Information and Communication Technologies — S6 — Work n°17

# Databases — Part 4 (and last part)

The objective of this sequence of works is to understand the concept of databases and to handle it with the SQL language: we will manipulate this language thanks to the `sqlite3` module in Python.

## 1 The company example — Work that sums up what we know

In this last part of the work, we review all our knowledge on another example: a company. This company hires employees having different skills in different departments. Each employee has a computer, and some of them are also computer administrators. The big picture of the database is the following:

- departments: name, nb_machines, nb_employees, #admin_id
- computers: #dpt_name, id, os, #employee_id
- employees: id, name, skill
- assignments: #dpt_name, #employee_id, #employee_name, year

The keys that start with a # are foreign keys (linked to another table). The underlined keys are the primary keys of each table. Of course, to create the tables, it is mandatory that all tables from which we use keys are already created. Thus, we have to create first employees, then departments, then computers, and then assignments.

As in previous work, the sql file that contains all the necessary data to create those tables is available at:
`http://www.barsamian.am/2020-2021/S6ICT/TP17_Company_creations.sql`
You can use this sql file to create the database. For example, you can download the python file from last works and modify it to suit your needs:
`http://www.barsamian.am/2020-2021/S6ICT/TP15_Hotels_database.py`
Then we have to fill the tables.

1. Fill the departments table via the following SQL commands:

```
INSERT INTO departments VALUES ('research', 45, 453, 27);
INSERT INTO departments VALUES ('marketing', 73, 89, 11);
INSERT INTO departments VALUES ('production', 54, 1245, 11);
INSERT INTO departments VALUES ('finance', 24, 24, 27);
INSERT INTO departments VALUES ('workshop', 62, 7, 11);
```

2. Fill the employees table via the Listing 1, by first downloading the file:

`http://www.barsamian.am/2020-2021/S6ICT/TP17_Company_employees.txt`

```
  if (len(cursor.execute("SELECT * FROM employees").fetchall()) == 0):
    f = open("TP17_Company_employees.txt", "r", encoding="utf-8")
    for line in f:
      values = line.strip().split(";")
      # WARNING: splitting the line, e.g., "12;jean;accountant;" leads to
      # FOUR sub-strings: "12", "jean", "accountant" and "". We have put
      # THREE "?" in the request, so we cannot directly use values (it
      # is an array with 4 cells), we instead use values[0:3] which is
      # an array with 3 cells: [values[0], values[1], values[2]]
      cursor.execute("INSERT INTO employees VALUES(?, ?, ?)", values[0:3])
    f.close()
```

Listing 1: Python code for question 2.

3. Fill the computers and assignments tables by downloading the following files and writing the correct python commands. Warning: in "TP17_Company_employees.txt", the separator was ";" whereas in the two following files, it is ",".

http://www.barsamian.am/2020-2021/S6ICT/TP17_Company_assignments.txt

http://www.barsamian.am/2020-2021/S6ICT/TP17_Company_computers.txt

4. Don't forget to save the database !

Now, you should be able to write and test the following requests:

5. What are the departments in which there are (strictly) more than 50 employees?
6. What are the names of employees from the research department?
7. In which departments is there at least one employee whose skill is management?
8. How many employees are in the production department? (the answer "3, of course" does not explain how to get the request in the general case...)
9. How many employees are there in each department? (use `GROUP BY dpt_name` instead of `WHERE ...` or write a loop in python).
10. Sort by ascending order of hiring date the employees of this company.
11. How many employees have a skill that begin with "i"?
12. What are the skills that have a "g" somewhere in their name?
13. What are the departments that do not use macOs as an OS?

## 2 The password example — Bonus work

1. Create a new database with the following scheme:

   - people: id, surname, first_name
   - identifiers: #id, login, pwd

   (the id field in the identifiers table is a foreign key from the people table)

2. In the people table, add 5 people of your choice.
3. Write a function that, given a surname and a first_name, outputs a login. The function must insert this login inside the identifiers table for each people in the people table (the pwd is a random string that uses 4 digits).

   Hint: Don't forget to handle the case where a login already exists. For the passwords, do you have to take special care? If yes, give a solution; if not, explain why.