République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie Houari Boumédiène

Faculté d'Informatique
Département d'Intelligence Artificielle et Sciences de Données

Master Systèmes Informatiques intelligents

Module : Base de données avancées

Rapport de projet de TP

Réalisé par :
BOUROUINA Rania, 181831052716
CHIBANE Ilies, 181831072041

Année universitaire : 2021 / 2022

1 Partie I : Importation de la BD

Pour commencer, il faudrait d'abord installer MongoDB (comme nous l'avons fait en TP), et faire les configurations nécessaires en rajoutant une variable d'environnement et en créant les fichiers de données nécessaires.

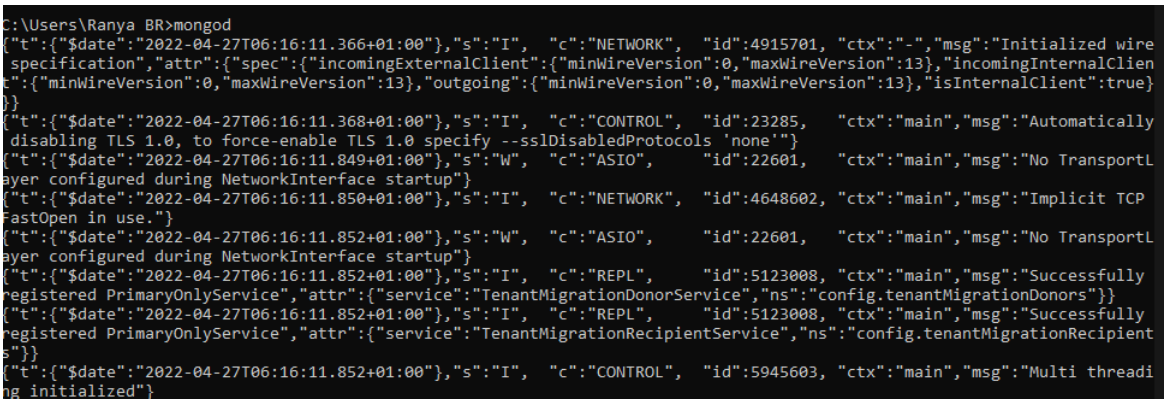


FIGURE 1 – Installation de MongoDB sous Windows

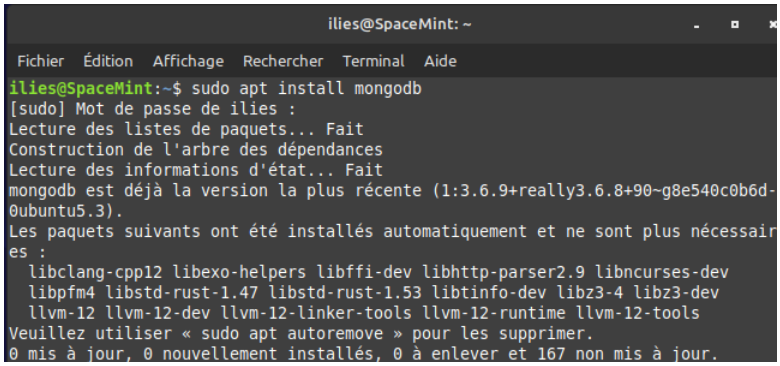
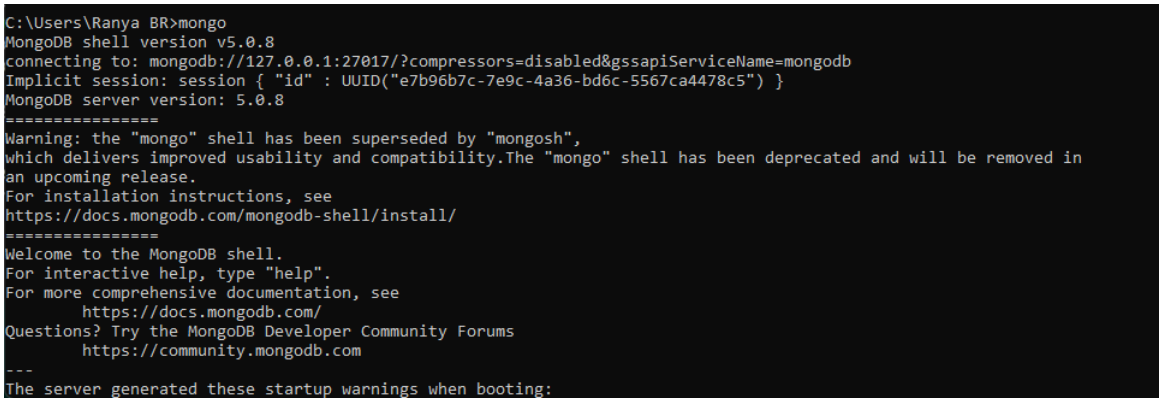
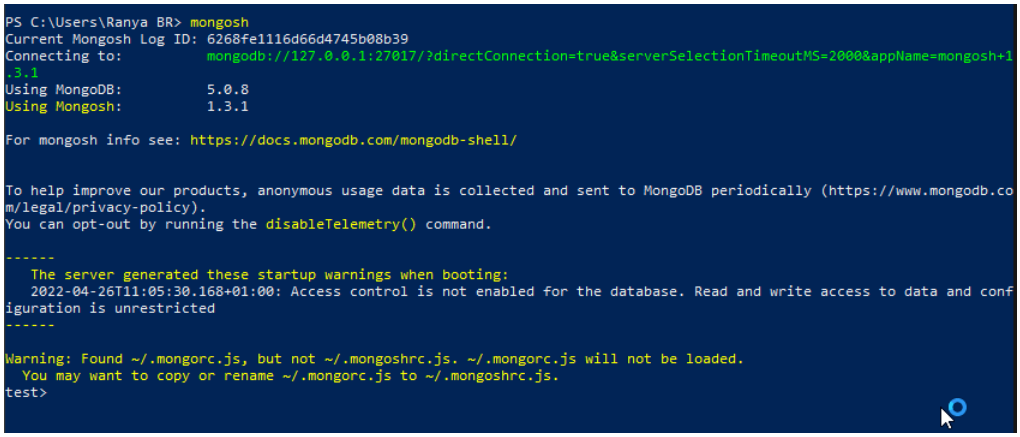


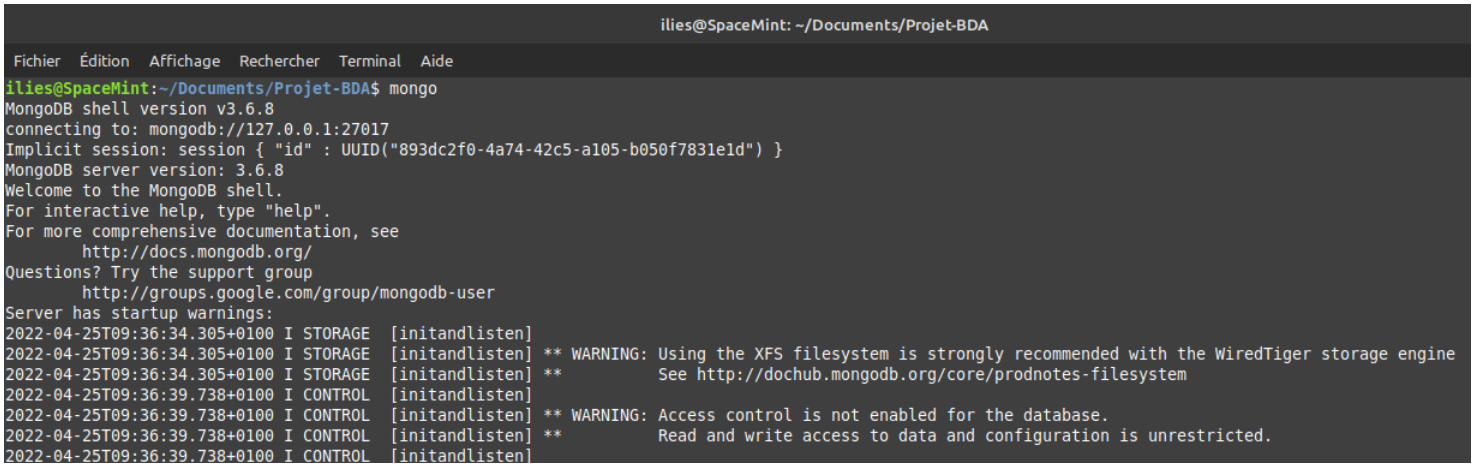
FIGURE 2 – Installation de MongoDB sous Linux

On utilise cette commande pour vérifier l'existence du fichier data.





Après avoir téléchargé MongoDB Shell, on lance la commande mongosh pour pouvoir exécuter des commandes. Sous Linux, il suffit de taper la commande mongo dans le terminal pour pouvoir utiliser MongoDB



1.1 Téléchargement du fichier et création d'une base de données BDD et une collection world

MongoDB Compass est un GUI qui nous facilite les taches de création de base de données, importation de données et création de collection. Cependant, pour ce TP, nous allons utiliser la ligne de commandes.

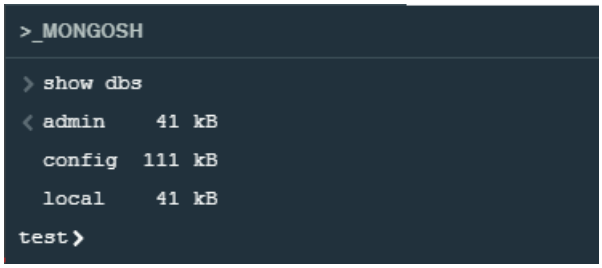


FIGURE 3 – Les bases de données avant la création

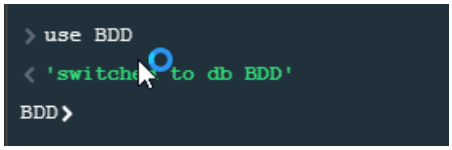


FIGURE 4 – Création de BDD sous Windows

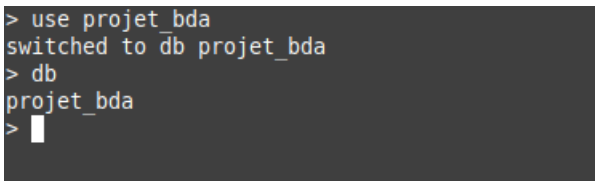


FIGURE 5 – Création de BDD sous Linux

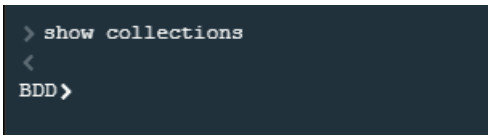


FIGURE 6 – Liste des collections avant la création de world sous Windows

FIGURE 7 – Liste des collections avant la création de world sous Linux

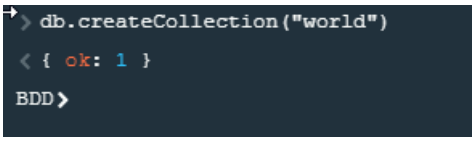
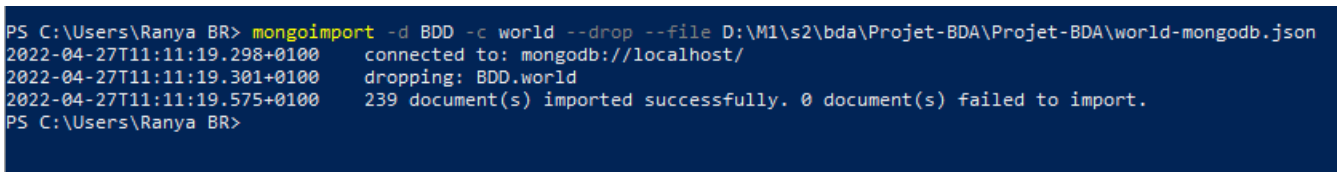


FIGURE 8 – Création de la collection world sous Windows

FIGURE 9 – Création de la collection world sous Linux

1.2 Importation des données dans cette collection



```
ilies@SpaceMint: ~/Documents/Projet-BDA
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
ilies@SpaceMint:~/Documents/Projet-BDA$ mongoimport --db projet_bda --collection world --file world-mongodb.json
2022-04-25T10:05:23.812+0100    connected to: localhost
2022-04-25T10:05:23.868+0100    imported 239 documents
ilies@SpaceMint:~/Documents/Projet-BDA$
```

1.3 Lancement de robo3t.exe pour vérifier les documents

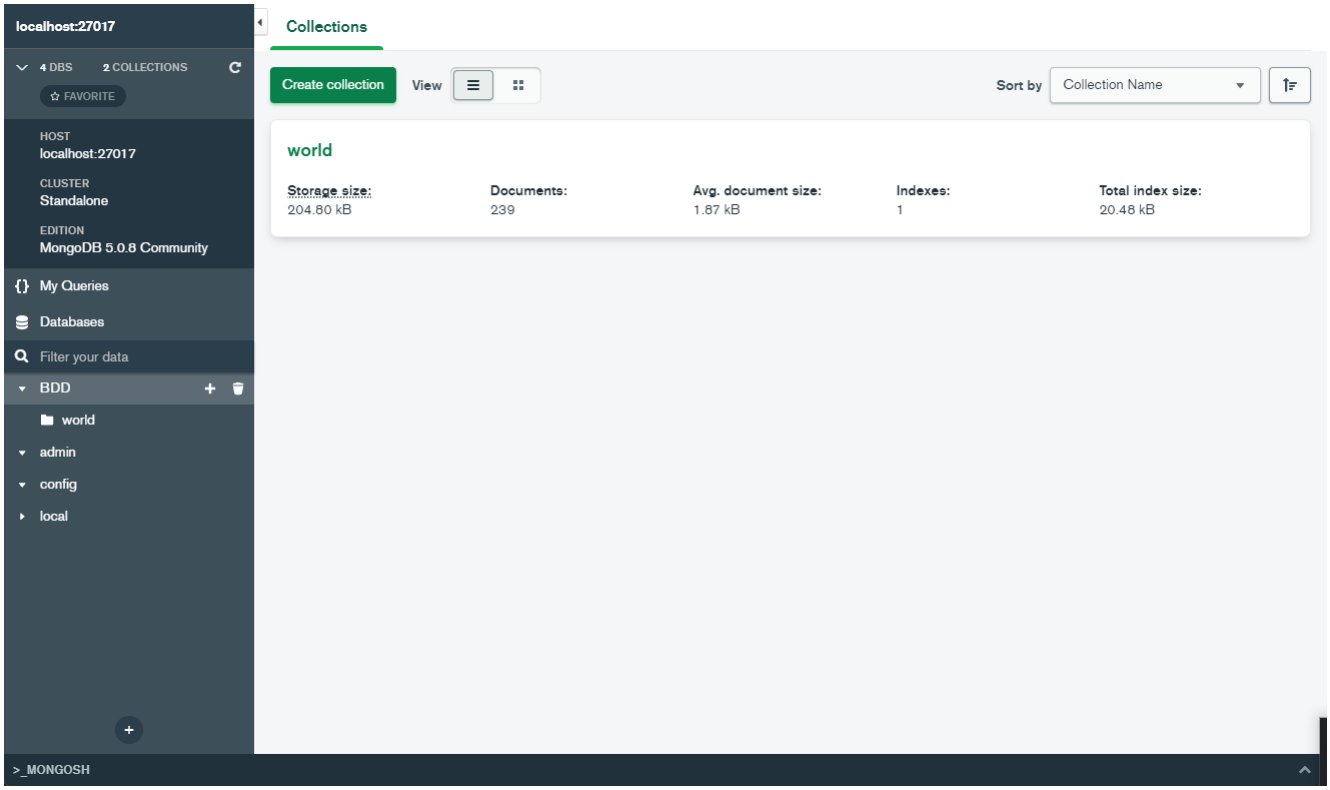


FIGURE 10 – Vérification des documents avec mongoDB Compass

2 Partie II : MongoDB – PyMongo - Python

2.1 1. Déterminer le nombre exact de pays

```
>_MONGOSH
> db.world.distinct("Name").length
< 239
BDD>
```

FIGURE 11 – Execution de la commande avec Mongo Shell

```python
def QST1(): # Déterminer le nombre exact de pays
    return len(collection.distinct("Name")) #or return collection.distinct("Name").length

```

FIGURE 12 – La fonction Utilisée

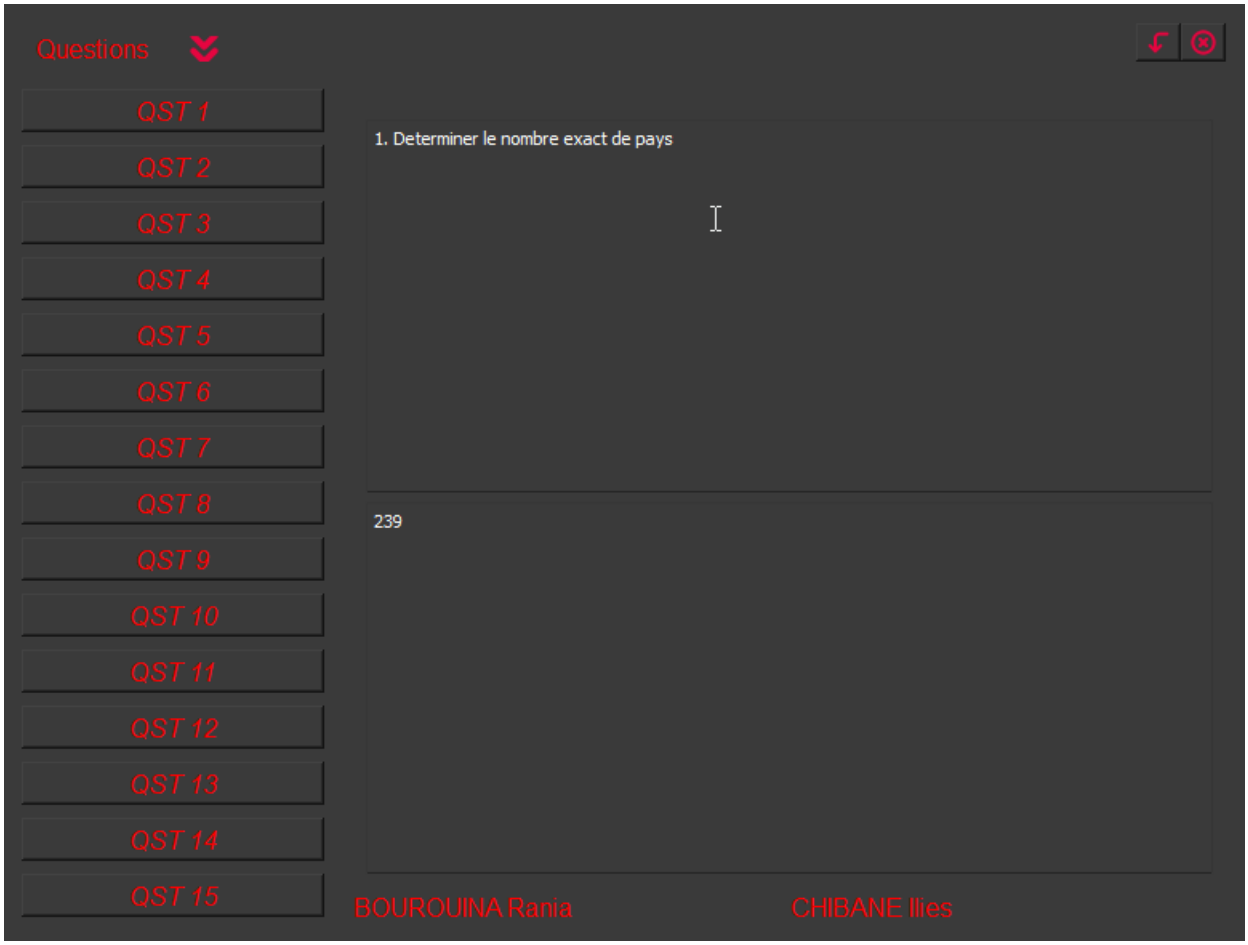


FIGURE 13 – Le résultat sur l'interface

2.2 2. Lister les différents continents

```
>_MONGOSH

> db.world.distinct("Continent")
< [
    'Africa',
    'Antarctica',
    'Asia',
    'Europe',
    'North America',
    'Oceania',
    'South America'
  ]
BDD>
```

FIGURE 14 – Execution de la commande avec Mongo Shell

```
1  def QST2(): # Lister les différents continents
2      return collection.distinct("Continent")
3
```

FIGURE 15 – La fonction Utilisée

FIGURE 16 – Le résultat sur l'interface

2.3 3. Lister les informations de l'Algérie

```
>_MONGOSH

> db.world.findOne({"Name":"Algeria"})
< { _id: ObjectId("626916c7e77e9c4a05cdf687"),
    Code: 'DZA',
    Name: 'Algeria',
    Continent: 'Africa',
    Region: 'Northern Africa',
    SurfaceArea: 2381741,
    IndepYear: 1962,
    Population: 44700000,
    LifeExpectancy: 69.7,
    GNP: 49982,
    GNPOld: 46966,
    LocalName: 'Al-Jaza'ir/Algérie',
    GovernmentForm: 'Republic',
    HeadOfState: 'Abdelmadjid Tebboune',
    Capital: { ID: 35, Name: 'Alger', District: 'Alger', Population: 2168000 },
    Code2: 'DZ',
    Cities:
     [ { ID: 36, Name: 'Oran', District: 'Oran', Population: 609823 },
       { ID: 37,
         Name: 'Constantine',
         District: 'Constantine',
         Population: 443727 },
```

FIGURE 17 – Execution de la commande avec Mongo Shell

FIGURE 18 – La fonction Utilisée

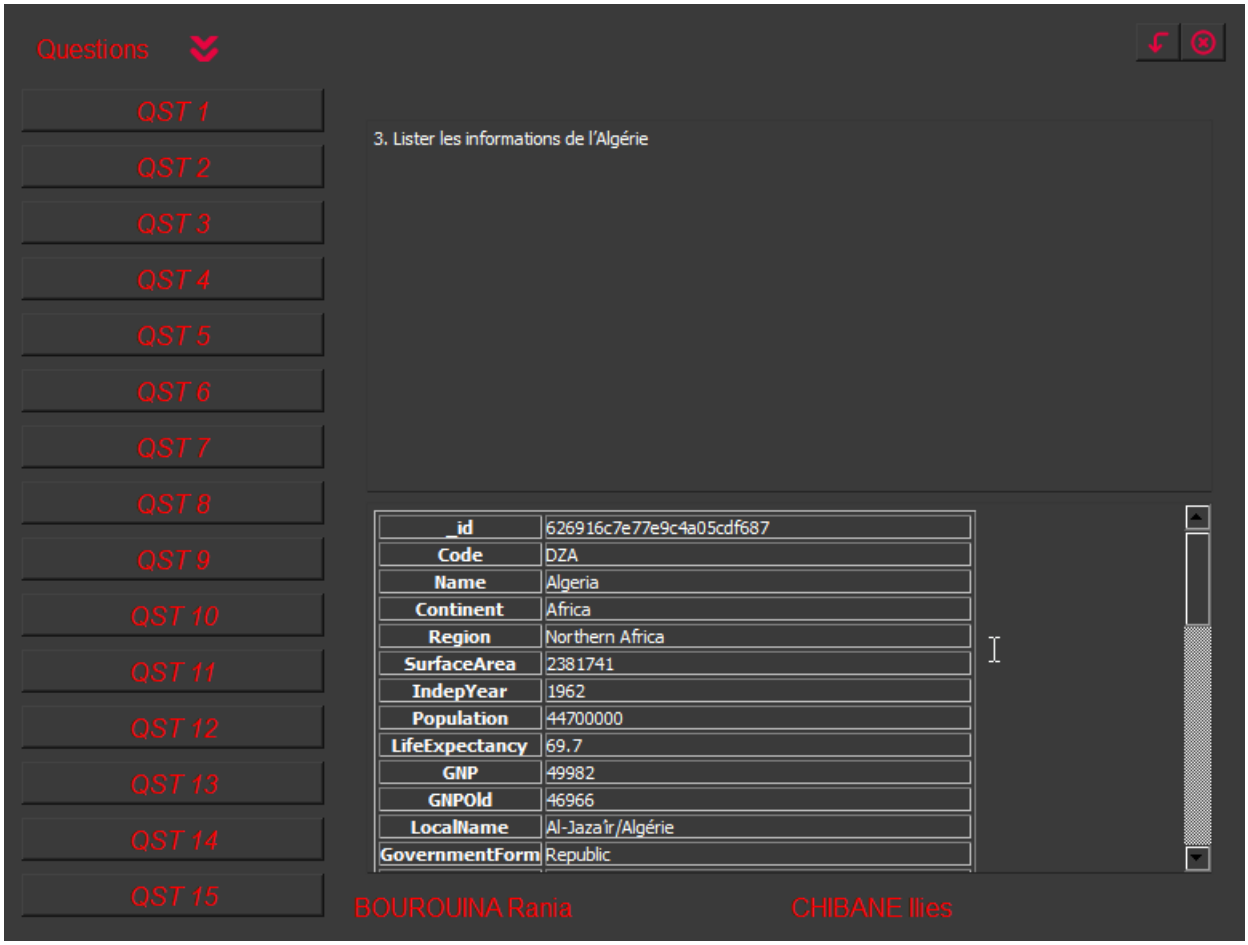


FIGURE 19 – Le résultat sur l'interface

2.4 4. Lister les pays du continent Africain, ayant une population inférieure à 100000 habitants

```
> db.world.find({"Continent":"Africa", "Population":{"$lt": 100000}},{"Name" :1}).sort("Name")
< { _id: ObjectId("626916c7e77e9c4a05cdf6ac"),
    Name: 'British Indian Ocean Territory' }
  { _id: ObjectId("626916c7e77e9c4a05cdf703"),
    Name: 'Saint Helena' }
  { _id: ObjectId("626916c7e77e9c4a05cdf711"),
    Name: 'Seychelles' }
BDD>
```

FIGURE 20 – Execution de la commande avec Mongo Shell

```python
def QST4(): #  Lister les pays du continent Africain, ayant une population inférieure à 100000 habitants
    countries = []
    for post in (collection.find({"Continent":"Africa", "Population":{"$lt": 100000}},{"Name" :1}).sort("Name")):
        countries.append(post["Name"])
    return countries

```

FIGURE 21 – La fonction Utilisée

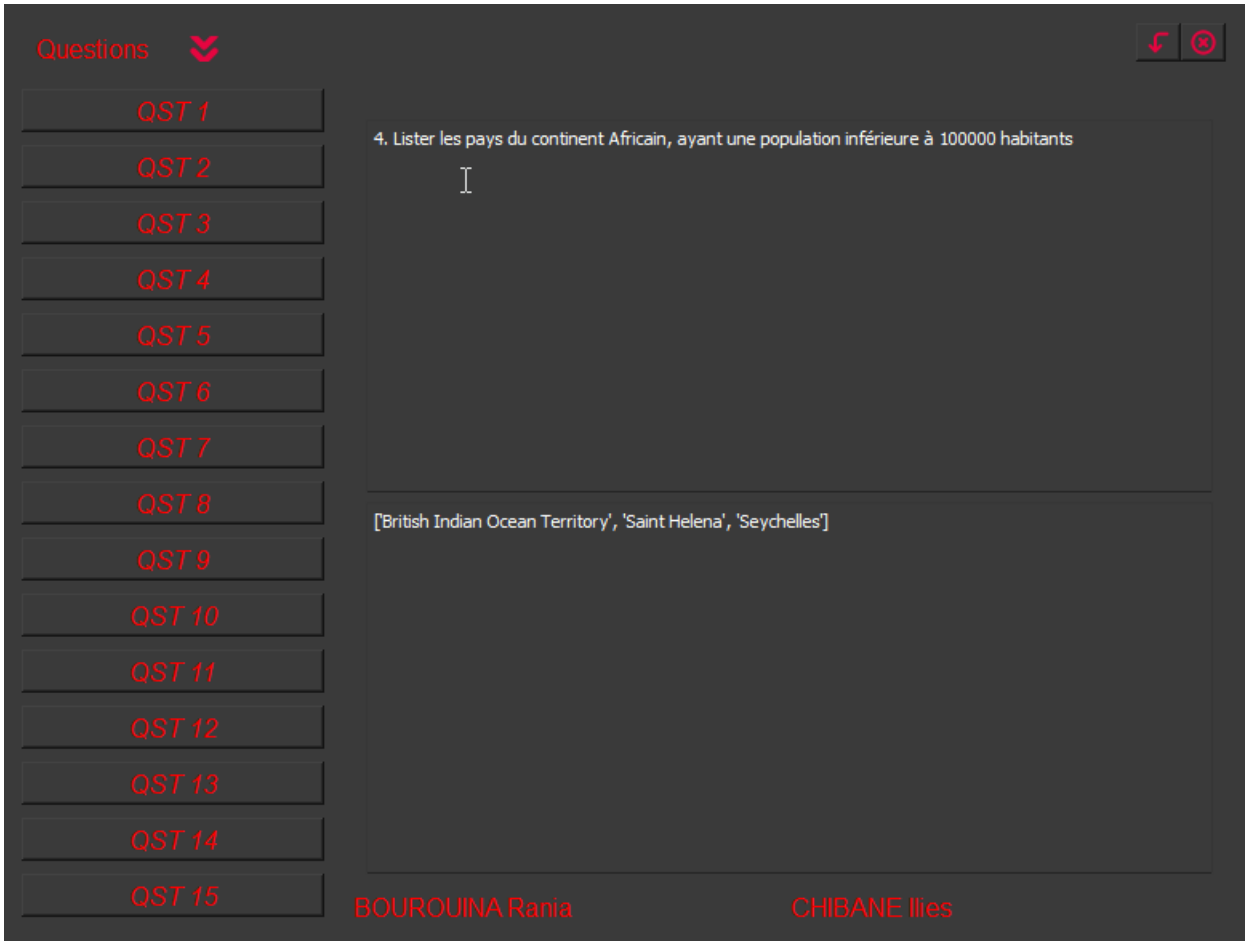


FIGURE 22 – Le résultat sur l'interface

2.5 5. Lister les pays indépendant du continent océanique

```
> db.world.find({"Continent":"Oceania", "IndepYear":{"$ne": "NA"}},{"Name" :1}).sort("Name")
< { _id: ObjectId("626916c7e77e9c4a05cdf659"), Name: 'Australia' }
  { _id: ObjectId("626916c7e77e9c4a05cdf68c"),
    Name: 'Fiji Islands' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6b7"), Name: 'Kiribati' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6cd"),
    Name: 'Marshall Islands' }
  { _id: ObjectId("626916c7e77e9c4a05cdf691"),
    Name: 'Micronesia, Federated States of' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6e6"), Name: 'Nauru' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6e9"),
    Name: 'New Zealand' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6f2"), Name: 'Palau' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6f3"),
    Name: 'Papua New Guinea' }
  { _id: ObjectId("626916c7e77e9c4a05cdf72c"), Name: 'Samoa' }
  { _id: ObjectId("626916c7e77e9c4a05cdf707"),
    Name: 'Solomon Islands' }
  { _id: ObjectId("626916c7e77e9c4a05cdf719"), Name: 'Tonga' }
  { _id: ObjectId("626916c7e77e9c4a05cdf71d"), Name: 'Tuvalu' }
  { _id: ObjectId("626916c7e77e9c4a05cdf72a"), Name: 'Vanuatu' }
BDD>
```

FIGURE 23 – Execution de la commande avec Mongo Shell

```python
def QST5(): # Lister les pays indépendant du continent océanique
    countries = []
    for post in (collection.find({"Continent":"Oceania", "IndepYear":{"$ne": "NA"}},{"Name" :1}).sort("Name")):
        countries.append(post["Name"])
    return countries

```

FIGURE 24 – La fonction Utilisée

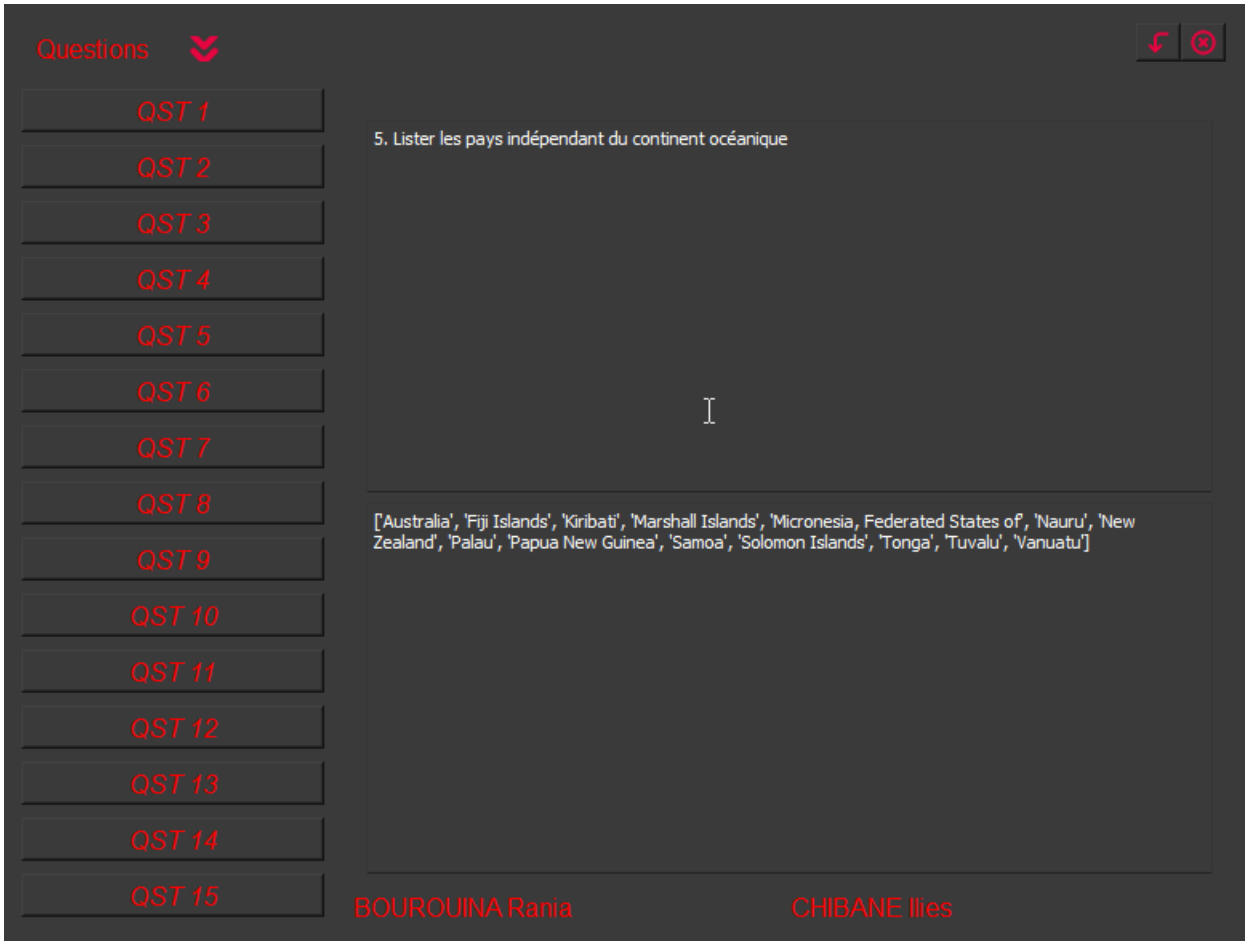


FIGURE 25 – Le résultat sur l'interface

2.6 6. Quel est le plus gros continent en termes de surface ? (un seul continent affiché à la fin)

```
>_MONGOSH
> db.world.find().sort("SurfaceArea", -1).limit(1)
< { _id: ObjectId("626916c7e77e9c4a05cdf728"),
    Code: 'VAT',
    Name: 'Holy See (Vatican City State)',
    Continent: 'Europe',
    Region: 'Southern Europe',
    SurfaceArea: 0.4,
    IndepYear: 1929,
    Population: 1000,
    LifeExpectancy: 'NA',
    GNP: 9,
    GNPOld: 'NA',
    LocalName: 'Santa Sede/Città del Vaticano',
    GovernmentForm: 'Independent Church State',
    HeadOfState: 'Johannes Paavali II',
    Capital:
     { ID: 3538,
       Name: 'Città del Vaticano',
       District: '-',
       Population: 455 },
    Code2: 'VA',
    OffLang: [ { Language: 'Italian', Percentage: 0 } ] }
BDD>
```

FIGURE 26 – Execution de la commande avec Mongo Shell

```python
def QST6(): # Quel est le plus gros continent en termes de surface ? (un seul continent affiché à la fin)
    countries = []
    for post in (collection.find().sort("SurfaceArea", -1).limit(1)):
        countries.append(post["Name"])
    return countries

```

FIGURE 27 – La fonction Utilisée

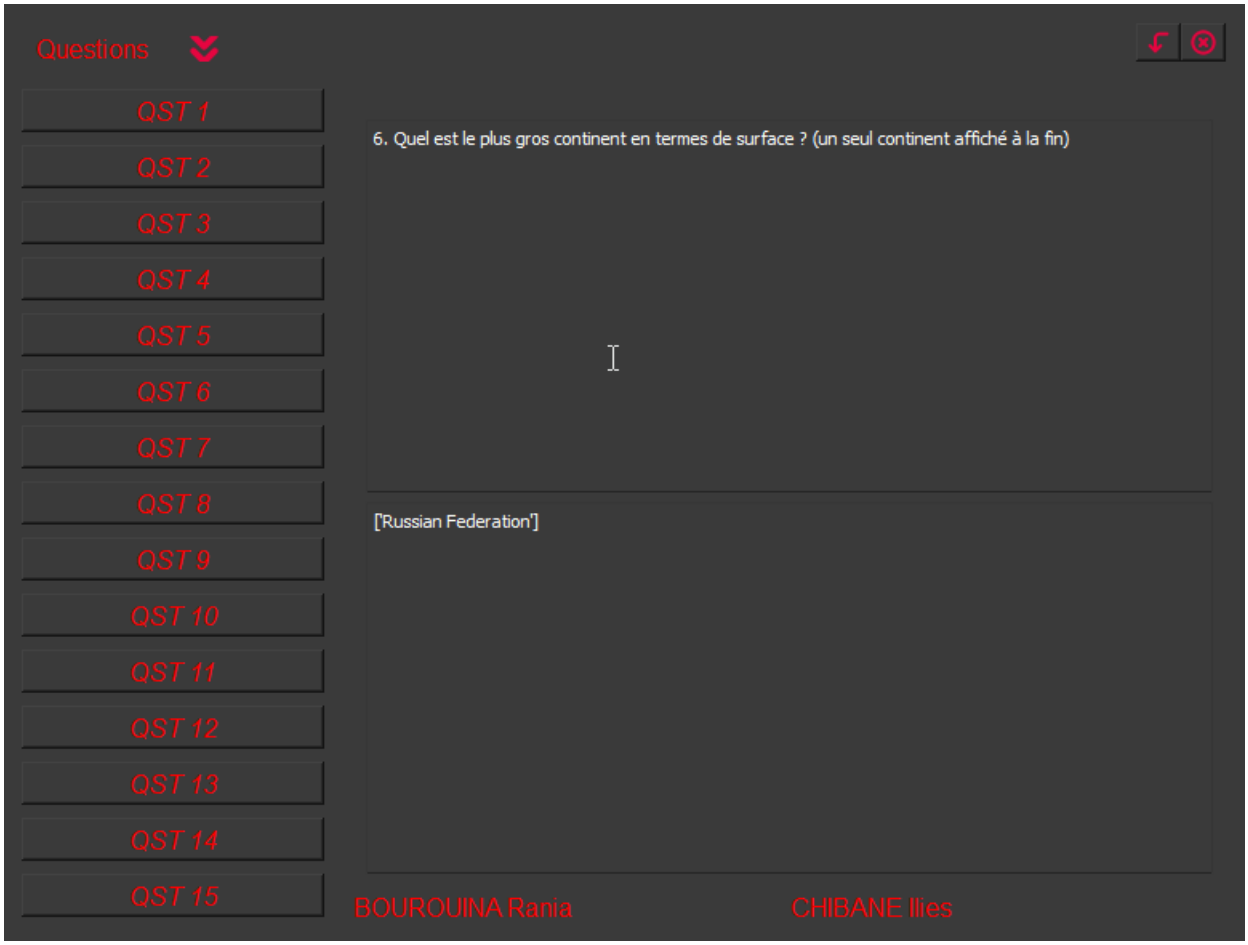


FIGURE 28 – Le résultat sur l'interface

2.7 7. Donner par continents le nombre de pays, la population totale et en bonus le nombre de pays indépendant.

```
> db.world.aggregate( [{ $group:{ _id:"$Continent", 'total': {'$sum': '$Population'}, 'pays':{'$sum':1} }}])
< { _id: 'Oceania', total: 30401150, pays: 28 }
  { _id: 'Asia', total: 3705025700, pays: 51 }
  { _id: 'North America', total: 482993000, pays: 37 }
  { _id: 'Europe', total: 730074600, pays: 46 }
  { _id: 'Antarctica', total: 0, pays: 5 }
  { _id: 'Africa', total: 797704000, pays: 58 }
  { _id: 'South America', total: 345780000, pays: 14 }
BDD>
```

FIGURE 29 – Execution de la commande avec Mongo Shell

Le bonus (nombre de pays indépendants par continent sera rajouté lors de l'implémentation avec python.

```python
def QST7(): #  Donner par continents le nombre de pays, la population totale et en bonus le nombre de pays indépendant.
    continents_infos = {}
    for post in collection.distinct("Continent"):
            continent_infos = {}
            count = collection.count_documents({"Continent":post})
            count_indi = collection.count_documents({"Continent":post, "IndepYear":{"$ne": "NA"}})
            continent_infos["Number of countries"] = count
            continent_infos["Size of the population"] = 0
            continent_infos["Number of independent countries"] = count_indi
            continents_infos[post] = continent_infos

    pipe = [{'$group': {'_id': "$Continent", 'total': {'$sum': '$Population'}}}]
    for post in collection.aggregate(pipeline=pipe):
        continents_infos[post["_id"]]["Size of the population"] = post["total"]

    return continents_infos

```

FIGURE 30 – La fonction Utilisée

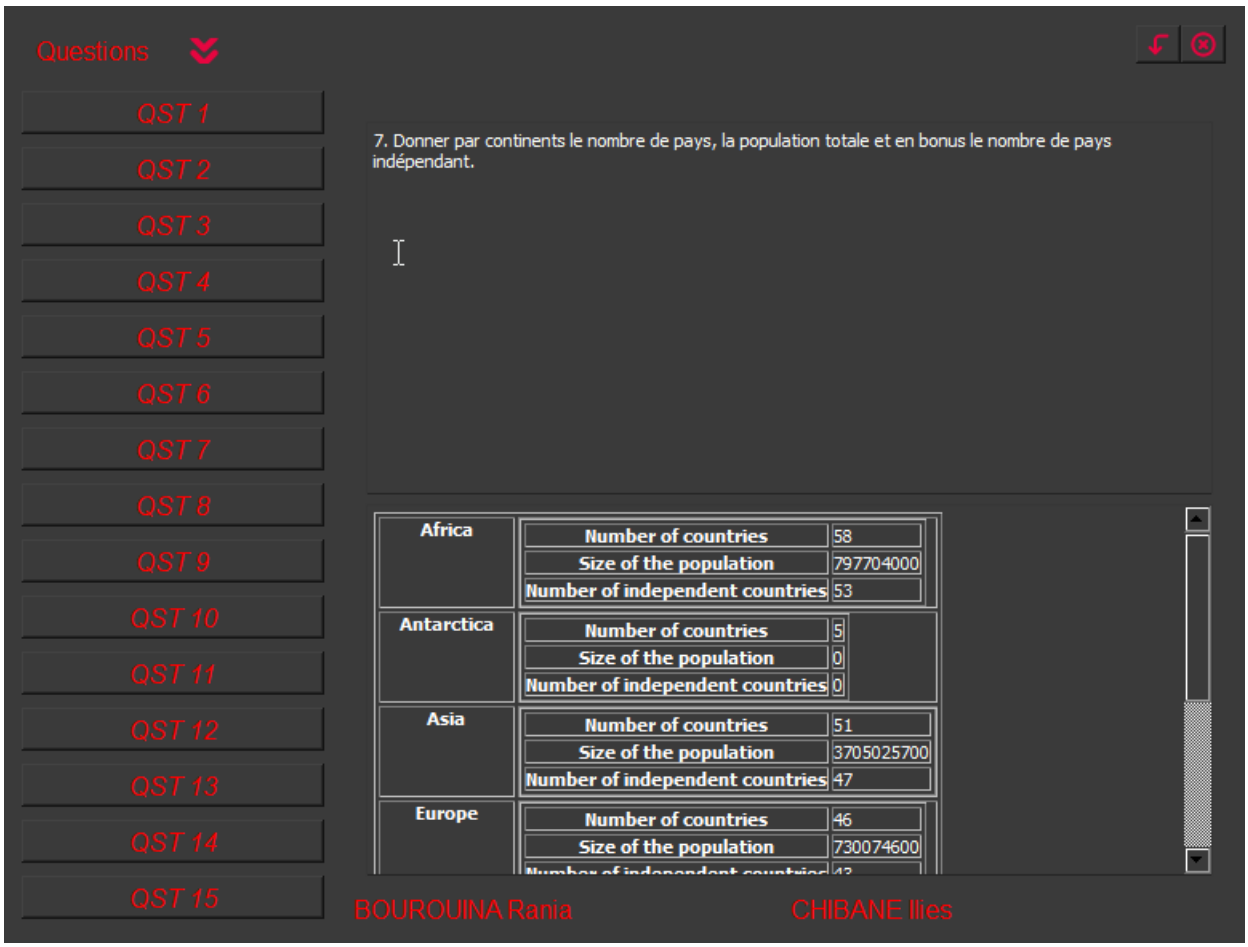


FIGURE 31 – Le résultat sur l'interface

NB : pour avoir le nombre de pays par continent, il suffit de compter le nombre d'occurences du document car le code

du pays est unique dans chacun.

2.8 8. Donner la population totale des villes d'Algérie

```
>_MONGOSH
> db.world.aggregate(
    [
        {$match: {"Name": "Algeria"}},
        {$group: {
            _id:'$Cities',
            'total': {'$sum': '$Population'}
        }}
    ]
)
< { _id:
   [ { ID: 36, Name: 'Oran', District: 'Oran', Population: 609823 },
     { ID: 37,
       Name: 'Constantine',
       District: 'Constantine',
       Population: 443727 },
     { ID: 38, Name: 'Annaba', District: 'Annaba', Population: 222518 },
     { ID: 39, Name: 'Batna', District: 'Batna', Population: 183377 },
     { ID: 40, Name: 'Sétif', District: 'Sétif', Population: 179055 },
     { ID: 41,
       Name: 'Sidi Bel Abbès',
       District: 'Sidi Bel Abbès',
       Population: 153106 },
     { ID: 42, Name: 'Skikda', District: 'Skikda', Population: 128747 },
     { ID: 43, Name: 'Biskra', District: 'Biskra', Population: 128281 },
     { ID: 44,
```

FIGURE 32 – Execution de la commande avec Mongo Shell

```python
def QST8(): #  Donner la population totale des villes d'Algérie
    n = collection.find_one({"Name":"Algeria"})["Cities"]
    count = sum(map(lambda x: int(x['Population']), n))

    return count

```

FIGURE 33 – La fonction Utilisée

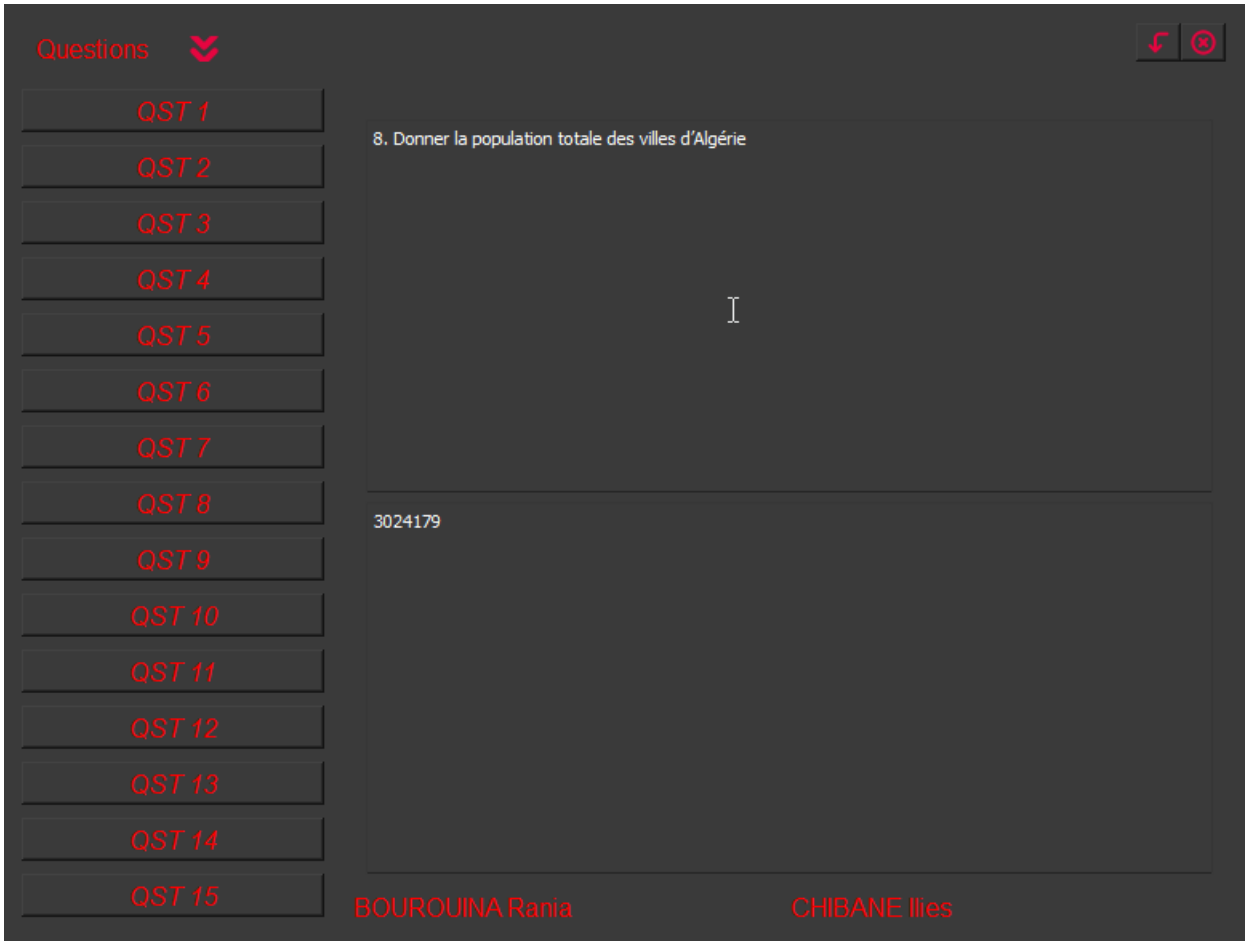


FIGURE 34 – Le résultat sur l'interface

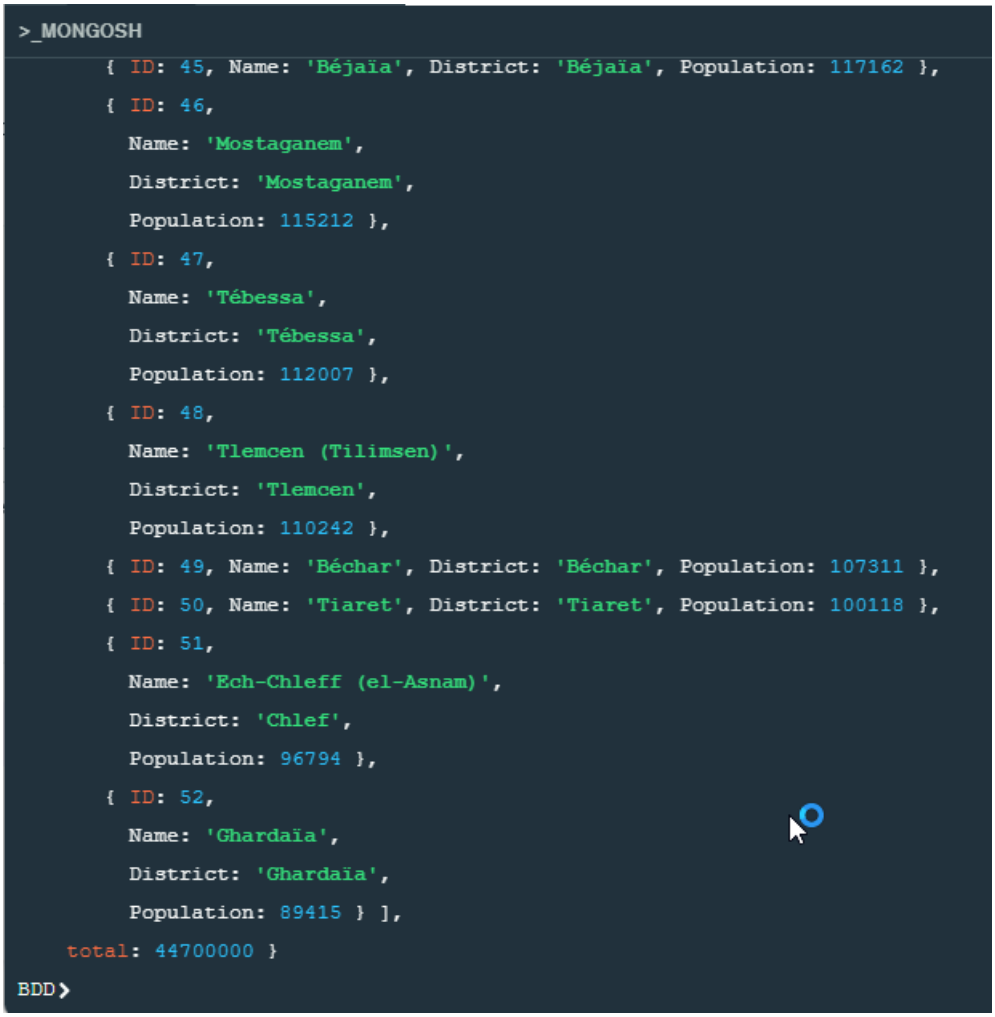


FIGURE 35 – Execution de la commande avec Mongo Shell

2.9 9. Donner la capitale (uniquement nom de la ville et population) d'Algérie

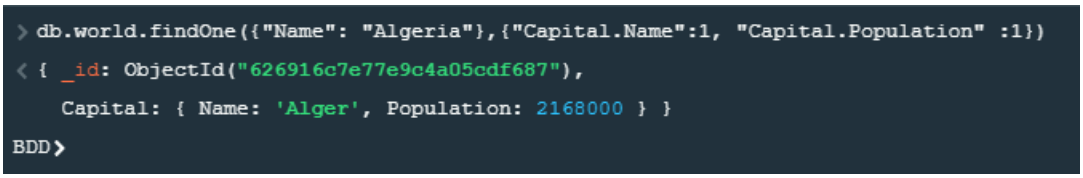


FIGURE 36 – Execution de la commande avec Mongo Shell

```python
def QST9(): #  9. Donner la capitale (uniquement nom de la ville et population) d'Algérie
    capital = collection.find_one({"Name": "Algeria"})["Capital"]
    keys = ["Name", "Population"]
    capital_info = [capital[key] for key in keys]
    return capital_info

```

FIGURE 37 – La fonction Utilisée

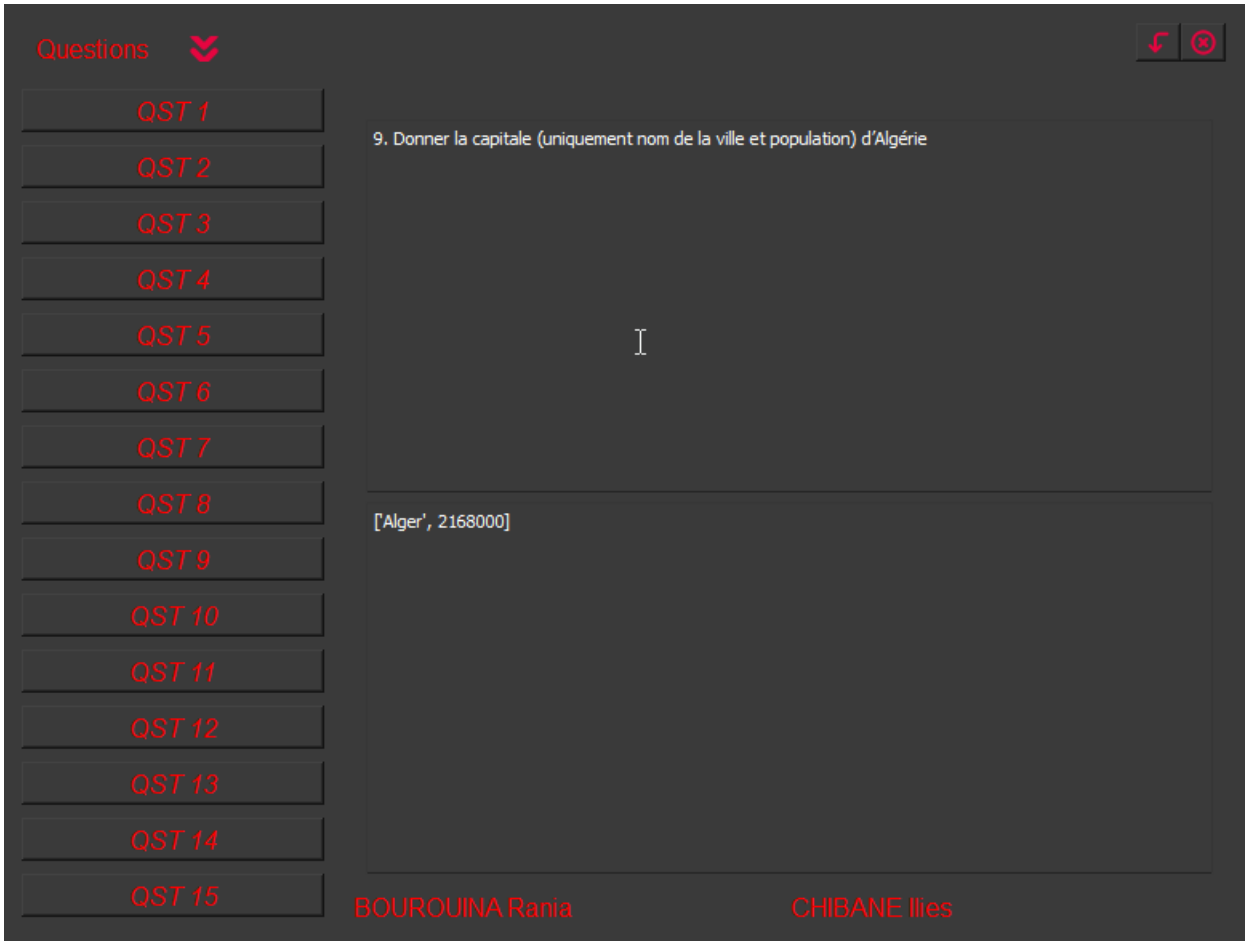


FIGURE 38 – Le résultat sur l'interface

2.10 10. Quelles sont les langues parlées dans plus de 15 pays ?

```
>_MONGOSH

> db.world.aggregate([
  {
  $addFields: { Langue : {$concatArrays: ["$OffLang","$NotOffLang" ] } }
  },
  {$unwind: "$Langue"
  },
  { $group: {
      _id:"$Langue.Language",
      "nb" : {$sum :1}
  }
  },
  { $match: { "nb": { $gt: 15 } } },
 ])
{ _id: 'German', nb: 17 }
{ _id: 'Spanish', nb: 25 }
{ _id: 'Chinese', nb: 17 }
{ _id: 'Russian', nb: 17 }
{ _id: 'Arabic', nb: 29 }
{ _id: 'French', nb: 18 }
{ _id: 'English', nb: 44 }
BDD>
```

FIGURE 39 – Execution de la commande avec Mongo Shell

FIGURE 40 – La fonction Utilisée

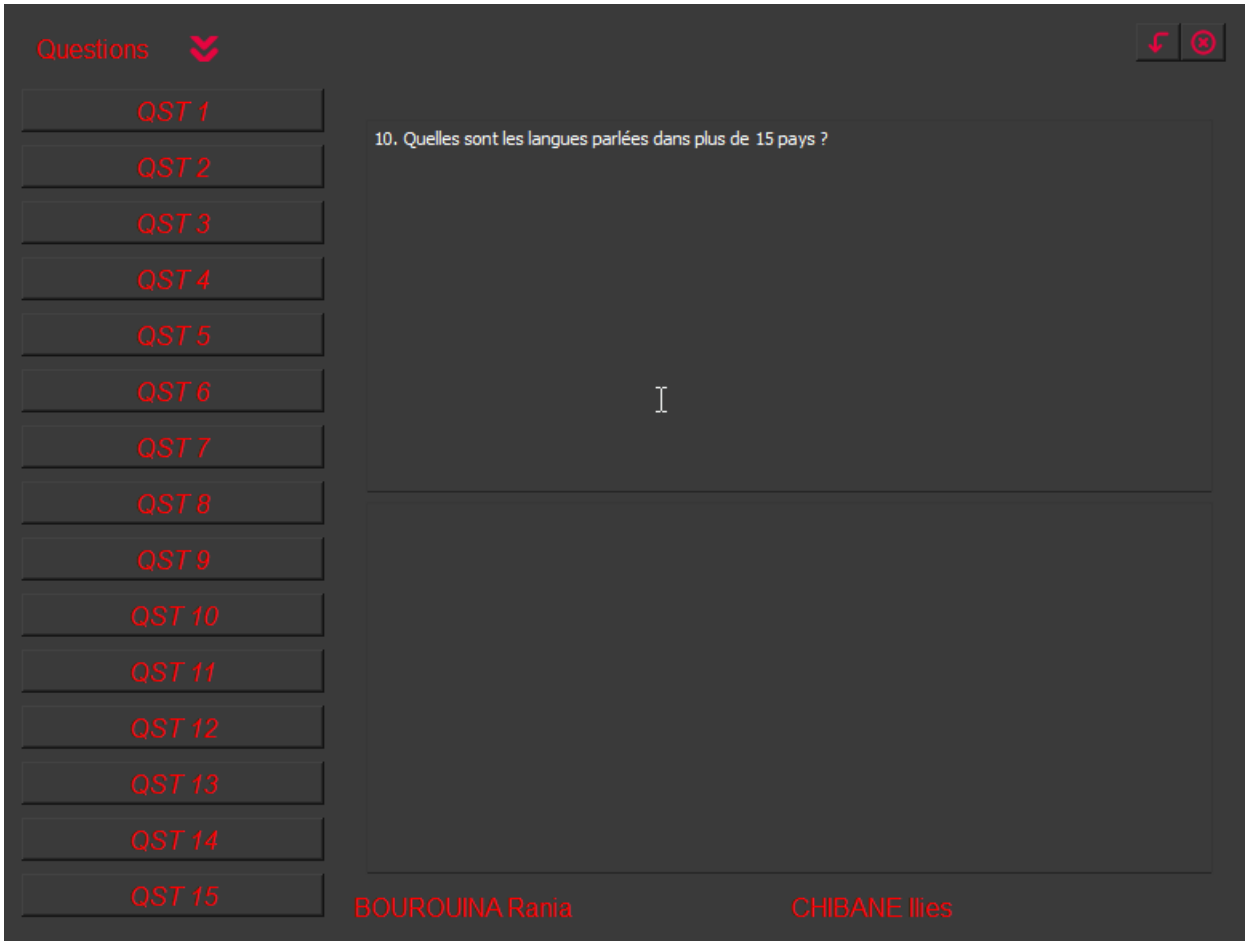


FIGURE 41 – Le résultat sur l'interface

2.11 11. Calculer pour chaque pays le nombre de villes (pour les pays ayant au moins 100 villes), en les triant par ordre décroissant du nombre de villes

```
>_MONGOSH

> db.world.aggregate([
      {
          $addFields: { villes : {$size: { "$ifNull": [ "$Cities", [] ] } } }
      },
      {
        $match: {
          villes: { $gt: 100 }
        }
      },
      {   $sort: {"villes":-1}  },
      {
         $project: {
              _id: "$Name",
              "Number of Cities" :"$villes"
         }
      }
  ])
< { _id: 'China', 'Number of Cities': 362 }
  { _id: 'India', 'Number of Cities': 340 }
  { _id: 'United States', 'Number of Cities': 273 }
  { _id: 'Brazil', 'Number of Cities': 249 }
  { _id: 'Japan', 'Number of Cities': 247 }
  { _id: 'Russian Federation', 'Number of Cities': 188 }
  { _id: 'Mexico', 'Number of Cities': 172 }
  { _id: 'Philippines', 'Number of Cities': 135 }
BDD>
```

FIGURE 42 – Execution de la commande avec Mongo Shell

```python
def QST11(): # 11. Calculer pour chaque pays le nombre de villes (pour les pays ayant au moins 100 villes), en les triant par ordre décroissant du nombre de villes
    '''
    db.world.aggregate([
    {
        $addFields: { villes : {$size: { "$ifNull": [ "$Cities", [] ] } } }
    },
    {
      $match: {
        villes: { $gt: 100 }
      }
    },
    {   $sort: {"villes":-1}  },
    {
       $project: {
            _id: "$Name",
            "Number of Cities" :"$villes"
       }
    }
])
    '''
    pays = []
    pipe = [{ '$addFields': { "villes" : {'$size': { "$ifNull": [ "$Cities", [] ] } } }}, {'$match': {"villes": { '$gt': 100 }}},{   '$sort': {"villes":-1}  },{'$project': {"_id": "$Name","Number of Cities" :"$villes"}}]
    for post in  collection.aggregate(pipeline=pipe):
        pays.append(post)
    return pays
```

FIGURE 43 – La fonction Utilisée

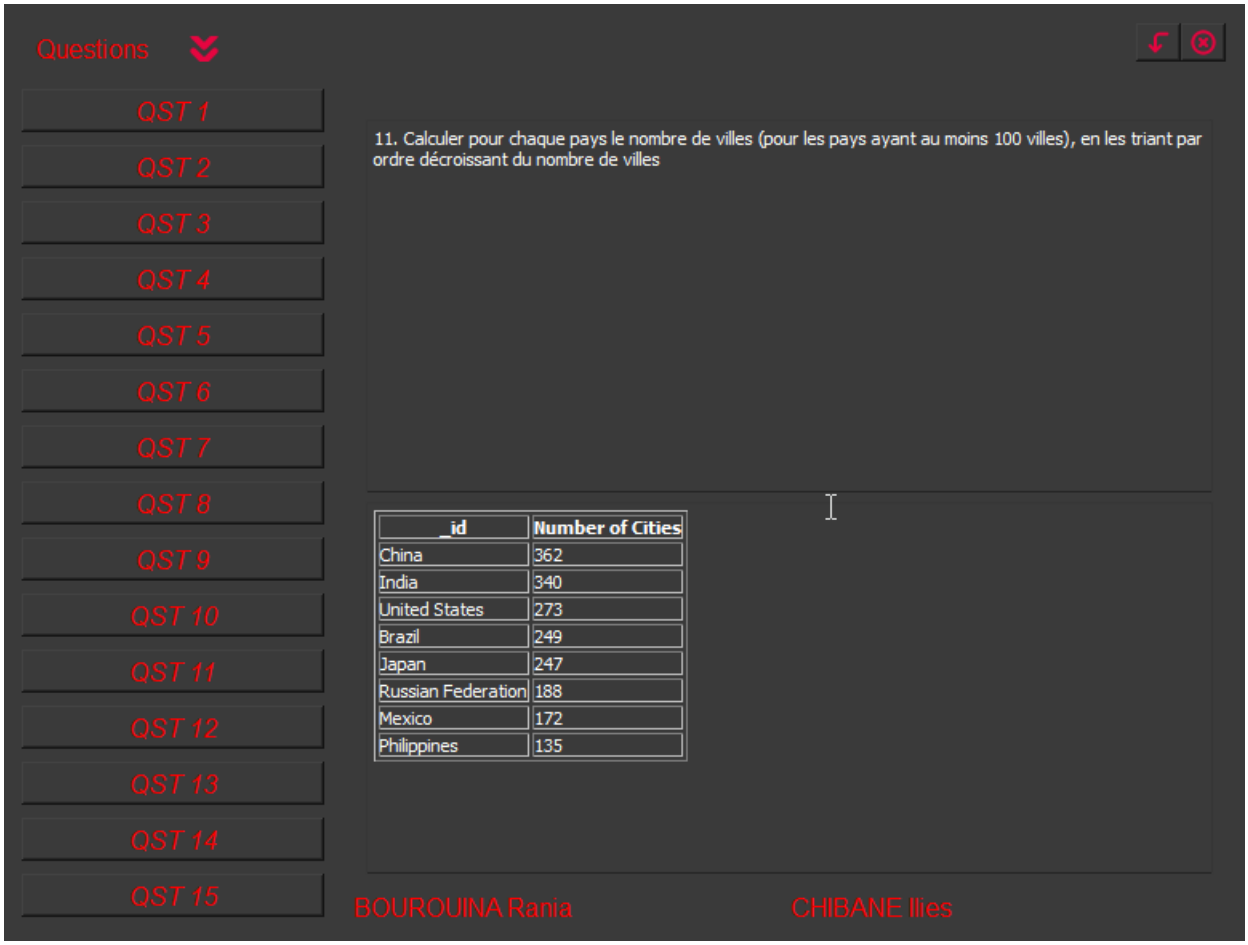


FIGURE 44 – Le résultat sur l'interface

2.12 12. Lister les 10 villes les plus habitées, ainsi que leur pays, dans l'ordre décroissant de la population

```
>_MONGOSH
> db.world.aggregate([
{$unwind: "$Cities"
},
{$sort: {"Cities.Population":-1} },
{ $limit : 10 },
{$sort: {"Cities.Population":1} },
{
$project: {
    _id:"$Cities.Name",
    "Country":"$Name",
    "City Population":"$Cities.Population"
        }
}
])
{ _id: 'Tianjin', Country: 'China', 'City Population': 5286800 }
{ _id: 'Rio de Janeiro',
  Country: 'Brazil',
  'City Population': 5598953 }
{ _id: 'Chongqing',
  Country: 'China',
  'City Population': 6351600 }
{ _id: 'Delhi', Country: 'India', 'City Population': 7206704 }
{ _id: 'New York',
  Country: 'United States',
  'City Population': 8008278 }
{ _id: 'Istanbul',
```

FIGURE 45 – Execution de la commande avec Mongo Shell

```
>_MONGOSH
      ])
{ _id: 'Tianjin', Country: 'China', 'City Population': 5286800 }
  { _id: 'Rio de Janeiro',
    Country: 'Brazil',
    'City Population': 5598953 }
  { _id: 'Chongqing',
    Country: 'China',
    'City Population': 6351600 }
  { _id: 'Delhi', Country: 'India', 'City Population': 7206704 }
  { _id: 'New York',
    Country: 'United States',
    'City Population': 8008278 }
  { _id: 'Istanbul',
    Country: 'Turkey',
    'City Population': 8787958 }
  { _id: 'Karachi',
    Country: 'Pakistan',
    'City Population': 9269265 }
  { _id: 'Shanghai', Country: 'China', 'City Population': 9696300 }
  { _id: 'São Paulo',
    Country: 'Brazil',
    'City Population': 9968485 }
  { _id: 'Mumbai (Bombay)',
    Country: 'India',
    'City Population': 10500000 }
BDD>
```

FIGURE 46 – Execution de la commande avec Mongo Shell

```
def QST12(): # 12. Lister les 10 villes les plus habitées, ainsi que leur pays, dans l'ordre décroissant de la population
    '''
            db.world.aggregate([
        {$unwind: "$Cities"
        },
        {$sort: {"Cities.Population":-1} },
        { $limit : 10 },
        {$sort: {"Cities.Population":1} },
        {
        $project: {
            _id:"$Cities.Name",
            "Country":"$Name",
            "City Population":"$Cities.Population"
                }
        }
    ])
    '''
    villes = []
    pipe = [{'$unwind': "$Cities"},{'$sort': {"Cities.Population":-1} },{ '$limit' : 10 },{'$sort': {"Cities.Population":1} },{'$project': { "_id":"$Cities.Name","Country":"$Name","City Population":"$Cities.Population"}}]
    for post in  collection.aggregate(pipeline=pipe):
        villes.append(post)
    return villes
```

FIGURE 47 – La fonction Utilisée

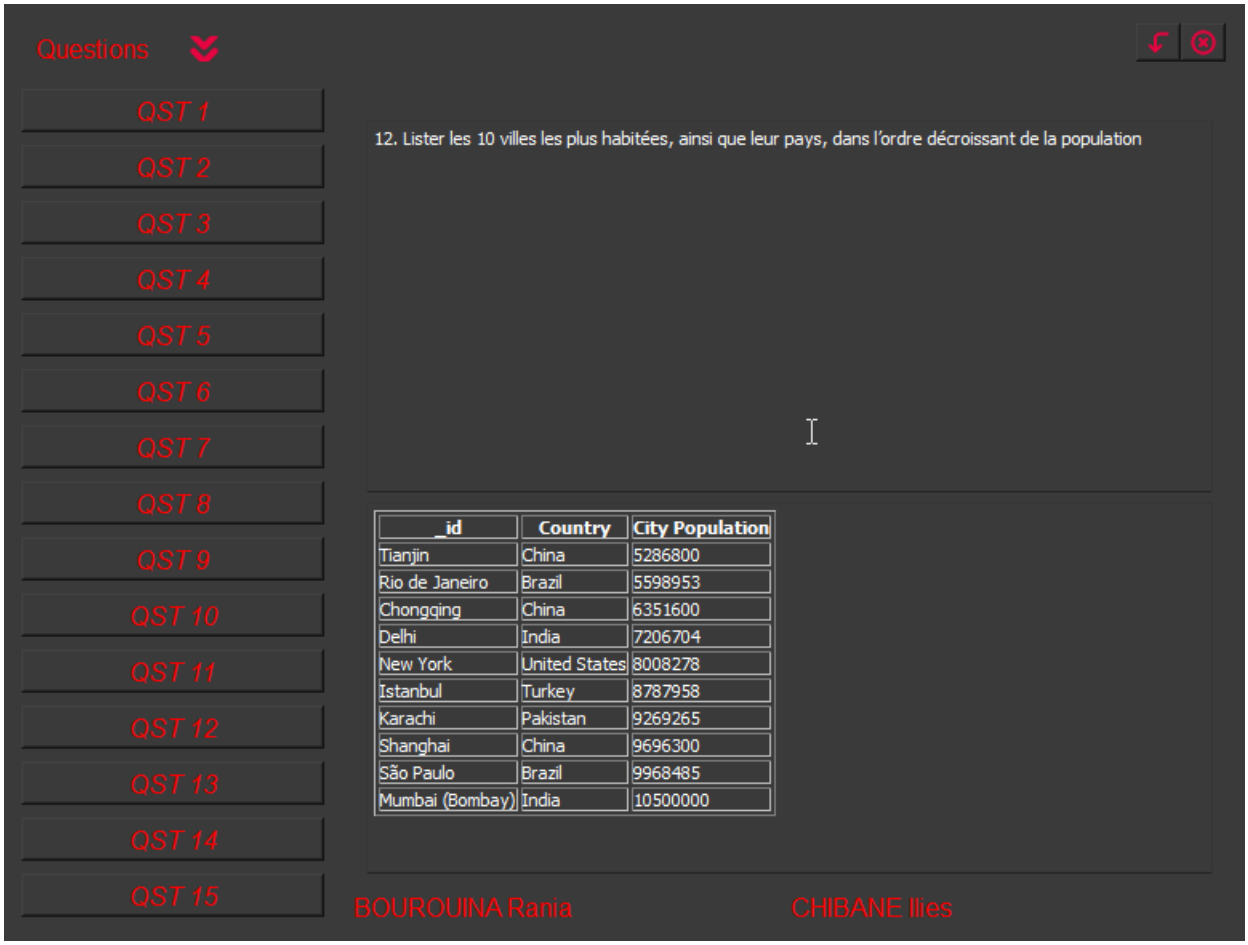


FIGURE 48 – Le résultat sur l'interface

2.13 13. Lister les pays pour lesquels l'Arabe est une langue officielle

```
>_MONGOSH
> db.world.find({"OffLang.Language":{$eq: "Arabic"}},{"Name":1})
< { _id: ObjectId("626916c7e77e9c4a05cdf653"),
    Name: 'United Arab Emirates' }
  { _id: ObjectId("626916c7e77e9c4a05cdf65c"), Name: 'Bahrain' }
  { _id: ObjectId("626916c7e77e9c4a05cdf67e"), Name: 'Djibouti' }
  { _id: ObjectId("626916c7e77e9c4a05cdf684"),
    Name: 'Western Sahara' }
  { _id: ObjectId("626916c7e77e9c4a05cdf687"), Name: 'Algeria' }
  { _id: ObjectId("626916c7e77e9c4a05cdf694"), Name: 'Egypt' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6b1"), Name: 'Iraq' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6b2"), Name: 'Israel' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6b3"), Name: 'Jordan' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6bc"), Name: 'Kuwait' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6bd"), Name: 'Lebanon' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6c0"),
    Name: 'Libyan Arab Jamahiriya' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6d3"), Name: 'Morocco' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6ea"), Name: 'Oman' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6f7"), Name: 'Qatar' }
  { _id: ObjectId("626916c7e77e9c4a05cdf700"),
    Name: 'Saudi Arabia' }
  { _id: ObjectId("626916c7e77e9c4a05cdf704"), Name: 'Sudan' }
  { _id: ObjectId("626916c7e77e9c4a05cdf70d"), Name: 'Somalia' }
  { _id: ObjectId("626916c7e77e9c4a05cdf712"), Name: 'Syria' }
  { _id: ObjectId("626916c7e77e9c4a05cdf714"), Name: 'Chad' }
```

FIGURE 49 – Execution de la commande avec Mongo Shell

```
>_MONGOSH
> db.world.find({"OffLang.Language":{$in: ["Arabic"]}},{"Name":1})
< { _id: ObjectId("626916c7e77e9c4a05cdf653"),
    Name: 'United Arab Emirates' }
  { _id: ObjectId("626916c7e77e9c4a05cdf65c"), Name: 'Bahrain' }
  { _id: ObjectId("626916c7e77e9c4a05cdf67e"), Name: 'Djibouti' }
  { _id: ObjectId("626916c7e77e9c4a05cdf684"),
    Name: 'Western Sahara' }
  { _id: ObjectId("626916c7e77e9c4a05cdf687"), Name: 'Algeria' }
  { _id: ObjectId("626916c7e77e9c4a05cdf694"), Name: 'Egypt' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6b1"), Name: 'Iraq' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6b2"), Name: 'Israel' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6b3"), Name: 'Jordan' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6bc"), Name: 'Kuwait' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6bd"), Name: 'Lebanon' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6c0"),
    Name: 'Libyan Arab Jamahiriya' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6d3"), Name: 'Morocco' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6ea"), Name: 'Oman' }
  { _id: ObjectId("626916c7e77e9c4a05cdf6f7"), Name: 'Qatar' }
  { _id: ObjectId("626916c7e77e9c4a05cdf700"),
    Name: 'Saudi Arabia' }
  { _id: ObjectId("626916c7e77e9c4a05cdf704"), Name: 'Sudan' }
  { _id: ObjectId("626916c7e77e9c4a05cdf70d"), Name: 'Somalia' }
  { _id: ObjectId("626916c7e77e9c4a05cdf712"), Name: 'Syria' }
  { _id: ObjectId("626916c7e77e9c4a05cdf714"), Name: 'Chad' }
```

FIGURE 50 – Execution de la commande avec Mongo Shell

```python
def QST13(): # 13. Lister les pays pour lesquels l'Arabe est une langue officielle
    '''
    db.world.find({"OffLang.Language":{$eq:"Arabic"}},{"Name":1})
    '''
    countries =[]
    for post in (collection.find({"OffLang.Language":{'$eq':"Arabic"}},{"Name":1})):
        countries.append(post["Name"])
    return countries

```

FIGURE 51 – La fonction Utilisée

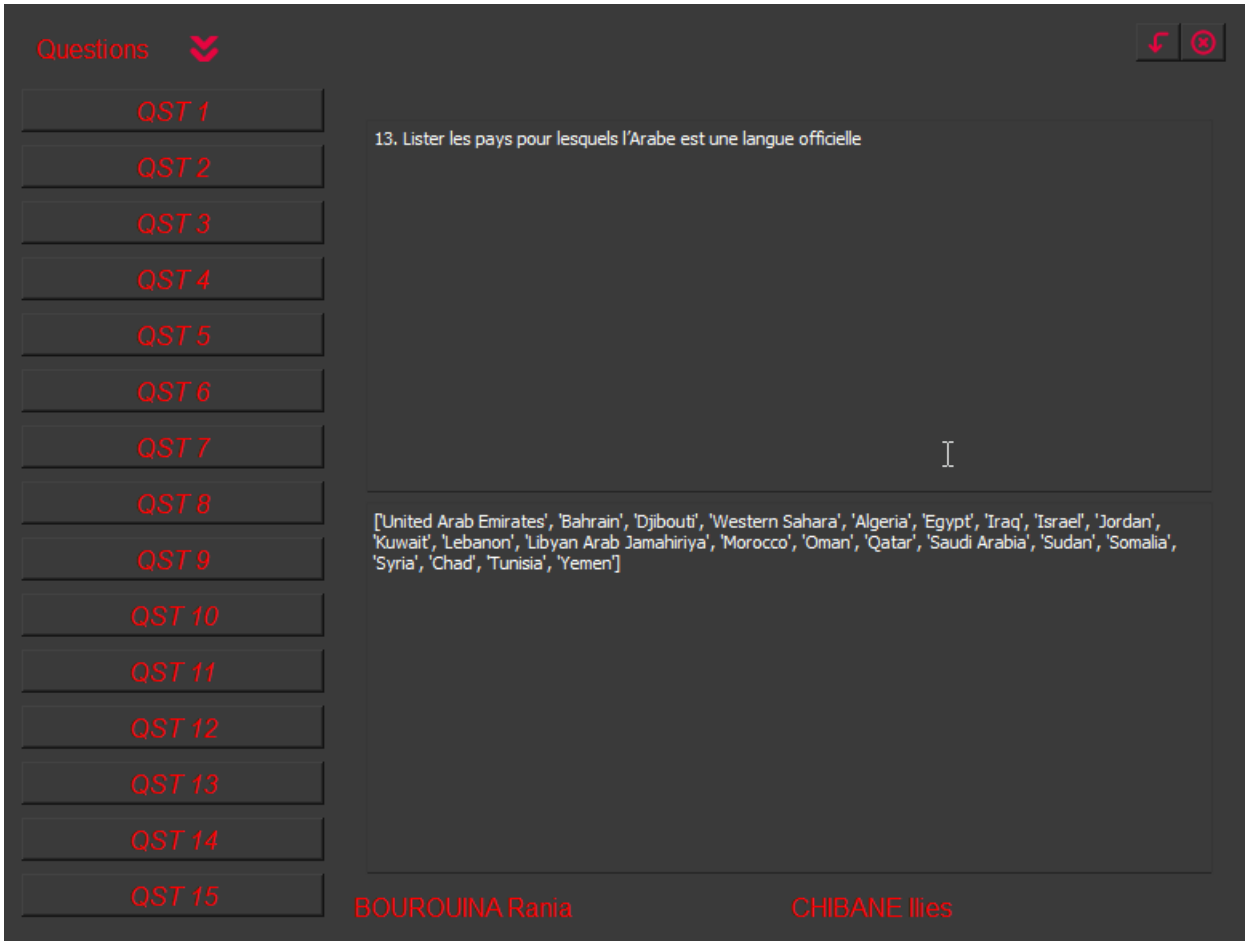


FIGURE 52 – Le résultat sur l'interface

2.14 14. Lister les 5 pays avec le plus de langues parlées

```
>_MONGOSH

> db.world.aggregate([
      {
          $addFields: { c : {$concatArrays: ["$OffLang","$NotOffLang" ] } }
      },
      {
          $addFields: { langs : {$size: { "$ifNull": [ "$c", [] ] } } }


      },
      {
          $sort: {"langs":-1}
      },
      { $limit : 5 },
      {
         $group: {
              _id:"$Name"
         }
      }
  ])
< { _id: 'China' }
  { _id: 'Russian Federation' }
  { _id: 'India' }
  { _id: 'United States' }
  { _id: 'Canada' }
BDD>
```

FIGURE 53 – Execution de la commande avec Mongo Shell

FIGURE 54 – La fonction Utilisée

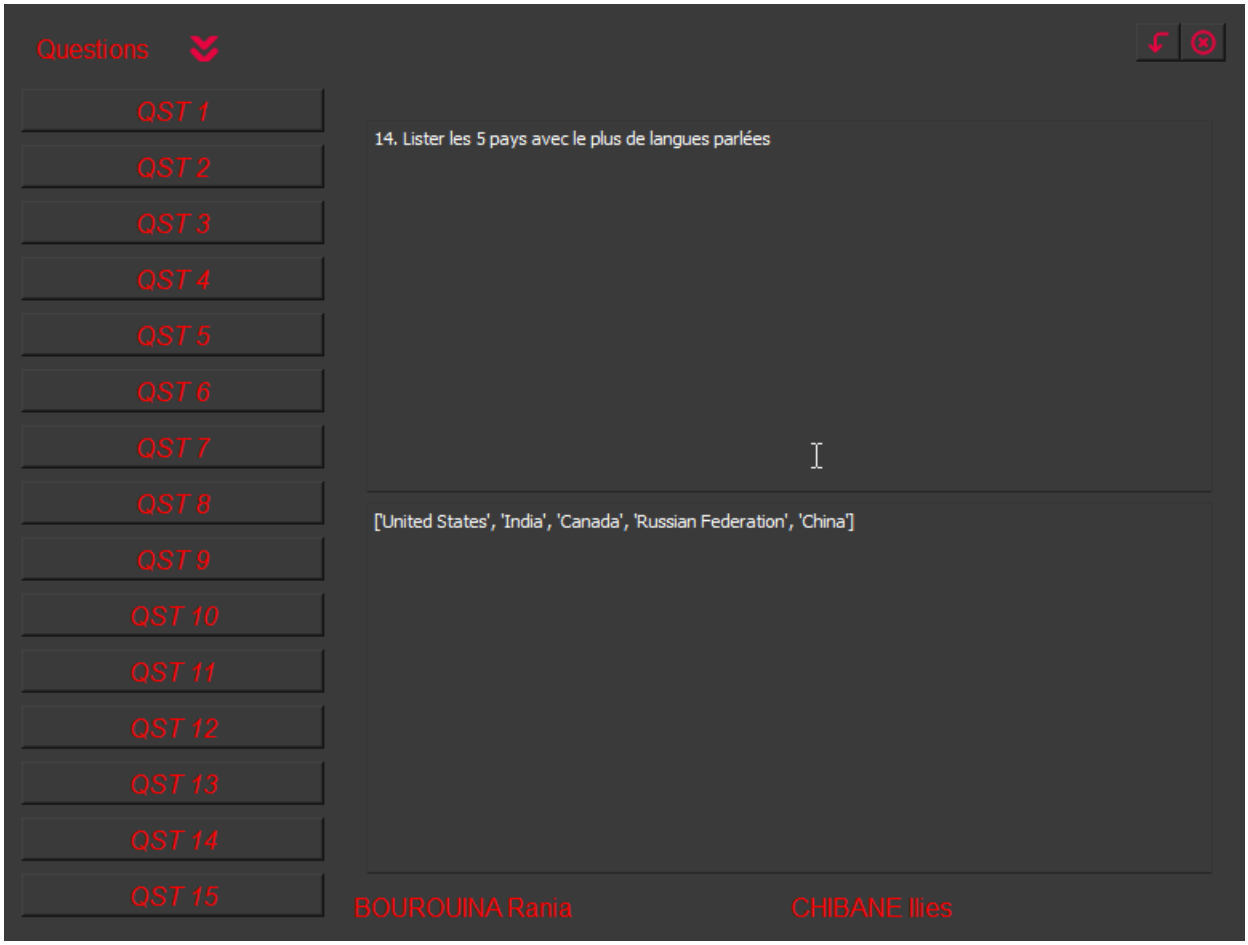


FIGURE 55 – Le résultat sur l'interface

2.15 15. Lister les pays pour lesquels la somme des populations des villes est supérieure à la population du pays.

```
>_MONGOSH
> db.world.aggregate([
    {
      $addFields: { somme : {$sum: "$Cities.Population"} }
    },
    {
      $addFields:{comp : {$cmp: ['$somme','$Population']}}
    },
    {$match: {comp:{$gt:1}}},
    {
      $project: {
          _id:"$Name",
          "Total Cities Population":"$somme",
          "Country Population":"$Population"
      }
    }
  ])
BDD>
```

FIGURE 56 – Execution de la commande avec Mongo Shell

```
def QST15(): # 15. Lister les pays pour lesquels la somme des populations des villes est supérieure à la population du pays.
    '''
        db.world.aggregate([
    {
        $addFields: { somme : {$sum: "$Cities.Population"}  }
    },
    {
        $addFields:{comp : {$cmp: ['$somme','$Population']}}
    },
    {$match: {comp:{$gt:1}}},
    {
      $project: {
            _id:"$Name",
            "Total Cities Population":"$somme",
            "Country Population":"$Population"
      }
    }
])
    '''
    countries =[]
    pipe = [    {
        '$addFields': { "somme" : {'$sum': "$Cities.Population"}  }
    },
    {
        '$addFields':{"comp" : {'$cmp': ['$somme','$Population']}}
    },
    {'$match': {"comp":{'$gt':1}}},
    {
      '$project': {
            "_id":"$Name",
            "Total Cities Population":"$somme",
            "Country Population":"$Population"
      }
    }]

    for post in collection.aggregate(pipeline=pipe):
        countries.append(post["_id"])

    return countries

```

FIGURE 57 – La fonction Utilisée

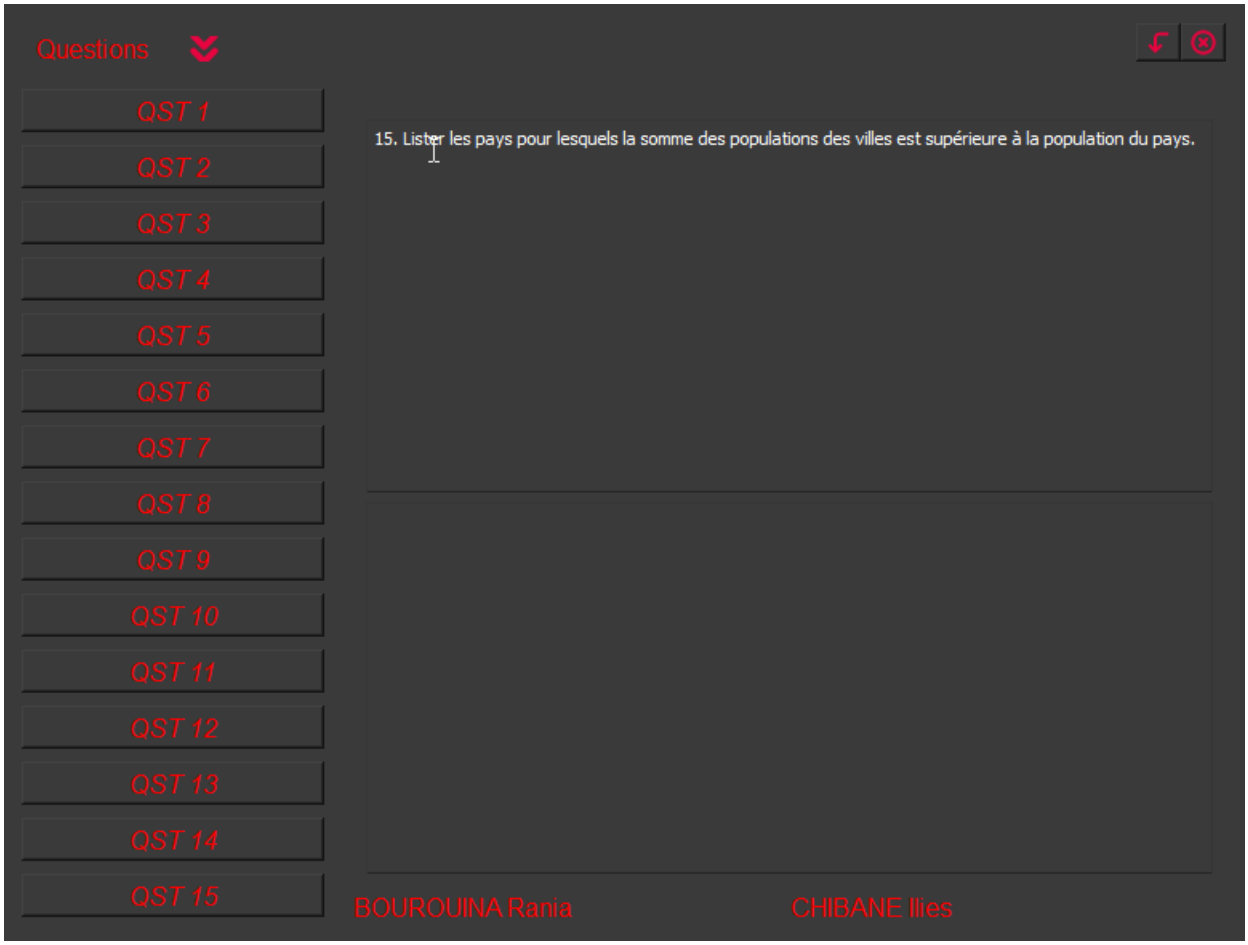


FIGURE 58 – Le résultat sur l'interface

Il n'y a aucun pays dont la somme des populations de ses villes est supérieure à sa population. Pour confirmer que ceci est vrai on exécute la commande suivante pour avoir les noms des pays qui ont une population supérieure à la somme des populations de ses villes.

```
>_MONGOSH
> db.world.aggregate([
    {
      $addFields: { somme : {$sum: "$Cities.Population"}  }
    },
    {
      $addFields:{comp : {$cmp: ['$somme','$Population']}}
    },
    {$match: {comp:{$eq:-1}}},
    {
      $project: {
            _id:"$Name",
            "Total Cities Population":"$somme",
            "Country Population":"$Population"
      }
    }
])
{ _id: 'Armenia',
  'Total Cities Population': 384400,
  'Country Population': 3520000 }
{ _id: 'American Samoa',
  'Total Cities Population': 5200,
  'Country Population': 68000 }
{ _id: 'Afghanistan',
  'Total Cities Population': 552100,
  'Country Population': 22720000 }
```

FIGURE 59 – Execution de la commande avec Mongo Shell