

Name:

Vorname:

Geburtsdatum:

Matrikelnummer:

Platz:

Klausur
Informatische Werkzeuge in den
Geistes- und Sozialwissenschaften 2

25. Juli 2024

	To be used for grading, do not write here							
prob.	1.1	2.1	3.1	3.2	3.3	4.1	Sum	grade
total	20	16	5	5	4	10	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 (Korrektur von Datenbank-Befehlen)

20 Pkt

20 Min

Ihre gute Freundin Beatrice Beispiel ist auf den neuesten Trend aufgesprungen und hat sich eine eigene "KI" zusammengebastelt, die sie "IWGS-GPT" nennt und die ihr ab sofort einfach die Antwort auf alle Hausaufgaben vorsagen soll. Leider funktioniert das nicht wie geplant und sie braucht abermals Ihre Hilfe.

In den folgenden Aufgaben interagiert Beatrice mit den folgenden Datenbanktabellen. Die Spalten **book_id** und **author_id** sind hierbei Primärschlüssel ("primary keys") und die Spalte **author_id** der Tabelle **books** ist ein Fremdschlüssel ("foreign key"), welcher auf die gleichnamige Spalte in der Tabelle **authors** verweist.

Identifizieren Sie für jede Teilaufgabe den Fehler in dem von Beatrice generierten SQL-Befehl und geben Sie eine korrigierte Version des Befehls an, die tatsächlich tut, was sie erreichen wollte.

book_id	title	author_id	published_year	number_of_pages
1	Einzigartiges Tierreich	1	2015	1200
2	Zweisamkeit auf Ganymede	2	2018	900
3	Dreiste Diamantendiebe	3	2011	1500
4	Viertel Vor Viel Zu Spät	1	2021	1100

Tabelle 1: Books

author_id	name	genre
1	Elisabeth Eins	Thriller
2	Zoey Zwei	Romantik
3	Dorian Drei	Sci-Fi

Tabelle 2: Authors

1. **Ziel:** Beatrice möchte alle Bücher neuer als 2016 und ihre Seitenzahlen abfragen.

Befehl: IWGS-GPT schlägt folgenden Befehl vor:

```
SELECT book_title, number_of_pages
FROM books
WHERE published_year > 2016;
```

Lösung: Das Problem hier ist, dass es keine Spalte **book_title** in der Tabelle gibt.

Richtig wäre:

```
SELECT title, number_of_pages
FROM books
WHERE published_year > 2016;
```

Bewertung: Jeweils 2 Punkte für das korrekte Identifizieren eines Fehlers als auch für den korrigierten SQL-Befehl.

AC1 Fehler erkannt

Punkte: +2.0

Rückmeldung: Fehler in der Vorgabe wurde korrekt benannt.

AC2 Fehler halb erkannt.

Punkte: +1.0

Rückmeldung: Fehler in der Vorgabe wurde benannt, aber Argumentation enthält Fehler

- AC3 Fehler nicht erkannt
Punkte: -0.0
Rückmeldung: Fehler in der Vorgabe wurde nicht korrekt benannt.
- AC4 SQL korrekt
Punkte: +2.0
Rückmeldung: Korrigierter SQL-Befehl ist korrekt.
- AC5 SQL mit Fehlern
Punkte: +2.0
Rückmeldung: Korrigierter SQL-Befehl ist vorhanden aber nicht komplett korrekt.
- AC6 SQL falsch
Punkte: +2.0
Rückmeldung: Korrigierter SQL-Befehl ist falsch.
- AC7 Kein SQL vorhanden
Punkte: +2.0
Rückmeldung: Kein korrigierter SQL-Befehl vorhanden.
-
-

2. **Ziel:** Beatrice möchte den Autor “Vladimir Vier” hinzufügen.

Befehl: IWGS-GPT schlägt folgenden Befehl vor:

```
INSERT INTO authors (author_id, name, genre)
VALUES (NULL, 'Vladimir Vier', 'Horror');
```

Lösung: Hier ist das Problem, dass ein Primärschlüssel niemals NULL sein darf. Besser wäre:

```
INSERT INTO authors (author_id, name, genre)
VALUES (4, 'Vladimir Vier', 'Horror');
```

Bewertung: Jeweils 2 Punkte für das korrekte Identifizieren eines Fehlers als auch für den korrigierten SQL-Befehl.

- AC1 Fehler erkannt
Punkte: +2.0
Rückmeldung: Fehler in der Vorgabe wurde korrekt benannt.
- AC2 Fehler halb erkannt.
Punkte: +1.0
Rückmeldung: Fehler in der Vorgabe wurde benannt, aber Argumentation enthält Fehler
- AC3 Fehler nicht erkannt
Punkte: -0.0
Rückmeldung: Fehler in der Vorgabe wurde nicht korrekt benannt.
- AC4 SQL korrekt
Punkte: +2.0
Rückmeldung: Korrigierter SQL-Befehl ist korrekt.
- AC5 SQL mit Fehlern
Punkte: +2.0
Rückmeldung: Korrigierter SQL-Befehl ist vorhanden aber nicht komplett korrekt.
- AC6 SQL falsch
Punkte: +2.0
Rückmeldung: Korrigierter SQL-Befehl ist falsch.
- AC7 Kein SQL vorhanden
Punkte: +2.0
Rückmeldung: Kein korrigierter SQL-Befehl vorhanden.

3. **Ziel:** Dieses Mal ist Beatrice nur am Titel des neuesten Buches interessiert.

Befehl: IWGS-GPT schlägt folgenden Befehl vor:

```
SELECT title
FROM books
ORDER BY published_year ASC
LIMIT 1;
```

Lösung: Beatrice sortiert hier die Ergebnisse falsch herum. Wenn sie nur ein Ergebnis aus einer aufsteigend (ASC = ascending = aufsteigend) sortierten Liste abfragt, bekommt sie das Ergebnis mit dem niedrigsten Wert, also das *älteste* Buch. Besser wäre eine absteigend (DESC = descending = absteigend) sortierte Liste.

```
SELECT title
FROM books
ORDER BY published_year DESC
LIMIT 1;
```

Bewertung: Jeweils 2 Punkte für das korrekte Identifizieren eines Fehlers als auch für den korrigierten SQL-Befehl.

AC1 Fehler erkannt

Punkte: +2.0

Rückmeldung: Fehler in der Vorgabe wurde korrekt benannt.

AC2 Fehler halb erkannt.

Punkte: +1.0

Rückmeldung: Fehler in der Vorgabe wurde benannt, aber Argumentation enthält Fehler

AC3 Fehler nicht erkannt

Punkte: -0.0

Rückmeldung: Fehler in der Vorgabe wurde nicht korrekt benannt.

AC4 SQL korrekt

Punkte: +2.0

Rückmeldung: Korrigierter SQL-Befehl ist korrekt.

AC5 SQL mit Fehlern

Punkte: +2.0

Rückmeldung: Korrigierter SQL-Befehl ist vorhanden aber nicht komplett korrekt.

AC6 SQL falsch

Punkte: +2.0

Rückmeldung: Korrigierter SQL-Befehl ist falsch.

AC7 Kein SQL vorhanden

Punkte: +2.0

Rückmeldung: Kein korrigierter SQL-Befehl vorhanden.

4. **Ziel:** Dieses Mal möchte Beatrice den Titel und Namen der Autor*Innen für alle Bücher abfragen, die nach 2010 erschienen sind.

Befehl: IWGS-GPT schlägt folgenden Befehl vor:

```
SELECT title, name
FROM books, authors
```

```
WHERE published_year > 2010;
```

Lösung: Es werden hier Daten aus zwei Tabellen abgefragt, aber es ist nicht möglich, das einfach durch eine Liste von Tabellennamen im FROM-Teil der Anfrage zu realisieren. Hier braucht es einen richtigen JOIN.

```
SELECT b.title, a.name
FROM books b JOIN authors a
ON b.author_id = a.author_id
WHERE b.published_year > 2010;
```

Bewertung: Jeweils 2 Punkte für das korrekte Identifizieren eines Fehlers als auch für den korrigierten SQL-Befehl.

AC1 Fehler erkannt

Punkte: +2.0

Rückmeldung: Fehler in der Vorgabe wurde korrekt benannt.

AC2 Fehler halb erkannt.

Punkte: +1.0

Rückmeldung: Fehler in der Vorgabe wurde benannt, aber Argumentation enthält Fehler

AC3 Fehler nicht erkannt

Punkte: -0.0

Rückmeldung: Fehler in der Vorgabe wurde nicht korrekt benannt.

AC4 SQL korrekt

Punkte: +2.0

Rückmeldung: Korrigierter SQL-Befehl ist korrekt.

AC5 SQL mit Fehlern

Punkte: +2.0

Rückmeldung: Korrigierter SQL-Befehl ist vorhanden aber nicht komplett korrekt.

AC6 SQL falsch

Punkte: +2.0

Rückmeldung: Korrigierter SQL-Befehl ist falsch.

AC7 Kein SQL vorhanden

Punkte: +2.0

Rückmeldung: Kein korrigierter SQL-Befehl vorhanden.

5. **Ziel:** Zum Schluss möchte Beatrice beide Tabellen komplett löschen.

Befehl: IWGS-GPT schlägt folgenden Befehl vor:

```
DELETE FROM books;
DELETE FROM authors;
```

Lösung: Diese Befehle löschen alle *Einträge* in den Tabellen, aber nicht die Tabellen als solche.

Richtig ist:

```
DROP TABLE books;
DROP TABLE authors;
```

Bewertung: Jeweils 2 Punkte für das korrekte Identifizieren eines Fehlers als auch für den korrigierten SQL-Befehl.

AC1 Fehler erkannt

Punkte: +2.0

Rückmeldung: Fehler in der Vorgabe wurde korrekt benannt.

AC2 Fehler halb erkannt.

Punkte: +1.0

Rückmeldung: Fehler in der Vorgabe wurde benannt, aber Argumentation enthält Fehler

AC3 Fehler nicht erkannt

Punkte: -0.0

Rückmeldung: Fehler in der Vorgabe wurde nicht korrekt benannt.

AC4 SQL korrekt

Punkte: +2.0

Rückmeldung: Korrigierter SQL-Befehl ist korrekt.

AC5 SQL mit Fehlern

Punkte: +2.0

Rückmeldung: Korrigierter SQL-Befehl ist vorhanden aber nicht komplett korrekt.

AC6 SQL falsch

Punkte: +2.0

Rückmeldung: Korrigierter SQL-Befehl ist falsch.

AC7 Kein SQL vorhanden

Punkte: +2.0

Rückmeldung: Kein korrigierter SQL-Befehl vorhanden.

2 Bild

Aufgabe 2.1 (Kantenerkennung)

16 Pkt

In dieser Aufgabe geht es um automatische Kantenerkennung in Bildern, wie in der Vorlesung besprochen. Dafür werden wir in einem ersten Schritt ein gegebenes Bild in Graustufen umwandeln. In einem zweiten Schritt werden danach die Kanten gefunden.

16 Min

Hinweis: Ein paar nützliche Informationen finden Sie im Folgenden:

- Sowohl die Parameter `img` als auch die Rückgabewerte Ihrer Funktionen sind Pillow-Bilder.
- Mit `img.copy()` können Sie eine exakte Kopie eines Bildes erstellen. Diese lässt das Original unangetastet.
- 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), (r,g,b))`
- Folgende Funktionen sind gegeben und können von Ihnen verwendet werden:

```
def grey_v(trip):
    """Returns greyscale value of a pixel."""
    g = int(0.21*trip[0] + 0.71*trip[1] + 0.08*trip[2])
    return (g,g,g)

def avg(trip):
    """Returns the average colour value of a pixel."""
    return (trip[0]+trip[1]+trip[2])/3
```

-
1. Schreiben Sie eine Python-Funktion `greyscale(img, threshold)`, die ein Bild `img` als Parameter nimmt und eine Kopie des Inputs in Graustufen zurück gibt. Nutzen Sie dazu *nicht* die entsprechende Methode aus der `Pillow`-Bibliothek.

Lösung: Hier ist eine mögliche *Implementation*:

```
def greyscale(img):
    """Returns a greyscale copy of the image"""
    grey = img.copy()

    for x in range(img.width):
        for y in range(img.height):
            colour = img.getpixel((x,y))
            grey.putpixel((x,y), grey_v(colour))

    return grey
```

Bewertung: Korrekte Bearbeitung dieser Aufgabe gibt 6 Punkte.

2. Schreiben Sie eine Python-Funktion `edge_detection(img, threshold)`, die ein Bild `img` und eine Zahl `threshold` als Parameter nimmt. Dieses Bild soll überall dort, wo keine Kanten sind, weiß sein. Die Pixel, die zu einer Kante gehören, sollen schwarz eingefärbt werden. Verwenden Sie *nicht* die `Pillow`-Funktion `img.filter`.

Wir gehen davon aus, dass eine Kante vorliegt, wenn der Unterschied im Farbwert zwischen dem unteren und oberen Nachbarpixel und/oder zwischen dem linken und rechten Nachbarpixel den Grenzwert übersteigt. Für diese Aufgabe genügt es, wenn Sie die direkten Nachbarn eines Pixels betrachten, die diagonalen Nachbarn können Sie ignorieren.

Lösung: Hier ist eine mögliche *Implementation*:

```
def edge_detection(img, tau):
    """Returns copy of input, with edges black, non-edges white"""
    edges = img.copy()
    grey = greyscale(img)
    white = (255,255,255)
    black = (0,0,0)

    for x in range(img.width):
        for y in range(img.height):
            edges.putpixel((x,y), white)

    for x in range(1,(img.width-1)):
        for y in range(1,(img.height-1)):
            p_t = avg(grey.getpixel((x,y+1)))
            p_r = avg(grey.getpixel((x+1,y)))
            p_b = avg(grey.getpixel((x,y-1)))
            p_l = avg(grey.getpixel((x-1,y)))

            hor = p_b - p_t > tau
            ver = p_r - p_l > tau

            if hor or ver:
                edges.putpixel((x,y), black)
```



```
return edges
```

Bewertung: Korrekte Bearbeitung dieser Aufgabe gibt 10 Punkte.

3 Semantische Netzwerke & Kulturelles Erbe

Aufgabe 3.1 (SPARQL-Anfrage Entwerfen)

5 Pkt

Geben Sie eine vollständige SPARQL Query für DBpedia oder eine vergleichbare Quelle an, die alle Fußballspieler*Innen abfragt, die aktuell in einem Team in einem anderen Land spielen, als in ihrem Geburtsland.

5 Min

Hinweis: Wir erwarten nicht, dass Sie alle relevanten Ontologien für spezialisierte Symbole wie zum Beispiel "Fußballspieler*In" 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 Typ Fußballspieler*In als `uco:SoccerPlayer` modelliert sein.

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

```
SELECT DISTINCT *
{
  # 'a' ist in SPARQL eine Kurzschreibweise von 'rdf:type'
  ?soccerplayer a          dbo:SoccerPlayer .
  ?soccerplayer dbo:birthPlace ?countryOfBirth .
  ?soccerplayer dbo:team      ?team .
  ?team          dbo:ground    ?countryOfTeam .

  FILTER (?countryOfTeam != ?countryOfBirth)
}
ORDER BY ?soccerplayer
```

Bewertung: Jeweils einen Punkt für das Modellieren von Beruf, Geburtsort und aktuellen Spielort. Die verbleibenden zwei Punkte für den Filter (oder sonstige Festsetzung), dass der Geburtsort und der Spielort unterschiedlich sein müssen.

AC1 Beruf formalisiert.

Punkte: +1.0

Rückmeldung: Der Beruf Fußballspieler*In wurde erfolgreich formalisiert.

AC2 Geburtsort formalisiert.

Punkte: +1.0

Rückmeldung: Der Geburtsort wurde erfolgreich formalisiert.

AC3 Spielort formalisiert.

Punkte: +1.0

Rückmeldung: Der Ort an dem aktuell gespielt wird wurde erfolgreich formalisiert.

AC4 Geburtsort is not Spielort

Punkte: +2.0

Rückmeldung: Es wurde festgesetzt, dass Geburtsort und Spielort nicht gleich sein dürfen.

Aufgabe 3.2 (Ontologien)

5 Pkt

Erklären Sie in wenigen Sätzen den Begriff einer *Ontologie* im Kontext von SPARQL-Anfragen. Können zwei sonst gleich formulierte SPARQL-Anfragen an die gleiche Datenbank unterschiedliche Ergebnisse haben, wenn sie lediglich unterschiedliche Ontologien (für die gleichen Sachverhalte, z.B. `dbo:genre` und `dbp:genre` für das Genre eines Musikstücks) verwenden? Warum / Warum nicht?

5 Min

Lösung: Eine Ontologie ist eine formale Spezifikations eines Modells, normalerweise ein Modell über Wissen von (bestimmten Teilen) der Welt. Was eine Ontologie behandelt wird oft ihre Domäne genannt. Ontologien führen das Vokabular ein, mit dem dann formal über Konzepte, Objekte und ihre Relationen untereinander geredet werden kann und verankert dessen Semantik.

Wenn sich zwei Ontologien (und folglich zwei Modelle der Welt) unterscheiden, kann es bei SPARQL-Anfragen, die diese nutzen, sehr wohl zu unterschiedlichen Ergebnissen kommen. Ein einfaches Beispiel wäre, dass die Semantik für die Anzahl an Einwohner*Innen einer Stadt von Ontologie zu Ontologie unterscheiden kann (z.B. nur Innenstadt vs. inkl. Umland), was selbstverständlich zu unterschiedlichen Ergebnissen führen würde.

Aufgabe 3.3 (SPARQL-Anfrage Erklären)

4 Pkt

Gegeben ist folgende SPARQL Query für DBPedia. Beschreiben Sie kurz aber genau, was genau hier abgefragt wird. Natürlich müssen Sie nicht das konkrete Ergebnis nennen, beschreiben Sie stattdessen die Eigenschaften die es aufweist, um von dieser Anfrage ausgewählt zu werden.

4 Min

```
# PREFIXes omitted.
```

```
SELECT ?t WHERE {
  ?b rdf:type          dbo:Book .
  ?b dbo:numberOfPages ?pg .
  ?b dbo:releaseDate  ?rd .
  ?b dbp:title        ?t .
  ?b dbo:isbn          ?i .

  FILTER (?pg >= 750)
  FILTER (YEAR(?rd) >= 2000)
}
ORDER BY DESC(?i)
LIMIT 1
```

Lösung: Wir suchen den Titel des Buches mit der höchsten ISBN-Nummer, welches mindestens 750 Seiten hat und seit dem Jahr 2000 erschienen ist.

Bewertung:

- AC1 Titel erkannt
Punkte: +1.0
Rückmeldung: Es wurde erfolgreich erkannt, dass wir den Titel eines Buches suchen
- AC2 ISBN erkannt
Punkte: +1.0
Rückmeldung: Es wurde erfolgreich erkannt, dass wir die höchste ISBN-Nummer suchen
- AC3 Seitenzahl erkannt
Punkte: +1.0
Rückmeldung: Es wurde erfolgreich erkannt, dass das Buch mehr als 750 Seiten haben soll.
- AC4 Datum erkannt
Punkte: +1.0
Rückmeldung: Es wurde erfolgreich erkannt, dass das Buch in diesem Jahrhundert erschienen sein soll.
- AC5 Falsche Sortierung
Punkte: -0.5
Rückmeldung: Sortierung von aufsteigend/absteigend wurde verwechselt.
- AC6 Limit nicht beachtet
Punkte: -0.5
Rückmeldung: Es wurde übersehen, dass wir nach dem Titel von genau einem Buch suchen.
-
-

4 Geistiges Eigentum

Aufgabe 4.1 (Creative Commons Tabelle)

10 Pkt

10 Min

Beatrice meldet sich abermals mit einer Frage nach Lizenzen. Sie hat die folgenden vier Kunstwerke zu den angegebenen Creative Commons-Lizenzen¹ gefunden und möchte gerne möglichst viele davon in ihrem nächsten Projekt (ein Kinderbuch über IWGS-GPT) benutzen. Sie ist sich allerdings nicht sicher, welche Lizenz sie dann verwenden müsste. Sie lässt die Bitte unausgesprochen und bedankt sich nur für die Unterstützung.



Abbildung 1:
Rabe: CC0



Abbildung 2:
Seepferd: CC-BY-SA



Abbildung 3:
Fuchs: CC-BY-ND

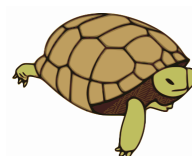








Abbildung 4:
Schildkröte: CC-BY-NC

Füllen Sie die nicht ausgegrauten Zellen der folgenden Tabelle mit einer Creative Commons Lizenz, unter der ein Remix der beiden betreffenden Werke lizenziert werden dürfte. Ist dies nicht möglich, setzen Sie bitte ein X.

Beantworten Sie außerdem kurz (!) die folgende Frage: Gibt es eine Lizenz, unter der ein Remix von allen vier Kunstwerken lizenziert werden könnte? Wenn ja, welche? Wenn nicht, warum nicht?

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

Lösung:

- **Rabe + Seepferd:** CC0 + CC BY-SA muss (wegen SA) wieder CC BY-SA sein.
- **Rabe + Fuchs:** CC0 + CC BY-ND darf nicht geremixt werden (wegen ND).
- **Rabe + Schildkröte:** CC0 + CC BY-NC kann wieder CC BY-NC sein oder auf Wunsch noch ND oder SA hinzufügen.
- **Seepferd + Fuchs:** CC BY-SA + CC BY-ND darf nicht geremixt werden (wegen ND)
- **Seepferd + Schildkröte:** CC BY-SA + CC BY-NC muss wegen SA wieder unter CC BY-SA veröffentlicht werden.
- **Fuchs + Schildkröte:** CC BY-ND und CC BY-NC darf nicht geremixt werden (wegen ND).

Es gibt keine Lizenz, die einen Remix für alle vier Kunstwerke erlaubt, da eine der Lizenzen explizit jede Sorte von Derivaten verbietet (der Fuchs mit CC BY-ND).

Bewertung: 1.5 Punkte für jedes korrekt ausgefüllte Feld der Tabelle (insgesamt 9 Punkte). Der letzte verbleibende Punkt für die Abwesenheit der Lizenz für alle.

AC1 Rabe + Seepferd korrekt

Punkte: +1.5

Rückmeldung: Korrekte Angabe der Lizenz für Rabe + Seepferd

AC2 Rabe + Fuchs korrekt

Punkte: +1.5

Rückmeldung: Korrekte Angabe der Inkompatibilität von Rabe + Fuchs

AC3 Rabe + Schildkröte korrekt

Punkte: +1.5

Rückmeldung: Korrekte Angabe der Lizenz für Rabe + Schildkröte

AC4 Seepferd + Fuchs korrekt

Punkte: +1.5

Rückmeldung: Korrekte Angabe der Inkompatibilität von Seepferd + Fuchs

- AC5 Seepferd + Schildkröte korrekt
Punkte: +1.5
Rückmeldung: Korrekte Angabe der Lizenz für Seepferd + Schildkröte
- AC6 Fuchs + Schildkröte korrekt
Punkte: +1.5
Rückmeldung: Korrekte Angabe der Inkompatibilität von Fuchs + Schildkröte
- AC7 Keine Lizenz für alle korrekt
Punkte: +1.0
Rückmeldung: Korrekt erkannt dass es keine Lizenz für alle vorliegenden Werke gibt.
- AC8 Keine Begründung für Alle
Punkte: -0.5
Rückmeldung: Es wurde keine Begründung genannt, warum es keine Lizenz für alle Kunstwerke zusammen gibt.
- AC9 Falsche Lizenz für Alle
Punkte: -0.0
Rückmeldung: Es wurde inkorrektweise eine Lizenz für alle Kunstwerke zusammen angegeben.
- AC10 Lizenz für Alle ausgelassen
Punkte: -0.0
Rückmeldung: Die Frage nach der Lizenz für alle Kunstwerke gemeinsam wurde übersprungen.
-
-