in einem Tableau-Report darstellen zu können, wird ein spezieller Treiber benötigt, der über <https://www.tableau.com/support/drivers> erhältlich ist.

Die Verbindung mit Snowflake muss in Tableau konfiguriert werden. Dazu sind die folgenden Parameter einzutragen, die bei der Erstellung eines Snowflake-Accounts festgelegt werden:

| ***Konfigurationsparameter*** | ***Wert*** |
| --- | --- |
| Server | Server-URL aus Account-Einrichtung |
| Rolle | SYSADMIN |
| Warehouse | COMPUTE\_WH |
| Authentifizierung | Benutzername und Kennwort |
| Benutzername | Username aus Account-Einrichtung |
| Kennwort | Bei Account-Einrichtung vergebenes Kennwort |

Ein Bild, das Text, Screenshot, Zahl, Software enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung 4: Konfigurationsdialog

### Bericht

Der Bericht aoec.tbw stellt einen PoC dar, der zeigt, wie Daten aus Snowflake mit einem 3d-Party-Tool visualisiert werden können.[[3]](#footnote-1)

Die Berichtsdatei ist im [GitHub-Repository](#_Github-Repository) abgelegt.

Ein Bild, das Text, Screenshot, Karte enthält.

KI-generierte Inhalte können fehlerhaft sein.

# Anhang

## Glossar

| ***Abkürzung/Begriff*** | ***Bedeutung*** |
| --- | --- |
| AoEC | [Atlas of Economical Complexity](https://dataverse.harvard.edu/dataverse/atlas) |
| DBMS | Database Management System |
| DMAS | Data Management At Sulzer |
| ggf. | Gegebenenfalls |
| SN | Snowflake |
| Snowsight | Web-Interface von Snowflake |
| sog. | sogenannter, sogenannte, sogenanntes |

## Dateien

### GitHub-Repository

Die folgenden Dateien sind im Github-Repository [mbasul/DMAS](https://github.com/mbasul/DMAS) abgelegt.

Tabelle 9: Github-Repository MBASUL/DMAS

|  |  |
| --- | --- |
| Repository | https://github.com/mbasul/DMAS |
| User | matthias.balzer@sulzer.de |
| Passwort | OluNF0oPDMQo#jbqYJml |

Tabelle 10: Dateien des Github-Repositories

|  |
| --- |
| DMAS/DMAS.AEC/10.DMAS\_AEC.Materialien/snowflake\_sqlalchemy-1.7.3-py3-none-any.whl |
| DMAS/DMAS.AEC/10.DMAS\_AEC.Materialien/wheel\_loader.py |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/01.DB-Strukturen/dmas\_setup.Datamart.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/01.DB-Strukturen/dmas\_setup.DB.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/01.DB-Strukturen/dmas\_setup.Git.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/01.DB-Strukturen/dmas\_setup.Tables.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/02.Upload\_MasterData.programmatically/import\_MasterData\_directly.py |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/02.Upload\_MasterData.putCopy/aoec.masterdata.load.bat |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/02.Upload\_MasterData.putCopy/aoec.masterdata.load.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/02.Upload\_MasterData.putCopy/aoec.masterdata.upload.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/03.Upload\_HS12.programmatically/hs12.statistics.dta.all.upload.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/03.Upload\_HS12.programmatically/import\_HS12\_directly.csv.py |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/03.Upload\_HS12.programmatically/import\_HS12\_directly.stata.py |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/03.Upload\_HS12.programmatically/IMPORT\_HS12\_INTERNAL\_STAGE\_STATA/environment.yml |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/03.Upload\_HS12.putCopy/hs12.statistics.csv.all.copy.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/03.Upload\_HS12.putCopy/hs12.statistics.csv.all.upload.bat |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/03.Upload\_HS12.putCopy/hs12.statistics.csv.all.upload.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/04.Datamart/dmas.Datamart.DM\_setup.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/04.Datamart/dmas.Datamart.etl.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/04.Datamart/dmas\_setup.Datamart.sql |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/AOEC\_LOAD\_PRODUCTS\_COUNTRY2\_HS92/notebook\_app.ipynb |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/Dokumentation/dmas.AoEC.datenfluss.drawio |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/Dokumentation/dmas.AoEC.showcase.docx |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/Dokumentation/dmas.AoEC.showcase\_ovw.drawio |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/Upload/AOEC\_LOAD\_HS12\_STATA\_ALL/AOEC\_LAOD\_HS12\_STATA\_ALL.ipynb |
| DMAS/DMAS.AEC/40.DMAS\_AEC.Umsetzung/Visualisierung/Tableau/aoec.twb |
| DMAS/DMAS.AEC/dmas.xlsx |
| DMAS/README.md |

## Snowflake-Voraussetzungen

### Snowflake-Account

Falls kein regulärer Snowflake-Account verfügbar ist, dann kann über [www.snowflake.com/](https://www.snowflake.com/) ein Trial-Account eingerichtet werden. Dieser Account besitzt dann ein von Snowflake spendiertes Anfangsbudget von 400 USD und ist 30 Tage gültig. Nach Ablauf dieser Frist ist ein Arbeiten mit dem Account nicht mehr möglich, es sei denn, er würde in einen regulären Account umgewandelt werden.

Für den Showcase genügt ein Account mit der Standard-Edition.

Nach der Beantragung eines Account erhält man E-Mails von 'Snowflake Computing':

* 'Activate your Snowflake account' zur Aktivierung des Account mit einer URL zum Login. In dieser URL ist auch die von Snowflake erteilte Company-ID und der Account-Name enthalten
* 'Welcome to Snowflake!', in der gewählte Username bestätigt und die dedicated URL mitgeteilt wird.

### Snowflake-Logon

Ein Bild, das Text, Screenshot, Software, Computersymbol enthält.

KI-generierte Inhalte können fehlerhaft sein.

dedicated URL aus Anmeldungs-Mail

### Arbeiten mit einem Snowflake-Worksheet

Snowsight ist das Web-Interface zu Snowflake.

Worksheet erstellen: Snowsight > Projects > Worksheets > + > SQL-Worksheet

Ein Bild, das Text, Screenshot, Software, Computersymbol enthält.

KI-generierte Inhalte können fehlerhaft sein.

Snowsight > Projects > Worksheets > +

Ein Bild, das Text, Screenshot, Software, Webseite enthält.

KI-generierte Inhalte können fehlerhaft sein.Im SQL-Worksheet muss die Default-Rolle und das Warehouse (=Prozessor) angegeben sein:

Rolle zum Anlegen der Datenstrukturen

Default Warehouse

Ein Bild, das Text, Software, Computersymbol, Zahl enthält.

KI-generierte Inhalte können fehlerhaft sein.Default-Datenbank und -Schema sollten auch angegeben werden:

Ein Bild, das Text, Screenshot, Software, Computersymbol enthält.

KI-generierte Inhalte können fehlerhaft sein.SQL-Kommandos aus den Skript-Dateien können mit Cut&Paste in ein SQL-Worksheet übertragen werden und dort durch Positionieren des Cusors und <CTRL-Return> oder  ausgeführt werden

### Arbeiten mit snowSQL

#### Installation

Download: <https://developers.snowflake.com/snowsql/>

Anleitung: https://docs.snowflake.com/en/user-guide/snowsql-install-config

#### Anwendung von snowSQL

snowSQL werden Dateien, die nicht in einer der von Snowflake erschlossenen Clouds AWS, Azure, GCS liegen, mit Snowflake-Mitteln geladen. Die zu ladenden Dateien müssen dazu vom Rechner aus gelesen werden können, auf dem snowSQL installiert ist.

Der grundlegende, von snowSQL auszuführende Befehl für das Hochladen einer Datei ist:

PUT file://<absolut\_path\_to\_file> @<internal\_stage>

Um in snowSQL Komandos auszuführen, die in einer Kommandodatei stehen, muss snowSQL unter Windows aufgerufen werden mit:

1. set SNOWSQL\_PWD=<passwort>
2. snowsql -a <account-id> -u <user-name> -f <kommando-datei>

1. In GitHub-Repository DMAS abgelegt. [↑](#endnote-ref-1)
2. In GitHub-Repository DMAS abgelegt. [↑](#endnote-ref-2)
3. Auf eine möglichst genau Nachbildung eines Reports aus dem Harvard-AoEC wurde bewusst verzichtet. [↑](#footnote-ref-1)