EXAMEN CDI 2019(7h)

Partie 1 :

Questions

-

Réponses

Q1: Définir Big data

«

Big data

» désigne des ensembles de données devenus trop volumineux pour les

capacités humains d’analyse ou même des outils informatiques classiques de

gestion de base de donnée ou de l’information.

Q2 : Définir persistance des données

La persistance des données est tout simplement le fait que des données soient

conservées malgré la fermeture d’un programme (sauvegardes, préférences

utilisateur ...)

Q3 : Qu'est ce qu'une Base de données relationnelle

Une base de données relationnelle est une base de données sous forme de tables à

deux entrées

: clef/valeur. Les tables peuvent de plus être reliées entre elles par des

foreign key.

Q4 : ACID : propriété des bases de données relationnelles

Les propriétés ACID

(atomicité, cohérence, isolation et durabilité) sont un ensemble

de

propriétés

qui garantissent qu'une transaction informatique est exécutée de façon

fiable. Dans le domaine des

bases

de données, une opération sur les données est

appelée une transaction ou transaction informatique.

Q5: Définir une base de données relationnelle

Idem reponse 3

Q6 : Définir le NoSQL

Le NoSQL (pour «

not only sql

») est une catégorie de SGBD de SBGD

s'affranchissant du modèle relationnel des SGBDR. L’unité de stockage n’est plus la

table, et les données ne sont plus manipulées avec le SQL. Les schémas

relationnels ne sont plus nécessaires et le NoSQL renonce aux propriétés ACID des

SGBDR.

Q7 : Définir les différentes familles du NoSQL et Q8 :Pour chaque famille du NoSQL donner un exemple de

base sur le marché

:

Il y a 4 familles

:

-

Orienté colonne

:

Le stockage orienté colonne se focalise sur chaque attribut

et le distribue, il est alors possible de focaliser les requêtes sur une ou

plusieurs colonnes sans avoir à traiter les informations inutiles (les autres

colonnes). Cela permet les gros calculs analytiques. L’orienté colonne est par

contre beaucoup moins approprié pour la lecture de données spécifiques

comme pour les clés/valeurs. Cela fonctionne avec un id par colonne. (ex

:

Col1

: id

«

Nicolas

» type

:

«

prof

», col2. Id «

Nicolas

» lieu «CNAM

».Cela

peut servir surtout pour les comptages (vote en ligne par ex), journalisation,

recherche de produits dans une catégorie... Un exemple sur le marché

:

Cassandra.

-

Orienté Document

:

Les bases orientées documents servent à manipuler des

documents contenant des informations avec une structure complexe

(json,xml...). Il repose sur le principe clé/valeur mais la valeur peut être un

document qui contient à son tour des clés et des valeurs. Le document a une

structure arborescente. Chaque attribut du document/sous-document est

manipulable et cela permet de nombreuses interactions. Ce sont parmi les

bases noSQL les plus répandues, notamment pour ce qui concerne les

données client, la gestion catalogue des produits, le web analytics... Un

exemple sur le marché

:

MongoDB.

-

Orienté graph

:

Les bases orientées Graphes ont pour objectif de stocker les

données en se basant sur la théorie des graphes. Cela s’appuie sur les

notions de nœuds

: ils ont chacun leur propre structure

– mais aussi sur les

notions de relation entre les nœuds et leurs propriétés (nœuds ou relations).

Cela facilite la représentation du monde réel, pratique pour traiter par exemple

les données des réseaux sociaux et géographiques, ou de toutes les données

fortement connectées de manière générale. Ces bases sont beaucoup plus

rapides que les autres systèmes de stockage pour manipuler les données très

connectées. Un exemple sur le marché

: Neo4J.

-

Orienté clé/valeur

:

Cela fonctionne un peu comme une base de données relationnelle mais sans

schéma ni structure pour le stockage de données. Tout repose sur le couple

clé/valeur

: la clé identifie la colonne de manière unique et permet de la gérer,

la valeur peut contenir n’importe quel type de donnée. Le but est l’efficacité et

la simplicité. Les seules opérations de CRUD peuvent être utilisées. Ces

bases sont surtout utilisées pour des données qu’on ne souhaite pas

conserver longtemps. (Exemple

: gestion de panier d’achat, cache pour

conserver les sessions d’un site web...) . En revanche cela ne permet pas

d’effectuer de requêtes sur le contenu des objets stockés et on ne conserve

pas les relations entre les objets. Elles ne sont pas faite pour les

modélisations métier complexes. Un exemple clé valeur sur le marché

:

Redis.

Q9 : la famille orienté graphe se base sur les notions de

clé/valeur : vrai ou faux ?

Faux, elle se base sur des nœuds et leurs relations (arcs). Mais les propriétés

«

clé/valeur

» peuvent enrichir les nœuds et les relations afin d’y ajouter du contexte.

Par exemple (comme nous ne l’avons pas vu en cours je me suis permise d’aller

chercher sur wikipedia)

:

En

base de données orientée graphe

, la table client sera représentée par un ensemble de

nœuds pour chaque instance, donc le même client sera représenté par un nœud suivant

:

"Client

:

identifiant

: id01

nom

: Dupont

prénom

: Jean

adresse

: 96 rue des lilas

numéro client

: 010203".

L'entreprise sera représentée de la même manière par un nœud

:

"Entreprise

:

siret

: 1221

nom

: Entreprise D

domaine

: automobile"

Et la relation entre les deux sera matérialisée par un arc partant du nœud Entreprise EntrepriseD

vers le nœud Dupont Jean, nommée "a\_pour\_client".

Q10 : Dans une base orienté Graphe ou se trouve les

propriétés ?

Dans les labels

Q11: Définir le théorème BASE

Les propriétés BASE ont été proposées pour caractériser les bases NoSQL.

Basically Available

: Quelque soit la charge de la base de données

(données/requêtes), le système garantit un taux de disponibilité de la donnée.

Soft-state

: la base peut changer pendant la mise à jour ou quand on

ajoute/supprime un serveur. La base NoSQL n’a pas à être cohérente à tout instant.

Q12 : Définir le théorème de CAP

Le théorème de CAP (Consistency, Availability, Partition tolerance

, dit

: Dans toute

Base de données vous ne pouvez respecter au plus que deux propriétés parmi la

cohérence, la disponibilité et la distribution. Les bases de données relationnelles par

exemple gèrent la cohérence et la disponiblité, mais pas la distribution.

Q13 : Positionner les Base de données relationnelles sur le

triangle de CAP, ainsi que mongoDB

:

Q14: Définir le Sharding et l'élasticité

L’elasticité est la capacité du système à s’adapter automatiquement en fonction du

nombre de serveurs dont il dispose et la quantité de données à répartir. Par exemple

lors d’un pic d’activité, si l’on rajoute des serveurs à des serveurs déjà existants,

l’élasticité va permettre de répartir uniformément les données sur les 2000 serveurs

et inversement lorsque le pic d’activité s’achève.

Le sharding est une technique qui permet de distribuer des morceaux de fichier

(chunk) sur un ensemble de serveurs avec la capacité de gérer l’élasticité

(serveurs/données) et la tolérance aux pannes.

Q15 : Donner les différentes techniques pour faire le

sharding

Clustered Index(basé sur la distribution), Consistent Hashing(basé sur les tables de

Hashage), HDFS

(basé sur la distribution)

Q16 : Donner les ports par défaut de MongoDB, Redis et

Cassandra

:

MongoDB

: 27017

Redis

: 6379

Cassandra

: 7199

Q17 : mongos sert à créer une instance de mongo

Faux

Q18 : A quoi sert mongodump

A faire un back-up de la base de données.

Q19 : Quelle est la commande à exécuter pour tester la

bonne installation de cassandra ?

cassandra.bat

Q20: Quelle est la structure d'une base de données

mongo ?

Une base possède des collections qui possèdent des documents.

Q21:Quelle est la structure d'une base de données Redis ?

Lists, sets ou Hashs.

Q22 : Quelle est la structure d'une base de données

Cassandra?

Les tables sont des colonnes qui ont 1 id/1type.

Q23:Quelle est la structure d'une base de données neo4j ?

Nœuds/Arcs (avec des labels qui peuvent contenir des propriétés pour préciser les

relations)

Q24:Quelles sont les classes permettant d'établir une

connexion d'une application Java vers une base de donnée

mongoDB ?

MongoClient

Q25 :Quelles sont les classes permettant d'établir une

connexion

d'une application Java vers une base de donnée Redis ?

Jedis

Q26 : Quelle est la commande mongo permettant d'afficher

tous les documents d'une collection

db.NomCollection.find()

;

Q27 : A quoi servent l'objet cursor et sa méthode

hasNext() ?

Lors d’une requête mongodb «

getAll

» pour trouver des documents, le curseur se

place sur le premier document trouvé, c’est lui qui va récupérer le résultat de la

requête, pour qu’on puisse en extraire les valeurs ensuite. Le hasnext interroge

cursor+1 pour voir si le curseur est arrivé au dernier document ou non.

Q28 : A quoi servent les fonction $match, $project et

$unwind

{ $match:{} } : Correspond au premeir paramètre de la requête find. Il permet de filtrer

le

contenu

d'une

collection

{ $project:{} } : Correspond au second paramètre de la requête find. Il donne le

format de sortie des documents (projection).

{ $unwind:{} } : Il prend une liste de valeur et produit pour chaque élément de la liste

un nouveau document en sortie avec cet élément.

Q29 : Quelles sont les types de données d'une base de

donnée Redis?

Strings, Lists, Type Set, Type Hash, Type Set trié.

Q30 : Quelle est la classe permettant de se connecter à une

base Redis avec une interface Java

La classe Jedis