EXAMEN DE FIN D'ÉTUDES SECONDAIRES – Sessions 2024

QUESTIONNAIRE

Date :	05.06.24		*Horaire :*	08:15 - 11:15	*Durée :*	180 minutes
Discipline :	AMINF	*Type :*	écrit	*Section(s) :*	CI	
				Numéro du candidat :		

Available documents:

- `Answers.dotx:` template used to insert your answers into.
- `Translation.drawio:` contains the CDM that is to be translated.
- `Database.sql:` MySQL database with a sample dataset.
- `Functions.pdf:` contains a list of known SQL functions.

Preparation and hand-in

In your repository (to be defined by each school), you will find a folder named **EXAMEN_AMINF**. Rename this folder by replacing its current name with your exam code (example: **LXY_CI_07**). All your files must be saved inside this folder which will be referred to as **your folder** in the remainder of this document.

Now, open the file **Answers.dotx** which resides in your folder. Fill in the header of this file by adding your candidate number and the date. Save the resulting file inside of your folder by adapting its name to match the following example pattern: **LXY_CI_07_Answers.docx**. The prefix LXY is to match your school.

At the end of the exam, create a PDF from your answer file by matching the following example naming pattern: **LXY_CI_07_Answers.pdf**. Save all your queries by matching the following naming convention: **QueryX** where X is the number of the question, e.g. Query3.

Verify that:

- The PDF file contains all your answers,
- That all screenshots are legible.
- That all SQL queries are provided, one query per file.

Please, make sure to regularly save to your folder!

You will be evaluated solely on the contents of the PDF file!

Question 1 – Establishing a CDM 20p

Create a conceptual database model (CDM) of a dungeon management system for a pen and paper role playing game. Your model needs to handle the dungeon, its levels, as well as its inhabitants and the adventuring parties exploring the dungeon to find its riches. Use draw.io to establish the model and insert a screenshot (white background, no grid) into your answer document.

Image 1: The entry to a long-forgotten dungeon. Image composed by DALL·E 3.

1. Each dungeon has a unique name, a type as well as an indication whether it has been thoroughly explored or not. Furthermore, the dungeon should advertise its recommended level for adventurers to explore it. This is a sample of the data to store:

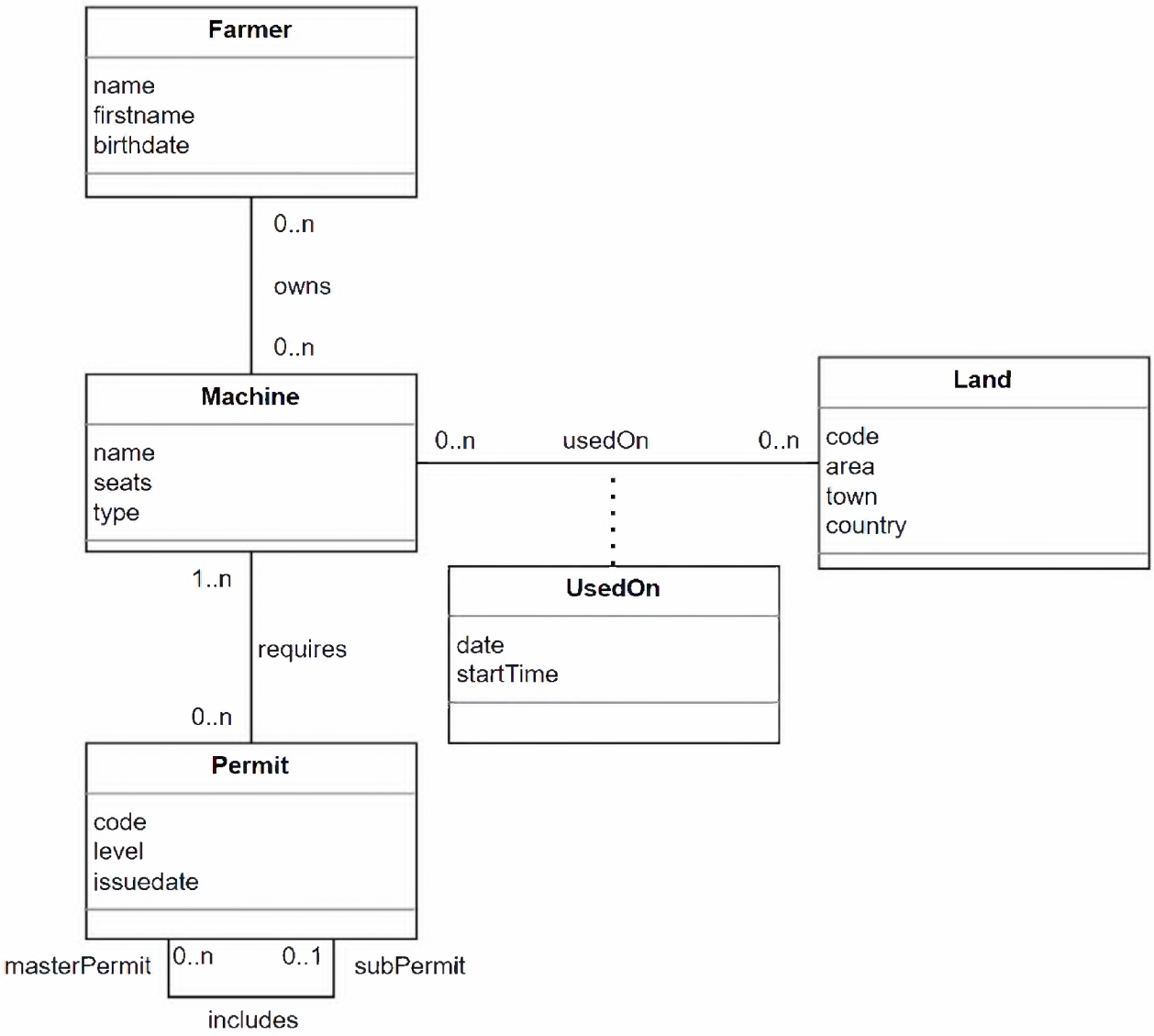
Name	Type	Recommended Level	Explored?
The Fortress of Doom	Fortress	12	No
Goblin Caverns of An'kabur	Subterranean Cavern	8	Yes
Haunted Ruins of Castelrock	Ruins	6	No
Fortress of Solitude	Fortress	10	Yes

2. Each dungeon has at least one floor. For each floor, we register its name (if it has one), its type and an indication as to whether the floor is illuminated or not. Some creatures prefer dwelling in darkness after all. Each floor is uniquely tied to the dungeon it features in.
3. A floor is composed of multiple rooms, ranging from huge rooms taking up the whole floor to tens of different rooms. Each room is registered with its size and an indication whether it is occupied and contains treasure. Furthermore, a room may contain one or more other rooms. Lastly, rooms are never spread across multiple floors.
4. Creatures may have made a dungeon their only home. For creatures, we save their name, type, role, and level. Creatures may work and patrol on one or multiple floors. However, it is not guaranteed that any creatures are working on a floor. Still, patrols make sure to cover each floor. To make sure a patrol is always present on each floor, creatures share information on the start and end time of their patrol shifts to make sure they run smoothly, and their homes are well secured.
5. Adventurers organize into parties to explore dungeons. For each party, we store their name, the number of adventurers in the given party, and their party level (the average of all adventurer's levels). Each party has a single leader chosen among the party members. A party may explore multiple dungeons, but they aren't really considered a party until they have at least explored one dungeon. Due to the huge number of available dungeons, some might never have been explored by a party. To be able to track parties across play sessions, we need to store the room the party is currently located in. Since some dungeons are hugely popular, some might see multiple parties explore them at the same time, possibly even encountering each other in the same room.

Question 2 – Translation – CDM to LDM 10p

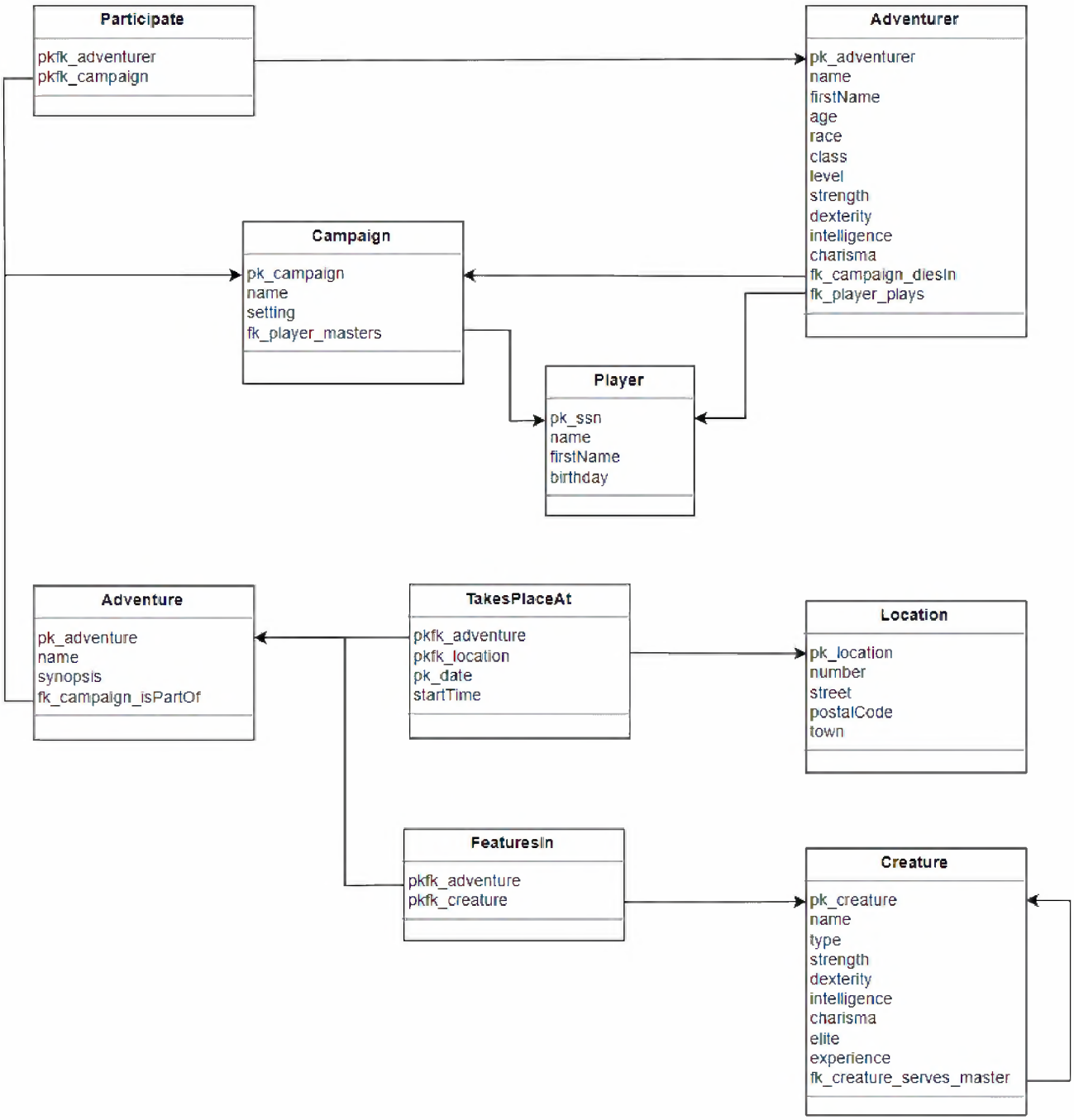
Translate the following conceptual database model (CDM) into its corresponding logical database model (LDM). Use draw.io and the provided `Translation.drawio` file. Insert a screenshot of your solution (white background, no grid) into your answer document.

Note: Consider real-life scenarios when translating the relations between machines and a plot of land. In general, a machine can be used on the same plot of land multiple times per day.



Question 3 – SQL Queries and Comprehension 30p

Create your database by importing the provided `Database.sql` file into MySQL Workbench. The database corresponds to the LDM provided below. Use the provided data set to test your queries. You may need to, and should, add additional entries to verify your queries are correct.



Solve the following problem statements. Insert your solution and the corresponding output, if applicable, into your answer document. Save each query in a separate SQL file.

1. Develop the SQL code to return the number of planned adventures that are yet to happen. The time reference must not be provided as a constant value. Provide the output as "Planned Adventures". **3p**
2. Develop the SQL code to return the name of the campaign, the first and last name of the player as well as the first and last name of the adventurer for all campaigns and living adventurers in ascending order of the campaign name. **5p**
3. Develop the SQL code to retrieve the name of all campaigns and their setting which do not feature any encounters with creatures. **3,5p**
4. Develop the SQL code to return all elite creatures who have never appeared in an adventure and that are the master of at least one other creature. **5p**
5. Develop the SQL code to return the name of all campaigns and the name of all their adventures. Display all the adventure names per campaign in a single column "Adventures". Order the result as by the example shown below and double check with data present in the initial data set: **5p**

Campaign	Adventures
Baldur's Gate	Escaping Candlekeep, The Mines of Nashkel, Baldur's Gate awaits, Murder!
Hoard of the Dragon Queen	Greenest in Flames, Raider's Camp, On the Road, Castle Naerytar
Full Force Ahead	The Derelict, A chance encounter, Hyperspace mysteries
On Elven Shores	The Eye of Uldûr, A sacrifice is made, Uldûr beckons
Lands of the Dark Wicche	Entering the domain of the Wicche, The Fall of the Wicche
Paradise Lost	Johnson, John Johnson, The origins of Paradise

6. Analyse the following SQL statements. Which statement will return a correct result (given a dataset that can return such a result) and why?

```
SELECT name
FROM Adventure
WHERE LOWER(name) = "%escape%";
```

```
SELECT name
FROM Adventure
WHERE LOWER(name) LIKE "%escape%";
```

2,5p

7. For each completed adventure, experience points are awarded to all participating players. This amount corresponds to the sum of experience provided by defeated creatures in each adventure as well as a one-time bonus of 300 points. Given that the database does not contain a field to store this value, the amount would have to be calculated at the end of each adventure. Since the number of creatures and the one-time bonus does not change, it may be better to store this data directly.
   Discuss this possibility by providing arguments in favour and against this solution and reach a final decision whether, in your professional opinion, the storage of this data is the best choice. **6p**