Maximilian Stock, Daniel Motz
Gruppenname: *„BECKS“*

Knowledge Graphs ÜS 1

Competency Questions

- Welche Toppings hat eine Pizza Hawaii?
- Welche Teigsorten gibt es?
- Welche Saucen können auf einer Pizza sein?
- Kann ich Brokkoli auf eine Pizza Hawaii legen?
- Ist eine Pizza Hawaii ohne Ananas immer noch eine Pizza Hawaii?
- Was gehört zu einer Pizza Margherita?
- Welche Zutaten hat eine Calzone?
- Welche Sorten von Pizza gibt es?
- Welche Pizzen sind vegetarisch?
- Welche Pizzen sind vegan?
- Gibt es glutenfreie Pizzen?
- Welche Pizzen enthalten Mozzarella?
- Welche Pizzen gibt es bei Mekan?
- Wie wird die „Pizza FCC“ bei Mekan gemacht?
- Wie viel kostet eine Pizza Frutti di Mare bei Giovannis Pizzeria?
- Welche Maße hat die Pizza?
- Welche Form hat die Pizza?
- Welche Pizza bei Mekan ist am billigsten?
- Welche Pizzen enthalten Schinken?
- Welche laktosefreien Pizzen gibt es?

Derived Terms

- **Pizza**
- **Toppings → Zutaten**
- **Sauce**
- Sorten von Pizza → Unterklassen von Pizza mit Constrains, also „geschützte Sorten“ wie „Pizza Hawaii“
- **vegetarisch**
- **vegan**
- **glutenfrei**
- Mekan → **Lokalität**
- **Form** / **Maße**
- **Preis**
- Laktosefrei → **Milchproduktzutatsklasse**

Maximilian Stock, Daniel Motz
Gruppenname: *„BECKS“*

Terms and Classes

Unsere Ontologie modelliert nicht konkrete, real existierende Pizzen, sondern Einträge einer Speisekarte. Anfangs modellierten wir eine am Prozess der Fastfoodkette „Dominos“ orientierte Kategorisierung. Wir stellten jedoch fest, dass dies aufwändig ist. Wir stellten jedoch die Anforderung, dass die Ontologie sowohl Pizzen die in einer Pizzeria, als auch für solche, die in den „eigenen vier Wänden“, produziert werden, abbilden kann. Die Klasse Pizza umfasst daher die für eine Pizza wesentlichen Eigenschaften, wie etwa eine Zutatenliste, jedoch ist die Eigenschaft „Preis“ nur im Zusammenhang mit einer Pizzeria präsent.

Wir entschieden außerdem, dass weit verbreitete Rezepturen, wie etwa „Pizza Hawaii“, als Unterklasse von **Pizza** mit Einschränkungen im Wertebereich der Zutaten modelliert werden. Dafür ist es notwendig, dass wir entweder Klassen für Zutaten anlegen, wie etwa „Schinken“ oder aber wir liefern zu unserer TBox eine ABox. Wir haben uns für letzteres entschieden. Dies entstand aus der CQ „Kann man eine Pizza Hawaii ohne Ananas herstellen“.

Wir haben uns außerdem dazu entschieden, dass gewisse Zutaten eine eigene Klasse erhalten, nämlich Teig, Sauce und Käse, um erzwingen zu können, dass eine Pizza mindestens aus einem Teig und einer Sauce bestehen muss.

Zu Pizzerien haben wir folgende Gedanken: Eine Pizza kann bei einer Pizzeria hergestellt werden. Dann wird sie zu einem konkreten Preis verkauft und der Preis variiert zudem mit der Größe. In unserer Feldstudie im Raum Jena stellten wir fest, dass es Pizzerien gibt, die an ihrer Speisenkartentafel eigens Klassifizierungen vornehmen. So fanden wir beispielsweise die Aufschrift „Alle Pizzen mit Tomatensauce und Käse[g]“. Diese Hilfestellung würdigen wir durch Aufnahme in unsere Ontologie als eine eigene Klasse „**Pizza_bei_Mekan**“. Wir wissen, a posteriori, dass man auch bei Mekan keine Pizza ohne Teig herstellt, auch wenn der Speisekartenersteller dies nicht expliziert hat.

Anfangs modellierten wir „Toppings“ bzw. Beläge (bspw. Schinken, Brokkolli, Blumenkohl und Zwiebeln, aber auch Saucen) in der Klasse **Zutat** und fügten data properties für wie `istGlutenfrei` ein. Dies ersetzten wir durch Typen wie etwa `Glutenfreie`, `Vegetarische` und `Vegane`.

Wir haben Teig nicht als vegan modelliert, damit wir ggf. auch Teige mit Ei erlauben können. Die Einschränkung in vegetarisch erscheint sinnvoll, jedoch wird die Zubereitung des Teigs in unserer Ontologie nicht näher spezifiziert.

Unsere Ontologie enthält ein Beispiel für eine unerfüllbare Klasse: `VegetarischePizza_Hawaii`, modelliert als Schnitt zwischen `Vegetarische_Pizza` und `Pizza_Hawaii`. Die Bedingungen sind also, dass jedes Individuum dieses Typs ausschließlich vegetarische Zutaten enthält, sowie die für eine Pizza Hawaii notwendigen (Ananas, Tomatensauce, Teig/Boden und Schinken). Da Schinken nicht den Typ `Vegetarische` besitzt, kann es die Anforderungen der Klasse `Vegetarische_Pizza` nicht erfüllen.

Informelle Hierarchisierung

Die Datei befindet sich im Repositorium unter `terminology-hierarchy.pdf`.

Ontologie

Die Datei befindet sich ebenfalls im Repositorium unter `ontology.xml`.

„[g]Milch“

Maximilian Stock, Daniel Motz
Gruppenname: *„BECKS“*

OntoClean: Rigidity, Unity, Identity

Wir haben festellen müssen, dass die Validierung von Rigidity, Identity und Unity für die Klassen unserer Ontologie weder trivial, noch (zumindest in einigen Fällen) eindeutig ist. Die folgenden Modellierungen haben wir nach bestem Wissen und Gewissen derart vorgenommen, sodass diese einerseits widerspruchsfrei sind und andererseits (aus unserer Sicht) dem Zweck der Ontlogie entsprechen.

Rigidity

Von den Competency Questions ausgehend war es sinnvoll für Klassen wie **Pizza_Hawaii** oder bspw. **Pizza_bei_Mekan** anti-rigidity ($\sim R$) zu modellieren, denn man kann bspw. bei der Bestellung einer Pizza bestimmte Zutaten weglassen oder dazuwählen. Da jedoch eine Pizza Hawaii insbesondere durch die Zutat „Ananas“ charakterisiert wird, sollte mindestens diese Zutat wesentlich für die Klassenzugehörigkeit sein. Eine Pizza ist allerdings noch dieselbe, wenn jemand die Ananas von ihr herunterisst, jedoch dann keine Pizza Hawaii mehr.

Die Klasse **Zutat** kann ebenso kritisch bewertet werden. Brokkolli muss nicht notwendigerweise Zutat einer Pizza sein. Im Rahmen der Competency Questions ist diese Unterscheidung irrelevant. Für eine kurze Zeit überlegten wir „Zutat“ als Rolle zu definieren, sodass ein Produkt, bspw. Brokkoli, seine Eigenschaft als Zutat verlieren kann, sofern er keine Zutat bei einer Pizza ist. **Zutat** erhält daher $+R$, wie auch alle Unterklassen. Etwa als wir die property `enthaeltZutat` modellierten, war für die Beschreibung der range restriction notwendig die Vereinigung aller möglichen Zutatenklassen zu finden. Mit unserer Zutatenklassifizierung gab es jedoch Zutaten, die in keine Klasse fielen. Ein Beispiel hierfür ist Hinterkochschinken – er enthält Gluten, ist nicht vegetarisch, kein Teig und so weiter. Es ergab daher Sinn die Klasse Zutaten einzufügen.

Für die restlichen Klassen entschieden wir, dass sie die Property $+R$ erhalten. Eine Pizza könnte zwar auch gegessen werden und eine Pizzeria kann auch auf indisches Essen umstellen, jedoch ist dies von unseren Competency Questions ausgehend keine potentielle Anfrage.

Identity

Wir haben uns nachträglich dazu entschieden, ausschließlich die Klasse **Pizzeria** mit $+I$ zu modellieren und haben dafür die Eigenschaft „Adresse“ hinzugefügt. In allen anderen Klassen lässt sich aus unserer Sicht nicht eindeutig sagen, ob zwei Instanzen gleich (oder unterschiedlich) sind. Diese enthalten dementsprechend $-I$.

Unity

Instanzen der Klasse **Pizza**, so genannte Pizzen, bilden ein zusammenhängendes Ganzes und werden daher mit $+U$ markiert. Diese Eigenschaft wird logischerweise an alle Subklassen wie **Pizza_Hawaii** oder **Pizza_bei_Mekan** vererbt. Gleiches gilt für die Klasse **Pizzeria**.

Für die meisten anderen Klassen kann aus unserer Sicht keine eindeutige Aussage über deren Unity getroffen werden. Wenn wir die Klasse **Zutat** betrachten, enthält diese sowohl Instanzen, die als ein Ganzes angesehen werden *können* (eine Tomate, ein Brokkolirösschen etc.), es gibt aber auch Zutaten, die eindeutig kein Unity-Kriterium erfüllen (Käseraspeln, Tomatensauce etc.). Dennoch haben wir uns dazu entschlossen, die Klasse **Zutat** sowie all ihre Subklassen als anti-unity ($\sim U$) zu modellieren und dementsprechend *alle* in unserer Ontologie potentiell vorkommenden Zutaten nicht als ein physisches Ganzes zu betrachten. Beispielsweise betrachten wir anstatt einer Tomate eine Tomatenscheibe als Zutat, welche auch nach Teilen dieser immer noch eine Tomatenscheibe ist.