Dies ist eine Klausur mit geschlossenen Büchern: Die einzigen erlaubten Hilfsmittel sind leeres Papier, Stifte und Ihr Kopf, aber Sie können eine handgeschriebene A4-Seite mit Informationen verwenden, die Sie für die Lösung der Aufgaben für nützlich halten. Erläutern Sie Ihre Überlegungen. Schreiben Sie klar und deutlich, im Sinne von Logik, Sprache und Lesbarkeit. Die Klarheit Ihrer Erklärungen wirkt sich auf Ihre Note aus. Viel Glück!

Schreiben Sie Ihren Namen und Ihre Matrikelnummer auf *alle* Lösungsblätter und hier. Name: .....
Am Ende der Prüfung geben Sie bitte *alle* Blätter, die Sie erhalten haben, einschließlich dieses, ab. Matrikelnummer: .....

Problem 1 [30%] Entwerfen Sie das Entity-Relationship-Schema einer Anwendung zur Verwaltung von Bergexkursionen, für die folgende Informationen von Interesse sind. Von jeder *Exkursion* interessieren uns der Code (Identifikator), die Kosten, die Personen, die an der Exkursion teilgenommen haben (mindestens eine), und die enthaltenen Bergbesteigungen (mindestens eine). Für jede Exkursion findet höchstens eine Besteigung pro Tag statt, und von jeder solchen *Besteigung* (die spezifisch für diese Exkursion ist) interessieren uns das Datum, an dem sie stattfand, die Dauer und der Hauptberg, der bestiegen wurde. Zusätzlich gibt es *Sonderbesteigungen*, bei denen uns auch das erforderliche Niveau und die zusätzlich zum Hauptberg bestiegenen Berge (mindestens einer) interessieren (diese Berge werden als sekundäre Berge der Besteigung bezeichnet), jeweils mit der zusätzlichen Besteigungszeit und der Person, die als Führer bei der Besteigung dieses sekundären Berges fungierte. Beachten Sie, dass eine Person nicht als Führer bei mehr als einer Besteigung eines sekundären Berges am selben Tag fungieren kann (wohl aber an verschiedenen Tagen). Von jedem *Berg* interessieren uns der Name (Identifikator), die Höhe und die GPS-Koordinaten. Von jeder *Person* interessieren uns die Sozialversicherungsnummer (Identifikator), der Vorname, der Nachname und das Alter.

Problem 2 [40%] Führen Sie das logische Design der Datenbank durch und erstellen Sie das vollständige relationale Schema mit Constraints unter Berücksichtigung der folgenden Hinweise: *(i)* jedes Mal, wenn wir auf eine Besteigung zugreifen, interessieren wir uns auch für ihren Hauptberg; *(ii)* jedes Mal, wenn wir auf die Informationen über eine Besteigung zugreifen, möchten wir wissen, ob es sich um eine Sonderbesteigung handelt, und falls ja, möchten wir das erforderliche Niveau wissen.

In Ihrem Design sollten Sie der im Kurs vorgestellten Methodik folgen und Folgendes erstellen:

1. [7%] das restrukturierte Entity-Relationship-Schema (falls nötig mit externen Constraints),
2. [25%] die direkte Übersetzung in das relationale Modell (falls nötig mit externen Constraints), und
3. [8%] das restrukturierte relationale Schema (erneut mit Constraints).

Sie sollten explizit begründen, wie die obigen Hinweise Ihr Design beeinflussen.

Problem 3 [20%] Betrachten Sie eine Datenbank B, die die folgenden zwei Tabellen enthält: *(i)* `Nodes(node)`, die alle Knoten eines gerichteten Graphen G speichert, und *(ii)* `Edges(start,end)`, die alle Kanten von G speichert, wobei eine Kante von Knoten n_1 zu Knoten n_2 durch das Tupel $t = \langle n_1, n_2 \rangle$ in der Relation `Edges` dargestellt wird, für das $t.\texttt{start} = n_1$ und $t.\texttt{end} = n_2$ gilt.

Wenn G die Kante von n_1 zu n_2 enthält, dann wird n_2 als *Nachfolger* von n_1 in G bezeichnet und n_1 als *Vorgänger* von n_2 in G. Darüber hinaus wird ein Knoten als *Quelle* bezeichnet, wenn er mindestens einen Nachfolger und keinen Vorgänger hat.

Wir wissen, dass die Datenbank B sowohl

(i) die Fremdschlüsselbeschränkung von `start` in `Edges` auf `node` in `Nodes` erfüllt, d.h., $\texttt{Edges}[\texttt{start}] \subseteq \texttt{Nodes}[\texttt{node}]$, und

(i) die Fremdschlüsselbeschränkung von `end` in `Edges` auf `node` in `Nodes` erfüllt, d.h., $\texttt{Edges}[\texttt{end}] \subseteq \texttt{Nodes}[\texttt{node}]$.

Beantworten Sie die folgenden Fragen.

1. [8%] Schreiben Sie eine *SQL*-Abfrage, die die durchschnittliche Anzahl von Vorgängern der Knoten von G zurückgibt (denken Sie daran, auch Knoten ohne Vorgänger zu berücksichtigen).
2. [12%] Schreiben Sie eine *relationale Algebra*-Abfrage, die alle Knoten von G berechnet, die als Vorgänger nur Quellenknoten haben (d.h. alle Knoten von G, die keinen Vorgänger haben, der kein Quellenknoten ist).

Problem 4 [10%] Beziehen Sie sich weiterhin auf die in Problem 3 beschriebene Datenbank B, und nehmen Sie an, dass die Fremdschlüsselbeschränkung von `start` in `Edges` auf `node` in `Nodes` als `ON DELETE RESTRICT` definiert ist, während die Fremdschlüsselbeschränkung von `end` in `Edges` auf `node` in `Nodes` als `ON DELETE CASCADE` definiert ist.

Beantworten Sie die folgenden Fragen.

1. [5%] Ist es immer möglich, das Löschen eines beliebigen Tupels aus der `Nodes`-Relation des Graphen G nur durch Ausführen von Operationen der Form "`DELETE FROM Nodes WHERE ...`" zu erreichen? Wenn die Antwort positiv ist, geben Sie Gründe an; wenn die Antwort negativ ist, sagen Sie, welche Knoten gelöscht werden können und welche nicht, und geben Sie erneut Gründe an.

2. [5%] Beantworten Sie die Frage im vorherigen Punkt, wenn der Graph G ein Baum ist, der, wie Sie sich erinnern, ein spezieller Graph ist, bei dem eine Kante von n_1 zu n_2 die hierarchische Beziehung zwischen dem Elternknoten n_1 und dem Kindknoten n_2 anzeigt.