Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

Herbst 2019

66116

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

— Prüfungsaufgaben —

Fach: **Informatik (vertieft studiert)**

Einzelprüfung: **Datenbanksysteme, Softwaretechnologie**

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

Anzahl der Druckseiten dieser Vorlage: **17**

Bitte wenden!

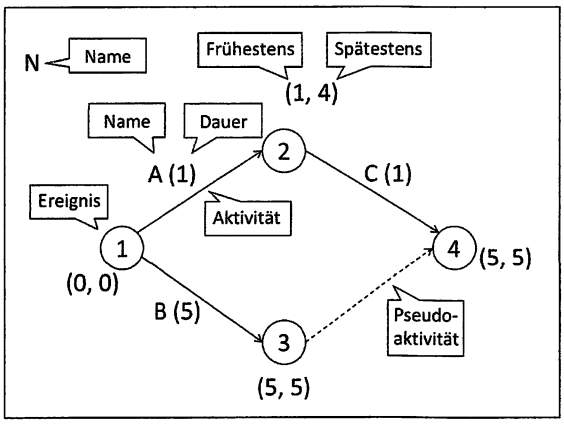
Thema Nr. 1

Teilaufgabe I: Softwaretechnologie

Aufgabe 1: (Klassen- und Objektdiagramme) (30 Punkte)

Ein CPM-Netzwerk („Critical Path Method") ist ein benannter Projektplan, der aus Ereignissen und Aktivitäten besteht. Ein Ereignis wird durch eine ganze Zahl > 0 identifiziert. Jede Aktivität führt von einem Quellereignis zu einem Zielereignis. Eine reale Aktivität hat einen Namen und eine Dauer (eine ganze Zahl > 0). Eine Pseudoaktivität ist anonym. Ereignisse und Pseudoaktivitäten verbrauchen keine Zeit. Zu jedem Ereignis gibt es einen frühesten und einen spätesten Zeitpunkt (eine ganze Zahl ≥ 0), deren Berechnung nicht Gegenstand der Aufgabe ist.

Beispiel:



a) Erstellen Sie ein UML-Klassendiagramm zur Modellierung von CPM-Netzwerken. Geben Sie für Attribute jeweils den Namen und den Typ an. Geben Sie für Assoziationen den Namen und für jedes Ende den Rollennamen und die Multiplizität an. Nutzen Sie ggf. abstrakte Klassen, Vererbung, Komposition oder Aggregation. Verzichten Sie auf Operationen und Sichtbarkeiten.

b) Erstellen Sie für das Klassendiagramm aus a) und das Beispiel aus der Aufgabenstellung ein Objektdiagramm. Geben Sie Rollennamen nur an, wenn es notwendig ist, um die Enden eines Links (Instanz einer Assoziation) zu unterscheiden.

Fortsetzung nächste Seite!

Aufgabe 2: (Anwendungsfall- und Aktivitätsdiagramme) (30 Punkte)

Eine Dienstreise an einer Universität wird folgendermaßen abgewickelt: Ein Mitarbeiter erstellt einen Dienstreiseantrag und legt ihn seinem Vorgesetzten zur Unterschrift vor. Mit seiner Unterschrift befürwortet der Vorgesetzte die Dienstreise. Verweigert er die Unterschrift, wird der Vorgang abgebrochen. Nach der Befürwortung wird der Antrag an die Reisekostenstelle weitergeleitet, die über die Annahme des Antrags entscheidet. Im Falle einer Ablehnung wird der Vorgang abgebrochen; sonst genehmigt die Reisekostenstelle den Antrag. Der Mitarbeiter kann nun einen Abschlag beantragen. Stimmt der Vorgesetzte zu, so entscheidet die Reisekostenstelle über den Abschlag und weist ggf. die Kasse der Universität an, den Abschlag auszuzahlen. Nach Ende der Dienstreise erstellt der Mitarbeiter einen Antrag auf Erstattung der Reisekosten an die Reisekostenstelle. Die Reisekostenstelle setzt den Erstattungsbetrag fest und weist die Kasse an, den Betrag auszuzahlen.

a) Erstellen Sie ein Anwendungsfalldiagramm (Use-Case-Diagramm) für Dienstreisen.

b) Erstellen Sie ein Aktivitätsdiagramm, das den oben beschriebenen Ablauf modelliert. Ordnen Sie den Aktionen die Akteure gemäß a) zu. Beschränken Sie sich auf den Kontrollfluss (keine Objektflüsse).

Fortsetzung nächste Seite!

Aufgabe 3: (White-Box-Tests) (30 Punkte)

Eine Dezimalzahl hat ein optionales Vorzeichen, dem eine nichtleere Sequenz von Dezimalziffern folgt. Der anschließende gebrochene Anteil ist optional und besteht aus einem Dezimalpunkt, gefolgt von einer nichtleeren Sequenz von Dezimalziffern.

Die folgende Java-Methode erkennt, ob eine Zeichenfolge eine Dezimalzahl ist:

```
 1  public static boolean istDezimalzahl(char[] zeichen){
 2      boolean resultat; int laenge = zeichen.length;
 3      if (laenge == 0)
 4          resultat = false;
 5      else {
 6          int i = 0;
 7          if (zeichen[i] == '+' || zeichen[i] == '-')
 8              i++;
 9          int j = i;
10          while (i < laenge && '0' <= zeichen[i] && zeichen[i] <= '9')
11              i++;
12          if (i == j)
13              resultat = false;
14          else {
15              if (i < laenge && zeichen[i] == '.')
16                  do
17                      i++;
18                  while (i < laenge && '0' <= zeichen[i] &&
                            zeichen[i] <= '9');
19              resultat = i == laenge && '0' <= zeichen[i-1] &&
                            zeichen[i-1] <= '9';
20          }
21      }
22      return resultat;
    }
```

a) Konstruieren Sie zu dieser Methode einen Kontrollflußgraphen. Markieren Sie dessen Knoten mit Zeilennummern des Quelltexts.

b) Geben Sie eine minimale Testmenge an, die das Kriterium der Knotenüberdeckung erfüllt. Geben Sie für jeden Testfall den durchlaufenen Pfad in der Notation 1 -> 2 -> ... an.

c) Verfahren Sie wie in b) für das Kriterium der Kantenüberdeckung.

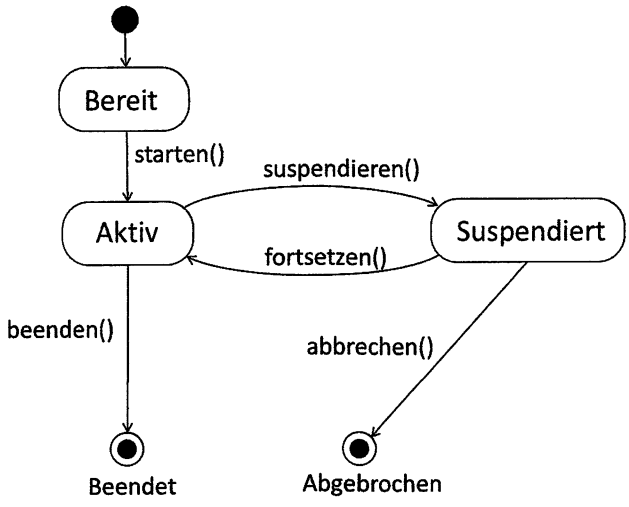
d) Wie stehen die Kriterien der Knoten- und Kantenüberdeckung zueinander in Beziehung? Begründen Sie Ihre Anwort.

Hinweis: Eine Testmenge ist minimal, wenn es keine andere Testmenge mit einer kleineren Zahl von Testfällen gibt. Die Minimalität braucht nicht bewiesen zu werden.

Fortsetzung nächste Seite!

Aufgabe 4: (Entwurfsmuster) (30 Punkte)

Zu den Aufgaben eines Betriebssystems zählt die Verwaltung von Prozessen. Jeder Prozess durchläuft verschiedene Zustände; Transitionen werden durch Operationsaufrufe ausgelöst. Folgendes Zustandsdiagramm beschreibt die Verwaltung von Prozessen:



Implementieren Sie dieses Zustandsdiagramm in einer Programmiersprache Ihrer Wahl mit Hilfe des Zustandsmusters; geben Sie die gewählte Sprache an. Die Methoden für die Transitionen sollen dabei die Funktionalität der Prozessverwaltung simulieren, indem der Methodenaufruf auf der Standardausgabe protokolliert wird. Falls Transitionen im aktuellen Zustand undefiniert sind, soll eine Fehlermeldung ausgegeben werden.

Teilaufgabe II: Datenbanken

Aufgabe 1: (Wissensfragen) (9 Punkte)

Antworten Sie kurz und prägnant.

1. [1 Punkt] Nennen Sie einen Vorteil und einen Nachteil der Schichtenarchitektur.

2. [1 Punkt] Wie ermöglicht es ein Datenbankensystem, verschiedene Sichten darzustellen?

3. [3 Punkte] Was beschreibt das Konzept der Transitiven Hülle? Erklären Sie dies kurz und nennen Sie ein Beispiel für (1) die Transitive Hülle eines Attributes bei funktionalen Abhängigkeiten und (2) die Transitive Hülle einer SQL Anfrage.

4. [4 Punkte] Nennen Sie zwei Indexstrukturen und beschreiben Sie jeweils ihren Vorteil.

Fortsetzung nächste Seite!

Aufgabe 2: (ER-Modellierung) (40 Punkte)

Entwerfen Sie ein ER-Diagramm für eine Schule aus einer imaginären Film-Reihe. Geben Sie alle Attribute an und unterstreichen Sie Schlüsselattribute. Für die Angabe der Kardinalitäten von Beziehungen soll die Min-Max-Notation verwendet werden. Führen Sie wenn nötig einen Surrogatschlüssel ein.

An der Schule werden Schüler ausgebildet. Sie haben einen Namen, ein Geschlecht und ein Alter. Da die Schule klein ist, ist der Name eindeutig. Jeder Schüler ist Teil seines Jahrgangs, bestimmt durch Jahr und Anzahl an Schüler (Jahrgänge ohne Schüler sind erlaubt), und besucht mit diesem Kurse. Dabei wird jeder Kurs von min. einem Jahrgang besucht und jeder Jahrgang hat zwischen 2 und 5 Kurse. Kurse haben einen Veranstaltungsort und einen Namen.
Außerdem wird jeder Schüler einem von vier Häusern zugeordnet. Diese Häuser sind Gryffindor, Slytherin, Hufflepuff und Ravenclaw. Jedes Haus hat eine Anzahl an Mitgliedern.

Um die Organisation an der Schule zu erleichtern, gibt es pro Haus einen Vertrauensschüler und pro Jahrgang einen Jahrgangssprecher. Außerdem können Schüler Quidditch spielen. Dabei können sie die Rollen Sucher, Treiber, Jäger und Hüter spielen. Jedes Haus der Schule hat eine Mannschaft. Diese besteht aus genau einem Sucher, einem Hüter, drei Jäger und zwei Treiber. Jedes Jahr gibt es an der Schule eine Trophäe zu gewinnen. Diese ist abhängig von der Mannschaft und weiterhin durch das Jahr identifiziert.

Fortsetzung nächste Seite!

Aufgabe 3: (SQL) (30 Punkte)

Formulieren Sie folgende Anfragen in SQL gegen die angegebene Datenbank aus einer imaginären Serie.

Figur = {Id, Name, Schwertkunst, Lebendig, Titel}
gehört_zu = {Id, Familie,
FK (Id) references Figur(Id),
FK (Familie) references Familie(Id) }
Familie = {Id, Name, Reichtum, Anführer}
Drache = {Name, Lebendig}
besitzt = {Id, Name,
FK (Id) references Figur(Id),
FK (Name) references Drache(Name) }
Festung = {Name, Ort, Ruine}
besetzt = {Familie, Festung,
FK (Familie) references Familie(Id),
FK (Festung) references Festung(Name) }
lebt = {Id, Name,
FK (Id) references Figur(Id),
FK (Name) references Festung(Name) }

1. [2 Punkte] Geben Sie für alle Figuren an, wie oft alle vorhandenen Titel vorkommen.

2. [3 Punkte] Welche Figuren (Name ist gesucht) kommen aus "Kings Landing"?

3. [5 Punkte] Geben Sie für jede Familie (Name) die Anzahl der zugehörigen Charaktere und Festungen an.

4. [11 Punkte] Gesucht sind die besten fünf Schwertkämpfer aus Festungen aus dem Ort "Westeros". Es soll der Name, die Schwertkunst und die Platzierung ausgegeben werden. Die Ausgabe soll nach der Platzierung sortiert erfolgen.

5. [2 Punkte] Schreiben Sie eine Anfrage, die alle Figuren löscht, die tot sind. Das Attribut Lebendig kann dabei die Optionen "ja" und "nein" annehmen.

6. [2 Punkte] Löschen Sie die Spalten "Lebendig" aus der Datenbank.

7. [5 Punkte] Erstellen Sie eine weitere Tabelle mit dem Namen Waffen, welche genau diese auflistet. Eine Waffe ist genau einer Figur zugeordnet, hat einen eindeutigen Namen und eine Stärke zwischen 0 und 5. Wählen Sie sinnvolle Typen für die Attribute.

Fortsetzung nächste Seite!

Aufgabe 4: (Relationale Algebra) (14 Punkte)

Übertragen Sie die folgenden Ausdrücke in die relationale Algebra. Beschreiben Sie diese Ausdrücke umgangssprachlich, bevor Sie die Umformung durchführen. Das Schema der Datenbank entspricht dem Schema aus Aufgabe 3.

1. [6 Punkte] $\{f \mid f \in \text{Figur} \wedge \neg\exists g \in \text{gehört\_zu}(f.\text{Id} = g.\text{Id})\}$

2. [8 Punkte] $\{f \mid f \in \text{Figur} \wedge \exists l \in \text{lebt}(l.\text{Id} = c.\text{Id}) \wedge \exists f \in \text{Festung}(l.\text{Festung} = f.\text{Name}) \wedge \exists b \in \text{besetzt}(f.\text{Name} = b.\text{Festung}) \wedge \exists f2 \in \text{Familie}(b.\text{Familie} = f2.\text{Id}) \wedge f2.\text{Name} = \text{Stark})\}$

Fortsetzung nächste Seite!

Aufgabe 5: (Entwurfstheorie) (27 Punkte)

Gegeben sind folgende funktionale Abhängigkeiten.

$$X \rightarrow YZ$$
$$Z \rightarrow WX$$
$$Q \rightarrow XYZ$$
$$V \rightarrow ZW$$
$$ZW \rightarrow YQV$$

1. [10 Punkte] Berechnen Sie die kanonische Überdeckung.

Gegeben ist folgende Tabelle.

JedID	Name	Rasse	Lichtschwert	Seite der Macht
2	Yoda	Unbekannt	Grün	Gute Seite
3	Anakin Skywalker	Mensch	Blau, Rot	Gute Seite, dunkle Seite
4	Mace Windou	Mensch	Lila	Gute Seite
5	Count Dooku	Mensch	Rot	Dunkle Seite
6	Ahsoka Tano	Togruta	Grün	Gute Seite
7	Yoda	Mensch	Rot	Dunkle Seite

2. [2 Punkte] Geben Sie zuerst die funktionalen Abhängigkeiten in der Tabelle an.

3. a) [1 Punkt] Geben Sie die zentrale Eigenschaft der 1. NF an.

   b) [3 Punkte] Nennen Sie alle Stellen, an denen das Schema die 1. NF verletzt.

   c) [3 Punkte] Überführen Sie die Tabelle in die 1. NF.

4. a) [1 Punkt] Geben Sie die Definition der 2. NF an.

   b) [2 Punkte] Arbeiten Sie bitte mit folgender, nicht korrekten Zwischenlösung weiter. Erläutern Sie, inwiefern dieses Schema die 2. NF verletzt.

JedID	Name	Rasse	Lichtschwert	Seite der Macht
2	Yoda	Unbekannt	Grün	Gute Seite
3	Skywalker	Mensch	Blau	Gute Seite
4	Windou	Mensch	Lila	Gute Seite
5	Dooku	Mensch	Rot	Dunkle Seite
6	Tano	Togruta	Grün	Gute Seite
2	Yoda	Mensch	Rot	Dunkle Seite

   c) [3 Punkte] Überführen Sie die Tabelle in die 2. NF.

5. a) [1 Punkt] Geben Sie die Definition der 3. NF an.

   b) [1 Punkt] Erläutern Sie, ob und wenn ja, wie das von Ihnen in 3c) neu erstellte Schema die 3. NF verletzt.

Thema Nr. 2

Teilaufgabe I: Softwaretechnologie

Aufgabe 1: (Modellierung und Muster) (75 Punkte)

Hierarchische Dateisysteme bestehen aus den FileSystemElements Ordner, Dateien und Verweise. Ein Ordner kann seinerseits Ordner, Dateien und Verweise beinhalten; jedem Ordner ist bekannt, welche Elemente (children) er enthält. Mit Ausnahme des Root-Ordners auf der obersten Hierarchieebene ist jeder Ordner, jede Datei und jeder Verweis Element eines Elternordners. Jedem Element ist bekannt, was sein Elternordner ist (parent). Ein Verweis verweist auf einen Verweis, eine Datei oder einen Ordner (link). Wenn ein Ordner gelöscht wird, werden alle seine Bestandteile ggf. rekursiv ebenfalls gelöscht. Sie dürfen die Lösungen für Aufgabenteil b) und c) in einem gemeinsamen Code kombinieren.

a) [23 Punkte] Modellieren Sie diesen Sachverhalt mit einem UML-Klassendiagramm. Benennen Sie die Rollen von Assoziationen und geben Sie alle Kardinalitäten an. Ihre Lösung soll mindestens eine sinnvolle Spezialisierungsbeziehung enthalten.

b) [20 Punkte] Implementieren Sie das Klassendiagramm als Java- oder C++-Programm. Jedes Element des Dateisystems soll mindestens über ein Attribut *name* verfügen. Übergeben Sie den Elternordner jedes Elements als Parameter an den Konstruktor; der Elternordner des Root-Ordners kann dabei als *null* implementiert werden. Dokumentieren Sie Ihren Code.

c) [17 Punkte] Ordnen Sie eine Methode *delete*, die Dateien, Ordner und Verweise rekursiv löscht, einer oder mehreren geeigneten Klassen zu und implementieren Sie sie. Zeigen Sie die Löschung jedes Elements durch eine Textausgabe von *name* an. *toString()* müssen Sie dabei nicht implementieren. Gehen Sie davon aus, dass Verweis- und Ordnerstrukturen azyklisch sind und dass jedes Element des Dateisystems höchstens einmal existiert. Wenn ein Verweis gelöscht wird, wird sowohl der Verweis als auch das verwiesene Element bzw. transitiv die Kette der verwiesenen Elemente gelöscht. Bedenken Sie, dass die Löschung eines Elements immer auch Konsequenzen für den dieses Element beinhaltenden Ordner hat. Es gibt keinen Punktabzug, wenn Sie die Löschung des Root-Ordners nicht zulassen.

d) [6 Punkte] Was kann im Fall von *delete* passieren, wenn die Linkstruktur zyklisch ist oder die Ordnerstruktur zyklisch ist? Kann es zu diesen Problemen auch dann führen, wenn weder die Linkstruktur zyklisch ist, noch die Ordnerstruktur zyklisch ist? Wie kann man im Programm das Problem lösen, falls man Zyklizitäten zulassen möchte?

e) [9 Punkte] Was ist ein Design Pattern? Nennen Sie drei Beispiele und erläutern Sie sie kurz. Welches Design Pattern bietet sich für die Behandlung von hierarchischen Teil-Ganzes-Beziehungen an, wie sie im Beispiel des Dateisystems vorliegen?

Fortsetzung nächste Seite!

Aufgabe 2: (Assertions) (17 Punkte)

Methoden in Programmen funktionieren nicht immer für alle möglichen Eingaben – klassische Beispiele sind die Quadratwurzel einer negativen Zahl oder die Division durch Null. Zulässige (bzw. in negierter Form unzulässige) Eingabewerte sollten dann spezifiziert werden, was mit Hilfe sog. assertions geschehen kann. Assertions prüfen zur Laufzeit den Wertebereich der Parameter einer Methode, bevor der Methodenkörper ausgeführt wird. Wenn ein oder mehrere Argumente zur Laufzeit unzulässig sind, wird eine Ausnahme geworfen.

Betrachten Sie das folgende Java-Programm:

```
public static double[][] magic(double[][] A, double[][] B, int m) {
      double[][] C = new double[m][m];
      for (int i = 0; i < m; i++)
            for (int j = 0; j < m; j++)
                  for (int k = 0; k < m; k++)
                        C[i][j] += A[i][k] * B[k][j];
      return C;
}
```

a) [2 Punkte] Beschreiben Sie kurz, was dieses Programm tut.

b) [8 Punkte] Implementieren Sie drei nützliche Assertions, die zusammen verhindern, dass das Programm abstürzt.

c) [4 Punkte] Wann und warum kann es sinnvoll sein, keine expliziten Assertions anzugeben? Erläutern Sie, warum das potentiell gefährlich ist.

d) [3 Punkte] Assertions können wie oben sowohl für Vorbedingungen am Anfang einer Methode als auch für Nachbedingungen am Ende einer Methode verwendet werden. Solche Spezifikationen bilden dann sog. Kontrakte. Kontrakte werden insbesondere auch statisch verwendet, d. h. nicht zur Laufzeit. Skizzieren Sie ein sinnvolles Anwendungsgebiet und beschreiben Sie kurz den Vorteil der Verwendung von Kontrakten in diesem Anwendungsgebiet.

Fortsetzung nächste Seite!

Aufgabe 3: (Softwarearchitektur und Agilität) (28 Punkte)

Die Komponentenarchitektur eines Softwaresystems beschreibt die unterschiedlichen Softwarebausteine, deren Schnittstellen und die Abhängigkeiten von Softwarekomponenten wie beispielsweise Klassen. Wir unterscheiden zwischen der Soll- und der Ist-Architektur eines Systems.

a) [6 Punkte] Nennen und definieren Sie drei Qualitätsattribute von Software, die durch die Architektur beeinflusst werden und charakterisieren Sie jeweils eine „schlechte" Architektur, die diese Qualitätsattribute negativ beeinflusst.

b) [5 Punkte] Erläutern Sie, was Information Hiding ist. Wie hängt Information Hiding mit Softwarearchitektur zusammen? Wie wird Information Hiding auf Klassenebene in Java implementiert? Gibt es Situationen, in denen Information Hiding auch negative Effekte haben kann?

c) [2 Punkte] Erklären Sie, was Refactoring ist.

d) [15 Punkte] Skizzieren Sie die Kernideen von Scrum inkl. der wesentlichen Prozessschritte, Artefakte und Rollen. Beschreiben Sie dann die Rolle von Ist- und Soll-Architektur in agilen Entwicklungskontexten wie Scrum.

Fortsetzung nächste Seite!

Teilaufgabe II: Datenbanken

Aufgabe 1: (ER-Modellierung) (35 Punkte)

Erstellen Sie ein möglichst einfaches ER-Schema, das alle gegebenen Informationen enthält. Attribute von Entitäten und Beziehungen sind anzugeben, Schlüsselattribute durch Unterstreichen zu kennzeichnen. Verwenden Sie für die Angabe der Kardinalitäten von Beziehungen die Min-Max-Notation. Führen Sie Surrogatschlüssel nur dann ein, wenn es nötig ist und modellieren Sie nur die im Text vorkommenden Elemente.

Ein örtlicher Sportverein möchte seine Vereinsangelegenheiten mittels einer Datenbank verwalten. Der Verein besteht aus verschiedenen Abteilungen, welche eine eindeutige Nummer und einen aussagekräftigen Namen besitzen. Für jede Abteilung soll zudem automatisch die Anzahl der Mitglieder gespeichert werden, wobei ein Mitglied zu mehreren Abteilungen gehören kann. Die Mitglieder des Vereins können keine, eine oder mehrere Rollen (auch Ämter genannt) einnehmen. So gibt es die Ämter: 1. Vorstand, 2. Vorstand, Kassier, Jugendleiter, Trainer sowie einen Abteilungsleiter für jede Abteilung. Es ist dabei auch möglich, dass ein Abteilungsleiter mehrere Abteilungen leitet oder ein Mitglied mehrere Aufgaben übernimmt, mit der Einschränkung, dass die Vorstandsposten und Kassier nicht von der gleichen Person ausgeübt werden dürfen. Zu jedem Trainer wird eine Liste von Lizenzen gespeichert. Jeder Trainer ist zudem in mindestens einer Abteilung eine bestimmte Anzahl von Stunden tätig. Zu allen Mitgliedern werden Mitgliedsnummer, Name (bestehend aus Vor- und Nachname), Geburtsdatum, E-Mail, Eintrittsdatum, Adresse (bestehend aus PLZ, Ort, Straße, Hausnummer), IBAN und die Vereinszugehörigkeit in Jahren gespeichert.

Im Verein fallen Finanztransaktionen an. Zu jeder Transaktion wird ein Zeitstempel, der Betrag und eine eindeutige Transaktionsnummer gespeichert. Die Mitglieder leisten Zahlungen an den Verein. Umgekehrt erstattet der Verein auch bestimmte Kosten. Im Verein existieren drei verschiedene Mitgliedsbeiträge. So gibt es einen Kinder-und-Jugendlichen-Tarif, einen Erwachsenentarif und einen Familientarif.

Mitgliedern entstehen des Öfteren Fahrtkosten. Für jede Fahrtkostenabrechnung werden das Datum, die gefahrenen Kilometer und Start und Ziel, der Zweck sowie das Mitglied gespeichert, welches den Antrag gestellt hat. Zu jeder Fahrtkostenabrechnung existiert genau eine Erstattung.

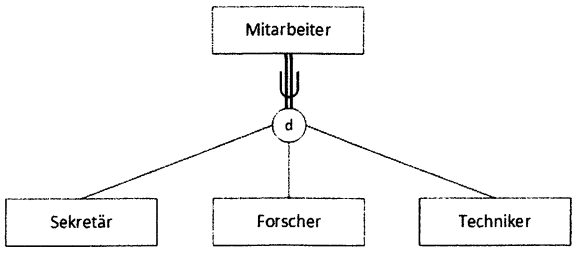
Durch die Teilnahme an verschiedenen Wettbewerben besteht die Notwendigkeit die von einem Team (also mehreren Mitgliedern zusammen) oder Mitgliedern erzielten sportlichen Erfolge, d. h. Platzierungen, zu verwalten. Jeder Wettkampf besitzt eine eindeutige ID, ein Datum und eine Kurzbeschreibung.

Das Vereinsleben besteht aus zahlreichen Terminen, die durch Datum und Uhrzeit innerhalb einer Abteilung eindeutig identifiziert werden können. Zu jedem Termin wird zusätzlich eine Kurzbeschreibung gespeichert.

Fortsetzung nächste Seite!

Aufgabe 2: (ER-Modellierung) (4 Punkte)

In einer Datenbank zur Mitarbeiterverwaltung werden die Mitarbeiter über ihr Ausscheiden aus dem Betrieb hinaus (z. B. Ruhestand oder Arbeitsplatzwechsel) gespeichert. Im Folgenden ist ein Ausschnitt aus dem ER-Diagramm dargestellt. Erweitern Sie das Diagramm um genau eine Entität, welche die derzeit aktiven Mitarbeiter aus allen Unterklassen umfasst. Benennen und erläutern Sie das von Ihnen verwendete Modellierungskonstrukt.



Aufgabe 3: (Relationale Algebra) (6 Punkte)

Gegeben seien die folgenden beiden Relationen:

R1	P	Q	S
	10	a	5
	15	b	8
	25	a	6

R2	A	B	C
	10	b	6
	25	c	3
	10	b	5

Geben Sie die Ergebnisse der folgenden relationalen Ausdrücke an:

a) [1,5 Punkte] $R_1 \bowtie_{R1.P=R2.A} R_2$ (Equi-Join)

b) [1,5 Punkte] $R_1 ⟕_{R1.Q=R2.B} R_2$ (Right-Outer-Join)

c) [3 Punkte] Es ist bekannt, dass die minimale Menge relationaler Operatoren Selektion, Projektion, Vereinigung, Differenz und kartesisches Produkt umfasst. Wie kann die Division zweier Relationen mit diesen Operatoren ausgedrückt werden? Begründen Sie kurz die einzelnen Bestandteile Ihres relationalen Ausdrucks.

Fortsetzung nächste Seite!

Aufgabe 4: (Normalisierung) (22 Punkte)

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

```
A B    -> C
A      -> D
F      -> B
D E    -> B
B      -> A
```

a) [5 Punkte] Bestimmen Sie sämtliche Schlüsselkandidaten der Relation `R` und begründen Sie, warum es keine weiteren Schlüsselkandidaten geben kann.

b) [8 Punkte] Ist die gegebene Menge an funktionalen Abhängigkeiten minimal? Fall sie minimal ist begründen Sie diese Eigenschaft ausführlich, anderenfalls minimieren Sie `FD` schrittweise. Vergessen Sie nicht die einzelnen Schritte entsprechend zu begründen.

c) [9 Punkte] Überführen Sie falls nötig das Schema in dritte Normalform. Ist die dritte Normalform bereits erfüllt, begründen Sie dies ausführlich.

Aufgabe 5: (Anfrageoptimierung) (10 Punkte)

Gegeben seien die beiden Relationen *Professor* und *Vorlesung* mit folgendem Umfang:
Relation Professor: 5 Spalten, 164 Datensätze
Relation Vorlesung: 10 Spalten, 333 Datensätze.

Optimieren Sie auf geeignete Weise folgende SQL-Anweisung möglichst gut und berechnen Sie wie stark sich die Datenmenge durch jede Optimierung reduziert (nehmen Sie für Ihre Berechnung an, dass Herr Mustermann genau zwei Vorlesungen hält). Geben Sie jeweils den Operatorbaum vor und nach Ihren jeweiligen Optimierungen an.

```
SELECT Titel FROM Professor, Vorlesung WHERE Name = ´Mustermann` AND PersNr =
gelesenVon;
```

Fortsetzung nächste Seite!

Aufgabe 6: (Vermischte Fragen) (25 Punkte)

Begründen oder erläutern Sie Ihre Antworten.

a) [2 Punkte] Erklären Sie kurz den Unterschied zwischen einem Natural-Join und einem Equi-Join.

b) [1 Punkt] Erläutern Sie kurz was man unter einem Theta-Join versteht.

c) [3 Punkte] Was versteht man unter Unionkompatibilität? Nennen Sie drei SQL-Operatoren welche Unionkompatibilität voraussetzen.

d) [4 Punkte] Erläutern Sie Backward und Forward Recovery und grenzen Sie diese voneinander ab.

e) [5 Punkte] Erklären Sie das Zwei-Phasen-Freigabe-Protokoll.

f) [5 Punkte] Erläutern Sie Partial Undo / Redo und Global Undo / Redo und deren Bedeutung für die Umsetzung des ACID-Prinzips. Geben Sie zu jeder dieser Konzepte an, ob System-, Programm- oder Gerätefehler damit korrigiert werden können.

g) [2 Punkte] Erklären Sie das WAL-Prinzip (Write ahead logging)!

h) [3 Punkte] Erklären Sie den Begriff „Datenbankindex“ und nennen Sie zwei häufige Arten.

Fortsetzung nächste Seite!

Aufgabe 7: (SQL) (18 Punkte)

Gegeben sind folgende Relationen aus einem Verwaltungssystem für die jährlichen Formel 1-Rennen:

```
Strecke(Strecken_ID, Streckenname, Land, Länge)
Fahrer(Fahrer_ID, Fahrername, Nation, Rennstall)
Rennen(Strecken_ID, Jahr, Wetter)
            FK (Strecken_ID) referenziert Strecke(Strecken_ID)
Rennteilnahme(Fahrer_ID, Strecken_ID, Jahr, Rundenbestzeit, Gesamtzeit,
disqualifiziert)
            FK (Fahrer_ID) referenziert Fahrer(Fahrer_ID)
            FK (Strecken_ID, Jahr) referenziert Rennen(Strecken_ID, Jahr)
```

Der Einfachheit halber wird angenommen, dass Fahrer den Rennstall nicht wechseln können. Das Attribut „disqualifiziert" kann die Ausprägungen „ja" und „nein" haben. Formulieren Sie folgende Abfragen in SQL. Vermeiden Sie nach Möglichkeit übermäßige Nutzung von Joins und Views.

a) [3 Punkte] Geben Sie für jeden Fahrer seine ID sowie die Anzahl seiner Disqualifikationen in den Jahren 2005 bis 2017 aus. Ordnen Sie die Ausgabe absteigend nach der Anzahl der Disqualifikationen.

b) [4 Punkte] Gesucht sind alle Länder, aus denen noch nie ein Fahrer disqualifiziert wurde.

c) [8 Punkte] Gesucht sind die ersten fünf Plätze des Rennens von 2011 in „Abu Dhabi" (Streckenname). Die Ausgabe soll nach der Platzierung absteigend erfolgen. Geben Sie Fahrer_ID, Fahrername, Nation und Rennstall mit aus.

d) [3 Punkte] Führen Sie eine neue Spalte Gehalt in die Tabelle Fahrer ein. Da sich die Prämien für die Fahrer nach einem Rennstallwechsel ändern, soll ein Trigger geschrieben werden, mit dem das Gehalt des betreffenden Fahrers um 10% angehoben wird.