University of Basel

Prof. Dr. Heiko Schuldt
David Lengweiler, M. Sc.
Martin Vahlensieck, M. Sc.

Introduction to Databases — Autumn 2024

Exercise 2

(Theory) **Hand-in: 23.10.2024 (ADAM, 23:59)**
(Practical) **Hand-in: 23.10.2024 (during exercise)**

Solving the Exercises: The exercises can be solved in small groups of a maximum of two people. Use the notations introduced in the lecture. The DMI plagiarism guidelines apply for this lecture.

Submission Information: Please upload all (Theory) deliverables BEFORE the deadline to ADAM as **a single PDF**. For (Practical) exercise check the **Hand-In** instructions. The solutions must contain your names. Solutions that are handed in too late cannot be considered.

Task (Practical) 1: Connecting to a Database (3 Points)

This week you will do the same SQL queries as last week — but this time with Python.

To connect to PostgreSQL using Python, you first need a so-called driver. This driver knows how to talk to PostgreSQL, and translates between Python and PostgreSQL types. For Python you can use the `psycopg2` driver (can be install the `psycopg2-binary` package).

After installing the driver you need to connect to the database. Use the same connection details and credentials as for the last exercise. Once connected you need a *cursor*. This cursor can then be used to *execute* statements and retrieve the results.

Here is an example on how to print the PostgreSQL server version:

```
import psycopg2

con = psycopg2.connect(
    host='fillme',
    dbname='fillme',
    user='fillme',
    password='fillme',
)

cur = con.cursor()
cur.execute('SELECT version()')

print('Version:', cur.fetchone()[0])
```

You need to do the following tasks:

- Avoid hard-coding the password in the source file (e.g. prompt the user or use environment variables).
- Print how many games were won by white.
- Print the number of games per month. Print the name of the month, not the number.
- Print the total number of games.

Hints:

- Use virtual environments to manage Python packages.

Hand-In: Show your code and the printed results to an assisstant/tutor.

Task (Theory) 2: ER University (4 Points)

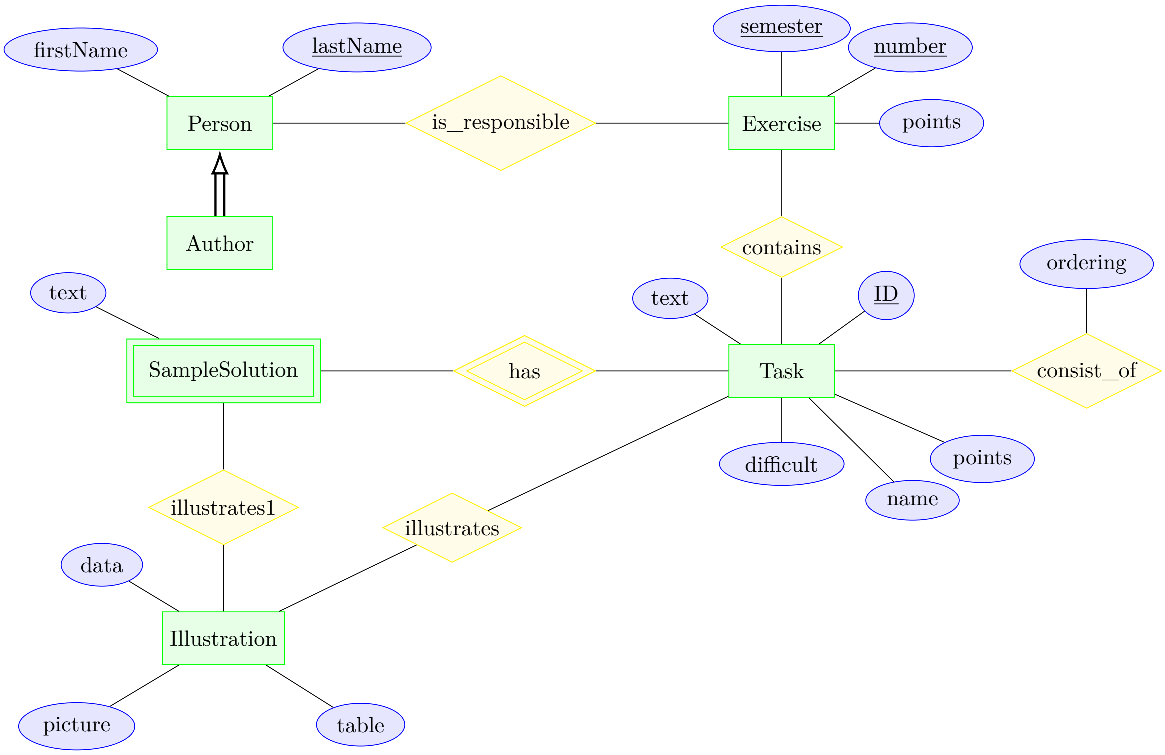
For this exercise you will find errors in an entity-relationship (ER) model for a provided scenario, which is as follows:

You are tutor of the lecture "Introduction to Databases" and you have a large pool of theoretical and practical tasks. You want to administer the tasks in a database to be able to create the exercises for the current term more simply and faster.

The requirements for the database are the following:

> Tasks have an identifier (ID) and one or several authors who are identified by their first and last name. In addition, the tasks contain a text and possibly also pictures and / or tables. Furthermore, a task possesses a name, a certain degree of difficulty and an amount of points assigned to it. A task can consist of several subtasks whereby also the sequence of the subtasks has to be specified. A subtask is also a task which can contain by itself further subtasks. The points of a task correspond to the sum of the points of the subtasks. Tasks can optionally have a sample solution which again consists of text and possibly pictures and / or tables.
>
> Exercises should be arranged from the task pool. An exercise has a clear identification by semester and sequence number; one to four people are responsible for it. Furthermore, the exercise has a total score which is calculated by the sum of the points of its individual tasks.

You came up with the following ER, which is missing cardinalities and includes 5 mistakes.



Hand-In: Add the missing cardinalities and correct the 5 mistakes. For each mistake give a short explanation (one sentence) why it is wrong. Hand in your solutions via ADAM.

Task (Theory) 3: ER University Extended (3 Points)

For this exercise you will adjust the entity-relationship (ER) model from the previous scenario:

> Since you have successfully modeled your exercise pool, you have the idea that you could use this pool not only to create the exercises but also to arrange examinations. What changes do you have to make to your model to be able to do that? Note that there are examinations and re-examinations; their contents must not overlap and tasks used in exercises must not be used in examinations. In general, an examination task should be used only once while exercise tasks can be used several times. Which of the conditions stated above cannot be represented in an ER diagram?

Hand-In: Do not enter your additions into the model of the task 2. Draw only the parts which change by the new circumstances. Hand in your solutions via ADAM.