HAPPY
LEARNER

BUILD A BRIGHT FUTURE

DATABASE FINAL EXAM REPORT

Ernist Askar, Baltabaev Zhamin, Ubaydullaev Feruz, Stocco Clément, Ryskeldi Zhuldyz

Designing and Implementing script of a database for HL company

Task Description:

Our company, Happy Learner, is planning to create an online course platform for students and teachers who can teach and learn on this platform. As for this final project, we decided to develop a database for our system.

Happy learner database requirements:

- The database stores the student's details. Each student gets a unique id (8 digits) after signing up. They must enter their name, age, gender, language, email and educational background (high school, bachelor, master degree). A student can take as many courses as wanted in Korean, English, or both.

- The online payment subsystem enables a student to enroll in any course. Once the payment is completed, the admins should approve and enroll the student to the course. The database also records the enrollment id (8 digits), the paid amount, and the enrollment status.

- Happy Learner is looking for the best educator so the database stores the following information about each instructor: the educator's id (8 digits), name, age, gender, email, language, and an educator`s specialization, certified by its educational background achievement. The educational background shows the university or the company the instructor is coming from. An educator can teach different courses. An educator can teach in both Korean and English as long as he/she speaks the language fluently.

- A course has a scheduled duration which represents the number of hours of the online video course. A course can have more than one educator. The database stores the course id (8 digits), name, its description, the number of hours for a course to be completed, the course price, the educator id, the course category, the course language, the admin id who approved the course. A course can be in Korean, in English or both depending on the educator. Each course should be approved by an admin to be published.

- The educators provide documents for the students. It can be video contents, written documents. The database records the course material id, the document name and type.

- A course belongs to a category. For instance “Branding” and “Consumer behavior” belong to the “Marketing” category. The category id (6 digits) and its name are stored in the database.

- The students can provide feedback by commenting and rating the course. There is also a section in which the students’ questions and educators’ answers are sorted so that the next student may read the answer to their question or get the help of the admin. The database records the students' comments, feedback, and course rating.

- Each course offers a practical project or study case to ensure the user has understood the course thoroughly. These exams are designed by the educator and aim to cover the whole course. The student has to submit the final project that will be reviewed by the educator. The educator reviews the project of the student and decides if the project is satisfying. If it is, the review is “completed”, if not the review is “incomplete” and the student must submit a new project. The database stores the exam id (6 digits), the review status, the educator id, the course id, and student id, admin id. A completed exam means the student has finished the course.

- As soon as a course is fully completed, Happy Learner provides a certification to the student. The certification certifies they have followed a high-quality course. It guarantees that the user has developed important skills thanks to the company and the educator. Happy Lerner’s database stores the certification’s id (8 digits) and date, the course id and the student id. It allows us to prove that the user has effectively completed the course, and supports the user if a company wants to know more about the course the user attended. The student can put the course name and the certification id on his/her resume to demonstrate the skills he/she has got.

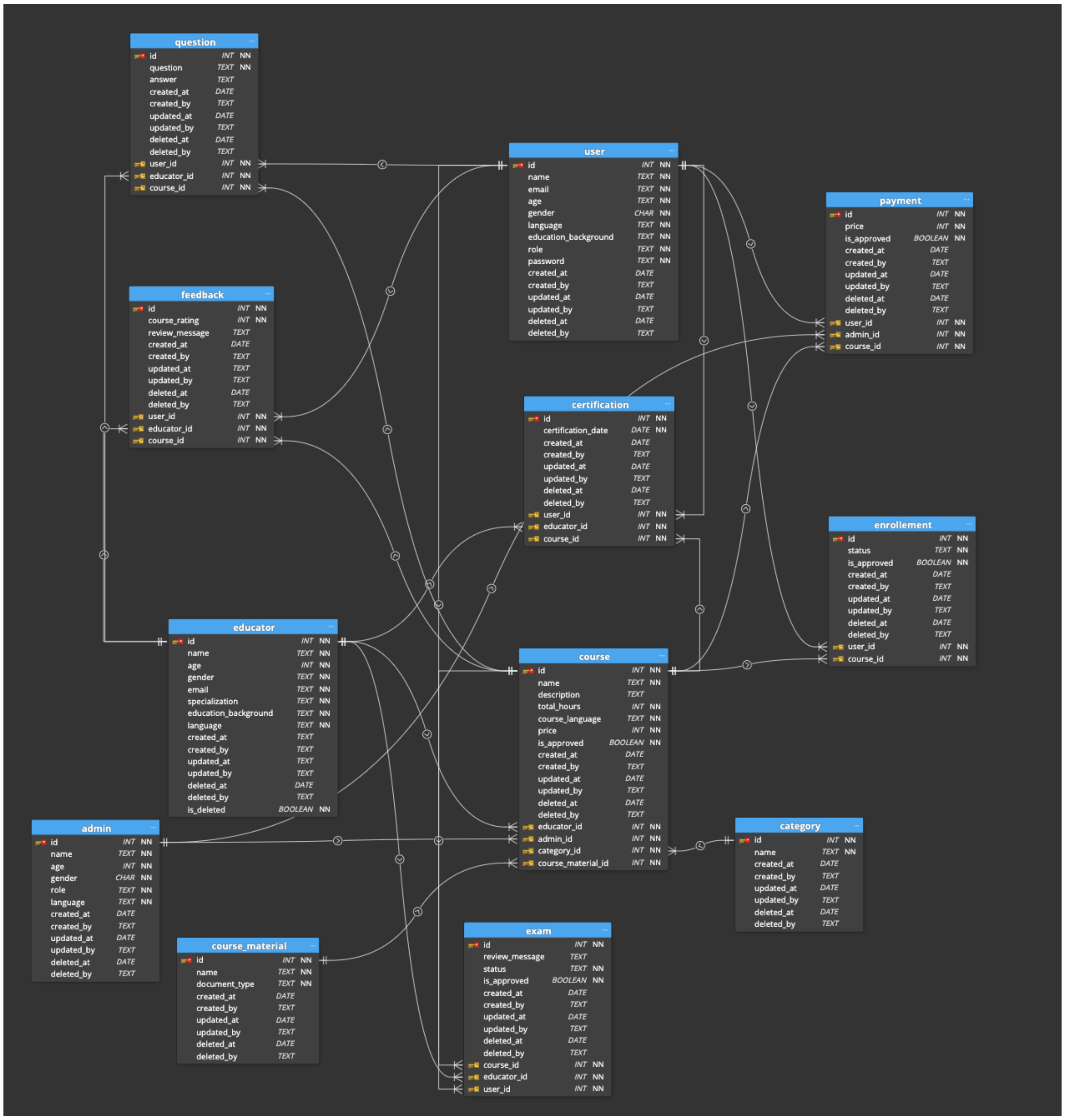
- The database records the details of each user. A user is given an id and a password to sign-in on the platform. All the credentials stored in the database are encrypted. There are 2 types of roles in this table (student and educator).

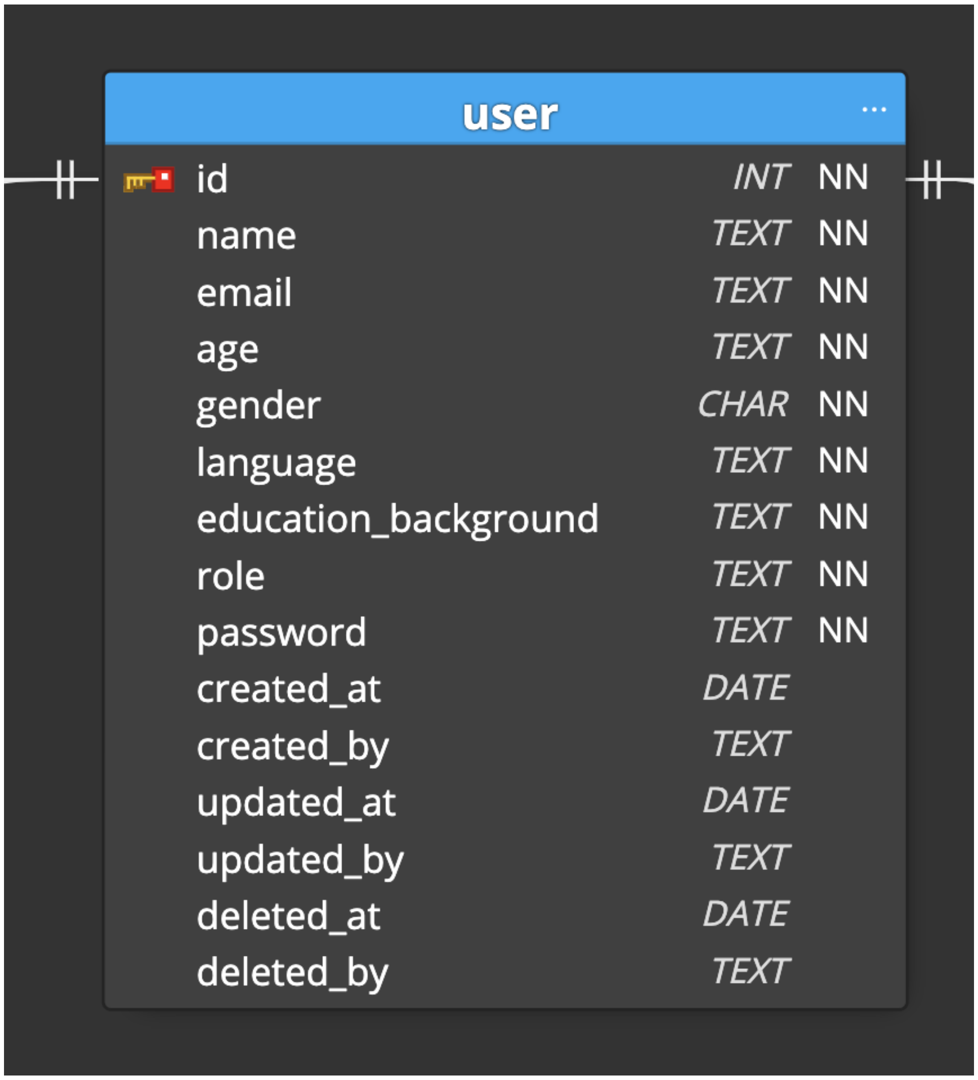
- Finally, the admins are managing our online platform and any trouble the user may encounter. For instance, they’ve got to approve the enrollment, the exam and the course before it is all published. The database stores the admin’s id, name, gender, role and language spoken.

- The database stores the created_on, created_by, changed_on, changed_by, deleted_on, deleted_by attributes for every table. And also the is_deleted attribute for the required tables (educator). These attributes enable us to look at the history of the information in the database.

ERD MODELING

- 12 tables
- one-to-one
- one-to-many
- Many-to-many

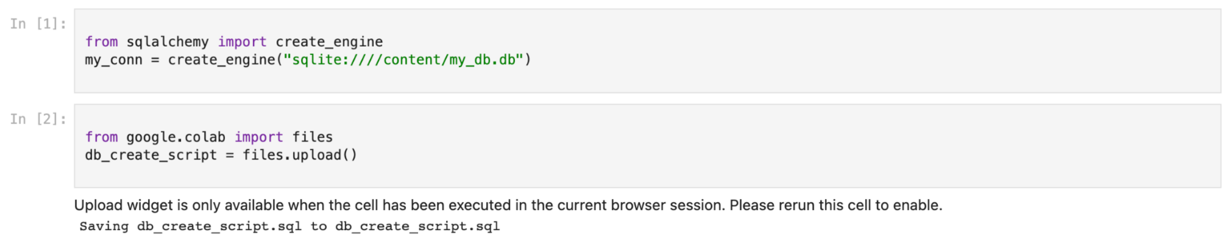




user		
id	INT	NN
name	TEXT	NN
email	TEXT	NN
age	TEXT	NN
gender	CHAR	NN
language	TEXT	NN
education_background	TEXT	NN
role	TEXT	NN
password	TEXT	NN
created_at	DATE	
created_by	TEXT	
updated_at	DATE	
updated_by	TEXT	
deleted_at	DATE	
deleted_by	TEXT	

In every table we have included created_at, created_by, updated_at, updated_by, deleted_at, and deleted_by as it was requested

SOURCE CODE FOR SQL PART



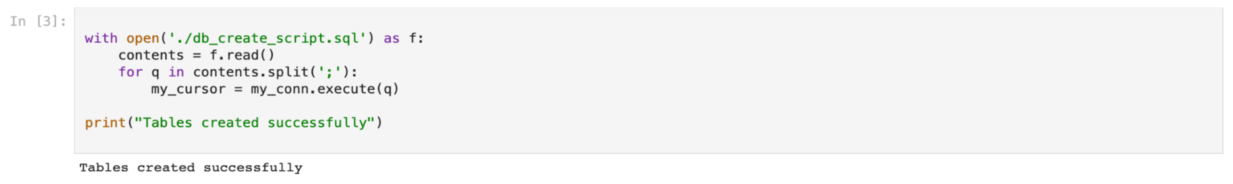
```
In [1]:
from sqlalchemy import create_engine
my_conn = create_engine("sqlite:////content/my_db.db")
```

```
In [2]:
from google.colab import files
db_create_script = files.upload()
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving db_create_script.sql to db_create_script.sql
```

Moving on to the Google Colab, the first script creates a .db file to be used later, the second one asks for the input file which creates a database table. Here the ‘db_create_script.sql’ file should be uploaded. The file is provided in the zip archive.

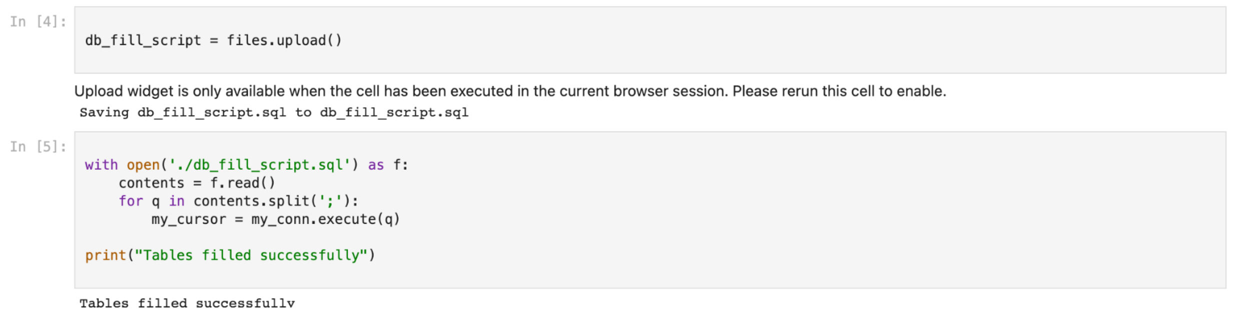


```
In [3]:
with open('./db_create_script.sql') as f:
    contents = f.read()
    for q in contents.split(';'):
        my_cursor = my_conn.execute(q)

print("Tables created successfully")
```

```
Tables created successfully
```

Then, when this script is run, you can see the output that the tables are created. So, our sql commands are executed successfully and database tables got created.



```
In [4]:
db_fill_script = files.upload()
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

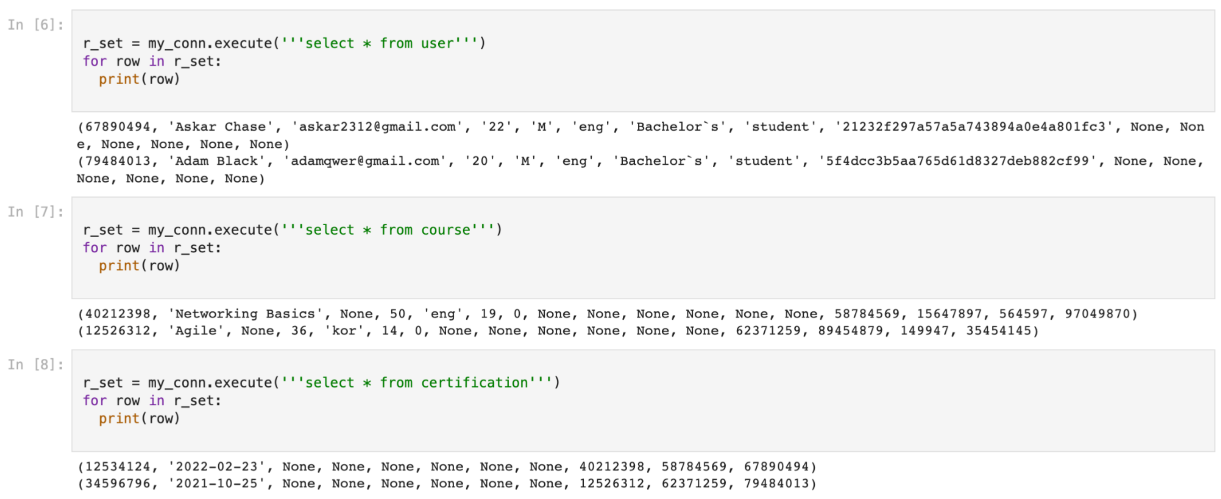
```
Saving db_fill_script.sql to db_fill_script.sql
```

```
In [5]:
with open('./db_fill_script.sql') as f:
    contents = f.read()
    for q in contents.split(';'):
        my_cursor = my_conn.execute(q)

print("Tables filled successfully")
```

```
Tables filled successfullv
```

Next, it asks for another input file to fill the database with data. The file is called ‘db_fill_script.sql’ which is also provided in the zip archive. And when the next script is executed, all sql commands inside the file are run, and you can see the output saying that tables are filled with data.



```
In [6]:
r_set = my_conn.execute('''select * from user''')
for row in r_set:
  print(row)
```

```
(67890494, 'Askar Chase', 'askar2312@gmail.com', '22', 'M', 'eng', 'Bachelor`s', 'student', '21232f297a57a5a743894a0e4a801fc3', None, Non
e, None, None, None, None)
(79484013, 'Adam Black', 'adamqwer@gmail.com', '20', 'M', 'eng', 'Bachelor`s', 'student', '5f4dcc3b5aa765d61d8327deb882cf99', None, None,
None, None, None, None)
```

```
In [7]:
r_set = my_conn.execute('''select * from course''')
for row in r_set:
  print(row)
```

```
(40212398, 'Networking Basics', None, 50, 'eng', 19, 0, None, None, None, None, None, None, 58784569, 15647897, 564597, 97049870)
(12526312, 'Agile', None, 36, 'kor', 14, 0, None, None, None, None, None, None, 62371259, 89454879, 149947, 35454145)
```

```
In [8]:
r_set = my_conn.execute('''select * from certification''')
for row in r_set:
  print(row)
```

```
(12534124, '2022-02-23', None, None, None, None, None, None, 40212398, 58784569, 67890494)
(34596796, '2021-10-25', None, None, None, None, None, None, 12526312, 62371259, 79484013)
```

To check whether everything is done well, you can run these scripts which display the data from different tables of the database. 'None' means NULL in the database. It is because some attributes can have NULL values sometimes.

Github Repo: https://github.com/citizenseven/final_project