Tutorium 9
Graphmodellierung, Cypher
Big Data Engineering

Prof. Dr. Jens Dittrich

bigdata.uni-saarland.de

11./12. Juli 2022

Verbesserung Übungsblätter - Häufige Fehler

Aufgabe 2:

- Vererbung nicht modelliert, also Mitarbeiter*innen nicht übertragen (was ein Problem ist, da Mitarbeiter*innen weder Köche/Köchinnen noch Servicekräfte sein können)

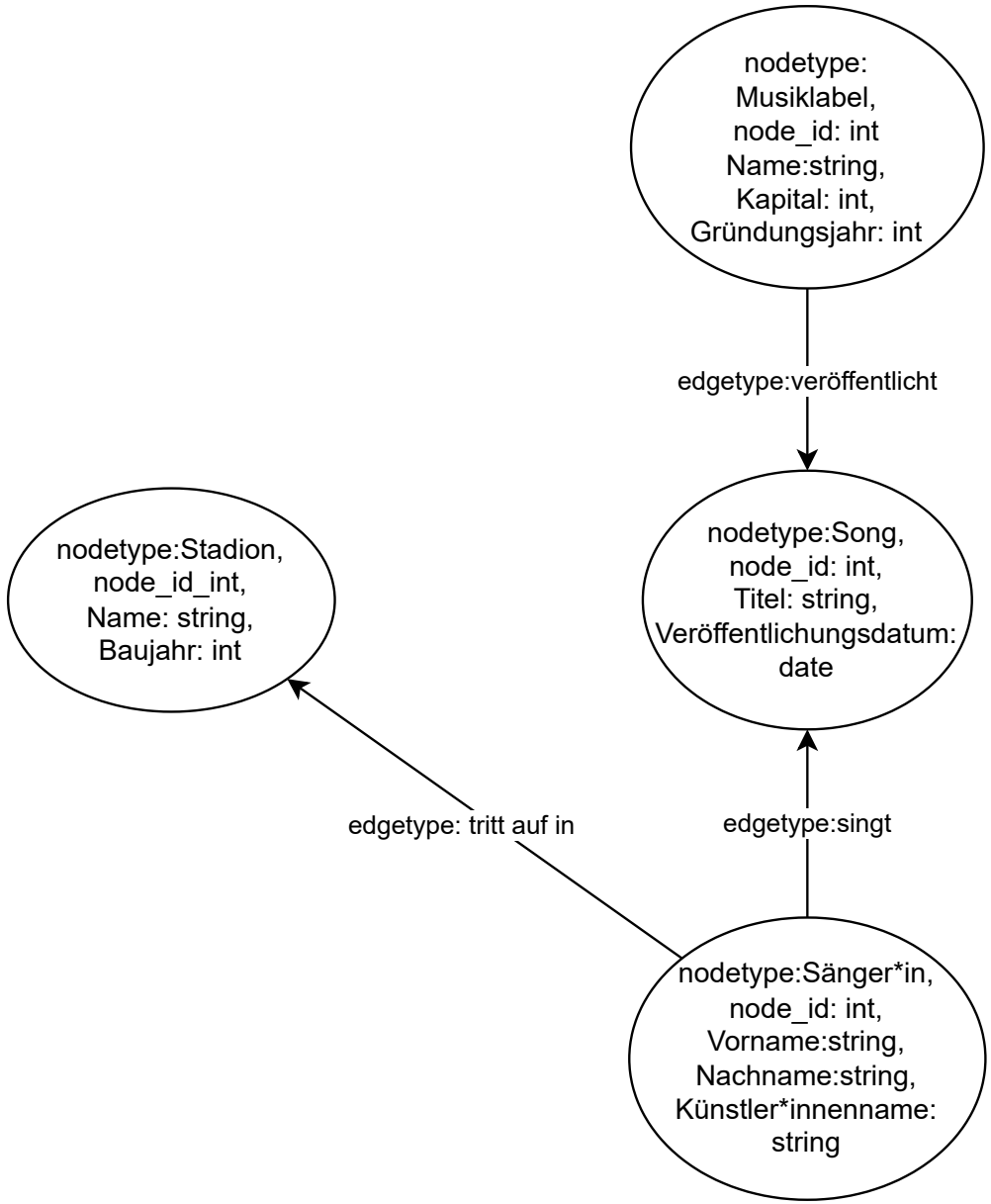
Aufgabe 3:

- Falsche Symbole für Vererbungen (z.B. Rauten statt 6-Ecke).

Aufgabe 4:

- `DISTINCT` im `COUNT` vergessen.

Cypher - Das heutige Modell als Graphschema



Wiederholung - Frage 1

Frage

Ist die ER-Modellierung für die Modellierung von Graphen geeignet?

Wiederholung - Frage 1

Frage

Ist die ER-Modellierung für die Modellierung von Graphen geeignet?

Lösung

Ja, die ER-Modellierung (bzw. auch das relationale Modell) ist perfekt geeignet, da wir hier Graphen auf ganz unterschiedliche Art und Weise modellieren können. Zudem können Kardinalitäten von Beziehungen dargestellt werden, was mit unserem Graphmodell nicht möglich ist. Dies gilt auch für Generalisierungen, die ebenso mit dem Graphmodell lediglich simuliert werden können.

Wiederholung - Frage 2

Frage

Was ist das Ergebnis der folgenden SQL Anfrage?

```
WITH RECURSIVE foo(n) AS (
      VALUES (1)
   UNION
      SELECT n*2 FROM foo
      WHERE n < 8
)
SELECT * FROM foo;
```

(A): {(2), (4), (6)}

(B): {(1), (2), (4), (8)}

(C): {(2), (4), (6), (8)}

(D): {(1), (2), (4)}

Wiederholung - Frage 2

Frage

Was ist das Ergebnis der folgenden SQL Anfrage?

```
WITH RECURSIVE foo(n) AS (
      VALUES (1)
   UNION
      SELECT n*2 FROM foo
      WHERE n < 8
)
SELECT * FROM foo;
```

Lösung

Die richtige Antwort lautet (B):

Die obige SQL Anfrage berechnet rekursiv die Exponentialfunktion mit Basis 2. Da zunächst die Bedingung im `WHERE` geprüft wird und erst danach `n*2` dem Ergebnis hinzugefügt wird, ist (8) auch noch Teil des Ergebnisses.

Wiederholung - Frage 3

Frage

Was ist das Ergebnis der folgenden SQL Anfrage?

```
WITH RECURSIVE foo(n,i) AS (
      VALUES (3,1)
  UNION
      SELECT n*n, i+1 FROM foo
      WHERE i < 3
)
SELECT n FROM foo;
```

(A): {(3, 1), (9, 2), (81, 3)}

(B): {(3), (9)}

(C): {(3), (9), (81)}

(D): {(3), (9), (27)}

Wiederholung - Frage 3

Frage

Was ist das Ergebnis der folgenden SQL Anfrage?

```
WITH RECURSIVE foo(n,i) AS (
      VALUES (3,1)
   UNION
      SELECT n*n, i+1 FROM foo
      WHERE i < 3
)
SELECT n FROM foo;
```

Lösung

Die richtige Antwort lautet (C):

Die obige SQL Anfrage quadriert den Wert n in jedem Schritt, und zählt dabei die Variable i hoch, bis diese den Wert 3 erreicht hat. Anschließend wird lediglich n ausgegeben.

Wiederholung - Frage 4

Frage

Ist SQL oder Cypher besser geeignet um Anfragen an Graphen zu stellen?

Wiederholung - Frage 4

Frage

Ist SQL oder Cypher besser geeignet um Anfragen an Graphen zu stellen?

Lösung

Cypher ist deutlich besser geeignet, da SQL besonders bei komplexen rekursiven Anfragen schnell unübersichtlich werden kann, was bei Cypher durch seine schlanke und elegante Syntax kein Problem darstellt.

Wiederholung - Frage 5

Frage

Was ist die Ausgabe der folgenden Cypher Anfrage?

```
MATCH   (st:Stadion) -- (sa: Sänger*in)
WHERE   st.Name = 'RockArena'
        AND sa.Vorname = 'Jim'
RETURN COUNT(DISTINCT sa) AS num;
```

Wiederholung - Frage 5

Frage

Was ist die Ausgabe der folgenden Cypher-Anfrage?

```
MATCH   (st:Stadion) -- (sa: Sänger*in)
WHERE   st.Name = 'RockArena'
        AND sa.Vorname = 'Jim'
RETURN COUNT(DISTINCT sa) AS num;
```

Lösung

Die Anzahl der verschiedenen Sänger*innen, die den Vornamen 'Jim' haben und im Stadion mit dem Namen 'RockArena' aufgetreten sind.

Wiederholung - Frage 6

Frage

Was ist die Ausgabe der folgenden Cypher Anfrage?

```
MATCH   w = (s:Sänger*in) - [*] - (m:Musiklabel)
WHERE   s.Vorname = 'Jane' AND s.Nachname = 'Doe'
        AND m.Gründungsjahr < 2000
RETURN COUNT(DISTINCT w);
```

(A): Die Anzahl der Pfade zwischen den Sänger*innen mit dem Namen 'Jane Doe' und Musiklabels, die vor 2000 gegründet wurden.

(B): Die Anzahl der unterschiedlichen Pfade zwischen den Sänger*innen mit dem Namen 'Jane Doe' und Musiklabels, die vor 2000 gegründet wurden.

(C): Die Anzahl der Kanten zwischen den Sänger*innen mit dem Namen 'Jane Doe' und Musiklabels, die vor 2000 gegründet wurden.

(D): Die Anzahl der verschiedenen Kanten zwischen den Sänger*innen mit dem Namen 'Jane Doe' und Musiklabels, die vor 2000 gegründet wurden.

Wiederholung - Frage 6

Frage

Was ist die Ausgabe der folgenden Cypher Anfrage?

```
MATCH   w = (s:Sänger*in) - [*] - (m:Musiklabel)
WHERE   s.Vorname = 'Jane' AND s.Nachname = 'Doe'
        AND m.Gründungsjahr < 2000
RETURN COUNT(DISTINCT w);
```

Lösung

Die richtige Antwort lautet (B):

Zunächst werden alle Pfade zwischen den Sänger*innen mit dem Namen 'Jane Doe' und den Musiklabels, die vor 2000 gegründet wurden berechnet. Diese werden anschließend gezählt, wobei durch das `DISTINCT` lediglich unterschiedliche Pfade betrachtet werden.

Aufgabe 1

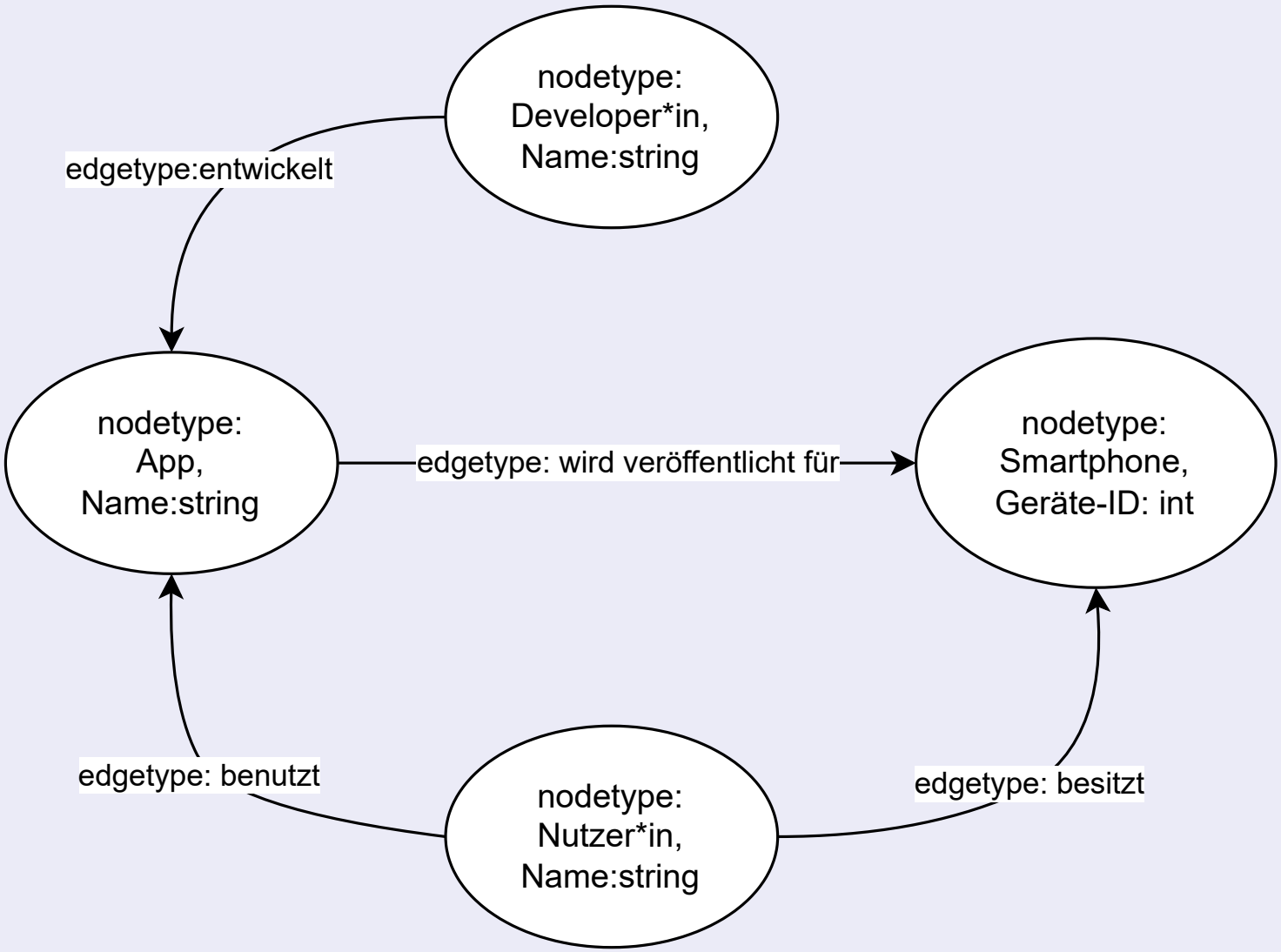
Frage

Erstellen Sie zu folgender Spezifikation das entsprechende Graphmodell:

- Jede*r Nutzer*in und Developer*in hat einen eindeutigen Namen.
- Smartphones haben eine eindeutige Geräte-ID, während Apps einen eindeutigen Namen haben.
- Nutzer*innen besitzen Smartphones und benutzten Apps.
- Apps werden für Smartphones veröffentlicht.
- Developer*innen entwickeln Apps.

Aufgabe 1

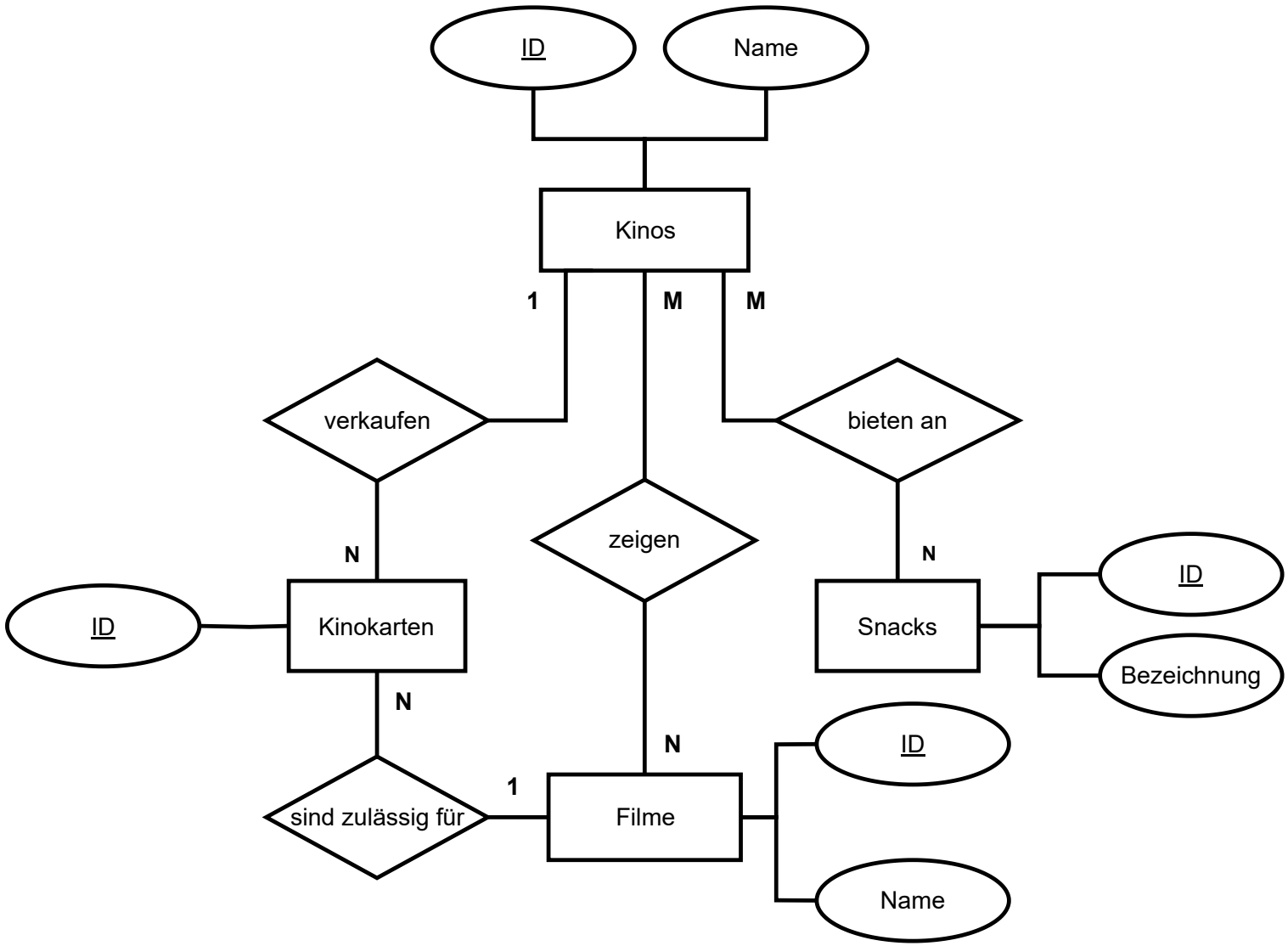
Lösung



Aufgabe 2

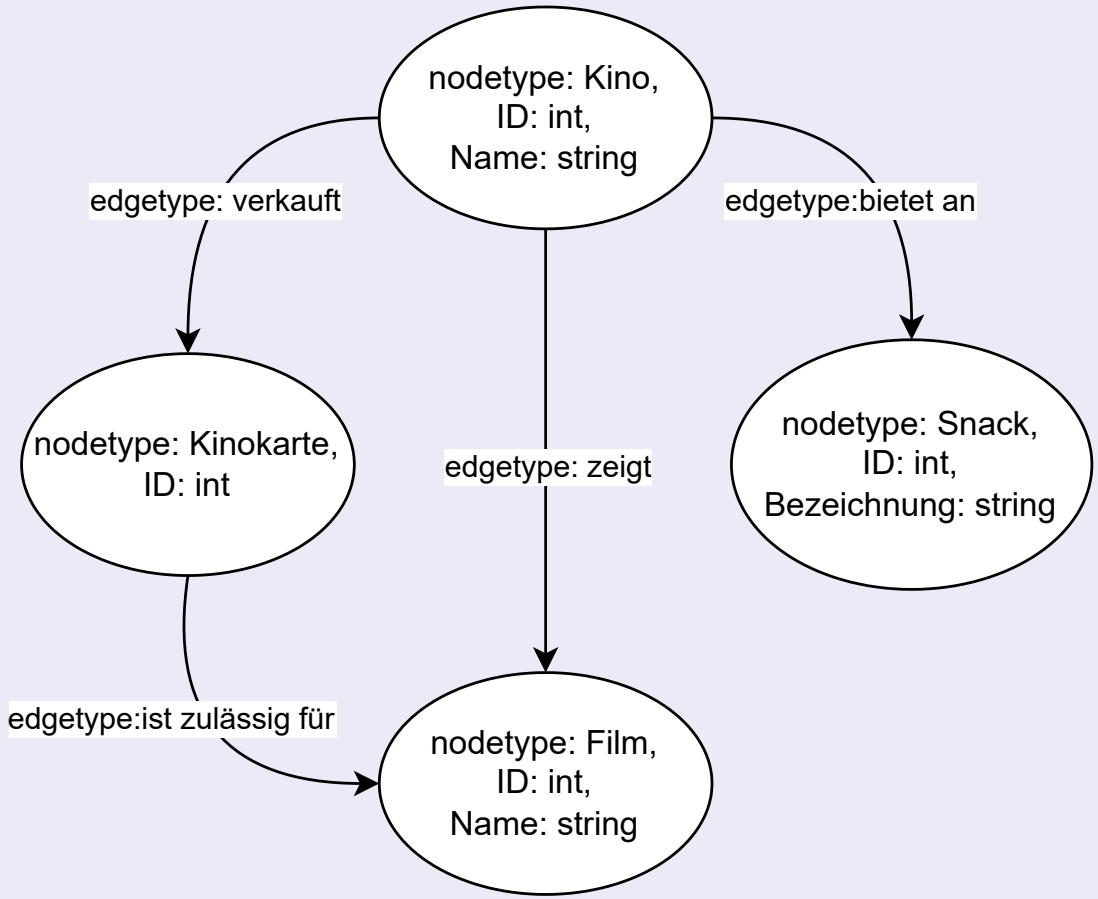
Frage

Überführen Sie folgendes ER-Modell in ein Graphmodell.



Aufgabe 2

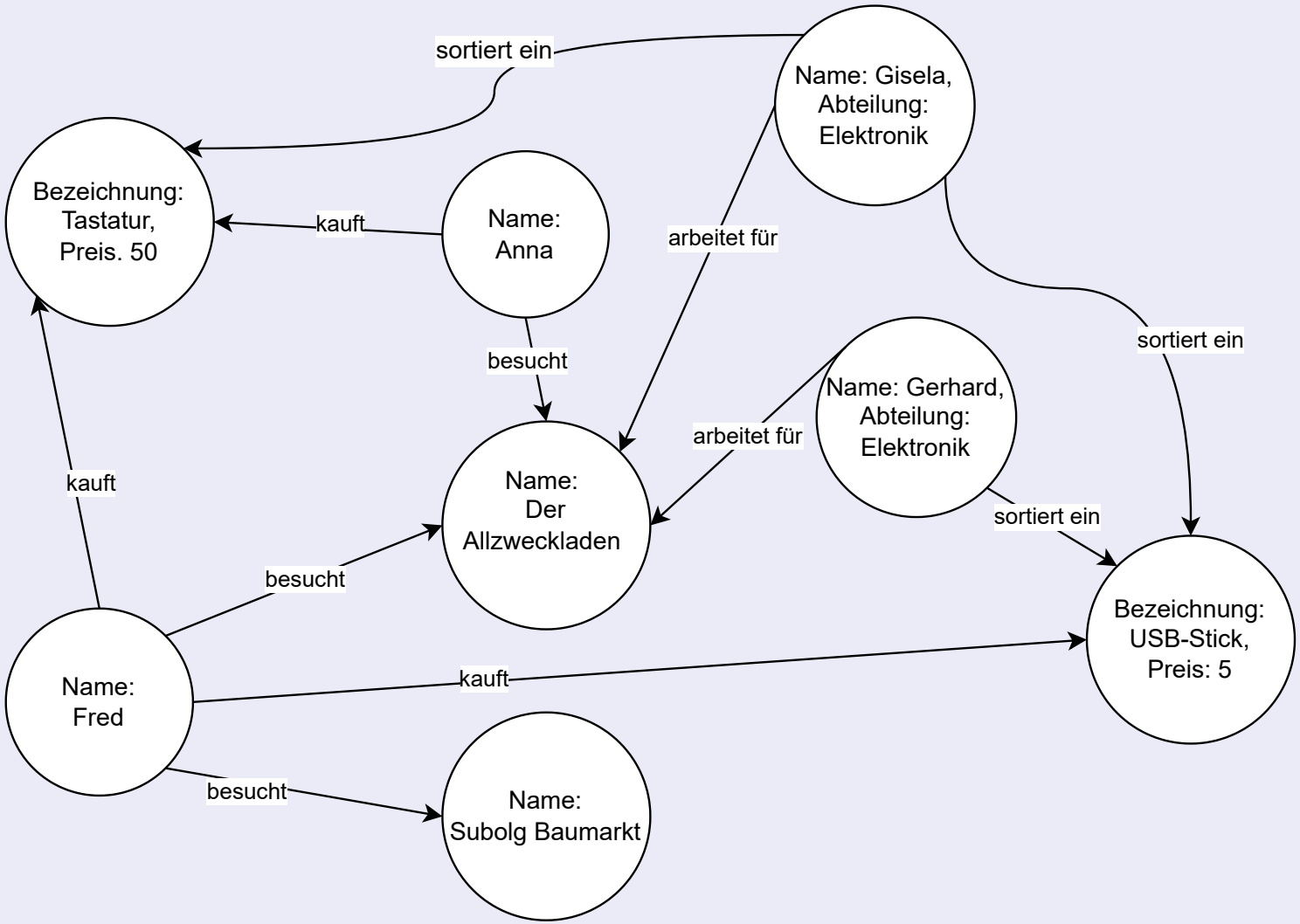
Lösung



Aufgabe 3

Frage

Betrachten Sie folgende Grapheninstanz:



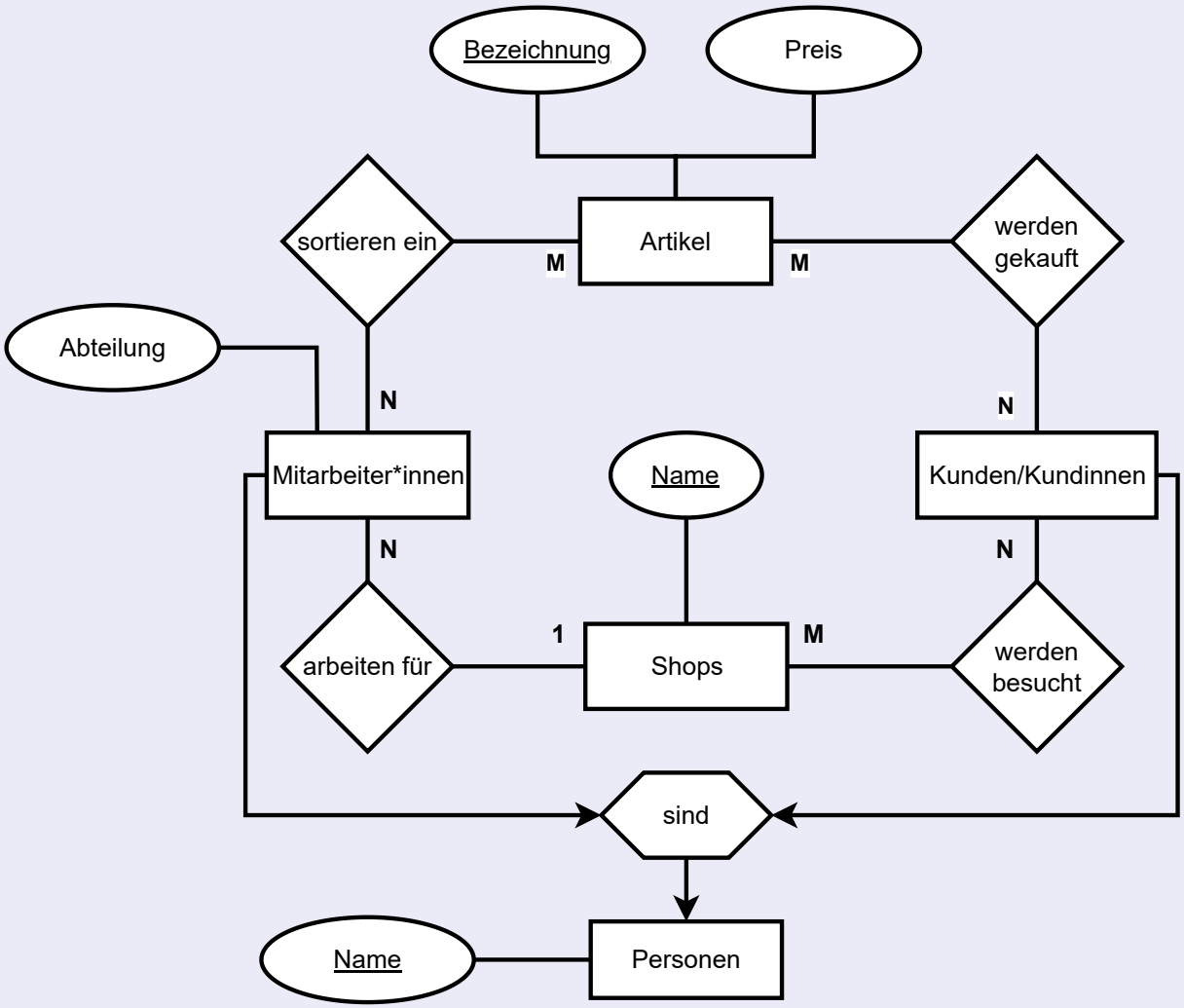
Aufgabe 3

Frage

Überführen Sie obige Grapheninstanz in ein ER-Modell. Gehen Sie davon aus, dass das jeweils erste Attribut eines Knotens eindeutig ist. Verwenden Sie zudem die Chen-Notation und setzten Sie die Funktionalitäten so, dass Sie exakt der Instanz entsprechen. Machen Sie zudem Gebrauch von Vererbungen, sofern dies möglich sind.

Aufgabe 3

Lösung



Aufgabe 4.1

Frage

Geben Sie eine Cypher Anfrage an, die das höchste Gründungsjahr der Musiklabels berechnet, die ein Kapital von mindestens 200.000 haben und vor 2000 einen Song veröffentlicht haben.

Aufgabe 4.1

Lösung

```
MATCH   (s:Song) -- (m:Musiklabel)
WHERE   s.Veröffentlichungsdatum < '2000-01-01'
        AND m.Kapital >= 200.000
RETURN  MAX(m.Gründungsjahr);
```

Aufgabe 4.2

Frage

Geben Sie eine Cypher Anfrage an, die den Künstler*innenname der Sänger*innen ausgibt, die in einem Stadion aufgetreten sind, das nach 1990 gebaut wurde und maximal zwei verschiedene Songs gesungen haben. Geben Sie die Ergebnisse in absteigender Reihenfolge der Nachnamen der Sänger*innen aus.

Aufgabe 4.2

Lösung

```
MATCH     (st:Stadion) -- (sa:Sänger*in)
          -- (so:Song)
WHERE     st.Baujahr > 1990
WITH      sa, COUNT(DISTINCT so) AS numSung
WHERE     numSung < 3
RETURN    sa.Künstler*innenname
ORDER BY  sa.Nachname DESC;
```

Aufgabe 4.3

Frage

Geben Sie eine Cypher Anfrage an, die die maximale Länge der ungerichteten Pfade zwischen einem der neuesten Songs und einem der ältesten Musiklabels ausgibt. Benennen Sie die Ausgabe in 'maxLength' um.

Aufgabe 4.3

Lösung

```
MATCH   (s:Song), (m:Musiklabel)
WITH    MAX(s.Veröffentlichungsdatum) AS newest,
        MIN(m.Gründungsjahr) AS oldest
MATCH   (s:Song), (m:Musiklabel)
WHERE   s.Veröffentlichungsdatum = newest
        AND m.Gründungsjahr = oldest
MATCH   w = s - [*] - m
RETURN  MAX(LENGTH(w)) AS maxLength
```

Appendix 1 - wichtige Cypher Schlüsselwörter

Schlüsselwörter

- `MATCH`: gibt an, nach welchen Patterns in der Datenbasis gesucht werden soll
- `WHERE`: Selektion; filtert Tupel nach einer Bedingung
- `WITH`: definiert eine lokale Sicht, führt implizit eine Gruppierung durch und macht daher das Nutzen von Aggregatfunktionen möglich
- `RETURN`: spezifiziert die Daten, die ausgegeben werden sollen; auch hier ist die Verwendung von Aggregatfunktionen möglich

Appendix 2 - wichtige Cypher Schlüsselwörter

Schlüsselwörter

- `AS`: Umbenennung einer Tabelle
- `ORDER BY`: sortiert die Ausgabe nach den gegebenen Attributen
- `ASC/DESC`: aufsteigend/absteigend
- `LIMIT`: limitiert die Anzahl der Tupel die ausgegeben werden, z.B. LIMIT 3 → höchstens 3 Tupel werden ausgegeben
- `DISTINCT`: verhindert Duplikate in der Ausgabe
- `NOT/AND/OR`: logische Operatoren für Bedingungen

Appendix 3 - ASCII-Anfragesyntax für Cypher

Schlüsselwörter

- `(a:A) --> (b:B)`: besucht alle gerichteten Kanten von A nach B
- `(a:A) -- (b:B)`: besucht alle Kanten (gerichtet und ungerichtet) von A nach B
- `(a:A) - [r:R] -> (b:B)`: besucht alle gerichteten Kanten vom Typ R von A nach B
- `(a:A) - [r:R] - (b:B)`: besucht alle Kanten (gerichtet und ungerichtet) vom Typ R von A nach B
- `(a:A) - [*3..7] -> (b:B)`: besucht alle gerichteten Pfade der Länge 3 bis 7 zwischen A und B
- `(a:A) - [*3..7] - (b:B)`: besucht alle Pfade (gerichtet und ungerichtet) der Länge 3 bis 7 zwischen A und B
- `(a:A) - [*] - (b:B)`: besucht alle Pfade (gerichtet und ungerichtet) zwischen A und B.