1 Relationale Algebra und SQL (5 Punkte)

Gegeben sei das Relationenschema aus der zweiten und dritten Übung.

[Personen] : {[PID: int, Name: string, Geburtsjahr: int, Wohnort: string]}
[Schüler*innen] : {[SID:(Personen→PID), Klassenstufe: int, Klassenraum: int, Schulform: string]}
[Klausuren] : {[KID: int, Fach: string, Thema: string, Dauer: int]}
[Lehrer*innen] : {[LID:(Personen→PID), Hauptfach:string, Gehalt:int, Dienstjahre:int]}
[unterrichten] : {[Schüler*in:(Schüler*innen→SID), Datum:date, Uhrzeit:time,
Lehrer*in:(Lehrer*innen→LID), Fach: string)]}
[korrigieren] : {[Klausur:(Klausuren→KID), Schüler*in:(Schüler*innen→SID),
Lehrer*in:(Lehrer*innen→LID), Note: int]}

(a) Übersetzen Sie die folgenden Ausdrücke der relationalen Algebra in SQL Anfragen ohne Unteranfragen:

1. $R_1 := ((\sigma_{\text{Wohnort = 'Saarbrücken'}} \text{ Personen}) \bowtie_{\text{PID = SID}} (\sigma_{\text{Schulform = 'Grundschule'}} \text{ Schüler*innen}))$
   $\pi_{\text{Geburtsjahr}} R$
2. $R_1 := \rho_{\text{Schüler*in'←Schüler*in,Lehrer*in'←Lehrer*in,Fach'←Fach}} (\sigma_{\text{Fach = 'Englisch'}} \text{ unterrichten})$
   $R_2 := (\text{korrigieren} \bowtie_{\text{Klausur = KID}} (\sigma_{\text{Fach = 'Französisch'}}) \text{ Klausuren}) \bowtie_{\text{Lehrer*in = Lehrer*in'}} R_1$
   $\pi_{\text{Gehalt}} R_2$
3. $R_1 := (\text{Personen} \bowtie_{\text{PID = SID}} \text{Schüler*innen}) \bowtie_{\text{SID = Schüler*in}} \text{korrigieren}$
   $\pi_{\text{Name, min(Note)}} (\sigma_{\text{avg(Note) <= 3}} (\gamma_{\text{SID, Name,min(Note), avg(Note)}} R_1))$

(b) Übersetzen Sie die folgenden SQL Anfragen in Ausdrücke der relationalen Algebra. Ersetzen Sie wenn möglich kartesische Produkte durch Joins:

1.
```
SELECT *
FROM   Schüler*innen, unterrichten
WHERE  Klassenraum = 202
       AND Fach = 'Geschichte'
       AND SID  = Schüler*in;
```

2.
```
SELECT   DISTINCT MIN(NOTE)
FROM     korrigieren, Lehrer*innen
WHERE    Gehalt >= 2000 AND Hauptfach = 'Physik'
         AND Lehrer*in = LID
GROUP BY LID
HAVING   MAX(NOTE) < 5;
```

2 Natürliche Sprache und SQL (5 Punkte)

Betrachten sie erneut das Relationenschema aus der zweiten und dritten Übung.

[Personen] : {[PID: int, Name: string, Geburtsjahr: int, Wohnort: string]}
[Schüler*innen] : {[SID:(Personen→PID), Klassenstufe: int, Klassenraum: int, Schulform: string]}
[Klausuren] : {[KID: int, Fach: string, Thema: string, Dauer: int]}
[Lehrer*innen] : {[LID:(Personen→PID), Hauptfach:string, Gehalt:int, Dienstjahre:int]}
[unterrichten] : {[Schüler*in:(Schüler*innen→SID), Datum:date, Uhrzeit:time,
Lehrer*in:(Lehrer*innen→LID), Fach: string)]}
[korrigieren] : {[Klausur:(Klausuren→KID), Schüler*in:(Schüler*innen→SID),
Lehrer*in:(Lehrer*innen→LID), Note: int]}

(a) Übersetzen sie folgende Ausdrücke der natürlichen Sprache in SQL Anfragen ohne Unteranfragen. Ihre Ausgaben sollen jeweils keine Duplikate enthalten:

1. Die SID von Schüler*innen der Klassenstufe 5, zusammen mit der jeweiligen Anzahl, wie oft diese bereits um 14 Uhr im Fach 'Erdkunde' unterrichtet wurden.
2. Die Klassenstufen der Schüler*innen, die genau 5 mal die Note 1 erreicht haben.
3. Das durchschnittliche Gehalt der Lehrer*innen, die schon mindestens 4 mal denselben/dieselbe Schüler*in im Fach 'Sport' unterrichtet haben oder durchschnittlich eine 3 bei Klausuren vergeben.
   Anmerkung: Bei dieser Anfrage dürfen Sie Unteranfragen nutzen.

(b) Übersetzen Sie folgende SQL Anfragen in natürliche Sprache:

1.
```sql
SELECT   DISTINCT Geburtsjahr
FROM     Personen
         JOIN Schüler*innen
              ON PID = SID
WHERE    Schulform = 'Gymnasium' AND Wohnort LIKE '%brücken'
ORDER BY Klassenstufe DESC;
```

2.
```sql
SELECT   k.Lehrer*in, MIN(u.Datum)
FROM     Klausuren
         JOIN korrigieren AS k
              ON KID = Klausur
         JOIN unterrichten AS u
              ON u.Lehrer*in = k.Lehrer*in
WHERE    u.Fach = 'Musik'
GROUP BY k.Lehrer*in
HAVING   MAX(NOTE) < 6;
```

3 SQL Fehlersuche (3 Punkte)

In den folgenden SQL Anfragen haben sich Fehler - im Sinne von Verletzungen von der in der Vorlesung formal eingeführten Syntax und Semantik, als auch vom SQL-Standard, jedoch nicht zwingend von der Syntax und Semantik von SQLite - eingeschlichen. Nennen Sie diese und beschreiben Sie kurz, wie sie behoben werden können. Die Anfragen wurden basierend auf dem aus der Vorlesung und dem SQL.ipynb-Notebook bekannten FotoDB Schema entwickelt:

[persons] : {[id: int, lastname: string, firstname: string, birthday: string]}
[employees] : {[personId:(persons→id), salary: int, experience: int]}
[seniors] : {[employeeId:(employees→personId), numGreyHairs: int, bonus: int]}
[salespersons] : {[employeeId:(employees→personId), areaOfExpertise:string]}
[photographers] : {[employeeId:(employees→personId)]}
[cameras] : {[id: int, brand: string, model: string)]}
[photos] : {[id: int, location: string, unix_time: int,
photographerId:(photographers→employeeId), cameraId:(cameras→id)]}

1.
```
SELECT  employeeId, (42+-1=NULL) AS nice
FROM    photographers AS P
        JOIN seniors AS S ON S.employeeId = P.employeeId
HAVING  COUNT(*) = 2;
```

2.
```
SELECT   model
FROM     cameras
GROUP BY brand
WHERE    brand <> 'Nokia' AND id >= 20;
```

3.
```
SELECT   bonus, SUM(salary) AS SalarySum
FROM     employees
GROUP BY employeeId
HAVING   SalarySum > 100;
```

4 NSA und SQL (7 Punkte)

Gegeben sei das folgende Relationenschema, das auf dem Datensatz aus dem Notebook NSA.ipynb aus der Vorlesung basiert. Sie finden das entsprechende Notebook im Github Repository der Vorlesung.

[households] : {[id: int, street: string, postcode: int, city: string, floor: int]}
[citizens] : {[id: int, firstname: string, lastname: string, birthday: string]}
[livingIn] : {[citizen_id:(citizens→id), start: string, until: string, household_id:(households→id)]}
[articles] : {[id: int, label: string, unit: int]}
[nutritionalValues] : {[id: int, calories: int)]}
[purchases] : {[article_id:(articles→id), citizen_id:(citizens→id), date: string, amount: real)]}

Ihre Aufgabe ist es, die folgenden Anfragen in natürlicher Sprache in passende SQL Anfragen zu übersetzen. Sie können Ihre Anfragen durch Einfügen an die vorgesehene Stelle des beigefügten Notebooks testen. Reichen Sie Ihre Abgabe für diese Aufgabe bitte als `jupyter.txt`, die alle drei Anfragen klar voneinander getrennt enthält, ein.

Beachten Sie, dass Daten (*start* und *until* in `livingIn`, *birthday* in `citizens`, sowie *date* in `purchases`) in diesem Schema immer dem Format `YYYY-MM-DD HH:MI:SS` folgen.

1. Identifizieren Sie die Haushalte, die eine 13 als Hausnummer haben und in denen bereits insgesamt mindestens zwei Bürger*innen gelebt haben oder leben, die vor 1920 geboren wurden. Ihre Ausgabe soll dabei aus den folgenden Werten bestehen und aufsteigend nach dem Straßennamen sortiert sein:

   - Die Straße des Haushalts als „Straße“,
   - Die Postleitzahl des Haushalts als „PLZ“,
   - Die Stadt des Haushalts als „Stadt“.

   Hinweis: Beachten Sie hierbei, dass die Hausnummer immer am Ende des Attributs *street* in `households` steht.

2. Identifizieren Sie für jeden individuellen Artikel, welche Bürger*innen in einem Einkauf am meisten von diesem gekauft haben. Die Ausgabe sollte dabei aus den folgenden Werten bestehen und absteigend nach der Bezeichnung des Lebensmittels sortiert sein und zudem nur die ersten 10 Tupel enthalten:

   - Die Bezeichnung des Lebensmittels als „Bezeichnung“,
   - Die maximale Menge (in Litern oder Kilogramm), die in einem Einkauf von diesem Artikel gekauft wurde als „Menge“,
   - Der Vorname des/der Bürgers/Bürgerin als „Vorname“,
   - Der Nachname des/der Bürgers/Bürgerin als „Nachname“

3. Identifizieren Sie die Bürger*innen, die zwischen 1900 und 1949 (jeweils inklusive) in genau zwei Haushalte eingezogen sind. Berechnen Sie zusätzlich die von diesen Bürger*innen in dem Zeitraum zwischen 1900 und 1949 (beides inklusive) jeweils gekauften Kalorien. Die Ausgabe sollte absteigend sortiert sein nach den gekauften Kalorien:

   - Den Vornamen des/der Bürgers/Bürger*in als „Vorname“,
   - Den Nachnamen des/der Bürger*in als „Nachname“,
   - Die in diesem Zeitraum insgesamt von dieser Bürger*in gekauften Kalorien als „Kalorien“.

   Hinweis: Beachten Sie, dass die *calories* in `nutritionalValues` in kcal/100 g angegeben sind, während die Menge an dem gekauften Artikel in `purchases` (*amount*) in kg angegeben ist. Sie müssen also kcal/100g in kcal/kg umwandeln.

Abgabe

Lösungen sind in Teams von 2 bis 3 Studierenden bis zum 19. Mai 2022, 10:15 Uhr über Ihre persönlichen Statusseite im CMS einzureichen. Nutzen Sie hierfür die Team Groupings Funktionalität im CMS.

Ihre Abgabe muss dem folgenden Format entsprechen:

```
abgabe.zip
├── abgabe.pdf
└── jupyter.txt
```

Hierbei enthält `abgabe.pdf` Ihre Lösungen zu Aufgabe 1, 2 und 3 und `jupyter.txt` Ihre Lösung zu Aufgabe 4. Achten Sie darauf, dass Sie nur die von Ihnen zu ergänzenden Jupyter Zellen so kopieren, dass Einrückung und Formatierung korrekt sind.

Abgaben, die nicht den oben angegeben Vorgaben entsprechen, führen zu Punktabzug. Einzelabgaben werden nicht mehr korrigiert und mit 0 Punkten bewertet.