Semesterarbeit des Moduls  
Data Collection, Integration and Pre-processing (CIP02)

ETL-Projektdokumentation

zum Thema  
Entwicklung von Sportgrossanlässen und Bruttoinlandprodukt sowie deren Korrelation von 1950-2019

Hochschule Luzern HSLU

Studiengang: MSc Applied Information and Data Science

Studenten: Régis Andréoli, Micha Käser  
Einreichdatum: 19.05.2021

Gruppennummer: 50

**Inhaltsverzeichnis**

[1 Ausgangslage 3](#_Toc72352751)

[2 Datenquellen 4](#_Toc72352752)

[3 Tools 4](#_Toc72352753)

[4 Fragestellungen 4](#_Toc72352754)

[5 ETL-Prozess 5](#_Toc72352755)

[6 Arbeitsteilung 5](#_Toc72352756)

[7 Lösungsschritte 5](#_Toc72352757)

[7.1 Extract 5](#_Toc72352758)

[7.1.1 A1 – a1\_rgdpna\_stage.csv 5](#_Toc72352759)

[7.1.2 B1 – b1\_web\_scrapper.py 6](#_Toc72352760)

[7.1.3 B2 6](#_Toc72352761)

[7.1.4 B3 6](#_Toc72352762)

[7.1.5 C1 – c1\_web\_scrapper.py 6](#_Toc72352763)

[7.1.6 C2 – c2\_laendercode\_src.csv 7](#_Toc72352764)

[7.2 Transform 7](#_Toc72352765)

[7.2.1 A1 7](#_Toc72352766)

[7.2.2 B1 – b1\_cleaning.py 7](#_Toc72352767)

[7.2.3 B2 8](#_Toc72352768)

[7.2.4 B3 8](#_Toc72352769)

[7.2.5 C1 8](#_Toc72352770)

[7.2.6 C2 9](#_Toc72352771)

[7.3 Load 10](#_Toc72352772)

[8 Fragestellungen 10](#_Toc72352773)

[8.1 Frage 1 10](#_Toc72352774)

[8.2 Frage 2 10](#_Toc72352775)

[8.3 Frage 3 12](#_Toc72352776)

[8.4 Frage 4 12](#_Toc72352777)

[9 Reflexion 14](#_Toc72352778)

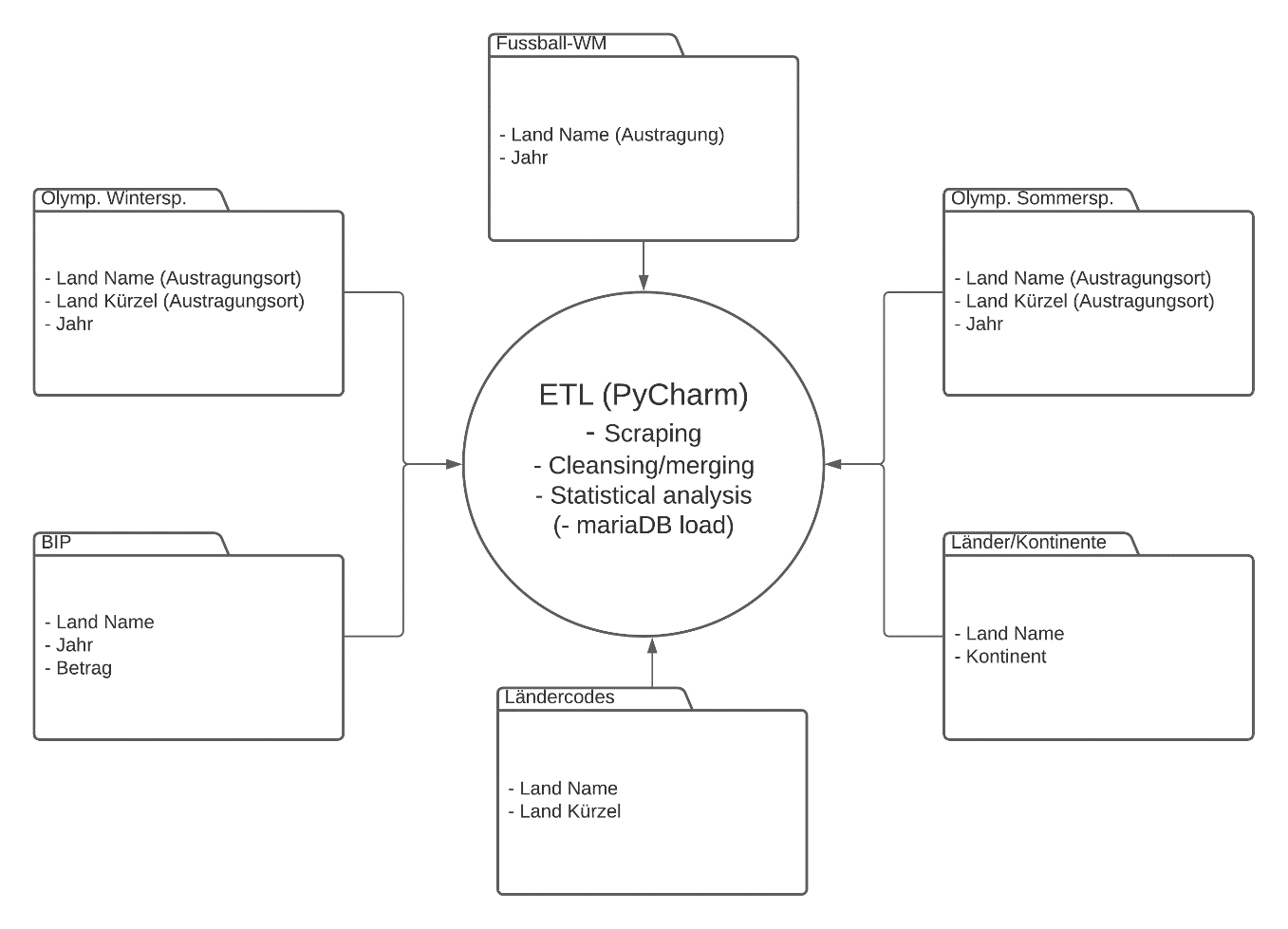
[10 Einwilligung 14](#_Toc72352779)

# Ausgangslage

Während des Masterstudiengangs Applied Information and Data Science erarbeiten die Studierenden innerhalb des Moduls CIP02 eine Semesterarbeit, welche die erlernten Extract-, Transform- und Load-Methoden der Vorlesung in die Praxis umsetzt. Ursprünglich zu dritt gestartet, brach ein Mitstudent das Studium ab – der knappen Zeit geschuldet setzen Régis Andréoli und Micha Käser das Projekt nun zu zweit um.

Als grosse Sportenthusiasten lag es nahe, ein entsprechendes Thema zu wählen. Damit auch eine gewisse Zeitachse (Time Series) gewährleistet ist, fiel die Wahl schlussendlich auf Sportgrossanlässe (Sommer- und Winterolympiaden/Fussballweltmeisterschaften) und deren Auswirkungen auf das Bruttoinlandprodukt (BIP) der jeweiligen austragenden Nation. Solche Anlässe sind gut dokumentiert und lassen sich viele Jahre zurückverfolgen, auch das BIP als wichtige wirtschaftliche Kennzahl lässt sich bis zur Industrialisierung nachvollziehen. Damit die Resultate verlässlich und aussagekräftig sind, werden die Daten allerdings erst ab der Nachkriegszeit (1950) bis 2019 ausgewertet.

Das nachfolgende Kontextdiagramm zeigt die sechs verschiedenen Datenquellen, die zur Beantwortung der Fragestellungen benötigt werden. Bis auf die Datenquelle «BIP» wurden alle mit Python gescraped.



# Datenquellen

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Daten** | **Attribute** | **Link** |
| A1[[1]](#footnote-1) | BIP nach Land [[2]](#footnote-2) 1950-2019 | * Land Name * Jahr * Betrag | <https://febpwt.webhosting.rug.nl/Dmn/AggregateXs/PivotShow> |
| B1 | Fussball-Weltmeisterschaften 1930-2018 | * Land Name Austragungsort * Jahr | <https://www.fussball-wm-total.de/History/histehre.html> |
| B2 | Olympische Winterspiele 1924-2018 | * Land Name  Austragungsort * Land Kürzel * Jahr | <https://www.taschenhirn.de/sport/olympische-winterspiele/> |
| B3 | Olympische Sommerspiele 1896-2016 | * Land Name  Austragungsort * Land Kürzel * Jahr | <https://www.taschenhirn.de/sport/olympische-sommerspiele/> |
| C1 | Länder und Kontinente | * Land Name * Kontinent | [https://www.bernhard-gaul.de/wissen/staatenerde.php#uebneu](https://www.bernhard-gaul.de/wissen/staatenerde.php%23uebneu) |
| C2 | Ländercodes | * Land Name * Land Kürzel | https://www.laenderdaten.de/kuerzel/iso\_3166-1.aspx |

# Tools

|  |  |
| --- | --- |
| **Zweck** | **Tools** |
| Repository Code | GitHub (<https://github.com/michakaeser/CIP_Project>) |
| Repository Daten | GitHub/Switch Drive |
| Kommunikation | Slack, Zoom |
| Entwicklungsumgebung | Ubuntu (VM oder nativ), PyCharm, mariaDB, Microsoft Excel/LibreOffice Calc (manuelle Verunreinigung Daten) |
| Diagramme | Lucidchart |

# Fragestellungen

1. Welches Land war zwischen 1950-2019 am häufigsten Gastgeber eines Sportgrossanlasses (Sommer- und Winterolympiade, Fussballweltmeisterschaft)?
2. Welcher Kontinent verzeichnet den grössten BIP-Zuwachs in der Periode von 2000-2019?
3. In welcher Dekade wurden auf welchem Kontinent die meisten Sportgrossanlässe (Sommer- und Winterolympiade, Fussballweltmeisterschaft) durchgeführt?
4. Gibt es eine statistisch feststellbare, signifikante Korrelation zwischen der Durchführung von Sportgrossanlässen und der Veränderung des BIPs der Gastgebernation?

# ETL-Prozess

Zu Beginn werden die Robots.txt-Files gecheckt und keine relevanten Scraping-Einschränkungen festgestellt. Nach dem Scrapen der Daten müssen einige CSV-Files (B1, C1) verunreinigt werden, da sie bereits eine hohe Qualität aufweisen. Dafür werden in Microsoft Excel oder LibreOffice Calc BIP-Beträge in verschiedene Währungen umgerechnet, Strings mit Zahlen angereichert und unterschiedliche Sprachen verwendet. Die Quellen B2 und B3 enthielten bereits Unreinheiten und wurden so weiterverarbeitet. C2 wurde als «Hilfsquelle» für Joins nach dem Scrapen im bereits reinen Zustand belassen.

Im nächsten Schritt werden die Verunreinigungen mit PyCharm korrigiert bzw. bereinigt. Die sauberen Tabellen werden gemerged und die Fragestellungen in PyCharm ausgewertet und ausgerechnet. Zusätzlich sollen die sauberen Datasets in eine mariaDB-Datenbank eingelesen werden.

# Arbeitsteilung

|  |  |  |
| --- | --- | --- |
| **Wer** | **Datenquelle** | **Was** |
| Régis Andréoli | A1, B1, C1, C2 | Datenquelle scrapen (B1, C1, C2), CSV exportieren (A1) und Daten mit Python/Pandas bereinigen/anreichern |
| Micha Käser | B2, B3 | Datenquelle scrapen (B2, B3) und Daten mit Python/Pandas bereinigen/anreichern |
| Team | Alle | Daten mergen, joinen und gruppieren. Fragestellungen beantworten. Danach mariaDB aufsetzen und bereinigte Datasets einlesen |

# Lösungsschritte

Im folgenden Abschnitt wird die Projektarbeit nach dem ETL-Schema, Extract, Transform und Load dokumentiert. Da der dokumentierte Quellcode mit dieser Arbeit abgegeben wird, wird in dieser Projektdokumentation darauf verzichtet, den kompletten Code erneut wiederzugeben. Stattdessen werden ausgewählte, spannenden Sachverhalte und Codeabschnitte aufgezeigt und erläutert.

## Extract

In der Extract Phase werden Daten aus Quellen im Internet gesammelt. Die Daten werden allesamt, bis auf A1 und C2, mittels eines Webscappers von einer Webpage extrahiert.

### A1 – a1\_rgdpna\_stage.csv

Die A1 Daten enthalten das BIP pro Jahr pro Land. Die Länder-Spezifikation ist jeweils im ISO-3166 alpha-3 festgehalten.

Die Daten werden von einer Webpage manuell via CSV-File heruntergeladen. Der ursprüngliche Versuch, dieses CSV via Python herunterzuladen, wurde eingestellt. Dies, da sich herausstellte, dass sich das CSV dynamisch bildet mittels einer Javascript Funktion im HTML Code. Allen Anschein nach triggert der Browser Button eine Javascript-Funktion, die aus dem eigenen HTML Informationen herauszieht, um die Funktion selbst zu vervollständigen. Danach wird der komplette Javascript Befehl ausgeführt und das CSV erstellt. Die Daten sowie die Javascript Funktion sind beide im HTML Code versteckt oder verteilt und macht die Extraktion kompliziert. Daher wird von der Python Lösung abgesehen, zumal diese Datenquelle nicht der Web-Scrapping Auflage unterliegt.

### B1 – b1\_web\_scrapper.py

Der B1 Web Scrapper ladet und erstellt aus einer Webseite eine List mit den Fussballweltmeisterschaften und das dazugehörige Event Jahr.

import requests  
from bs4 import BeautifulSoup as bs  
  
*#download html file and convert into soup object*raw\_html = requests.get(**"https://www.fussball-wm-total.de/History/histehre.html"**)  
soup\_html = bs(raw\_html.text, **"html.parser"**)  
  
*#get most distinctive html structure to target input*extract = soup\_html.find(**"table"**, width=**"430"**, border=**"1"**, cellpadding=**"10"**)  
  
*#loop through html lines, eliminate unwanted input values and store as csv*csv\_file = open(**"b1\_wm\_src.csv"**, **"w"**, encoding=**'utf-8'**)  
for p in extract.select(**"tr"**):  
 y = p.select(**"td"**)  
 try:  
 a = y[0].text  
 b = y[1].text  
 csv\_file.write(a + **","** + b + **"**\n**"**)  
 print(a + **","** + b + **"**\n**"**)  
 except:  
 1  
csv\_file.close()

Das Scrappen der WM-Fussballspiele ist nicht komplett konventionell. Die Webpage www.fussball-wm-total.de hat eine multiple, vernestete Struktur aus HTML Tabellen Tags. Es wurde generell wenig bis gar nicht mit Attributen gearbeitet und so ist die richtige HTML Struktur zum Scrappen schwierig zu erkennen.

Nach dem Download werden die gesuchten Daten aus dem HTML extrahiert und abgespeichert. Dies erfolgt zeitgleich in der For-Loop

### B2

### B3

### C1 – c1\_web\_scrapper.py

Der C1 Web Scrapper downloaded einen HTML Code und erstellt daraus eine Liste mit Ländername und das dazugehörige Kontinent.

Ähnlich wie bei B1, ist die Webpage aus HTML Sicht sehr einfach formatiert. Die Webpage beinhaltet nahezu keine brauchbare Struktur, und wäre an sich nicht scrappbar. Da sich aber der Webinhalt mehrheitlich auf das begrenzt, was benötigt wird, wird einfachheitshalber grob gescrappt. Die dabei entstehende Datenverunreinigung wird später im cleaning behoben.

Wie bei B1 werden die Daten in der For-Loop verarbeitet und anschliessend im CSV abgespeichert.

import requests  
from bs4 import BeautifulSoup as bs  
  
*#download html file and convert into soup object*raw\_html = requests.get(**"https://www.bernhard-gaul.de/wissen/staatenerde.php#uebneu"**)  
soup\_html = bs(raw\_html.text, **"html.parser"**)  
  
*#load extracted text into csv file*csv\_file = open(**"c1\_country\_src.csv"**, **"w"**, encoding=**'utf-8'**)  
for p in soup\_html.select(**"tr"**):  
 y = p.select(**"td"**)  
 a = y[0].text  
 b = y[1].text  
 csv\_file.write(a + **","** + b + **"**\n**"**)  
 print(a + **","**)  
csv\_file.close()

### C2 – c2\_laendercode\_src.csv

Die C2 Daten enthalten die Länder-Spezifikation im ISO-3166 alpha-3 Format sowie die vollständige Länderbezeichnung.

Die C2 Daten werden manuell gewonnen. Ein Copy-Paste aus einer Web-Tabelle macht dies zu einer schnellen Aufgabe.

## Transform

In der Transform Phase werden die Daten bereinigt und für die weitere Verarbeitung passend formatiert.

### A1

Die A1-Daten werden bereits als CSV in höchster Datengüte abgespeichert und werden auch nicht manuell verunreinigt. Der Datensatz bedingt keinen weiteren Massnahmen.

### B1 – b1\_cleaning.py

Die B1-Daten werden manuell verunreinigt und müssen auf folgendes korrigiert werden:

1. Umlauten umschreiben
2. Zeilenfehler wegen Semikolon
3. Gross- & Kleinschreibung
4. Fehlerhaftes Datum
5. replace() findet alle Umlauten und wandelt diese um. regex=True ist dabei notwendig für die Suche nach dem Umlauten.

df = df.replace(**'ä'**, **'ae'**, regex=True)  
df = df.replace(**'ö'**, **'oe'**, regex=True)  
df = df.replace(**'ü'**, **'ue'**, regex=True)  
df = df.replace(**'Ä'**, **'Ae'**, regex=True)  
df = df.replace(**'Ö'**, **'Oe'**, regex=True)  
df = df.replace(**'Ü'**, **'Ue'**, regex=True)

1. Folgend ein Beispiel wie mittels Dataframe Index eine spezifische Zeile angezeigt und anschliessend korrigiert wird.

print(df[**'Land'**][14])  
df[**'Land'**][14] = **"USA"**df[**'Jahr'**][14] = **"1994"**df.count()

1. Diese For-Loop vermagt eine DF Spalte durchzuschlaufen und erkennt dabei Kleinschreibung. Beginnt ein Name mit ein Kleinbuchstabe, wird dieser mit dem Gleichen in Grossschreibung ersetzt.

count = 0  
for p in df[**"Land"**]:  
 if df[**"Land"**][count][0].islower():  
 print(p)  
 df[**"Land"**][count] = (df[**"Land"**][count][0].upper() + df[**"Land"**][count][1:])  
 count += 1

1. Diese For-Loop überprüft ob die Daten in einer Spalten einen bestimmten Format einhalten. Hier, dürfen die Felder nicht mehr als vier Charaktere haben. Falls dem nicht so ist, wird von links beginnend alles weggeschnitten, was vier Charaktere übersteigt.

count = 0  
for p in df[**"Jahr"**]:  
 if len(df[**"Jahr"**][count]) > 4:  
 print(p)  
 df[**"Jahr"**][count] = df[**"Jahr"**][count][-4:]  
 count += 1

Anschliessend werden die Veränderungen im ‘b1\_wm\_stage.csv’ abgespeichert.

### B2

### B3

### C1

Die C1-Daten werden Manuel verunreinigt und müssen auf folgendes korrigiert werden:

1. Spalten- und Zeilenfehler
2. Wörterkorrektur
3. Gross- & Kleinschreibung

Natürliche Verunreinigungen kommen ebenfalls vor:

1. Titel enthält ungültige Zeichen
2. Im Datensatz folgen ungewollte Zeilen
3. Da der Titel Fehlerhaft ist und das einlesen des CSVs nicht funktioniert, wird dieser ignoriert (header=none) und die Zeile ausgelassen (skiprows=1). Der Titel wird gleich manuell angegeben (names=[…)

df = pd.read\_csv(**'c1\_country\_src\_dirty.csv'**, header=None, skiprows=1, names=[**'Kontinent'**, **'Land'**], encoding=**'utf-8'**)

1. Mit .isnull() werden jeweils Null Zeilen ermittelt. Dabei ist der DF Index Sichtbar. Der Index kann anschliessend genutzt werden, um die Zeilen direkt aufzurufen und zu verändern.

df[df[**'Land'**].isnull()]

df.loc[23]  
df.loc[23] = [**"Afrika"**, **"Benin"**]

1. Mit der Übersichtsausgabe aus groupby() wird ersichtlich, dass Europa ein paar mal falsch geschrieben ist. Die Abfrage \*== ‘Europe’ wiedergibt alle fehlerhaften Zeile, und es kann wieder mittels Index korrigiert werden.

print(df.groupby(**'Kontinent'**).count())  
df[df[**'Kontinent'**] == **'Europe'**]  
df[**'Kontinent'**][4] = **'Europa'**df[**'Kontinent'**][81] = **'Europa'**

1. Mit df[-15] werden die letzten Einträge angeschaut und es wird visual ermittelt, wo im Index der überflüssige Abschnitt anfängt.

df[-15:]  
  
df.drop(df.loc[242:252].index, inplace=True)

### C2

Die C2-Daten werden nicht manuell verunreinigt, enthalten allerdings Elemente die verändert werden müssen.

Die Umlauten werden wie zuvor umgeformt und zwei Spalten werden gedropt.

## Load

# Fragestellungen

## Frage 1

*Welches Land war zwischen 1950-2019 am häufigsten Gastgeber eines Sportgrossanlasses (Sommer- und Winterolympiade, Fussballweltmeisterschaft)?*

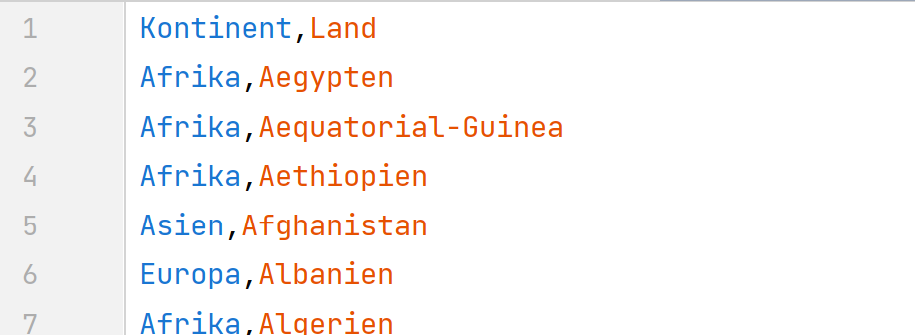
## Frage 2

*Frage Welcher Kontinent verzeichnet den grössten BIP-Zuwachs in der Periode von 2000-2019?*

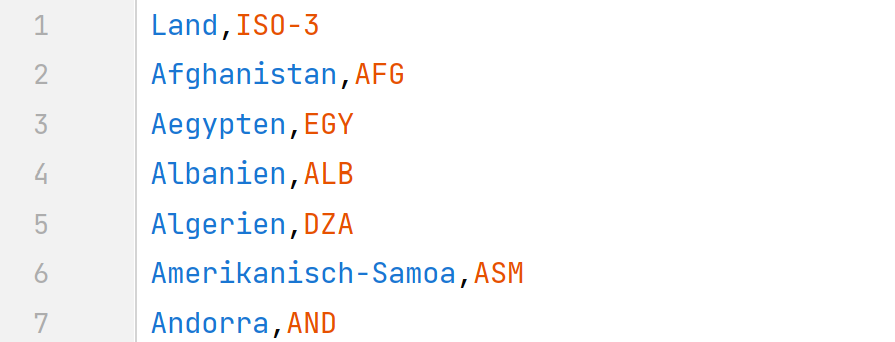
*Lösung Der Kontinent Asien schlägt mit dem fast vierfachen BIP-Zuwachs das zweitplatzierte Kontinent Nordamerika.*

Um die Fragestellung 2 zu beantworten, werden drei Datensätze miteinander verbunden:

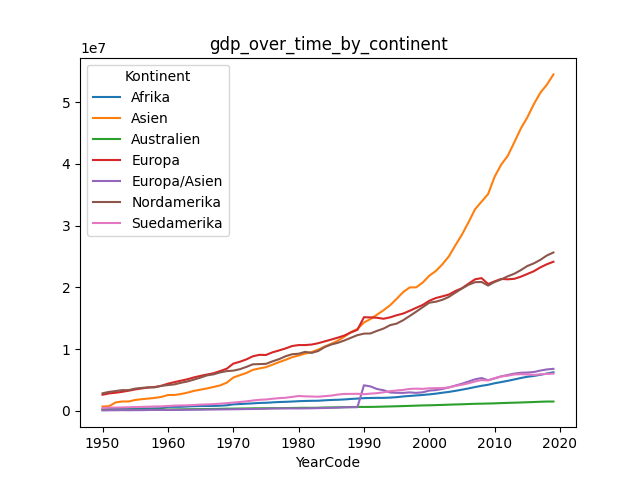
Die zwei folgenden Datensätze enthalten die relevanten Inforamtionen um die Fragestellung zu lösen…

**a1\_rgdpna\_stage.csv** **c1\_country\_stage.csv**

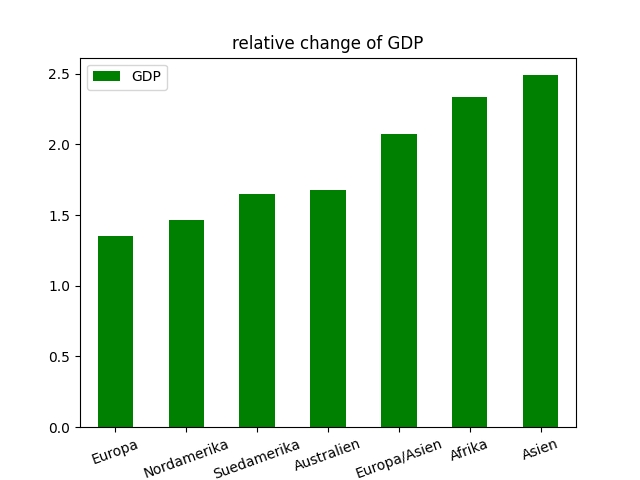
…während der dritte Datensatz dazu benötigt wird, um die ersten zwei Datensätze (A1 & C1) zusammführen zu können.

**c2\_laendercode\_stage.csv**

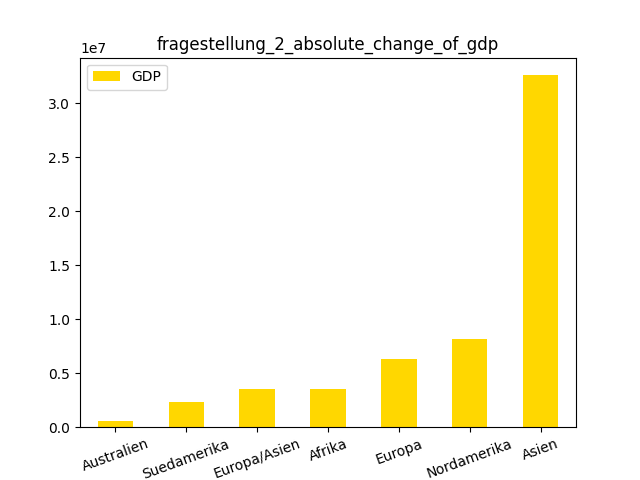
Nachdem die Daten gemerget und diversen Filter- und Auswahlverfahren durchlaufen werden, wie z.B. groupby(), werden diese in Plots visualisiert.

Folgend sind die drei Plots die erstellt werden. Der Plot ‘fragestellung\_2\_absolute\_change\_of\_gdp’ beantwortet die eigentliche Fragestellung. Die beiden anderen Plots liefern Zusatzinhalt, für das Interesse und die Neugier.

Die Grafik ‘gdp\_over\_time\_by\_continent’ zeigt das BIP pro Jahr pro Kontinent. Zu erkennen ist der sprunghafte Aufschung von Asien ende 90er oder die Einführung der Kategorie Europa/Asien ende der 80er.



Die Grafik ‘relative change of GDP’ zeigt die relative Veränderung des BIP zwischen dem Jahr 2000 und 2019. Das BIP im Jahre 2000 dient als Basis und würde in dieser Grafik den Wert 1 erhalten. Abgebildet wird die Veränderung im Jahre 2019, als Rate zum Jahre 2000.

Zu vermerken ist, das aus therminologischer Sicht, bei dieser Rechnung, eher die ‘Rate’ [x2/x1] ausgerechnet wurde als die ‘Relative Veränderung’ [(x2-x1)/x1]

Die foglende Grafik ‘fragestellunf\_2\_absolute\_change\_of\_gdp’ zeigt die absolute differenz zwischen dem Jahr 2019 und 2000. Diese Grafik löst die gestellte Frage und die Anwort ist, dass *der Kontinent Asien den grössten BIP-Zuwachs in der Periode von 2000-2019 verzeichnet.*

Alle drei Grafiken werden als png abgespeichert und zusätzlich in das Excel File ‘Result\_Question\_02’ zusammengeführt.

## Frage 3

*In welcher Dekade wurden auf welchem Kontinent die meisten Sportgrossanlässe (Sommer- und Winterolympiade, Fussballweltmeisterschaft) durchgeführt?*

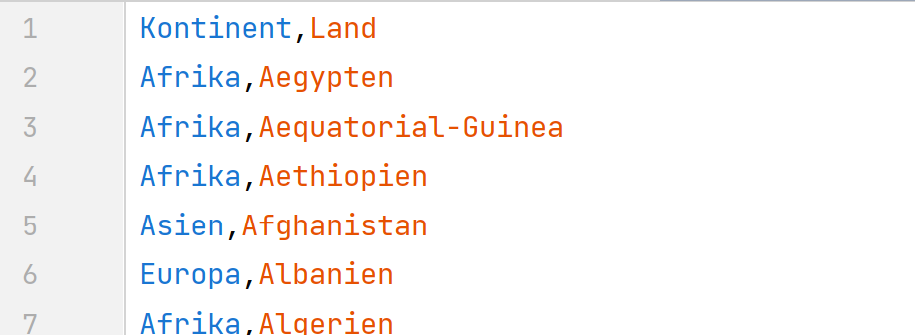
## Frage 4

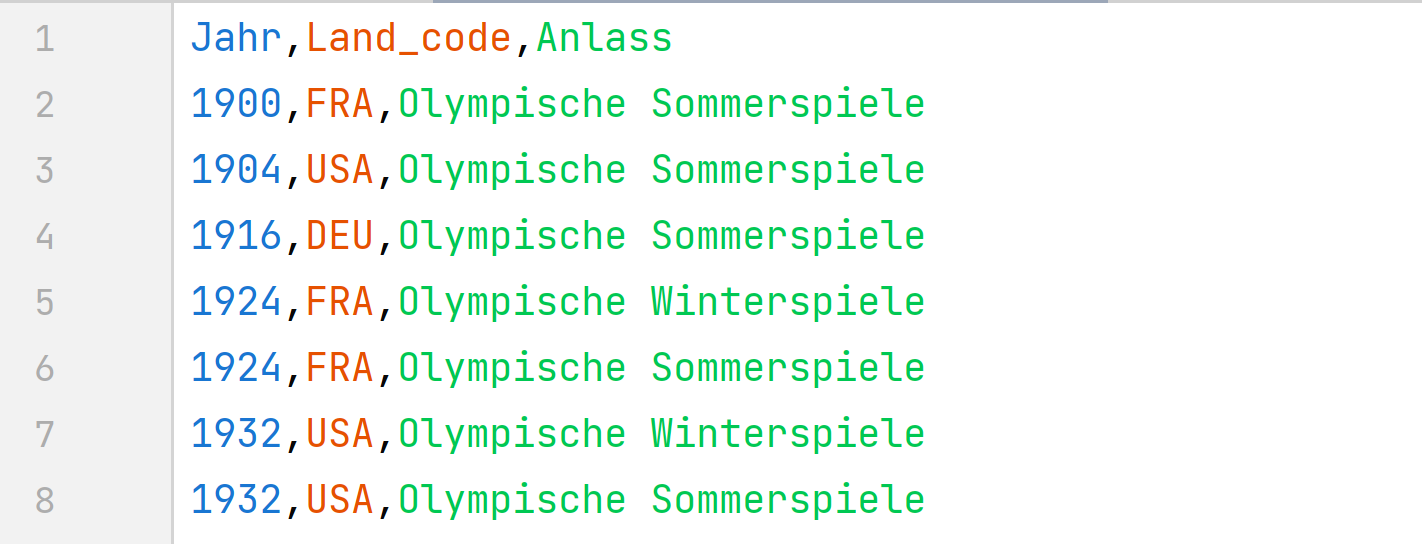
*Frage: Gibt es eine statistisch feststellbare, signifikante Korrelation zwischen der Durchführung von Sportgrossanlässen und der Veränderung des BIPs der Gastgebernation?*

*Lösung: Die ausgewählten Methoden liefern kein eindautiges Resultat, resp. ist eine Veränderung nicht nach visueller Inspektion feststellbar. Siehe Grafiken.*

Um die Frage 4 zu beantworten, werden die Datensets 'a1\_rgdpna\_stage.csv' und 'c2\_laendercode\_stage.csv' benötigt und zusammen gemerget. Der dritte Datenset 'fragestellung\_4\_top4\_laender.csv' ist eine Liste mit Sportevents und Jahr, der vier Nationen, die am häufigsten Gastgeber solcher Sportanlässe waren.

**Datensets**

**a1\_rgdpna\_stage.csv** **c1\_country\_stage.csv**

**fragestellung\_4\_top4\_laender.csv**

Die Datensets werden eingelesen, gemerget und im brauchbaren Format gebracht. Die nachfolgende Sequenz, hier am Beispiel der USA, wird für die Ländern Frankreich, Italien und Deutschland wiederholt.

1. Erstellen zweier, jeweils Länderspezifischen DF: df\_usa mit der time series des BIP und df\_date\_usa, mit einer Liste der Daten von Sportanlässen.

df\_usa = df\_j[df\_j[**'Land'**] == **'USA'**]  
df\_date\_usa = df\_4c[**'Jahr'**][(df\_4c.Land\_code == **"USA"**) & ((df\_4c.Jahr > 1949) & (df\_4c.Jahr < 2020))]

1. Danach wird mit der Time Series des BIP ein erstes Plot erstellt und als PNG abgespeichert.

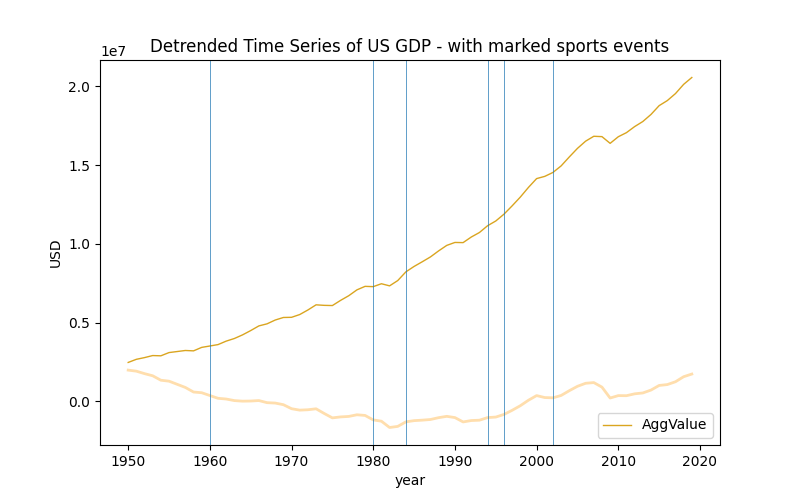
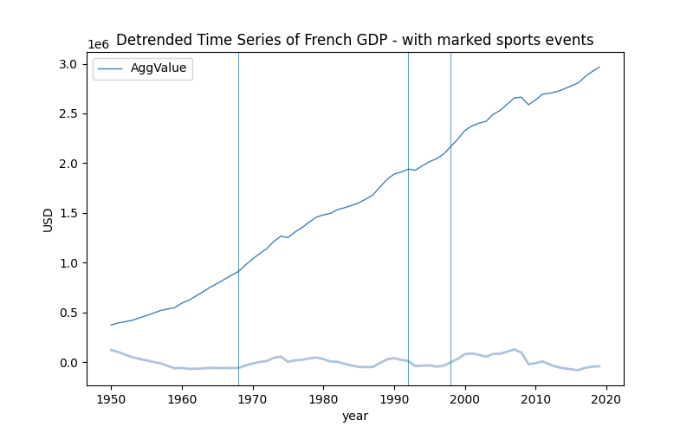
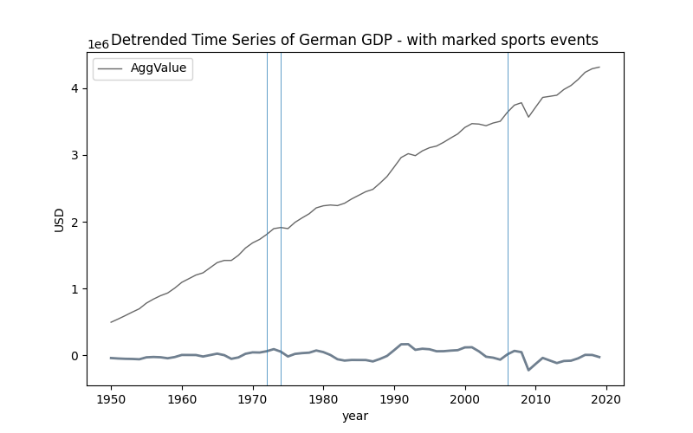
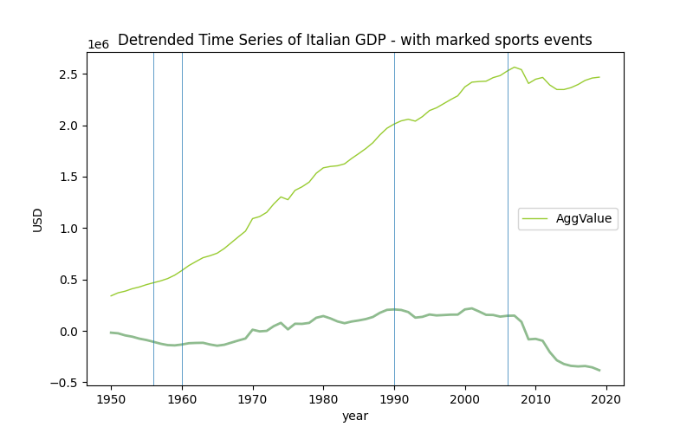
df\_usa.plot(x=**'YearCode'**, y=**'AggValue'**, marker=**''**, color=**'goldenrod'**, linewidth=1.0, figsize=(8,5))  
plt.xlabel(**'year'**)  
plt.ylabel(**'USD'**)  
plt.title(**'Time Series of US GDP'**)  
plt.savefig(**'fragestellung\_4\_usa\_bip\_time\_series'**)

1. Die Time Series wird dann entrendet

usa\_detrended = signal.detrend(df\_usa[**'AggValue'**])

1. Die entredete Time Series wird zum Schluss ebenfalls in einen Plot visulasiert. Dabei werden noch die Jahre, an denen einen Sportevent statt fand, als Linie eingefügt. Da die Funktion axvline() nur eine einzelne Linie im Plot einfügen vermag, muss eine For-Loop dieses Prozedere immer wiederholen.

plt.figure(1, figsize=(8, 5))  
plt.plot(df\_usa[**'YearCode'**], usa\_detrended, label=**"GDP\_detrended"**, color=**'navajowhite'**, linewidth=2)  
x = df\_date\_usa.tolist()  
for a in x: plt.axvline(a, linewidth=0.5)  
plt.title(**'Detrended Time Series of US GDP - with marked sports events'**)  
plt.savefig(**'fragestellung\_4\_usa\_time\_series\_analyse'**)  
plt.show()

Es liegen nun zwei Grafiken pro Land vor, eine Grafik mit dem BIP über die Zeitachse und eine mit der entrendeten Time Series. Folgend jeweils die zweite Grafik, mit der entrendeten Time Series und die Eventkennzeichnung. Die Grafik enthaltet ebenfalls die normale BIP Time Series. Diese unterscheidet sich durch ihren trendigen anstieg über die Zeit.

# Reflexion

Wenni ned no vier anderi Projekt am start hätt hätti glaubs no spass chönne ha

Haha! :D

# Einwilligung

Diese Arbeit beinhaltet keine vertraulichen Daten und darf frei geteilt werden.

1. CSV-Export [↑](#footnote-ref-1)
2. Tabelle unter Quickstart Query «RGDPNA (1950-2019)» [↑](#footnote-ref-2)