TP2

Le ministre de l'Enseignement supérieur cherche un Data Analyste afin d'exploiter leurs data(s)

Le besoin est formulé comment suit

Partie1 : Load Data

Question1 :

def getEtudiantsAndProf(path: String):Seq[DataFrame]

Cette fonction lit un fichier **JSON** et retourne deux DataFrames, un qui contient les informations des étudiants qui ont perçu une **bourse excellence** et un autre celles des professeurs récompensés

```
val dfEtudiant=getEtudiantsAndProf(path)(0)
dfEtudiant.show
```

▸ (2) Spark Jobs

```
+-----+-----------+-------------+------+----------+------+---------+
|Annee|Bourse_Exel|CodeFormation| IdEtu|Universite|Niveau|Fomration|
+-----+-----------+-------------+------+----------+------+---------+
| 2019|     100000|  Thies@M1@GL|   1ET|     Thies|    M1|       GL|
| 2020|     150000| Thies@M1@MSD|   2ET|     Thies|    M1|      MSD|
| 2019|      15000|  UCAD@L1@MPI|1EUCAD|      UCAD|    L1|      MPI|
| 2020|     140000| Thies@M1@MSD|   1ET|     Thies|    M1|      MSD|
| 2020|      30000|Thies@L2@MATH|   4ET|     Thies|    L2|     MATH|
| 2019|      20000|    UGB@M1@GL| 1EUGB|       UGB|    M1|       GL|
| 2019|      13000|    UGB@L3@LA| 2EUGB|       UGB|    L3|       LA|
| 2019|     131000|   UGB@M1@SES| 3EUGB|       UGB|    M1|      SES|
| 2017|      10000|  UCAD@M1@MSD|2EUCAD|      UCAD|    M1|      MSD|
| 2019|      10000| UCAD@M1@MATH|3EUCAD|      UCAD|    M1|     MATH|
| 2019|      33000| UCAD@L2@MATH|4EUCAD|      UCAD|    L2|     MATH|
+-----+-----------+-------------+------+----------+------+---------+
```

```
val dfProf=getEtudiantsAndProf(path)(1)
display(dfProf)
```

▸ (2) Spark Jobs

▸ dfProf: org.apache.spark.sql.DataFrame = [Annee: long, CodeFormations: array ... 1 more fields]

	Annee	CodeFormations	IdProf
1	2019	▸ ["Thies@M1@MSD", "Thies@M2@MSD"]	1PT
2	2020	▸ ["Thies@L2@MATH", "Thies@M1@GL"]	2PT
3	2019	▸ ["UGB@M1@GL"]	2PUGB
4	2017	▸ ["UCAD@M1@MSD", "UCAD@L2@MSD", "UCAD@M1@MSD"]	1PUCAD

Partie 2 Agrégation simple

Question1 :

def SumBourseByNivAndUniv(df:DataFrame):DataFrame

Cette fonction calcule la **somme** d'argent versée par **Niveau d'étude** pour chaque **université**

```
SumBourseByNivAndUniv(dfEtudiant).orderBy("Universite","Niveau").show
```

▸ (2) Spark Jobs

```
+----------+------+---------------+
|Universite|Niveau|Sum_Bourse_Exel|
+----------+------+---------------+
|     Thies|    L2|          30000|
|     Thies|    M1|         390000|
|      UCAD|    L1|          15000|
|      UCAD|    L2|          33000|
|      UCAD|    M1|          20000|
|       UGB|    L3|          13000|
|       UGB|    M1|         151000|
+----------+------+---------------+
```

Question2 :

def CountBourseUnivEachYear(df:DataFrame):DataFrame

Cette fonctionne calcule le **nombre** de bourse donné dans une **université** pour chaquee **années**

```
CountBourseUnivEachYear(dfEtudiant).show
```

▸ (6) Spark Jobs

```
+----------+----+----+----+
|Universite|2017|2019|2020|
+----------+----+----+----+
|       UGB|   0|   3|   0|
|     Thies|   0|   1|   3|
|      UCAD|   1|   3|   0|
+----------+----+----+----+
```

Partie3 agrégation avec windows partition

Question1 :

def TopOnBourseForUnivEachYear(df:DataFrame):DataFrame

Donne les informations de l'etudiant qui perçu la somme d'argent la plus importante pour chaque **université** lors d'une **année**

```
TopOnBourseForUnivEachYear(dfEtudiant).orderBy("Annee", "Universite").show
```

▸ (2) Spark Jobs

```
+-----+-----------+-------------+------+----------+------+---------+
|Annee|Bourse_Exel|CodeFormation| IdEtu|Universite|Niveau|Fomration|
+-----+-----------+-------------+------+----------+------+---------+
| 2017|      10000|  UCAD@M1@MSD|2EUCAD|      UCAD|    M1|      MSD|
| 2019|     100000|  Thies@M1@GL|   1ET|     Thies|    M1|       GL|
| 2019|      33000|UCAD@L2@MATH|4EUCAD|      UCAD|    L2|     MATH|
| 2019|     131000|   UGB@M1@SES| 3EUGB|       UGB|    M1|      SES|
| 2020|     150000|Thies@M1@MSD|   2ET|     Thies|    M1|      MSD|
+-----+-----------+-------------+------+----------+------+---------+
```

Question2 :

def DiffBetwMinAndNex(df:DataFrame):DataFrame

Calcule l'**écart** de bourse pour chaque **année** entre le **plus petit** et le **suivant** (plus proche)

```
DiffBetwMinAndNex(dfEtudiant).show
```

▸ (2) Spark Jobs

```
+-----+-----------+-------------+------+----------+------+---------+------+----------------+
|Annee|Bourse_Exel|CodeFormation| IdEtu|Universite|Niveau|Fomration|   lag|dif_between_next|
+-----+-----------+-------------+------+----------+------+---------+------+----------------+
| 2017|      10000|  UCAD@M1@MSD|2EUCAD|      UCAD|    M1|      MSD|  null|   c est  le min|
| 2019|      10000| UCAD@M1@MATH|3EUCAD|      UCAD|    M1|     MATH|  null|   c est  le min|
| 2019|      13000|    UGB@L3@LA| 2EUGB|       UGB|    L3|       LA| 10000|            3000|
| 2019|      15000|  UCAD@L1@MPI|1EUCAD|      UCAD|    L1|      MPI| 13000|            2000|
| 2019|      20000|    UGB@M1@GL| 1EUGB|       UGB|    M1|       GL| 15000|            5000|
| 2019|      33000|UCAD@L2@MATH|4EUCAD|      UCAD|    L2|     MATH| 20000|           13000|
| 2019|     100000|  Thies@M1@GL|   1ET|     Thies|    M1|       GL| 33000|           67000|
| 2019|     131000|   UGB@M1@SES| 3EUGB|       UGB|    M1|      SES|100000|           31000|
| 2020|      30000|Thies@L2@MATH|   4ET|     Thies|    L2|     MATH|  null|   c est  le min|
| 2020|     140000| Thies@M1@MSD|   1ET|     Thies|    M1|      MSD| 30000|          110000|
| 2020|     150000| Thies@M1@MSD|   2ET|     Thies|    M1|      MSD|140000|           10000|
+-----+-----------+-------------+------+----------+------+---------+------+----------------+
```

Partie 4 Agrégation Combinée

Question 1 :

def CubeUniversiteAndAnnee (df:DataFrame):DataFrame

Fait une agrégation combinée de l'année et **université** en utilisant le fonction **cube**

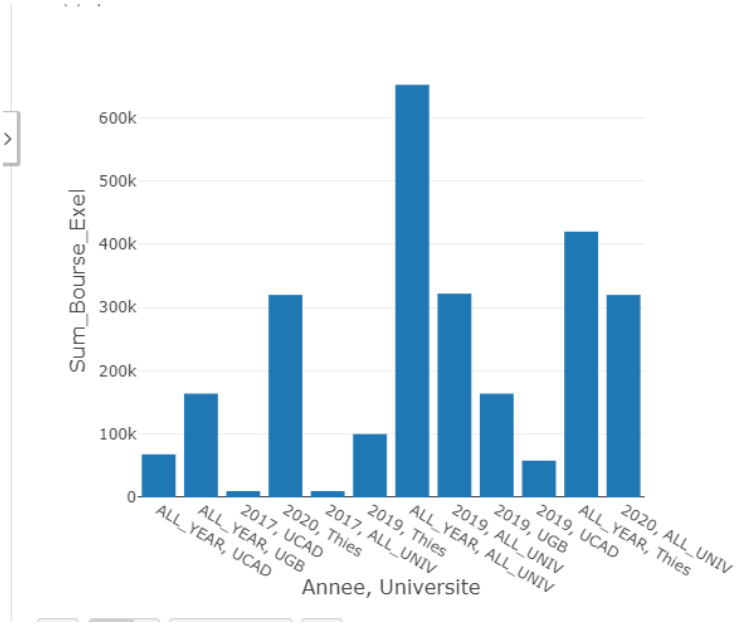
```
CubeUniversiteAndAnnee(dfEtudiant).show
```

▸ (2) Spark Jobs

```
+----------+--------+---------------+
|Universite|   Annee|Sum_Bourse_Exel|
+----------+--------+---------------+
|      UCAD|ALL_Year|          68000|
|       UGB|ALL_Year|         164000|
|      UCAD|    2017|          10000|
|     Thies|    2020|         320000|
|  ALL_Univ|    2017|          10000|
|     Thies|    2019|         100000|
|  ALL_Univ|ALL_Year|         652000|
|  ALL_Univ|    2019|         322000|
|       UGB|    2019|         164000|
|      UCAD|    2019|          58000|
|     Thies|ALL_Year|         420000|
|  ALL_Univ|    2020|         320000|
+----------+--------+---------------+
```

Question 2 :

a) Réaliser le graphe de la question Question1 en utilisant l'Api SQL



b) Tirez une conclusion

Partie5 Cross data

Question 1 :

def CheckProfRecomp(dfEtu:DataFrame, dfProf:DataFrame):DataFrame

La fonction fait la **jointure** du Dataframe **etudiant** et celui de **prof** afin de savoir quel est l'id du prof récompensé pour chaque étudiant

Si l'étudiant n'a pas de prof récompensé la valeur de l'idProf sera **ProfS_No_Recompensés**

```
CheckProfRecomp(dfEtudiant, dfProf).show
```

▸ (1) Spark Jobs

```
+-----+-----------+------+----------+------+---------+--------------------+
|Annee|Bourse_Exel| IdEtu|Universite|Niveau|Fomration|              IdProf|
+-----+-----------+------+----------+------+---------+--------------------+
| 2019|     100000|   1ET|     Thies|    M1|       GL|Profs_No_Recompensés|
| 2020|     150000|   2ET|     Thies|    M1|      MSD|Profs_No_Recompensés|
| 2019|      15000|1EUCAD|      UCAD|    L1|      MPI|Profs_No_Recompensés|
| 2020|     140000|   1ET|     Thies|    M1|      MSD|Profs_No_Recompensés|
| 2020|      30000|   4ET|     Thies|    L2|     MATH|                2PT|
| 2019|      20000| 1EUGB|       UGB|    M1|       GL|               2PUGB|
| 2019|      13000| 2EUGB|       UGB|    L3|       LA|Profs_No_Recompensés|
| 2019|     131000| 3EUGB|       UGB|    M1|      SES|Profs_No_Recompensés|
| 2017|      10000|2EUCAD|      UCAD|    M1|      MSD|              1PUCAD|
| 2019|      10000|3EUCAD|      UCAD|    M1|     MATH|Profs_No_Recompensés|
| 2019|      33000|4EUCAD|      UCAD|    L2|     MATH|Profs_No_Recompensés|
+-----+-----------+------+----------+------+---------+--------------------+
```