NOM BOUKHELOUA

Prénom Rémi

Promo 2020

Date

19

BOUKHELOUA Rémi
M1-2018-TD BD

MATIÈRE Data Structuring

1. Untidy data examples:

- if their are variables in column names.
  for example observations from different dates.

	2015	2016
Paris	3	4

It should be

	year	obs
Paris	2015	3
Paris	2015	4.

(We do it with stack.

- if you have a same type of observation over several tables :

table A

Paris	2015	3

table B

Paris	2016	4.

It should be in the same table; we use concatenation.

- if you have different type of observation in a single table. For example, temperatures and pressures. We should normalize it to be in two different tables.
- if you have several variables in the same column For example: city and Country. They should be in two different columns (thumbs to split).

2. ACID stands for Atomicity, Consistency, Isolation, Durability. It means that a transaction is atomic, so ~~while it is performed, nothing can occur~~ it's considered as one point in time. ~~We should not have~~ While it's performed, nothing else can modify the part it modifies (Isolation).

4. MongoDB hierarchy is Database - Collection - Document.

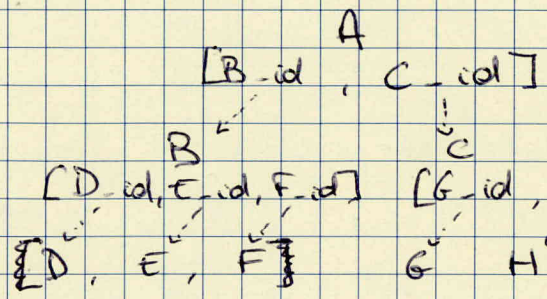
5. In order to model a "Foreign key" in MongoDB, we use the ids of document we create. How we model it is dependent of the relation we have.

If it's a 1 to 1: We can store the id of the first one in the second document. ~~If we are sure that We are a~~

If it's a 1 to Many: We can either store the id of the 1 in each document of the Many or store all ids of the many in a map / array in the 1. ~~To extend~~ In the 2nd choice, in order to add one more relation to the one we will only push the new document's id in the array.

If it's a Many to Many: we may do the same thing that we used in 1 to Many but as much time as we ~~did~~ have doublons in one of the Many sides of the relation.

To store hierarchical data such as list of employees with management hierarchy, we will use the 1 to Many in order to form a "tree":



Other wise we could store information of B and C in an array in A and within B information we would store D, E and F information and so on...

{ ... , 'manage': [{ ... , 'manage': [{ ... }, { ... }, { ... }] }] }
A info B info D info E info F info

~~But this is not considered~~ This solution is more complex and heavy but makes queries quicker. Overall I prefer the first one even though joins may not be the best.

6. We use MongoDB through PyMongo where db.coll is the collection storing the given data.

a. db.coll.find({"Keywords": "python"}) will give us a cursor to go through in order to fetch the elements.

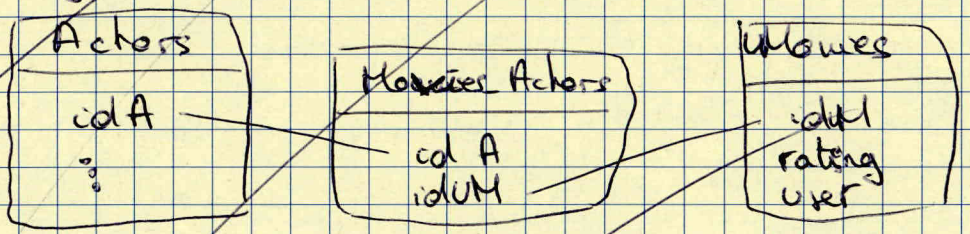
b. db.coll.find({"author_name": {"$eq": "A.BC"}})

c. db.coll.update({"author": {"eq": "DEF"}}, {"time": 100 * "time"})
We may want to add a third parameter {"multi": "true"} if there are several documents with DEF Author.

7. One of the difference between XML and JSON is that XML can store informations such as variable names or versions in its (balises in French) meanwhile JSON can't.

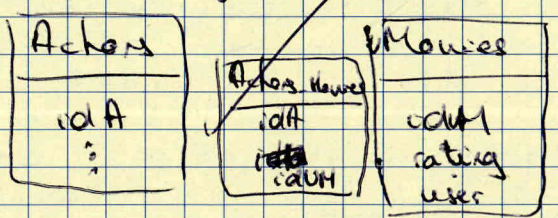
8. We have several possibilities depending on what we want to do with it and how large is the dataset:

- We could use SQL database if the dataset is not to big, we would store it like that:



It would be the easiest way to do it and would allow us to do every business questions we need if we only focus on Movies, Actors or the Ratings.

- If the number of record is to import we could decide to store it in HBase through HDFS and use HiveQL to querry it:



NOM BOURHELOUA

Prénom Rémi

Promo 2020

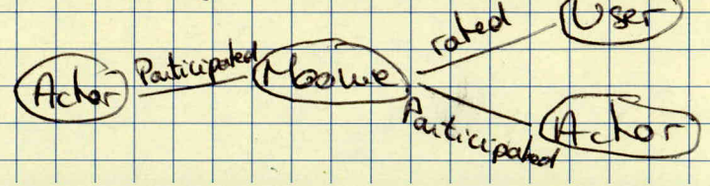
Date 2

MATIÈRE Data Structuring (2).

8. Model will depend on how large the dataset is and what we want to focus on:

- If we want to have an easy and simple view of the dataset, we would use SQL with the following schema:

- If the dataset is to large, we would rather use HBase and query it through Hive. Both are easy to join and look for lists of movies and counting their reviews or making mean of rates given by a single user.
- We could also use MongoDB or any document oriented database. That would be more interesting if we wanted to focus on user ratings or on actors. We would use same implementations as described in Q5: either simulating "Foreign Key" and then joining or integrating in an Actor document all movies an Actor's document all movies he participated in and the their ratings. It would be easier to search about how well defining the actor's domain and how well/bad he is regarding user's ratings.
- Finally we could use a Graph database such as Neo4J to focus mainly on links. It would be something like that:



we can easily figure a relation between two movies, actors or users. We could also search movies to propose to one user based on the ones that other user liked.

9. Wikipedia webpages are semi-structured. As it's an important amount of data, we may not be able to store in SQL. However it would not fit very well as elements are not always the same.
So, we should go either to MongoDB storing a webpage as a document: { 'title': ..., '_id': ...,
'link': ...,
'topic': ..., ... }
and the references to other webpages would be stored thanks to '_id' and images will store their direction in the system.
Otherwise, as Wikipedia webpages are html, or through XML for a dictionnary, we could put them in a File System such as Hadoop which handles semi-structured datas. Then we will be able to use Hive or Spark to process it and find out easily what we need.
HDFS would fit well because it has been designed for handling such type of data and large datasets.

efrei PARIS
Ecole d'ingénieurs du numérique

Ne rien inscrire dans ce cadre

Prénom ....Rémi.....

Nom .....BOUKHELOUA.....

Promotion .....2020.....

Groupe .....Big Data.....

M1

Data structuring and NoSQL databases

ST2DST

DE - 1h45 min

Date Horaire

Sujet proposé par :

Calculatrice autorisée : ☐ **OUI** ☒ **NON**

Documents autorisés : ☐ **OUI** ☒ **NON** **Type de documents :**

Ordinateur portable autorisé : ☐ **OUI** ☒ **NON**

Internet : ☐ **OUI** ☒ **NON**

Traducteur électronique, dictionnaire : ☐ **OUI** ☐ **NON**

Consigne :

Merci de restituer uniquement : **les copies quadrillées à rendre accompagnées de l'annexe**

Rappel :

- Tous les appareils électroniques (téléphones portables, ordinateurs, tablettes, montres connectées ...) doivent être éteints et rangés.
- Il est interdit de communiquer.
- Toute fraude ou tentative de fraude fera l'objet d'un rapport de la part du surveillant et sera sanctionnée par la note zéro, assortie d'une convocation devant le conseil de discipline. Aucune contestation ne sera possible. Tous les documents et supports utilisés frauduleusement devront être remis au surveillant.
- Aucune sortie de la salle d'examen ne sera autorisée avant la moitié de la durée de l'épreuve.

Ne rien inscrire dans ce cadre

1. Provide 4 different examples of « Un-tidy » data (you can sketch them if necessary)
2. Explain the meaning of ACID letters for SQL transactions
3. Provide examples of XML queries and explain what they do
4. What is MongoDB schema hierarchy? (like Database/Table/Row for SQL)
5. How to model a "Foreign Key" in MongoDB? What are different modelling options? For example, how to store hierarchical data, like a list of employees with management hierarchy?
6. For a data

{'_id': ObjectId('5ba207f488811f06f433e7f7'), 'author': 'ABC', 'author_name': 'A. B C', 'icon_filename': 'icon1.png', 'text': 'Some text', 'time': 1234487, 'keywords': ['test', 'python', 'mongodb'], 'responses': [{'author': 'DEF', 'author_name': 'D. E F', 'icon_filename': 'icon2.png', 'text': 'thanks a lot!'}, {'author': 'ABC', 'author _name': 'A. BC', 'icon_filename': 'icon1.png', 'text': 'cheers!', 'responses': [{'aut hor': 'ABC', 'author_name': 'D. EF', 'text': 'Thanx again'}]}]}

write the Mongo query to

   a. Fetch only the elements with a keyword 'python'
   b. Fetch only the elements with "author_name" == 'A.BC'
   c. Increase time by 100 for author "DEF"

7. What is a difference between XML and JSON?
8. You have data on
   - Users watching and rating movies
   - Actors playing in the movies

   What are your different options to model these data and store it in a single SQL or NoSQL database (sketch it if necessary)? What databases can you use? Provide examples of "business" questions that are better adapted to each storage/modeling option.

9. You have downloaded a copy of Wikipedia (or of an encyclopedia), how would you store it? Why? What are other options and how organize in this case (SQL, NoSQL, or something else)?