

Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer
--------------------	----------------	----------------------

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

**Frühjahr
2019**

46116

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

— Prüfungsaufgaben —

Fach: **Informatik (Unterrichtsfach)**

Einzelprüfung: **Softwaretechnologie/Datenbanksystme**

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

Anzahl der Druckseiten dieser Vorlage: **11**

Bitte wenden!

Thema Nr. 1

Teilaufgabe I: Softwaretechnik

Aufgabe 1 (Anforderungserhebung)

Es soll eine Webanwendung zur automatischen Überwachung und zum Handel von Aktien entwickelt werden. Sie sind für die Anforderungserhebung zuständig.

- a) Erklären Sie, was an der folgenden Anforderung nicht optimal ist:
„Die Aktien sollen in einer relationalen Datenbank gespeichert werden.“
Gehen Sie dabei davon aus, dass das Domänenkonzept Aktien an anderer Stelle schon definiert ist.

- b) Kategorisieren Sie die Anforderungen in funktional oder nicht-funktional. Kreuzen Sie die richtige Antwort an.

Anforderung	funktional	Nicht-funktional
Der Kunde soll Aktien zu einer Liste von überwachten Aktien hinzufügen können.		
Die Antwortzeit aller Operationen muss unter 1 Sekunde liegen.		
Das System verkauft Aktien und informiert den Nutzer, sobald gewisse Aktienkurse einen vordefinierten Schwellenwert über- oder unterschreiten.		
Die Software soll barrierefrei sein und dies durch eine «Access for all» Zertifizierung nachweisen.		
Die Antwortzeit darf nicht schneller als 1 Millisekunde sein, da dies in Deutschland sonst unter den erlaubnispflichtigen Hochfrequenzhandel fallen könnte.		
Die Software muss den Regeln des deutschen Börsengesetzes folgen.		

Hintergrundinformationen für die Aufgaben 2-5: Modellierung und Implementierung eines Programms

Ein autonomes Auto hat einen Namen und fährt mit einer bestimmten Geschwindigkeit (gemessen in Meter/Sekunde). Es bekommt von verschiedenen Sensoren (z. B. Kameras) jede Zehntelsekunde neue Informationen über seine Umwelt. Wir betrachten hier nur die Situation, wenn es gerade eine neue Information bekommen hat. Ein Sensor hat einen Namen und eine Aussage über seine Funktionsfähigkeit (ausgeschaltet, schlechte Funktionsfähigkeit, gute Funktionsfähigkeit). Außerdem liefert er zwei Aussagen, ob eine rote Ampel bzw. ob eine gelbe Ampel gefunden wurde (jeweils als Boolescher Wert). Weiterhin wird der Abstand zur Ampel zurückgegeben: als Zahl in Metern.

Wenn mindestens ein Sensor mit guter Funktionsfähigkeit eine rote Ampel meldet, soll das Fahrzeug bremsen. Wenn das nicht der Fall ist und mindestens ein Sensor mit guter Funktionsfähigkeit eine gelbe Ampel findet, soll das Fahrzeug abhängig von seiner Position und Geschwindigkeit entweder bremsen oder normal weiterfahren. Die Regel ist: Es soll bremsen, wenn gilt:
(Geschwindigkeit in m/s * 2 Sekunden) < Abstand_zur_Ampel (in Meter).

Im Folgenden soll ein Klassendiagramm für das Auto und die Sensoren beschrieben werden. Das Auto soll eine Methode "bremsen" haben, die abhängig vom Zustand des Autos und der Sensoren einen Booleschen Wert "wahr" (es soll bremsen) oder "falsch" (es soll weiterfahren) zurückgibt.

Aufgabe 2 (UML-Modellierung: Klassendiagramm)

Geben Sie ein minimales UML-Klassendiagramm an, das alle angegebenen Informationen enthält.
Hinweis: Bei den Aufgaben 4 und 5 wird Konsistenz des Aktivitätsdiagramms bzw. des Codes mit dem Klassendiagramm verlangt.

Aufgabe 3 (Pseudocode und Aktivitätsdiagramm)

- Geben Sie Pseudo-Code für die Methode „bremsen“ an. Benutzen Sie dazu eine separat definierte Hilfsmethode für die Entscheidung, ob das Auto bei gelber Ampel bremsen soll. Achten Sie auf die Konsistenz zum Klassendiagramm (Inkonsistenzen führen zu Punktabzügen).
- Geben Sie zu dem Pseudo-Code aus Aufgabe 3 a zwei syntaktisch korrekte UML-Aktivitätsdiagramme an. Schleifen und Verzweigungen müssen durch strukturierte Knoten dargestellt werden (d. h. kein graphisches „goto“).

Aufgabe 4 (oo-Programmierung)

Implementieren Sie die Methode „bremsen“ der Klasse Auto in einer objektorientierten Programmiersprache (z. B. Java oder andere mit Nennung). Sie sollen nur den Code für die Methoden angeben. Sie brauchen keinen Code für Klassendefinitionen angeben, sondern können sich auf das UML-Klassendiagramm aus Aufgabe 2 beziehen.

Aufgabe 5: (JUnit Test)

Beim Programmieren ist es häufig nützlich, vor der Implementierung einige Tests mit Eingabe-Ausgabe-Spezifikationen zu schreiben, was die Methode leisten soll. Implementieren Sie dazu einen Test (z. B. in Java (JUnit) oder einer anderen Programmiersprache) für das korrekte Verhalten der Methode „bremsen“, der automatisch ausgeführt werden kann.

Teilaufgabe II: Datenbanken

Aufgabe 1: ER-Modellierung

Hinweis: Bei Wahl dieser Aufgabe wird Wissen über das erweiterte Entity-Relationship-Modell (beispielsweise schwache Entity-Typen, Vererbung) sowie die Verfeinerung eines relationalen Schemas vorausgesetzt.

Gegeben seien folgende Informationen:

- Universitäten bieten Fachrichtungen an, deren Name nur innerhalb der Universität eindeutig ist. Eine Universität hat neben einem eindeutigen Namen auch eine Adresse und ein Gründungsjahr.
 - Fachrichtungen wiederum bestehen aus mehreren Lehrstühlen, deren Name ebenfalls nur innerhalb der Fachrichtung eindeutig ist.
 - An jedem Lehrstuhl arbeitet genau ein Professor.
 - Universitäten bezahlen Angestellte. Diese werden über eine PersID identifiziert und besitzen zudem einen Namen. Angestellte werden unter anderem unterschieden in Professor, Hilfswissenschaftler (Hiwi) und wissenschaftliche Mitarbeiter, wobei ein Angestellter immer nur eines davon sein kann.
 - Für einen Professor wird zudem die Anzahl seiner Veröffentlichungen gespeichert, für einen Hiwi wird die Anzahl an Semestern gespeichert. Jeder wissenschaftliche Mitarbeiter wird von genau einem Professor betreut. Jeder Hiwi unterstützt wissenschaftliche Mitarbeiter.
 - Eine Universität kann zusammen mit einem Lehrstuhl einen Antrag stellen, bei dem genau ein Professor beteiligt ist.
- a) Erstellen Sie für das oben gegebene Szenario ein geeignetes ER-Diagramm. Verwenden Sie dabei – wenn angebracht – das Prinzip der Spezialisierung. Kennzeichnen Sie die Primärschlüssel der Entity-Typen, totale Teilnahmen und schwache Entity-Typen. Zeichnen Sie die Funktionalitäten der Relationship-Typen in das Diagramm ein.
- b) Überführen Sie Ihr in Aufgabe a) erstelltes Modell in ein verfeinertes relationales Schema. Kennzeichnen Sie die Schlüssel durch Unterstreichen. Datentypen müssen nicht angegeben werden. Die einzelnen Schritte müssen angegeben werden.

Aufgabe 2: (SQL)

Gegeben sei der folgende Ausschnitt des Schemas eines Schulverwaltungssystems:

```

Schueler : {[  
    ID : INTEGER,  
    Name : VARCHAR(255),  
    Wohnort : VARCHAR(255)  
}  
  

Zaubererschule : {[  
    Name : VARCHAR(255),  
    Standort : VARCHAR(255)  
}  
  

Unterrichtsfach : {[  
    Name : VARCHAR(100),  
    Jahr : INTEGER,  
    Lehrer : VARCHAR(255),  
    AnzahlSchueler : INTEGER,  
    wird_angeboten_von : VARCHAR(255)  
}
  
```

○

```

besucht : {[  
    Schueler : INTEGER,  
    Schulname : VARCHAR(255),  
    von : DATE,  
    bis : DATE  
}  
  

setzt_voraus : {[  
    GrundlageName : VARCHAR(100),  
    GrundlageJahr : INTEGER,  
    FortgeschrittenName : VARCHAR(100),  
    FortgeschrittenJahr : INTEGER,  
    Mindestnote : {1, 2, 3, 4, 5}  
}
  
```

○

Die Tabelle *Schueler* enthält Informationen über Schüler. Die Tabelle *Zaubererschule* beschreibt Zaubererschulen anhand ihres Namens und des Standortes. Die Tabelle *besucht* gibt an, welcher Schüler zu welchem Zeitpunkt welche Zaubererschule besucht hat. Die Tabelle *Unterrichtsfach* enthält Informationen über die Unterrichtsfächer bezüglich deren Name und Jahr, den Namen des Lehrers, die Anzahl der teilnehmenden Schüler und den Namen der Schule als Referenz auf Zaubererschule, die dieses Unterrichtsfach anbietet. Die Tabelle *setzt_voraus* beinhaltet Informationen darüber, welche Unterrichtsfächer welche anderen Unterrichtsfächer mit welcher Mindestnote voraussetzen, jeweils mit Referenz auf Unterrichtsfach.

Beachten Sie bei der Formulierung der SQL-Anweisungen, dass die Ergebnisrelationen keine Duplikate enthalten dürfen. Sie dürfen geeignete Views definieren.

- Schreiben Sie eine SQL-Anweisung, welche die Tabelle Unterrichtsfach mit allen Constraints (Primärschlüssel-, Fremdschlüssel- und Check-Constraints) anlegt. Stellen Sie dabei sicher, dass die Obergrenze für die Anzahl der Schüler bei 20 liegt.
- Schreiben Sie eine SQL-Anweisung, die für jede Zaubererschule den Namen des Unterrichtsfaches zurückgibt, an dem die meisten Schüler teilnehmen (unabhängig von Jahr).
- Schreiben Sie eine SQL-Anweisung, welche die ID und den Namen aller Schüler bestimmt, die an Zaubererschulen waren, deren Standort an ihrem Wohnort ist.
- Schreiben Sie eine SQL-Anweisung, die die Namen aller Unterrichtsfächer zurückgibt, welche mehr als ein anderes Unterrichtsfach mit einer Mindestnote von „2“ voraussetzen.

Fortsetzung nächste Seite!

- e) Schreiben Sie eine SQL-Anweisung, die die ID und den Namen folgender Schüler ermittelt:
Entweder besucht dieser Schüler keine Schule oder an allen von ihm besuchten Schulen wird das Unterrichtsfach mit dem Namen „Zaubertränke“ (unabhängig von Jahr) nicht angeboten.

Aufgabe 3: (Entwurfstheorie)

Gegeben sei folgendes relationales Schema R in erster Normalform:

$$R : \{[A, B, C, D]\}$$

und die Zerlegung $\rho = \{R_1, R_2\}$ von R mit $R_1 = \{C, D\}$ und $R_2 = \{A, B, D\}$.

Für R gelte folgende Menge FD funktionaler Abhängigkeiten:

$$FD = \{$$

$$\begin{aligned} BC &\rightarrow AD; \\ B &\rightarrow A; \\ D &\rightarrow C \end{aligned}$$

$$\}$$

- a) Bestimmen Sie alle Kandidatenschlüssel/Schlüsselkandidaten von R mit FD .

Hinweis: Die Angabe von Attributmengen, die keine Kandidatenschlüssel sind, führt zu Abzügen.

- b) Prüfen Sie, ob R mit FD in 2NF und/oder 3NF ist.

- c) Zeigen oder widerlegen Sie, dass die Zerlegung ρ von R abhängigkeitserhaltend bzgl. FD ist.

- d) Zeigen oder widerlegen Sie, dass die Zerlegung ρ von R verlustfrei bzgl. FD ist.

- e) Bestimmen Sie eine kanonische Überdeckung FD_C von FD .

i. Führen Sie eine Linksreduktion von FD durch. Geben Sie die Menge funktionaler Abhängigkeiten nach der Linksreduktion an (FD_L).

ii. Führen Sie eine Rechtsreduktion des Ergebnisses der Linksreduktion (FD_L) durch. Geben Sie die Menge funktionaler Abhängigkeiten nach der Rechtsreduktion an (FD_R).

iii. Bestimmen Sie eine kanonische Überdeckung FD_C von FD auf Basis des Ergebnisses der Rechtsreduktion (FD_R).

- f) Zerlegen Sie R mit FD_C mithilfe des Synthesealgorithmus in 3NF. Geben Sie zudem alle funktionalen Abhängigkeiten der erzeugten Relationenschemata an.

Thema Nr. 2

Teilaufgabe I:

Aufgabe 1 (Programmierrichtlinien)

Entscheiden Sie jeweils, ob folgende Aussagen zutreffend sind oder nicht. Begründen Sie jeweils Ihre Antwort.

- a) Lesbarkeit

Programmierrichtlinien sind ein Mittel, die Lesbarkeit von Quelltext zu erhöhen.

- b) Vollständigkeit

Formatieren zwei Programmierer den gleichen Quelltext nach denselben Richtlinien, wird das Ergebnis immer identisch sein.

- c) Objektivität

Für eine gegebene Programmiersprache gibt es objektiv betrachtet genau eine beste Programmierrichtlinie.

- d) Fehlervermeidung

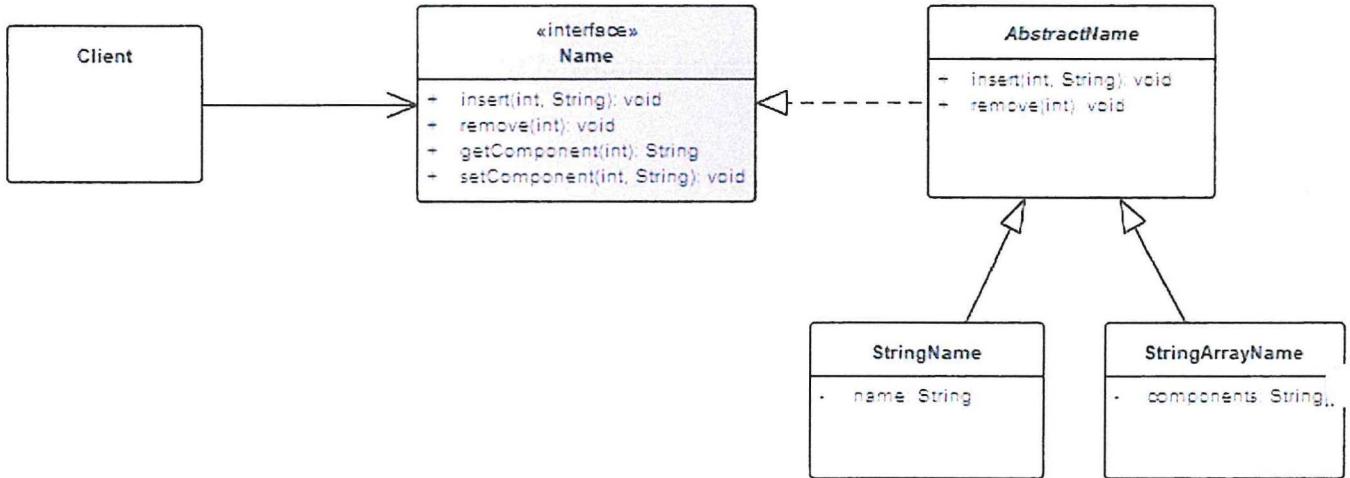
Einheitlich nach einer Programmierrichtlinie formatierter Quelltext hilft, Programmierfehler zu vermeiden.

- e) Teamarbeit

Ein Programmierer, der Fehler in den Programmierrichtlinien eines Projekts findet, sollte diese allein beheben und umsetzen.

Aufgabe 2: (Programmierung)

Das im Folgenden dargestellte Diagramm ist das von Aufgaben 2a) - e) referenzierte Diagramm.



Das im Diagramm dargestellte Klassenmodell stellt einen einfachen Entwurf zur objektorientierten Modellierung von homogenen Namensobjekten dar. Ein homogener Name ist ein Bezeichner, der aus mehreren gleichartigen Komponenten besteht. Eine Komponente hat den Basistyp String. Beispiele für homogene Namen sind Domänennamen, z.B. "www.google.com" oder Java-Paket- und Klassennamen, z.B. "java.lang.Object". Es gilt Folgendes für den Entwurf:

- Der Entwurf ist nicht vollständig; es fehlen Operationen.
- Trennzeichen, z. B. '.', sind nicht modelliert worden; nehmen Sie an, dass fix der Punkt '.' verwendet wird.
- Die Klasse AbstractName definiert keine Felder, welche den Namen im Speicher repräsentieren. Nur die Unterklassen machen dies.

a) Schnittstellen

Erläutern Sie den technischen (nicht fachlichen) Zweck der Schnittstelle Name.

b) Abstrakte Oberklasse

Erläutern Sie den technischen (nicht fachlichen) Zweck der abstrakten Oberklasse AbstractName.

c) Instanziierbarkeit

Muss die Klasse StringName die Methode getComponent(int) implementieren, um instanzierbar zu sein? Begründen Sie Ihre Antwort.

d) Speicher

Welche der beiden Unterklassen `StringName` und `StringArrayName` verbraucht bei gleichem Inhalt mehr Speicher? Begründen Sie Ihre Antwort.

e) Zugriff

Was ist die Komplexität des Zugriffs auf eine beliebige Namenskomponente jeweils für `StringName` und `StringArrayName`? Verwenden Sie die O-Notation und begründen Sie Ihre Antwort.

Aufgabe 3 (Modellierung)

Ein Code-Review-System ist ein Werkzeug zur Unterstützung von Code-Reviews in Softwareentwicklungsprozessen.

Gegeben ist die folgende Beschreibung eines Teils eines solchen Systems:

“Nutzer des Systems können Autoren oder Reviewer sein. Autoren reichen veränderte Dateien ein, Reviewer begutachten die Änderungen. Zu jeder Datei soll deren Name mitgeführt werden. Jede Änderung an einer Datei hat ein Datum (Datentyp Date) und muss zuvor einen Code-Review durchlaufen, welcher von einem Reviewer ausgeführt wird. Ein Autor darf seine eigenen Änderungen nicht selbst begutachten. Zu jedem Review soll der Erstellungszeitpunkt des Reviews mitgeführt werden. Das System wählt automatisch eine bestimmte Anzahl an Reviewern aus, die ihr Reviewergebnis (Enumeration Accepted, Rejected) mit einem Zeitstempel (Datentyp Date) abgeben.”

Erstellen Sie ein UML-Klassendiagramm, welches die gegebene Beschreibung des Code-Review-Systems getreu wiedergibt.

Aufgabe 4 (Modellierung)

Gegeben ist die folgende Beschreibung eines Softwaresystems zur Bibliotheksverwaltung:

“Nutzende des Systems sind Studierende und Bibliothekare oder Bibliothekarinnen. Nutzende können das Inventar der Bibliothek über eine Suchmaske in einer Web-basierten Benutzungsschnittstelle durchsuchen. Das Suchergebnis zeigt die gefundenen Bücher sowie deren Verfügbarkeit an. Ein Nutzer oder eine Nutzerin kann ein Buch zur Ausleihe vormerken, sofern dieses verfügbar ist und er oder sie über ein Nutzerkonto verfügt. Beim Ausleihvorgang wird das Nutzerkonto überprüft und die Ausleihe von einem Bibliothekar oder einer Bibliothekarin autorisiert, welche das Buch aushändigt. Bei der Rückgabe des Buches wird ebenfalls das Nutzerkonto geprüft; wurde das Buch zu spät zurückgegeben, wird eine Strafgebühr erhoben.”

Modellieren Sie die beschriebenen Anwendungsfälle in einem UML-Anwendungsfalldiagramm.

Teilaufgabe II: Datenbanken

Aufgabe 1: ER-Modellierung

Wir betrachten eine Datenbank, die einem Webshop zur Verwaltung seiner Daten dient. Sie verfügt über die folgenden Eigenschaften:

Für jedes Produkt kann der Name sowie das Gewicht erfasst werden. Jedes Produkt hat eine eindeutige EAN-Nummer und einen Nettopreis.

Es gibt verschiedene Steuerklassen. Für diese wird jeweils ein Bezeichner erfasst (z. B. *reduzierte MwSt.*) sowie die Steuerhöhe (z. B. *0.07*) und eine eindeutige Nummer. Jedes Produkt ist genau einer Steuerklasse zugeordnet.

Produkte können als passend zu anderen deklariert werden. Bspw. kann zu einem *Notebook* ein passendes *Netzteil* und ein passender *Akku* hinterlegt werden.

Ein Kunde ist charakterisiert durch seinen Vornamen, Nachnamen, die Postadresse sowie das Geburtsjahr. Jeder Kunde hat eine eindeutige Kundennummer.

Für jede Bestellung wird das Datum abgespeichert. Jede Bestellung ist genau einem Kunden zugeordnet. Die Bestellung enthält eine Bestellnummer, die zusammen mit der Kundennummer eindeutig ist. Eine Bestellung besteht aus beliebig vielen Artikeln, zu denen jeweils die Artikelanzahl abgespeichert wird. So können in einer Bestellung z. B. gleich zwei *Netzteile* gekauft werden.

Mögliche Dateneinträge sind *kursiv* hervorgehoben und dienen einzig der Veranschaulichung.

- a) Modellieren Sie das oben genannte Datenbanksystem möglichst vollständig in einem erweiterten ER-Diagramm. Kennzeichnen Sie die Schlüssel der Entity-Typen und geben Sie für die Relationship-Typen jeweils Funktionalitätsbedingungen (in N:M-Notation) oder Kardinalitätsbedingungen (in (MIN, MAX)-Notation) an. Sollte die o. g. Spezifikation nicht ausreichend sein, geben Sie Ihre genutzten Annahmen kurz an.
- b) Übertragen Sie Ihr ER-Modell in das relationale Datenmodell. Erstellen Sie hierzu die Tabellen der Entities Steuerklasse, Produkt und Bestellung mit Hilfe von CREATE TABLE Statements aus SQL. Verwenden Sie sinnvolle Datentypen.
- c) Geben Sie ein passendes CREATE TABLE SQL-Statement zur Erstellung der *passt_zu*-Tabelle an. Definieren Sie, wenn möglich, Fremdschlüssel und geben Sie die ON UPDATE- und die ON DELETE-Modi an.

Aufgabe 2: (SQL)

Gegeben sind folgende Relationen aus einer Musiksammlung:

$$\begin{aligned} \text{INTERPRET}(\text{MID}, \text{Name}, \text{Label}) \\ \text{ALBUM}(\text{AID}, \text{MID}[\text{INTERPRET}], \text{Name}, \text{Verkaeufe}) \\ \text{SONG}(\text{AID}[\text{ALBUM}], \text{TID}, \text{Titel}, \text{Laenge}) \end{aligned}$$

Zur Vereinfachung enthält also jedes Album ausschließlich Titel desselben Interpreten.

- a) Schreiben Sie eine SQL-Anweisung, die einen neuen Interpreten namens *Nena* vom Label *NDW* anlegt.
- b) Schreiben Sie eine SQL-Anweisung, die einen View erzeugt, der zu jedem Interpretennamen die Gesamtzahl all seiner verkauften Alben enthält.
- c) Schreiben Sie eine SQL-Anweisung, die alle Alben löscht, zu denen kein Song vorliegt.
- d) Schreiben Sie eine SQL-Anweisung, die alle Songtitel zurückgibt, deren Länge größer ist als die durchschnittliche Länge aller Songs. Die Songs sollen absteigend nach ihrer Länge ausgegeben werden. Sie können davon ausgehen, dass die Songlänge in Sekunden als Ganzzahl hinterlegt ist.
- e) Schreiben Sie eine SQL-Anweisung, die zu jedem Label die Anzahl der gelisteten Interpreten zurückliefert.
- f) Schreiben Sie eine SQL-Anweisung, die die Verkaufszahlen aller Alben um eins erhöht.
- g) Diskutieren Sie kurz, ob Name statt MID ein geeigneter Primärschlüssel für `INTERPRET` gewesen wäre.

Aufgabe 3: (Entwurfstheorie)

Gegeben sei das Relationenschema $R = (U, F)$ mit der Attributmenge U und der Menge F von funktionalen Abhängigkeiten:

$$\begin{aligned} U &= \{A, B, C, D, E\}, \\ F &= \{\{B, E\} \rightarrow \{A, D\}, \{B, C\} \rightarrow \{A, E\}, \{E\} \rightarrow \{C\}\}. \end{aligned}$$

Geben Sie für alle Antworten jeweils Begründungen an.

- a) Erklären Sie die bekannten Anomalien anhand von R .
- b) Geben Sie alle Schlüssel sowie die Nichtschlüsselattribute für das Relationenschema R an.
- c) Zeigen Sie, dass R in 3NF ist.
- d) Ist R auch in BCNF?
Geben Sie – falls R nicht in BCNF ist – eine BCNF-Zerlegung von R an.
- e) Geben Sie eine Basis G von F an und führen Sie damit die 3NF-Synthese aus.