Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

Frühjahr 2018

46116

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

— Prüfungsaufgaben —

Fach: **Informatik (Unterrichtsfach)**

Einzelprüfung: **Softwaretechnologie/Datenbanksysteme**

Anzahl der gestellten Themen (Aufgaben): **2**

Anzahl der Druckseiten dieser Vorlage: **14**

Bitte wenden!

Thema Nr. 1

Teilaufgabe 1

Aufgabe 1 (Anwendungsfälle und Sequenzdiagramme)

Wir betrachten ein Netzwerk von Geldautomaten (engl. ATM = Automated Teller Machine) und Banken. Alle Geldautomaten und Banken sind mit einer Zentrale verbunden, über die die Kommunikation zwischen ATMs und Banken geregelt wird. Abb. 1 zeigt die Struktur des Netzwerks als Objektdiagramm für den Fall von drei ATMs und zwei Banken.

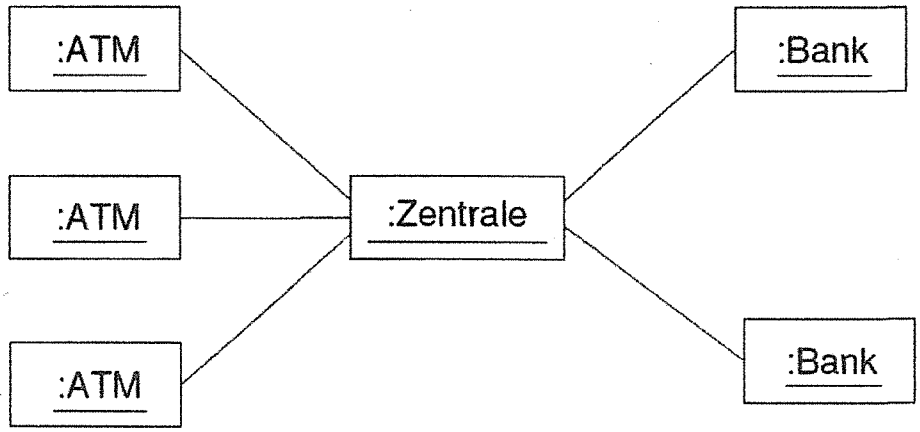


Abbildung 1: Netzwerk von Geldautomaten und Banken mit Zentrale

Wir betrachten den Anwendungsfall "Geld abheben am ATM". Für diesen Anwendungsfall gibt es einen Hauptablauf, der wie unten angegeben textuell beschrieben ist. Die Substantive Kreditkarte und Karte werden hier und im Folgenden synonym verwendet.

Hauptablauf:

1. Das ATM fordert den Kunden auf, eine Kreditkarte einzugeben.
2. Der Kunde gibt seine Kreditkarte ein.
3. Das ATM liest die Kreditkarte und erfragt daraufhin die Geheimzahl.
4. Der Kunde gibt die Geheimzahl ein.
5. Das ATM überprüft die Geheimzahl und lässt dann die Karte bei der Zentrale überprüfen.
6. Die Zentrale überprüft die BIC (Bank Identifier Code) und lässt dann die Karte von der Bank überprüfen.
7. Die Bank überprüft, ob die Karte gesperrt ist und benachrichtigt dann die Zentrale, dass alles in Ordnung ist.
8. Daraufhin gibt auch die Zentrale dem ATM ihr Okay.
9. Das ATM fragt den Kunden nach dem abzuhebenden Betrag.

Fortsetzung nächste Seite!

10. Der Kunde gibt den gewünschten Betrag ein.
11. Das ATM überprüft, ob der Betrag nicht höher als 1000 Euro ist und fordert dann die Zentrale auf, die Abhebung zu veranlassen.
12. Die Zentrale leitet die Anforderung an die Bank weiter, die daraufhin das Konto auffordert, den Betrag abzuheben.
13. Das Konto überprüft den Abhebungsbetrag (der nicht den Kreditrahmen des Kunden überschreiten darf) und führt dann die Abhebung durch.
14. Das Konto benachrichtigt die Bank, dass die Abhebung erfolgreich war.
15. Das Bank gibt diese Nachricht an die Zentrale weiter, die dem ATM die erfolgreiche Abhebung mitteilt.
16. Das ATM gibt den gewünschten Betrag in Bargeld aus.
17. Der Kunde entnimmt das Geld, woraufhin das ATM die Karte ausgibt.
18. Der Kunde entnimmt die Karte.

a) Geben Sie ein Sequenzdiagramm an, das die im obigen Hauptablauf beschriebenen Interaktionen zwischen dem Kunden (als Aktor), dem ATM, der Zentrale, der Bank und dem Konto zeigt. Für die Kreditkarte braucht kein Objekt im System eingeführt zu werden. Beachten Sie, dass bestimmte Aktionen auch nur ein Objekt betreffen können und dann als Selbstaufrufe darzustellen sind, wie z.B. die Aktion `Karte lesen` in Abb. 2.

b) Das Sequenzdiagramm in Abb. 2 spezifiziert einen Ausnahmeablauf, der durchgeführt wird, wenn die Kreditkarte nicht lesbar ist.

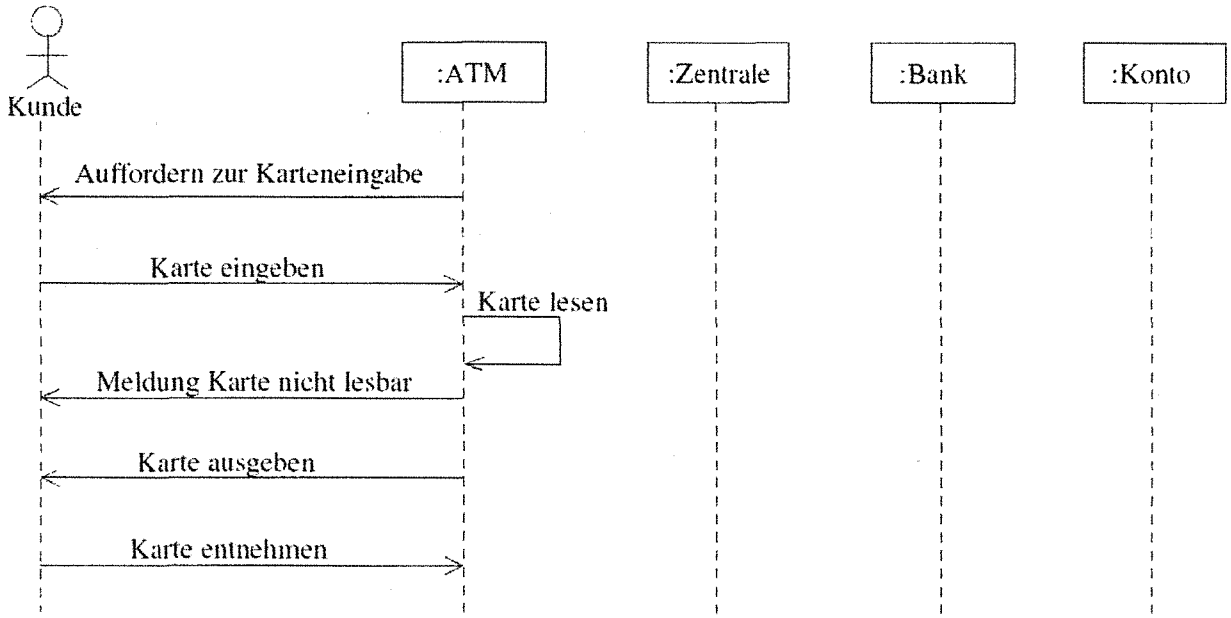


Abbildung 2: Ausnahmeablauf, falls Karte nicht lesbar

Fortsetzung nächste Seite!

Zeichnen Sie ein Sequenzdiagramm mit einer Alternative, so dass sowohl der Hauptablauf als auch der oben spezifizierte Ausnahmeablauf mögliche Abläufe des Sequenzdiagramms sind. Sie müssen dazu nicht alle Interaktionen des Hauptablaufs nochmal einzeichnen, sondern es genügt, wenn deutlich erkennbar ist, wo welche Interaktionen des Hauptablaufs vorkommen sollen.

c) Schildern Sie kurz in Worten vier weitere Situationen, in denen es **nicht** zu einem erfolgreichen Abschluss der Abhebung kommt. Geben Sie an, in welchem Schritt des Hauptablaufs die Ausnahmesituation auftreten kann. (Alternativabläufe brauchen hierzu nicht angegeben zu werden.)

Aufgabe 2 (Objektdiagramme, Klassendiagramme, Vererbung)

Gegeben sei das Objektdiagramm in Abb. 3.

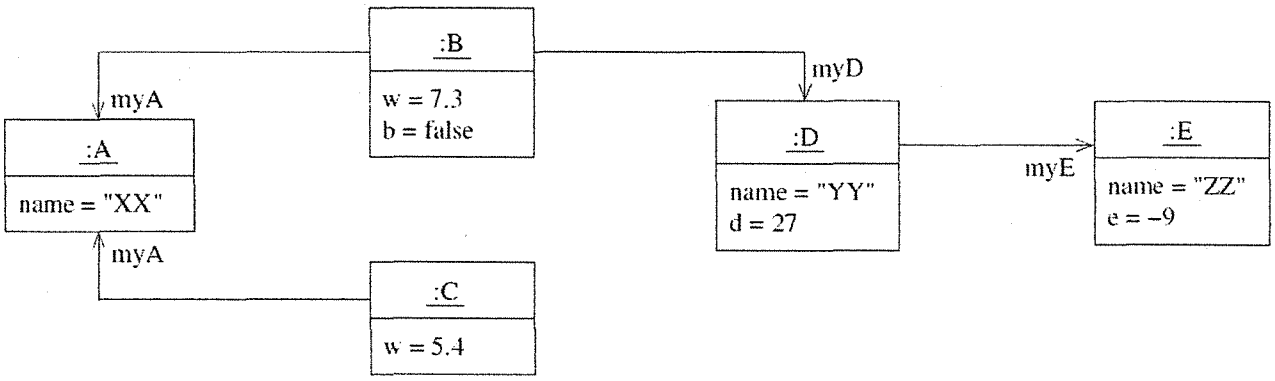


Abbildung 3: Objektdiagramm

Konstruieren Sie ein Klassendiagramm, das genau fünf Klassen `A`, `B`, `C`, `D` und `E` enthält und folgende Bedingungen erfüllt:

1. Das Objektdiagramm in Abb. 3 ist eine korrekte Instanziierung des Klassendiagramms. Insbesondere sollen Links im Objektdiagramm Instanzen von gerichteten Assoziationen im Klassendiagramm sein. An den Enden der gerichteten Assoziationen sind die aus dem Objektdiagramm ersichtlichen Rollennamen anzugeben. Multiplizitäten können weggelassen werden. Die Attribute sind mit zu den Werten im Objektdiagramm passenden Datentypen zu versehen.

2. Es gibt keine zwei Klassen, die ein Attribut mit demselben Namen enthalten und es gibt auch keine zwei Klassen, die eine ausgehende gerichtete Assoziation mit demselben Rollennamen am gerichteten Assoziationsende haben.

Hinweis: Um Bedingung 2 zu erfüllen, sind geeignete Vererbungsbeziehungen im Klassendiagramm zu verwenden.

Fortsetzung nächste Seite!

Aufgabe 3 (Objektorientierte Implementierung)

Für die nächste Fußballweltmeisterschaft möchte ein Wettbüro ein Programm zur Verwaltung von Spielern, Vereinen und (National-)Mannschaften entwickeln. Dazu wurde bereits das folgende UML-Klassendiagramm entworfen.

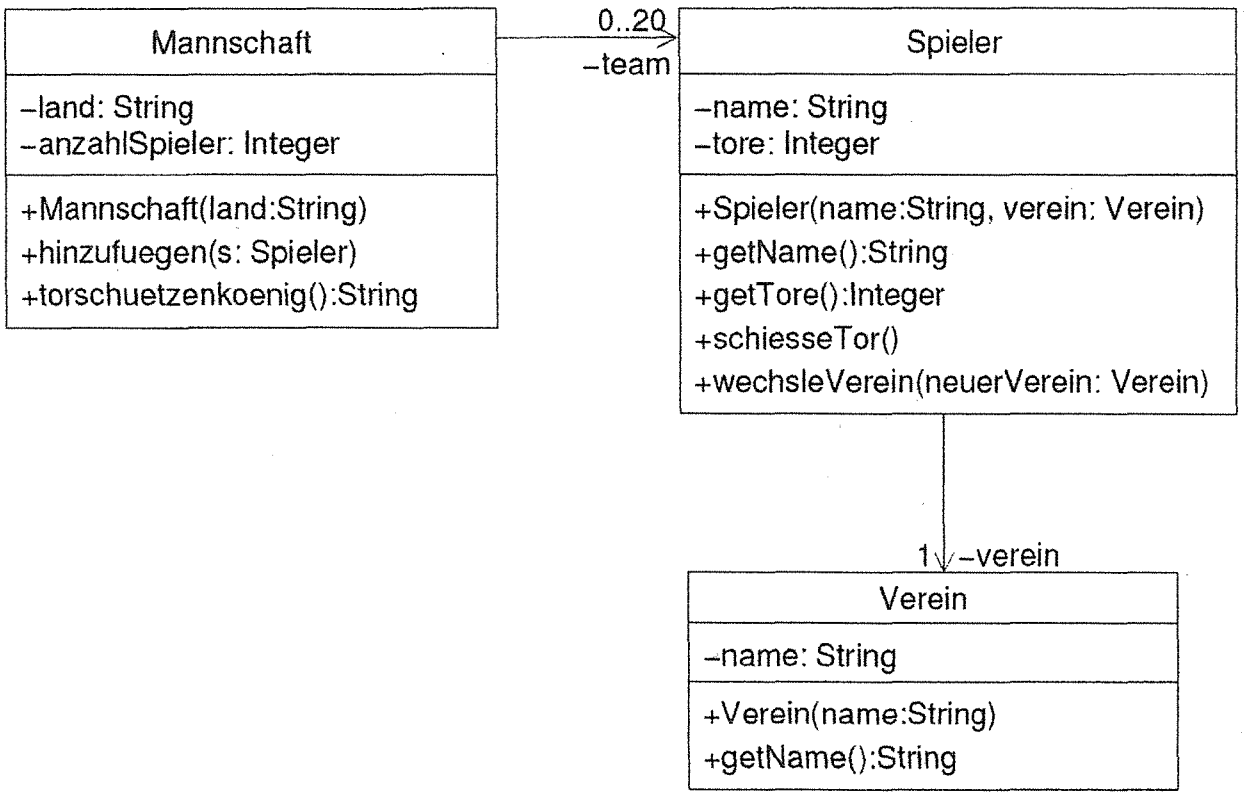


Abbildung 4: UML-Klassendiagramm

Es kann angenommen werden, dass die Klasse `Verein` bereits implementiert ist. In den folgenden Implementierungsaufgaben können Sie eine objektorientierte Programmiersprache Ihrer Wahl verwenden. Die verwendete Sprache ist anzugeben. Zu beachten sind jeweils die im Klassendiagramm angegebenen Sichtbarkeiten von Attributen, Rollennamen, Konstruktoren und Operationen.

a) Es ist eine Implementierung der Klasse `Spieler` anzugeben. Der Konstruktor soll die Instanzvariablen mit den gegebenen Parametern initialisieren, wobei die Anzahl der Tore nach der Objekterzeugung gleich 0 sein soll. Ansonsten kann die Funktionalität der einzelnen Operationen aus deren Namen geschlossen werden.

b) Es ist eine Implementierung der Klasse `Mannschaft` anzugeben. Der Konstruktor soll das Land initialisieren, die Anzahl der Spieler auf 0 setzen und das Team mit einem noch "leeren" Array der Länge 20 initialisieren. Das Team soll mit der Methode `hinzufuegen` um einen Spieler erweitert werden. Die Methode `torschuetzenkoenig` soll den Namen eines Spielers aus dem Team zurückgeben, der die meisten Tore für die Mannschaft geschossen hat. Ist

Fortsetzung nächste Seite!

das für mehrere Spieler der Fall, dann kann der Name eines beliebigen solchen Spielers zurückgegeben werden. Ist noch kein Spieler im Team, dann soll der String `"Kein Spieler vorhanden"` zurückgegeben werden.

c) Schreiben Sie den Rumpf einer `main`-Methode, so dass nach Ausführung der Methode eine deutsche Mannschaft existiert mit zwei Spielern Namens "Hugo Meier" und "Frank Huber". Beide Spieler sollen zum selben Verein "FC Staatsexamen" gehören. "Hugo Meier" soll nach Aufnahme in die deutsche Mannschaft genau ein Tor geschossen haben, während "Frank Huber" noch kein Tor erzielt hat. (Wir abstrahieren hier von der Realität, in der ein 2-er Team noch gar nicht spielbereit ist.)

Teilaufgabe 2

1. Grundlagen

Beantworten Sie die folgenden Fragen und begründen oder erläutern Sie Ihre Antwort in jeweils ein bis zwei Sätzen.

a) Sind die Tupel in einer Relation in Einfügereihenfolge abgelegt?

b) Muss eine Transaktion die Datenbank in einem konsistenten Zustand hinterlassen?

c) Was versteht man unter Persistenz?

d) Aus wie vielen Attributen kann der Primärschlüssel einer Relation minimal und aus wie vielen maximal bestehen?

e) Was versteht man unter referentieller Integrität?

2. ER-Modellierung

Im Folgenden finden Sie die Beschreibung einer Terminverwaltungssoftware. Erstellen Sie zu dieser Beschreibung ein erweitertes ER-Diagramm. Kennzeichnen Sie die Primärschlüssel durch passendes Unterstreichen und geben Sie die Kardinalitäten in Chen-Notation (= Funktionalitäten) an. Kennzeichnen Sie auch die totale Teilnahme (= Existenzabhängigkeit, Partizipität) von Entitytypen.

Ein Termin hat einen Startzeitpunkt und einen Endzeitpunkt, sowie eine Beschreibung. Eindeutig gekennzeichnet ist ein Termin durch seine ID. Ein Termin kann ein Serientermin sein. Serientermine haben zusätzlich zu den Attributen anderer Temine eine Wiederholungsperiode und eine Wiederholungsanzahl.

In der Software werden auch Personen verwaltet. Eine Person ist eindeutig durch ihre E-Mail-Adresse gekennzeichnet und hat einen Vornamen, einen Nachnamen sowie ein Geburtsdatum. Zu jeder Person ist auch noch das Alter abrufbar, das sich aus dem Geburtsdatum ergibt. Personen können an Terminen teilnehmen.

Weiterhin werden Räume verwaltet. Räume befinden sich in Gebäuden und haben innerhalb ihres Gebäudes eine eindeutige Raumnummer. Weiterhin wird zu Räumen deren Kapazität abgelegt. Gebäude haben eine eindeutige Adresse, die sich aus Postleitzahl, Straße und Hausnummer zusammensetzt. Personen können für Termine Räume buchen. Ein bestimmter Raum wird für einen bestimmten Termin immer von höchstens einer Person gebucht. Zu jeder Raumbuchung wird das Datum der Buchung gespeichert.

Fortsetzung nächste Seite!

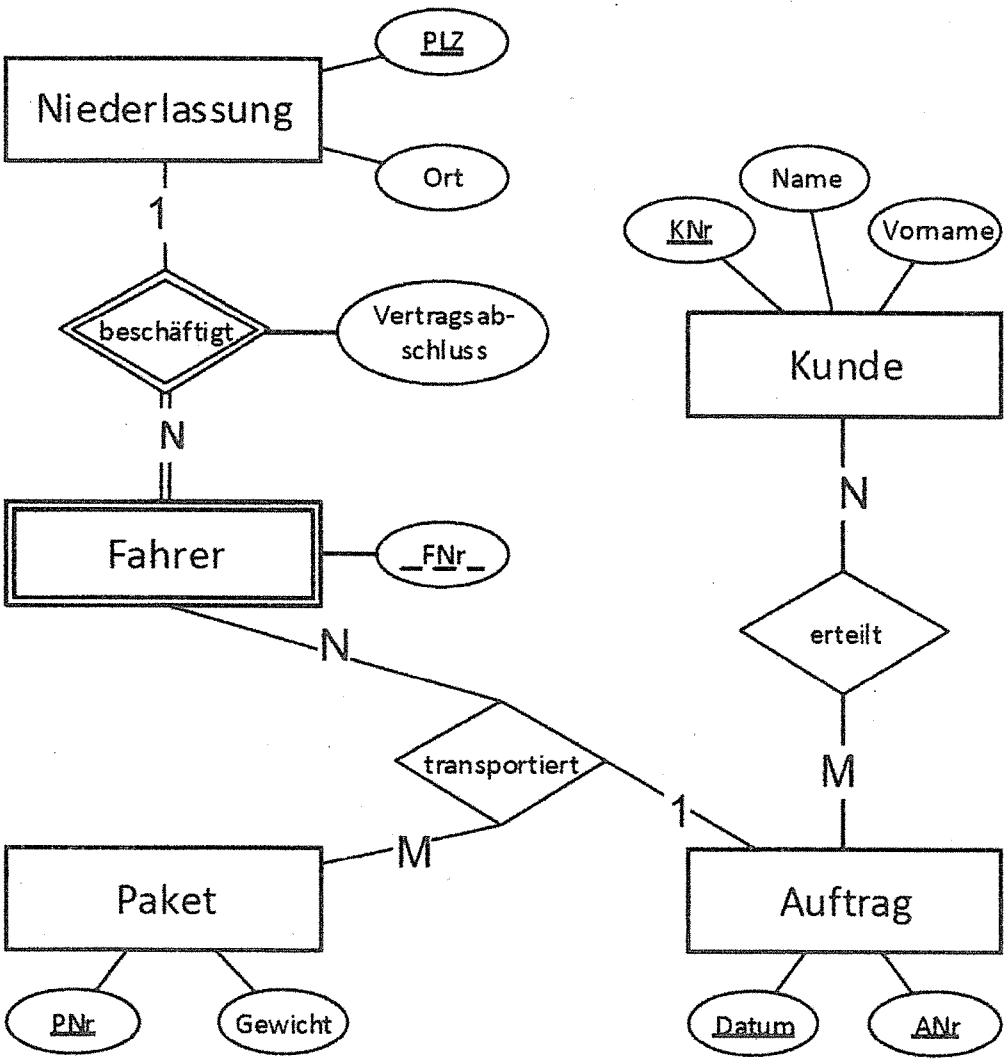
3. Relationaler Entwurf

Entwerfen Sie zum untenstehenden ER-Diagramm ein Relationenschema in dritter Normalform (3 NF) mit möglichst wenig Relationen.

Verwenden Sie dabei folgende Notation: Primärschlüssel werden durch Unterstreichen gekennzeichnet, Fremdschlüssel durch die Nennung der Relation, auf die sie verweisen, in eckigen Klammern hinter dem Fremdschlüsselattribut. Attribute zusammengesetzter Fremdschlüssel werden durch runde Klammern als zusammengehörig markiert.

Beispiel:

```
Relation1(Primärschlüssel, Attribut1, Attribut2,
Fremdschlüsselattribut1[Relation1],
(Fremdschlüssel2_Attribut1, Fremdschlüssel2_Attribut2)[Relation2])
```



Fortsetzung nächste Seite!

4. SQL

Gegeben sind folgende Relationen aus der Schulaufgaben-Planungssoftware einer Schule:

```
Schueler (ID, Nachname, Vorname)
Kurs(Bezeichnung, Jahrgang, Lehrername)
Mitglied(ID[Schueler], (Bezeichnung, Jahrgang)[Kurs])
Schulaufgabe((Bezeichnung, Jahrgang)[Kurs], Nummer, Datum)
```

Verwenden Sie im Folgenden nur Standard-SQL und keine produktspezifischen Erweiterungen. Sie dürfen bei Bedarf Views anlegen. Geben Sie einen Datensatz nicht mehrfach aus.

a) Schreiben Sie eine SQL-Anweisung, die die Tabelle Schulaufgabe (inklusive Schlüsseln) anlegt. Verwenden Sie sinnvolle Datentypen.

b) Schreiben Sie eine SQL-Anweisung, die Datum, Bezeichnung, Jahrgang und Nummer aller Schulaufgaben ausgibt, an denen Schüler mit dem Nachnamen „Mustermann" teilnehmen. Die Ausgabe soll nach Datum absteigend sortiert sein.

c) Schreiben Sie eine SQL-Anweisung, die Nachname und ID aller Schüler ausgibt, die in mindestens einem Kurs Mitglied sind, in dem noch ein oder mehrere andere Schüler mit gleichem Vornamen und Nachnamen Mitglied sind.

d) Schreiben Sie eine SQL-Anweisung, die ID, Nachname und Vorname aller Schüler ausgibt, die in keinem Kurs Mitglied sind.

e) Schreiben Sie eine SQL-Anweisung, die alle Schulaufgaben ermittelt, die an einem Datum liegen, an dem mindestens einer der teilnehmenden Schüler noch mindestens eine weitere Schulaufgabe hat. Ausgegeben werden sollen Bezeichnung und Jahrgang des Kurses, die Nummer der Schulaufgabe, sowie die Anzahl der betroffenen Schüler.

f) Schreiben Sie eine SQL-Anweisung, die alle Kurse löscht, die weder Mitglieder noch Schulaufgaben haben.

5. Normalformen

Gegeben ist die Relation $R(a_1, a_2, a_3, a_4, a_5)$. Sie besitze den Schlüsselkandidaten a_2a_4 und die funktionalen Abhängigkeiten $a_4 \rightarrow a_5$ und $a_3 \rightarrow a_1$.

a) Begründen Sie in ein bis zwei Sätzen, warum R sich nicht in zweiter Normalform befindet. Zerlegen Sie R so, dass die Ergebnisrelationen die zweite Normalform erfüllen. Erzeugen Sie dabei so wenig Relationen wie möglich.

b) Begründen Sie in ein bis zwei Sätzen, warum sich die Ergebnisrelationen der vorherigen Teilaufgabe nicht alle in der dritten Normalform befinden. Zerlegen Sie sie so weiter, dass alle Relationen die dritte Normalform erfüllen. Erzeugen Sie dabei so wenig Relationen wie möglich.

Thema Nr. 2

Teilaufgabe 1

Aufgabe 1: „Entwurfsmuster"

Das sogenannte **Vermittler**-Muster (*Mediator Pattern*) ist eines der ursprünglichen GoF-Verhaltensmuster.

a) Stellen Sie das allgemeine **Vermittler**-Muster als UML-Klassendiagramm dar und beschreiben Sie kurz die einzelnen Akteure des Musters zusammen mit ihrer Rolle in Ihrem Diagramm.

b) Nennen und begründen Sie kurz je einen Vorteil und einen Nachteil bei der Anwendung des **Vermittler**-Musters.

c) Beschreiben Sie kurz zwei Anwendungsbeispiele für das **Vermittler**-Muster. Welche Varianz oder nachträgliche Änderung an der Anwendungslogik könnte auftreten und wie bzw. wo würde man diese konkret implementieren?

d) Welches andere GoF-Entwurfsmuster ist dem **Vermittler**-Muster am ähnlichsten? Was sind die entscheidenden Unterschiede zwischen den beiden? Skizzieren und begründen Sie jeweils kurz, worin die Ähnlichkeiten bzw. Unterschiede bestehen.

Aufgabe 2: „Kopplung, Kohäsion, Komplexität"

a) Beschreiben Sie kurz jede der folgenden sechs Kopplungsarten nach IEEE 610 (Standard Glossary of Software Engineering Terminology):

   i. Pathologische Kopplung (pathological coupling)

   ii. Inhaltskopplung (content coupling)

   iii. Bereichskopplung (common-environment coupling)

   iv. Hybridkopplung (hybrid coupling)

   v. Kontrollkopplung (control coupling)

   vi. Datenkopplung (data coupling)

b) Beschreiben Sie kurz jede der folgenden Kohäsionsarten:

   i. Zeitliche Kohäsion

   ii. Prozedurale Kohäsion

   iii. Kommunikative Kohäsion

   iv. Sequentielle Kohäsion

Fortsetzung nächste Seite!

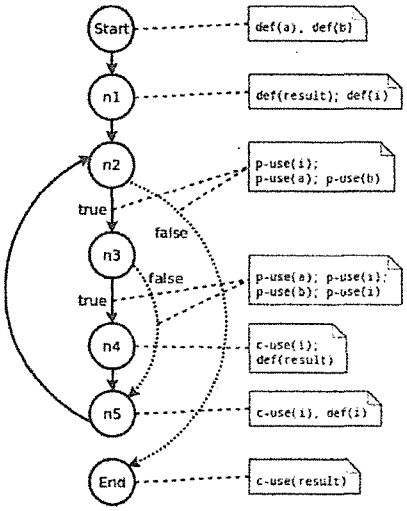
c) Die sogenannte zyklomatische Komplexität M nach McCabe stellt die Anzahl der linear unabhängigen Pfade auf dem Kontrollflussgraphen eines Moduls dar.

i. Geben Sie eine allgemeine Rechenvorschrift an, um M zu berechnen. Erklären Sie ggf. kurz die beteiligten Größen.

ii. Beschreiben Sie kurz den Zusammenhang zwischen M und der strukturellen Überdeckung (white-box coverage).

iii. Geben Sie die zyklomatische Komplexität der folgenden Methode ggT an:

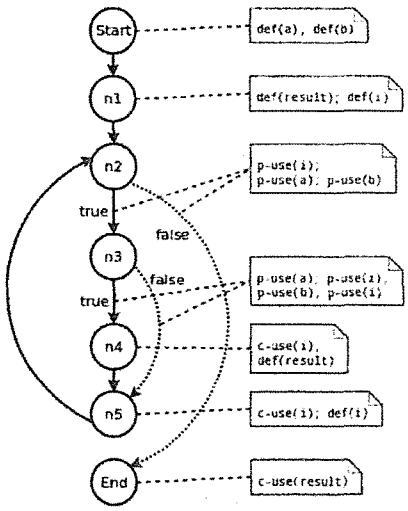
```
public int ggT(int a, int b) {
    int result = 1;
    for (int i = 1; i <= Math.min(a,b); i++) {
        if ((a%i==0)&(b%i==0)) {
            result = i;
        }
    }
    return result;
}
```



Aufgabe 3: „Testen"

Gegeben sei folgende Methode und ihr datenflussannotierter Kontrollflussgraph:

```
public int ggT(int a, int b) {
    int result = 1;
    for (int i = 1; i <= Math.min(a,b); i++) {
        if ((a%i==0)&(b%i==0)) {
            result = i;
        }
    }
    return result;
}
```



Fortsetzung nächste Seite!

a) Geben Sie je einen Repräsentanten aller Pfadklassen im Kontrollflussgraphen an, die zum Erzielen einer vollständigen

   i. Verzweigungsüberdeckung (Branch-Coverage, C_1)

   ii. Schleife-Inneres-Überdeckung (Boundary-Interior-Coverage, $C_{\infty,2}$) genügen würden.

b) Welche der vorangehend ermittelten Pfade sind mittels Testfälle tatsächlich überdeckbar („feasible")? Falls der Pfad ausführbar ist, geben Sie den Testfall (also die Werte der Parameter **a** und **b**) an – andernfalls begründen Sie kurz, weshalb der Pfad nicht überdeckbar ist.

   i. Falls Ihr vorangehend genannter Pfad für die C_1-Überdeckung nicht ausführbar ist, geben Sie nun einen anderen Pfad samt zugehörigem Testfall an, so dass Sie C_1-Überdeckung erreichen.

   ii. Geben Sie mindestens einen Pfad im Kontrollflussgraphen an, der *nicht* durch einen Testfall überdeckt werden kann („infeasible"). Begründen Sie kurz Ihre Antwort.

c) Geben Sie je einen Repräsentanten aller Pfadklassen im Kontrollflussgraphen an, die zum Erzielen einer vollständigen All-Uses-Überdeckung genügen würden. Nennen Sie bei jedem Pfad, welche def/use-Paare als Tripel (Variable, Knoten mit def, Knoten/Kante mit c/p-use!) Sie damit „überdecken".

   Zur Vereinfachung ist ein Pfad zusammen mit einem davon überdeckten Tripel vorgegeben – ergänzen Sie den Rest der Zusammenstellung:

   Pfad 1: `Start-n1-n2-End`

   - *Variable* `a`, *def in* `Start`, *p-use in* `(n2-End)`

   -

Teilaufgabe 2

Aufgabe 1: Allgemeine Fragen

a) Was versteht man unter der Anomalie Lost Update, die im Mehrbenutzerbetrieb auftreten kann? Geben Sie ein Beispiel (frei formuliert) an, in dem ein Lost Update auftritt.

b) Gegeben sei das Relationenschema $R = (\underline{A},B,C,D)$. Über die Relation sei bekannt, dass A ein Schlüssel ist, und es keine weiteren Schlüsselkandidaten gibt. Über weitere funktionale Abhängigkeiten sei nichts bekannt, es kann aber weitere funktionale Abhängigkeiten geben. Nehmen Sie an, dass R die erste Normalform erfüllt.

   i. Welche Normalformen erfüllt die Relation R noch? Begründen Sie Ihre Aussage.

   ii. Welche Normalformen erfüllt die Relation R möglicherweise nicht? Begründen Sie Ihre Aussage.

Fortsetzung nächste Seite!

c) Gegeben seien die Relationen $R = (A,B,C)$ und $S = (C,D,E)$. Relation R enthalte 50 Tupel und Relation S enthalte 10 Tupel. Gegeben seien außerdem folgende Anfragen:

A_1:

```
SELECT *
FROM R, S;
```

A_2:

```
SELECT *
FROM R NATURAL JOIN S;
```

- Wieviele Ergebnistupel liefert die Anfrage A_1?
- Wieviele Attribute enthält die Ergebnisrelation von Anfrage A_1?
- Wieviele Ergebnistupel liefert die Anfrage A_2 **mindestens**?
- Wieviele Ergebnistupel liefert die Anfrage A_2 **höchstens?**
- Wieviele Attribute enthält die Ergebnisrelation von Anfrage A_2?

d) Was versteht man unter dem "Cursor-Konzept" und weshalb wird es eingesetzt?

Aufgabe 2: Tupel- und Bereichskalkül

Gegeben sei das folgende Datenbankschema, alle Attribute seien vom Datentyp String:

```
Linie(LNr, LName, LZielort)
Zug(ZNr, LNr, Abfahrt)
Fahrgast(FGNr, FGName, FGAlter)
Fahrt(ZNr, FGNr, Start)
```

Geben Sie für die folgende verbal formulierte Anfrage jeweils einen aquivalenten Ausdruck **a) im Tupelkalkül und b) im Bereichskalkül** an.

Gesucht sind Nummer, Name, und Alter aller Fahrgäste, die niemals einen Zug nach Bielefeld genommen haben.

Tupelkalül:

Bereichskalkül:

Fortsetzung nächste Seite!

Aufgabe 3: SQL-Anfragen und ER-Diagramm

Gegeben sei das folgende Datenbank-Schema, das für die Speicherung der Daten einer Schule entworfen wurde. Die Primärschlüssel-Attribute sind jeweils unterstrichen.
Die Relation *Lehrer* enthält allgemeine Daten zu den Lehrern, nämlich sein 1. und 2. Unterrichtsfach, sowie den Spitznamen, den er von den Schülern bekommt und das Kürzel, mit dem er unterschreibt. Lehrer unterrichten Klassen in jeweils maximal einem bestimmten Fach.

Eine Klasse wird eindeutig bestimmt durch die Jahrgangsstufe sowie einen Buchstaben und hat ein ihr zugewiesenes Klassenzimmer.

Jeder Schüler ist in genau einer Klasse, hat ein Lieblingsfach und eine Durchschnittsnote und ist eindeutig bestimmt durch seinen Namen.

Die Durchschnittsnote, die Dauer und das Datum einer Schulaufgabe hängen davon ab, welcher Lehrer sie in welcher Klasse stellt und die wievielte Schulaufgabe es ist. Für jede Schulaufgabe wird eine Anwesenheitsliste geführt, auf der steht, welche Schüler daran teilgenommen haben, in die auch alle Noten eingetragen werden.

Lehrer (LName, Fach1, Fach2, Spitzname, Kuerzel)
Schueler (SName, Durchschnittsnote, Lieblingsfach, Jhgstufe, Buchstabe)
Unterrichtet (LName, Jhgstufe, Buchstabe, Fach)
Klasse (Jhgstufe, Buchstabe, Klassenzimmer)
Schulaufgabe (LName, Jhgstufe, Buchstabe, Nummer, Durchschnitt, Dauer, Datum)
NimmtTeil (SName, LName, Jhgstufe, Buchstabe, Nummer, Note)

a) Erstellen Sie ein vollständiges Entity-Relationship-Diagramm!

Formulieren Sie die folgenden Anfragen in SQL.

b) Geben Sie die Anweisung in **SQL-DDL** an, die notwendig ist, um die Relation **NimmtTeil** zu erzeugen. Achten Sie dabei auf Fremdschlüsselbeziehungen. Es sollen nur die Noten 1, 2, 3, 4, 5 und 6 erlaubt sein.

c) Geben Sie den Namen des Schülers mit der besten Durchschnittsnote an.

d) Geben Sie an, wie viele Schüler Herr Meier insgesamt unterrichtet.

e) Geben Sie für alle Schüler der 8b den Namen und den jeweiligen Durchschnitt aller Noten von Schulaufgaben an, die seit dem 15.12.2017 geschrieben wurden.

Fortsetzung nächste Seite!

f) Gesucht ist eine Liste von Schulaufgaben, die Frau Sommer gestellt hat, die jeweils die Nummer und die vollständige Bezeichnung der Klasse, in der sie gehalten wurde, enthält, sowie die Durchschnittsnote der Schulaufgabe selbst und die Gesamtdurchschnittsnote der ganzen Klasse.

Aufgabe 4: Normalisierung

Gegeben sei das Relationenschema $R(A,B,C,D,E,F)$, sowie die Menge der zugehörigen funktionalen Abhängigkeiten F:

- $C \rightarrow B$
- $B \rightarrow A$
- $CE \rightarrow D$
- $E \rightarrow F$
- $CE \rightarrow F$
- $C \rightarrow A$

a) Bestimmen Sie den Schlüsselkandidaten der Relation R und begründen Sie, warum es keine weiteren Schlüsselkandidaten gibt.

b) Überführen Sie das Relationenschema R mit Hilfe des Synthesealgorithmus in die dritte Normalform. Führen Sie hierfür jeden der vier Schritte durch und kennzeichnen Sie Stellen, bei denen nichts zu tun ist.