Evaluation sur les BDD / POO – Nov. 2025

Durée : 1h30

Exercice 1 : POO (6 points)

1/ **Ecrire** une classe **Eleve** comportant comme attributs, le *nom*, le *prénom*, la *classe* et une *liste de notes.*

2/ **Créer** une instance appelée *eleve1* de la classe précédente ayant pour attributs respectifs *Durand*, *Paul* et deux notes de *14* et *15*.

3/ **Créer** une instance appelée *eleve2* de la classe précédente ayant pour attributs respectifs *Dupond*, *Pierre* et deux notes de *13* et *16*.

4/ **Ecrire** une méthode *affiche(self)* qui donne le nom, prénom et les notes obtenues. **Tester** avec *eleve1*.

5/ **Ecrire** une méthode *ajoute(self, note)* qui ajoute une note à l'élève. **Ajouter** alors la note de 18 à l'objet *eleve1*. **Proposer** un test de validité de la note ajoutée.

6/ **Ecrire** une fonction *compare(elev1, elev2)* qui renvoie le nom, prénom et la meilleure note entre deux élèves. Si ces derniers ont la même meilleure note, elle affichera « égalité ».

Exemple : *compare(eleve1, eleve2)* renvoie Dupond, Pierre, 16 avec les instances des questions 2/ et 3/.

Exercice 2 : SGBD (14 points)

Principaux thèmes abordés : bases de données (modèle relationnel, base de données relationnelle et langage SQL).

Dans notre monde, l'information a de plus en plus de valeur et d'importance mais nous sommes de plus en plus confrontés à l'infobésité.

Considérons l'utilisation des données issues de la table de Mendeleïev (tableau périodique des éléments). Il est contraignant de faire des recherches sur des moteurs dédiés à chaque fois qu'une valeur est nécessaire (masse volumique, rayon de covalence, point de fusion…).

Les lignes 3, 4 et 5 de cette table Mendeleïev ont permis de construire, en **annexe 1 de l'exercice 3,** une base de données des différents atomes correspondants.

1. Donner le nom du langage informatique utilisé pour accéder aux données dans une base de données ?
2. a) Lister les différents attributs des tables ATOMES et VALENCE en précisant le type du domaine de chacun.

   b) Déterminer si des attributs de la table ATOMES peuvent avoir un rôle de clé primaire et/ou de clé étrangère. Justifier.

   c) Donner le schéma relationnel pour les deux tables ATOMES et VALENCE.

3. Donner les réponses des deux requêtes suivantes :

a) `SELECT nom FROM ATOMES WHERE L='3' ORDER BY Sym`

b) `SELECT DISTINCT C FROM ATOMES`

4. Donner la requête SQL:

a) Pour afficher le nom et la masse atomique des atomes.

b) Pour afficher le symbole des atomes dont la couche de valence est s.

5. On a remarqué une erreur de saisie dans la table ATOMES, la masse atomique de l'argon (Ar) n'est pas 29,948 $g.mol^{-1}$ mais 39,948 $g.mol^{-1}$. Écrire la requête SQL pour corriger cette erreur de saisie.

ANNEXE

Relation « ATOMES »

Z	nom	Sym	L	C	masse_atom
11	sodium	Na	3	1	22,9897693
12	magnesium	Mg	3	2	24,305
13	aluminium	Al	3	13	26,9815386
14	silicium	Si	3	14	28,0855
15	phosphore	P	3	15	30,973762
16	soufre	S	3	16	32,065
17	chlore	Cl	3	17	35,453
18	argon	Ar	3	18	**29,948**
19	potassium	K	4	1	39,0983
20	calcium	Ca	4	2	40,078
21	scandium	Sc	4	3	44,955912
22	titane	Ti	4	4	47,867
23	vanadium	V	4	5	50,9415
24	chrome	Cr	4	6	51,9961
25	manganese	Mn	4	7	54,938045
26	fer	Fe	4	8	55,845
27	cobalt	Co	4	9	58,933195
28	nickel	Ni	4	10	58,6934
29	cuivre	Cu	4	11	63,546
30	zinc	Zn	4	12	65,409
31	gallium	Ga	4	13	69,723
32	germanium	Ge	4	14	72,64
33	arsenic	As	4	15	74,9216
34	selenium	Se	4	16	78,96
35	brome	Br	4	17	79,904
36	krypton	Kr	4	18	83,798
37	rubidium	Rb	5	1	85,4678
38	strontium	Sr	5	2	87,62
39	yttrium	Y	5	3	88,90585
40	zirconium	Zr	5	4	91,224
41	niobium	Nb	5	5	92,90638
42	molybdene	Mo	5	6	95,94
43	technetium	Tc	5	7	98
44	ruthenium	Ru	5	8	101,07
45	rhodium	Rh	5	9	102,9055
46	palladium	Pd	5	10	106,42
47	argent	Ag	5	11	107,8682
48	cadmium	Cd	5	12	112,411
49	indium	In	5	13	114,818
50	etain	Sn	5	14	118,71
51	Antimoine	Sb	5	15	121,76
52	Tellure	Te	5	16	127,6
53	Iode	I	5	17	126,90447
54	Xenon	Xe	5	18	131,293

Relation « VALENCE »

Col	Couche
1	s
2	s
3	d
4	d
5	d
6	d
7	d
8	d
9	d
10	d
11	d
12	d
13	p
14	p
15	p
16	p
17	p
18	p

Z : Numéro atomique ;
Sym : symbole ;
L : lignes ;
C ou Col : Colonne ;
Couche : Couche de valence

Corrigé

Exercice 1 :

```python
class Eleve:
    def __init__(self, nom: str, prenom: str, classe: str, notes: list):
        self.nom = nom
        self.prenom = prenom
        self.classe = classe
        self.notes = notes

    def affiche(self) -> None:
        """Affiche nom, prénom et la liste des notes."""
        print(f"{self.nom}, {self.prenom} - notes : {self.notes}")

    def ajoute(self, note: float) -> None:
        """Ajoute une note comprise entre 0 et 20."""
        assert 0 <= note <= 20, "La note doit être comprise entre 0 et 20."
        self.notes.append(float(note))

    def meilleure_note(self) -> float:
        """Retourne la meilleure note de l'élève."""
        return max(self.notes)
```

```python
def compare(elev1: Eleve, elev2: Eleve):
    """Compare deux élèves et renvoie celui qui a la meilleure note."""
    n1 = elev1.meilleure_note()
    n2 = elev2.meilleure_note()
    if n1 > n2:
        return (elev1.nom, elev1.prenom, n1)
    elif n2 > n1:
        return (elev2.nom, elev2.prenom, n2)
    else:
        return "égalité"
```

```python
# 2/ Création de eleve1
eleve1 = Eleve("Durand", "Paul", "Terminale", [14, 15])

# 3/ Création de eleve2
eleve2 = Eleve("Dupond", "Pierre", "Terminale", [13, 16])

# 4/ Test de la méthode affiche avec eleve1
eleve1.affiche()   # -> Durand, Paul - notes : [14.0, 15.0]

# 5/ Ajout de la note 18 à eleve1
eleve1.ajoute(18)
eleve1.affiche()   # -> Durand, Paul - notes : [14.0, 15.0, 18.0]
```

```python
# 6/ Test de compare
res = compare(eleve1, eleve2)
print(res)  # ('Durand', 'Paul', 18.0)
```

Exercice 2 :

1

Pour effectuer des requêtes sur une base de données relationnelle, on utilise le langage SQL

2a

ATOME (Z : INT, nom : TEXT, Sym : TEXT, L : INT, C : INT, masse_atom : FLOAT)
VALENCE (Col : INT, Couche : TEXT)

2b

l'attribut Z peut jouer le rôle de clé primaire car il existe un Z unique pour chaque élément chimique.
l'attribut C va jouer le rôle de clé étrangère car cet attribut va permettre d'établir une "liaison" avec l'attribut Col de la table VALENCE

2c

ATOME (Z : INT, nom : TEXT, Sym : TEXT, L : INT, #C : INT, masse_atom : FLOAT)
VALENCE (Col : INT, Couche : TEXT)

3a

On obtient la liste de nom d'atomes suivante :
aluminium, argon, chlore, magnesium, sodium, phosphore, soufre, silicium

3b

On obtient la liste des colonnes :
1, 2 ,3 ,4 ,5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18

4a

```
SELECT nom, masse_atom
FROM ATOMES
```

4b

```
SELECT Sym
FROM ATOMES
INNER JOIN VALENCE ON ATOMES.C = VALENCE.Col
WHERE Couche = 's'
```

5

```
UPDATE ATOMES
SET mass_atom = 39.948
WHERE nom = 'argon'
```