1.9 Üben und Vertiefen

Aufgabe 1:

a)
```
SELECT  Titel, Jahr
FROM    Buch
WHERE   Jahr > 1980;
```

Ergebnisdatensätze: 31

Titel	Jahr
Zu den Sternen	1982
Planlos im All	1985
Aufpasser	2009
Sudon	1996
Manieren des Lobs	1989
Der Planet mit Siegen	2000
Hornkönige	2001
Seelen am Ende des Tages	2010
Meine Hemden	2011
Ein verblassender Nagel	1989
Gegner ohne Inhalte	2000
Unser Riff	2011
Keine Liebe	1992

b)
```
SELECT  *
FROM    Buch
WHERE   Autor = 'B. D. Decker'
```

Ergebnisdatensätze: 4

Id	Titel	Autor	Jahr
1	Zu den Sternen	B. D. Decker	1982
3	Planlos im All	B. D. Decker	1985
6	Aufpasser	B. D. Decker	2009
7	Sudon	B. D. Decker	1996

c)
```
SELECT  Id
FROM    Kunde
WHERE   Name = 'von Brehm'
        AND Vorname = 'George'
```

Ergebnisdatensätze: 1

Id
6

Aufgabe 2:

a)
```
SELECT    Name, Größe
FROM      Basketball
ORDER BY  Größe DESC
```

Ergebnisdatensätze: 10

Name	Größe
Ehrlicher	182.3
Green	180
Daecher	178.9
Schnitzer	177.4

b)
```
SELECT    Name, Größe, Klasse, R
FROM      Basketball
WHERE     Klasse = '10a'
ORDER BY  R DESC
```

Ergebnisdatensätze: 2

Name	Größe	Klasse	R
Ehrlicher	182.3	10a	11
Kreß	170.8	10a	0

c)
```
SELECT  Name, Vorname, 1Pkt+2*2Pkt+3*3Pkt AS Punkte
FROM    Basketball
```

Ergebnisdatensätze: 10

Name	Vorname	Punkte
Ehrlicher	David	44
Daecher	Christian	39
Kreß	Torben	29
Green	Lothar	23
Haberland	Konstantin	15
Haid	Erich	34
Schnitzer	Ulrich	11
Gilmer	Noah	32
Hill	Hannes	41
Panhorst	Hans-Peter	18

d)
```
SELECT    Klasse, AVG(R)
FROM      Basketball
GROUP BY  Klasse
```

Ergebnisdatensätze: 4

Klasse	avg(R)
10a	5.5000
10b	4.2500
10c	4.0000
10d	4.3333

e)
```
SELECT  COUNT(*) AS Anzahl
FROM    Basketball
WHERE   1Pkt < 3Pkt
```

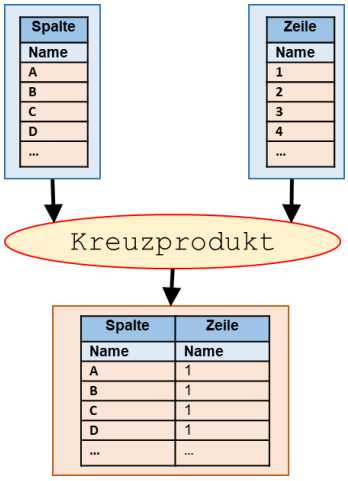
Ergebnisdatensätze: 1

Anzahl
5

Aufgabe 3:

a) Das Kreuzprodukt liefert ergibt alle möglichen Zelladressen der Form A1, A2, …

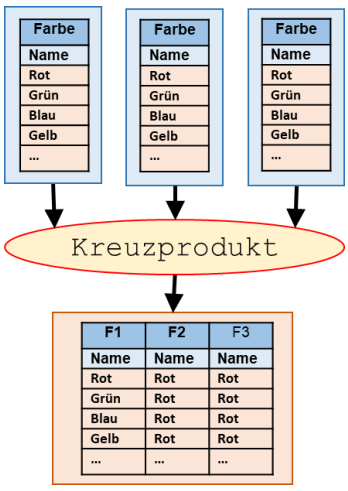
```
SELECT  Spalte.Name, Zeile.Name
FROM    Spalte, Zeile
```



Diskussion am Beispiel Excel:
Die Zeilennummern enden bei 1.048.576. In der Tabelle Zeile müssten also alle Zahlen von 1 bis 1.048.576 enthalten sein. Nach der Spalte Z folgt die Spalte AA, nach ZZ die Spalte AAA usw. Die letzte Spalte hat den Namen XFD. In der Tabelle Spalte müssen also alle Einträge von A bis XFD vorhanden sein.
Sind diese Bedingungen erfüllt, so werden alle vorhandenen Zelladressen erzeugt.

b) Das dreifache Kreuzprodukt aus der gleichen Tabelle ergibt alle Kombinationen der in der Tabelle gespeicherten Farben.

```
SELECT  F1.Name, F2.Name, F3.Name
FROM    Farbe AS F1, Farbe AS F2,
        Farbe AS F3
```



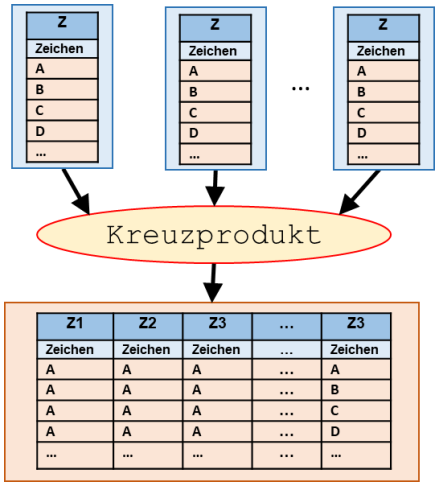
Diskussion:
Der Begriff Trikolore schließt auch senkrechte Streifen mit ein (z. B. Belgien ), allerdings solche mit ungleicher Streifenbreite (z. B. Kolumbien ) aus.
Gefordert werden zudem drei *verschiedene* Farben, so dass z. B. die österreichische  nicht als Trikolore bezeichnet wird.
Um diese Forderung zu erreichen, muss folgende Bedingung mit angegeben werden:

```
WHERE   F1.Name <> F2.Name AND F1.Name <> F3.Name
```

```
                                AND F2.Name <> F3.Name
```

c) Marius erhält alle Kombinationen aus acht Zeichen, die aus denen der Tabelle gebildet werden können.

```
SELECT  Z1.Zeichen, Z2.Zeichen,
        Z3.Zeichen, …,
        Z8.Zeichen
FROM    Z AS Z1, Z AS Z2, Z AS
        Z3, …, Z AS Z8
```



Diskussion:
Marius erhält zwar alle achtstelligen Kombinationen aus den vorgegebenen Zeichen, da aber die Tabelle nach den Zeichen sortiert ist, werden extrem viele von sehr „einfachen“ Passwörtern der Form „AAAAAAAA“ oder „AAAABAAB“ ausgegeben. Schnellere und bessere Vorschläge erhält er, wenn er das Problem mit einem passenden Algorithmus löst.
Schwierig ist es auch Randbedingungen zu erfüllen, z. B.:

- muss mindestens eine Ziffer und ein Sonderzeichen enthalten
- nicht mehr als zwei gleiche Zeichen

Aufgabe 4:

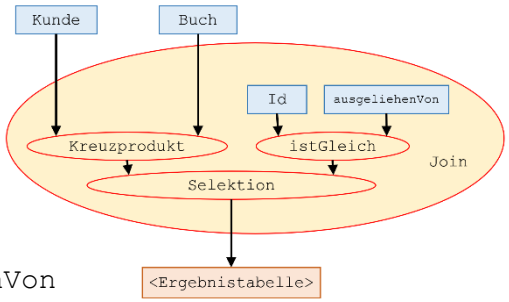
a) `Buch(Id: Zahl; Titel: Text; Autor: Text; Jahr: Zahl; ausgeliehenVon: Zahl)`

b) Wird der Fremdschlüssel in die Tabelle `Kunde` eingefügt, könnte jeder Kunde nur ein Buch ausleihen bzw. man müsste mehrere Fremdschlüsselattribute einführen.

c) 1. Möglichkeit:
Der entsprechende Attributwert bleibt leer. Diese Datensätze können durch
`WHERE ausgeliehenVon IS NULL`
gefunden werden.

2. Möglichkeit:
Es wird ein „Dummykunde“ z. B. mit `ID = 0` eingeführt.
Dann lautet die Bedingung:
`WHERE ausgeliehenVon = 0.`

d) `Join(Kunde; Buch) = Selektion(Kreuzprodukt(Kunde; Buch); Id = ausgeliehenVon)`




```
SELECT DISTINCT Kunde.Vorname,
                Kunde.Name
FROM   Kunde, Buch
WHERE  Kunde.Id = Buch.ausgeliehenVon
```

Aufgabe 5:

a)

```
SELECT Mond.Name, Planet.Name
FROM   Mond, Planet
WHERE  Mond.Planet = Planet.Name
       AND Mond.entdeckt = Planet.entdeckt
```

Ergebnisdatensätze: 1

Name	Name
Triton	Neptun

b)

```
SELECT Mond.Name, Planet.Name
FROM   Mond, Planet
WHERE  Mond.Planet = Planet.Name
       AND Mond.Halbachse >
       Planet.Durchmesser*100
```

Ergebnisdatensätze: 25

Name	Name
Albiliorix	Saturn
Ananke	Jupiter
Caliban	Uranus
Carme	Jupiter
Erriapus	Saturn
Ferdinand	Uranus
Halimede	Neptun
Laomediia	Neptun
Margaret	Uranus
Nereid	Neptun
Neso	Neptun

c)

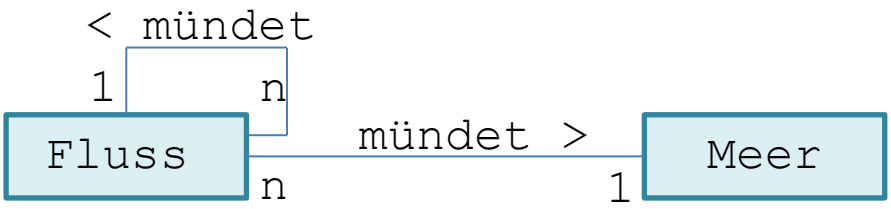
```
SELECT Mond.Name, Planet.Name
FROM   Mond, Planet
WHERE  Mond.Planet = Planet.Name
       AND (Mond.Durchmesser/2)/
       (Mond.Halbachse-Planet.Durchmesser/2) > 0.0046
```

Ergebnisdatensätze: 4

Name	Name
Charon	Pluto
Dysnomia	Eris
Io	Jupiter
Oberon	Uranus

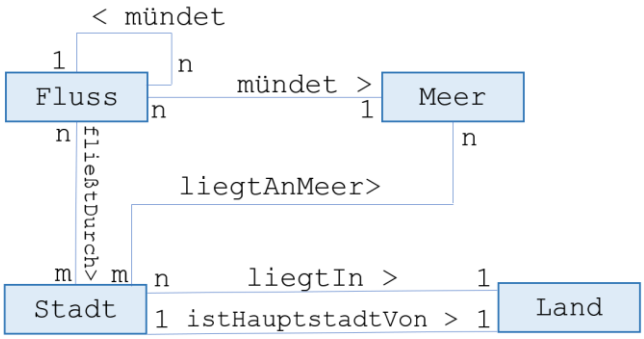
Aufgabe 6:

a)

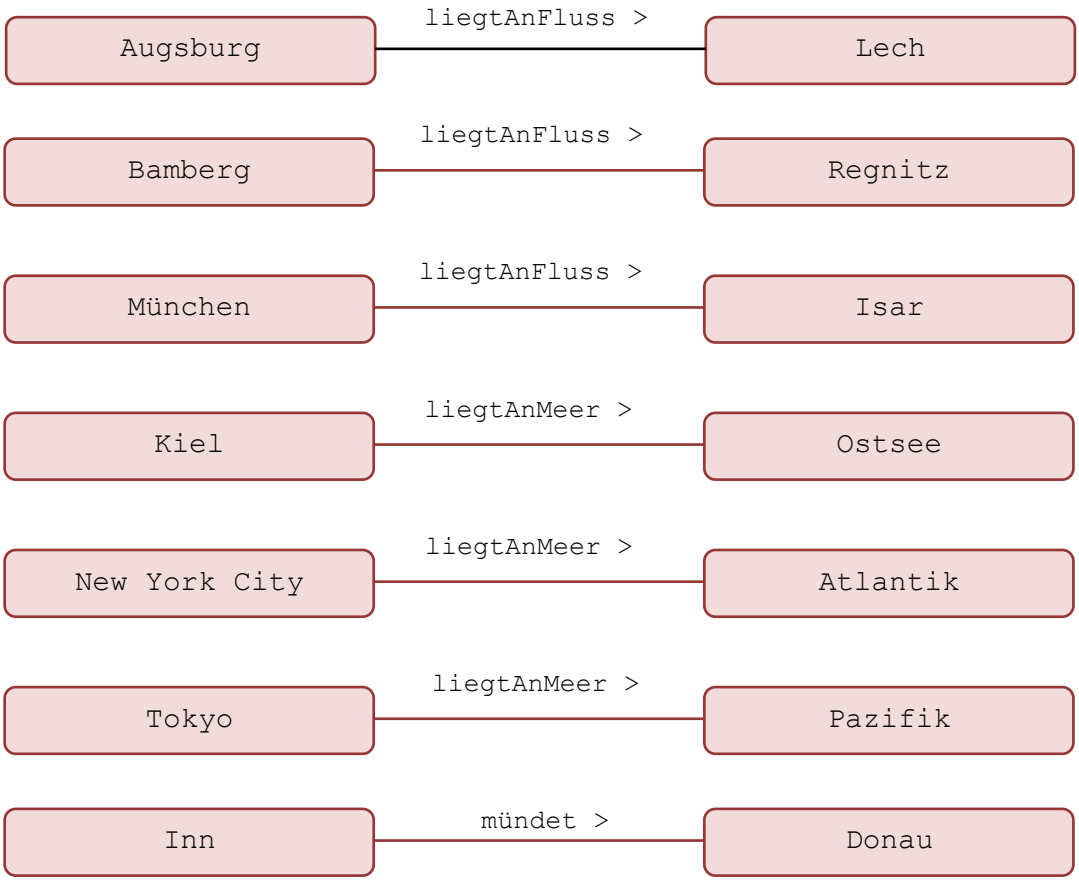


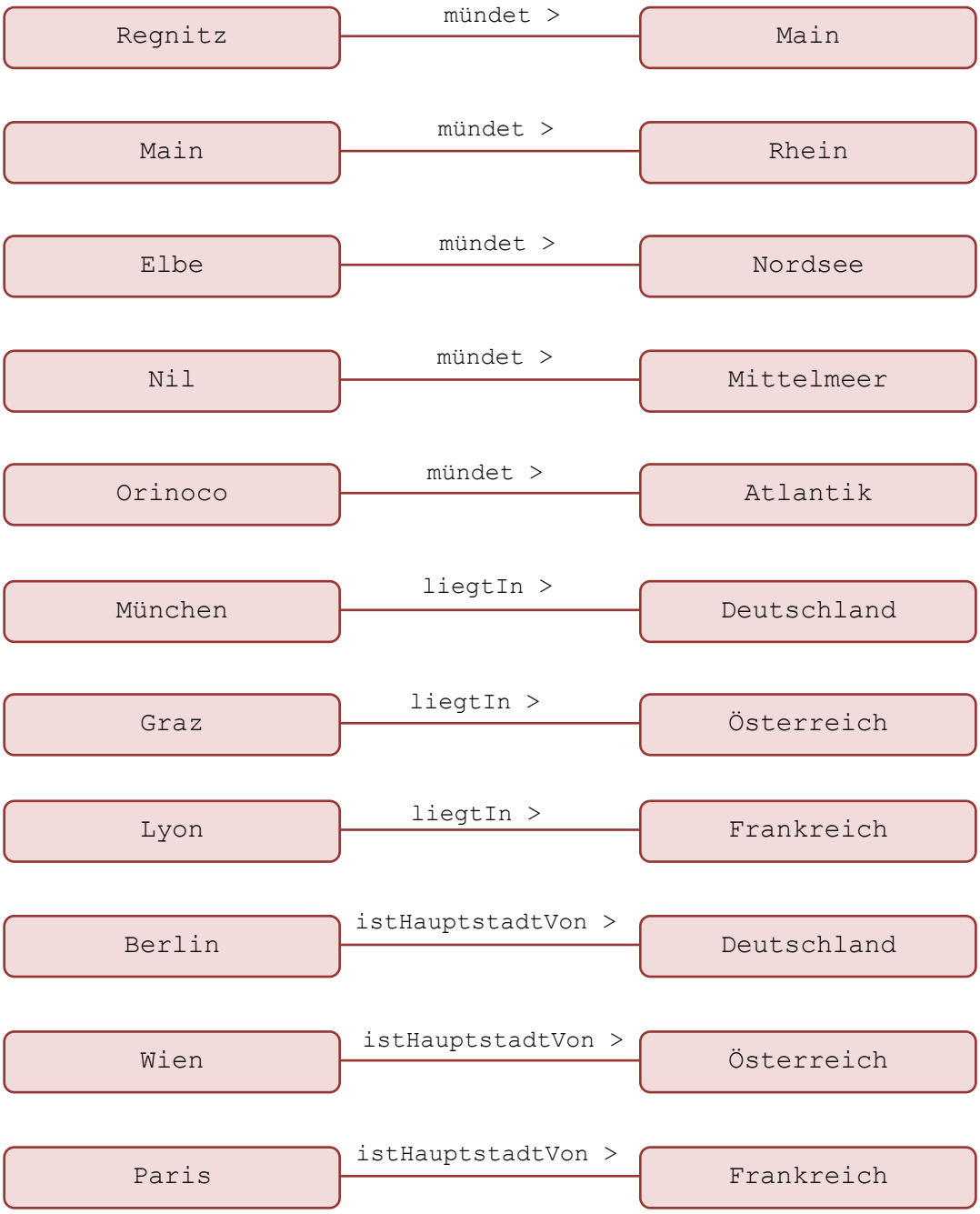
Ein Fluss hat als Zufluss beliebig viele Flüsse.
Ein Meer hat als Zufluss beliebig viele Flüsse.

b)



c) individuelle Lösungen, z. B.:





d) Siehe b)

1:1-Beziehung: istHauptstadtVon

1:n-Beziehungen: mündet, liegtIn

n:m-Beziehungen: fließtDurch, liegtAnMeer

e) Bei der Beziehung istHauptstadtVon kann 0..1 gesetzt werden, da die meisten Städte keine Hauptstädte sind, z. B. Nürnberg.

Aufgabe 7:

a) 1:n

b) n:m

c) 1:1

d) 1:n

Aufgabe 8:

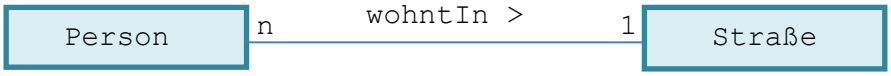
a)

Anton: Person | Beate: Person | Carmen: Person

Daniel: Person

Badestraße: Straße | Hauptstraße: Straße

Veilchenstraße: Straße

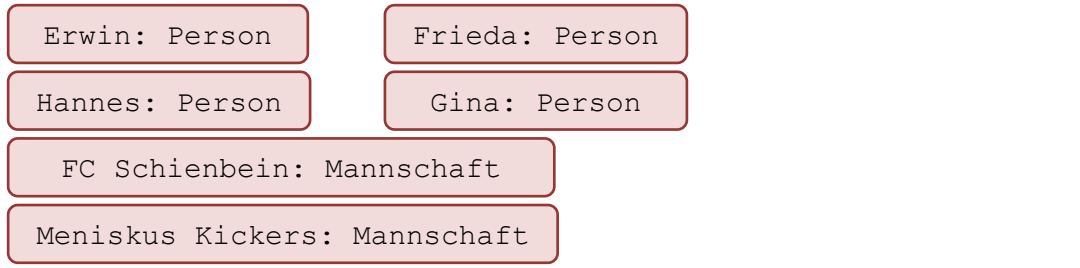


Person(Name: Text; wohntIn: Zahl)
Straße(SId: Zahl; Name: Text)

Person	
Name	wohntIn
Anton	1
Beate	2
Carmen	1
Daniel	3
…	

Straße	
SId	Name
1	Badestraße
2	Hauptstraße
3	Veilchenstraße
4	Baumweg
…	

b)

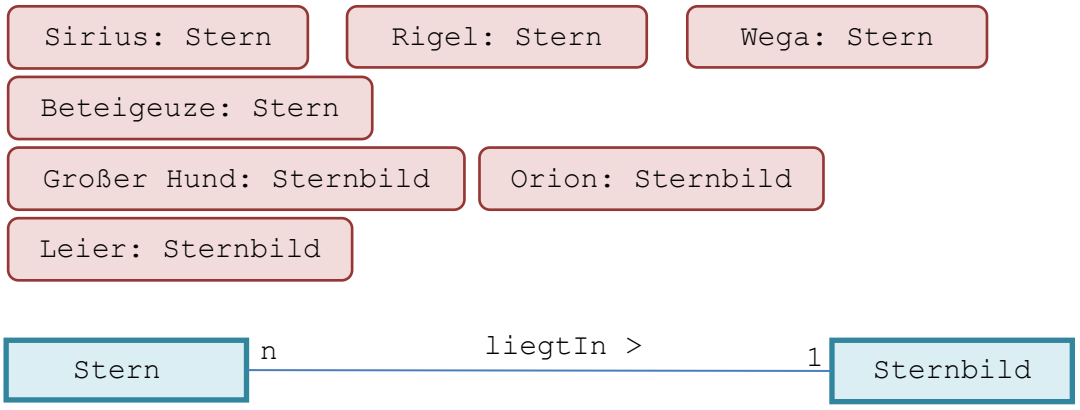


Person(Name: Text; istMannschaftsführerIn: Zahl)
Mannschaft(MId: Zahl; Name: Text)

Person	
Name	istMannschaftsführerIn
Erwin	1
Frieda	
Gina	2
Hannes	
…	

Mannschaft	
MId	Name
1	FC Schienbein
2	Meniskus Kickers
3	SC Wadenbein
…	

c)



Stern(Name: Text; liegtIn: Zahl)
Sternbild(SId: Zahl; Name: Text)

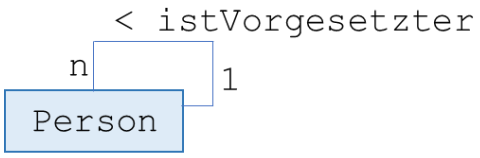
Stern	
Name	liegtIn
Sirius	1
Rigel	2
Wega	3
Orion	2
…	

Sternbild	
SId	Name
1	Großer Hund
2	Orion
3	Leier
…	

Aufgabe 9:

a) `Vorgesetzter` bezieht sich auf die `Id` einer Person.

b)



c) Ein Attribut `Untergebener` könnte nur eine untergeordnete Person speichern. Normalerweise hat aber eine Person mehr als einen Untergebenen, so dass mehrere Attribute `Untergebener1`, `Untergebener2` usw. eingefügt werden müssten.

d)

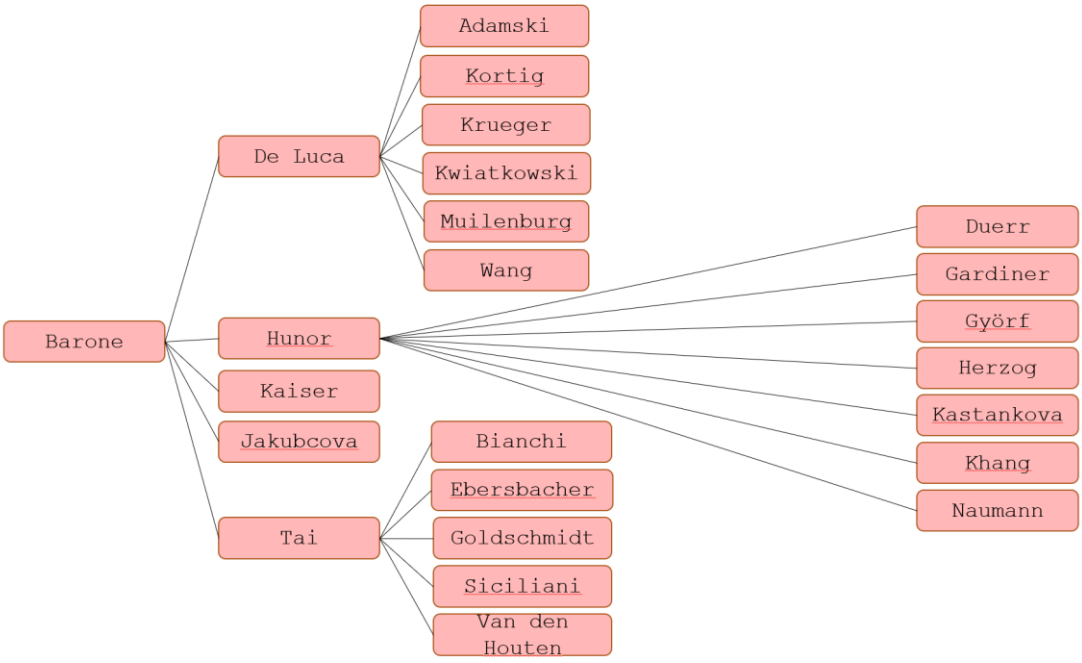
```
SELECT  Vorname, Name
FROM    Person
WHERE   Vorgesetzter = 1009
```

Ergebnisdatensätze: 5

Vorname	Name
Sos	Hunor
Jasmin	Kaiser
Ivana	Jakubcova
Bi	Tai
Leo	De Luca

e) Mit folgernder Abfrage kann die Struktur erkannt werden:

```
SELECT    P1.Name AS Vorgesetzter, P2.Name
FROM      Person P1, Person P2
WHERE     P2.Vorgesetzter = P1.Id
ORDER BY  P1.Name
```

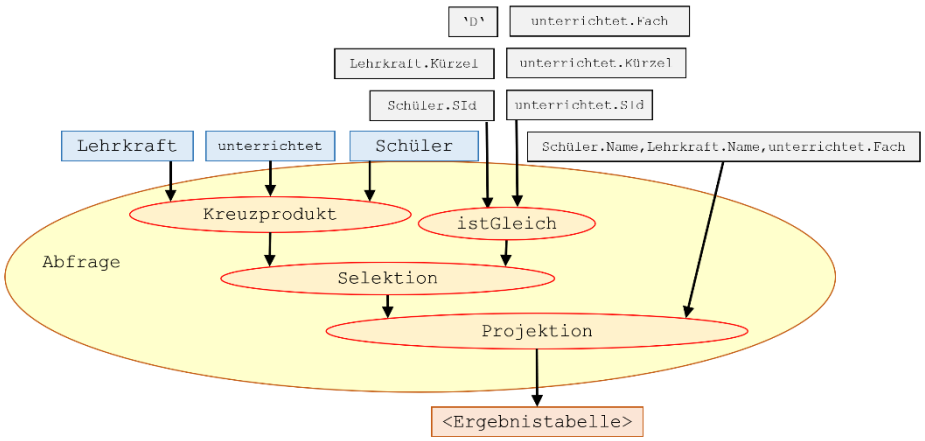


f) Dargestellt ist eine hierarchische Struktur in Form eines Baums mit der Wurzel Barone. Dieser hat keinen Vorgesetzten und ist somit der „Chef“ des Unternehmens.

Aufgabe 10:

a) Die SQL-Abfrage listet zu jedem Schüler auf, von welcher Lehrkraft er im Fach Deutsch unterrichtet wird. Da es sich um eine n:m-Beziehung handelt, ist zur Verknüpfung der beiden Tabellen `Schüler` und `Lehrkraft` die Beziehungstabelle `unterrichtet` nötig.
Hinter `FROM` werden alle drei Tabellen genannt, so dass das entsprechende Kreuzprodukt erstellt wird. Die Bedingung hinter `WHERE` erzeugt einen Join, der nur die richtigen Beziehungen selektiert. Dann erfolgt eine Projektion auf die Namen des Schülers und der Lehrkraft sowie dem Fach.

b)
```
Projektion(
    Selektion(
        Kreuzprodukt(Schüler; unterrichtet; Lehrkraft);
        UND(Schüler.SId = unterrichtet.SId;
            unterrichtet.Kürzel = Lehrkraft.Kürzel
            unterrichtet.Fach = 'D'));
    (Schüler.Name, Lehrer.Name, unterrichtet.Fach) )
```

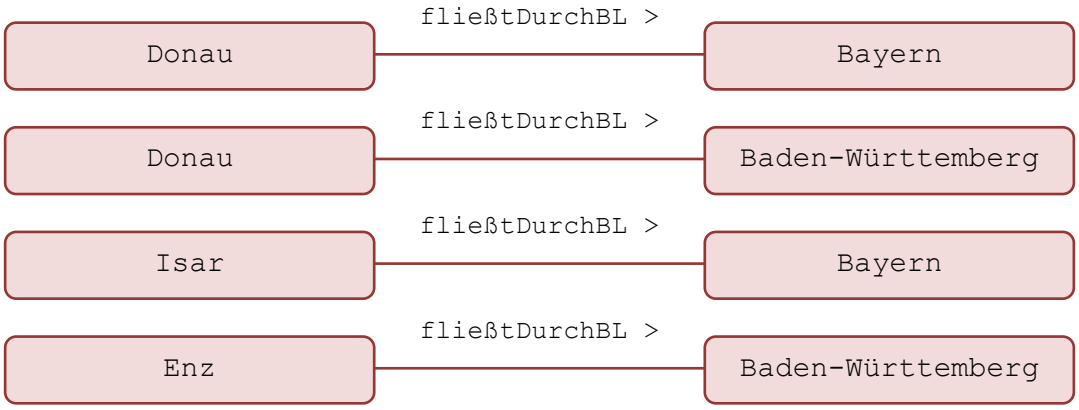


Hinweis:
Die Vergleiche der Selektion wurden verkürzt dargestellt.

c)
```
SELECT  Schüler.Name
FROM    Schüler, unterrichtet
WHERE   Schüler.SId = unterrichtet.SId
        AND Fach = 'Inf'
```

Aufgabe 11:

a)



b)
```
SELECT Fluss.*, Bundesland.*
FROM   Fluss, fließtDurchBL, Bundesland
WHERE  Fluss.Name = fließtDurchBL.Flussname
       AND Bundesland.Name = fließtDurchBL.Bundeslandname
```

c)
```
SELECT Fluss.Name, Stadt.Name, Bundesland.Name
FROM   Fluss, fließtDurchBL, Bundesland , Stadt, fließtDurch
WHERE  Fluss.Name = fließtDurchBL.Flussname
       AND Bundesland.Name = fließtDurchBL.Bundeslandname
       AND Stadt.Name = fließtDurch.Stadtname
       AND Fluss.Name = fließtDurch.Flussname
       AND Stadt.Einwohner >= 1000
```

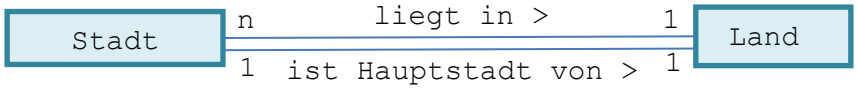
Hinweis:
Da die Einwohnerzahl in der Einheit 1000 Einwohner angegeben ist, muss als Bedingung `Stadt.Einwohner >= 1000` formuliert werden.

Aufgabe 12:

a) Die Aussage ist korrekt, da bei n:m-Beziehungen zusätzliche Tabellen nötig sind. Beispiel:

drei Tabellen: `Lehrkraft`, `Schüler`, `unterrichtet`

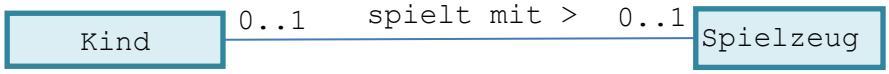
b) Die Aussage ist korrekt, da bei mehreren 1:n-(oder 1:1-)Beziehungen in einer Klasse mit einer anderen mehrere Fremdschlüssel in dieser Tabelle nötig sind. Beispiel:



Fremdschlüssel in Stadt: `liegtIn`, `istHauptstadtVon`

c) Die Aussage ist korrekt, da in einem Tabellenschema zusätzlich die Datentypen angegeben werden.

d) Die Aussage ist korrekt. Beispiel:



Ein Kind spielt mit keinem oder einem Spielzeug.
Ein Spielzeug wird benutzt von einem oder keinem Kind.

e) Die Aussage ist in dem Sinne korrekt, da sonst nachträglich der Aufbau einer Datenbank geändert und evtl. viele Daten neu eingetragen werden müssen.

Beispiel:
Wird bei einem Buch nur ein Attribut für den Autor berücksichtigt, müssen dort bei Bücher, die mehrere Autoren haben, alle eingetragen werden. Die Suche nach allen Büchern, an denen ein bestimmter Autor (mit)geschrieben hat, ist dann schwierig. Um dies zu berücksichtigen, müssen die Autoren entweder auf mehrere Attribute aufgeteilt oder besser noch eine neue Klasse Autor erstellt werden. Zwischen `Buch` und `Autor` besteht dann eine n:m-Beziehung, so dass noch eine Beziehungstabelle nötig ist. Alle Autoren müssen dann noch in die Klasse Autor ein getragen und die Beziehungstabelle mit Daten gefüllt werden. Zudem müssen alle bisher erstellten Abfragen, die auch Autoren betreffen, neu formuliert werden.

Aufgabe 13:

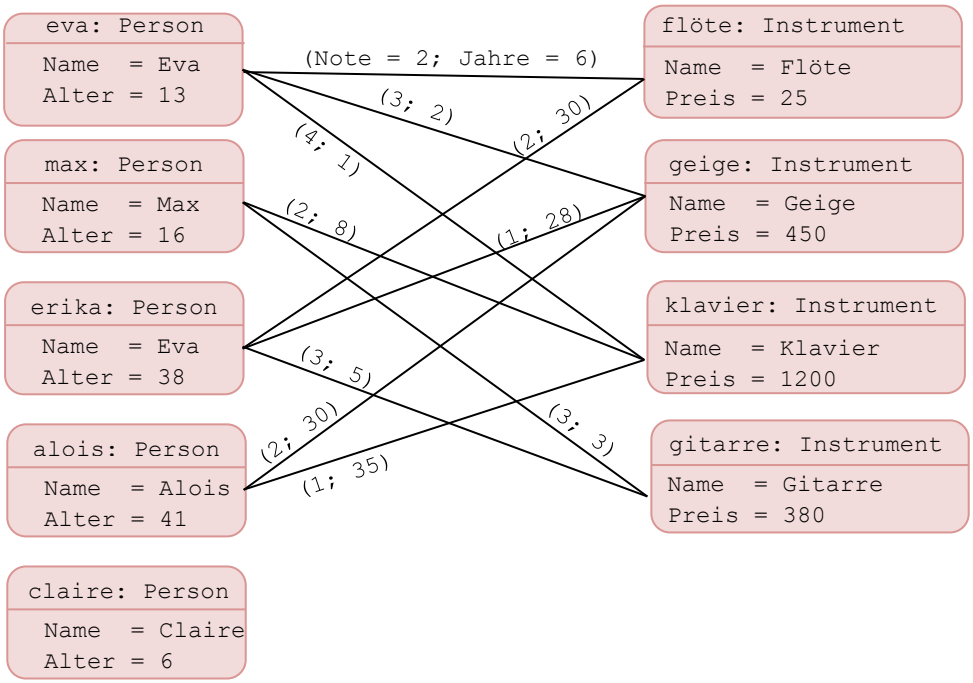
a) Klassen Objekte

`Instrument` `Flöte`, `Geige`, `Klavier`, `Gitarre`

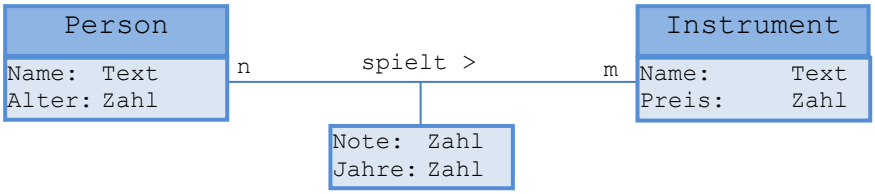
`Person` `Eva`, `Max`, `Erika`, `Alois`, `Claire`

Da zwischen `Instrument` und `Person` eine n:m-Beziehung besteht, ist bei der Implementierung eine weitere Klasse `spielt` nötig.

b)



c)



d) Schemata des relationalen Modells:

```
Person(Name: Text; Alter: Zahl)
Instrument(Name: Text; Preis: Zahl)
spielt(Person: Text; Instrument: Text; Note: Zahl;
       Jahre: Zahl)
```

e) Siehe Lösungsdatei `1-9_Aufgabe13_Loesung.odb`

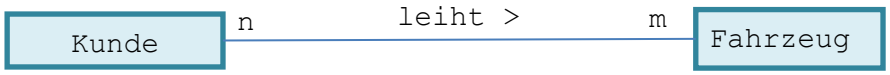
f)

```
SELECT    Person.Name, Person.Alter
FROM      Person, spielt
WHERE     Person.Name = spielt.Person
          AND spielt.Instrument = 'Klavier'

SELECT    Person.Name, spielt.Instrument
FROM      Person, spielt
WHERE     Person.Name = spielt.Person
          AND spielt.Jahre > 5

SELECT    Person.Name, spielt.Instrument, Person.Alter
FROM      Person, spielt, Instrument
WHERE     Person.Name = spielt.Person
          AND spielt.Instrument = Instrument.Name
          AND Instrument.Preis > 400
```

Aufgabe 14:

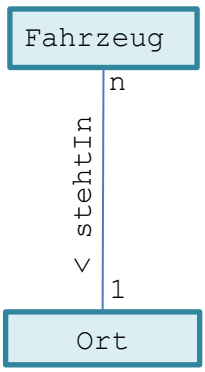
a) Die Beziehung zwischen `Kunde` und `Fahrzeug` muss umgekehrt beschriftet werden, d.h. es muss `leiht >` stehen. Es sollte eine n:m-Beziehung sein, welche zusätzlich das Datum des Verleihs speichert, da eine Kunde bei einer Mietautofirma zu verschiedenen Zeitpunkten beliebig viele Autos leihen kann und jedes Mietauto von beliebig vielen verschiedenen Personen geliehen werden kann.

Insgesamt:

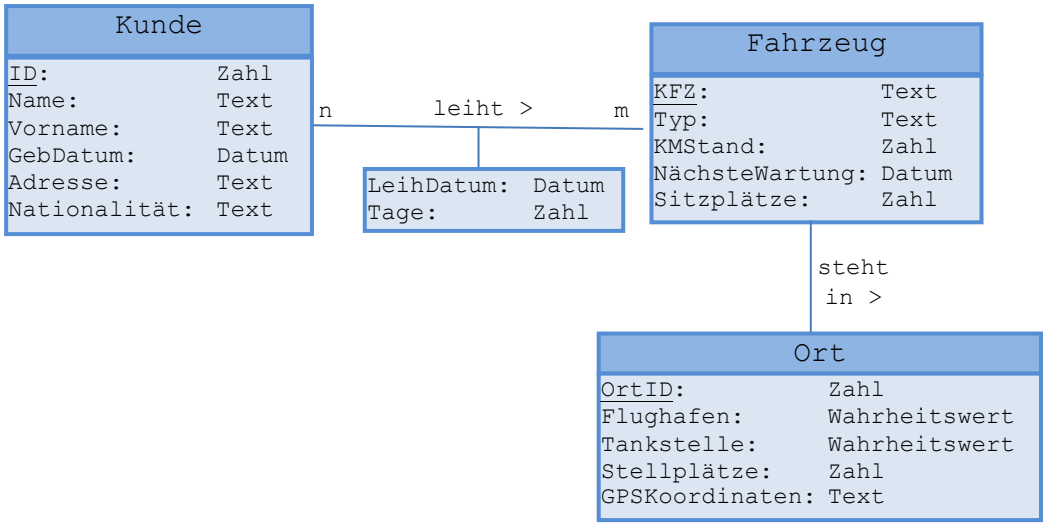


Die Beziehung `Ausleihort` zwischen `Kunde` und `Ort` ist unnötig, da durch die Beziehung `stehtIn >` schon der Ausleihort gegeben ist.
Die Beziehung `stehtIn` zwischen `Fahrzeug` und `Ort` muss eine n:1-Beziehng sein, da ein Fahrzeug nur einen Standort haben kann, an einem Standort aber beliebig viele Fahrzeuge stehen können. Zudem würde die angegebene n:n-Beziehung fordern, dass auf beiden Seiten immer genau gleich viele Objekte beteiligt sein müssen, was selten sinnvoll ist.

Insgesamt:



b) individuelle Lösungen, z. B.:



c)
```
Kunde(ID: Zahl; Name: Text; Vorname: Text;
      GebDatum: Datum; Adresse: Text;
      Nationalität: Text)
Fahrzeug(KFZ: Text; Typ: Text; KMStand: Zahl;
         NächsteWartung: Datum; Sitzplätze: Zahl;
         stehtIn: Zahl)
Ort(OrtID: Zahl; Flughafen: Wahrheitswert;
    Tankstelle: Wahrheitswert; Stellplätze: Zahl;
    GPSKoordinaten: Text)
leiht(ID: Zahl; KFZ: Text; LeihDatum: Datum; Tage: Zahl)
```

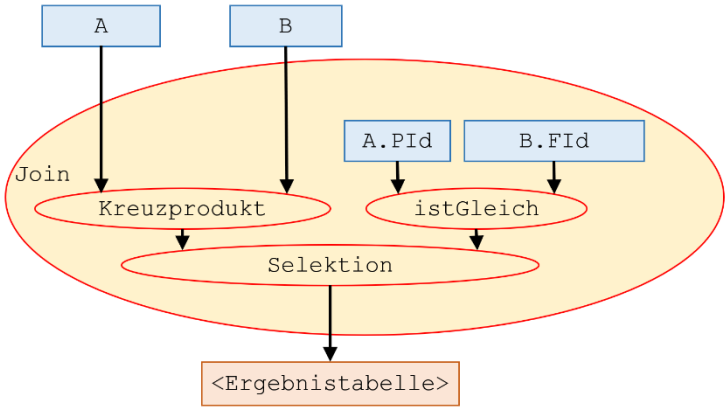
d) Damit der Abgabeort bei jedem Leihauftrag gespeichert werden kann, muss die Beziehung `leiht` erweitert werden und ein zusätzlicher Fremdschlüssel mit der `OrtID` gespeichert werden

```
leiht(ID: Zahl; KFZ: Text; LeihDatum: Datum; Tage: Zahl;
      AbgabeOrt: Zahl)
```

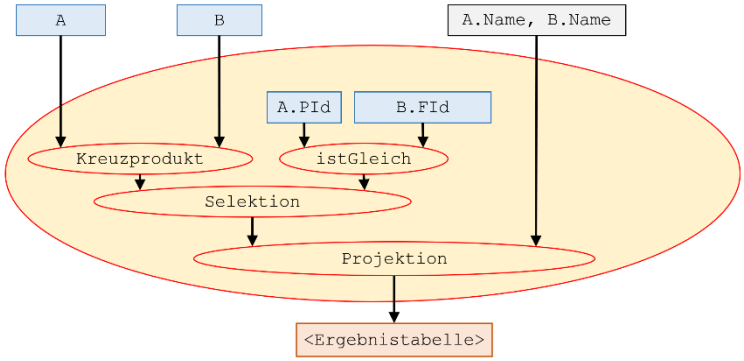
Aufgabe 15:

a) Beide Abfragen beschreiben dieselbe Abfrage, wenn sich `FId` auf `PId` bezieht. Dies sollte auch die einzige Beziehung zwischen beiden Tabellen sein, da sonst `Join(A, B)` nicht eindeutig ist.

b)



c)




```
Projektion(Selektion(Kreuzprodukt(A; B);
                     istGleich(A.PId; B.FId));
           (A.Name, B.Name))
```

Aufgabe 16:

a) `SId` bezieht sich auf den Primärschlüssel des benoteten Schülers.
`Kürzel` bezieht sich auf den Primärschlüssel entsprechenden Lehrers.
`Fach` bezieht sich auf den Primärschlüssel des benoteten Fachs.

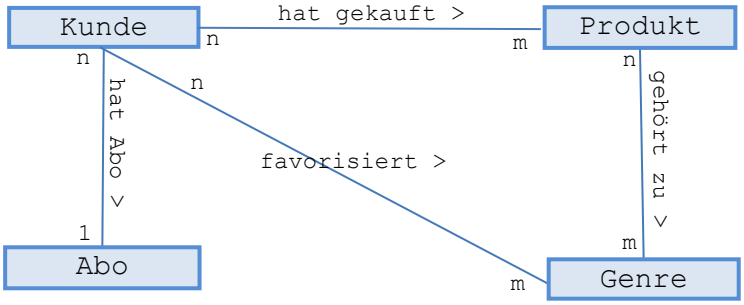
b) `Kürzel` muss aufgenommen werden, da ein Schüler von mehr als einer Lehrkraft eine Note erhält.
`Fach` muss aufgenommen werden, da ein Schüler von einer Lehrkraft in zwei verschiedenen Fächern eine Note erhalten kann, z.B. in D und G.
`Datum` muss ebenfalls Teil des Primärschlüssels sein, da eine Lehrkraft in einem Fach einem Schüler mehrere Noten geben kann. Normalerweise geschieht dies an unterschiedlichen Tagen. Der (allerdings kaum vorstellbare) Fall, dass ein Schüler von derselben Lehrkraft im selben Fach mehr als eine Note bekommt, ist dann immer noch nicht erfasst.

c) Die Attribute `Note`, `Datum` und `Art` sind mit genau einer Benotung eines Schülers verknüpft.
Sollten sie in einer anderen Tabelle gespeichert werden, würden z.B. die Noten in D mit mehreren Schulaufgaben und vielen anderen Leistungsnachweisen eine Vielzahl von neuen Spalten in z.B. der Tabelle `Schüler` verlangen. Diese würde dann sehr unübersichtlich und begrenzt die Anzahl der speicherbaren Leistungsnachweise unnötigerweise.

Aufgabe 17:

a) Klassen: `Kunde`, `Abo`, `Produkt`, `Genre`
Beziehungen: `hatGekauft`, `favorisiert`, `gehörtZu`
Dies erkennt man an den Tabellennamen und an den eingetragenen Fremdschlüsseln.

b)



c)
```
Kunde(KId: Zahl; Vorname: Text; Name: Text; AboNr: Zahl)
Abo(AId: Zahl; Typ: Text)
Produkt (PId: Zahl; Name: Text; Typ: Text)
Genre (GId: Zahl; Name: Text)

hatGekauft(KId: Zahl; PId: Zahl)
favorisiert(KId: Zahl; GId: Zahl)
gehörtZu(PId: Zahl; GId: Zahl)
```

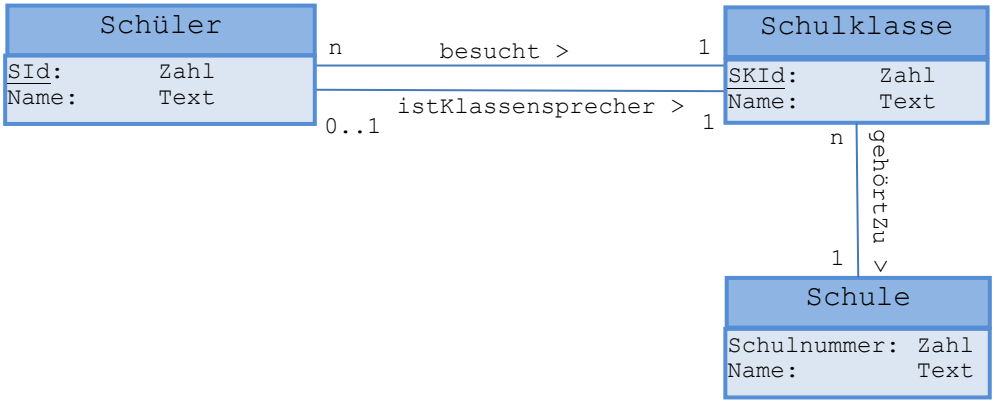
d)
```
SELECT  *
FROM    Kunde, Produkt, hatGekauft
WHERE   Kunde.KId = hatGekauft.PId
        AND hatGekauft.KId = Produkt.PId

SELECT  *
FROM    Kunde, Genre, favorisiert
WHERE   Kunde.KId = favorisiert.KId
        AND favorisiert.GId = Genre.GId

SELECT  *
FROM    Produkt, Genre, gehörtZu
WHERE   Produkt.PId = gehörtZu.PId
        AND gehörtZu.GId = Genre.GId
```

e) Die Tabelle `hatGekauft` würde dann vier Attribute besitzen, z. B.
`Kundenvorname, Kundenname, Name, Typ`
SQL-Abfragen, die diese Beziehung verwenden, würden viel komplizierter werden, z. B.:

```
SELECT  *
FROM    Kunde, Produkt, hatGekauft
WHERE   Kunde.Name = hatGekauft.Kundenname
        AND Kunde.Vorname   = hatGekauft.Kundenvorname
        AND hatGekauft.Name = Produkt.Name
        AND hatGekauft.Typ  = Produkt.Typ
```

Aufgabe 18:

a)



b) Da es sich um 1:1- und 1:n-Beziehungen handelt, können diese durch passende Fremdschlüssel in den entsprechenden Schemata umgesetzt werden:

```
Schüler(SId: Zahl; Name: Text; Klasse: Text)
Schulklasse(SKId: Zahl; Name: Text;
            Klassensprecher: Zahl; Schulnummer: Zahl)
Schule(Schulnummer: Zahl; Name: Text)
```

Hinweis:
Da es sich bei `istKlassensprecher` um eine 0..1:1-Beziehung handelt, ist es günstig, im Gegensatz zum Datenmodell den Fremdschlüssel `Klassensprecher` in der Klasse `Schulklasse` einzufügen. Da pro Klasse nur ein Schüler (erster) Klassensprecher ist, wären bis auf diesen bei allen anderen Schülern die Attributwerte leer. Viele leere Attributwerte sind oft ein Hinweis auf ein schlecht konstruiertes Model. Falsch ist es jedoch nicht.

c) Hinweis:
Die Datenbank enthält nicht alle Schüler einer Schule, sondern nur einen Teil. Die Modellierung aus a) wurde entsprechend der Datenbank gemacht, die Attributnamen können sich von der Modellierung der Schüler unterscheiden und müssen entsprechend angepasst werden.

```
SELECT  Schüler.Name, Schulklasse.SKId
FROM    Schüler, Schulklasse
WHERE   Schüler.SId = Schulklasse.Klassensprecher
```

```
SELECT  Schüler.Name, Schulklasse.SKId
FROM    Schüler, Schulklasse
WHERE   Schüler.Klasse = Schulklasse.SKId
        AND Schulklasse.Name = "10c"
        AND Schulklasse.Schulnummer = 0153
```

```
SELECT  Schüler.Name, Schule.Name
FROM    Schüler, Schulklasse, Schule
WHERE   Schüler.SId = Schulklasse.Klassensprecher
        AND Schulklasse. Schulnummer = Schule.Schulnummer
```

Aufgabe 19:

a) Da sich der Ranglistenplatz ändern kann (meist zu Beginn der Saison), ist die Zuordnung zu einer Spielerin langfristig nicht fest. Es müssten dann alle bereits formulierten SQL-Abfragen geändert werden, in denen er vorkommt.

b)
```
SELECT  S1.Vorname AS Spielerin1,
        S2.Vorname AS Spielerin2
FROM    Spielerin AS S1, Spielerin AS S2
```

c)
```
SELECT  S1.Vorname AS Spielerin1,
        S2.Vorname AS Spielerin2
FROM    Spielerin AS S1, Spielerin AS S2
WHERE   S1.Nr <> S2.Nr
```

```
Projektion(Selektion(Kreuzprodukt(Spielerin S1;
                                  Spielerin S2);
                     S1.Nr ≠ S2.Nr);
           (S1.Vorname, S2.Vorname))
```

d)
```
SELECT  S1.Vorname AS Spielerin1,
        S2.Vorname AS Spielerin2
FROM    Spielerin AS S1, Spielerin AS S2
WHERE   S1.Nr < S2.Nr
```

e) Da jede Spielerin gegen jede andere Spielerin einmal spielen muss, entsteht eine n:m-Beziehung der Tabelle mit sich selbst.

f) spieltGegen(Spielerin1: Zahl; Spielerin2: Zahl;
Punkt1: Zahl; Punkt2: Zahl)

g) siehe Datenbank Kap 1.9 Aufgabe 19.odb, Tabelle spieltGegen

h)
```
SELECT  S1.Nachname, S1.Vorname,
        spieltGegen.Punkt1, spieltGegen.Punkt2,
        S2.Nachname, S2.Vorname
FROM    Spielerin AS S1, Spielerin AS S2, spieltGegen
WHERE   S1.Nr = spieltGegen.Spielerin1
        AND S2.Nr = spieltGegen.Spielerin2

SELECT  S1.Vorname,
        spieltGegen.Punkt1, spieltGegen.Punkt2,
        S2.Vorname
FROM    Spielerin AS S1, Spielerin AS S2, spieltGegen
WHERE   S1.Nr = spieltGegen.Spielerin1
        AND S2.Nr = spieltGegen.Spielerin2
        AND (S1.Vorname = 'Lena' OR S2.Vorname = 'Lena')

SELECT  S1.Vorname, S1.R AS Rang1,
        spieltGegen.Punkt1 AS 1, spieltGegen.Punkt2 AS 2,
        S2.Vorname, S2.R AS Rang2
FROM    Spielerin S1, Spielerin S2, spieltGegen
WHERE   S1.Nr = spieltGegen.Spielerin1
        AND S2.Nr = spieltGegen.Spielerin2
        AND (spieltGegen.Punkt1 = 0 OR spieltGegen.Punkt2 = 0)
```