

BDLE 2024

date du document : 18/09/2024

✓ TP1 2024 Préparation de données

Préparation de données

Auteur :

runlin ZHOU 28717281

zhile ZHANG 21201131

✓ Préparation du TP


Vérifier que des ressources de calcul sont allouées à votre notebook (cf RAM de disque indiqués en haut à droite) . Sinon cliquer sur le bouton connecter pour obtenir des ressources.

Pour accéder directement aux fichiers stockées sur votre google drive. Renseigner le code d'authentification lorsqu'il est demandé

Ajuster le nom de votre dossier : MyDrive/ens/bdle/TP1

```
import os
from google.colab import drive
drive.mount("/content/drive")

drive_dir = "/content/drive/MyDrive/ens/bdle/TP1"
os.makedirs(drive_dir, exist_ok=True)
os.listdir(drive_dir)
```

 Mounted at /content/drive
[]

Commande permettant d'afficher le resultat d'une requete sous la forme d'un tableau interactif

```
!pip -q install itables
```

  1.4/1.4 MB 13.4 MB/s eta 0:00:0

```
from itables import init_notebook_mode
init_notebook_mode(all_interactive=True)
```



Redéfinir la fonction **display** pour afficher le resultat des requêtes dans un tableau

```
# affichage du résultat d'une requête

# pour spark
def display(query, n=30):
    return query.limit(n).toPandas()

# pour duckDB
# def display(query, n=30):
#     return query.limit(n).df()

# seulement dans colab (non préconisé en TP):
# from google.colab import data_table
# pour duckDB
# def display(query, n=100):
#     return data_table.DataTable(query.limit(n).df(), include_index=False, num
# pour spark
# def display(df, n=100):
#     return data_table.DataTable(df.limit(n).toPandas(), include_index=False, nu

# def display2(df, n=20):
#     pd.set_option('max_columns', None)
#     pd.set_option('max_colwidth', None)
#     return df.limit(n).toPandas()
```

Installer pyspark et findspark (l'installation dure environ 1 minute):

```
!pip install -q pyspark
!pip install -q findspark
```



317.3/317.3 MB 3.8 MB/s eta 0
Preparing metadata (setup.py) ... done
Building wheel for pyspark (setup.py) ... done

Démarrer la session spark

```
import os
import glob
```

```
pyspark_dir = glob.glob('/usr/local/lib/python*/dist-packages/pyspark')[0]
print("pyspark directory is", pyspark_dir)
os.environ["SPARK_HOME"] = pyspark_dir
os.environ["JAVA_HOME"] = "/usr"
```

➡ pyspark directory is /usr/local/lib/python3.10/dist-packages/pyspark

```
#!/find /usr/local/lib/python3.10/dist-packages/pyspark -name "*netty*"
```

```

# Principaux import
import findspark
from pyspark.sql import SparkSession
from pyspark import SparkConf

# pour les dataframe et udf
from pyspark.sql import *
from pyspark.sql.functions import *
from pyspark.sql.types import *
from datetime import *

# pour le chronomètre
import time

# initialise les variables d'environnement pour spark
findspark.init()

# Démarrage session spark
# -----
def demarrer_spark():
    local = "local[*]"
    appName = "TP"
    configLocale = SparkConf().setAppName(appName).setMaster(local).\
set("spark.executor.memory", "6G").\
set("spark.driver.memory", "6G").\
set("spark.sql.catalogImplementation", "in-memory")

    spark = SparkSession.builder.config(conf = configLocale).getOrCreate()
    sc = spark.sparkContext
    sc.setLogLevel("ERROR")

    spark.conf.set("spark.sql.autoBroadcastJoinThreshold", "-1")

    # On ajuste l'environnement d'exécution des requêtes à la taille du cluster (
    spark.conf.set("spark.sql.shuffle.partitions", "4")
    print("session démarrée, son id est ", sc.applicationId)
    return spark
spark = demarrer_spark()

```

🔄 session démarrée, son id est local-1727289454436

```

# on utilise 8 partitions au lieu de 200 par défaut
spark.conf.set("spark.sql.shuffle.partitions", "8")
print("Nombre de partitions utilisées : ", spark.conf.get("spark.sql.shuffle.pa

```

🔄 Nombre de partitions utilisées : 8

```
# Optionnel :  
# pour l'accès à spark UI : voir https://www.analyticsvidhya.com/blog/2020/11/a  
# !wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip  
# !unzip ngrok-stable-linux-amd64.zip  
# get_ipython().system_raw('./ngrok http 4050 &')  
# !curl -s http://localhost:4040/api/tunnels
```

Définir le tag **%%sql** pour pouvoir écrire plus simplement des requêtes en SQL dans une cellule

```
from IPython.core.magic import (register_line_magic, register_cell_magic, regis  
  
def removeComments(query):  
    result = ""  
    for line in query.split('\n'):  
        if not(line.strip().startswith("--")):  
            result += line + "\n"  
    return result  
  
@register_line_magic  
def sql(line, cell=None):  
    "To run a sql query. Use:  %%sql"  
    val = cell if cell is not None else line  
    tabRequetes = removeComments(val).split(";")  
    derniere = None  
    est_requete = False  
    for r in tabRequetes:  
        r = r.strip()  
        if len(r) > 2:  
            derniere = spark.sql(r)  
            est_requete = ( r.lower().startswith('select')or r.lower().startswith  
    if(est_requete):  
        return display(derniere)  
    else:  
        return print('ok')
```

```
# facultatif (non préconisé en TP)  
# %load_ext google.colab.data_table  
# %unload_ext google.colab.data_table
```

✓ Accès aux données

✓ URL pour l'accès aux datasets

```
# URL du dossier PUBLIC_DATASET contenant des fichiers de données pour les TP
# -----
# en cas de problème avec le téléchargement des datasets, aller directement sur
PUBLIC_DATASET_URL = "https://nuage.lip6.fr/s/LqD9N23kxrfHopr"
PUBLIC_DATASET=PUBLIC_DATASET_URL + "/download?path="

print("URL du dossier contenant les datasets ", PUBLIC_DATASET_URL)
```

➦ URL du dossier contenant les datasets <https://nuage.lip6.fr/s/LqD9N23kxrfH>

✓ Données de mobilité

Données issues du dataset YFCC

```
local_dir = "/local/data"
os.makedirs(local_dir, exist_ok=True)
os.listdir(local_dir)
```

➦ []

```

from urllib import request


# download dataset if not already downloaded
def download_file(web_dir, local_dir, file):
    local_file = local_dir + "/" + file
    web_file = web_dir + "/" + file
    if(os.path.isfile(local_file)):
        print(file, "is already stored")
    else:
        print("downloading from URL: ", web_file , "save in : " + local_file)
        request.urlretrieve(web_file , local_file)

# user visits
web_dir = PUBLIC_DATASET + "YFCC_POI_dataset_K_H_LIM/dataset_IJCAI_2015/data-ij"
download_file(web_dir, local_dir, "userVisits-Toro.csv")

# poi
web_dir = PUBLIC_DATASET + "YFCC_POI_dataset_K_H_LIM/dataset_IJCAI_2015/data-ij"
download_file(web_dir, local_dir, "POI-Toro.csv")

os.listdir(local_dir)

```

 downloading from URL: <https://nuage.lip6.fr/s/LqD9N23kxrfHopr/download?pat>
 downloading from URL: <https://nuage.lip6.fr/s/LqD9N23kxrfHopr/download?pat>
 ['POI-Toro.csv', 'userVisits-Toro.csv']


▼ Les visites

Lire les 2 premières lignes du fichier csv en python. Est ce que le fichier a une ligne d'entête ?
 Quel caractère délimite deux valeurs consécutives dans une ligne de données ?

```

f = open(local_dir + "/" + "userVisits-Toro.csv", "r")
print(f.readline()); print(f.readline())

```

 "photoID";"userID";"dateTaken";"poiID";"poiTheme";"poiFreq";"seqID"
 7941504100;"10007579@N00";1346844688;30;"Structure";1538;1

Lire le fichier des visites *sans* préciser le type des attributs. Par défaut, tous les attributs sont considérés comme étant de type string.

```
user_visits = spark.read.option("header", "True").option("delimiter", ";").from
user_visits.show(3)
user_visits.printSchema()
```



photoID	userID	dateTaken	poiID	poiTheme	poiFreq	seqID
7941504100	10007579@N00	1346844688	30	Structure	1538	1
4886005532	10012675@N05	1142731848	6	Cultural	986	2
4886006468	10012675@N05	1142732248	6	Cultural	986	2

only showing top 3 rows

root

```
-- photoID: string (nullable = true)
-- userID: string (nullable = true)
-- dateTaken: string (nullable = true)
-- poiID: string (nullable = true)
-- poiTheme: string (nullable = true)
-- poiFreq: string (nullable = true)
-- seqID: string (nullable = true)
```

Lire le fichier en précisant le schéma : nom et type des attributs


```

from pyspark.sql.types import StructType, StructField, StringType, IntegerType,

schema = StructType([
    StructField("photoID", StringType(), True),
    StructField("userID", StringType(), True),
    StructField("dateTaken", StringType(), True),
    StructField("poiID", IntegerType(), True),
    StructField("poiTheme", StringType(), True),
    StructField("poiFreq", IntegerType(), True),
    StructField("seqID", IntegerType(), True)
])

user_visits = spark.read.option("header", "True").option("delimiter", ";").csv(
user_visits.persist()
user_visits.createOrReplaceTempView("user_visits")
user_visits.show(10)
user_visits.printSchema()

```

```

⇒ +-----+-----+-----+-----+-----+-----+
| photoID|      userID| dateTaken|poiID| poiTheme|poiFreq|seqID|
+-----+-----+-----+-----+-----+-----+
|7941504100|10007579@N00|1346844688|  30|Structure|  1538|  1|
|4886005532|10012675@N05|1142731848|   6|Cultural|   986|  2|
|4886006468|10012675@N05|1142732248|   6|Cultural|   986|  2|
|4885404441|10012675@N05|1142732373|   6|Cultural|   986|  2|
|4886008334|10012675@N05|1142732445|   6|Cultural|   986|  2|
|4886009150|10012675@N05|1142916492|   6|Cultural|   986|  3|
|7054481539|10012675@N05|1319327174|  13|Cultural|   964|  4|
|6908387594|10012675@N05|1319328255|  13|Cultural|   964|  4|
|6908381912|10012675@N05|1319331463|  13|Cultural|   964|  4|
|6908398496|10012675@N05|1319331886|  13|Cultural|   964|  4|
+-----+-----+-----+-----+-----+-----+

```

only showing top 10 rows

```

root
|-- photoID: string (nullable = true)
|-- userID: string (nullable = true)
|-- dateTaken: string (nullable = true)
|-- poiID: integer (nullable = true)
|-- poiTheme: string (nullable = true)
|-- poiFreq: integer (nullable = true)
|-- seqID: integer (nullable = true)

```

Vérifier qu'il n'y a pas de date nulle

```

%%sql
select count(1) as nb_dates_nulles
from user_visits
where dateTaken is NULL

-- autre solution avec "case"
--select count(case when ..... then date end) as nb_date_nulles
--from user_visits

```



nb_dates_nulles
0

Vérifier qu'il n'y a aucune séquence associée à plusieurs utilisateurs

```

%%sql
select seqID
from user_visits
group by seqID
having count(poiID) >= 300

```



seqID
3935
4632
4635
3933
3011

✓ POI_sequence

La table **POI_sequence**(seqID, poiID). Les POI visités durant une séquence. Pour simplifier on compte chaque POI une seule fois par séquence et on ne considère pas l'ordre de visite des POI.

Rmq: il n'est pas nécessaire de préciser le userID pour identifier une séquence.

```
%%sql
create or replace temp view POI_sequence as
select seqID, poiID
from user_visits;

select * from POI_sequence
```



10 entries per page

Search:

seqID	poiID
1	30
2	6
2	6
2	6
2	6
3	6
4	13
4	13
4	13
4	13

Showing 1 to 10 of 30 entries

«

<

1

2

3

>

»

✓ Les lieux visités : POI

Ils sont appelés *Point Of Interest*

```
f = open(local_dir + "/" + "POI-Toro.csv", "r")
print(f.readline()); print(f.readline())
```



poiID;poiName;lat;long;theme

1;Air_Canada_Centre;43.64333; -79.37917;Sport

```
poi_schema = "poiID INT, poiName STRING, lat FLOAT, long FLOAT, theme STRING"
```

```
poi = spark.read.option("header", "True").option("delimiter", ";").csv(local_di  
poi.show(3)  
poi.printSchema()  
poi.createOrReplaceTempView("POI")
```

```
⇒ +-----+-----+-----+-----+-----+  
|poiID|      poiName|    lat|    long|theme|  
+-----+-----+-----+-----+-----+  
|    1| Air_Canada_Centre|43.64333|-79.37917|Sport|  
|    2|      BMO_Field|43.63278|-79.41861|Sport|  
|    3|Maple_Leaf_Gardens|43.66222|-79.38028|Sport|  
+-----+-----+-----+-----+-----+  
only showing top 3 rows
```

```
root  
|-- poiID: integer (nullable = true)  
|-- poiName: string (nullable = true)  
|-- lat: float (nullable = true)  
|-- long: float (nullable = true)  
|-- theme: string (nullable = true)
```

```
%%sql
cache table POI;
```

```
SELECT *
FROM POI
```



entries per page

Search:

poiID	poiName	lat	long	theme
1	Air_Canada_Centre	43.64333	-79.379173	Sport
2	BMO_Field	43.632778	-79.41861	Sport
3	Maple_Leaf_Gardens	43.66222	-79.38028	Sport
4	Rogers_Centre	43.641392	-79.389168	Sport
5	Woodbine_Racetrack	43.712524	-79.602043	Sport
6	Art_Gallery_of_Ontario	43.653889	-79.392776	Cultural
7	Hockey_Hall_of_Fame	43.646976	-79.377251	Cultural
8	Ripley's_Aquarium_of_Canada	43.642483	-79.386047	Cultural
9	Ontario_Science_Centre	43.716671	-79.338333	Cultural
10	Riverdale_Farm	43.66711	-79.361298	Cultural

Showing 1 to 10 of 30 entries

« < 1 2 3 > »

la liste des thèmes

```
%%sql
create or replace temp view themes as
select distinct theme
from POI
order by theme;

cache table themes;

select *
from themes
```



theme ◆

Amusement

Beach

Cultural

Shopping

Sport

Structure

✓ Exercice 1

✓ 1) Les 10 POI les plus photographiés

indications: on peut compléter une requête SQL avec *LIMIT N* pour n'obtenir que N tuples du résultat. Pour une requête dont le résultat est trié (order by) on obtient les N premiers tuples, sinon on obtient N tuples quelconques du résultat.

```
%%sql
select poiID, count(poiID) as nbPhoto
from user_visits
group by poiID
order by nbPhoto desc
limit 10
```



poiID	nbPhoto
11	4139
22	3603
21	3591
16	3553
1	3506
4	3056
7	2053
23	1866
8	1736
25	1701

✓ 2) Les visites avec date détaillée

Définir la table Visite_date(userID, seqID, poiID, date, annee, mois, jour, heure) contenant les visites avec la date détaillée composée des attributs : année, mois, jour, heure. Indication : la date est initialement au format "unix". Voir la fonction de conversion de date *from_unixtime* et la fonction *extract(... from ...)*.

```
%%sql
```

```
create or replace temp view Visite_Date as
select userID, seqID,
from_unixtime(dateTaken) as date,
extract(year FROM date) as annee,
extract(month FROM date) as mois,
extract(day FROM date) as jour,
extract(hour FROM date) as heure,
extract(minute FROM date) as minute,
cast(extract(second FROM date) as int) as seconde
from user_visits
order by seqID, date;
```

```
select * from Visite_Date
```



10 entries per page

Search:

userID	seqID	date	annee	mois	jour	heure
10007579@N00	1	2012-09-05 11:31:28	2012	9	5	1
10012675@N05	2	2006-03-19 01:30:48	2006	3	19	
10012675@N05	2	2006-03-19 01:37:28	2006	3	19	
10012675@N05	2	2006-03-19 01:39:33	2006	3	19	
10012675@N05	2	2006-03-19 01:40:45	2006	3	19	
10012675@N05	3	2006-03-21 04:48:12	2006	3	21	
10012675@N05	4	2011-10-22 23:46:14	2011	10	22	2
10012675@N05	4	2011-10-23 00:04:15	2011	10	23	
10012675@N05	4	2011-10-23 00:57:43	2011	10	23	
10012675@N05	4	2011-10-23 01:04:46	2011	10	23	

Showing 1 to 10 of 30 entries

« < 1 2 3 > »

✓ 3a) Le nombre de POI par utilisateur

Le résultat (userID, nbPOI) est trié par nombre décroissant de POI.


```
%%sql
SELECT userID, COUNT(DISTINCT poiID) AS nbPOI
FROM user_visits
GROUP BY userID
ORDER BY nbPOI DESC
```



10 entries per page

Search:

userID	nbPOI
34211328@N00	24
43139087@N00	23
84987970@N00	22
30624156@N00	20
63677124@N07	19
20741443@N00	19
48784629@N00	17
24854893@N00	16
59525924@N05	15
11191102@N07	15

Showing 1 to 10 of 30 entries

« ‹ 1 2 3 › »

✓ 3b) Le nombre de POI par séquence

Définir la table Seq3plus(seqID, nbPoi) correspondant aux séquences qui contiennent au moins 3 POI distincts. Afficher le résultat trié par nombre de POI décroissant puis numéro de séquence croissant.

```
%%sql
create or replace temp view Seq3plus as
select seqID, COUNT(DISTINCT poiID) as nbPOI
from user_visits
GROUP BY seqID
;

select * from seq3plus
order by nbPOI desc, seqID;
```



10 entries per page

Search:

seqID	nbPOI
298	13
4961	10
4351	9
5964	9
510	8
525	8
5369	8
5981	8
454	7
681	7

Showing 1 to 10 of 30 entries

«

<

1

2

3

>

»

4) Trajectoire

a) Définir une UDF qui prend en entrée une liste de couples (date, poi) et qui retourne une liste de Poi triés par date croissante. Le resultat ne contient pas 2 Poi consécutifs identiques.

```
def trierPOI(listeCouples: List[Tuple[int, int]]) -> List[int]:
    if listeCouples is None:
        return []
    listeCouples.sort(key=lambda x: x[0])
    result = []
    for i, (_, poi) in enumerate(listeCouples):
        if i == 0 or poi != listeCouples[i-1][1]: # Comparer avec le PoI précédent
            result.append(poi)
    return result

spark.udf.register("trierPOI", trierPOI, ArrayType(IntegerType()))

# test local
print(trierPOI([(13,0),(10,2), (12,1), (14,2),(11,2)]))
```

⇒ [2, 1, 0, 2]

Définir la vue Trajectoire(userID, seqID, listPOI).

listPOI est la liste des POI visités, pendant la séquence seqID, dans l'ordre chronologique

Indications: pour la sequence 687 la listPOI est [7,16,4,8,4,16]

```

%%sql
create or replace temp view Trajectoire as
select userID, seqID, trierPOI(collect_set((dateTaken, poiID))) as listePOI
from user_visits
group by userID, seqID
;

cache table Trajectoire;

--exemple
select * from Trajectoire
where size(listePOI) = 6
order by seqID

```



10 entries per page

Search:

userID	seqID	listePOI
14617133@N05	384	[22, 28, 22, 28, 22, 28]
20456447@N03	687	[7, 16, 4, 8, 4, 16]
20741443@N00	729	[23, 21, 22, 7, 28, 23]
27168489@N00	1358	[16, 8, 28, 22, 23, 21]
29352917@N00	1524	[29, 30, 16, 29, 16, 29]
30624156@N00	1598	[28, 7, 30, 28, 22, 27]
32827327@N03	1786	[20, 6, 25, 11, 30, 24]
33547369@N00	1877	[23, 21, 23, 22, 23, 22]
34128007@N04	1912	[8, 16, 4, 1, 29, 16]
34314322@N00	1993	[30, 22, 28, 29, 30, 16]

Showing 1 to 10 of 20 entries

«

<

1

2

>

»

✓ 5) Transitions

5a) Définir la table Transitions(poi1, poi2, nbTrans) avec nbTrans étant le nombre de déplacements directs de poi1 à poi2.

Deux POI apparaissant successivement (poi_1 suivi de poi_2) dans une séquence forment un déplacement direct entre poi_1 et poi_2 . Une séquence contenant un seul POI ne correspond à aucun déplacement.

Indication1: définir une UDF transition(listePoi) qui retourne une liste de couples de Pils (p_i , p_j).

```
def transition(listePOI):
    result = []
    for i in range(1, len(listePOI)):
        el = (listePOI[i-1], listePOI[i])
        result.append(el)
    return result

spark.udf.register("transition",
                    transition,
                    ArrayType(StructType([
                        StructField("p1", IntegerType()),
                        StructField("p2", IntegerType())
                    ])))

# test local
print(transition([2,6,1,6]))
```

⇒ [(2, 6), (6, 1), (1, 6)]

```

%%sql
create or replace temp view Transition1 as
select seqID, explode(transition(listePOI)) as transition
from Trajectoire
where size(listePOI)>1
order by seqID
;

select t.seqID, t.transition.p1, t.transition.p2
from Transition1 t

```



entries per page

Search:

seqID	p1	p2
8	23	24
33	21	25
58	7	11
58	11	27
59	11	27
67	28	23
67	23	22
67	22	28
71	22	28
71	28	23

Showing 1 to 10 of 30 entries

«

<

1

2

3

>

»

```

%%sql
create or replace temp view Transition2 as
select t.transition.p1 as poi1, t.transition.p2 as poi2, count(*) as nbtransiti
from Transition1 t
group by t.transition.p1 , t.transition.p2;

select * from Transition2
order by nbtransition desc;

```



10 entries per page

Search:

poi1	poi2	nbtransition
23	21	84
22	28	69
28	22	66
21	23	60
7	30	56
30	7	51
28	23	44
8	16	41
16	4	40
16	8	39

Showing 1 to 10 of 30 entries

«

<

1

2

3

>

»

b) Définir la vue

Transition_relative(poi1, poi2, p)

avec p étant un taux de transition relatif au nombre total de transitions partant d'un POI.

Indication: vous pouvez commencer par créer une vue

Total_transition(poi, t)

avec t étant le nombre total de transitions par poi.

```
%%sql
create or replace temp view Total_transition as
select poi1, sum(nbtransition) as t
from Transition2
group by poi1
order by poi1;

select * from Total_transition
```



entries per page

Search:

poi1	t
1	44
2	18
3	24
4	60
6	61
7	152
8	93
9	1
10	4
11	67

Showing 1 to 10 of 27 entries

«

<

1

2

3

>

»


```
%%sql
```

```
create or replace temp view Transition_relative as  
select t_t.poi1, t_t.poi2, t.nbtransition/t_t.t as rel  
from Total_transition t_t join Transition2 t on t_t.poi1 = t.poi1  
order by poi1, poi2;
```

```
select * from Transition_relative  
order by rel desc
```



entries per page

Search:

poi1	poi2	rel
9	11	1
14	17	0.8125
10	21	0.75
17	14	0.727273
2	17	0.666667
15	19	0.625
4	16	0.583333
27	11	0.513514
3	21	0.5
20	6	0.5

Showing 1 to 10 of 30 entries

«

<

1

2

3

>

»

✓ 6) DuréeVisitePOI

La durée moyenne de visite d'un POI. Indication, pour une série d'événements consécutifs associés à un même POI dans une séquence, la durée de visite est la différence de date entre le 1er et le dernier événement.

```

def dureeVisite(liste_couples):
    result = []
    if not liste_couples or len(liste_couples) == 1:
        return result

    # init : slow pointer starts at the first element of the list
    slow = 0
    n = len(liste_couples)

    # 快指针从第一个元素遍历到列表末尾
    # fast pointer traverses from index 1 to the end of the list (len(listecoup
    for fast in range(1, n):
        # 如果当前poi不同, 计算访问时长
        # if the poiID is different
        if liste_couples[fast][0] != liste_couples[slow][0]:
            # calculate the duration of this visite
            duration = liste_couples[fast - 1][1] - liste_couples[slow][1]
            if duration > 0:
                result.append((liste_couples[slow][0], duration))
            # 更新慢指针到新poi的起始点
            # update the position of pointers
            slow = fast

    # 处理最后一段连续相同poi的情况
    # in case of consecutive identical poi in the last segment
    duration = liste_couples[-1][1] - liste_couples[slow][1]
    if duration > 0:
        result.append((liste_couples[slow][0], duration))

    return result

spark.udf.register("dureeVisite", dureeVisite, ArrayType(
    StructType(
        [StructField("poiID", Integ
        StructField("duree", Integ

print(dureeVisite([(102, 8), (102, 11), (106, 14), (101, 15), (101, 17), (102,

```

➡ [(102, 3), (101, 2), (102, 4)]

```

%%sql
create or replace temp view DureeVisitePOI as
select
    userID,
    seqID,
    explode(dureeVisite(collect_list((poiID,cast(dateTaken as Integer)))))) as F
from user_visits
group by seqID, userID;

select *
from DureeVisitePOI d
order by userID, seqID

```



10 entries per page

Search:

userID	seqID	POI_duree
10012675@N05	2	(6, 597)
10012675@N05	4	(13, 5674)
10014440@N06	5	(24, 257)
10014440@N06	6	(23, 1065)
10014440@N06	7	(23, 573)
10014440@N06	8	(23, 1615)
10014440@N06	8	(24, 1359)
10014440@N06	9	(24, 296)
10014440@N06	10	(25, 7991)
10014440@N06	11	(17, 31482)

Showing 1 to 10 of 30 entries

«

<

1

2

3

>

»

```
%%sql
create or replace temp view DureeMoyenneVisitePOI as
select
    POI_duree.poiID as poi,
    AVG(POI_duree.duree) / 60 as duree_visite_moyenne -- transform the seconds
from DureeVisitePOI
group by POI_duree.poiID
order by duree_visite_moyenne desc;

select * from DureeMoyenneVisitePOI;
```



10 entries per page

Search:

poi	duree_visite_moyenne
17	143.449495
4	143.393171
9	122.456481
1	113.289239
8	105.01381
12	84.107609
16	84.08713
2	81.938739
11	81.0327
10	68.782083

Showing 1 to 10 of 29 entries

«

<

1

2

3

>

»

✓ 7) Durée moyenne de déplacement entre deux POIs.

Définir la vue **DureeDeplacement** (poi1, poi2, duree) calculant la duree moyenne de déplacement entre deux POI consécutifs dans une séquence.

开始借助 AI 编写或生成代码。

✓ Exercice 2

Charger les données de Geonames

```
import zipfile

#geonames
web_dir = PUBLIC_DATASET + "geonames_ALL"

download_file(web_dir, local_dir, "allCountries.zip")

#unzip
local_file = local_dir + "/" + "allCountries.txt"
if(os.path.isfile(local_file)):
    print(local_file, "is already stored")
else:
    with zipfile.ZipFile(local_dir + "/" + "allCountries.zip", 'r') as zip_ref:
        zip_ref.extractall(local_dir)

os.listdir(local_dir)
```

➡ downloading from URL: [https://nuage.lip6.fr/s/LqD9N23kxrfHopr/download?pat\['POI-Toro.csv', 'userVisits-Toro.csv', 'allCountries.zip', 'allCountries.txt'\]](https://nuage.lip6.fr/s/LqD9N23kxrfHopr/download?pat['POI-Toro.csv', 'userVisits-Toro.csv', 'allCountries.zip', 'allCountries.txt'])

```
geonames = spark.read.option("header", "False").option("delimiter", "\t").format("csv").load(local_dir)
geonames.createOrReplaceTempView("geonames")
geonames.show(3)
```

➡

_c0	_c1	_c2	_c3
2986043	Pic de Font Blanca	Pic de Font Blanca	Pic de Font Blanc... 42.
2994701	Roc Mélé	Roc Mele	Roc Mele,Roc Mele... 42.
3007683	Pic des Langounelles	Pic des Langounelles	Pic des Langounelles 42.

only showing top 3 rows

✓ 1) Geonames

Définir le schéma, limité au 9 premiers attributs donnant des informations sur l'identifiant d'un POI, ses noms, sa position GPS, le code du pays..., d'après le document [readme.txt](#) dans le dossier [geonames_ALL](#)

✓ 1a) schéma de Geonames

Définir Geonames2 en précisant le schéma : nom et type des attributs. Se limiter au 9 premiers attributs donnant des informations sur l'identifiant d'un POI, ses noms, sa position GPS, le code du pays...,), d'après le document [readme.txt](#) dans le dossier [geonames_ALL](#)

```

%%sql
create or replace temp view Geonames2 as
select cast(_c0 as long) as geonameID,
_c1 as name,
-- _c2 as ascii_name,
_c3 as alternate_name,
cast(_c4 as double) as latitude,
cast(_c5 as double) as longitude,
_c6 as feature_class,
_c7 as feature_code,
_c8 as country_code
from Geonames;

select * from Geonames2

```



10 entries per page

Search:

geonameID	name	alternate_name
2986043	Pic de Font Blanca	Pic de Font Blanca,Pic du Port
2994701	Roc Mélé	Roc Mele,Roc Meler,Roc Mélé
3007683	Pic des Langounelles	Pic des Langounelles
3017832	Pic de les Abelletes	Pic de la Font-Negre,Pic de la Font-Nègre,Pic de
3017833	Estany de les Abelletes	Estany de les Abelletes,Etang de Font-Negre,Ét
3023203	Port Vieux de la Coume d'Ose	Port Vieux de Coume d'Ose,Port Vieux de Cour
3029315	Port de la Cabanette	Port de la Cabanette,Porteille de la Cabanette
3034945	Port Dret	Port Dret,Port de Bareites,Port de las Bareytes,f
3038814	Costa de Xurius	None
3038815	Font de la Xona	None

Showing 1 to 10 of 30 entries

« < 1 2 3 > »

1b) Extrait pour le Canada

Définir la vue Geonames_canada pour les POI situés au Canada.

```
%%sql
create or replace temp view Geonames_canada as
select geonameID, name, alternate_name, latitude, longitude, feature_class, fea
from Geonames2
where ...

cache table Geonames_canada;

select * from Geonames_canada;
```

```
-----
ParseException                                Traceback (most recent call last)
<ipython-input-44-9bd04fd961e5> in <cell line: 1>()
----> 1 get_ipython().run_cell_magic('sql', '', 'create or replace temp
view Geonames_canada as\nselect geonameID, name, alternate_name, latitude,
longitude, feature_class, feature_code\nfrom Geonames2\nwhere ...\n\ncache
table Geonames_canada;\n\nselect * from Geonames_canada;\n')

----- 5 frames -----
/usr/local/lib/python3.10/dist-
packages/pyspark/errors/exceptions/captured.py in deco(*a, **kw)
    183             # Hide where the exception came from that shows a
non-Pythonic
    184             # JVM exception message.
--> 185             raise converted from None
    186         else:
    187             raise

ParseException:
[PARSE_SYNTAX_ERROR] Syntax error at or near '.'.(line 4, pos 6)

== SQL ==
create or replace temp view Geonames_canada as
select geonameID, name, alternate_name, latitude, longitude, feature_class,
feature_code
from Geonames2
where ...
-----^^^
```

2) Association entre les POI et Geonames

Compléter les POI avec des attributs de geonames (par exemple name et [feature code](#))

