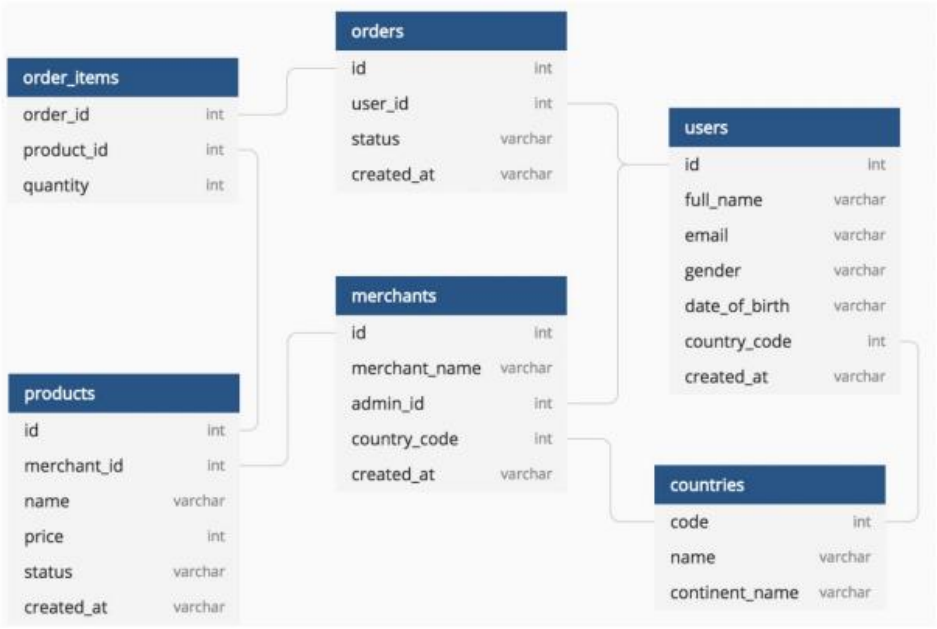
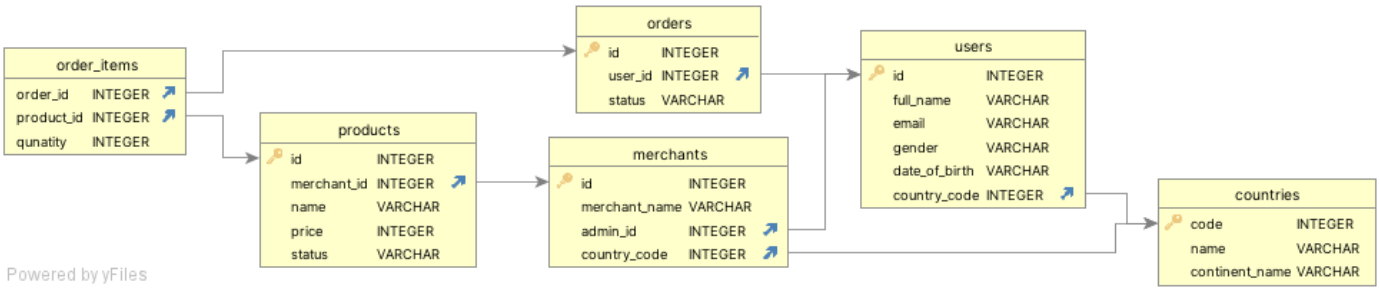
Part I

1. Use DB Browser for SQLite to create the following database and fill every table with at least two entries:



Submitted **Exercise_I.db**

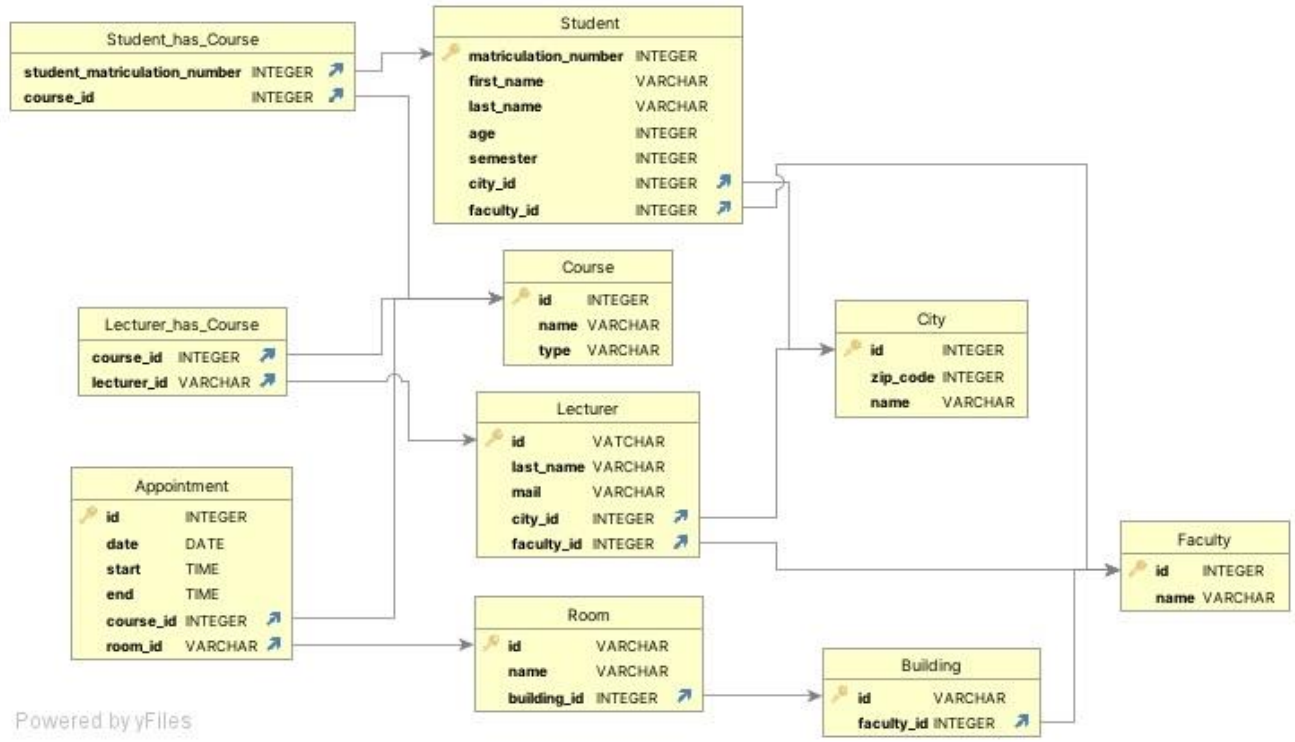
Database Schema:



Part II:

1. Download the "university.sqlite" database and create the corresponding database schema (drawing, PP, ...)

Database schema



2. Write SQL-Statements to answer the following questions:

I. How many students are named “Eibo”?

Ans: 11 students are named “Eibo”.

SQL>

```
SELECT * FROM Student WHERE Student.first_name = "Eibo";
```

```
1  SELECT *
2  FROM Student
3  WHERE Student.first_name = "Eibo";
4
```

	matriculation_number	first_name	last_name	age	semester	city_id	faculty_id
1	166075770	Eibo	Taglieber	22	5	583	1
2	166098618	Eibo	Vogler	21	3	961	3
3	166256706	Eibo	Muensinger	18	10	849	6
4	166471284	Eibo	Lieb	22	2	470	9
5	166481364	Eibo	Engert	27	2	571	2
6	166633236	Eibo	Lahmann	28	13	540	3
7	166637352	Eibo	Meixner	21	10	11	7
8	166721436	Eibo	Dellmann	27	13	781	10
9	166918836	Eibo	Reustlen	27	6	1089	2
10	166934754	Eibo	Lebhardt	18	8	396	6
11	166970832	Eibo	Seiff	26	12	180	4

```
Execution finished without errors.
Result: 11 rows returned in 33ms
At line 1:
SELECT *
FROM Student
WHERE Student.first_name = "Eibo";
```

II. What is their “matriculation number”?

```
SELECT Student.matriculation_number, Student.first_name
FROM Student
WHERE Student.first_name = "Eibo";
```

	matriculation_number	first_name
1	166075770	Eibo
2	166098618	Eibo
3	166256706	Eibo
4	166471284	Eibo
5	166481364	Eibo
6	166633236	Eibo
7	166637352	Eibo
8	166721436	Eibo
9	166918836	Eibo
10	166934754	Eibo
11	166970832	Eibo

```
Execution finished without errors.
Result: 11 rows returned in 34ms
At line 1:
SELECT Student.matriculation_number, Student.first_name
FROM Student
WHERE Student.first_name = "Eibo";
```

III. What are the most common last names?

Ans. Weiss, Bartenbach

SQL>

```
SELECT COUNT(Student.last_name) As Number, Student.last_name
FROM Student
GROUP BY Student.last_name
ORDER BY COUNT(Student.last_name) DESC;
```

```
SELECT COUNT(Student.last_name) As Number, Student.last_name
FROM Student
GROUP BY Student.last_name
ORDER BY COUNT(Student.last_name) DESC;
```

	Number	last_name
1	12	Weiss
2	12	Bartenbach
3	10	Staehelin
4	10	Schober
5	9	Schollerer
6	9	Roggenburg

```
Execution finished without errors.
Result: 12151 rows returned in 216ms
At line 1:
SELECT COUNT(Student.last_name) As Number, Student.last_name
FROM Student
GROUP BY Student.last_name
ORDER BY COUNT(Student.last_name) DESC;
```

IV. What rooms belong to the Architecture faculty?

Ans. 50.35 Fasanengarten- Hörsaal (HS a.F.), 50.35 SR a. F. (R 101)

SQL>

```
SELECT Room.id, Room.name AS room_name, Room.building_id
FROM Faculty
JOIN Building ON Faculty.id = building.faculty_id
JOIN Room ON Building.id = Room.building_id
WHERE Faculty.name = 'Architektur';
```

```
1  SELECT Room.id, Room.name AS room_name, Room.building_id
2  FROM Faculty
3  JOIN Building ON Faculty.id = building.faculty_id
4  JOIN Room ON Building.id = Room.building_id
5  WHERE Faculty.name = 'Architektur';
```

	id	room_name	building_id
1	0x2AE6A5667E982D40932A83E233C4C2D2	50.35 SR a. F. (R 101)	50.35
2	0x843007CD16CEDF4DB19740F3FA75C9BF	50.35 Fasanengarten-H�rsaal (HS a.F.)	50.35

```
Execution finished without errors.
Result: 2 rows returned in 12ms
At line 1:
SELECT Room.id, Room.name AS room_name, Room.building_id
FROM Faculty
JOIN Building ON Faculty.id = building.faculty_id
JOIN Room ON Building.id = Room.building_id
WHERE Faculty.name = 'Architektur';
```

V. In aggregate, which city do the most lecturers and students come from?

ANS. Most lecturers come from Neufra; while most students come from Bischweier.

But in aggregrate, most of the lecturers and students come from Bischweier.

SQL>

```
SELECT City.id, City.name,
       COUNT(DISTINCT Student.matriculation_number) AS
student_count,
      COUNT(DISTINCT Lecturer.id) AS lecturer_count
FROM City
LEFT JOIN Student ON City.id = Student.city_id
LEFT JOIN Lecturer ON City.id = Lecturer.city_id
GROUP BY City.id, City.name
```

```
ORDER BY (student_count + lecturer_count) DESC;
```

```
SELECT City.id, City.name,
       COUNT(DISTINCT Student.matriculation_number) AS student_count,
       COUNT(DISTINCT Lecturer.id) AS lecturer_count
FROM City
LEFT JOIN Student ON City.id = Student.city_id
LEFT JOIN Lecturer ON City.id = Lecturer.city_id
GROUP BY City.id, City.name
ORDER BY (student_count + lecturer_count) DESC;
```

	id	name	student_count	lecturer_count
1	456	Bischweier	49	3
2	908	Krauchenwies	42	5
3	333	Niederschlettenbach	42	2
4	396	Hainfeld	40	4
5	959	Wildberg	43	1
6	35	Schwörstadt	39	3
7	60	Schopfheim	41	1
8	423	Berg (Pfalz)	42	0
9	573	Haßmersheim	41	1
10	231	Gailingen am Hochrhein	39	2
11	699	Aichwald	35	6
12	917	Albstadt	39	2
13	919	Neufra	34	7

VI. What are the matriculation numbers of all of Lecturer Fichtner's students?

ANS.

SQL>

```
SELECT DISTINCT s.matriculation_number, l.last_name

FROM Student s

JOIN Student_has_Course shc ON s.matriculation_number =
shc.student_matriculation_number

JOIN Lecturer_has_Course lhc ON shc.course_id =
lhc.course_id

JOIN Lecturer l ON lhc.lecturer_id = l.id

WHERE l.last_name = "Fichtner";
```

VII. Which buildings do not have lecture rooms assigned?

SQL>

```
SELECT b.id AS building_id, b.faculty_id

FROM Building b

LEFT JOIN Room r ON b.id = r.building_id

WHERE r.id IS NULL;
```

VIII. What is the matriculation number and name of the students who come from cities with a maximum of 15 students?

ANS.

SQL>

```
SELECT DISTINCT s.matriculation_number, s.first_name || ' '
|| last_name AS FullName

FROM Student s

JOIN (

    SELECT city_id, COUNT(*) AS student_count

    FROM Student

    GROUP BY city_id

    HAVING student_count <= 15

) city_count ON s.city_id = city_count.city_id;
```