ISN written exam
2nd year - November 2024

Duration : *2h*
Autorised documents : *Aucun.*

— The grading scale is 20.
— There are 7 pages - with 4 exercises.

Exercise 1 : Algo and Python Programming 1st year (4 pts)

In this exercise, **the only functions for lists available are `len` and `append`**. You are not allowed to use slicing.

1.1 Indexes of max values

(1.1) *Write in python a function having as parameters a list of numbers, and which produces the list of indexes corresponding to the maximal value(s) in the first list. For instance, for the list* $[12, 5, 12, 8, 12, 7]$ *the function returns the list : [0, 2, 4].*

1.2 index of the last appearance of a letter in a sentence

(1.2) *In Python, write a function that takes a string sentence and a single letter as parameters, and returns the index of the last occurrence of this letter in the sentence (in a left-to-right reading direction). It returns -1 if the letter is not present. Example : for the text "data base" and the letter 'e', the function returns : 8 ; for 'q', it returns -1.*

1.3 Sorting by list selection

Reminder : Descending sorting by selection consists of sorting a list by searching for the maximum value from each position, then placing it in its definitive position by performing a permutation if necessary.

(1.3) *Write a program that performs a descending sort of a list of integer lists using the selection sort algorithm. The sort criterion is the length of the list. Make sure the program is divided into functions.*

Example : for the list [[3],[1,2],[],[10,2,5]], the sorted list will be : [[10,2,5],[1,2],[3],[]].

Exercise 2 : SQL queries (8 pts)

A company offers introductory sports courses, which require a specific type of equipment kit. The type of kit specifies the sport, the level and the number of people involved. An initiation is booked for a given date and location by a customer, and provided by two members of staff. An equipment kit is borrowed on a certain date by a member of staff for a given reservation. Usually (but not always), it is the person responsible for the reservation, or the person assisting him/her, who carries out this borrowing. When the equipment is returned, the date of return is recorded. A comment can be made at the time of borrowing, and modified when the kit is returned. The information system uses the following relationships, where the attributes forming the key are underlined. The graphical representation of the relational schema is shown in figure 1.

initiations(<u>idInit</u>, titre, duree, prix, idType)
An initiation has an id (idInit), a title (titre), a duration (durée, in minutes) and a price (prix, in euros). We also store the id of the initiation kit type, idType, which is the foreign key refering to the relation **typeKits**.
typeKits(<u>idType</u>, sport, niveau, description, nbPers)
A kit type has an id (idType). We store the sport and the level (for instance, 'diving', 'beginner'), a description of the equipment included in this type of kit, and the number of people for whom it is designed.
kits(<u>idKit</u>, idType, etat, dateVerif)
A kit has an id(idKit) and belongs to a type of kit. Its status is memorized (e.g. 'good'), as well as the date on which the material in this kit was checked. the attribute idType is a foreign key refering to the relation **typeKits**.
personnels(<u>idPers</u>, nomPers, telPers, sportSpe)
Each member of the company has an id (idPers). We store his/her name (nomPers), her/his phone number and his/her sport of expertise.
resas(<u>idResa</u>, idInit, idPersResp, idPersAide, dateInit, idClient, idLieu)
This relationship memorizes both past and future reservations. A reservation has an identifier (idResa), it concerns a given initiation (idInit), is ensured by responsible staff (idPersResp), helped by another person (idPersAide). We also memorize the date and place where the initiation is to take place. (dateInit, idLieu) and the customer who made the reservation (idClient). In this relation, idInit, idPersResp, idPersAide, idLieu, et idClient are foreign keys referencing respectively **initiations**, **personnels**, **personnels**, **lieux**, **clients**.
emprunts(<u>idKit</u>, <u>idResa</u>, <u>idPers</u>, dateEmprunt, dateRetour, commentaire)
This relation memorizes kit borrowings and returns. A kit (idKit)is borrowed for a reservation (idResa), by a staff member (idPers) at a date (dateEmprunt). The attribute idKit (respectively, idResa, idPers) is the foreign key referencing **kits** (respectively, **resas** and **personnels**). the attribute dateRetour has an indefinite value until the kit is returned. The attribute commentaire can be left blank, or filled in when borrowing and/or returning. For example, a comment "all ok" when borrowing may be changed to "lamp problem" on return.
clients(<u>idClient</u>, nomClient, telClient)
A customer has an identifier, a name and a telephone number.
lieux(<u>idLieu</u>, adresse, lat, long, telContact)
A location has an identifier, and we store its postal address, latitude and longitude, as well as the contact number of a local person.

The description of the attribute type is omitted for clarity and has no impact on your work. You may consider that dateVerif, dateInit, dateEmprunt, dateRetour, are of the type Date ; that duree, prix, niveau, nbPers, lat and long are of type number ; and all the other attribute are of type varchar (e.g strings).

Question : Write the SQL queries corresponding to the following searches. Take care to eliminate duplicates when the result should not contain any. Use as few tables as possible for each query.

Indication : Dates will be written in the form 'DD-MM-YY'. For example, '25-11-24' for November 25, 2024.

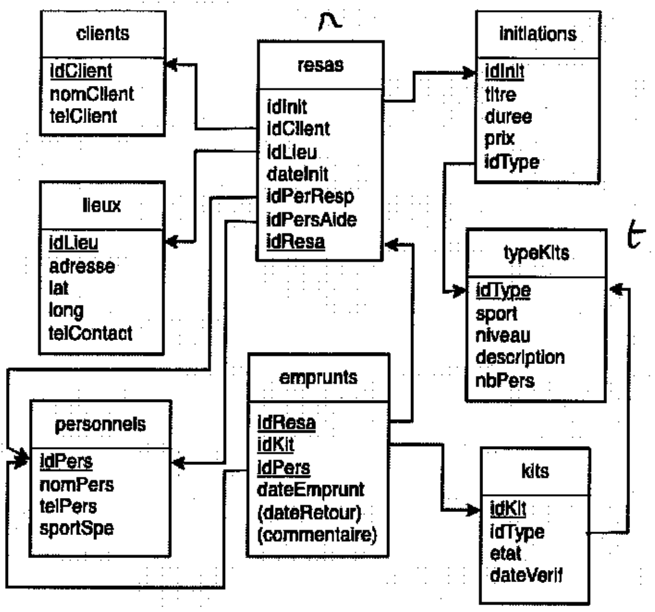

FIGURE 1 – Graphical representation of the database relational schema

1. Display all initiations booked for a date after December 1, 2024, specifying for each initiation the date on which it is scheduled, its title and the sport concerned. They will be sorted by sport (alphabetical order) and title (reverse alphabetical order). **Indication** : Use > to compare dates

2. How many kits were returned more than eight days after the date of initiation ? **Indication** : it is assumed that the expression `d + 8`, where `d` is a date, gives the date 8 days after `d`.

3. What are the reservations for customer ID 'C33Lyon', for which the person responsible for the reservation, or his/her assistant, has borrowed an equipment kit ? Display the reservation ID, the name of the person who made the reservation and the borrowing date.

4. How many kits of each sport and level in 'new' condition does the company have ?

5. Looking for initiations in the same sport as those scheduled for December 25, 2024 at the "LacAnnecy" ID location ? Display their title and level.

6. To secure certain initiations in case of malfunction, several kits, of same type, can be borrowed. Display the identifiers of reservations for which this is the case.

7. Among the kit types that have been the subject of at least one initiation, are there any that have never been the subject of a reservation at the 'LacAnnecy' ID location ? Display the identifier and sport of these kit types.

8. Knowing that a reservation is invoiced for the sum indicated in the `price` attribute only when the material has been returned, with a comment 'all ok', display the total sum paid by each customer since the reservations were saved. Display the customer's name and total amount.

Exercise 3 : Modeling (3 points)

The conceptual schema of figure 2 was designed on the basis of the following mini specifications :

Specifications

INSA students - all of whom are represented in the database - are largely housed in the school's residences. Each year, if they so request, the INSA management assigns them a room. They may be assigned the same room in different years. It is assumed that there is no change of assignment in the course of a year. All previous assignments remain stored in the database. Each residence is managed by a concierge. The residences' management wishes to store in the database the last concierge in charge of each residence. It can happen that a concierge is not responsible for any residence, or on the contrary, is responsible for several residences at the same time.

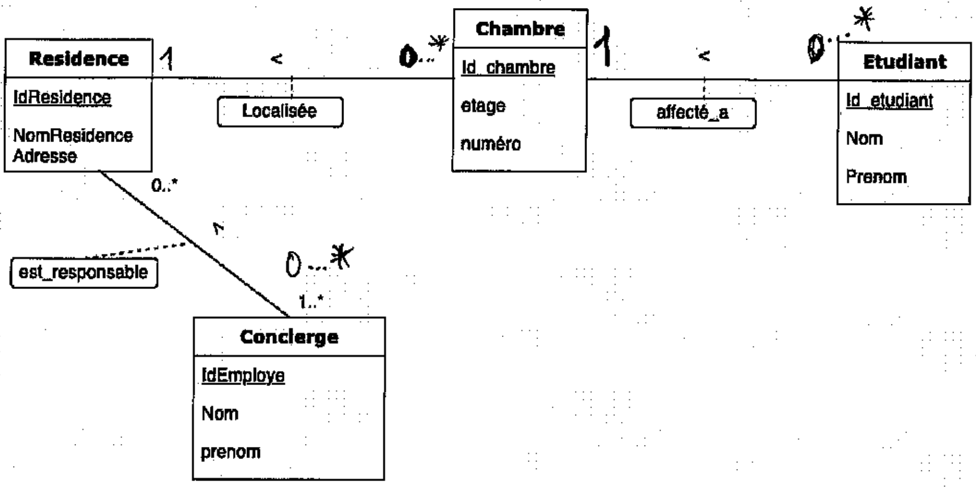

FIGURE 2 – Conceptual schema of the database "résidences"

(3.1) *Compare this diagram with the specifications provided, and complete and/or correct anything you feel doesn't match the specifications. Justify your proposals.*

(3.2) *Give the relational diagram corresponding to the completed/corrected conceptual diagram.*

Exercise 4 : Dictionaries (5 pts)

The Mendeleïev table (see figure 3) is a reference document used by chemists. It classifies chemical elements into groups, periods, etc.
We have a dictionary called `mendeleiev` which represents all the chemical elements in this table as follows :

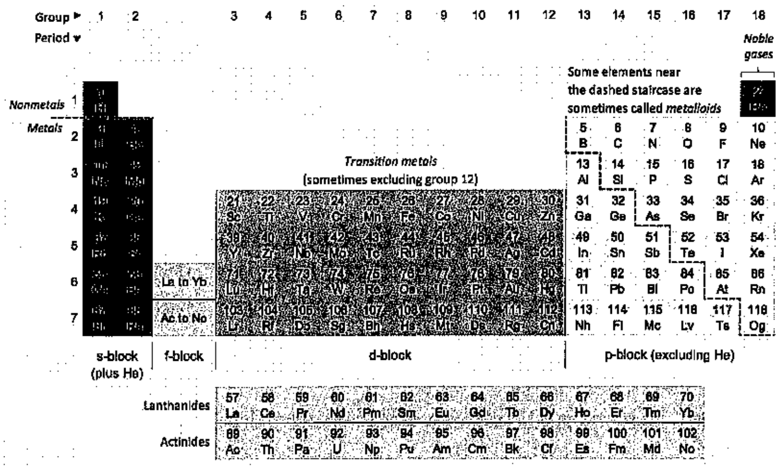
FIGURE 3 – Mendeleiev periodic table

— Dictionary keys are symbols for chemical elements. Examples of keys : 'O', 'Mn', 'Fe'.

— For each key (symbol of a chemical element), the value is a dictionary describing the chemical element.

Here is an extract from the dictionary `mendeleiev` for the chemical elements 'H' (Hydrogen) and 'He' (Helium) :

```
mendeleiev=
{
'H': {
    'atomic_number': 1,
    'name': 'Hydrogen',
    'atomic_mass': 1.008,
    'group': 1,
    'period': 1,
    'category': 'Nonmetals',
    'electron_configuration': '1s',
    'electronegativity': 2.2,
    'atomic_radius': 25.0
    },
'He': {
    'atomic_number': 2,
    'name': 'Helium',
    'atomic_mass': 4.002602,
    'group': 18,
    'period': 1,
    'category': 'Noble gases',
    'electron_configuration': '1s2',
    'electronegativity': None,
    'atomic_radius': 120.0},
    ...
}
```

Note : In this extract, hydrogen belongs to group 1. Helium belongs to group 18.

(4.1) Write a Python function to count the number of chemical elements in a group given as a parameter.

We propose to represent a molecule by a dictionary giving its structure. For example, the molecule $KMnO_4$ is represented by the following dictionary :

```
{
"nom" : "permanganate de potassium",
"composition":{
   'K': 1,
    'Mn': 1 ,
    'O'  : 4
    }
}
```

(4.2) *Write a function `afficher_molecule` to display the name of the molecule, its chemical formula in parenthesis, and the total number of atoms.
For instance, for the molecule '$KMnO_4$', the function displays :
permanganate de potassium($KMnO4$) -> 6 atomes*

We have a list of molecules `liste_mol`, where each molecule is represented by a dictionary similar to the previous example.

4.1 5

(4.3) *Write a function `extrait_liste_mol_contenant(list_mol, at, min_nb)` to create, from a list of molecules given as a parameter, a new list of molecules in which only molecules with a minimum number 'min_nb' (given as a parameter) of a chemical element 'at' also given as a parameter are present.*

(4.4) *Your solution to question (4.3) includes a test to determine whether a molecule should be added to the new list. Estimate the number of times this test is run.*

It is assumed that the function `extrait_liste_mol_contenant(list_mol, at, min_nb)` is used <u>a</u> <u>lot</u>.

(4.5) *Propose a structure that would reduce the overall number of tests performed. Justify your answer by comparing the cost of the proposed solution with the estimated cost of the question. (4.4).*

We now wish to :

— calculate the total mass of a molecule

— Add mass as a new entry in the dictionary representing a molecule.

Reminder : the mass of each chemical element is given in the dictionary. `mendeleev`.

4.1 R

(4.6) *Write the function `calculer_et_mem_masse(molecule,mendeleev)` which creates a new "mass" key for a molecule, with the molecule's mass as its value.*

We would like to complete the following main program for :

— print the number of chemical elements in group 18,

— display all molecules in the list `liste_mol`,

— extract the list of molecules with at least one oxygen atom and

— calculate and store the molecular weight of each molecule in the list `liste_mol`

```
   #main program

with open('dictionary_mendeleiev_by_symbol.json','r', encoding='utf_8') as mon_fichier:
   contenu=json.load(mon_fichier)

print(contenu)
liste_mol = [{"nom" : "permanganate de potassium",
             "composition":{
'K': 1,
'Mn': 1 ,
'O'  : 4

}},{"nom" : "potassium ferroCyanide",
  "composition":{
    'K': 4,
 'Fe': 1 ,
 'Cn'  : 6

 }},{"nom" : "Sulfate de potassium",
     "composition": {
  'K': 2,
  'S': 1 ,
  'O'  : 4

  }}]
```

(4.7) Complete the main program.