

tac

TAC-POT #3



PowerQuery



PowerQuery vs. Pandas

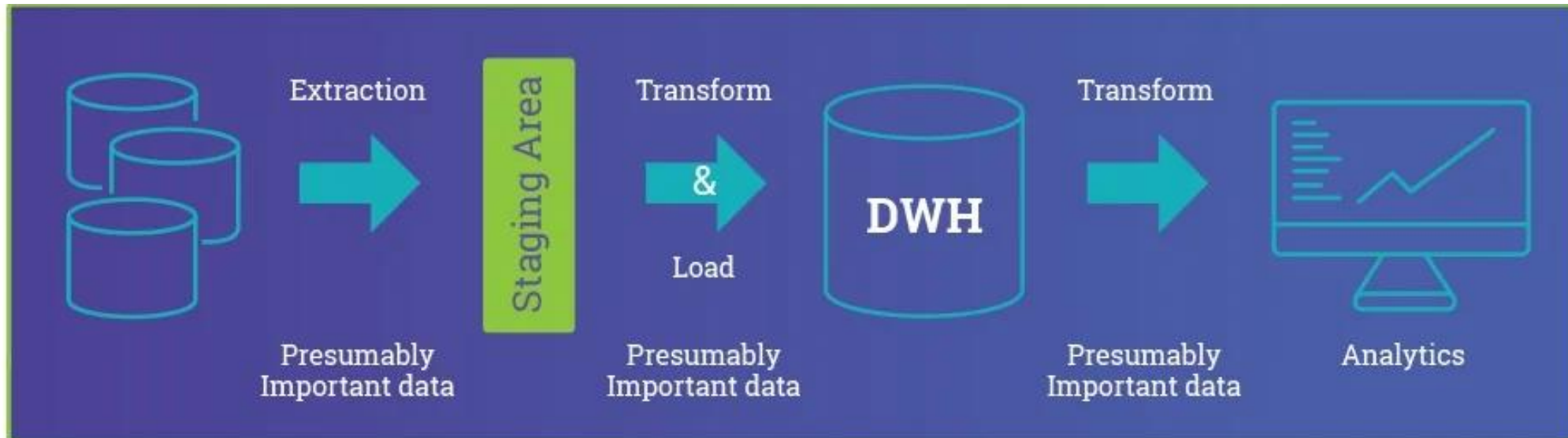
1. Adressierte Problemstellungen – ETL
2. Power Query & Pandas Überblick
3. Power Query in der Microsoft Familie
4. Pro / Cons

Beispiele

Webdaten Download
Lade Ordner mit gleicher File Struktur
Transformation von Daten (Append/Merge)

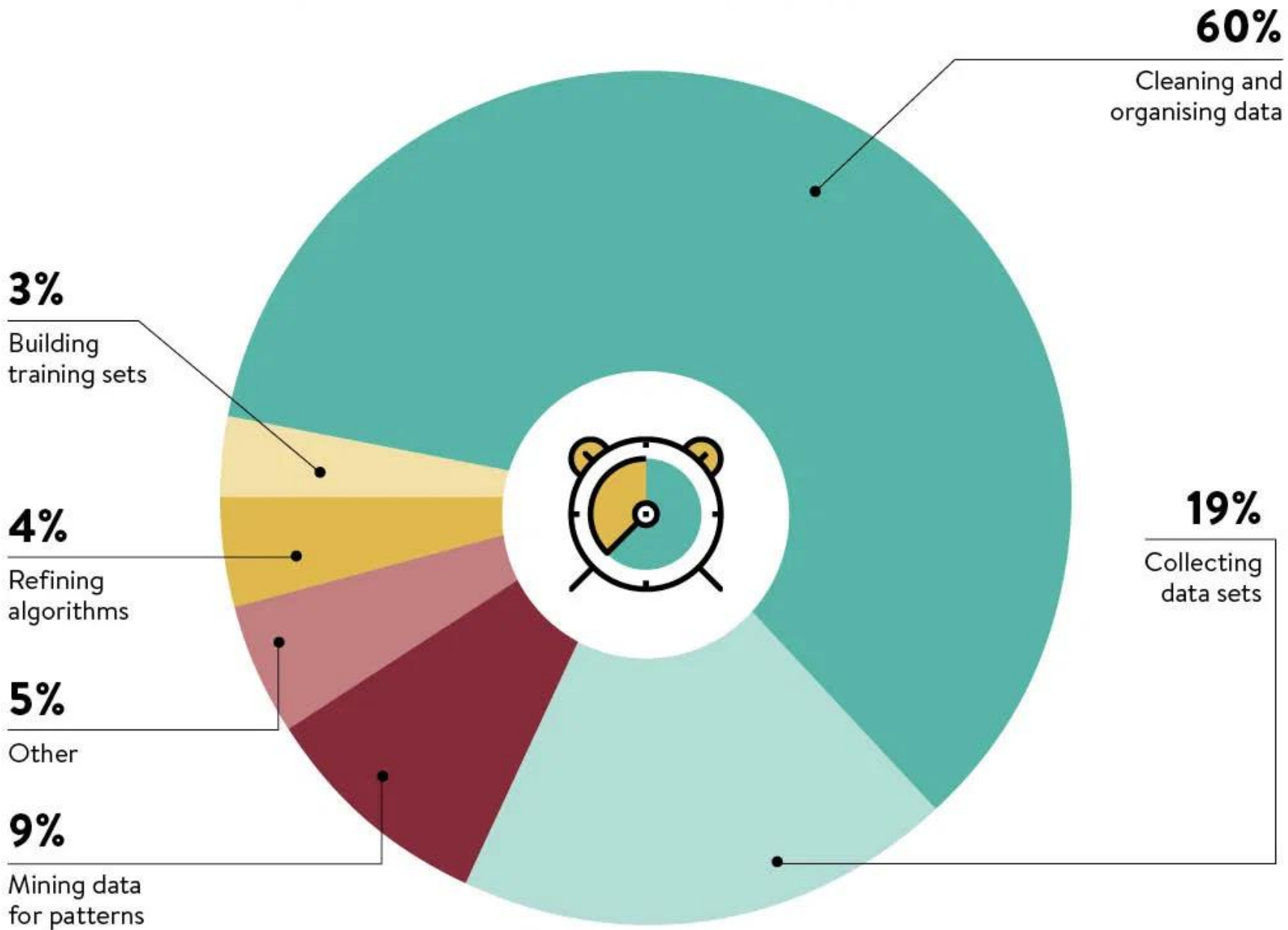
Primärer Zweck dieser Tools: ETL (Extract, Transform, Load)

- ✓ Daten aus **unterschiedlichen Quellen** werden **regelmäßig** in ein Model laden (**Extract**)
- ✓ Daten müssen **standardmäßig anders dargestellt oder aufbereitet** werden (**Transform**), damit diese Sinnvoll analysiert werden können
- ✓ Daten werden nach standardisierter Aufbereitung in ein „Datenmodel“ geladen oder weiter genutzt (**Load**)



WHAT DATA SCIENTISTS SPEND THE MOST TIME DOING

tac





PowerQuery

- Standard Modul in Excel seit 2013
- User-Interaktiv mit „Schritt-Anzeige“ zur Anwendung von Datenladen und Transformieren für Excel Daten Model bzw. Flat Files
- Optional: Verwendung von M-Language Code für „Erfahrenere User“
- Verwendung zur ETL in Microsoft Power BI



- Paket / Library für Python
- Anwendung:
 - Datenlade Funktionen
 - Gängige Transformation Funktionen & Ausgabe Funktionen
 - > Arbeit mit einem Dataframe (Tabelle mit Header und Index)
- Nur Code basiert
- Hoher Skalierungsgrad -> Anwendung bei mittleren/hohen Automatisierungsgrad oder hoher Datenmenge

Ziel: *Daten aus unterschiedlichen Quellen konsolidieren/laden und ins gewünschte Datenformat des Zielmodels transformieren*



- 2008 Developer Wes McKinney started working on pandas in 2008 while at AQR Capital Management out of the need for a high performance, flexible tool to perform quantitative analysis on financial data. Before leaving AQR he was able to convince management to allow him to open source the library
- 2012 Another AQR employee, Chang She, joined the effort in 2012 as the second major contributor to the library
- 2015 In 2015, pandas signed on as a fiscally sponsored project of NumFOCUS, a nonprofit charity in the United States



function

- **DataFrame object for data manipulation with integrated indexing**
- Tools for reading and writing data between in-memory data structures and different file formats
- **Data alignment and integrated handling of missing data**

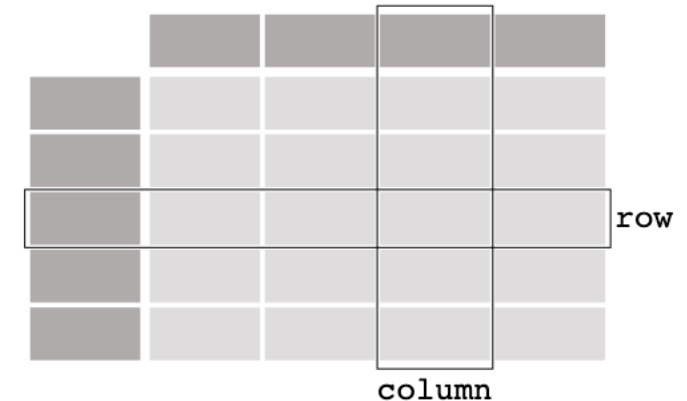
- **Reshaping and pivoting of data sets**
- Label-based slicing, fancy indexing, and subsetting of large data sets
- **Data structure column insertion and deletion**
- **Group-by-engine allowing split-apply-combine operations on data sets**
- **Data set merging and joining**

- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure
- Time series-functionality: Date range generation and frequency conversion, moving window statistics, linear regressions, date shifting and lagging
- Provides data filtration

language

The library is highly optimized for performance, with critical code paths written in Cython or C

DataFrame

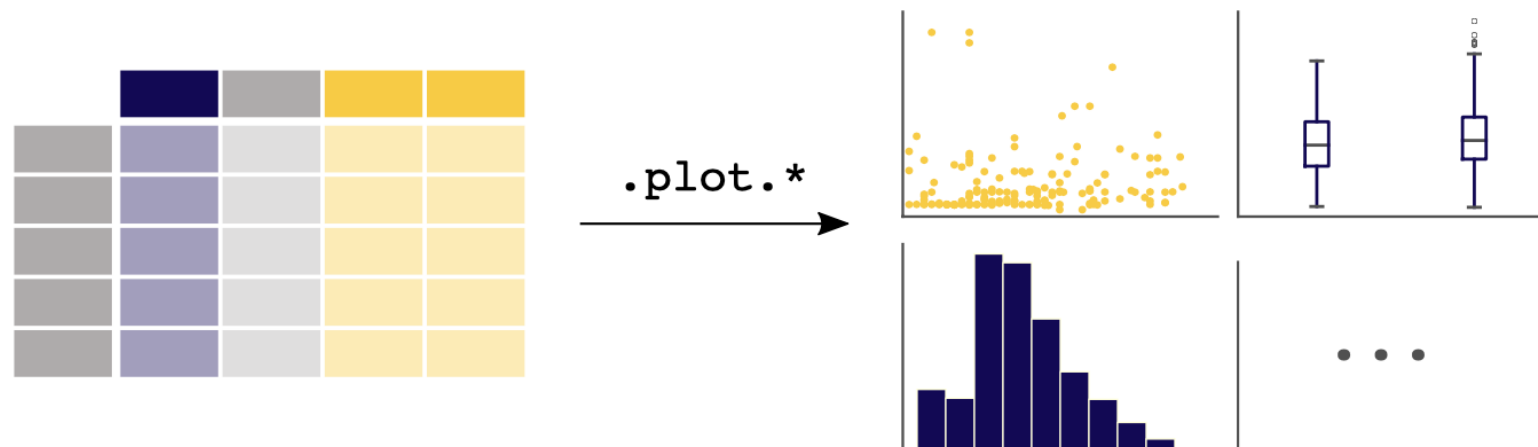




usage

Pandas is mainly used for data analysis. Pandas allows importing data from various file formats such as comma-separated values, JSON, SQL, Microsoft Excel.

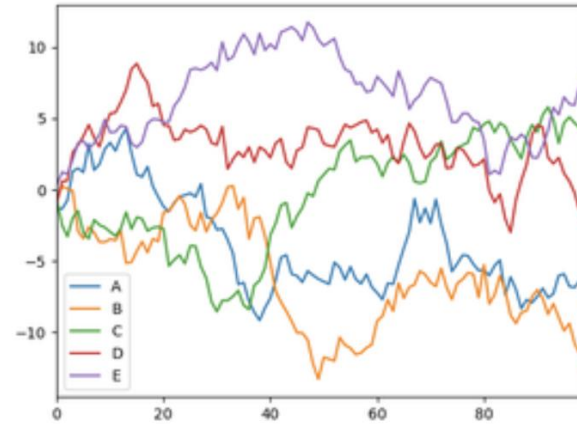
Pandas allows various data manipulation operations such as **merging**, **reshaping**, **selecting**, as well as **data cleaning**, and **data wrangling features**.



Kurven

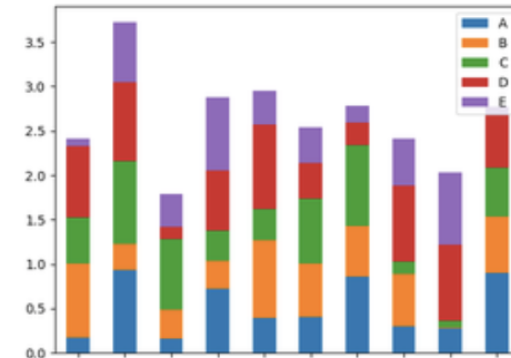
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

df = pd.DataFrame(np.random.randn(100, 5), columns=list('ABCDE'))
df=df.cumsum()
df.plot()
plt.show()
```



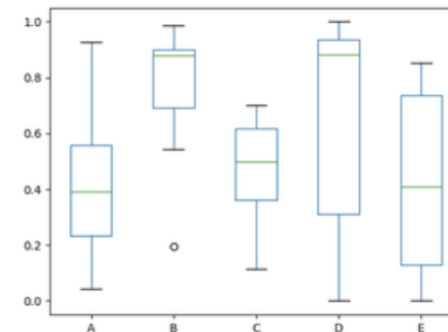
Stabdiagramme

```
df = pd.DataFrame(np.random.rand(10, 5), columns=list('ABCDE'))
df.plot.bar(stacked=True)
plt.show()
```



Boxplot

```
df = pd.DataFrame(np.random.rand(7, 5), columns=list('ABCDE'))
df.plot.box()
plt.show()
```





Excel

Module in Excel



PowerQuery



PowerPivot



PowerView

Microsoft Power Platform

The low-code platform that spans Office 365, Azure, Dynamics 365, and standalone applications



Power BI
Business analytics



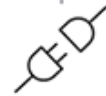
Power Apps
Application development



Power Automate
Process automation



Power Virtual Agents
Intelligent virtual agents



Data connectors



AI Builder

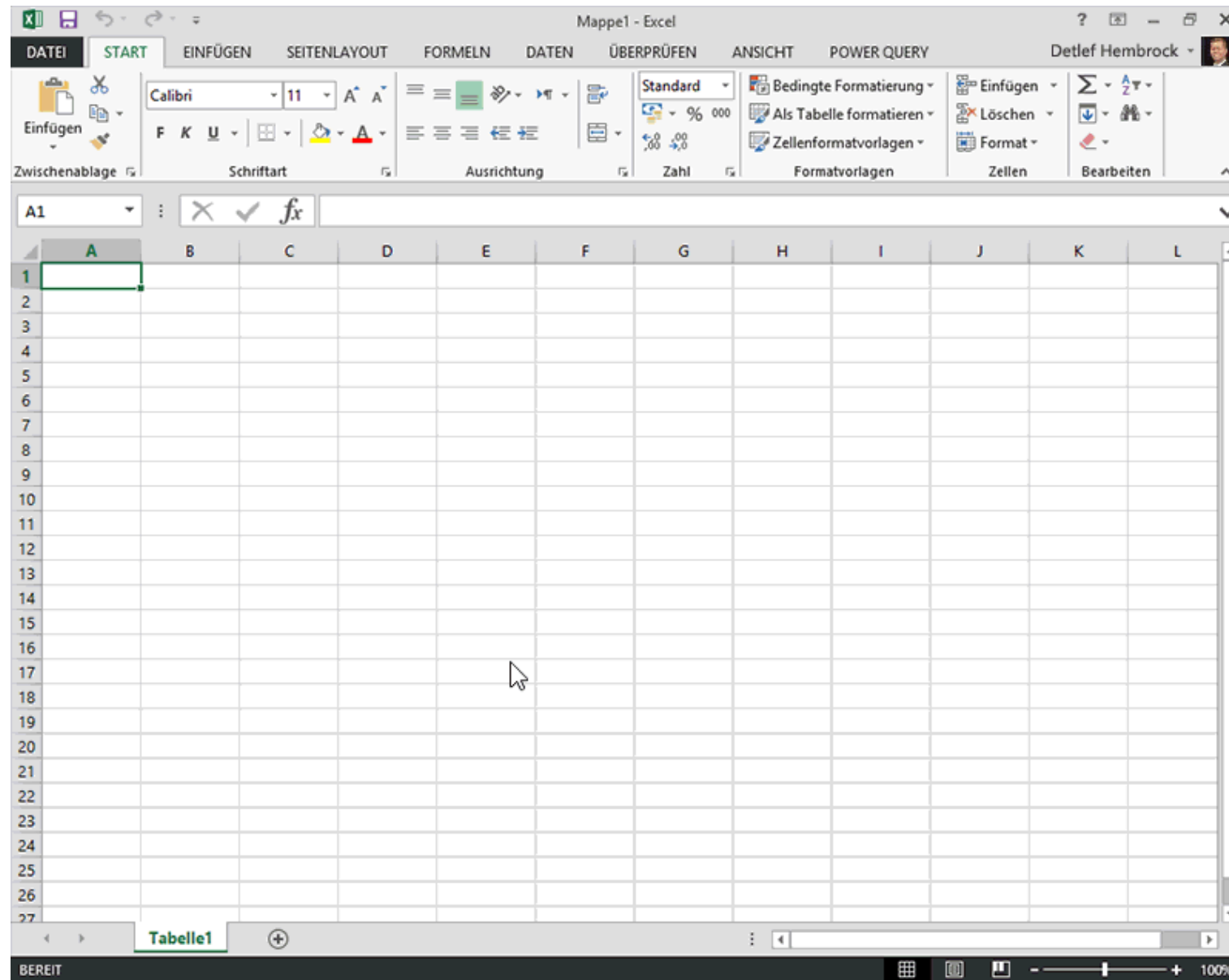


Common Data Service

Die alte Corporate Welt:
ELT + Data Model + Visualisierung

PowerQuery – Start in Excel

tac



Powerquery in Excel – User UI

tac

DAX-Werte – Abfrage-Editor

Datei Start Transformieren Spalte hinzufügen Ansicht

Schließen & laden Vorschau aktualisieren Eigenschaften Erweiterter Editor Spalten auswählen Spalten entfernen Zeilen verringern Sortieren Spalte teilen Gruppieren nach Datentyp: Datum Erste Zeile als Überschriften verwenden Werte ersetzen Kombinieren Neue Quelle Zuletzt verwendete Quellen

Schließen Abfrage Spalten verwalten Sortieren Transformieren Neue Abfrage

Code Zeile

```
= Table.TransformColumnTypes(#"Höher gestufte Header" & "Date", {"Open", Int64.Type}, {"High",
```

	Date	Open	High	Low	Close	Volume	Adj Close
1	29.04.2016	10235469727	10252080078	10038969727	10038969727	109096100	10038969727
2	28.04.2016	10222110352	10331929688	10125519531	10321150391	90272400	10321150391
3	27.04.2016	10283099609	1032325	10220570312	10299830078	96099700	10299830078
4	26.04.2016	10359769531	10385230469	10213549805	10259589844	77506700	10259589844
5	25.04.2016	10379160156	10399299805	10233419922	10294349609	72904900	10294349609
6	22.04.2016	10377219727	10422860352	10324830078	10373490234	81023600	10373490234
7	21.04.2016	10456990234	10474379883	10341969727	10435730469	98372000	10435730469
8	20.04.2016	10312639648	10440129883	10304139648	10421290039	93428400	10421290039
9	19.04.2016	10170179688	10370799805	10150669922	10349589844	96480300	10349589844
10	18.04.2016	9933700195	10147849609	9920900391	1012030957	66507700	1012030957
11	15.04.2016	10063889648	10070040039	10020339844	10051570312	79507400	10051570312
12	14.04.2016	10042280273	1009844043	10016169922	10093650391	79620200	10093650391
13	13.04.2016	9901129883	10026099609	9893419922	10026099609	103441800	10026099609
14	12.04.2016	971675	9770469727	9617830078	9761469727	73635500	9761469727
15	11.04.2016	959555957	9751929688	9524660156	9682990234	76125300	9682990234
16	08.04.2016	9576400391	9675030273	9572570312	9622259766	75984700	9622259766
17	07.04.2016	9648549805	9702179688	948475	9530620117	80710200	9530620117
18	06.04.2016	9582459961	9635910156	9505900391	9624509766	89810800	9624509766
19	05.04.2016	964780957	9661299805	9553169922	9563360352	90899600	9563360352
20	04.04.2016	9789769531	9907049805	9732820312	9822080078	71278700	9822080078

Abfrageeinstellungen

EIGENSCHAFTEN

Name
DAX-Werte

Alle Eigenschaften

ANGEWENDETE SCHRITTE

- Quelle
- Höher gestufte Header
- Geänderter Typ

Anwendungsschritte

powerquery-m

Start Abfrage

let

Source = Text.Proper("hello world") Code Bereich

Start Ausgabe

in

Source Ausgabe des Objektes

Link <https://docs.microsoft.com/de-de/powerquery-m/>

✓ Power Query M-Funktionen

Übersicht über Power Query M-Funktionen

Verstehen von Power Query M-Funktionen

> Zugreifen auf Datenfunktionen

> Binary functions

> Kombinationsfunktionen

> Vergleichsfunktionen

> Datumsfunktionen

> DateTime-Funktionen

> DateTimeZone functions

> Dauerfunktionen

> Fehlerbehandlung

> Ausdrucksfunktionen

> Funktionswerte

> Zeilenfunktionen

> Listenfunktionen

> Logische Funktionen

> Nummernfunktionen

> Datensatzfunktionen

> Ersetzungsfunktionen

> Splitter functions

> Tabellenfunktionen

> Textfunktionen

> Zeitfunktionen

> Typfunktionen

> URI-Funktionen

> Value-Funktionen



Pro

- Nutzerfreundlich/ Intuitive Bedienung (Click&Drop) & Schritteübersicht, welcher in Code gewandelt wird
- **Jeder Corporate User hat Excel als Standard Programm (Python nicht wirklich)**
- Funktionsumfang ist ausreichend, für die meisten Datentransformations- Anwendungen
- Ausgabe der Daten in Excel -> oder Pivot (Komplex in Python) / PowerPivot Datenmodell
- Optional: Native M-Language Anwendung für mehr Komplexere Transformationen
- Power Query in Power BI, bietet Funktionalitäten von Einbindung externen Codes (Python – nur in Power BI Umgebung)

Contra

- Bearbeitung von großen Dateien bringt PowerQuery in Excel schnell an seine Kapazitätsgrenzen. (64Bit Version wird nicht richtig genutzt)
- In Excel: Beschränkt auf gegebene Standard Funktionalitäten (neues nur über Community Feedback Votes)

Pro

- Umfangreiche Daten Transformations sowie Daten Analyse Funktionen
- Kann mit enorm großen Datensets umgehen und diese schnell bearbeiten
- Ausführliche Dokumentation und große Community bei Fragestellungen
- Erweiterbar durch Python, um weitere sinnvolle Pakete für Daten Analysen/Visualisierung

Contra

- Python Programmierumgebung erforderlich sowie „generische Programmier Erfahrung“ notwendig

1) Online Daten Extrahieren

-> Beispiel Finanzen.net Aktienkurs Daten

2) Mehrere gleiche Input Daten (Excel) auf einmal verarbeiten und anhängen

3) Merge & Unpivot Funktionen

- <https://powerbi.microsoft.com/de-de/>
- <https://support.microsoft.com/de-de/office/einf%c3%bchrung-in-microsoft-power-query-f%c3%bcr-excel-6e92e2f4-2079-4e1f-bad5-89f6269cd605?ui=de-de&rs=de-de&ad=de>
- <https://de.wikipedia.org/wiki/ETL-Prozess> X
- https://resources.fivetran.com/deutsche/der-unverzichtbare-leitfaden-f%C3%BCr-die-datenintegration?campaignid=10265563821&adgroupid=101951711506&network=g&device=c&gclid=CjwKCAjwrcH3BRApEiwAxjdPTbrFvrpuPdP7ILrGDKfMREfuypzqZpor9U7SyBj0hFug-rBI9g--2hoC08wQAvD_BwE
- https://pandas.pydata.org/docs/user_guide/index.html#user-guide

Backup

Users guide 1/2

- IO tools (text, CSV, HDF5, ...)
- CSV & text files
- JSON
- HTML
- Excel files
- OpenDocument Spreadsheets
- Binary Excel (.xlsb) files
- Clipboard
- Pickling
- msgpack
- HDF5 (PyTables)
- Feather
- Parquet
- ORC
- SQL queries
- Google BigQuery
- Stata format
- SAS formats
- SPSS formats
- Other file formats
- Performance considerations
- Indexing and selecting data
- Different choices for indexing
- Basics
- Attribute access
- Slicing ranges
- Selection by label
- Selection by position
- Selection by callable
- IX indexer is deprecated
- Indexing with list with missing labels is deprecated
- Selecting random samples
- Setting with enlargement
- Fast scalar value getting and setting
- Boolean indexing
- Indexing with isin
- The where() Method and Masking
- The query() Method
- Duplicate data
- Dictionary-like get() method
- The lookup() method
- Index objects
- Set / reset index
- Returning a view versus a copy
- MultiIndex / advanced indexing
- Hierarchical indexing (MultiIndex)
- Advanced indexing with hierarchical index
- Sorting a MultiIndex
- Take methods
- Index types
- Miscellaneous indexing FAQ

- MultiIndex / advanced indexing
- Hierarchical indexing (MultiIndex)
- Advanced indexing with hierarchical index
- Sorting a MultiIndex
- Take methods
- Index types
- Miscellaneous indexing FAQ
- Merge, join, and concatenate
- Concatenating objects
- Database-style DataFrame or named Series joining/merging
- Timeseries friendly merging
- Reshaping and pivot tables
- Reshaping by pivoting DataFrame objects
- Reshaping by stacking and unstacking
- Reshaping by Melt
- Combining with stats and GroupBy
- Pivot tables
- Cross tabulations
- Tiling
- Computing indicator / dummy variables
- Factorizing values
- Examples
- Exploding a list-like column
- Working with text data
- Text Data Types
- String Methods
- Splitting and replacing strings
- Concatenation
- Indexing with .str
- Extracting substrings
- Testing for Strings that match or contain a pattern
- Creating indicator variables
- Method summary
- Working with missing data
- Values considered "missing"
- Inserting missing data
- Calculations with missing data
- Sum/prod of empties/nans
- NA values in GroupBy
- Filling missing values: fillna
- Filling with a PandasObject
- Dropping axis labels with missing data: dropna
- Interpolation
- Replacing generic values
- String/regular expression replacement
- Numeric replacement
- Experimental NA scalar to denote missing values

- Categorical data
- Object creation
- CategoricalDtype
- Description
- Working with categories
- Sorting and order
- Comparisons
- Operations
- Data munging
- Getting data in/out
- Missing data
- Differences to R's factor
- Gotchas
- Nullable integer data type
- Construction
- Operations
- Scalar NA Value
- Nullable Boolean Data Type
- Indexing with NA values
- Kleene Logical Operations
- Visualization
- Basic plotting: plot
- Other plots
- Plotting with missing data
- Plotting Tools
- Plot Formatting
- Plotting directly with matplotlib
- Computational tools
- Statistical functions
- Window Functions
- Aggregation
- Expanding windows
- Exponentially weighted windows
- Group By: split-apply-combine
- Splitting an object into groups
- Iterating through groups
- Selecting a group
- Aggregation
- Transformation
- Filtration
- Dispatching to instance methods
- Flexible apply
- Other useful features
- Examples

Users guide 2/2

- Time series / date functionality
- Overview
- Timestamps vs. Time Spans
- Converting to timestamps
- Generating ranges of timestamps
- Timestamp limitations
- Indexing
- Time/date components
- DateOffset objects
- Time Series-Related Instance Methods
- Resampling
- Time span representation
- Converting between representations
- Representing out-of-bounds spans
- Time zone handling
- Time deltas
- Parsing
- Operations
- Reductions
- Frequency conversion
- Attributes
- TimedeltaIndex
- Resampling
- Styling
- Building styles
- Finer control: slicing
- Finer Control: Display Values
- Builtin styles
- Sharing styles
- Other Options
- Fun stuff
- Export to Excel
- Extensibility
- Options and settings
- Overview
- Getting and setting options
- Setting startup options in Python/IPython environment
- Frequently Used Options
- Available options
- Number formatting
- Unicode formatting
- Table schema display
- Enhancing performance
- Cython (writing C extensions for pandas)
- Using Numba
- Expression evaluation via eval()
- Scaling to large datasets
- Load less data
- Use efficient datatypes
- Use chunking
- Use other libraries

- Sparse data structures
- SparseArray
- SparseDtype
- Sparse accessor
- Sparse calculation
- Migrating
- Interaction with scipy.sparse
- Frequently Asked Questions (FAQ)
- DataFrame memory usage
- Using if/truth statements with pandas
- NaN, Integer NA values and NA type promotions
- Differences with NumPy
- Thread-safety
- Byte-Ordering issues
- Cookbook
- Idioms
- Selection
- MultiIndexing
- Missing data
- Grouping
- Timeseries
- Merge
- Plotting
- Data In/Out
- Computation
- Timedeltas
- Aliasing axis names
- Creating example data