
SINF803 • Bases de données réparties (Gr 1)

Enseignant: Edmond La Chance
Bureau: P4-6570
Courriel: edmond.lachance@gmail.com
Page web du cours: <https://github.com/edmondlachance/SINF803>

Contenu général

Certains sites et applications ont des millions d'utilisateurs. Ces derniers génèrent de l'information tous les jours. Toute cette information, ce data, doit être enregistré et peut être traité et/ou indexé afin de fournir un service aux utilisateurs. Ce domaine est également appelé le domaine du Big Data. Exemples de sites et applications Big Data : Google (engin de recherche), Netflix (suggestions), Instagram (images, commentaires, votes), Facebook (photos, posts, likes, suggestions d'amis) et tant d'autres.

Le premier problème d'une base de données répartie est le stockage des données. Il faut d'abord pouvoir répartir les données sur plusieurs machines. Nous allons voir deux approches : l'approche de la base de données relationnelle et l'approche d'un système de fichiers distribué comme le Hadoop File System. HDFS enregistre le nouveau data comme un log, ce qui facilite énormément le scaling horizontal et permet également d'utiliser les disques durs conventionnels plus efficacement.

Le deuxième défi est de pouvoir faire des requêtes sur les données. Dans le cas d'un système relationnel, on utilise le langage SQL. Dans le cas d'un système comme celui proposé dans la Lambda Architecture, on utilise plutôt MapReduce, un pipeline d'agrégation (MongoDB) ou une autre technologie pouvant être exécutée sur le cluster comme Apache Spark. Apache Spark permet d'exécuter MapReduce en mémoire vive, ce qui accélère énormément le temps de calcul comparé au MapReduce de Hadoop.

Il faut également apprendre à traiter des données realtime car la faiblesse des technologies de batch computing (Hadoop MapReduce, Spark) est le temps de réponse. Parfois il faut plusieurs heures pour construire des vues à partir du log des données. Certains site webs ou applications ne peuvent pas se permettre d'afficher leurs nouvelles données quelques heures en retard. La solution de la Lambda Architecture est donc d'avoir une couche d'application qui ne s'occupe que des données recentes. Cette couche utilise des technologies et base de données différentes. On parle ici de bases de données incrémentales comme MySQL, Cassandra, de système de Stream Processing comme Apache Storm et Apache Spark et d'une file distribuée comme Apache Kafka.

Le cours a comme objectif de peindre un portrait du monde du Big Data. En premier, on va regarder comment fonctionner avec un système relationnel lorsqu'on est confronté au Big Data. Ensuite, on va regarder la Lambda Architecture, une architecture à trois couches qui permet d'aller répondre à tous les critères d'un système Big Data, tout en étant moins compliqué qu'une architecture incrémentale. À l'intérieur de la Lambda Architecture, nous allons voir les technologies de bases de données ainsi que les technologies et algorithmes (MapReduce, Spark) qui permettent d'extraire ce qu'on veut des données.

Deux langages de programmation seront utilisés : Javascript pour MapReduce et Scala pour Spark.

Objectifs du cours

Au terme de ce cours, l'étudiant aura acquis les compétences suivantes:

- Retour sur la BD relationnelle centralisée.
- Normalisation (1FN, 2FN, 3FN)
- Définir les concepts importants relatifs aux bases de données réparties
- Algèbre relationnel
- Fragmentation horizontale et verticale
- Lambda Architecture : Batch Layer, Serving Layer et Speed Layer
- Programmation avec MapReduce
- Programmation avec Apache Spark
- Programmation avec Apache Spark GraphX
- Utilisation de MongoDB
- Connaissance du langage Scala et du langage Javascript

Sujets abordés

Le cours se divise en deux grandes parties.

1. La première moitié du cours est un retour sur la BD relationnelle, l'algèbre relationnel et la normalisation.
2. La deuxième moitié du cours porte sur la Lambda Architecture, MapReduce, Spark, l'approche incrémentale vs l'approche de précomputation

Évaluation

L'évaluation consistera en les éléments suivants:

- Deux travaux à faire en équipes de deux à quatre personnes, à remettre les **7 mars** et **18 avril**. Les travaux seront composés de questions à développement, ainsi que d'exercices de programmation faisant appel aux technologies et aux concepts présentés en classe. Tout travail remis en retard sans motif valable sera pénalisé de 10% par jour de retard.
- Un examen intra à la séance du **14 mars**. Cet examen compte pour 25 points.
- Un examen final à la séance du **11 avril**. Cet examen compte pour 25 points.

La note finale, sur 100, est la somme de toutes ces évaluations. La note de passage est fixée à **60%** ou D.

Date limite d'abandon sans mention d'échec

20% de l'évaluation aura été transmise à l'étudiant avant la date limite d'abandon sans mention d'échec, soit le Jeudi 16 mars 2017.

Qualité du français écrit

Tout travail remis doit être conforme aux exigences de la politique institutionnelle en matière de maîtrise du français écrit. Comme il en est fait mention dans le Manuel de gestion (3.1.1-012).¹ Tout travail dont la qualité du français serait jugée non conforme par l'enseignant pourra être pénalisé jusqu'à concurrence de 10% du résultat maximal prévu.

Évaluation de la qualité de l'enseignement

Ce cours sera évalué en fonction de la Procédure relative à l'évaluation des activités aux programmes d'études de cycles supérieurs (Manuel de gestion, 3.1.2-008).

Formule pédagogique

Le cours sera dispensé en classe par un professeur. Il n'y aura pas de séance de travaux pratiques associées à ce cours.

Situation du cours dans le programme

Le cours est optionnel pour les étudiants inscrits aux programmes de cycles supérieurs. Aucun cours ne lui est préalable.

Références

Aucun livre ou recueil de notes n'est obligatoire pour ce cours. Les diapositives utilisées durant les séances et les lectures préalables seront rendues disponibles en format électronique sur le site du cours.

Ces livres ont été utilisés pour construire le cours :

- Nathan Marz (2015). Big Data: Principles and best practices of scalable realtime data systems
- Michael S. Malak and Robin East (2016). Spark GraphX in Action
- Petar Zečević and Marko Bonaći (2016). Spark in Action

¹http://www.uqac.ca/direction_services/secretariat_general/manuel/index.pdf