

Asie - 2026 - sujet1 - Correction

Exercice 1 (6 points)

Partie A - Base de données

1. Ajout de la chorégraphie « Tout autour »

```
INSERT INTO choregraphie  
VALUES ('Tout autour', 2015, 9);
```

👉 *Commentaire : les valeurs suivent l'ordre des attributs de la table : nomchore, annee, choregraphie. On peut aussi les préciser : INSERT INTO choregraphie (nomchore, annee, choregraphie) VALUES (...).*

2. Nombre de chorégraphies de 1983

```
SELECT COUNT(*)  
FROM choregraphie  
WHERE annee = 1983;
```

3. Résultat de la requête

La requête affiche, pour **chaque chorégraphie**, son nom et le **nom de son chorégraphe** (la personne dont l'identifiant est la valeur de l'attribut choregraphe).

4. Danseurs du spectacle « Tout autour »

```
SELECT DISTINCT personne.nom  
FROM personne  
JOIN participe ON personne.id = participe.idpersonne  
JOIN spectacle ON participe.idspectacle = spectacle.id  
WHERE spectacle.titre = 'Tout autour';
```

👉 *Commentaire : deux jointures : personne → participe, puis participe → spectacle. DISTINCT évite d'afficher plusieurs fois un danseur qui danse plusieurs chorégraphies dans ce spectacle (ou qui a dansé lors de plusieurs représentations).*

5. Supprimer un spectacle

La table participe contient la clé étrangère idspectacle, qui référence spectacle. Il faut d'abord **supprimer les lignes de participe** qui concernent ce spectacle, puis le spectacle lui-même ; sinon des participations feraient référence à un spectacle inexistant et le SGBD refuserait la suppression (contrainte d'intégrité référentielle).

Partie B – Chargement du véhicule

6. Ligne 3

```
n = 0
for clef in dmateriel:
    n += dmateriel[clef]['nb']
```

👉 *Commentaire : `dmateriel[clef]` est le sous-dictionnaire du matériel `clef` ; on y lit le nombre d'éléments avec la clé `'nb'`.*

7. Lignes 5 et 6

```
for clef, dico in dmatos.items():
    tab_clefs.append(clef)          # ligne 5
    tab_volumes.append(dico['volume']) # ligne 6
```

👉 *Commentaire : `items()` donne les couples (clé, valeur) : `dico` est le sous-dictionnaire associé à `clef`.*

8. Type d'algorithme

C'est un **algorithme glouton** : à chaque étape, on fait le choix qui paraît le meilleur sur le moment (le matériel le plus volumineux qui rentre encore), sans revenir sur les choix précédents.

👉 *Commentaire : un algorithme glouton ne donne pas toujours le remplissage optimal, mais il est simple et rapide.*

9. Un algorithme de tri

Par exemple, le **tri par insertion** (ou le tri par sélection) a un coût **quadratique**, de l'ordre de n^2 dans le pire des cas.

Le **tri fusion** a un coût de l'ordre de $n \log n$.

10. Fonction remplissage (lignes 6 et 11 à 15)

```
def remplissage(dmatos, vmax):
    vol = 0
    clefs_triees = clefs_triees_selon_volume(dmatos)
    i = len(clefs_triees)-1
    elements = []
    while i >= 0 and vol < vmax:          # ligne 6
        clef = clefs_triees[i]
        if dmatos[clef]['nb'] == 0:
            i -= 1
        elif vol + dmatos[clef]['volume'] <= vmax:
            vol += dmatos[clef]['volume'] # ligne 11
```

```

        elements.append(clef)                                # ligne 12
        dmatos[clef]['nb'] -= 1
    else:
        i -= 1                                                # ligne 15
return elements

```

👉 *Commentaire : on parcourt les clés de la plus volumineuse (indice $\text{len} - 1$) à la moins volumineuse (indice 0). Si un élément ne rentre plus, on passe au matériel suivant, plus petit (ligne 15). On vérifie : remplissage(dmatériel, 160) renvoie bien ['chaise_bois', 'chaise_bois', 'chaise_bois', 'chapeau_B', 'chapeau_A'] ($50 + 50 + 50 + 7 + 3 = 160$).*

11. Fonction nb_voitures

```

def nb_voitures(dmatos, vmax):
    n = 0
    while remplissage(dmatos, vmax) != []:
        n = n + 1
    return n

```

👉 *Commentaire : chaque appel de remplissage charge une voiture et retire ce matériel de dmatos ; on s'arrête quand plus rien n'est chargé. Avec dmatériel et $\text{vmax} = 160$, il faut 14 voitures. Attention : si un élément a un volume supérieur à vmax , il n'est jamais chargé et la fonction s'arrête sans l'avoir compté.*