

DS : Requêtes SQL / Dictionnaires

1 Exercice 1 : Tables de hachage

On propose ici un exemple simple de pré-traitement des chaînes de caractères avant hachage.

Nous allons nous servir du codage ASCII qui est donné sous Python par la fonction `ord(c)`, qui renvoie le code ASCII sur 8 bits du caractère c .

Inversement, l'instruction `chr(code)` renvoie le caractère associé à un code ASCII.

On souhaite transformer une chaîne de caractères en un entier, puis utiliser cet entier pour déterminer une case dans une table de hachage.

1. Transformer une chaîne en entier

On considère une chaîne de caractères ch de longueur n .

On souhaite lui associer l'entier :

$$f(ch) = \sum_{k=0}^{n-1} \text{ord}(ch[k]) 256^k.$$

Écrire une fonction Python :

```
def chaineversentier(ch):  
    ...
```

qui prend une chaîne de caractères ch en argument et renvoie l'entier défini ci-dessus.

2. Calculer la valeur associée à une chaîne

On considère la chaîne :

```
ch = 'ABC'
```

On donne :

$$\text{ord}('A') = 65, \quad \text{ord}('B') = 66, \quad \text{ord}('C') = 67.$$

Calculer à la main la valeur de `chaineversentier('ABC')`.

3. Construire la fonction de hachage

On souhaite réserver une table de hachage de longueur m .

La fonction de hachage associée à une chaîne ch le reste de la division euclidienne de `chaineversentier(ch)` par m .

Écrire une fonction Python :

```
def chaineversentiermodulo(ch, m):  
    ...
```

qui prend une chaîne ch et un entier m en arguments et renvoie le numéro de la case de la table associée à ch .

Préciser l'intervalle dans lequel se trouve nécessairement la valeur renvoyée par cette fonction.

4. Étudier une petite table de hachage

On considère les chaînes suivantes :

"AB"	"AC"	"AD"	"BA"
"BB"	"BC"	"CA"	"CB"
"CC"	"DA"	"DB"	"DC"

et une table de hachage de longueur $m = 4$.

On donne :

$$256 \equiv 0 \pmod{4}.$$

- Trouver la valeur de `chaineversentiermodulo(ch, 4)` pour chacune des douze chaînes.
- Représenter la répartition des chaînes dans les quatre cases de la table sous la forme d'un dictionnaire Python :

```
{  
    numero_de_case : nombre_de_chaines  
}
```

5. Générer des mots et étudier leur répartition

On souhaite maintenant étudier expérimentalement la répartition des valeurs de hachage.

On rappelle que `randint(a,b)` renvoie un entier aléatoire compris entre a et b , bornes incluses, et que les codes ASCII des lettres majuscules vont de 65 ('A') à 90 ('Z').

Écrire une fonction *repartition*(N, m) qui doit :

- générer aléatoirement N mots de longueur exactement 7;
- choisir chaque caractère parmi les lettres majuscules de l'alphabet;
- déterminer, pour chaque mot, la case dans laquelle il est placé à l'aide de `chaineversentiermodulo`;
- compter, à l'aide d'un dictionnaire, le nombre de mots placés dans chaque case;
- renvoyer ce dictionnaire.

On pourra vérifier que la somme des valeurs du dictionnaire est égale à N .

6. Tester l'uniformité

On souhaite déterminer si la distribution obtenue à la question précédente est proche d'une distribution uniforme.

On considère $N = 1000$ mots et une table de hachage de longueur $m = 12$.

- (a) Dans le cas d'une distribution parfaitement uniforme, l'effectif théorique d'une case est donné par :

$$\frac{N}{m}.$$

Calculer cet effectif théorique.

- (b) Expliquer comment utiliser le dictionnaire renvoyé par **repartition(1000,12)** pour déterminer si la distribution obtenue est proche d'une distribution uniforme.
- (c) On définit l'**écart maximal** comme le plus grand des écarts absolus entre le nombre de mots présents dans une case et l'effectif théorique.

Écrire une fonction :

```
def ecart_maximal(repartition, N, m):  
    ...
```

qui prend en paramètres le dictionnaire obtenu précédemment, le nombre de mots N et la taille m de la table, et qui renvoie l'écart maximal.

2 Exercice 2 : Requêtes SQL

On considère une base de données permettant de gérer des étudiants, des formations et les inscriptions des étudiants aux formations.

Structure de la base

Table ETUDIANT

NumEtudiant	Nom	Prenom	Ville
1	Martin	Alice	Marseille
2	Dupont	Hugo	Lyon
3	Bernard	Emma	Marseille
4	Petit	Lucas	Nice
5	Robert	Chloé	Lyon

Table FORMATION

NumFormation	Intitule	Responsable	Ville
101	Informatique	Durand	Marseille
102	Mathématiques	Morel	Lyon
103	Physique	Durand	Nice

Table INSCRIPTION

NumEtudiant	NumFormation	Annee	Note
1	101	2025	15
1	102	2025	12
2	101	2025	14
2	103	2025	11
3	101	2025	17
3	102	2025	16
4	103	2025	13
5	102	2025	10

1. Afficher le nom, le prénom et la ville de tous les étudiants.
2. Afficher le nom et le prénom des étudiants habitant à Marseille.
3. Afficher les différentes villes dans lesquelles sont situées les formations, sans répétition.
4. Afficher le nom et le prénom de chaque étudiant ainsi que sa note obtenue. La requête doit utiliser les tables ETUDIANT et INSCRIPTION.
5. Afficher le nom et le prénom des étudiants ayant obtenu une note supérieure ou égale à 15. La requête doit utiliser les tables ETUDIANT et INSCRIPTION.
6. Afficher le nom de l'étudiant et le numéro de la formation pour chaque inscription. La requête doit utiliser uniquement les tables ETUDIANT et INSCRIPTION.
7. Afficher, pour chaque étudiant, son numéro, son nom et sa moyenne obtenue aux différentes formations auxquelles il est inscrit. La requête doit utiliser uniquement les tables ETUDIANT et INSCRIPTION.