Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

Herbst 2017

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: **18**

Bitte wenden!

Thema Nr. 1

Teilaufgabe 1

1. Grundwissen Datenbanksysteme

1. Benennen Sie die vier ACID-Eigenschaften.

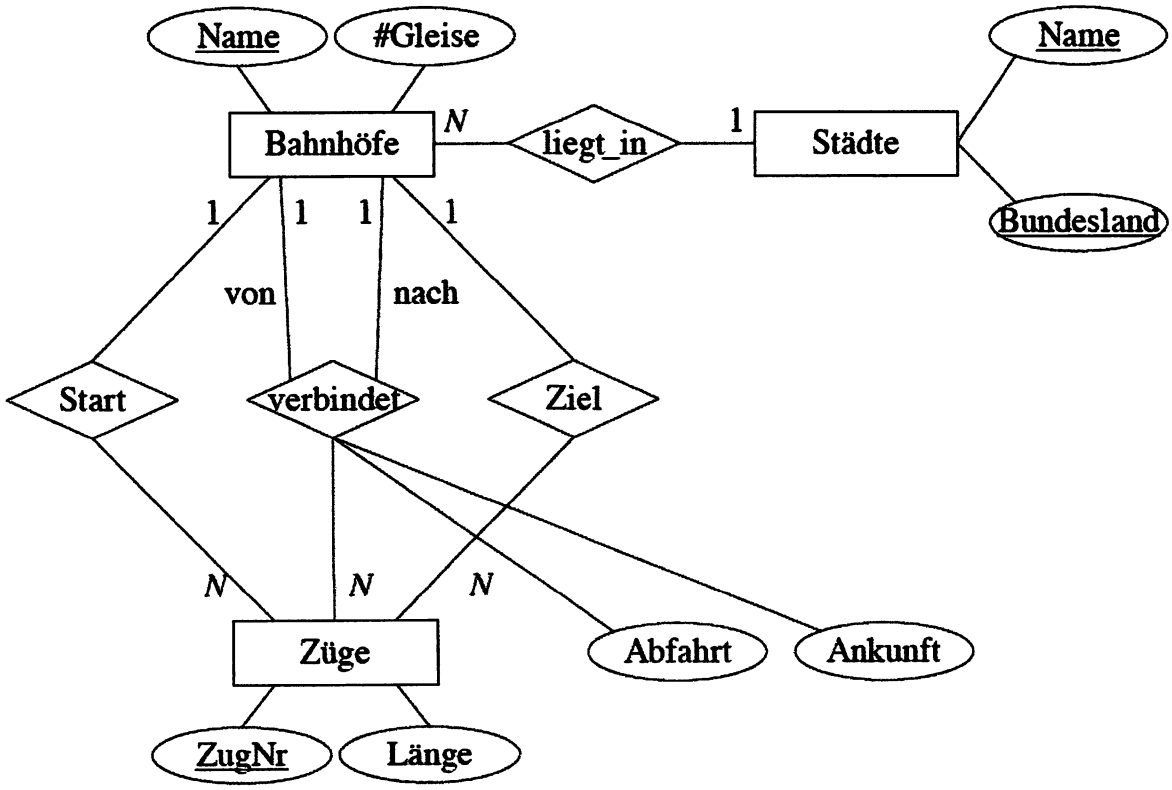
2. Grenzen Sie die Begriffe *Superschlüssel* und *Kandidatenschlüssel* mit einer kurzen Erklärung voneinander ab.

3. Nennen Sie zwei mögliche Vorteile, die durch den Zusammenschluss von Festplatten zu einem RAID-System entstehen.

4. Eignet sich eine Hash-Tabelle zur Bearbeitung von Bereichsanfragen? Begründen Sie Ihre Antwort und nennen Sie gegebenenfalls eine Alternative.

2. Relationale Modellierung

Das folgende *ER* -Diagramm modelliert ein Zugauskunftssystem.



Fortsetzung nächste Seite!

1. Setzen Sie das *ER*-Diagramm in ein relationales Schema um.

   - Fassen Sie, wenn möglich, Relationen zusammen.
   - Markieren Sie Primärschlüssel.
   - Markieren Sie Fremdschlüssel.
   - Wählen Sie sinnvolle Typen für die Attribute.

2. Geben Sie die Schemadefinitionen für die Tabellen *verbindet* und *Bahnhöfe* inklusive Primär- und Fremdschlüsseldefinitionen in SQL an.

3. Ist es in dem gegebenen Modell möglich einen Zug einzutragen, welcher von München nach Berlin fährt (Start & Ziel) und als Zwischenstation in Nürnberg hält? Begründen Sie Ihre Antwort und formulieren Sie die entsprechenden Befehle in SQL.
   Sie können davon ausgehen, dass es in der Datenbank bereits einen Bahnhof für Berlin, München und Nürnberg (mit selbigen Namen) gibt. Das Datum für Ankunft und Abfahrt ist dabei jeweils frei wählbar und muss nicht realistisch sein.

3. Konsistenzbedingungen

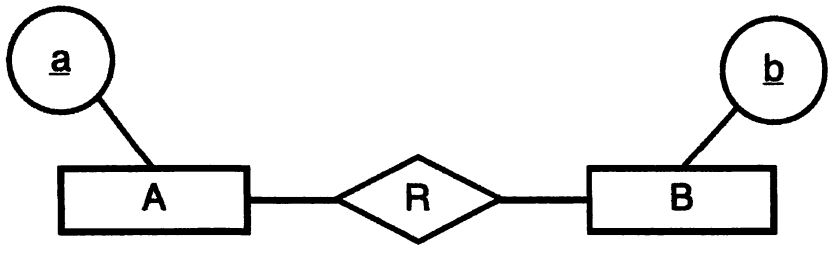
Betrachten Sie das folgende *ER*-Diagramm sowie die gegebenen Ausprägungen der Relationen A und B.

A:

a
1
2
3

B:

b
1
2
3



R_1:

a	b
1	1
1	1
2	1
2	2
3	2

R_2:

a	b
1	1
1	1
2	2
2	2

R_3:

a	b
1	1
1	1
1	1
2	1
3	2
3	2

Zeichnen Sie das angegebene *ER*-Diagramm ab und:

1. Annotieren Sie das Diagramm mit Min/Max-Angaben so, dass Ausprägung R_1 eine gültige Ausprägung der Beziehung R ist, nicht jedoch Ausprägungen R_2 bzw. R_3.

2. Fügen Sie dem Diagramm nun auch Funktionalitätsangaben hinzu, so dass die Ausprägung R_1 erlaubt ist.

Fortsetzung nächste Seite!

4. Normalformen

a) Geben Sie für die vier Relationen (R_0, R_1, R_2 und R_3) jeweils die höchste Normalform an (1. Normalform, 2. Normalform, 3. Normalform oder Boyce-Codd Normalform), welche die jeweilige Relation mit gegebenen FDs noch erfüllt.

$R_0(A,B,C,D)$

- $AB \rightarrow C$
- $C \rightarrow D$

$R_1(A,B,C,D)$

- $A,B \rightarrow C,D$
- $C \rightarrow A$

$R_2(A,B,C,D,E,F)$

Keine nicht-trivialen FDs.

$R_3(A,B,C,D,E,F)$

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

b) Gegeben sei die Relation $R(A,B,C,D,E,F,G)$ mit den FDs

1. $A \rightarrow BCDE$
2. $E \rightarrow FB$
3. $F \rightarrow A$

i) Geben Sie alle Kandidatenschlüssel an.

ii) Überführen Sie die Relation – wenn nötig mittels Synthesealgorithmus – in die 3. NF. Geben Sie ***alle Relationen in der 3. NF*** an und ***unterstreichen*** Sie in ***jeder*** einen ***Kandidatenschlüssel.***

Fortsetzung nächste Seite!

5. Schemadefinition

Gegeben ist die folgende Definition zweier Tabellen:

```
CREATE TABLE R2 (
        b integer not null,
        c integer unique,
        primary key (b)
);

CREATE TABLE R1 (
        a integer not null,
        b integer references R2,
        primary key (a)
);
```

Geben Sie jeweils an, ob das Statement syntaktisch korrekt ist und ob es von der gegebenen Datenbank ausgeführt werden kann.
Beantworten Sie jede der folgenden Fragen unabhängig von allen anderen, d.h. es liegt immer das hier gezeigte Schema vor und alle Relationen sind leer.

1. ```
   DELETE FROM R1;
   ```

2. ```
   INSERT INTO R2 VALUES (1,1);
   INSERT INTO R1 VALUES (1,1);
   INSERT INTO R1 VALUES (2,1);
   INSERT INTO R1 VALUES (3,1);
   ```

3. ```
   INSERT INTO R2 VALUES (1,1);
   INSERT INTO R2 VALUES (2,2);
   INSERT INTO R1 VALUES (1,1);
   DELETE FROM R2 WHERE b=a;
   ```

4. ```
   INSERT INTO R1 SELECT * FROM R1;
   ```

5. ```
   DROP TABLE R2 FROM DATABASE;
   ```

Fortsetzung nächste Seite!
   ```

6. Relationale Anfragen in SQL

Folgende Tabellen veranschaulichen eine Ausprägung eines Fluginformationssystems:

Flughäfen

Code	Stadt	Transferzeit (min)
LHR	London	30
LGW	London	20
JFK	New York City	60
EWR	New York City	35
MUC	München	30
FRA	Frankfurt	45
⋮	⋮	⋮

Verbindungen

ID	Von	Nach	Linie	Abflug (MEZ)	Ankunft (MEZ)
410	MUC	FRA	LH	2016-02-24 07:00:00	2016-02-24 08:10:00
411	MUC	FRA	LH	2016-02-24 08:00:00	2016-02-24 09:10:00
412	FRA	JFK	LH	2016-02-24 10:50:00	2016-02-24 19:50:00
⋮	⋮	⋮	⋮	⋮	⋮

Hinweise

- Formulieren Sie alle Abfragen in *SQL – 92* (insbesondere sind LIMIT, TOP, FETCH FIRST, ROWNUM und dergleichen *nicht* erlaubt).
- Alle Datum/Zeit-Angaben erlauben arithmetische Operationen, beispielsweise wird bei der Operation *ankunft+transferzeit* die *transferzeit* auf den Zeitstempel *ankunft* addiert.
- Es müssen *keine* Zeitverschiebungen berücksichtigt werden. Alle Zeitstempel sind in MEZ.

1. Ermitteln Sie die Städte, in denen es mehr als einen Flughafen gibt.
2. Ermitteln Sie die Städte, in denen man mit der Linie „LH“ an mind. zwei verschiedenen Flughäfen landen kann.
3. Ermitteln Sie die Flugzeit des kürzesten Direktflugs von München nach London.
4. Ermitteln Sie die kürzeste Roundtrip-Zeit (nur Direktflüge) zwischen den Flughäfen FRA und JFK (Transferzeit am Flughafen JFK beachten).

Fortsetzung nächste Seite!

Teilaufgabe 2

1. Prozessmodelle

Das V-Modell 97 und das Spiralmodell sind Prozessmodelle für die Durchführung von Softwareprojekten.

1. Nennen und erläutern Sie für jedes der beiden Prozessmodelle drei wichtige Merkmale.

2. Geben Sie je zwei Beispiele für Projekteigenschaften, bei denen zum einen das V-Modell 97 und zum anderen das Spiralmodell Vorteile hat.

2. Fragen zur Softwaretechnik

1. Erläutern Sie, was sich hinter dem Prinzip DRY – Don't Repeat Yourself – verbirgt und nennen sowie erläutern Sie zwei Arten von Dopplung.

2. Erläutern Sie jeweils an einem konkreten Beispiel die Begriffe Fehlerwirkung, Fehlerzustand und Fehlerhandlung.

3. Nennen Sie drei konkrete Software-Metriken und diskutieren Sie die Aussagekraft einer dieser Metriken.

3. Systementwicklung

Sie wurden mit der Entwicklung eines Online-Shops zur Vermarktung elektronischer Artikel beauftragt. Um Waren kaufen zu können, müssen sich die Käufer am System registrieren. Das System soll einen virtuellen Warenkorb bieten, zu dem die Käufer die gewünschten Artikel hinzufügen und anschließend den Kauf durchführen können.

1. Identifizieren Sie vier Stakeholder und nennen Sie dazu je zwei unterschiedliche Anwendungsfälle (Funktionen) des Online-Shops, in die diese involviert sind.

2. Beschreiben Sie textuell die Schritte des Szenarios beim Kauf eines DVD-Recorders. Berücksichtigen Sie dabei die Nutzeridentifikation und -authentifizierung, das Aussuchen der Waren mittels eines Warenkorbs und die Bezahlung.

3. Modellieren Sie das Szenario von 2 in Form eines Sequenzdiagramms.

4. Modellieren Sie in Form eines Zustandsübergangsdiagramms die Zustände eines Warengegenstandes im Laufe des Bestell- und Liefervorgangs beginnend vom Zustand „angeboten“ bis hin zu seiner Auslieferung. Berücksichtigen Sie auch Reklamationen durch Rückgabe und die Stornierung einer noch nicht versandten Bestellung.

Fortsetzung nächste Seite!

4. Qualitätssicherung

Die folgende Seite enthält Software-Quellcode, der einen Algorithmus zur binären Suche implementiert. Dieser ist durch Inspektion zu überprüfen. Im Folgenden sind die Regeln der Inspektion angegeben.

RM1	(Dokumentation)	Jede Quellcode-Datei beginnt mit einem Kommentar, der den Klassennamen, Versionsinformationen, Datum und Urheberrechtsangaben enthält.
RM2	(Dokumentation)	Jede Methode wird kommentiert. Der Kommentar enthält eine vollständige Beschreibung der Signatur sowie eine Design-by-Contract-Spezifikation.
RM3	(Dokumentation)	Deklarationen von Variablen werden kommentiert.
RM4	(Dokumentation)	Jede Kontrollstruktur wird kommentiert.
RM5	(Formatierung)	Zwischen einem Schlüsselwort und einer Klammer steht ein Leerzeichen.
RM6	(Formatierung)	Zwischen binären Operatoren und den Operanden stehen Leerzeichen.
RM7	(Programmierung)	Variablen werden in der Anweisung initialisiert, in der sie auch deklariert werden.
RM8	(Bezeichner)	Klassennamen werden groß geschrieben, Variablennamen klein.

1. Überprüfen Sie durch Inspektion, ob die obigen Regeln für den Quellcode eingehalten wurden. Erstellen Sie eine Liste mit allen Verletzungen der Regeln. Geben Sie für jede Verletzung einer Regel die Zeilennummer, Regelnummer und Kommentar an, z.B. (07, RM4, while nicht kommentiert). Schreiben Sie nicht in den Quellcode.

2. Entspricht die Methode 'binarySearch' ihrer Spezifikation, die durch Vor- und Nachbedingungen angeben ist? Geben Sie gegebenenfalls Korrekturen der Methode an.

3. Beschreiben alle Kommentare ab Zeile 24 die Semantik des Codes korrekt? Geben Sie zu jedem falschen Kommentar einen korrigierten Kommentar mit Zeilennummer an.

4. Geben Sie den Kontrollflussgraphen für die Methode 'binarySearch' an.

5. Geben Sie maximal drei Testfälle für die Methode 'binarySearch' an, die insgesamt eine vollständige Anweisungsüberdeckung leisten.

Fortsetzung nächste Seite!

```
00 /*
01 * BinarySearch.java
02 *
03 * Eine Implementierung der "Binaere Suche"
04 * mit einem iterativen Algorithmus
05 */
06 class BinarySearch {
07
08  /**
09   * Binaere Suche
10   * a:       Eingabefeld
11   * item:    zu suchendes Element
12   * returnValue: der Index des zu suchenden Elements oder -1
13   *
14   * Vorbedingung:
15   * a.length > 0
16   * a ist ein linear geordnetes Feld:
17   * For all k: ( 1 <= k < a.length ) ==> ( a[k-1] <= a[k] )
18   *
19   * Nachbedingung:
20   * Wenn item in a, dann gibt es ein k mit a[k] == item und returnValue == k
21   * Genau dann wenn returnValue == -1 gibt es kein k mit 0 <= k < a.length
22   * und a[k] == item.
23   */
24   public static int binarySearch(float a[], float item) {
25
26     int End; // exklusiver Index fuer das Ende des
27              // zu durchsuchenden Teils des Arrays
28
29     int start = 1; // inklusiver Index fuer den Anfang der Suche
30     End = a.length;
31
32     // Die Schleife wird verlassen, wenn keine der beiden Haelften das
33     // Element enthaelt.
34     while(start < End) {
35
36       // Teilung des Arrays in zwei Haelften
37       // untere Haelfte: [0, mid[
38       // obere Haelfte:  ]mid, End[
39       int mid = (start + End) / 2;
40
41       if (item > a[mid]) {
42         // Ausschluss der oberen Haelfte
43         start = mid + 1;
44       } else if(item < a[mid]) {
45         // Ausschluss der unteren Haelfte
46         End = mid-1;
47       } else {
48         // Das gesuchte Element wird zurueckgegeben
49         return (mid);
50       }
51     } // end of while
52
53     // Bei Misserfolg der Suche wird -1 zurueckgegeben
54     return (-1);
55   }
56 }
```

Thema Nr. 2

Teilaufgabe 1

1. *ER*-Modellierung

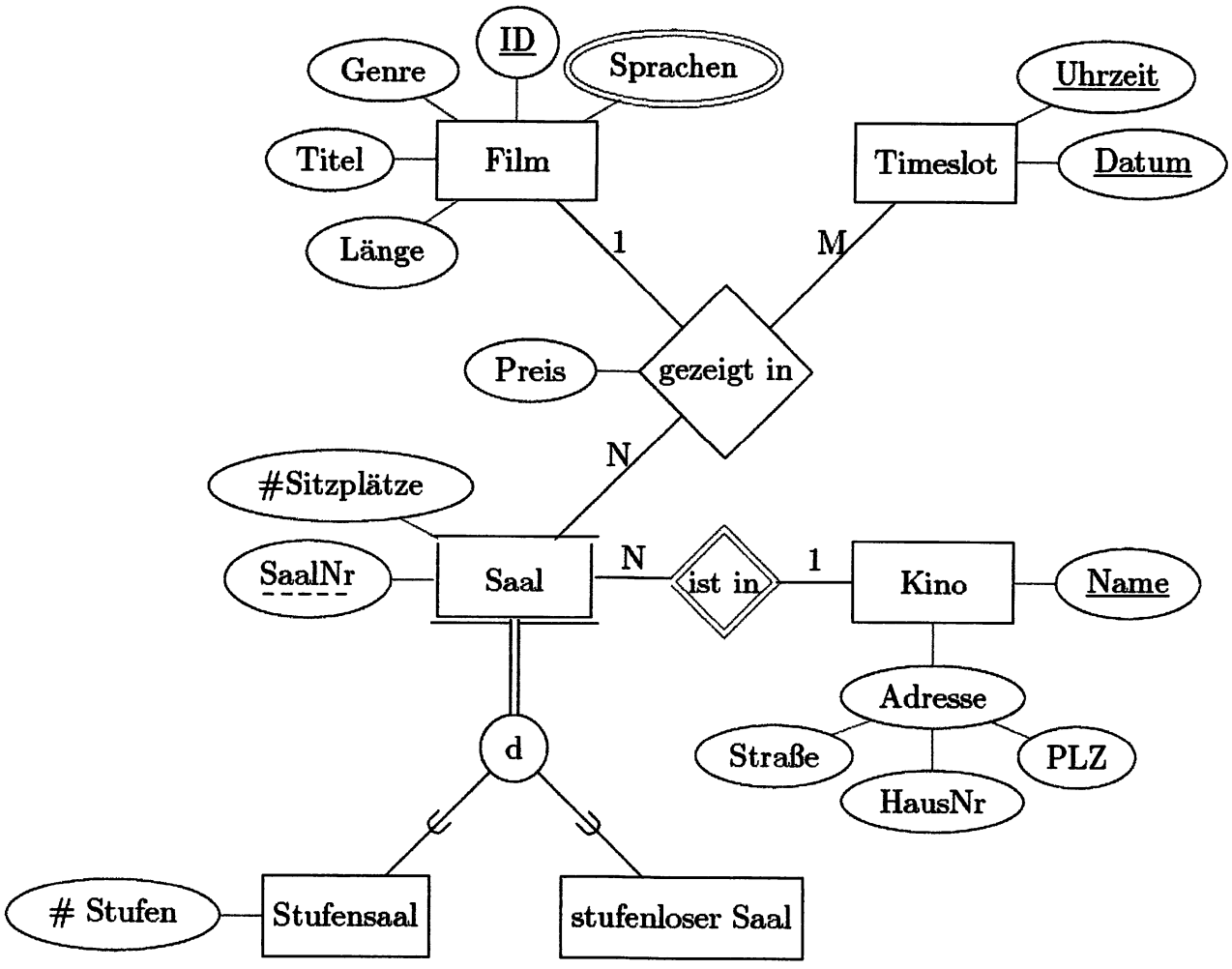
Für nachfolgend gegebene Miniwelt soll ein *ER*-Modell erstellt werden. Geben Sie Kardinalitäten in Chen-Notation an. Vergessen Sie nicht, dass zur Chen-Notation nicht nur Funktionalitäten, sondern auch Partizipitäten, also die Angabe von Existenzabhängigkeit bzw. totaler Teilnahme, zählen.

Das örtliche Tierheim möchte ein Datenbanksystem zur einfacheren Verwaltung seiner Tiere und Mitarbeiter anlegen. Hierfür speichert es zum einen alle Informationen über die beherbergten Tiere. Tiere lassen sich unterteilen in Hunde, Katzen, Vögel sowie Kleintiere. Im Tierheim bekommt jedes Tier einen Namen; dieser wird abgespeichert genau wie das Geschlecht und, soweit bekannt, das Geburtsdatum des jeweiligen Tieres. Außerdem soll festgehalten werden, ob das jeweilige Tier kastriert ist. Zu den Hunden wird ihre Rasse sowie ihre Schulterhöhe angegeben. Zu den Katzen wird ebenfalls die Rasse angegeben. Bei den Kleintieren wird angegeben, um was für ein Tier es sich handelt, z. B. Maus, Hamster, Kaninchen. Ab und zu kann es vorkommen, dass auch noch andere Tiere im Tierheim aufgenommen werden.
Jedes Tier bekommt eine bestimmte Zusammenstellung von mehreren Futtersorten. Jede Futtersorte hat einen Namen, einen Hersteller sowie eine Beschreibung. Für jedes Tier wird die Menge der jeweiligen Sorte festgehalten.
Die Mitarbeiter des Tierheims, von denen Name, Geburtsdatum und Anschrift, bestehend aus Straße, Hausnummer, Postleitzahl und Wohnort, gespeichert werden, lassen sich in feste Mitarbeiter mit Bezahlung sowie ehrenamtliche Mitarbeiter ohne Bezahlung unterteilen. Bei den festen Mitarbeitern wird die Höhe der Bezahlung in der Datenbank gespeichert. Für jeden ehrenamtlichen Mitarbeiter wird mindestens ein Tier symbolisch als „Patentier" eingetragen. Sind zu wenig ehrenamtliche Mitarbeiter verfügbar, kann es vorkommen, dass nicht jedes Tier einen Paten bekommt. Die festen Mitarbeiter kümmern sich aber natürlich um alle Tiere.

Fortsetzung nächste Seite!

2. Relationenmodell

Überführen Sie das folgende *ER* -Modell in ein Relationenschema, das die Bedingungen der 3. Normalform erfüllt. Geben Sie bei jedem Fremdschlüssel an, worauf er verweist. Überlegen Sie sich bei der Transformation von Generalisierung bzw. Spezialisierung, welche der zur Verfügung stehenden Möglichkeiten die beste ist. Beantworten Sie anschließend noch kurz die Frage, warum Sie sich für die von Ihnen gewählte Methode entschieden haben bzw. was gegen die anderen Möglichkeiten spricht.



Fortsetzung nächste Seite!

3. Normalisierung

Gegeben ist eine Relation R in 1. Normalform mit den Attributen {A, B, C, D, E, F, G, H}. In diesem Schema gelten die folgenden funktionalen Abhängigkeiten:

$AB \rightarrow D$

$ABC \rightarrow E$

$B \rightarrow C$

$FG \rightarrow HA$

$GA \rightarrow C$

3.1 Zweite Normalform

In einem ersten Schritt wurde diese Relation in zwei Relationen gespalten:

Relation1 (A, B, C, D, E)

Relation2 (F, G, H, A, C)

Überprüfen Sie für das gegebene Relationenschema aus den zwei Relationen, ob es in 2. Normalform ist. Begründen Sie sämtliche Ihrer Schlussfolgerungen und Entscheidungen.

Hinweis: Wie gehen Sie vor, wenn Sie eine Menge an Relationen gegeben haben? Was sagen die erfüllten Normalformen der einzelnen Relationen über die Normalform des Schemas aus?

3.2 Synthesealgorithmus

Wenden Sie den Synthesealgorithmus an, um das Schema in ein Schema in 3. Normalform zu überführen. Denken Sie daran, welche Voraussetzung dabei für die Menge der gegebenen funktionalen Abhängigkeiten gelten muss. Führen Sie auch die Schritte des Algorithmus kurz an, die keine Auswirkungen auf das Ergebnis haben, aber grundsätzlich zu beachten sind.

Fortsetzung nächste Seite!

4. SQL und Relationale Algebra

Gegeben ist folgendes Relationenschema für die Angebotsauswahl eines Wirtshauses:

Produkt(PNr, Name)
Getränk(PNr, Verkaufsvolumen, VPreis, EPreis, Alkoholgehalt)
FK(PNr) referenziert Produkt(PNr)
Speise(PNr, Beschreibung, VPreisP, EPreisP)
FK(PNr) referenziert Produkt (PNr)
Zusatzstoff(ZID, Name)
enthaelt(PNr,ZID)
FK(PNr) referenziert Produkt(PNr)
FK(ZID) referenziert Zusatzstoff(ZID)

VPreis und EPreis bezeichnen den Verkaufs- bzw. Einkaufspreis für 100 ml des jeweiligen Getränks. VPreisP und EPreisP bezeichnen den Verkaufs- bzw. Einkaufspreis für eine Portion der jeweiligen Speise.
Verwenden Sie für die folgenden Abfragen stets nur Standard-*SQL*-Befehle.

4.1
Wie viele verschiedene Speisen werden in dem Wirtshaus angeboten? Schreiben Sie eine *SQL*-Abfrage, welche das Ergebnis in einer Spalte mit dem Namen „Anzahl-Speisen" ausgibt.

4.2
Mit welchem Getränk bzw. welchen Getränken macht der Wirtshausbetreiber den größten Gewinn (auf den Preis bezogen)? Schreiben Sie eine *SQL*-Abfrage, welche die Nummer und den Namen des gefragten Getränks bzw. der gefragten Getränke ausgibt.

4.3
Welche angebotenen Produkte enthalten Zusatzstoffe mit dem Namensteil „phosphat"?
Schreiben Sie eine *SQL*-Abfrage, welche die Nummer und den Namen der Produkte ausgibt.

4.4
Welches sind die fünf teuersten Speisen im Angebot des Wirtshauses? Der *LIMIT*-Operator ist für diese Aufgabe nicht zulässig, da die Ausgabe sportlich fair sein soll. Sportlich fair bedeutet, dass falls beispielsweise ein Platz doppelt belegt ist, der nachfolgende Platz ausgelassen wird. Es können somit auch mehr als fünf Datensätze ausgegeben werden. Schreiben Sie eine *SQL*-Abfrage, welche die Nummer, den Namen, den Preis und die Platzierung aufsteigend, beim teuersten Gericht mit 1 beginnend, ausgibt.

4.5
Nennen Sie drei verschiedene Möglichkeiten, wie Löschregeln (*ON DELETE ...*) beim Anlegen von Tabellen definiert sein können. Wie lautet der jeweilige *SQL*-Befehl und welche Auswirkung hat er, wenn ein Eintrag einer referenzierten Tabelle gelöscht wird?

4.6
Formulieren Sie die Abfrage von Aufgabe 4.3 in relationaler Algebra.

Fortsetzung nächste Seite!

5. Grundlagen

5.1
Welche zwei Alternativen gibt es grundsätzlich, eine Datenbankanfrage zu optimieren? Nennen Sie diese beiden Arten der Optimierung und erklären Sie in ein bis zwei Sätzen, unter Zuhilfenahme welcher Methoden jeweils optimiert wird. Geben Sie für beide Alternativen je ein Beispiel an.

5.2
Für die Einhaltung der referentiellen Integrität gibt es in *SQL* für jeden der drei Begriffe Schlüsselkandidat, Primärschlüssel und Fremdschlüssel mindestens eine Beschreibungsmöglichkeit. Wie lautet diese jeweils bzw. falls es mehrere Möglichkeiten gibt, wie lauten diese?

5.3
Mit welchen zwei Möglichkeiten kann eine Transaktion abgeschlossen werden? Nennen Sie diese beiden Begriffe und die Auswirkungen auf die Datenbank.

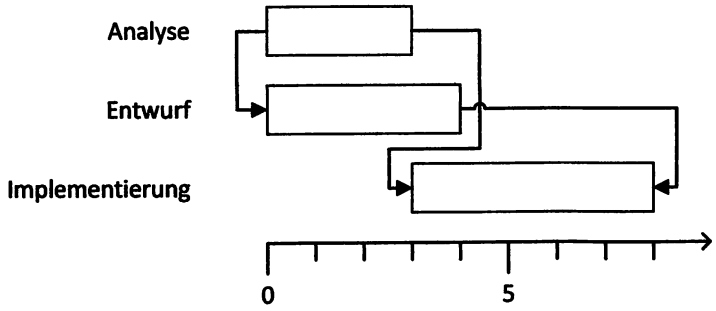
5.4
Erklären Sie anhand des *ANSI – SPRARC* – Schemas den Begriff der Datenunabhängigkeit in zwei bis drei Sätzen.

Fortsetzung nächste Seite!

Teilaufgabe 2

1. Projektmanagement

Gegeben ist das folgende **Gantt-Diagramm** zur Planung eines hypothetischen Softwareprojekts:



a) Konvertieren Sie das Gantt-Diagramm in ein **CPM-Netzwerk**, das die Aktivitäten und Abhängigkeiten äquivalent beschreibt. Gehen Sie von der Zeiteinheit „Monate" aus. Definieren Sie im CPM-Netzwerk je einen globalen Start- und Endknoten. Der Start jeder Aktivität hängt dabei vom Projektstart ab, das Projektende hängt vom Ende aller Aktivitäten ab.

b) Berechnen Sie für jedes Ereignis (d.h. für jeden Knoten Ihres CPM-Netzwerks) die **früheste Zeit**, die **späteste Zeit** sowie die **Pufferzeit**. Beachten Sie, dass die Berechnungsreihenfolge einer topologischen Sortierung des Netzwerks entsprechen sollte.

c) Geben Sie einen **kritischen Pfad** durch das CPM-Netzwerk an.
Welche **Aktivität** darf sich demnach wie lange verzögern?

Fortsetzung nächste Seite!

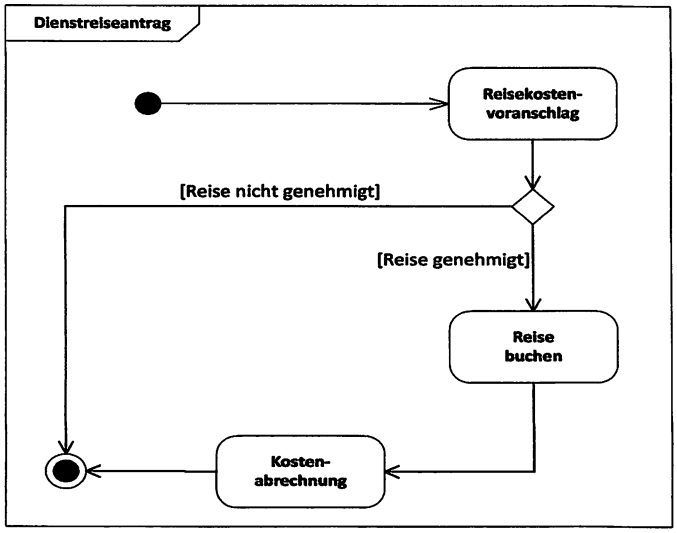
2. Objektorientierte Modellierung mit UML

Gegeben sei das folgende **Glossar**, welches die statische Struktur von einfachen **Aktivitätsdiagrammen** in natürlicher Sprache beschreibt:

> ***Aktivitätsdiagramm:*** *Benannter Container für Aktivitäten und Datenflüsse. Eine der definierten Aktivitäten ist als Start-Aktivität ausgezeichnet.*
>
> ***Aktivität:*** *Teil des beschriebenen Verhaltens. Man unterscheidet Start-, End-, echte Aktivitäten sowie Entscheidungen. Aktivitäten können generell mehrere ein- und auslaufende Kontrollflüsse haben.*
>
> ***Startaktivität****: Ist im Aktivitätsdiagramm eindeutig und dient als Einstiegspunkt des beschriebenen Ablaufs.*
>
> ***Endaktivität****: Wird eine solche Aktivität erreicht, ist der beschriebene Ablauf zu Ende.*
>
> ***Echte Aktivität****: Benannte Aktion, die nach Ausführung zu einer definierten nächsten Aktivität führt.*
>
> ***Entscheidung****: Aktivität, die mehrere Nachfolger hat. Welche davon als nächstes ausgeführt wird, wird durch entsprechende Bedingungen (s. Kontrollfluss) gesteuert.*
>
> ***Kontrollfluss:*** *Verbindet je eine Quell- mit einer Zielaktivität. Kann eine Bedingung enthalten, die erfüllt sein muss, damit die Zielaktivität im Falle einer Entscheidung ausgeführt wird.*

a) Geben Sie ein **UML-Klassendiagramm** an, welches die im Glossar definierten Konzepte und Beziehungen formal beschreibt. Geben Sie bei allen Attributen und Assoziationsenden deren Sichtbarkeit, Multiplizität und Typ an. Benennen Sie alle Assoziationen.

b) Nachfolgend ist ein Beispiel eines Aktivitätsdiagramms in der gängigen grafischen Notation abgebildet. Stellen Sie den beschriebenen Kontrollfluss als **UML-Objektdiagramm** *konform zum in Teilaufgabe a erstellten UML-Klassendiagramm* dar. Referenzieren Sie die dort definierten Klassen und Assoziationen; auf Objektbezeichner dürfen Sie verzichten.



Fortsetzung nächste Seite!

3. Entwurfsmuster

Verwenden Sie geeignete **Entwurfsmuster**, um die folgenden Sachverhalte mit Hilfe von **UML-Klassendiagrammen** zu beschreiben. Nennen Sie das zu verwendende Entwurfsmuster namentlich, wenden Sie es zur Lösung der jeweiligen *Fragestellung* an und erstellen Sie damit das problemspezifische UML-Klassendiagramm. Beschränken Sie sich dabei auf die statische Sicht, d.h. definieren Sie keinerlei Verhalten mit Ausnahme der Definition geeigneter Operationen.

a) Ein Unternehmen ist *baumartig* strukturiert. Abteilungen können wiederum selbst Unterabteilungen und/oder Personen beinhalten. Eine Person/Abteilung kann dabei nicht mehreren Abteilungen untergeordnet sein. Jede Abteilung wird außerdem von einer Person geleitet; umgekehrt können Personen beliebig viele Abteilungen leiten. Personen wie Abteilungen haben einen zahlenwertigen Identifikationsschlüssel. Bei Abteilungen wird eine Bezeichnung, bei Personen Name und Geburtsdatum hinterlegt.

b) Es gibt unterschiedliche Arten universitärer Veranstaltungen: Vorlesungen, Übungen und Seminare. Alle Veranstaltungen haben einen Namen und eine ID (Beispiel: „INF301"). Übungen beziehen sich immer auf eine Vorlesung. Seminare können wöchentlich, zweiwöchentlich oder im Block stattfinden. Definieren Sie eine Klasse „Hochschule", die die unterschiedlichen Veranstaltungen zusammenfasst und verwaltet. Es soll sichergestellt werden, dass es von dieser Klasse *nur eine einzige Instanz* gibt.

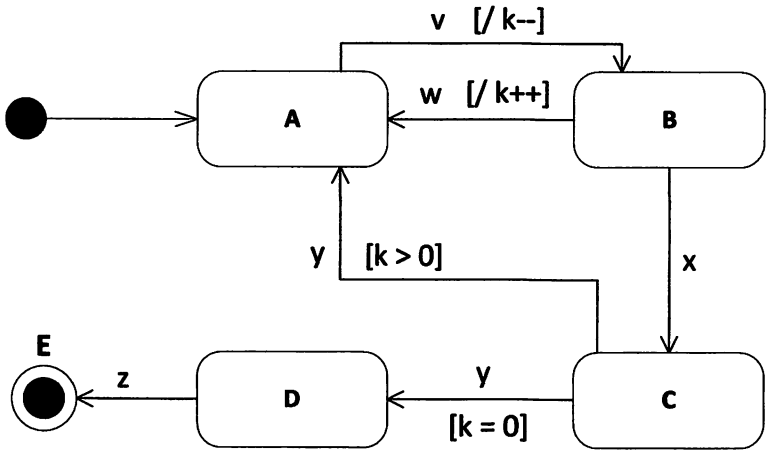
c) Eine Autowaschanlage besteht aus mehreren Reinigungsbürsten, vier Felgenreinigern und einem Trockengebläse. Alle Komponenten können unabhängig voneinander ein- und ausgeschaltet werden. Die Waschanlage bietet eine Operation an, die den Waschvorgang durchführt indem sie die einzelnen Komponenten in geeigneter Reihenfolge ein- und ausschaltet. Die Waschanlage bietet somit eine *vereinheitlichte Schnittstelle* für die Untersysteme (Bürsten, Felgenreiniger, ...) an.

Fortsetzung nächste Seite!

4. Zustandsbasiertes Testen

a) Geben Sie den allgemeinen Unterschied zwischen **Black-** und **White-Box-Tests** an. Zu welcher Kategorie gehört das **Zustandsbasierte Testen**?

Gegeben ist folgendes **UML-Zustandsdiagramm**. Die Variable *k* sei von ganzzahligem Typ.



b) Geben Sie **einen möglichst kurzen Testfall** an, der das Kriterium der **Zustandsüberdeckung** erfüllt.
Stellen Sie den Testfall als eine Sequenz von Operationsaufrufen dar. Notieren Sie bei jedem Zustandsübergang Vorzustand, Transition, Nachzustand, sowie den Wert der Variablen *k* vor und nach dem Operationsaufruf. Der Nachzustand bzw. Wert „k nachher" muss dem Vorzustand bzw. „k vorher" des nächsten Operationsaufrufs entsprechen. Den Initialwert von *k*, also das „k vorher" der ersten Zeile, sollen Sie dabei selbst bestimmen.

Beispiel:

Vorzustand	*k vorher*	*Transition*	*Nachzustand*	*k Nachher*
A	*3*	*v*	*B*	*2*
B	*2*	*w*	*A*	*3*

c) Geben Sie nun **einen möglichst kurzen Testfall** an, der das Kriterium der **Transitions-überdeckung** erfüllt.
Verwenden Sie für den Testfall eine Darstellung mit denselben Eigenschaften wie in Teil-aufgabe b.