

[Index des sujets 2023](#)

## 23-NSIJ1ME1 : Corrigé

Année : **2023**

Centre : **Métropole**

Jour : **J1**

Enoncé : [!\[\]\(e3f8612927870f2e0f9f5989e6dd3064\_img.jpg\) PDF](#)

### 1. Exercice 1 (3 points)

SQL

1. a. Pour qu'un attribut soit choisi comme clé primaire, il doit être unique.
  - b. Les deux clés étrangères de la relation `Commandes` font référence à la clé primaire d'une autre table. Elles permettent de mettre en relation un enregistrement de cette table avec un enregistrement d'une autre table. ici :
    - la clé étrangère `idClient` met en relation avec la table `Clients` et sa clé primaire `id`
    - la clé étrangère `idMeuble` met en relation avec la table `Meuble` et sa clé primaire `id`
  - c. `Meubles (id : INT, intitule : VARCHAR, prix : FLOAT, stock : INT, description : VARCHAR)`
  2. La requête renvoie :
- 62 - 2 - Armoire blanche 3 portes  
63 - 3 - Armoire noire 3 portes
- 3.

#### Requête SQL

```
SELECT nom, prenom
FROM Clients
WHERE ville = "Paris"
```

1.  Requête SQL

```
UPDATE Meubles
SET stock = 50
WHERE intitule = "hylla"
```

2.  Requête SQL

```
INSERT INTO Meubles
(id, intitule, prix, stock, description)
```

## VALUES

(65, "matta", 95.99, 25, "Tapis vert à pois rouges")

3.  Requête SQL

```
SELECT Clients.nom, Clients.prenom
FROM Clients
JOIN Commandes ON Commandes.idclient = Clients.id
WHERE Commandes.date ="30/04/2021"
```

**2. Exercice 2 (3 points)***Routage*

1. Le réseau est construit avec de multiples liaisons et de telle sorte qu'un chemin sera toujours possible entre 2 sites même si une liaison est coupée.  
Si on coupe une liaison, chaque site pourra encore communiquer avec les 3 autres.
2. Site B - R2 - R3 - R4 - R5 - Site C
- 3.

**Table Routeur R1**

DESTINATION	SUIVANT	Nombre de SAUTS
site A	Local	0
site B	R2	1
site C	R2	2
site D	R3	2

1. Dans le protocole RIP de nombreux chemins passent par la liaison 2, donc si elle a un débit très inférieur aux autres il n'est pas judicieux de l'utiliser. Le choix du protocole RIP n'est donc pas judicieux ici.
  2. a. On cherche la liaison dont le débit est le plus faible ce qui revient à chercher le coût est le plus élevé :
  3. La liaison 2 a le coût le plus élevé donc le débit le plus faible :  $D = \frac{10^{10}}{10^4} = 10 \text{ kbit/s.}$
- b. - Site A → R1 → R2 → R5 → Site C coût : 1 100 000  
- Site 1 → R1 → R3 → R4 → R5 → Site C coût : 50 015  
- Site A → R1 → R3 → R2 → R5 → Site C coût : 1 100 005  
- Site A → R1 → R2 → R3 → R4 → R5 → Site Coût : 100 020
- c. Table Routeur R1

DESTINATION	SUIVANT	COUT
site A	Local	0
site B	R3	50005
site C	R3	50015
site D	R3	50005

### 3. Exercice 3 (6 points)

Programmation et POO

#### Partie 1

- 1.
2. nom : attribut
3. tab\_voisines : attribut
4. tab\_couleurs\_disponibles : attribut
5. nom\_region est de type str : chaîne de caractère.
6. ge = Region ("Grand Est")

#### 7. Script Python

```
def renvoie_premiere_couleur_disponible(self):
    return self.tab_couleurs_disponibles[0]
```

- 8.

#### Script Python

```
def renvoie_nb_voisines(self) :
    return len(self.tab_voisines)
```

#### 1. Script Python

```
def est_coloriee(self):
    if self.couleur_attribuee == None :
        return False
    else :
        return True
```

#### 2. Script Python

```
def retire_couleur(self, couleur):
    if couleur in self.tab_couleurs_disponibles:
        self.tab_couleurs_disponibles.remove(couleur)
```

3.  Script Python

```
def est_voisine(self, region):
    for i in range(len(self.tab_voisines)) :
        if region == self.tab_voisines[i] :
            return True
    return False
```

4.  Script Python

```
def renvoie_tab_regions_non_coloriees(self):
    L=[]
    for region in self.tab_regions :
        if est_coloriee(region) == False :
            L.append(region)
    return L
```

5. a. La méthode renvoie `None` dans le cas où tout est colorié.

b. La région renvoyée est la région qui a le plus de voisines parmi celles qui ne sont pas colorierées.

6.  Script Python

```
def colorie(self):
    region_m = self.renvoie_max()
    while region_m:
        region_m.couleur_attribuee = region_m.renvoie_premiere_couleur_disponible()
        for voisine in region_m.tab_voisines:
            voisine.retire_couleur(region_m.couleur_attribuee)
    region_m = self.renvoie_max()
```