Name: Vorname:

Geburtsdatum: Matrikelnummer:

Platz:

Klausur
Informatische Werkzeuge in den Geistes- und Sozialwissenschaften 2

4. August 2022

	To be used for grading, do not write here										
prob.	1.1	1.2	2.1	3.1	3.2	4.1	4.2	4.3	5.1	Sum	grade
total	6	4	10	5	10	5	6	6	8	60	
reached											

Die „Lösungen“ der Aufgaben in diesem Dokument sollen den Studierenden als Anfangspunkt für die Beantwortung der Aufgaben dienen. Trotz aller Bemühungen kann es zu Unvollständigkeiten oder sogar Fehlern in den „Lösungen“ kommen. Da die Korrektur und Benotung der gestellten Aufgaben niemals lediglich auf einen „Vergleich mit der Musterlösung“ hinausläuft, ist dies auch nicht so schlimm. In jedem Fall sollten die Studierenden die Lösungen nachvollziehen und im Prinzip mittels des Lehrstoffs selbst verifizieren können.

Sollten Sie „Lösungen“ finden, die Sie nicht verstehen oder sogar für fehlerhaft halten diskutieren Sie diese am besten mit den Tutor*innen oder auf dem Kursforum und benachrichtigen Sie die Lehrenden; wir werden sie dann gegebenenfalls baldigst korrigieren.

1 Datenbanken

Aufgabe 1.1 (SQL Tabellenoperationen) 6 Pkt
Gegeben ist folgende Tabelle `AnimalActors`. Hierbei ist **ID** ein Primärschlüssel. 6 min

ID	Name	Tierart	Geburt	Tod	Produktion
1	Jimmy the Raven	Vogel	1934	1954	Der Zauberer von Oz
2	Mr Percival	Vogel	1976	2009	Storm Boy
3	Hercules	Bär	1975	2001	James Bond: Octopussy
4	Douglas	Vogel	1967	2019	Pippi und die Seeräuber
5	Keiko	Delfin	1976	2003	Free Willy
6	Pan-Kun	Affe	2001	`NULL`	Tensai! Shimura Dōbutsuen

Geben Sie jeweils die korrekten SQL-Befehle für folgende Operationen an:

1. Sie wollen der Tabelle einen weiteren Eintrag hinzufügen, für den Hund "Pal". Er wurde 1940 geboren, ist 1958 gestorben und spielte die Hündin "Lassie" in der Produktion "Lassie come home!".
2. Sie wollen die Namen (und *nur* die Namen) aller Tiere in der Datenbank einsehen, die im Jahr 2000 am leben waren (also vor/in 2000 geboren und nach/in 2000 gestorben sind).
3. Ein neues Gesetz verbietet Vogelvorratsdatenspeicherung. Sie wollen deshalb die Tabelle dahingehend verändern, dass keine Daten über Vögel mehr enthalten sind.

Solution: Hier sind sie entsprechenden Befehle:

```
1. INSERT INTO AnimalActors VALUES (7, "Pal", "Hund", "1940", "1958",
                                    "Lassie␣come␣home!");

2. SELECT Name FROM AnimalActors WHERE Geburt <= 2000 AND Tod >= 2000;

   (funktioniert fuer diese Tabelle, aber noch "richtiger" waere folgendes,
   immerhin koennten Tiere vor 2000 geboren sein und noch immer leben.)

   SELECT Name FROM AnimalActors WHERE Geburt <= 2000 AND
                                 (Tod >= 2000 OR Tod IS NULL);

3. DELETE FROM AnimalActors WHERE Tierart = "Vogel";
```

Aufgabe 1.2 (Primärschlüssel in der Praxis) 4 Pkt
Betrachten Sie abermals die Tabelle `AnimalActors` aus Aufgabe 1.1. Welche der Spalten dieser Tabelle eignen sich (außer ID) noch als Primärschlüssel für die abgebildete Tabelle? Begründen Sie Ihre Angaben. 4 min

---

Solution: Spalten, die als Primärschlüssel dienen sollen, dürfen keine doppelten Einträge enthalten (wie die Spalten **Geburt** und **Tod** in diesem Beispiel) und dürfen keine `NULL`-Einträge enthalten (wie die Spalte **Tod** in diesem Beispiel). Es kommen also nur die Spalte **Name** und **Produktion** als Primärschlüssel in Frage.

---

2 Bild

Aufgabe 2.1 (Doppelte Spiegelung) 10 Pkt
Schreiben Sie eine python-Funktion `double_mirror(img)` die ein Bild `img` als Parameter nimmt und eine veränderte Kopie des Bildes zurück gibt, die sowohl horizontal als auch vertikal gespiegelt wurde. 10 min

Dabei wird verlangt, dass beide Operationen "auf einmal" durchgeführt werden. Erstellen Sie also nicht erst eine in einer Achse gespiegelte Kopie welche Sie dann entlang der anderen Achse spiegeln. Verwenden Sie also auch *nicht* die vorgefertigten Funktionen im `Pillow`-Modul `ImageOps`.

Folgendes Codegerüst ist gegeben:

```
from PIL import Image

def double_mirror(img) :
    originalWidth = img.width
    originalHeight = img.height
    newWidth = originalHeight
    newHeight = originalWidth

    # Neues Bild erstellen.
    result = Image.new(mode="RGB", (newWidth, newHeight), 0)

    # Ihr Code hier...

    return result
```

Hinweis: Ein paar nützliche Informationen finden Sie im Folgenden:

- Sowohl der Parameter `img` als auch die Rückgabe Ihrer Funktion sind `Pillow`-Bilder.
- Sie können den Wert eines Pixels wie folgt auslesen:
  `img.getpixel((x, y))` gibt ein 3-Tupel `(r,g,b)` zurück.
- Sie können den Wert eines Pixels wie folgt setzen:
  `img.putpixel((x, y), value)`. `value` ist hier ebenfalls ein 3-Tupel.

Solution: Hier ist eine mögliche Implementation:

```
from PIL import Image

def double_mirror(img) :
    originalWidth = img.width
    originalHeight = img.height
    newWidth = originalHeight
    newHeight = originalWidth

    # Neues Bild erstellen.
    result = Image.new(mode="RGB", (newWidth, newHeight), 0)

    # Ihr Code hier...
    for y in range(0, originalHeight) :
        for x in range(0, originalWidth) :

            new_x = originalWidth - x
            new_y = originalHeight - y

            r, g, b = img.getpixel((x, y)) # Farbe im Inputbild.

            result.putpixel((new_x, new_y), (r,g,b))

    return result


img = Image.open("img.png")
rotated = fused_grey_rotate(img)
img.show()
rotated.show()
```

3 Web-Applikationen

Aufgabe 3.1 (Suffix-Listen) 5 Pkt

Schreiben Sie eine python-Funktion `suffixes(string)`, die ein einen herkömmlichen String als Argument entgegen nimmt und eine Liste von allen Suffixen des Strings zurück gibt. Ein Suffix eines Strings ist hier ein Teilstring bis zum Ende des ursprünglichen Strings (der String ‘‘`Maus`’’ hat also die Suffixe ‘‘`Maus`’’, ‘‘`aus`’’, ‘‘`us`’’ und ‘‘`s`’’). 5 min

Solution: Hier ist eine mögliche Implementation:

```
def suffixes(string):
    """Returns a list of all suffixes of the input"""

    # Start with empty list.
    suffix_list = []

    # Move starting position throught the whole string.
    for start in range(len(string)):
        # Add substring starting at position start, until end.
        suffix_list.append(string[start:])

    return suffix_list
```

Aufgabe 3.2 (Suffix-Listen) 10 Pkt

10 min

Schreiben Sie eine `bottle`-Route `/dictionary/<string>/` und die dazugehörige Funktion in `python`, die für den in der URL übergebenen Eingabestring eine HTML-Liste zurück gibt. Jedes Element dieser Liste soll ein Suffix des Eingabestrings sein.

Weiterhin soll das Element, neben dem Suffix einen Kommentar enthalten (z.B. einen String wie `"(Treffer!)"`), wenn das Suffix für sich genommen eines der 10000 häufigsten deutschen Wörter ist (so ist zum Beispiel `"ente"` ein Suffix von `"parlamente"`). Zu diesem Zweck liegt Ihnen eine Textdatei namens `top10000de.txt` vor, in der jeweils ein Wort pro Zeile aufgeführt ist.

Hinweis:

- Zur Erinnerung: HTML-Listen haben die Form `<ul><li>Element</li> ... </ul>`.
- Sie können für diese Aufgabe die Funktion `suffixes` aus Aufgabe 3.1 verwenden. Sollten Sie diese Aufgabe nicht vollendet haben, können Sie trotzdem davon ausgehen, dass Ihnen eine Funktion `suffixes` mit korrektem Verhalten zur Verfügung steht.
- Sie können ebenfalls davon ausgehen, dass für diese Aufgabe alle Strings (sowohl in der Eingabe als auch in der Datei) nur aus Kleinbuchstaben bestehen.

Solution: Hier ist eine mögliche Implementation:

```
from bottle import run, route

def suffixes(string):
    """Returns a list of all suffixes of the input"""

    # Start with empty list.
    suffix_list = []

    for start in range(len(string)):
        # Add substring starting at position start, until end.
        suffix_list.append(string[start:])

    return suffix_list

@route('/dictionary/<string>/')
def showAllandCheck(string):
    """Returns a html list with all suffixes of the input,
       with an additional comment if the suffix is in the
       dictionary file."""

    html = "<ul>"

    # Open dictionary file
    with open('top10000de.txt') as dfile:
        # Read contents into list
        dictionary = dfile.readlines()

        # compute suffixes
        suffix_list = suffixes(string)

        for suffix in suffix_list:
            if suffix in dictionary:
                comment = "␣(Treffer!)"
            else:
                comment = ""

            html += f"<li>{suffix}{comment}</li>"

    html += "</ul>"

    return html

run(host='0.0.0.0', port=32502)
```

4 Semantic Web & Kulturelles Erbe

Aufgabe 4.1 (SPARQL-Anfrage) 5 Pkt

5 min

Geben Sie eine vollständige SPARQL-Query für dbpedia oder eine vergleichbare Quelle an, die die Namen (und nur die Namen) aller Personen ausgibt, die in Hamburg geboren wurden und in Berlin gestorben sind. Die Ergebnisliste soll

nach dem Alter der Person sortiert sein (frühestes Geburtsjahr zuerst).

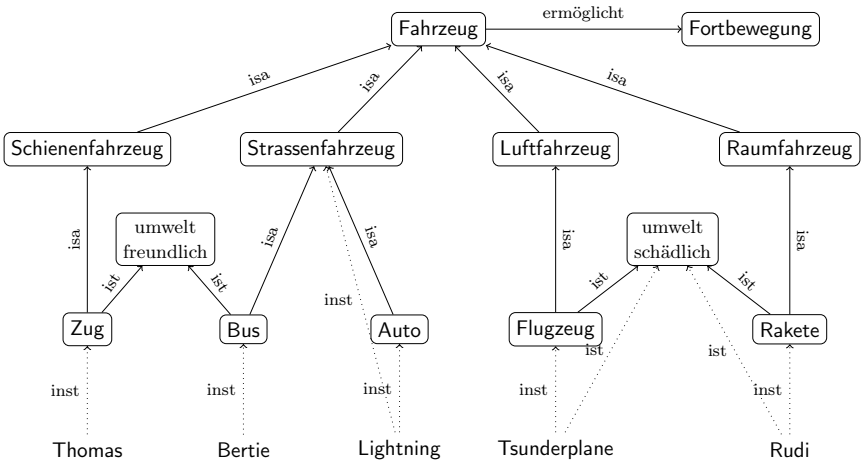
Hinweis: Wir erwarten nicht, dass Sie alle relevanten Ontologien für Properties wie "SterbeOrt" oder dergleichen auswendig können oder mitgebracht haben.

Sie können also die fiktive Ontologie UCO (für *"Universally Convenient Ontology"*) benutzen und sich relevante Object Properties dafür ausdenken. So könnte zum Beispiel der Sterbeort als die Property `uco:deathPlace` modelliert sein.

Solution: Hier ist eine mögliche Lösung (mit echten Ontologien statt UCO): (Funktioniert mit `https://dbpedia.org/snorql/`)

```
SELECT ?name WHERE {
    ?person dbo:deathPlace :Berlin .
    ?person dbo:birthPlace :Hamburg .
    ?person dbo:birthDate ?bd .
    ?person foaf:name ?name .
}
ORDER BY ?bd
```

Aufgabe 4.2 (Semantische Netze zu Fahrzeugen) 6 Pkt
Gegeben sei das folgende semantische Netz: 6 min



1. Geben Sie je drei Subjekt/Prädikat/Objekt-Tripel aus der TBox und der ABox dieses Netzes an.

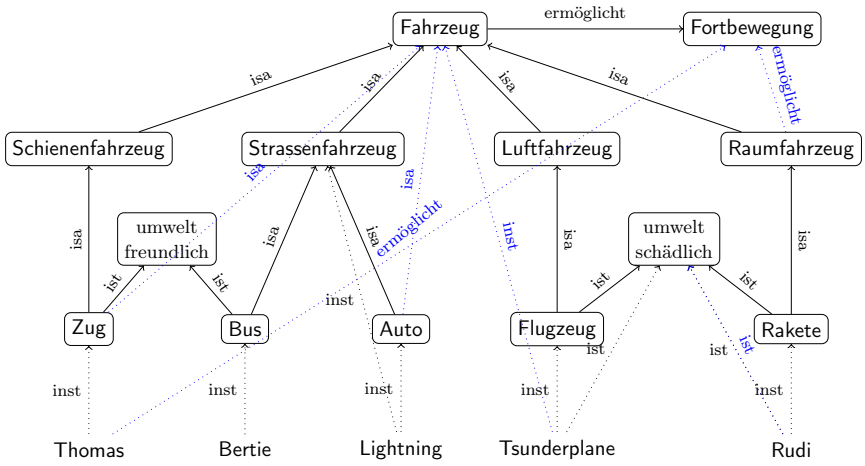
2. Zeichnen Sie jeweils drei Fakten und terminologische Axiome in das Netz ein, die im Netz inferiert werden können aber nicht explizit repräsentiert sind.

Solution:

1. Subjekt/Prädikat/Objekt

TBox	ABox
Bus/ist/umweltfreundlich	Tsunderplane/ist/umweltschädlich
Auto/isa/Strassenfahrzeug	Rudi/inst/Rakete
Fahrzeug/ermöglicht/Fortbewegung	Bertie/ermöglicht/Fortbewegung

2. Wir zeichnen die neuen Relationen in blau ein. Diese sind nicht vollständig, sondern lediglich Beispiele.



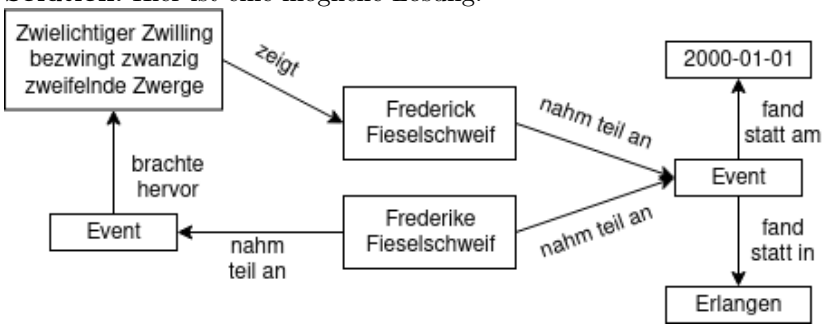
Aufgabe 4.3 (Ereignisorientierte Modellierung) 6 Pkt
Gegeben sind folgende zwei Aussagen: 6 min

Die Zwillinge Frederick und Frederike Fieselschweif
wurden am 01.01.2000 in Erlangen geboren.

Frederike Fieselschweif schuf das Bild
„Zwielichtiger Zwilling bezwingt zwanzig zweifelnde Zwerge“,
welches ihren Bruder zeigt.

Zeichnen Sie ein Netzwerk, welches diese Aussagen abbildet. Benutzen Sie dafür *“Ereignisorientierte Modellierung”* wie es auch im CIDOC-CRM verwendet wird (dafür können Sie passende selbst ausgedachte Relationen verwenden, es ist nicht notwendig mit CIDOC-CRM Relationen zu arbeiten).

Solution: Hier ist eine mögliche Lösung:



5 Geistiges Eigentum

Aufgabe 5.1 (Einbindung von CC-lizensierten Werken) 8 Pkt
8 min

Beatrice Beispiel hat diese beiden unter den angegebenen Creative Commons Lizenzen herausgegeben[1] Kunstwerke gefunden und möchte sie gerne in eigenen Projekten benutzen.

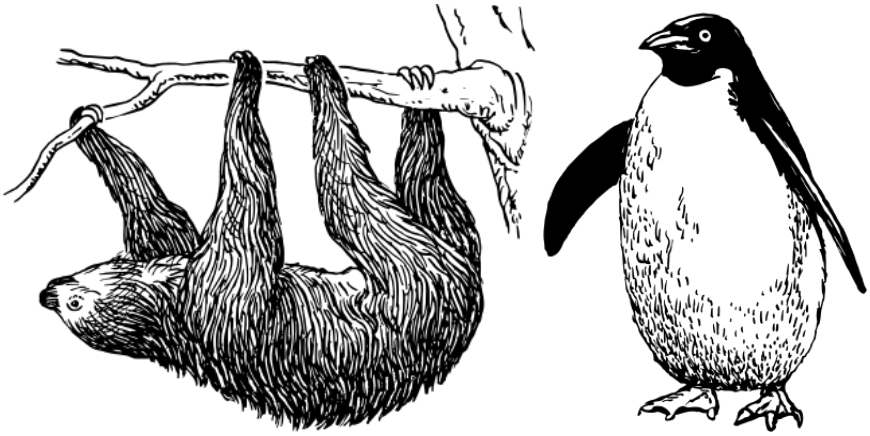


Abb. 1: Faultier hier unter CC-BY-SA, Pinguin hier unter CC-BY-NC

Beatrice hat zwei Projekte. Das erste Projekt ist ein Buch in dem es um Arbeitsmoral geht, welches unter CC-BY-NC-SA erscheinen soll. Das zweite Projekt ist eine digitale Collage zum Umweltschutz, die unter CC-BY erscheinen soll.

Geben Sie für alle vier Kombinationen an, ob das jeweilige Bild im jeweiligen Projekt benutzt werden darf. Sagen Sie auch kurz (!) warum / warum nicht.

[1] Wir nehmen die gegebenen Lizenzen nur für diese Aufgabe an, die wahren Lizenzen dieser Bilder sind anders.

Solution: Hier ist die entsprechende Tabelle.

	Buch	Collage
Faultier	Geht nicht NC darf wegen SA nicht hinzugefügt werden	Geht nicht SA muss erhalten bleiben
Pinguin	Geht da kein SA vorhanden ist, darf NC dazu	Geht nicht NC muss erhalten bleiben