# Base de données par TAG

L’objectif est de créé une application permettant à l’utilisateur de trier ses fichiers et lieu externe via des mots-clés. Un objet ou éléments de la base pourra avoir plusieurs tags qui lui seront accorder.

Pour une question d’ergonomie, le moteur sera sur une page HTML et fera la recherche en JS.

## Description de l’objet :

Un objet a plusieurs éléments qui lui son attribué :

1. Un nom
2. Une description
3. Type (image, vidéo, etc)
4. Un ID
5. Un emplacement (interne ou externe)

Les TAG ne sont pas stocké dans l’objet mais dans un fichier spécifique.

## Création de l’ID :

L’ID est un élément important car il doit être unique à chaque objet car deux objets peuvent avoir un nom en commun (car donner par l’utilisateur).

L’ID est constituer d’un préfixe et d’un suffixe.

Le préfixe sert à identifier le type de l’objet (ex : 1 pour les images et 2 pour les vidéos).

Le suffixe est une série de chiffre aléatoire. Le problème à résoudre et si par hasard (même infime sois t’il) un suffixe est identique à un autre comment ne pas tomber sur ce cas sans avoir à regénérer un suffixe. Ce problème peut arriver dans des bases avec un nombres d’objets très important.

### Probabilité de 2 suffixes identique

#### Paradoxe anniversaires

Si tu appelles la fonction **n fois**, quelle est la probabilité d’avoir **au moins un doublon** parmi les valeurs générées ?

C’est une variation du **paradoxe des anniversaires** :

Où :

* n est le nombre d’échantillons (appels de la fonction),
* N= 9 \* 1015 est le nombre total de valeurs possibles.

#### Conclusion

Pour que deux nombres de 16 chiffres sois identique il y a 0.00000000000001111 %

|  |  |
| --- | --- |
| **n objet** | **Proba d’au moins une collision** |
| 10 | 4.55\*10-15 (1 sur 2.2\*1014) |
| 1 000 | 4.99\*10-11 (1 sur 2.0\*1010) |
| 1 000 000 | 5.00\*10-5 (1 sur 20 000) |

## Recherche par TAG

La recherche se fait sois par nom (résultat des noms en commun) sois par tag.

Un TAG est un mot ou une chaine de caractère séparer par des « \_ » avec « # » comme préfixe afin de les différencier des noms. Lorsque l’utilisateur entre un tag, le moteur sais qu’il doit chercher un fichier qui a pour nom le tag en question sans le préfixe, dans celui si ce trouve les id des objets relier à ce tag. Si un mot a pour préfixe un « \_ » cela signifie que c’est un type. Si le type n’est pas préciser alors tous les type sont pris en compte.

### La recherche multiple

La recherche multiple est le cas ou l’utilisateur entre plusieurs mots dans le moteur de recherche (nom et tag).

Il y a plusieur moyen de les séparer :

1. Par un espace, qui signifiera que le ou les objets chercher doivent avoir apparaitre dans les différents fichiers qui toque les objet (si un nom a était entré) sois dans les fichiers de tag, ce qui a pour but de réduire de nombres de résultat
2. Par un « and », cela signifie que l’objet doit avoir obligatoirement les deux mots (tag ou nom) relier a lui.
3. Par un « OR », cela signifie que la recherche est élargie et que l’on prend les résultats les deux mots en même temps ce qui augment de nombre de résultat
4. Par « , » même cas que pour « OR »
5. S’il est dans une fonction not alors le mot est exclu, le résultat de la recherche ne prendra que de objets qui ne sont pas attacher à ce mot : not(mot)
6. Cas particulier « XOR » on prend les résultats d’un « AND » et d’un « OR » mais on retourne en premier les résultats du « AND », puis suivras les résultats du « OR »
7. Si deux mots sont collés par un « : » cela signifie que le premier et un dossier et l’autre un tag (obligatoirement un tag), mais le second peut-être un regroupement dans une parentaise. (Exemple : ville:(paris , limoges) le résultat renverra les objets qui sont dans ville et de tag « paris » ou « limoges ») cala permet de mieux trier les tags et de permettre plus de recherche.

On peut regrouper les mots de recherche dans des parentaises afin de mieux préciser ce que l’on veut. Si on ne précise pas le dossier ou sont ranger les fichiers de tag alors on prend en compte le tag même si le moteur doit aller le chercher dans un dossier.

Exemple

|  |
| --- |
| (image OR vidéo) #drole |
| Le résultat renvoi les images et les vidéos avec le tag #drole |

Autre

|  |
| --- |
| (vacances OR weekend) (#montagne not(#plage OR #parc)) |
| Le résultat de la recherche renvoi tout type de fichier qui dans leur nom « vacances » ou « weekend » et qui sont relier au tag « montagne » mais qui ne sont pas relier au tague « plage » ou « parc » |