

# **Rapport TP4**

# Architecture Base de données

# **Etudiants:**

BIBI thilleli KHEMSINE Lydia thillelibibi@gmail.com lydiakhemsine@outlook.com

# **Structure de TP4:**

notre projet JAVA est structuré en 3 classe :

## 1) TablesandBlocks.java:

#### -create\_relation

fonction qui prend en paramètre un entier nbr\_lignesR et retourne un tableau de string contenant des couples de lettres de l'alphabet en faisant le produit cartésiens de l'alphabet avec quelques lettres qui sont bien définies , puis on applique la fonction shuffle sur la liste afin de générer des couples aléatoires.

#### -make block

fonction qui prend en paramètre une liste de String, 2 paramètres pour les numéros des blocs et retourne un tableau des éléments des blocks en faisant la concaténation en code ASCII des lettres.

#### -all blocks

fonction qui appelle la fonction make\_block pour construire les blocks et les met dans airetable en appelant la fonction block\_to\_airtable de la classe airTable.java.

# 2) air table.java

## -block\_to\_airtable

fonction qui utilise une requete http (POST) pour mettre les blocs dans airtable, le code est le suivant :

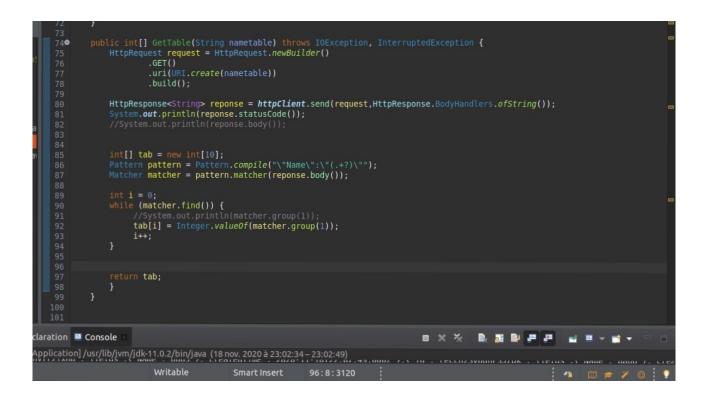
```
ton (nechectene . ver ston . mile_ 1 1)
       public static void block to airtable(int[] block, String blockname) throws IOException, InterruptedException {
60
            StringBuilder json = new StringBuilder() ;
json.append("{ \"records\": [");
                  for (int i=0;i < 9; i++) {
    json.append(" { \"fields\": { \"Name\":\"" + block[i] + "\" }},").toString();</pre>
                 json.append(" { \"fields\": { \"Name\":\"" + block[9] + "\"}}").toString();
json.append("]}");
                  String jsonString = json.toString();
                 HttpResponse response;
                  HttpRequest request = HttpRequest.newBuilder()
                                     .POST(HttpRequest.BodyPublishers.ofString(jsonString))
                                     .setHeader("User-Agent", "Java 11 HttpClient Bot") // add request header
.header("Content-Type", "application/json")
.build();
                  response = httpClient.send(request, HttpResponse.BodyHandlers.ofString());
                  System.out.println(response);
System.out.println(response.body());
                                                                                             m w x
                                                                                                           on 📮 Console
ation] /usr/lib/jvm/jdk-11.0.2/bin/java (18 nov. 2020 à 23:02:34 – 23:02:49)
```

#### -tabledescriptor :

fonction qui sert à mettre les descripteur dans airtable en utilisant la requête POST HTTP.

#### -GetTable :

fonction utilisée par la classe NestedLoop pour récupérer les blocs de airtable et faire la jointure en utilisant la requête GET de HTTP. Après on parse le résultat de la requête pour avoir le nom des blocks. Le code est le suivant :



#### 3) NestedLoop.java

#### -nestedloop :

fonction qui sert à faire la jointure entre les blocs de R et de S, le code de la jointure est le suivant :

```
while(lenr<Urir.length){
42
43
44
45
46
47
48
50
51
52
53
54
55
56
66
67
68
69
70
                     Blockr = airtable.GetTable(Urir[lenr]);
                      while(lens<Uris.length) {
    Blocks = airtable.GetTable(Uris[lens]);</pre>
                          int i=0;in1
                           int i=0;int j ;
while(i<Blockr.length) {
                                felse if (k2<10) {
    result2[k2] = Blockr[i];</pre>
                                         else if (k3<10) {
    result3[k3] = Blockr[i];
                                                k3++;
                                              e if (k4<10) {
result4[k4] = Blockr[i];
                                                k4++;
                                                                                                   ■ × %
tion 📃 Console
                                                                                                                 ication] /usr/lib/jvm/jdk-11.0.2/bin/java (18 nov. 2020 à 23:02:34 – 23:02:49)
                          Writable
```

# Les liens des bases :

https://airtable.com/shrVJ85vp083bMJVvhttps://airtable.com/shrmRPMtcbVv0CnxE

# **Exécution:**

Faut exécuter la classe TableandBlocks et aprés exécuter la classe NestedLoop pour faire la jointure.