MOOCEBFR4 : Ingénierie des Données du Big Data : SGBD NoSql et Lacs de Données avec Big Data SQL par la pratique

Gabriel MOPOLO-MOKE
Docteur en Informatique
Professeur chargé d'enseignements
Université Côte d'Azur (UCA)

2023 / 2024

Plan Général

- **Plan**
  - *Module M4.1 : Rappel sur les concepts du Big Data et des SGBD NOSQL*
  - *Module M4.2 : Introduction à Oracle NOSQL*
  - *Module M4.3 : Oracle NoSql et le Modèle Key/Document*
  - *Module M4.4 : INTRODUCTION A MONGODB ET LE MONGO SHELL*
  - *Module M4.5 : INTRODUCTION A MONGODB ET SON API JAVA*
  - **Module M4.6** : **Architectures Big Data et construction de lacs de Données avec Big Data SQL par la pratique**

Module M4.6 : Architectures Big Data et construction de lacs de Données avec Big Data SQL par la pratique

Gabriel MOPOLO MOKE
Docteur en Informatique
Professeur chargé d'enseignements
Université Côte d'Azur (UCA)

2022 / 2023

▪Module M4.6 : Architectures Big Data et construction de lacs de Données avec Big Data SQL par la pratique

ESTIA INSTITUTE OF TECHNOLOGY
UNIVERSITÉ CÔTE D'AZUR

➢Plan

UNIVERSITÉ CÔTE D'AZUR

ESTIA INSTITUTE OF TECHNOLOGY

DATUM ACADEMY

Module M4.6, section 1 : Du Dataware House au Lac de données

FUN FRANCE UNIVERSITÉ NUMÉRIQUE

anr

INVESTIR L'AVENIR

MIAGE RÉSEAU DES MIAGE DE FRANCE Nice

MBDS

BIHAR

Module M4.6, section 1 : Du Dataware House au Lac de données

ESTIA INSTITUTE OF TECHNOLOGY

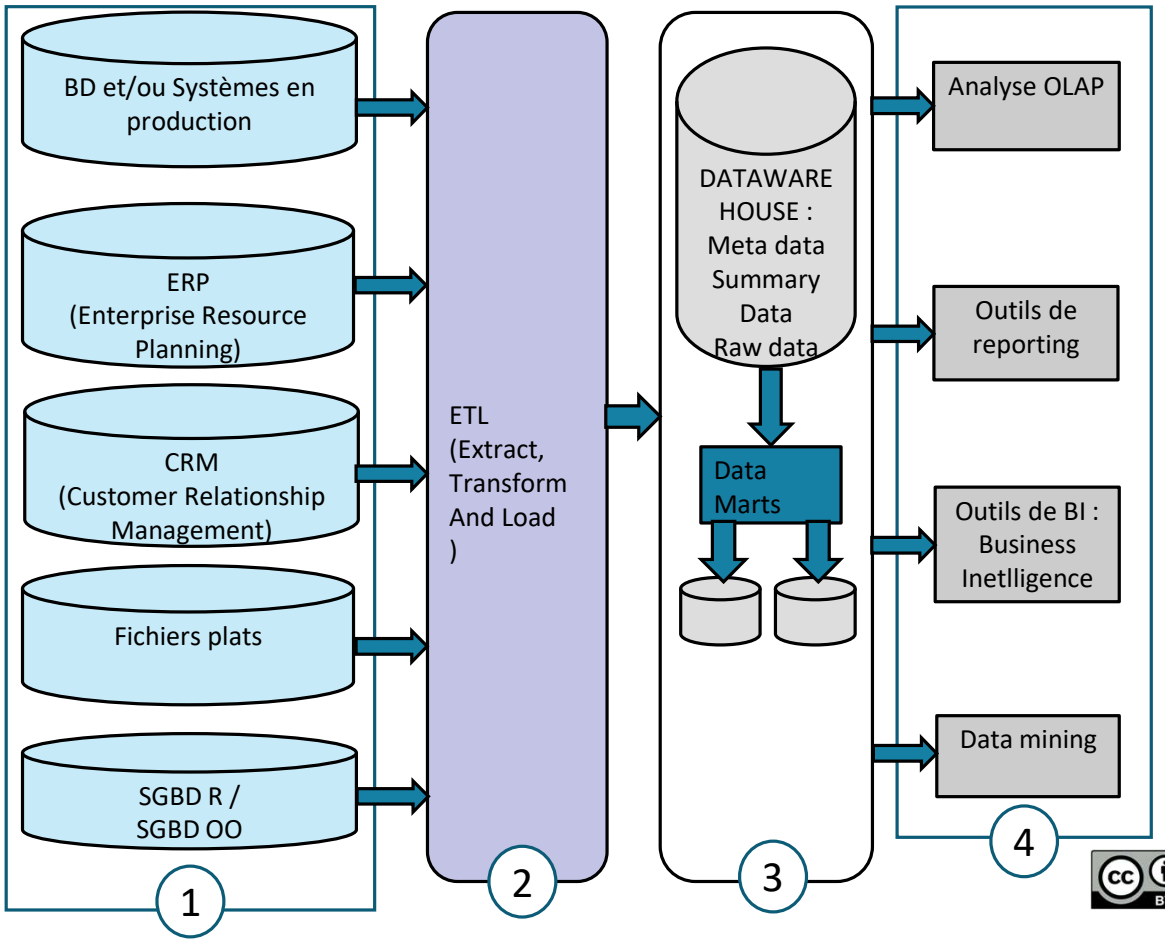
UNIVERSITÉ CÔTE D'AZUR

Plan

- Une Architecture Data Warehouse
- Data Warehouse : Définition de Inmon [4]
- Exemple de schéma de données d'un Data Warhouse (schéma en étoile)
- Exemple de sources de données pour un DWH
- Les ETL (Extract Transform and Load)
- Comparatif entre les bases de données de l'entreprise
- Données opérationnelles versus Données décisionnelles
- Une Architecture Lac de Données
- Définition d'un Lac de Données
- Les ELT (Extract, Load and Transform)
- Exemple de sources de données pour un lac de données
- Comparaison Lac de Données et Data warehouse

Module M4.6, section 1 : Du Dataware House au Lac de données

- **Une Architecture Data Warehouse**
  - **Sources** 1
  - **ETL** 2
  - **Data warehouse** 3
  - **Outils d'Analyses et de reporting** 4



Module M4.6, section 1 : Du Dataware House au Lac de données

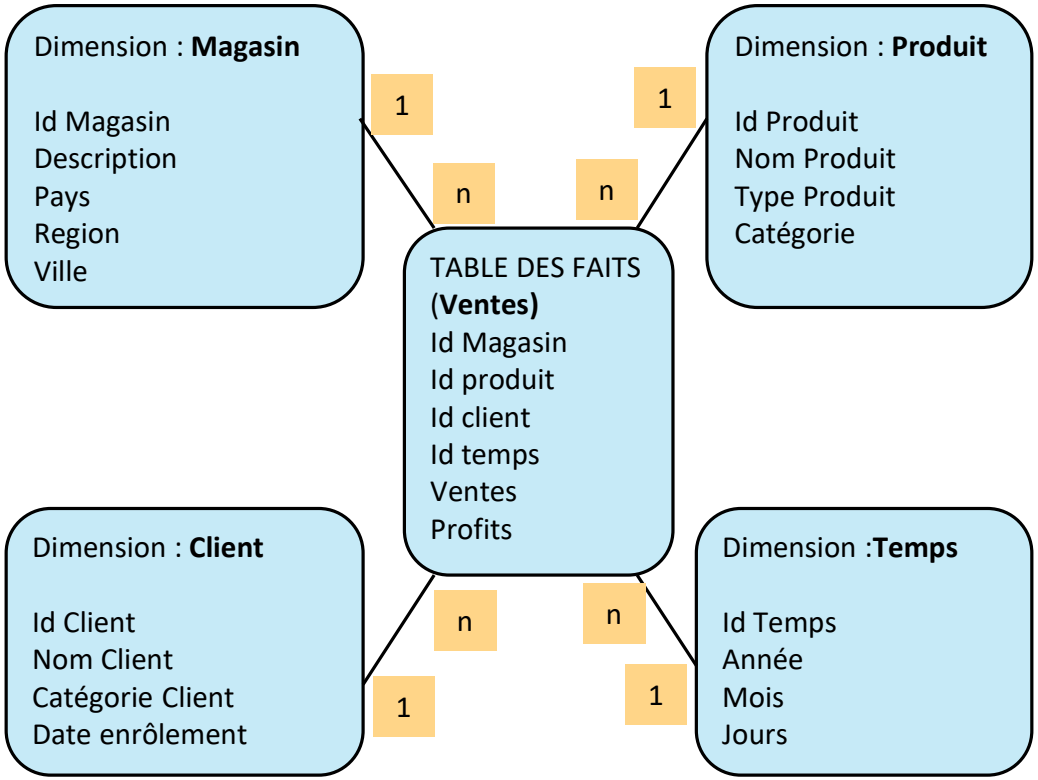
➢Data Warehouse : Définition de Inmon [4]

- **Le Data Warehouse est une collection de données :**

  - Orientées sujet
  - Intégrées
  - Non volatiles (sans mise à jour en continue)
  - Evolutives dans le temps
  - Organisées pour le support d'un processus d'aide à la décision.

Module M4.6, section 1 : Du Dataware House au Lac de données

- **Exemple de schéma de données d'un Data Warehouse (schéma en étoile)**



Module M4.6, section 1 : Du Dataware House au Lac de données

➢Exemple de sources de données pour un DWH

- **Données issues de l'ERP de l'entreprise**
  - Magasin (Id Magasin, Description, id Pays, id Region, id Ville)
  - Pays (id Pays, Nom Pays)
  - Region (id region, id pays, nom region)
  - Ville (id ville, id pays, id regin, nom ville)
  - Ventes (Id Magasin, Id produit, Id client, date vente, Prix unitaire, Quantite)
  - Client (Id Client, Nom Client, Catégorie Client, Date enrôlement)
  - Produit (Id Produit, Nom Produit, Type Produit, Catégorie)

- **Données de la base des ventes en lignes**
  - ProduitVEL(Id Produit, Nom Produit, Type Produit, Catégorie)
  - Localite Acheteur (Id Magasin, Description, id Pays, id Region, id Ville)
  - Ventes (Id Magasin, Id produit, Id client, date vente, Prix unitaire, Quantite)

Module M4.6, section 1 : Du Dataware House au Lac de données

➢Les ETL (Extract Transform and Load)

▪ Principales fonctionnalités

- **F1**: Ordonnancement des tâches
- **F2**: Gestion centralisée des métadonnées
- **F3**: Gestion des glossaires métier
- **F4**: Administration des scénarios d'alimentation
- **F5**: Administration de la qualité des données (Datastage)...
- **F6**: Accès aux données non structurées

Module M4.6, section 1 : Du Dataware House au Lac de données

➢Les ETL (Extract Transform and Load)

▪ Mode d'extraction

- **E1** : **Extraction batch** et/ou via MOM (Message Oriented Middleware)
- **E2** : **Extraction temps réel** et/ou via MOM

▪ Type d'architecture

- **A1** : Architecture : Hub (Moyeu) &Spoke (Rayon) : Réseau en étoile

Module M4.6, section 1 : Du Dataware House au Lac de données

➢ Les ETL (Extract Transform and Load)

- **Exemple de Règles de transformation dans un ETL**

Les Sources de Données	Exemple de Règle ETL	Base data Warehouse
Différents formats des dates selon les sources Exemple: DD/MM/YY, DD/MM/YYYY, YYYY/MM/DD)	Harmonisation du format des dates DD/MM/YY, DD/MM/YYYY, YYYY/MM/DD =>DD/MM/YYYY	un seul format de date (exemple : DD/MM/YYYY)
Différentes valeurs signifiant la même chose selon les sources (Exemple: Homme, H, H., etc.)	Harmonisation des valeurs signifiant la même chose (Homme, H., H=> Homme)	Une seule valeur dans le DWH (Homme)
Données en détail (ventes par ticket de casse par produit)	Règle de groupement (cumul journalier sur un produit pour tous les tickets de caisse)	Montant ventes journalières par produit
La même information dans plusieurs sources (exemple: client dans source 1, client dans source 2, ...)	regroupement de divers sources vers une seule destination (client:source1, client source2 : => client	Une seule destination (exemple 1 seule table client)
...	...	...

Module M4.6, section 1 : Du Dataware House au Lac de données

➢ Les ETL (Extract Transform and Load)

▪ Tableau comparatif

Solution	Editeur	Principales briques fonctionnelles	Extraction	type d'architecture	Connecteurs ERP natifs	Tarification	Principaux clients
Information service	IBM	F1, F2, F3, F4, F5	E1, E2	A1	SAP, Oracle, Siebel, PeopleSoft	>=88000	N/A
Power center	Informatica	F1, F2, F3, F4, F5, F6	E1, E2	A1	SAP, Oracle, Siebel, PeopleSoft	>50000	EDF, SG, Danone
ODI : Oracle Data Integrator	Oracle	F1, F2, F5	N/A	N/A	Oracle, Siebel, PeopleSoft	>25000	Arpège, Caisse
Sql server Integration Services	Microsoft	F1, F2, F5	E1, E2	A1	N/A	>24000$	N/A
Open Studio	Talent	F1, F2, F3, F4, F5	E1, E2	A1	SAP, OpenBravo, ...	N/A	Accor, GMF, ...

Module M4.6, section 1 : Du Dataware House au Lac de données

➢Comparatif entre les bases de données de l'entreprise

[https://fr.wikipedia.org/wiki/Entrep%C3%B4t_de_donn%C3%A9es}

Caractéristique	Base de données de production	Data warehouses	Datamarts
Opération	gestion courante, production	référentiel, analyse ponctuelle	analyse récurrente, outil de pilotage, support à la décision
Modèle de données	entité/relation	3NF, étoile, flocon	étoile, flocon
Normalisation	fréquente	maximum	rare (redondance d'information)
Données	actuelles, brutes, détaillées	historisées, détaillées	historisées, agrégées
Mise à jour	immédiate, temps réel	souvent différée, périodique	souvent différée, périodique
Niveau de consolidation	faible	faible	élevé
Perception	verticale	transverse	horizontale
Opérations	lectures, insertions, mises à jour, suppressions	lectures, insertions, mises à jour	lectures, insertions, mises à jour, suppressions
Taille	en gigaoctets	en téraoctets	en gigaoctets

Module M4.6, section 1 : Du Dataware House au Lac de données

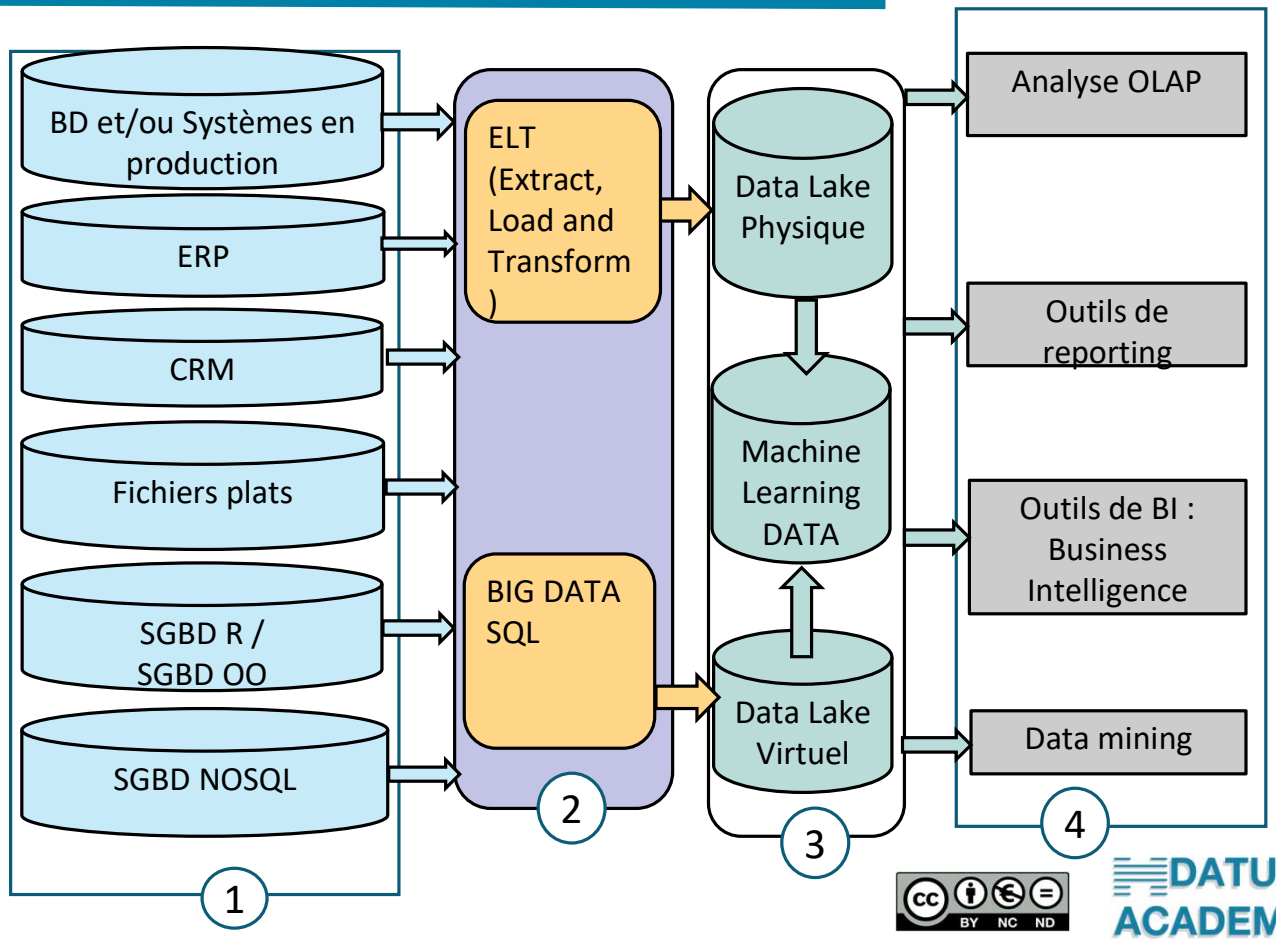
➢Données opérationnelles versus Données décisionnelles

Données opérationnelles	Données décisionnelles
Orientées application, détaillées, précises au moment de l'accès	Orientées activité (thème, sujet), condensées, représentent des données historiques
Mise à jour interactive possible	Pas de mise à jour interactive
Accès par une personne à la fois	Utilisées par l'ensemble des analystes
Haute disponibilité en continu	Haute disponibilité ponctuelle
Uniques(pas de redondance en théorie)	Peuvent être redondantes
Petite quantité de données utilisées par un traitement	Grande quantité de données utilisée par les traitements
Réalisation des opérations au jour le jour	Cycle de vie différent
Forte probabilité d'accès	Faible probabilité d'accès
Utilisées de façon répétitive	Utilisées de façon aléatoire

Module M4.6, section 1 : Du Dataware House au Lac de données

➢ Une Architecture Lac de Données

- **Sources** (1)
- **ELT / Big Data SQL** (2)
- **Data lake** (3)
- **Outils d'Analyses et de reporting** (4)



Module M4.6, section 1 : Du Dataware House au Lac de données

➢Définition d'un Lac de Données

- Un Data Lake est une **base de données** qui permet de stocker de larges volumes de données **structurées, semi-structurées et non-structurées** (SOURCES HETEROGENES).
- Il est un immense contenant prêt à accueillir de gros volumes de données « en vrac **: NON TRAITEES, NON ORGANISEES, PAS FORCEMENT UTILES AU MOMENT DU CHARGEMENT**»
- Le Data Lake à l'inverse du Data Warehouse évoque **le liquide, le fluide, le mélange, l'inorganique, le non-structuré**.
- Un lac de Données peut être :
  - **Physique**
  - **Virtuel**
  - **Mixte (Physique et Virtuel)**
- Un data lake peut être organisé dans un ou plusieurs systèmes de stockage

Module M4.6, section 1 : Du Dataware House au Lac de données

➢Définition d'un Lac de Données

▪ L'Alimentation du lac de données se fait avec :

- Les **ELT (Extract, Load and Transform)** : Outils d'Extraction, chargement et transformation de Données
- **Manuellement par programmation** ou utilisant des utilitaires spécifiques
- **Via des langages de type Big Data SQL** avec des tables externes
  - ✓ HiveQL
  - ✓ Oracle Big Data SQL
  - ✓ BigQuey de Google
- Des utilitaires du Big Data dans Hadoop tels que **FLUME et SQOOP**

Module M4.6, section 1 : Du Dataware House au Lac de données

➢Les ELT (Extract, Load and Transform)

- **Contrairement à l'ETL, l'ELT (Extract/Load/ Transform) permet de :**
  - **Rassembler** les informations provenant d'un nombre **illimité de sources**,
  - **Charger** dans un emplacement en vue de leur traitement,
  - Puis à les **transformer** en données décisionnelles actionnables.

Module M4.6, section 1 : Du Dataware House au Lac de données

➢Les ELT (Extract, Load and Transform)

- **Extraction – La première étape**
  - Ingestion des Flux bruts de données d'une infrastructure virtuelle, logiciels et applications en fonction de règles prédéfinies
- **Chargement - deuxième étape**
  - L'ELT charge toutes les données brutes dans le Data Lake (Une Base de Données, Un file System tel que HDFS, un mixte)
- **Transformation - troisième étape**
  - L'ELT tri et normalise les données à des fins de reporting ou d'analyse

Module M4.6, section 1 : Du Dataware House au Lac de données

➢ Exemple de sources de données

- **Données issues de l'ERP de l'entreprise**
  - Magasin (Id Magasin, Description, id Pays, id Region, id Ville)
  - Pays (id Pays, Nom Pays)
  - Region (id region, id pays, nom region)
  - Ville (id ville, id pays, id regin, nom ville)
  - Ventes (Id Magasin, Id produit, Id client, date vente, Prix unitaire, Quantite)
  - Client (Id Client, Nom Client, Catégorie Client, Date enrôlement)
  - Produit (Id Produit, Nom Produit, Type Produit, Catégorie)

- **Données de la base des ventes en lignes**
  - ProduitVEL(Id Produit, Nom Produit, Type Produit, Catégorie)
  - Localite Acheteur (Id Magasin, Description, id Pays, id Region, id Ville)
  - VentesVEL (Id Magasin, Id produit, Id client, date vente, Prix unitaire, Quantite)

Module M4.6, section 1 : Du Dataware House au Lac de données

➢Exemple de Lac de Données

- **Données brutes issues de chaque source**
  - Magasin (Id Magasin, Description, id Pays, id Region, id Ville)
  - Pays (id Pays, Nom Pays)
  - Region (id region, id pays, nom region)
  - Ville (id ville, id pays, id regin, nom ville)
  - Ventes (Id Magasin, Id produit, Id client, date vente, Prix unitaire, Quantite)
  - Client (Id Client, Nom Client, Catégorie Client, Date enrôlement)
  - Produit (Id Produit, Nom Produit, Type Produit, Catégorie)
  - ProduitVEL(Id Produit, Nom Produit, Type Produit, Catégorie)
  - Localite Acheteur (Id Magasin, Description, id Pays, id Region, id Ville)
  - VentesVEL (Id Magasin, Id produit, Id client, date vente, Prix unitaire, Quantite)

Module M4.6, section 1 : Du Dataware House au Lac de données

➢ **Comparaison Lac de Données et Data warehouse**

Paramètres	Data Lake	Data Warehouse
Stockage	Toutes les données sont stockées dans leur état brut. Elles ne sont transformées qu'en aval et suivant les	Un Data Warehouse rassemble des données nettoyées et transformées.
Histoire	Les technologies Big Data avec lesquelles les Data Lakes sont construits sont relativement récentes.	Le concept de Data Warehouse est ancien.
Collecte des Données	Le Data Lake capture toutes les sortes de données (structurées, semi-structurées, non-structurées)	Le Data Warehouse extrait des informations structurées et les organise dans des schémas de données designés pour répondre à des finalités précises.
Horizon temporel	Le Data Lake est utilisé pour stocke toutes les données de l'entreprise. Les données utiles maintenant ou avenir, on ne « jette » rien.	Dans le process de développement d'un Data Warehouse, on extrait uniquement les données correspondantes au modèle du Data Warehouse
Utilisateurs	Un Data Lake est idéal pour les Data Scientists ou les personnes souhaitant faire de la modélisation prédictive ou de l'analyse statistique avancée.	Bien structuré et facile à utiliser, le Data Warehouse est idéal pour les utilisateurs opérationnels. Obtenir des reports et indicateurs.
Coûts de stockage	Les données sont stockées dans des technologies Big Data (type Hadoop) qui sont relativement peu coûteuses.	Dans une architecture Data Warehouse, le coût de stockage des données est relativement élevé. La gestion est également plus chronophage, donc plus coûteuse.
Temps de processing	Permet l'accès aux données avant que celles-ci ne soient transformées, nettoyées et structurées. Par conséquent, le temps de processing pour obtenir des résultat est plus	Le Data Warehouse délivre des réponses à des questions prédéfinies sur des données prédéfinies. Tout changement dans le Data Warehouse est chronophage et est complexe.
Modélisation	Le schéma de données est défini en aval, après que les données aient été stockées. Agile mais lourd	Le schéma de données est défini en amont, avant l'entrée des données dans le Data Warehouse. Avantages : performance, sécurité, intégration.
Le Data Processing	Processus ELT : Extract, Load & Transform. On charge la donnée dans la base avant de la transformer.	Processus ETL : Extract, Transform & Load. On applique des transformations aux donnéespour s'adapter au schéma
Inconvenient	Les données sont stockées dans leur état brut et en vrac.	Un DataWarehouse est très difficile à faire évoluer;
Avantage fort	Permet de découvrir des réponses à des questions que l'on ne se posait pas. Adapté pour les cas d'usage avancés d'analyse (Machine learning, IA)	Permet de faire du reporting et de calculer des métriques clés.

Module M4.6, section 1 : QUIZ

- **Question 1** : Cochez ce qui caractérise le mieux un dataware house
  - A: Un schéma de données connu à l'avance
  - B: Un schéma de données relationnel SQL2 ou SQL3
  - C: Un schéma de données au format JSON
  - D: Un ensemble de fichiers dans Hadoop HDFS

- **Question 2** : Cochez ce qui caractérise le mieux un dataware house
  - A: Il permet de gérer les données des applications en production
  - B: Il permet de gérer les données historisées
  - C: Les données du DWH sont mise à jour fréquemment
  - D: Les données du DWH sont mises à jour de temps en temps

Module M4.6, section 1 : QUIZ

➢ **Question 3** : Cochez les définitions qui caractérisent le mieux un ETL (Extract, Transform, Load)

- A: C'est un outil qui permet de recopier les données des sources de données vers le DWH
- B: La recopie peut se faire en temps réel
- C: Si recopie il y a des transformations et des calculs peuvent être réalisés
- D: L'ETL peut procéder à des mises à jour des jours de **données directement dans le DWH**

➢ **Question 4** : Cochez ce qui correspond aux caractéristiques d'un ETL

- A: Le coût d'un ETL est négligeable
- B: Un ETL possède des fonctions d'ordonnancement des tâches
- C: Un ETL possède des fonctions d'administration des scénarios d'alimentation
- D: Un ETL ne se matérialise pas par des outils, c'est uniquement une philosophie

Module M4.6, section 1 : QUIZ

➢ **Question 5** : Cochez les données qu'on peut trouver dans les Data lake

- A: Des données relationnelles
- B: Des fichiers
- C: Des données semi-structurées
- D: Les données d'un ETL ou d'un ELT

➢ **Question 6** : Afin de copier physiquement les données dans un Data lake nous devons

- A: Utiliser un ETL
- B: Utiliser un ELT
- C: Prétraiter obligatoirement les données avant leur chargement
- D: Non, non le traitement se fait en principe après chargement

Module M4.6, section 1 : QUIZ

➢ **Question 7** : Cochez ce qu'est une base de données data lake

- A: Elle peut être virtuelle
- B: Elle peut être physique
- C: Elle peut être virtuellement et physique
- D: L'affirmation en B: est fausse

➢ **Question 8** : Cochez ce qu'est une base de données data lake

- A: C'est forcement une et une seule base de données relationnelles
- B: cela peut être des SGF, des BD NOSQL, des BD relationnelles
- C: Elle peut être accessible à travers des langages de type Big Data SQL
- D: C'est forcement une et une seule base de données NOSQL

UNIVERSITÉ CÔTE D'AZUR ESTIA INSTITUTE OF TECHNOLOGY

DATUM ACADEMY

Module M4.6, section 2, partie 1 : Architectures Big Data, Eco-système HADOOP

FUN FRANCE UNIVERSITÉ NUMÉRIQUE

anr

INVESTIR L'AVENIR

MIAGE RÉSEAU DES MIAGE DE FRANCE Nice

MBDS

BIHAR

Module M4.6, section 2, partie 1 : Architectures Big Data, Eco-système HADOOP

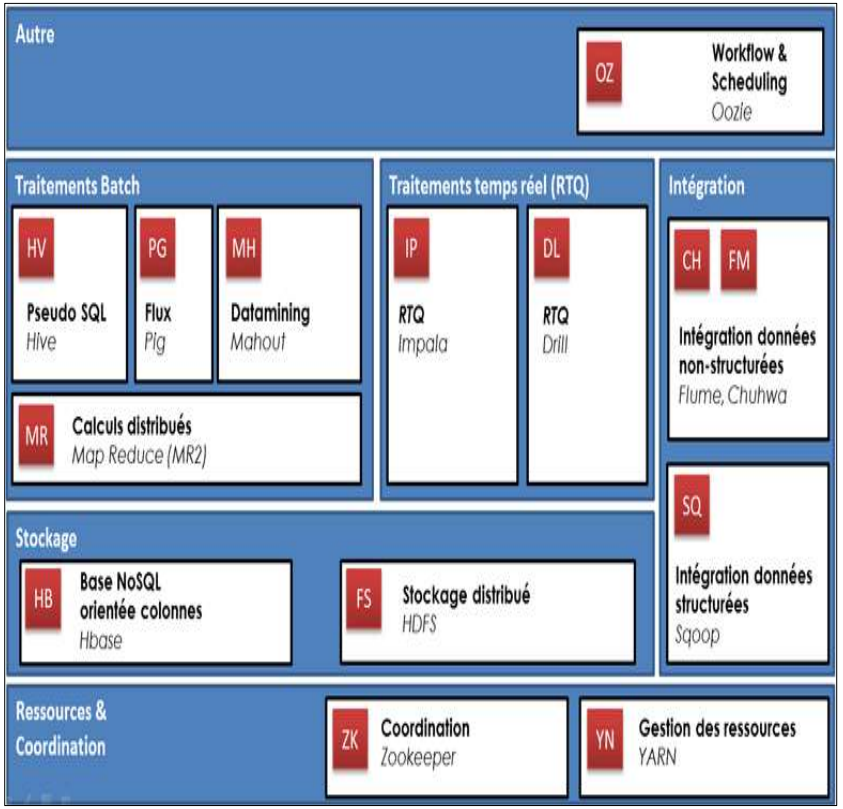
Plan

- Architecture des composants HADOOP
- Services de stockage dans Hadoop
- Algorithme Map Reduce
- Gestion et administration des composants dans Hadoop
- Couches d'abstraction de Map Reduce dans Hadoop
- Gestion des Streams
- Divers Utilitaires
- Modèle de calcul interactif ou Batch
- Zoom sur l'utilitaire SQOOP
- Zoom sur Apache KAFKA
- Zoom sur Apache FLUME
- Zoom sur Apache SPARK

Module M4.6, section 2, partie 1 : Architectures Big Data, Eco-système HADOOP

➢ Architecture des composants HADOOP

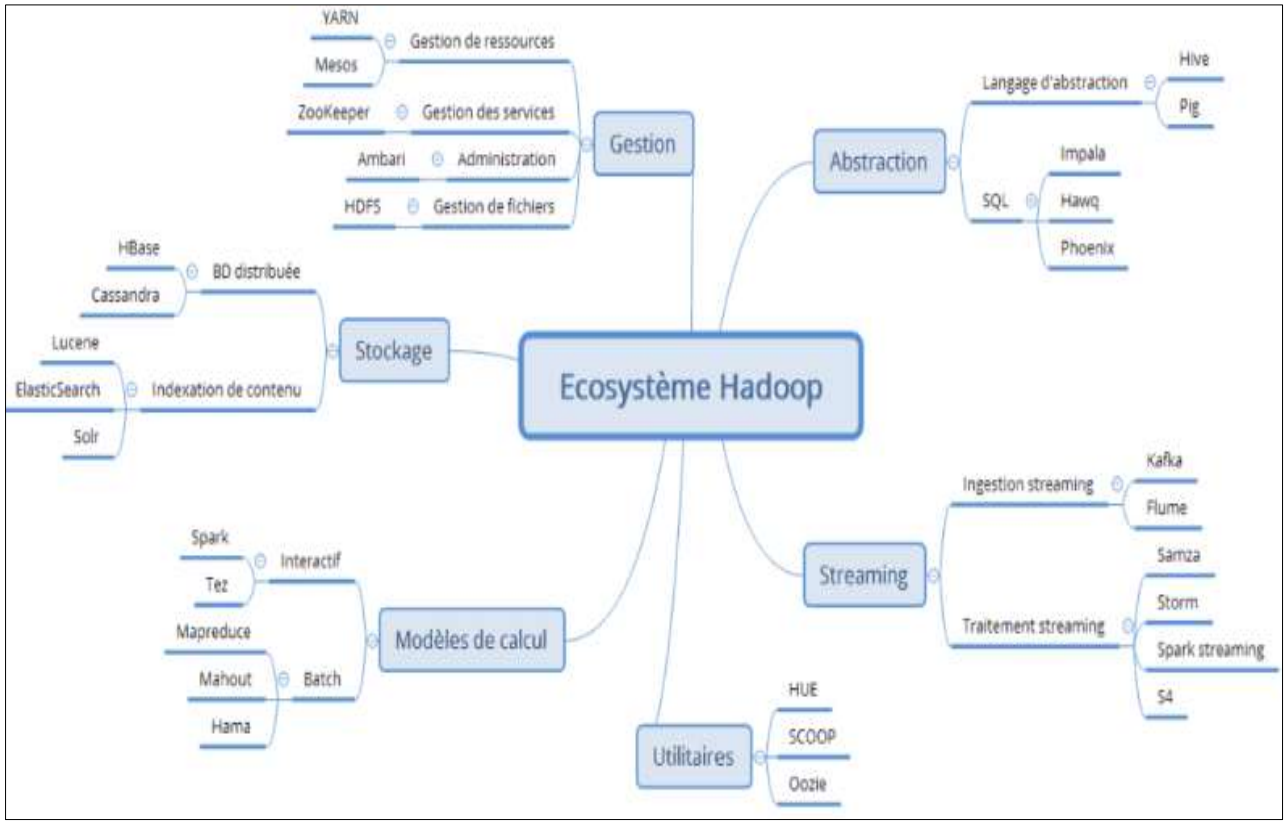
- **Calculs distribués :** MAP REDUCE
- **Traitement BACH :** Hive, Pig, Mahout
- **Traitement temps réel :** Impala, Drill
- **Intégration**
  - **Intégration de données NON structurées :** FLUME, …
  - **Intégration de données structurées :** SQOOP, …
- **Stockage :** HBASE, HDFS
- **Ressources et coordination**
  - **Coordination :** ZOOKEEPER
  - **Gestion des ressources :** YARN



Module M4.6, section 2, partie 1 : Architectures Big Data, Eco-système HADOOP

➢ Architecture des composants HADOOP

- **Algorithme Map Reduce**
- Services de Stockage
- Gestion et Administration
- Couches d'Abstractions
- Gestion des Streams
- Divers Utilitaires
- Modèles de calcul
- Zoom sur divers composants



Algorithme Map Reduce

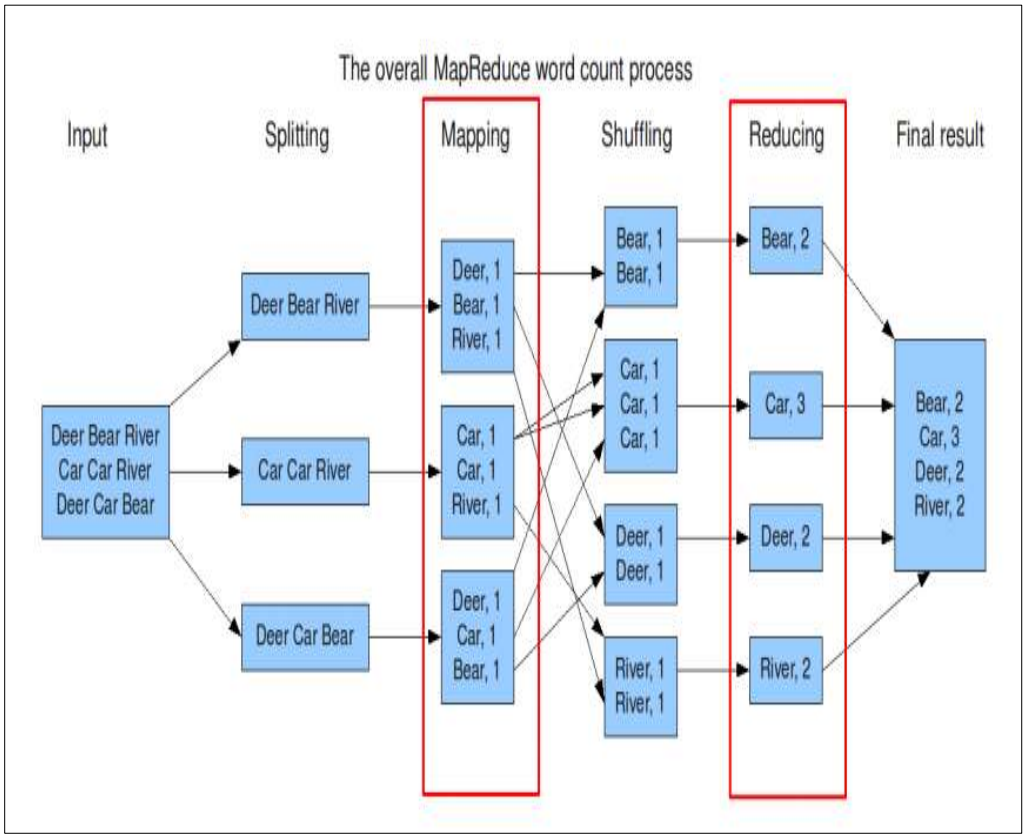
- Technique d'indexation et de recherche sur les données distribuées et volumineuses développée par GOOGLE
- **mapReduce**(listeClés, fonctionMap, fonctionReduce)
- **Activité en deux phases** : Map et Reduce
  - **Phase 1 : Map : function map()**
    - ✓ Extraire un ensemble de paires Clé/Valeur sur les données sous-jacentes
    - ✓ Avec **parallélisation** possible sur de **multiples nœuds**
  - **Phase 2 : Reduce : fonction reduce()**
    - ✓ Fusionner et trier l'ensemble des paires Clé/valeur
    - ✓ Les résultats pourront servir à de nouvelles recherches

Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

Algorithme Map Reduce

- De Mathieu Dumoulin (GRAAL)
- http://www2.ift.ulaval.ca/~quimper/Seminaires/files/Introduction_aux_algorithmes_MapReduce.pdf

- MapReduce et le comptage de mots



Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢Algorithme Map Reduce

- MapReduce peut être utilisé pour un grand nombre d'applications
  - grep distribué, tri distribué, inversion du graphe des liens web, vecteur de terme par hôte, statistiques d'accès au web, construction d'index inversé,
  - classification automatique de documents, apprentissage automatique, …

- MapReduce est **adapté pour des traitements Batch** et non temps réel

Module M4.6, section 2, partie 1 : Architectures Big Data, Eco-système HADOOP

➢Services de stockage dans Hadoop

- **BD distribuées/ répliquées/ NOSQL**
  - **HBASE** : HBase est un SGBD NOSQL distribué/répliqué, orienté-colonne qui fournit l'accès en temps réel aussi bien en lecture qu'en écriture aux données stockées sur le HDFS
  - **CASSANDRA** : Apache Cassandra est un SGBD NOSQL distribué/répliqué, orienté-colonne, conçu pour gérer des quantités massives de données sur un grand nombre de serveurs
- **Indexation de contenus**
  - **Lucene** : Lucene est une bibliothèque open source écrite en Java qui permet d'indexer et de chercher du text
  - **ElasticSearch** : Elasticsearch est un logiciel utilisant Lucene pour l'indexation et la recherche de données. Il fournit un moteur de recherche distribué et multi-entité à travers une interface REST
  - **Solr** : Solr est une plateforme logicielle de moteur de recherche s'appuyant sur la bibliothèque de recherche Lucene, créée par la Fondation Apache et distribuée et conçue sous licence libre
- **Gestion de fichiers**
  - **HDFS :** Le système de fichier distribué, qui gère le stockage distribué des données et fournit la tolérance aux pannes nécessaire lors de l'exploitation d'un cluster. **HDFS peut être utilisé comme DATA LAKE ou une partie du DATA LAKE**

Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢Gestion et administration des composants dans Hadoop

▪ Gestion des ressources

- **Yarn :** YARN permet de **faire tourner plusieurs moteurs de calcul dans le cluster** et d'exploiter son potentiel à son maximum
- **Mesos :** Le noyau Mesos fonctionne sur chaque machine et fournit aux applications (par exemple, Hadoop, Spark, Kafka, Elasticsearch) des API pour la gestion et l'ordonnancement des ressources dans des environnements entiers de centres de données et de nuages.la gestion et la planification des ressources dans l'ensemble du centre de données et des environnements en nuage

▪ Gestion des services

- **Zookeeper:** ZooKeeper prend **en charge la complexité inhérente de la synchronisation de l'exécution des tâches distribuées dans le cluster** et permet aux autres outils de l'écosystème Hadoop de ne pas avoir à gérer ce problème eux-mêmes

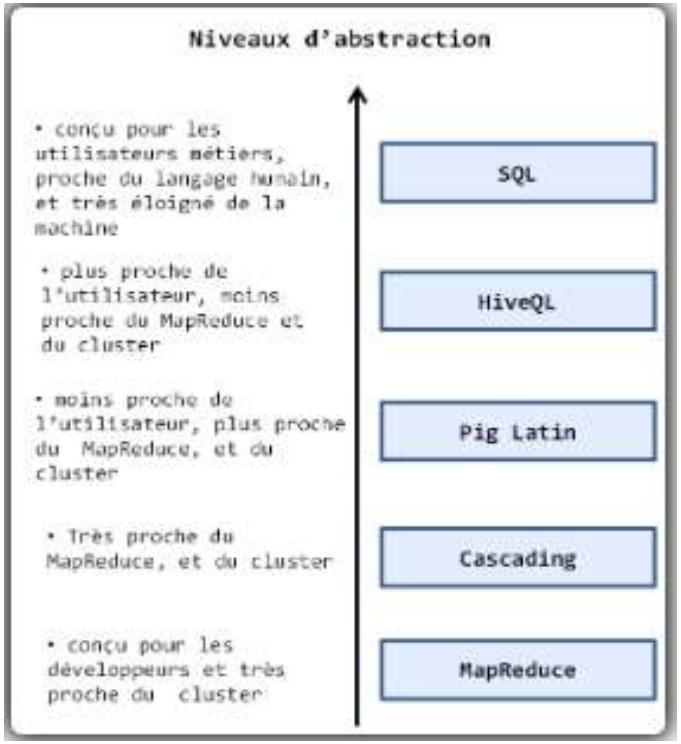
▪ Administration

- **Ambari :** Le projet Apache Ambari vise à **simplifier la gestion des clusters Apache Hadoop à l'aide d'une interface utilisateur Web**. Il s'intègre également à d'autres applications existantes grâce aux API REST d'Ambari[2].

Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢Couches d'Abstractions de Map Reduce dans Hadoop

- **Langage d'abstraction**
  - **HIVE :** Hive est un SGBD Relationnel SQL2 pour construire des **Data Warehouse/Data lake multi-sources**. Hive fournit des services de requêtes et d'agrégation de très gros volumes de données stockées sur un système de fichier distribué de type HDFS
  - **Pig :** Pig est une **plateforme haut niveau pour la création de programme MapReduce utilisé avec Hadoop**. Le langage de cette plateforme est appelé le Pig Latin
- **SQL**
  - **Impala** :Apache Impala est un **moteur de requêtes SQL** open source de Cloudera (MPP) pour les données stockées dans des clusters d'ordinateurs exécutant Apache Hadoop2
  - **Hawq** : Apache HAWQ est le **moteur de requêtes SQL Natif d'Hadoop**
  - **Phoenix** : Apache Phoenix **est un moteur de base de données relationnel open source, massivement parallèle, supportant OLTP pour Hadoop** utilisant Apache **HBase** comme support de sauvegarde



Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢Gestion des Streams

▪ Ingestion streaming

- **Kafka** : Apache Kafka est une **plateforme distribuée de diffusion de données en continu**, **capable de publier, stocker, traiter et souscrire à des flux d'enregistrement en temps réel**. Elle est conçue pour gérer des flux de données provenant de plusieurs sources et les fournir à plusieurs utilisateurs.
- **Flume** : **Flume est une solution de collecte, agrégation et transfert de gros volumes de logs**. Il a été pensé pour gérer des débits importants avec une fonctionnalité native d'écriture dans HDFS au fil de l'eau. Sert à intégrer les données non-structurées

▪ Traitement

- **Samza** : **Apache Samza est un framework de calcul asynchrone open source quasi temps-réel** pour le traitement de flux développé par Apache Software Foundation en langage Scala et Java .
- **Storm** : Il **permet de développer** sous Hadoop des applications qui traitent les données **en temps réel Spark streaming**
- **S4** : S4 (**S**imple **S**calable **S**treaming System) est une plate-forme polyvalente, distribuée, évolutive, tolérante aux pannes et enfichable qui permet aux programmeurs de **développer facilement des applications pour le traitement de flux de données continus et illimités**

Divers Utilitaires

- **HUE** : Hue (Hadoop user experience) **est une interface web open-source** sous licence Apache V2 **prenant en charge Hadoop et son écosystème**.
- **SQOOP :** Sqoop ou SQL-to-Hadoop est un outil qui permet **de transférer les données d'une base de données relationnelle vers Hadoop HDFS et vice-verça**
- **Oozie :** Comment fait-on pour gérer l'exécution de plusieurs jobs qui sont relatifs au même problème ? Pour gérer ce type de problème, la solution la plus simple actuellement consiste à **utiliser un planificateur de jobs, en l'occurrence Oozie**. Oozie est un planificateur d'exécution des jobs qui fonctionne comme un service sur un cluster Hadoop

Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢Modèles de calcul interactif ou Batch

▪ Modèle de calcul Interactif

- **SPARK** : Spark est un framework open source de calcul distribué. **C'est un moteur de calcul qui effectue des traitements distribués en mémoire sur un cluster**. Comparativement au MapReduce qui fonctionne en mode batch, le modèle de calcul de Spark fonctionne en mode interactif (plusieurs traitements en une étape)
- **Tez :** Apache Tez est un framework **d'exécution sur Hadoop YARN**. Il peut être utilisé à la place de **MapReduce**

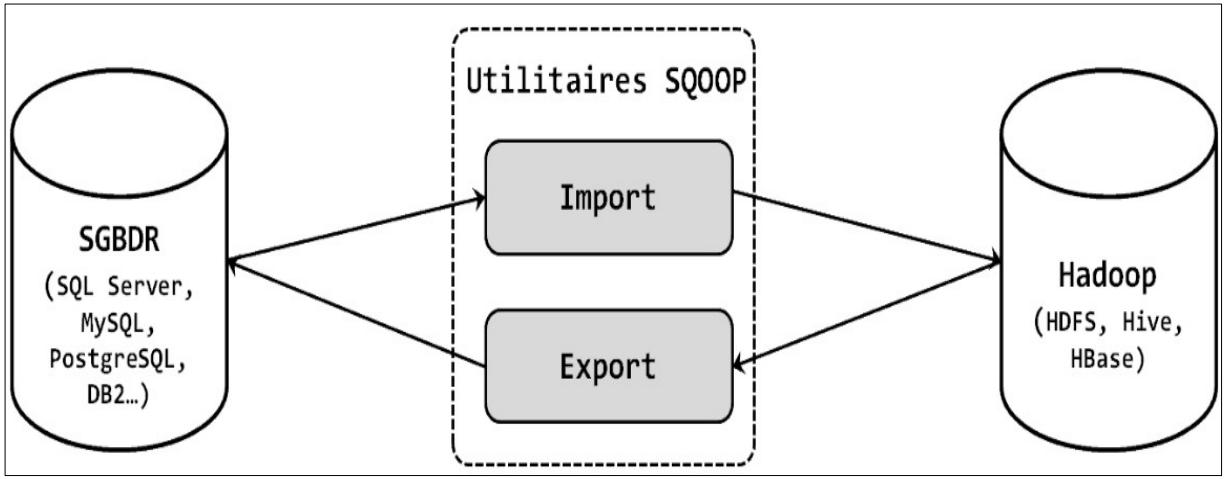
▪ Modèle de calcul Batch

- **Mapreduce :** MapReduce est un patron de conception de développement informatique, inventé par Google, dans lequel **sont effectués des calculs parallèles, et souvent distribués, de données** potentiellement très volumineuses, typiquement supérieures en taille à un téraoctet
- **Mahout :** Apache Mahout est un projet de la fondation Apache visant à **créer des implémentations d'algorithmes d'apprentissage automatique distribués**
- **Hama :** Apache Hama est **un framework de calcul distribué** basé sur des techniques de calcul parallèle synchrone en masse **pour des calculs scientifiques massifs, par exemple des algorithmes de matrice, de graphe et de réseau**.

Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢Zoom sur l'utilitaire SQOOP

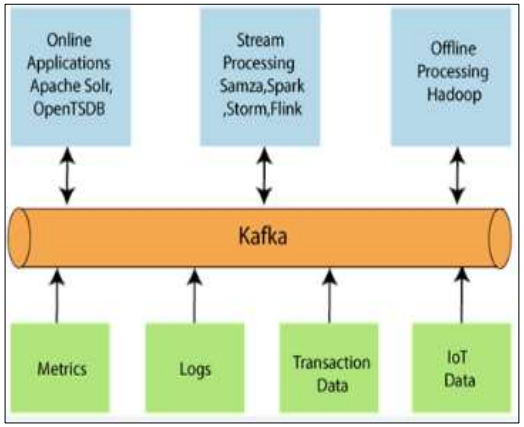
- **SQOOP** ou **SQ**l-to-Had**OOP** est un outil qui permet **de transférer les données d'une base de données relationnelle vers Hadoop HDFS et vice-versa**

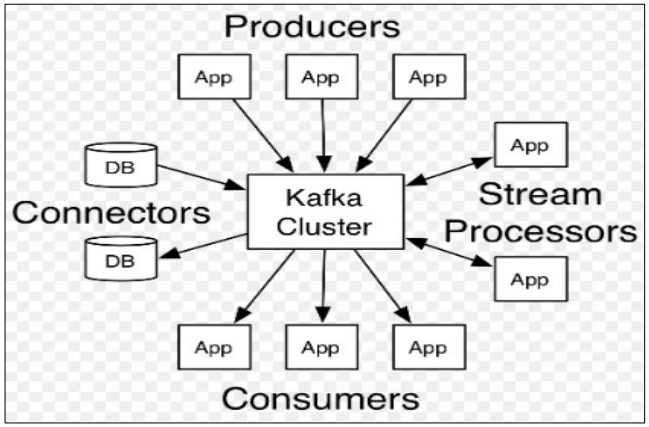


Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢Zoom sur Apache KAFKA

- Apache Kafka est une **plateforme distribuée de diffusion de données en continu**, **capable de publier, stocker, traiter et souscrire à des flux d'enregistrement en temps réel**. Elle est conçue pour gérer des flux de données provenant de plusieurs sources et les fournir à plusieurs utilisateurs

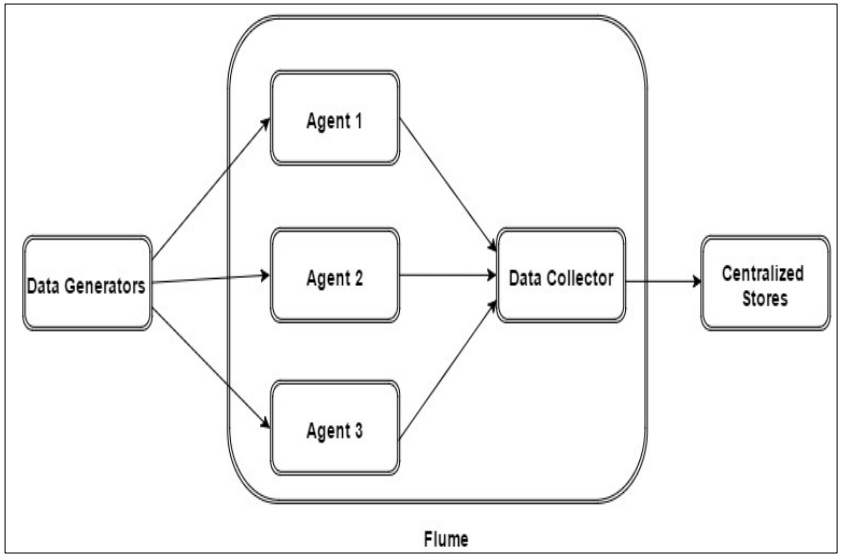




Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢Zoom sur Apache FLUME

- **Flume** : **Flume est une solution de collecte, agrégation et transfert de gros volumes de données**. Il a été pensé pour gérer des débits importants avec une fonctionnalité native d'écriture dans HDFS au fil de l'eau. Sert à intégrer des données non-structurées

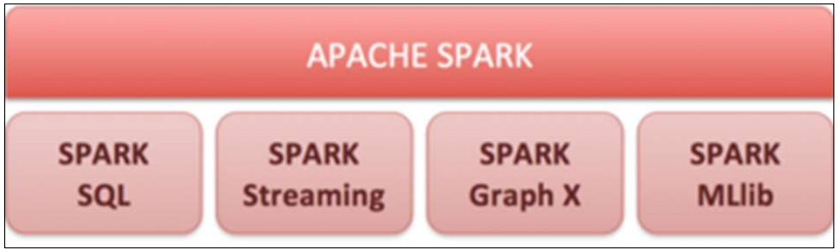


Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢ Zoom sur Apache SPARK

- **SPARK** : Spark est un framework open source de calcul distribué. **C'est un moteur de calcul qui effectue des traitements distribués en mémoire sur un cluster**. **Comparativement au MapReduce qui fonctionne en mode batch**, le modèle de calcul de Spark fonctionne en mode interactif (plusieurs traitements en une étape)

- **Composants SPARK**



- 3 fois plus rapide qu'Hadoop
- Fonctionne en mémoire
- Peut être utilisé avec Hadoop
- Ecrit en scala
- Plusieurs API natives : Scala, Java, Python, Sql, R

Module M4.6, section 2 , partie 1 : Architectures Big Data, Eco-système HADOOP

➢Zoom sur Apache SPARK

- **Composants SPARK**
  - **Spark SQL :** Spark SQL permet d'exécuter des requêtes en langage SQL sur des données hétérogènes
  - **Spark Streaming :** Permet d'effectuer un traitement des données en flux. Il utilise les données en temps-réel DStream (discretized stream) c'est-à-dire une série continue de RDD (Resilient Distributed Datasets). Par exemple Twitter ou Amazon Kinesis.
  - **Spark Graph X :** Spark Graph X permet de traiter les informations issues de graphes. Graph X étend les RDD de Spark
  - **Spark Mllib** : C'est une bibliothèque d'apprentissage automatique, qui contient tous les algorithmes et utilitaires d'apprentissage classiques, comme la classification, la régression, le clustering, le filtrage collaboratif ...

Module M4.6, section 2 , partie 1 : QUIZ

➢ **Question 1 : Le framework HADOOP a été conçu pour**

- A: Effectuer des calculs sur de gros volumes de données
- B: Fournir aux acteurs du Big Data un certain nombre d'outils
- C: Être la solution unique dès qu'on veut mettre en place un projet Big data
- D: Être l'architecture parallèle installée dans le cloud par la fondation Apache

➢ **Question 2 : Le framework HADOOP fournit un ensemble d'outils pour le traitement des données volumineuses. Cochez ce qui est juste**

- A: Les outils de stockages
- B: Des outils pour construire des Data warehouse
- C: Des outils de machine learning
- D: Des outils jouant le rôle d'ETL
- E: Des outils jouant le rôle 'ELT
- F: Des outils d'envoie des tweets

Module M4.6, section 2 , partie 1 : QUIZ

➢ **Question 3 : Cochez le ou les SGBD disponible dans l'environnement Hadoop**

- A: Cassandra
- B: Flume
- C: Cosmos DB
- D: MongoDB
- E: Voldemort
- F: HBASE
- G: HDFS
- H: NTFS

➢ **Question 4 : Cochez le ou les Systèmes de fichiers dans l'environnement Hadoop**

- A: Cassandra
- B: Flume
- C: Cosmos DB
- D: MongoDB
- E: Voldemort
- F: HBASE
- G: HDFS
- H: NTFS

Module M4.6, section 2 , partie 1 : QUIZ

- **Question 5 : Une ville du sud de la France qui souhaite augmenter la sécurité de ses concitoyens, s'est dotée de plusieurs centaines de caméras. Elle souhaite analyser au fil de l'eau les vidéos de ces dernières. Cochez les outils pouvant être utilisées à cette fin**
  - A: MAP REDUCE
  - B: Flume
  - C: Kafka
  - D: Mahout
  - E: Spark

- **Question 6 : Cochez ce qui est un composant SPARK**
  - A: Spark SQL
  - B: Spark C++
  - C: Spark Streaming
  - D: Spark Graph X
  - E: Spark Mlib
  - F: Spark Map Reduce

UNIVERSITÉ CÔTE D'AZUR

ESTIA INSTITUTE OF TECHNOLOGY

DATUM ACADEMY

Module M4.6, section 2, partie 2 : Architectures Big Data

FUN FRANCE UNIVERSITÉ NUMÉRIQUE

anr

INVESTIR L'AVENIR

MIAGE RÉSEAU DES MIAGE DE FRANCE Nice

MBDS

BIHAR

Module M4.6, section 2, partie 2 : Architectures Big Data

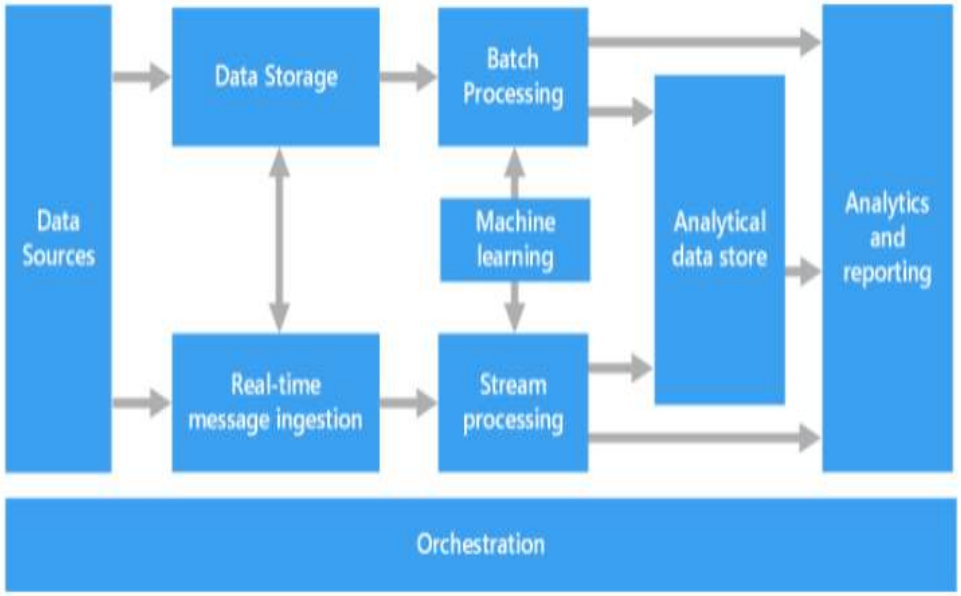
Plan

- Composants d'une architecture Big Data : Vision MicroSoft
- Architecture Big Data Oracle
- Architecture Big Data Google
- Architecture Big Data IBM
- Composants d'une architecture Big Data: Vision générique autour d'HADOOP et les BD NoSql

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Composants d'une architecture Big Data : Vision MicroSoft

https://docs.microsoft.com/fr-fr/azure/architecture/data-guide/big-data/



Module M4.6, section 2 , partie 2 : Architectures Big Data

➢ Composants d'une architecture Big Data : vision Microsoft

- **Data sources** : Sources de données diverses (Bd relationnelles, NoSql, fichiers, reseaux sociaux, etc.)
- **Data storage** : Entrepôt de données ou lac de données
- **Batch processing** : chargement et traitement en temps différé des données (ETL, extracteurs de donnes)
- **Real-time message ingestion** : Solution pour le traitement temps réel si l'architecture l'exige
- **Stream processing**. : Traitement temps réel des flots de données (filtrage, agrégation, préparation, …)

Composants d'une architecture Big Data : vision Microsoft

- **Analytical data store**.: Entrepôt de données pour l'analyse. Permet d'y stocker les données préparées venant du lac de données à des fin d'Analyse

- **Analysis and reporting** : Utilisation des outils d'analyses et de reporting, Utilisation des outils de machine learning, …

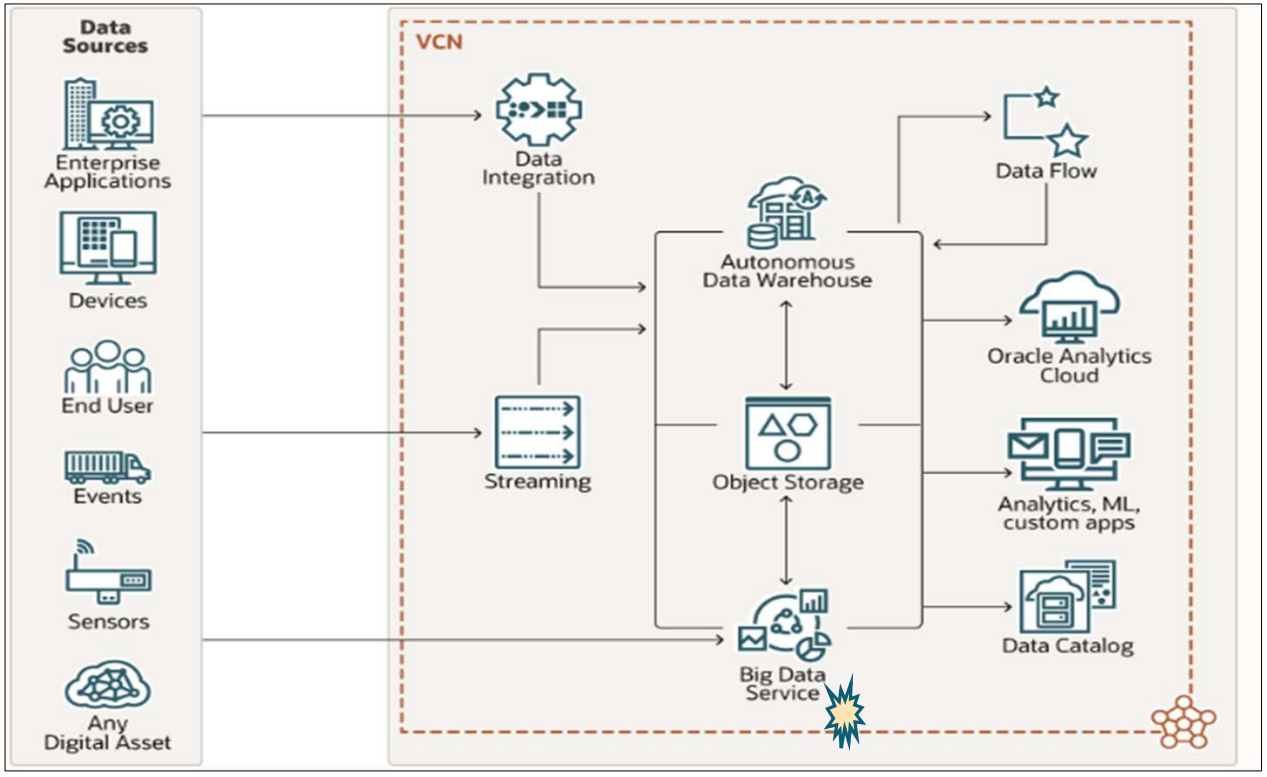
- **Orchestration.** Orchestration des opérations de traitement de données répétées, encapsulées dans des workflows (ELT: Extract Transform and Load avec workflow)
  - **Transformation** des données source,
  - **Déplacement des données entre plusieurs sources et récepteurs**,
  - **Chargement des données traitées** dans un magasin de données analytique

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Architecture Big Data Oracle

- **Avec Big Data Services** : autour de Hadoop

https://docs.oracle.com/fr/solutions/big-data-and-analytics/index.html#GUID-04F64035-CDDA-4FBF-BCCB-A578032FA90B

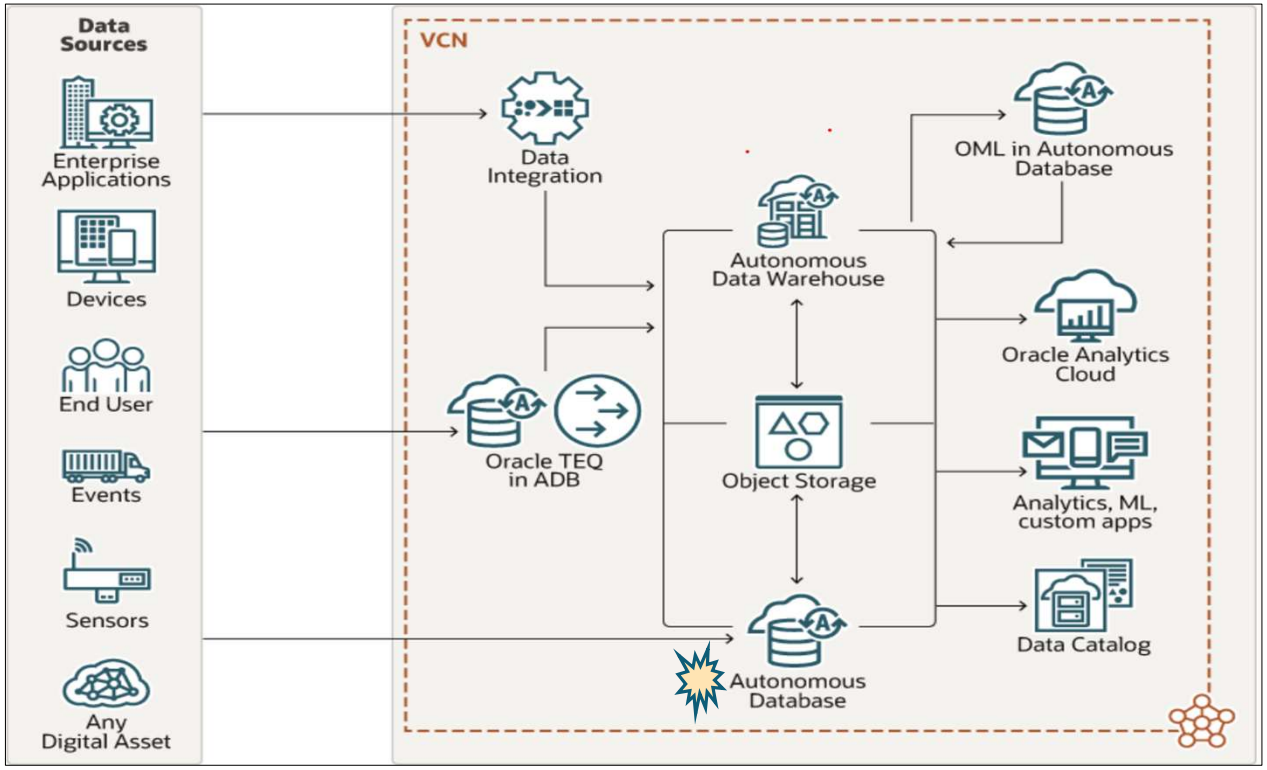


Module M4.6, section 2 , partie 2 : Architectures Big Data

➢ Architecture Big Data Oracle

- **Avec Big Data Services :**
  autour **D'Oracle Autonomous Database**

ADB : Autonomous DataBase
TEQ : Transactionnal Event Queue



Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Architecture Big Data Oracle

Composants Big Data	Description
Data sources	**Données venant de** : Applications Enterprise, **Périphériques**, Utilisateur final, **Evénements**, Capteurs, **Toutes les sources numériques (réseaux sociaux, …)**
VCN : Virtual Cloud Network	Un **VCN** est un réseau personnalisable et défini par logiciel que vous configurez dans une région Oracle Cloud Infrastructure
Data intégration	**Oracle Cloud Infrastructure Data Integration**: service intégré sans serveur. Simplifie les processus ETL et ELT dans les Data lake et DWH
Streaming	Le service **Oracle Cloud Infrastructure Streaming** offre une solution pour l'ingestion et l'utilisation de flux de données volumineuses en temps réel

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Architecture Big Data Oracle

Composants Big Data	Description
Big Data Service	Service cloud automatisé entièrement géré qui fournit aux clusters un environnement Hadoop. **Permet de déployer facilement des clusters Hadoop (HDFS, HIVE, HBASE, …). DATA LAKE AUTOUR D'HADOOP**
Oracle Autonomous Data Warehouse	**Service de base de données autonome**, auto-sécurisé et auto-réparateur optimisé pour jouer le rôle d'entrepôt de données ou de lac de données
Object Storage	Est un service qui fournit un stockage d'objet fiable, sécurisé et évolutif **sous forme de fichiers**. Il permet de télécharger, classifier (par niveau de mémoire de stockage) des fichiers dans des buckets
Autonomous Database	Permet dans la **2ème architecture** de remplacer le Big Data Service Hadoop

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Architecture Big Data Oracle

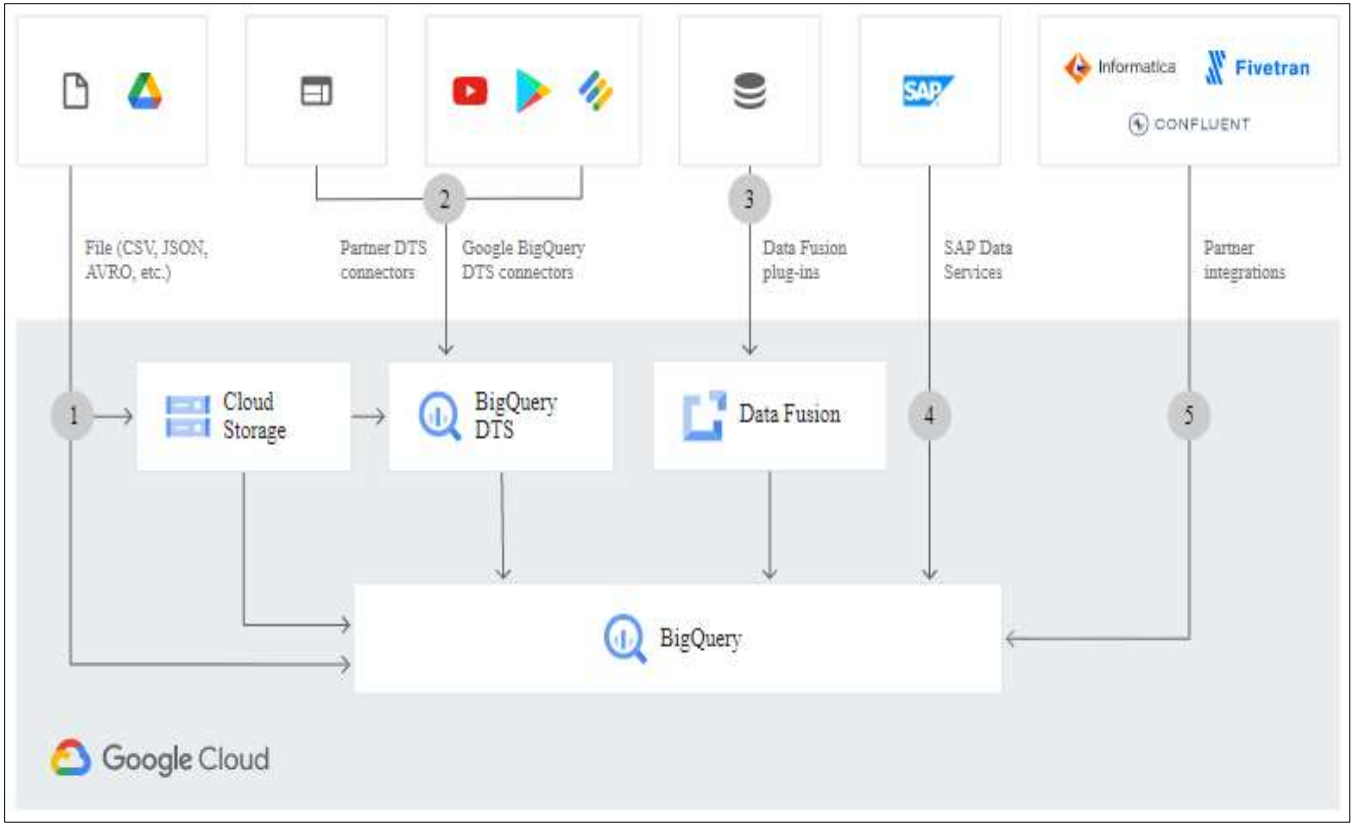
Composants Big data	Description
Oracle cloud data flow	Oracle Cloud Infrastructure Data Flow est une plate-forme d'analyse Spark de niveau PaaS entièrement gérée qui vous permet de créer, de modifier et d'exécuter des travaux Spark
Oracle Machine learning in autonomous DWH/Data lake	Services d'analyse, Oracle Machine Learning intégrés dans le SGBD Oracle
Oracle Analytics cloud	Oracle Analytics Cloud offre des services basées sur l'IA pour la préparation, la découverte et la visualisation des données, le reporting, le traitement et la génération du langage naturel
Analytics, machine learning customs applications	Services d'analyse, Oracle Machine Learning et applications personnalisées qui vont cataloguer, préparer, traiter et analyser le Big Data
Data catalog	Oracle Cloud Infrastructure Data Catalog est une solution de repérage et de gouvernance de données en libre-service entièrement gérée pour vos données d'entreprise

Module M4.6, section 2 , partie 2 : Architectures Big Data

Architecture Big Data
Google BigQuery

https://cloud.google.com/bigquery#all-features

BigQuery est un entrepôt de données d'entreprise sans serveurs à gérer



Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Architecture Big Data Google BigQuery

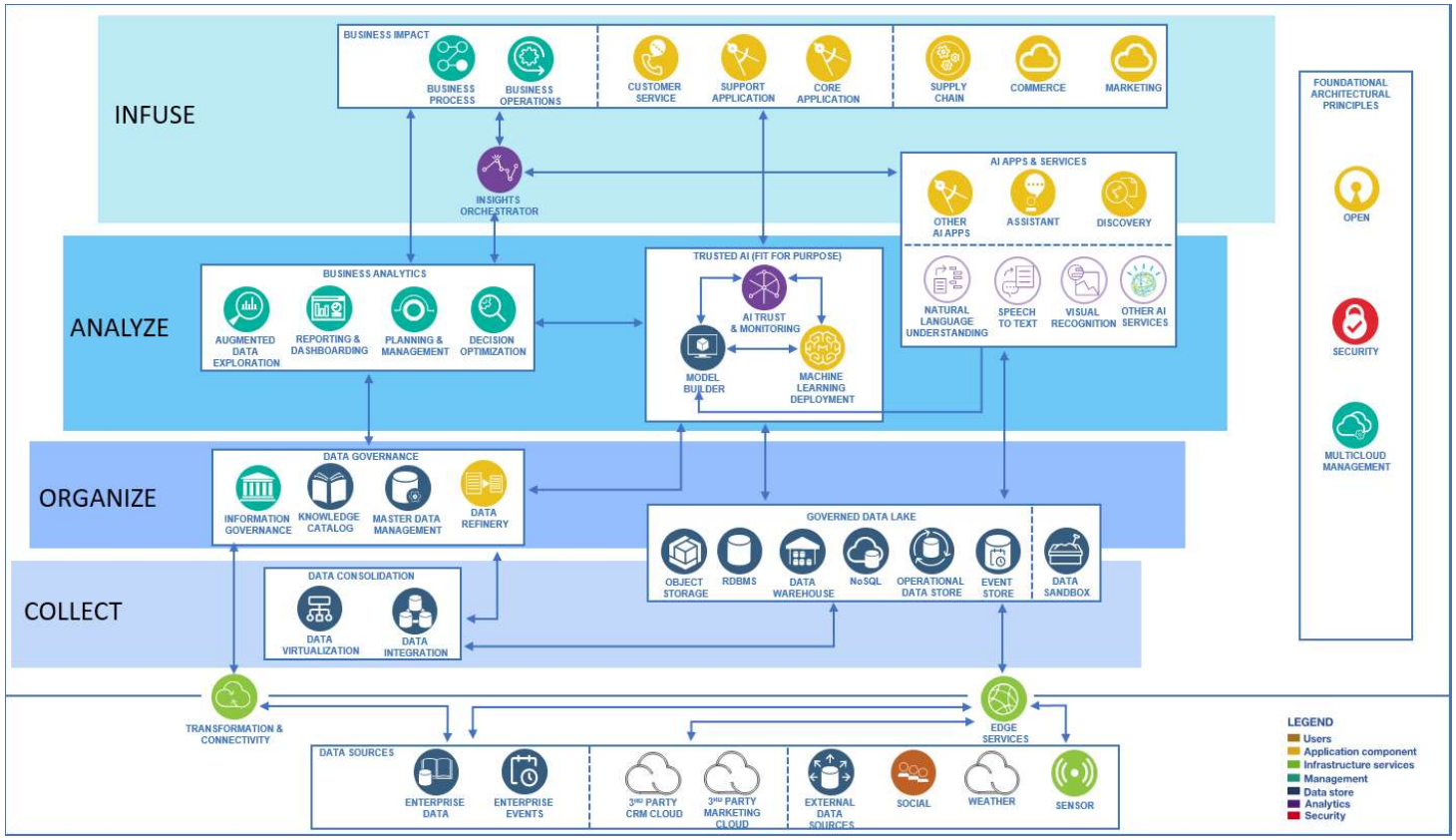
Composant Big Query	Description
Machine learning (ML) intégrée	Avec BigQuery ML les analystes peuvent créer et utiliser des modèles de ML sur des données structurées ou semi-structurées, et non structurées
Streaming intégré	Analyses en temps réel avec accélération des requêtes intégrées
BigLake intégré	Interrogez tous les types de données avec BigQuery : structurées, semi-structurées et non structurées. Utilisez BigLake pour explorer et unifier différents types de données, et créez des modèles avancés
BigQuery SQL intégré	Langage SQL s'appuyant entre autres sur les tables externes pour interroger les données BigLake ou d'autres sources externes (données hétérogènes et multi-sources)

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢ **Architecture Big Data**

IBM

https://www.ibm.com/cloud/architecture/architectures/dataArchitecture/reference-architecture/



Module M4.6, section 2 , partie 2 : Architectures Big Data

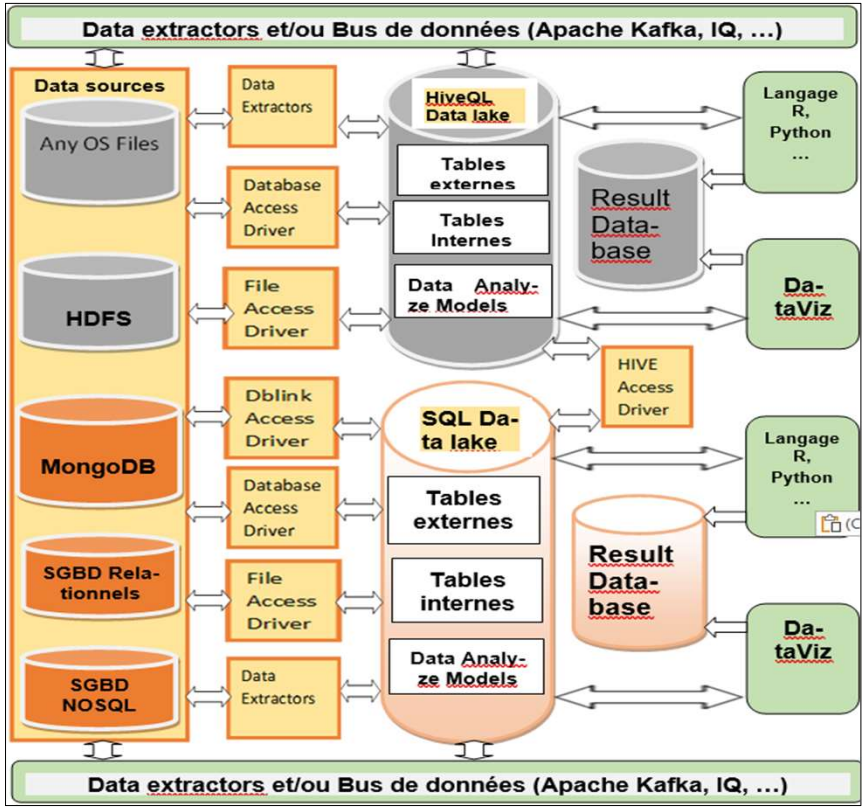
➢Architecture Big Data IBM

Composants Big Data IBM	Description
Data sources	Données des Entreprises, Evénements, logs, CRM, Capteurs, Toutes les sources numériques (réseaux sociaux, temps, …)
Data consolidation (collecte)	Data virtualisation, data intégration
Data lake gouverné (administré)	Object storage (non structurées), SGBDR, Data warehouse, sgbd nosql, operational data store(modèles), events store, bac à sable de données
Business Analytics	Rapport, tableaux de bord, aide à la décision
Applications et services d'intelligence artificielle	Machine learning, deap learning, natural language processing (NLP), traduction automaique, …

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢ **Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql**

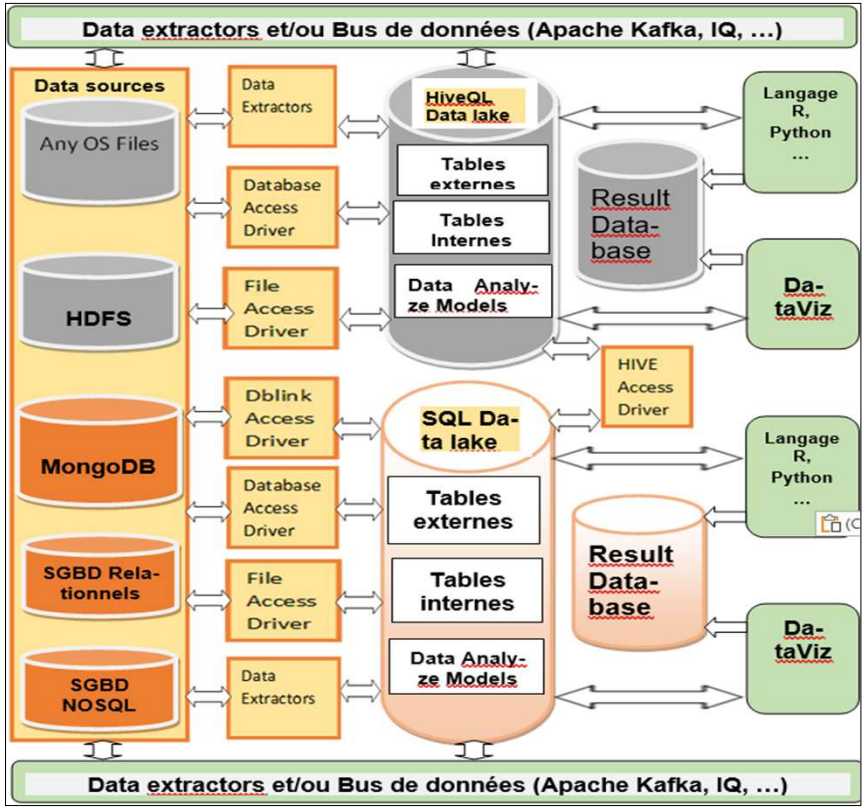
- **Sources de données**
  - **Fichiers d'un OS quelconque** (non structuré, flots)
  - **Fichier**s dans Hadoop **HDFS** (non structuré, flots)
  - **SGBD relationnel** (structurées)
  - **SGBD NOSQL** (non structurées, semi-structurées, structurées)



Module M4.6, section 2 , partie 2 : Architectures Big Data

➢ **Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql**

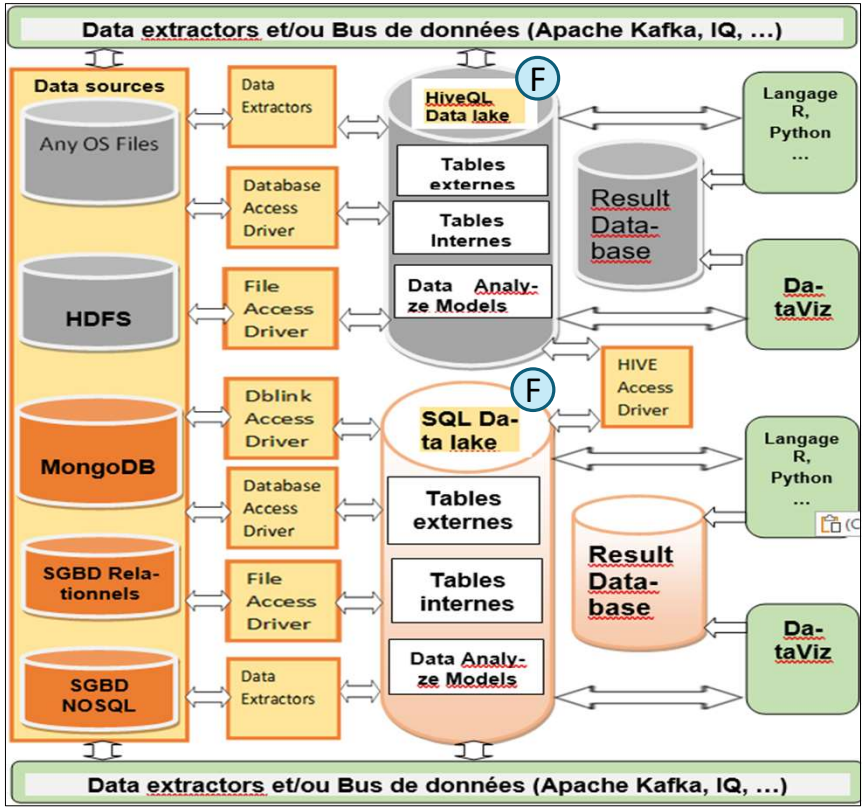
- **Lac de données BRUT**
  - **Fichiers d'un OS quelconque** (non structuré, flots)
  - **Fichier**s dans Hadoop **HDFS** (non structuré, flots)
  - **SGBD relationnel** (structurées)
  - **SGBD NOSQL** (non structurées, semi-structurées, structurées)



Module M4.6, section 2 , partie 2 : Architectures Big Data

➢ **Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql**

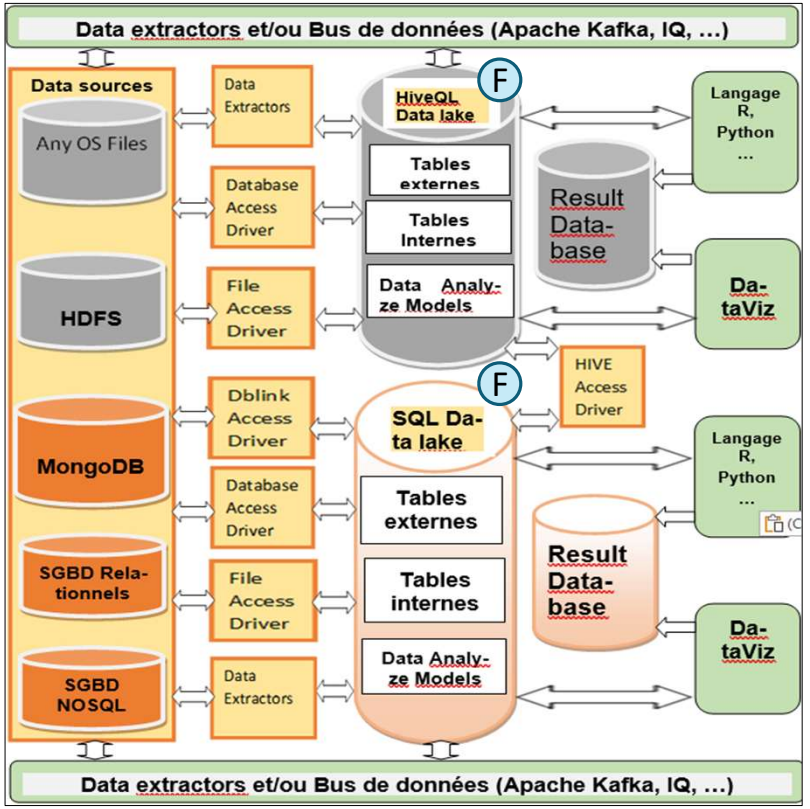
- **Lac de données SEMI-BRUT (c'est ce que nous appellerons lac de données par la suite)**
  - **FRONTAL BIG DATA SQL DU LAC**: SGBD relationnel dans Hadoop (HIVE, autres) et /ou SGBD Relationnel du marché (Oracle, Mysql, Microsoft, ...) avec des données physiques ou virtuelles (bon pour le temps réel et l'accès multi-sources)
  - **Fichiers d'un OS quelconque** : si Accès virtuel depuis le FRONTAL **BIG DATA SQL**
  - **Fichiers dans Hadoop HDFS** : si Accès virtuel depuis le **FRONTAL BIG DATA SQL**
  - **SGBD relationnel** : si Accès virtuel depuis le FRONTAL
  - **SGBD NOSQL**: si Accès virtuel depuis le **FRONTAL BIG DATA SQL**



Module M4.6, section 2 , partie 2 : Architectures Big Data

➢ Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql

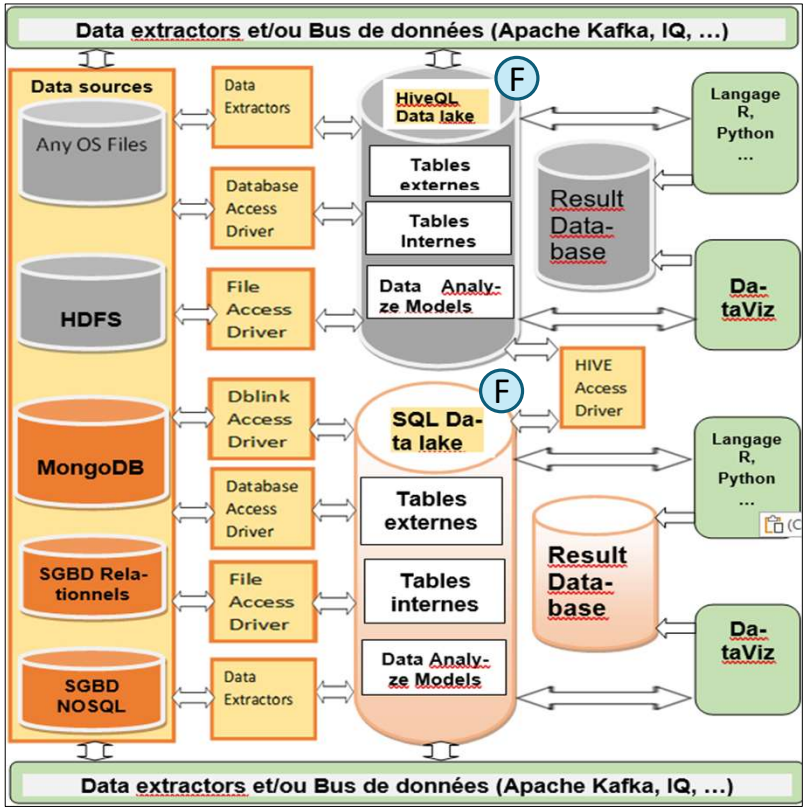
- **FRONTAL BIG DATA SQL DU LAC DE DONNEES**
  - **SGBD relationnel dans Hadoop** (HIVE, autres) **et /ou SGBD Relationnel du marché** (Oracle, Mysql, Microsoft, ...) avec des données physiques ou virtuelles
  - **Proche de la philosophie DATA WAREHOUSE(DWH)** : nos données sont visibles sous forme de tables internes ou externes. Les tables ici ne sont pas prémédités comme dans un DWH
  - **Accessible en SQL comme dans un DWH** (Big Data SQL ici)
  - **Défendu par plusieurs acteurs du Big Data** (**Google** : Big Query, **Oracle** : Oracle Big Data SQL, **Hadoop: HiveQL**, ...)



Module M4.6, section 2 , partie 2 : Architectures Big Data

➢ Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql

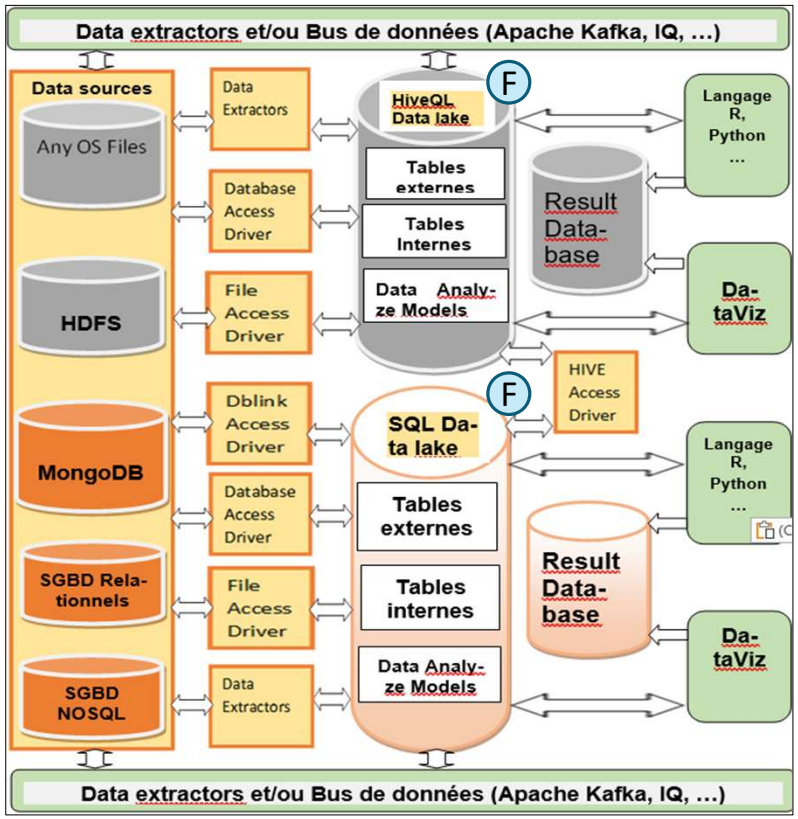
- **FRONTAL BIG DATA SQL DU LAC DE DONNEES**
  - **Les données virtuelles**
    - Existent **grâces aux tables externes** qui pointent vers des tables physiques ou virtuelles et/ou des fichiers physiques
    - Permettent de réduire **le coût des recopies**
    - Permettent **une analyse temps réel**
    - Permettent **le mapping des noms et types des champs**
    - Permettent **de bénéficier de l'algorithme MAP REDUCE**
    - Permettent **la manipulation via SQL de données** multi-sources de façon intelligente



Module M4.6, section 2 , partie 2 : Architectures Big Data

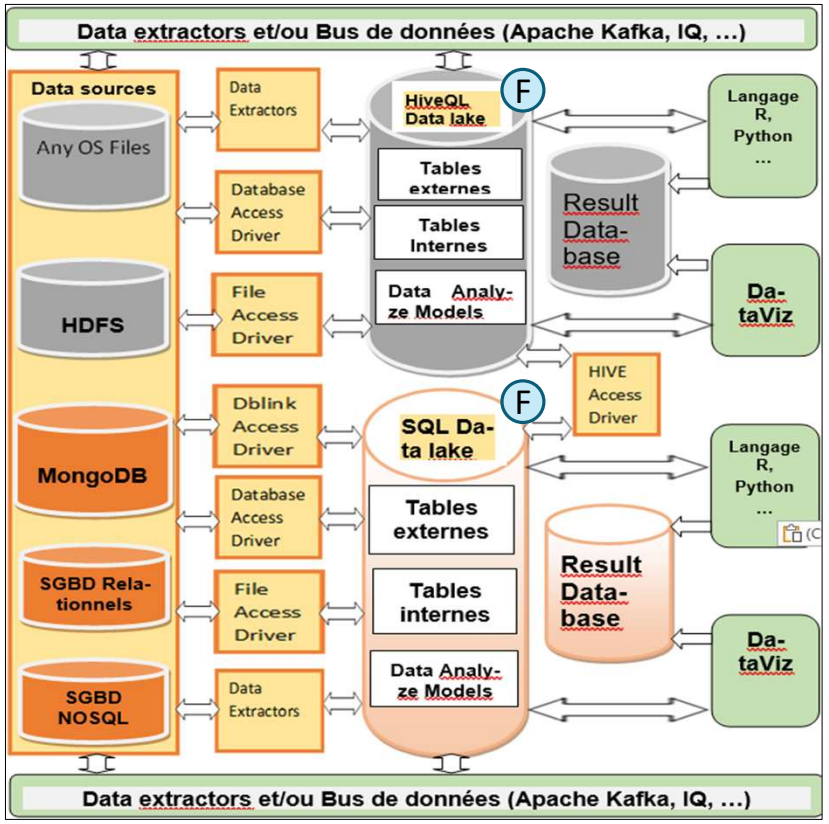
➢ **Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql**

- **FRONTAL BIG DATA SQL DU LAC DE DONNEES : LES BENEFICES**
  - **On peut faire tout ce qu'on faisait avec un DWH**
  - On peut y **brancher des outils d'indexation** (aucune obligation) : **ELASTIC SEARCH, …**
  - On peut y **brancher des outils de visualisation Olap: Cognos, Business object, SAS, …**
  - **,** On peut y brancher **des outils de visualisation Big Data: D3JS, KIBANA, …**
  - On peut utiliser **SQL comme outil de visualisation, traitement, transformation**



Module M4.6, section 2 , partie 2 : Architectures Big Data

- **Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql**
  - **FRONTAL BIG DATA SQL DU LAC DE DONNEES : LES BENEFICES**
    - ✓ On peut y brancher des **frameworks de machine learning : depuis R, depuis Python, Mahout dans Hadoop…**
    - ✓ On peut y brancher des **frameworks de machine learning intégrés dans les SGBD**



Module M4.6, section 2 , partie 2 : Architectures Big Data

➢ **Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql**

- **Les data extractors (**permettent la recopie physique de données**). Ils peuvent être :**
  - **ETL** (Etract Transform and Load). A l'origine ces outils ont été conçus pour alimenter un datawarehouse
  - **ELT** (Extract Load adnd Transform). A l'origine ces outils ont été conçus pour alimenter un data lake
  - **Un chargeur maison**. Il s'agit d'un programme chargé d'extraire et recopier les des données dans le lac de données (à utiliser en même temps ou alternative d'un ETL/ELT)
  - **Un chargeur du SGBD source ou du SGBD du datalake (Oracle :** sql loader**, MongoDB:** mongoImport**, …)**
  - **Un chargeur de l'environnement Hadoop** (Sqoop, Flume, …).

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Composants d'une architecture Big Data: Vision générique autour d'HADOOP et les SGBD NoSql

- **Les access drivers (**permettent l'accès virtuelle à une source**):**
  - **Database access driver** : ce sont des drivers qui vous permettront d'accéder aux données d'une base de données à partir du data lake sans déplacer les données de cette base
  - **Hive access driver** : ce sont des drivers qui vous permettront d'accéder aux données d'une base de données HIVE à partir du data lake sans déplacer les données de cette base
  - **Data file access driver** : ce sont des drivers qui vous permettront d'accéder aux fichiers du file system à partir du data lake sans charger ces fichiers
  - **DBlink access driver** : ce sont des drivers qui vous permettront d'accéder aux données distribuées d'une base de données à partir du data lake sont déplacer les données à travers des tuyaux appelées Database Link.

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql

- **Votre lac de données contiendra en résumé :**
  - **Les données physiques** (data lake physique) chargées avec vos data extractors
  - **Des données virtuelles** (data lake virtuelle) accessibles via vos access drivers et les tables externes
  - **Et des données des modèles d'analyses que vous devez construire** (à partir des datas lake physique et virtuel) avant d'utiliser des outils de machine learning, deap learning, etc. Ces données doivent être présentées sous forme de vues SQL.

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Composants d'une architecture Big Data : Vision générique autour d'HADOOP et les SGBD NoSql

▪ Bus de données

- Un bus de données joue un rôle important dans **l'échange de données en temps réel** entre les sous-systèmes.
- Un bus garantit le **découplage entre producteurs de données** (sources dans l'architecture) et **consommateurs** (data lake dans l'architecture).
- Un bus de données peut supporter des **consommateurs net des producteurs multiples**.

Module M4.6, section 2 , partie 2 : Architectures Big Data

➢Composants d'une architecture Big Da : Vision générique autour d'HADOOP et les SGBD NoSql

▪ La base de données des résultats

- Les **résultats** des analyses et de reporting **pourront être stockées dans une base de résultats**. Vous devez choisir vous-même la nature de cette base. Elle peut être relationnelle ou nosql

Module M4.6, section 2 , partie 2 : QUIZ

➢ **Question 1 : cochez ce qui est un composant de l'architecture Big Data Google BigQuery**

- A: Google BigQuery Machine learning (ML)
- B: Google BigLake (Data lake intégré)
- C: Google BigQuery Autonomous database
- D: Google BigQuery SQL
- E: Google BigPython
- F: Google BigJava

➢ **Question 2** : **cochez ce que peut être le lac de données dans l'architecture Big Data Oracle dans le cloud**

- A: Le lac de données peut être Oracle Autonomous Database
- B: Le lac de données peut Big Data Service (nœud Hadoop)
- C: Le lac de données peut être Oracle Cloud Data Flow
- D: Le lac de données MongoDB

Module M4.6, section 2 , partie 2 : QUIZ

➢ **Question 3 : Cochez ce qui est un composant de l'architecture Big Data Oracle dans le cloud**

- A: Oracle Big Data Service
- B: Oracle Machine learning
- C: Oracle Data catalog
- D: Oracle Object Storage
- E: Oracle Big Data Graph
- F: Oracle sqldeveloper

➢ **Question 4** : **Cochez ce qui ce qui est un dialecte Big data S**QL

- A: Air France Big Data SQL
- B: Google BigQuery SQL
- C: Tweeter Big Data SQL
- D: Oracle Big Data SQL
- E: Hadoop HiveQL
- F: IBM DB2 Big Query
- G: Big Data Microsoft SQL Server

Module M4.6, section 2 , partie 2 : QUIZ

➢ **Question 5 : Dans l'architecture générique, cochez ce qui permet d'alimenter un data lake physique**

- A: Un database link access driver
- B: Un ETL
- C: Un ELT
- D: Un bus de données tel que Kafka
- E: Un data file access drivers
- F: Un data base access drivers

➢ **Question 6 : Dans l'architecture générique, cochez ce qui permet d'alimenter un data lake virtuel**

- A: Un database link access driver
- B: Un ETL
- C: Un ELT
- D: Un bus de données tel que Kafka
- E: Un data file access drivers
- F: Un data base access drivers

Module M4.6, section 2 , partie 2 : QUIZ

➢ **Question 7 : Cochez le système qui peut être le frontal Big Data Sql du data lake dans l'architecture générique**

- A: HIVE, SGBD relationnel dans Hadoop
- B: SGBD Relationnel du marché intégré dans une architecture Big Data
- C: HDFS le système de fichiers d' Hadoop
- D: MongoDB le SGBD NOSQL le plus populaire
- E: HBASE le SGBD NOSQL intégré dans Hadoop

➢ **Question 8** : **Cochez le bénéfice des données virtuelles accessibles depuis le frontal Big Data SQL**

- A: Permettent d'augmenter le coût des recopies
- B: Ne permettent pas une analyse temps réel
- C: Permettent de bénéficier de l'algorithme MAP REDUCE
- D: Permettent la manipulation via SQL de données multi-sources de façon intelligente
- E: On peut y brancher des outils de visualisation et d'analyse sans déplacer de données
- F: On peut y brancher des frameworks de machine learning sans déplacer de données

UNIVERSITÉ CÔTE D'AZUR ESTIA INSTITUTE OF TECHNOLOGY

DATUM ACADEMY

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

FUN FRANCE UNIVERSITÉ NUMÉRIQUE

anr

INVESTIR L'AVENIR

MIAGE RÉSEAU DES MIAGE DE FRANCE Nice

MBDS

BIHAR

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

ESTIA INSTITUTE OF TECHNOLOGY

UNIVERSITÉ CÔTE D'AZUR

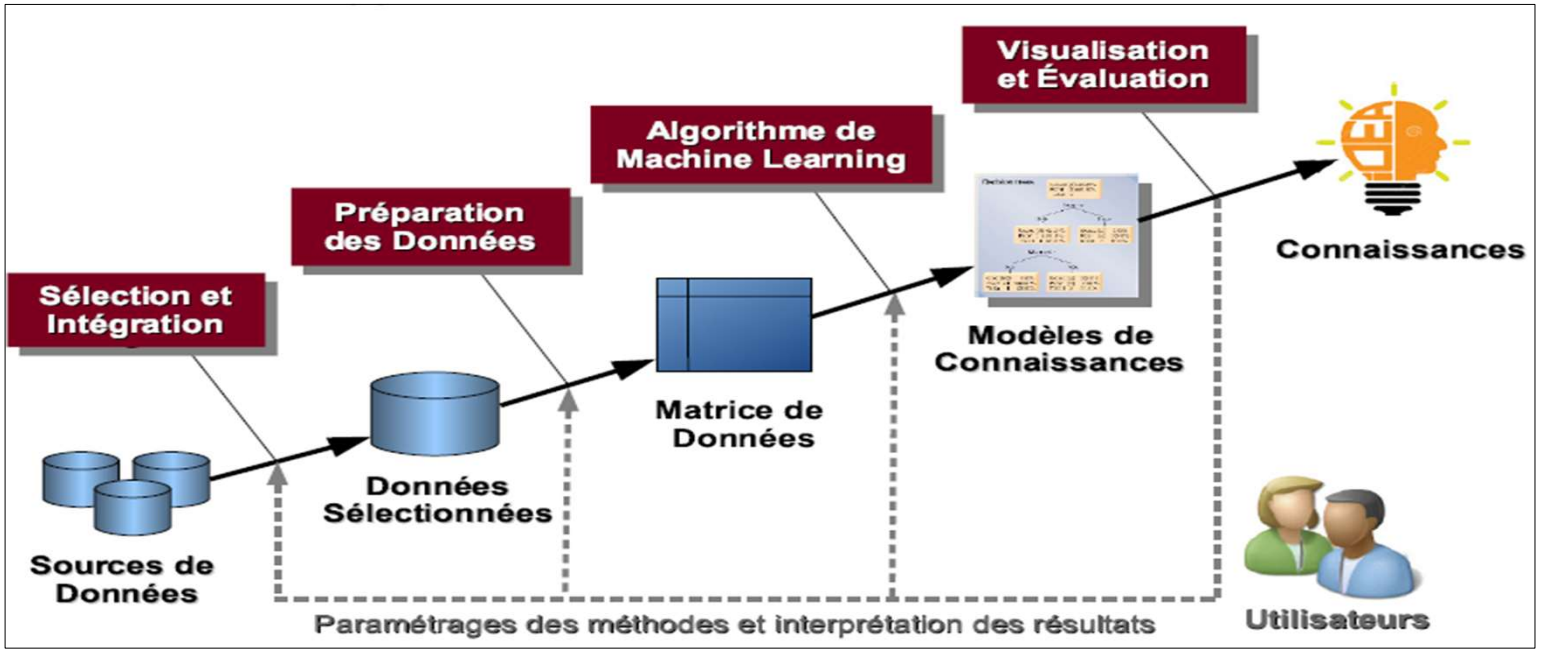
Plan

- Avant-propos
- Architecture d'un lac de données virtuel et/ou physique
- Accès aux données Oracle Nosql via HiveQL
- Accès aux données HDFS via HiveQL
- Consultation de données multi-sources via HiveQL
- Accès aux données multi-sources dans HIVE depuis R

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Avant propos

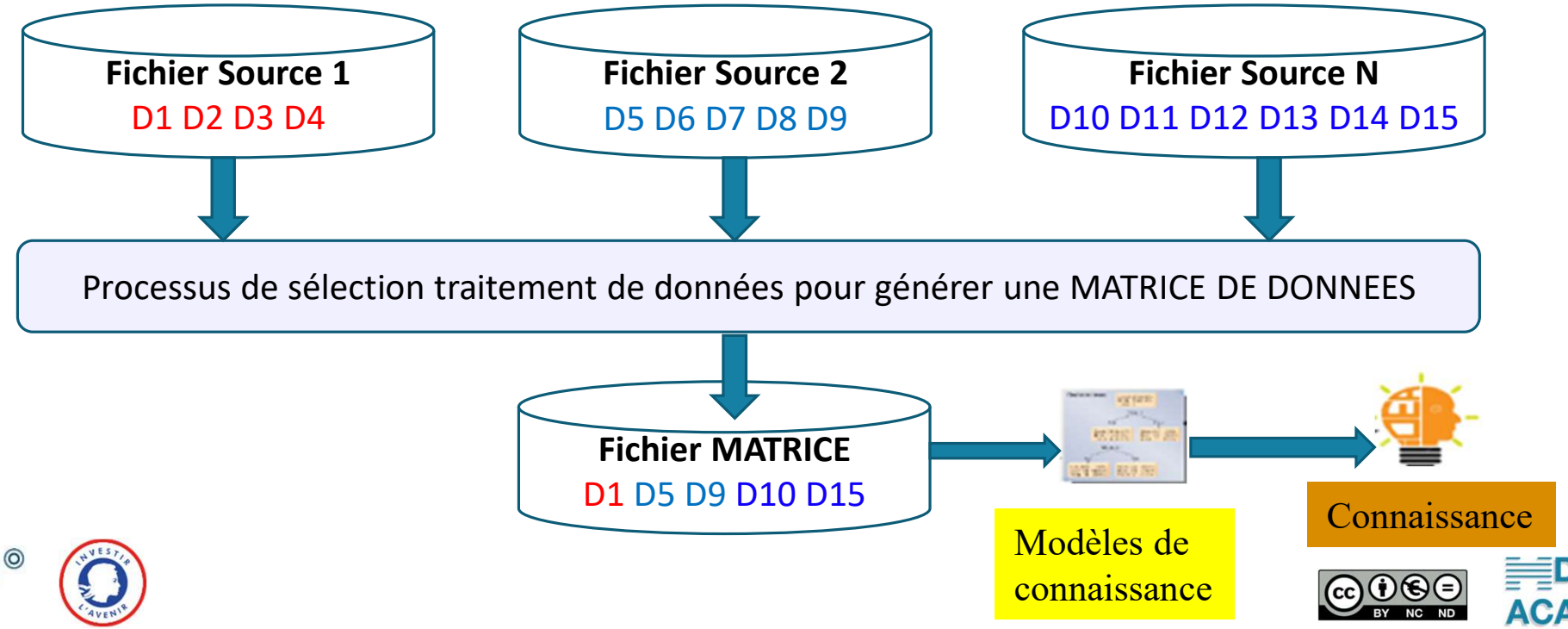
- Processus d'application interactif et itératif de Datamining



Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

Avant propos

- Processus d'application interactif et itératif de Datamining avec des fichiers comme source de données (exemple : Fichiers dans Hadoop HDFS)



Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

Avant propos

- **Processus d'application interactif et itératif de Datamining avec des fichiers comme source de données (exemple : Fichiers dans Hadoop HDFS)**

  - Dans la vie réelle des entreprises les données sont multi sources (**Fichiers**, Bases de données structurées, Bases de données semi-structurées , Bases de données non structurées)

  - Dans la vie réelle des entreprises les données sont hétérogènes (structurées, semi-structurées, **non-structurées**)

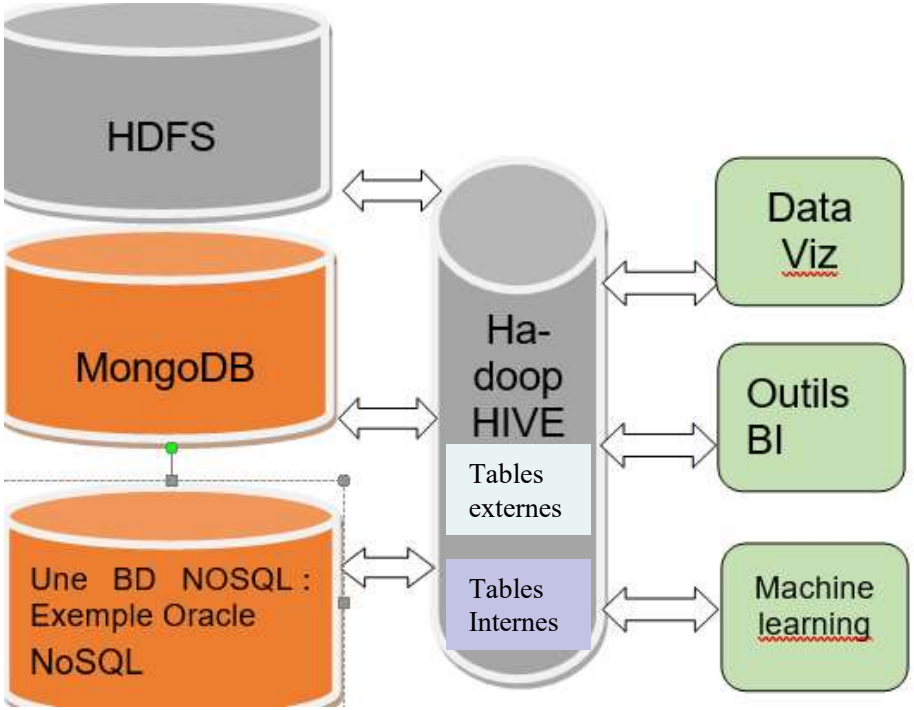
  - Une solution pourrait **être de transformer les données structurées et semi-structurées en fichiers**
    - **Problème 1** : Lourdeur lors des transformations
    - **Problème 2** : lourdeur lors de la manipulation des données
    - **Problème 3** : Impossibilité de faire du temps réel

  - Afin d'éviter ces inconvénients **NOUS ALLONS PROPOSER DES ARCHITECTURES S'APPUIYANT SUR BIG DATA SQL**

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Architecture d'un lac de données virtuel avec Hadoop Hive

- Intégration Big Data avec HiveQL

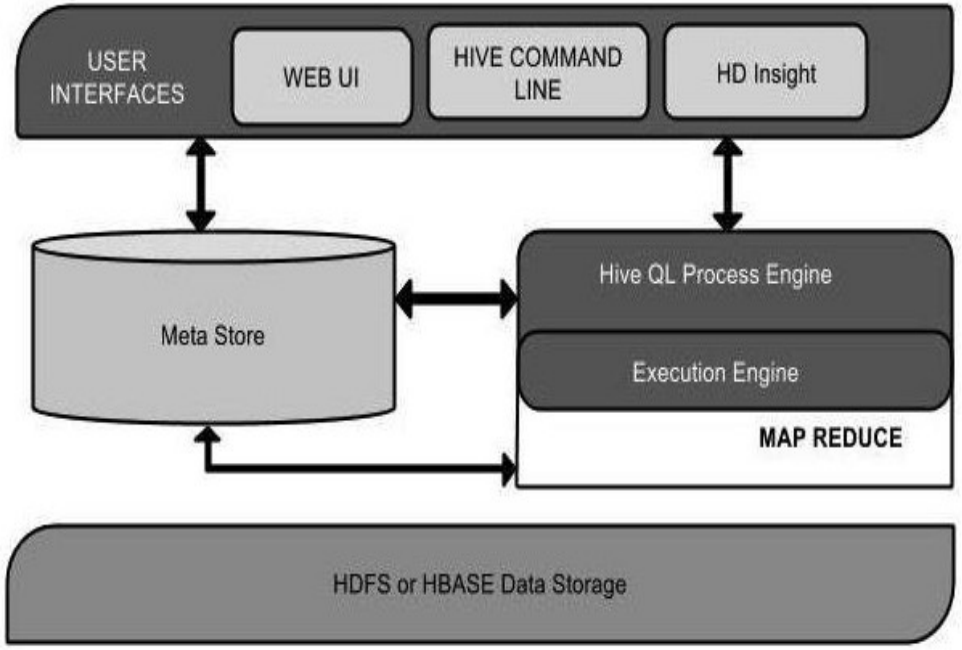


Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Architecture d'un lac de données virtuel avec Hadoop Hive

- **Architecture HIVE**

  https://www.tutorialspoint.com/hive/hive_quick_guide.ht



Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Architecture d'un lac de données virtuel avec Hadoop Hive

- **Architecture HIVE**

Unit Name	Operation
User Interface	Hive is a data warehouse infrastructure software that can create interaction between user and HDFS. The user interfaces that Hive supports are Hive Web UI, Hive command line, and Hive HD Insight (In Windows server).
Meta Store	Hive chooses respective database servers to store the schema or Metadata of tables, databases, columns in a table, their data types, and HDFS mapping.
HiveQL Process Engine	HiveQL is similar to SQL for querying on schema info on the Metastore. It is one of the replacements of traditional approach for MapReduce program. Instead of writing MapReduce program in Java, we can write a query for MapReduce job and process it.
Execution Engine	The conjunction part of HiveQL process Engine and MapReduce is Hive Execution Engine. Execution engine processes the query and generates results as same as MapReduce results. It uses the flavor of MapReduce.
HDFS or HBASE	Hadoop distributed file system or HBASE are the data storage techniques to store data into file system.

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Architecture d'un lac de données virtuel avec Hadoop Hive

▪ Les espaces de noms dans HIVE

- **Une base de données ici est d'abord un espace de noms**

```
CREATE DATABASE|SCHEMA [IF NOT EXISTS]
<database name>
```

Exemple :

Create database mynamespace;

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Architecture d'un lac de données virtuel avec Hadoop Hive

▪ Les types de données supportées dans HIVE

- **Column Types**
  - ✓ Integer Types (INT, …)
  - ✓ String Types (STRING, VARCHAR, CHAR)
  - ✓ Timestamp
  - ✓ Date
  - ✓ Decimals
  - ✓ Union Types
- **Complex Types**
  - ✓ Arrays
  - ✓ Maps
  - ✓ Struct

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Architecture d'un lac de données virtuel avec Hadoop Hive

▪ Les types de tables supportées dans HIVE

- **Les tables internes**

```
USE nomNameSpace;

CREATE TABLE [IF NOT EXISTS] [db_name.] table_name
[(col_name data_type [COMMENT col_comment], ...)] [COMMENT table_comment]
[ROW FORMAT row_format]
[STORED AS file_format]
```

- **Les tables Externes**

```
USE nomNameSpace;

CREATE [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.] table_name
[(col_name data_type [COMMENT col_comment], ...)] [COMMENT table_comment]
[ROW FORMAT row_format]
[STORED AS file_format]
```

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Architecture d'un lac de données virtuel avec Hadoop Hive

▪ Les types de requêtes supportées dans HIVE

- **Les requêtes CRUD**
  - ✓ Insert (Insert Into …select …)
  - ✓ Update
  - ✓ Delete
- **L'Ordre SQL Select avec les possibilités suivantes :**
  - ✓ La projection
  - ✓ La clause WHERE
  - ✓ La clause Group By
  - ✓ La clause Order by
  - ✓ La clause de jointure
    - Join
    - Outer Join
    - Left outer join
    - Rigth Outer Join
    - Full Outer Join

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

- **Architecture d'un lac de données virtuel avec Hadoop Hive**
  - **Tableau de correspondance des type ORACLE NOSQL – HIVE – ORACLE SQL**

Oracle NoSQL Database Type	Hive Type	RDBMS Type
FieldDef.Type.STRING	STRING	VARCHAR2(N)
	CHAR	
	VARCHAR	
FieldDef.Type.JSON	STRING	VARCHAR2(N)
FieldDef.Type.BOOLEAN	BOOLEAN	VARCHAR2(5)
FieldDef.Type.BINARY	BINARY	VARCHAR2(N)
FieldDef.Type.FIXED_BINARY	BINARY	VARCHAR2(N)
	TINYINT	
	SMALLINT	
FieldDef.Type.INTEGER	INT	NUMBER
FieldDef.Type.LONG	BIGINT	NUMBER
FieldDef.Type.FLOAT	FLOAT	NUMBER
FieldDef.Type.NUMBER	DECIMAL	NUMBER
FieldDef.Type.DOUBLE	DOUBLE	NUMBER
FieldDef.Type.ENUM	STRING	VARCHAR2(N)
FieldDef.Type.TIMESTAMP	java.sql.TIMESTAMP	TIMESTAMP
	DATE	
FieldDef.Type.ARRAY	ARRAY	VARCHAR2(N)
FieldDef.Type.MAP	MAP<STRING, data_type>	VARCHAR2(N)
FieldDef.Type.RECORD	STRUCT <col_name : data_type,...>	VARCHAR2(N)
	UNIONTYPE <data_type, data_type,...>	

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

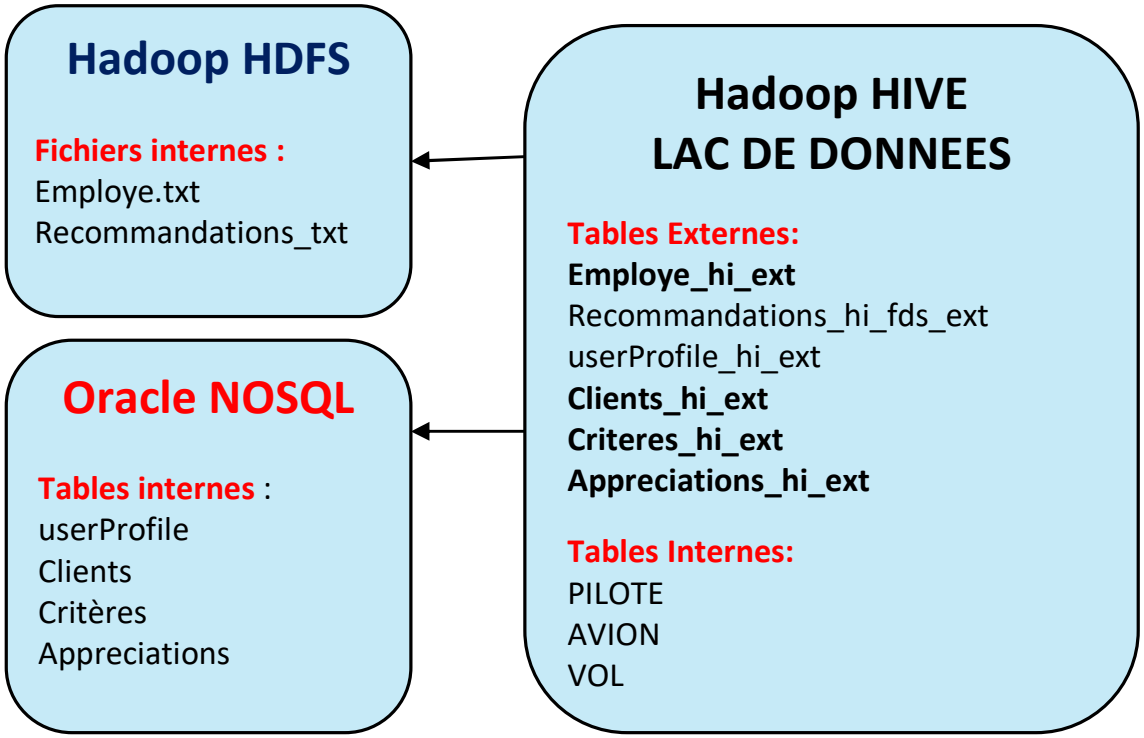
➢Accès aux données Oracle Nosql via HiveQL

- **Architecture**
  - **Mise en place par étapes**
    - **Etape 1** : création de tables sous oracle nosql
    - **Etape 2** : Création de tables externes HIVE pour une table oracle NoSQL

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données Oracle Nosql via HiveQL

- **Architecture**



Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Accès aux données Oracle Nosql via HiveQL

- **Mise en place par étape**
  - **Etape 1** : création de tables sous oracle nosql
  - **Etape 2** : Création de tables externes HIVE pour une table oracle NoSQL

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Accès aux données Oracle Nosql via HiveQL

- **Etape 1 : création de tables sous oracle nosql**

-- démarrer le serveur kvlite
java -Xmx256m -Xms256m -jar $KVHOME/lib/kvstore.jar kvlite

-- démarrer le client kvlite
java -jar $KVHOME/lib/kvstore.jar runadmin -port 5000 -host bigdatalite.localdomain

-- creation de la table
kv-> execute 'Create table userProfile (id integer, name string, address string, dnaiss string, zip_code string, primary key (id))'
Statement completed successfully

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données Oracle Nosql via HiveQL

▪ Etape 1 : création de tables sous oracle nosql

-- insertion de lignes
put table -name <name> [if-absent | -if-present] [-json <string>] [-file <file>] [-exact] [-update]

kv-> put table -name userProfile -json '{"id":1,"name":"joe","address":"12 rue du Congres à Valbonne", "dnaiss":"11/11/1960", "zip_code":"06560"}'
kv-> put table -name userProfile -json '{"id":2,"name":"jack","address":"11 rue du Begonias à Vallauris", "dnaiss":"13/11/1955", "zip_code":"06220"}'
kv-> put table -name userProfile -json '{"id":3,"name":"john","address":"13 Avenue de Marseille à Cassis", "dnaiss":"11/11/1962", "zip_code":"13260"}'
kv-> put table -name userProfile -json '{"id":4,"name":"Alumu","address":"60 Traverse des escaliers à Valbonne", "dnaiss":"11/08/1948", "zip_code":"06560"}'
kv-> put table -name userProfile -json '{"id":5,"name":"Malula","address":"61 Avenue du port à Vallauris", "dnaiss":"06/11/1957", "zip_code":"06220"}'
kv-> put table -name userProfile -json '{"id":6,"name":"Kasadi","address":"5 rue des pretres à Mougins", "dnaiss":"11/09/1980", "zip_code":"06250"}'

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données Oracle Nosql via HiveQL

- **Etape 1 : création de tables sous oracle nosql**

-- lecture d'une ligne insérée : derrière field doit apparaître la clé primaire
kv-> **get table -name** userProfile -field id -value 1

{"id":1,"name":"joe","address":"12 rue du Congres à Valbonne","dnaiss":"11/11/1960","zip_code":"06560"}

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données Oracle Nosql via HiveQL

- **Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL**

---- Se connecter à apache hive via beeline

[oracle@bigdatalite ~]$ beeline
Beeline version 1.1.0-cdh5.4.0 by Apache Hive

beeline> !connect jdbc:hive2://localhost:10000

scan complete in 7ms
Connecting to jdbc:hive2://localhost:10000
username:oracle password: welcome1

Enter username for jdbc:hive2://localhost:10000: oracle
Enter password for jdbc:hive2://localhost:10000: ********

Connected to: Apache Hive (version 1.1.0-cdh5.4.0)
Driver: Hive JDBC (version 1.1.0-cdh5.4.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://localhost:10000>

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données Oracle Nosql via HiveQL

- **Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL**

-- Création de la table externe à partir de HIVE : Syntaxe générale

CREATE EXTERNAL TABLE tablename colname coltype[, colname coltype,...]
STORED BY 'oracle.kv.hadoop.hive.table.TableStorageHandler'
TBLPROPERTIES (
"**oracle.kv.kvstore**" = "database",
"**oracle.kv.hosts**" = "nosql_node1:port[, nosql_node2:port...]",
"**oracle.kv.hadoop.hosts**" = "hadoop_node1[,hadoop_node2...]",
"**oracle.kv.tableName**" = "table_name");

-- Création effective de la table
0: jdbc:hive2://localhost:10000>

CREATE EXTERNAL TABLE userProfile_hive_ext (id int, name string, address string, dnaiss string, zip_code string)
STORED BY '**oracle.kv.hadoop.hive.table.TableStorageHandle**r'
TBLPROPERTIES (
"**oracle.kv.kvstore**" = "kvstore",
"**oracle.kv.hosts**" = "localhost:5000",
"**oracle.kv.hadoop.hosts**" = "localhost/127.0.0.1",
"**oracle.kv.tableName**" = "userProfile");
No rows affected (6.735 seconds)

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Accès aux données Oracle Nosql via HiveQL

▪ Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL

-- Consultation du catalogue HIVE

0: jdbc:hive2://localhost:10000> **show tables**;

```
+------------------------+--+
|       tab_name         |
+------------------------+--+
| cust                   |
| example_data           |
| example_data_1         |
| example_data_2         |
| example_data_3         |
| example_data_4         |
| movie                  |
...
| movielog               |
| session_stats          |
| user_movie             |
| userprofile_hive_ext   |
+------------------------+--+
18 rows selected (0.163 seconds)
```

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données Oracle Nosql via HiveQL

▪ Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL

-- Structure de la table externe userProfile_hive_ext

0: jdbc:hive2://localhost:10000> **describe** userProfile_hive_ext;

```
+-----------+------------+--------------------+--+
| col_name  | data_type  |      comment       |
+-----------+------------+--------------------+--+
| id        | int        | from deserializer  |
| name      | string     | from deserializer  |
| address   | string     | from deserializer  |
| dnaiss    | string     | from deserializer  |
| zip_code  | string     | from deserializer  |
+-----------+------------+--------------------+--+
```

5 rows selected (1.478 seconds)

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données Oracle Nosql via HiveQL

- **Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL**

--- Consultation des lignes de la table externe de la base nosql oracle
--- Les données sont dans la base nossql

0: jdbc:hive2://localhost:10000> **SELECT** * **FROM** userProfile_hive_ext;

id	name	address	dnaiss	zip_code
5	Malula	61 Avenue du port à Vallauris	06/11/1957	06220
1	joe	12 rue du Congres à Valbonne	11/11/1960	06560
3	john	13 Avenue de Marseille à Cassis	11/11/1962	13260
4	Alumu	60 Traverse des escaliers à Valbonne	11/08/1948	06560
2	jack	11 rue du Begonias à Vallauris	13/11/1955	06220
6	Kasadi	5 rue des pretres à Mougins	11/09/1980	06250

6 rows selected (5.259 seconds)

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Accès aux données HDFS via HiveQL

- **Mise en place par étapes**
  - **Etape 1 :** création fichiers HDFS
  - **Etape 2 :** Création de tables externes HIVE sur un fichier HDFS

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données HDFS via HiveQL

▪ Etape 1 : création fichiers HDFS

-- Création d'une directorie hadoop
hdfs dfs -mkdir /employe

oracle@bigdatalite gmDiversTests]$ **hdfs dfs -mkdir /employe**
[oracle@bigdatalite gmDiversTests]$

-- contenu du fichier employe.txt
KING | PARIS | 11/11/1989
KONG | YONGKONG | 11/11/1995
BLECK | BRISTOI | 12/12/2000

-- ajout d'un fichier dans hdfs
hdfs dfs -put employe.txt /employe

-- **vérification de l'ajout.**
[oracle@bigdatalite gmDiversTests]$ **hdfs dfs -ls /employe**
Found 1 items
-rw-r--r-- 1 oracle supergroup 74 2015-12-27 06:19 /employe/employe.txt

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données HDFS via HiveQL

- **Etape 2 : Création de tables externes HIVE sur un fichier HDFS**

-- Accès indirecte via hive : Création de la table externe emp_hive_hadoop_ext dans HIVE puis dans la base DWH Oracle 12c
-- La table externe 12c doit pointer vers la table externe HIVE.
-- La table externe HIVE pointe sur le fichier texte employe.txt de hadoop hdfs

```
0: jdbc:hive2://localhost:10000> CREATE EXTERNAL TABLE emp_hive_hadoop_ext (
name string,
address string,
dnaiss  string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION 'hdfs:/employe';
No rows affected (1.1 seconds)

0: jdbc:hive2://localhost:10000> SELECT * FROM emp_hive_hadoop_ext ;
+--------------------------+-----------------------------+----------------------+
| name                     | address                     | dnaiss               |
+--------------------------+-----------------------------+----------------------+
| KING                     | PARIS                       | 11/11/1989           |
| KONG                     | YONGKONG                    | 11/11/1995           |
| BLECK                    | BRISTOl                     | 12/12/2000           |
+--------------------------+-----------------------------+----------------------+
4 rows selected (4.513 seconds)
```

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

Consultation de données multi-sources via HiveQL

---- Se connecter à apache hive via beeline

```
[oracle@bigdatalite ~]$ beeline
Beeline version 1.1.0-cdh5.4.0 by Apache Hive

beeline>  !connect jdbc:hive2://localhost:10000

scan complete in 7ms
Connecting to jdbc:hive2://localhost:10000
# username:oracle password: welcome1

Enter username for jdbc:hive2://localhost:10000: oracle
Enter password for jdbc:hive2://localhost:10000: ********

Connected to: Apache Hive (version 1.1.0-cdh5.4.0)
Driver: Hive JDBC (version 1.1.0-cdh5.4.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://localhost:10000>
```

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Consultation de données multi-sources via HiveQL

```
select name, address, dnaiss
from userProfile_hive_ext
Union all
select name, address, dnaiss
from emp_hive_hadoop_ext;
```

_u1.name	_u1.address	_u1.dnaiss
KING	PARIS	11/11/1989
KONG	YONGKONG	11/11/1995
BLECK	BRISTOl	12/12/2000
joe	12 rue du Congres à Valbonne	11/11/1960
Alumu	60 Traverse des escaliers à Valbonne	11/08/1948
Sergio	5 rue de Moscou à Nice	11/09/1980
jack	11 rue du Begonias à Vallauris	13/11/1955
Kasadi	5 rue des pretres à Mougins	11/09/1980
john	13 Avenue de Marseille à Cassis	11/11/1962
Malula	61 Avenue du port à Vallauris	06/11/1957

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Accès aux données multi-sources dans HIVE depuis R

- Vous avez **choisi HIVE comme FRONTAL BIG DATA SQL de votre lac** de données
- Vous **avez construit des matrices grâce à des requêtes BIG DATA SQL** sur le Frontal
- Vous avez **choisi R comme outil d'Analyse** et vous souhaitez utiliser les matrices obtenues avec BIG DATA SQL (HiveQL)
- **Comment faire** ?

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Accès aux données multi-sources dans HIVE depuis R

- **Prérequis**
  - **R doit être installé**. Suivre les procédures selon votre OS
  - **Hive et Hadoop doivent avoir été installés.** . Voir les exemples de créations de tables externes

  - **Lancer HIVE** si ce n'est déjà fait. On obtient le prompt

  - **Lancer R** si ce n'est déjà fait

R

>

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢Accès aux données multi-sources dans HIVE depuis R

```
# chargement des librairies
> library(RJDBC)

# Chargement de la librairie JDBC et paramètres

> hive_jdbc_jar <- "/usr/lib/hive/lib/hive-jdbc-1.1.0-cdh5.13.1-standalone.jar"
> hive_driver <- "org.apache.hive.jdbc.HiveDriver"          # Driver hive jdbc
> hive_url <- "jdbc:hive2://localhost:10000 "               # Driver localisation de la base HIVE
> drv <- JDBC(hive_driver, hive_jdbc_jar)                   # Chargement du driver JDBC

# Connexion à HIVE
> conn <- dbConnect(drv, hive_url, "oracle", "")
```

Module M4.6, section 3 : Construction de lacs de données virtuel et/ou physique avec Hadoop HiveQL

➢ Accès aux données multi-sources dans HIVE depuis R

▪ Interrogation de la base de données HIVE depuis R

 # Exécution d''une requête SQL multi-sources

> rs <- **dbSendQuery**(**conn**, "SELECT name, address, dnaiss FROM **userProfile_hive_ext** UNION ALL SELECT name, address, dnaiss FROM **emp_hive_hadoop_ext**");

 # Affichage du résultat

> **fetch(rs);**

	NAME	ADDRESS	DNAISS
1	Kasadi	5 rue des pretres a Mougins	11/09/1980
2	joe	12 rue du Congres a Valbonne	11/11/1960
3	jack	11 rue du Begonias a Vallauris	13/11/1955
4	Alumu	60 Traverse des escaliers a Valbonne	11/08/1948
5	john	13 Avenue de Marseille a Cassis	11/11/1962
6	Malula	61 Avenue du port a Vallauris	06/11/1957
7	KING	PARIS	11/11/1989
8	KONG	YONGKONG	11/11/1995
9	BLECK	BRISTOl	12/12/2000

Module M4.6, section 3 : QUIZ

- **Question 1 : Quel est le frontal Big data SQL dans l'architecture Big Data mise en œuvre dans cette section**
  - A: Sql server de Microsoft
  - B: SGBD SQL Oracle
  - C: DB2 d'IBM
  - D: Hadoop HIVE
  - E: Hadoop Hawq
  - F: Hadoop Impala

- **Question 2** : **Les données du frontal Big data SQL dans l'architecture Big Data mise en œuvre dans cette section sont physiquement dans**
  - A: HDFS
  - B: MongoDB
  - C: La base de données frontal Big data Sql
  - D: Oracle NOSQL
  - E: HDFS, MongoDB, La base de données frontal Big data Sql, Oracle NOSQL
  - F: Oracle SQL

Module M4.6, section 3 : QUIZ

➢ **Question 3** : **A partir du frontal Big data SQL dans l'architecture Big Data mise en œuvre dans cette section, cochez les requêtes SQL possibles**

- A: Il est possible d'effectuer des mises à jour distribuées depuis le frontal du lac
- B: Il est possible d'effectuer consultations multi-sources depuis le frontal
- C: Il est possible de créer des tables n'importe ou depuis le frontal
- D: Il est possible d'effectuer des jointures multi sources depuis le frontal

➢ **Question 4** : **A partir du frontal Big data SQL dans l'architecture Big Data mise en œuvre dans cette section, il est possible de brancher des outils d'analyse tel que R. Cochez ce qui est juste**

- A: Il est possible d'effectuer depuis R des mises à jour distribuées sur le frontal du lac
- B: Il est possible d'effectuer depuis R des consultations multi-sources sur le frontal
- C: Il est possible de créer des tables depuis R sur le frontal dans les bases sources
- D: Il est possible d'effectuer depuis R des jointures sur le frontal
- E: R doit avoir été installé et configuré correctement

UNIVERSITÉ CÔTE D'AZUR ESTIA INSTITUTE OF TECHNOLOGY

DATUM ACADEMY

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

FUN FRANCE UNIVERSITÉ NUMÉRIQUE

anr

INVESTIR L'AVENIR

MIAGE RÉSEAU DES MIAGE DE FRANCE Nice

MBDS

BIHAR

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

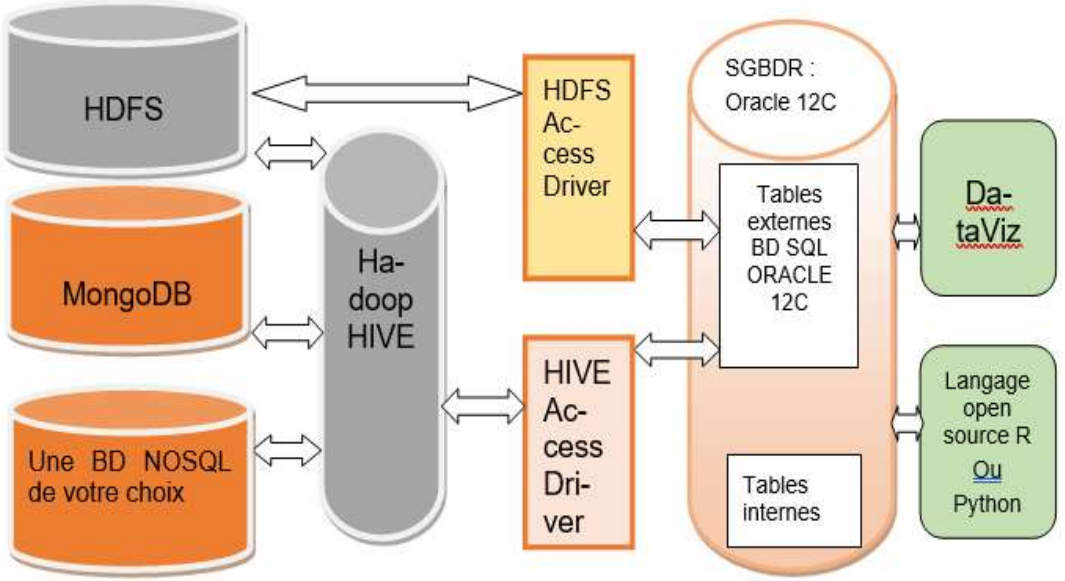
Plan

- Architecture d'un lac de données virtuel et / ou physique avec Oracle
- Accès aux données Oracle Nosql via Big data SQL
- Accès aux données HDFS via Big data SQL
- Consultation de données multi-sources via Oracle Big data SQL
- Accès Oracle depuis R

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

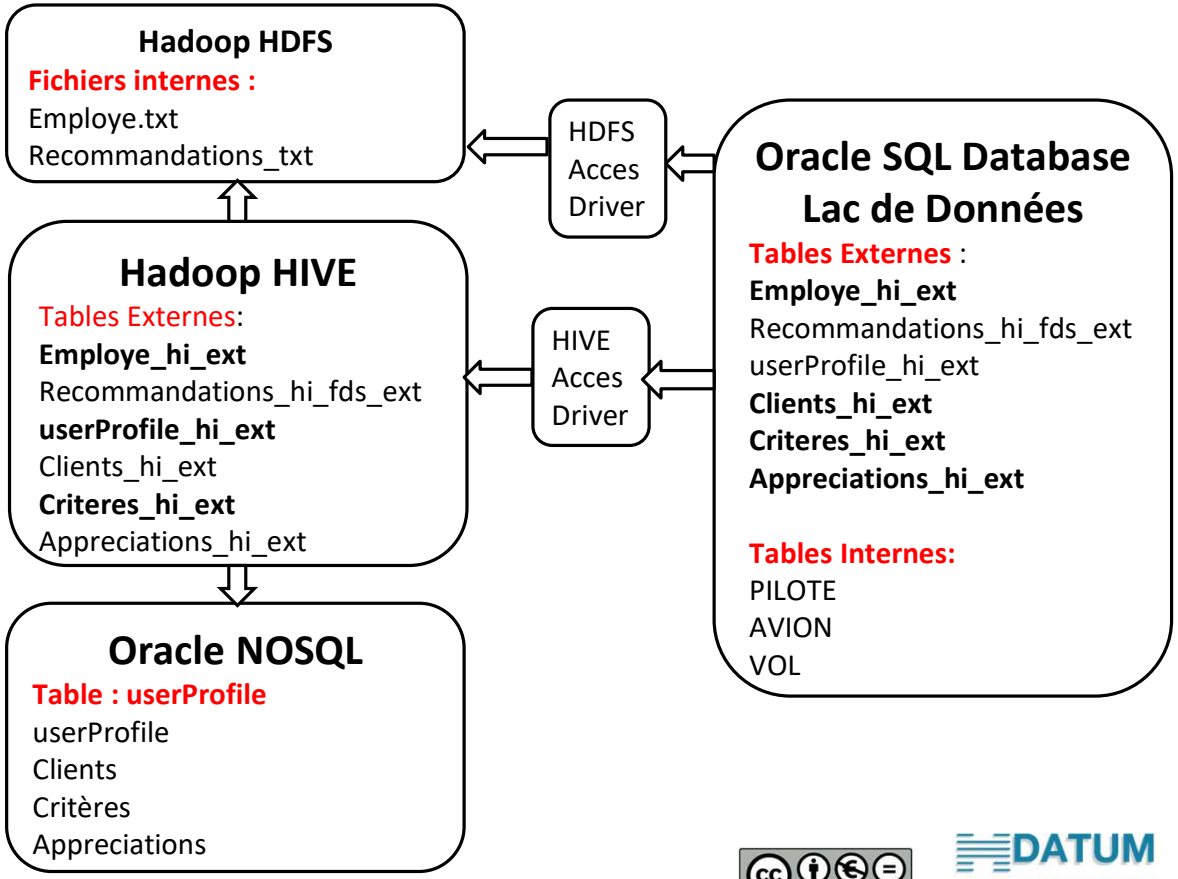
➢ Architecture d'un lac de données virtuel et/ou physique avec Oracle

- Intégration Big avec Oracle Big Data SQL
- Architecture adaptée pour les clients Oracle ayant déjà un entrepôt de données



Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

- **Architecture d'un lac de données virtuel et/ou physique avec Oracle**
  - **Un exemple**



Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢ Accès aux données Oracle Nosql via Oracle Big data SQL

▪ Mise en place par étapes

- **Etape 1** : création de tables sous oracle nosql
- **Etape 2** : Création de tables externes HIVE pour une table oracle NoSQL
- **Etape 3** : Création de tables externes Oracle Sql pour une table externe Hive
- **Etape 4** : Consultation de tables externes Oracle SQL

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

- **Etape 1 : création de tables sous oracle nosql**

-- Démarrer le serveur kvlite si ce n'est déjà fait
java -Xmx256m -Xms256m -jar $KVHOME/lib/kvstore.jar kvlite

-- Démarrer le client kvlite
java -jar $KVHOME/lib/kvstore.jar runadmin -port 5000 -host bigdatalite.localdomain

-- Creation de la table
kv-> **execute 'Create table userProfile** (id integer, name string,
address string, dnaiss string, zip_code string, primary key (id))'
Statement completed successfully

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢ Accès aux données Oracle Nosql via Big data SQL

▪ Etape 1 : création de tables sous oracle nosql

-- insertion de lignes
put table -name <name> [if-absent | -if-present]
[-json <string>] [-file <file>] [-exact] [-update]

kv-> execute 'Create table **userProfile** (id integer, name string, address string, dnaiss string, zip_code string, primary key (id))'

kv-**> put table -name userProfile** -json '{"id":1,"name":"joe","address":"12 rue du Congres à Valbonne", "dnaiss":"11/11/1960", "zip_code":"06560"}'
kv-> put table -name userProfile -json '{"id":2,"name":"jack","address":"11 rue du Begonias à Vallauris", "dnaiss":"13/11/1955", "zip_code":"06220"}'
kv-**> put table -name userProfile** -json '{"id":3,"name":"john","address":"13 Avenue de Marseille à Cassis", "dnaiss":"11/11/1962", "zip_code":"13260"}'
kv-> put table -name userProfile -json '{"id":4,"name":"Alumu","address": "60 Traverse des escaliers à Valbonne", "dnaiss":"11/08/1948", "zip_code":"06560"}'
kv-> **put table -name userProfile** -json '{"id":5,"name":"Malula","address":"61 Avenue du port à Vallauris", "dnaiss":"06/11/1957", "zip_code":"06220"}'
kv-> put table -name userProfile -json '{"id":6,"name":"Kasadi","address":"5 rue des pretres à Mougins", "dnaiss":"11/09/1980", "zip_code":"06250"}'

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

- **Etape 1 : Création de tables sous oracle nosql**

-- lecture d'une ligne insérée : derrière field doit apparaître la clé primaire
kv-> get table -name **userProfile** -field id -value 1

{"id":1,"name":"joe","address":"12 rue du Congres à Valbonne","dnaiss":"11/11/1960","zip_code":"06560"}

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

▪ Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL

---- Se connecter à apache hive via beeline

[oracle@bigdatalite ~]$ **beeline**
Beeline version 1.1.0-cdh5.4.0 by Apache Hive

beeline**>** **!connect jdbc:hive2://localhost:10000**

scan complete in 7ms
Connecting to jdbc:hive2://localhost:10000
username:oracle password: welcome1

Enter username for jdbc:hive2://localhost:10000: **oracle**
Enter password for jdbc:hive2://localhost:10000: ********

Connected to: Apache Hive (version 1.1.0-cdh5.4.0)
Driver: Hive JDBC (version 1.1.0-cdh5.4.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://localhost:10000>

Accès aux données Oracle Nosql via Big data SQL

- **Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL**

-- Création de la table externe à partir de HIVE: Syntaxe générale

```
CREATE EXTERNAL TABLE tablename colname coltype[, colname coltype,...]
STORED BY 'oracle.kv.hadoop.hive.table.TableStorageHandler'
TBLPROPERTIES ( "oracle.kv.kvstore" = "database",
  "oracle.kv.hosts" = "nosql_node1:port[, nosql_node2:port...]",
  "oracle.kv.hadoop.hosts" = "hadoop_node1[,hadoop_node3...]",
  "oracle.kv.tableName" = "table_name");
```

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

▪ Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL

-- Création effective de la table

0: jdbc:hive2://localhost:10000>
CREATE EXTERNAL TABLE userProfile_hive_ext (id int, name string, address string, dnaiss string, zip_code string)
STORED BY '**oracle.kv.hadoop.hive.table.TableStorageHandler**'
TBLPROPERTIES (
"**oracle.kv.kvstore**" = "kvstore",
"**oracle.kv.hosts**" = « localhost:5000",
"**oracle.kv.hadoop.hosts**" = "localhost/127.0.0.1",
"**oracle.kv.tableName**" = "userProfile");
No rows affected (6.735 seconds)

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

▪ Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL

-- Consultation du catalogue HIVE

0: jdbc:hive2://localhost:10000> **show tables**;

```
+-------------------------------+--+
|           tab_name            |
+-------------------------------+--+
| cust                          |
| example_data                  |
| example_data_1                |
| example_data_2                |
| example_data_3                |
| example_data_4                |
| …                             |
| user_movie                    |
| userprofile_hive_ext          |
+-------------------------------+--+
```

18 rows selected (0.163 seconds)

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

- **Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL**

-- Structure de la table externe userProfile_hive_ext

0: jdbc:hive2://localhost:10000> **describe userProfile_hive_ext;**

```
+-----------+-----------+-----------------------------------------+--+
| col_name  | data_type |                comment                  |
+-----------+-----------+-----------------------------------------+--+
| id        | int       | from deserializer                       |
| name      | string    | from deserializer                       |
| address   | string    | from deserializer                       |
| dnaiss    | string    | from deserializer                       |
| zip_code  | string    | from deserializer                       |
+-----------+-----------+-----------------------------------------+--+
```

5 rows selected (1.478 seconds)

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

▪ Etape 2 : Création de tables externes HIVE pour une table oracle NoSQL

--- Consultation des lignes de la table externe de la base nosql oracle. Les données sont dans la base nossql

0: jdbc:hive2://localhost:10000> **SELECT * FROM userProfile_hive_ext;**

id	name	address	dnaiss	zip_code
5	Malula	61 Avenue du port à Vallauris	06/11/1957	06220
1	joe	12 rue du Congres à Valbonne	11/11/1960	06560
3	john	13 Avenue de Marseille à Cassis	11/11/1962	13260
4	Alumu	60 Traverse des escaliers à Valbonne	11/08/1948	06560
2	jack	11 rue du Begonias à Vallauris	13/11/1955	06220
6	Kasadi	5 rue des pretres à Mougins	11/09/1980	06250

6 rows selected (5.259 seconds)

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

▪ Etape 3: Création de tables externes Oracle Sql pour une table externe Hive

```
-------------------------------------------------------------------------------------------
-- 3. Création de la table externe pour accéder depuis la base oracle CDH (DWH) au données de la table userProfile de la base oracle nosql 12c à
-- travers hadoop hive sachant que sur base HIVE une table externe userProfile_hive_ext permettant de voir les données nosql a déjà été crée
-------------------------------------------------------------------------------------------
-- Création de deux directory : ORACLE_BIGDATA_CONFIG et ORA_BIGDATA_CL_bigdatalite

-- Connexion à la base de données DWH/DATA LAKE Oracle
sql> connect pdbadmin/password
SQL> CREATE OR REPLACE DIRECTORY ORACLE_BIGDATA_CONFIG as '/u01/bigdatasql_config';
SQL> CREATE OR REPLACE DIRECTORY ORA_BIGDATA_CL_bigdatalite as '';

SQL> select DIRECTORY_NAME from dba_directories;
DIRECTORY_NAME
ORACLE_HOME
ORACLE_BASE
…
DATA_PUMP_DIR
…
ORACLE_BIGDATA_CONFIG
ORA_BIGDATA_CL_bigdatalite
12 rows selected.
```

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

▪ Etape 3: Création de tables externes Oracle Sql pour une table externe Hive

```
-------------------------------------------------------------------------------------------------------------------------------------------------------------------
-- Création de la table externe pour accéder depuis la base oracle (DWH / Dataware House) aux données de la table userProfile
-- de la base oracle à travers HADOOP HIVE sachant que sur la base HIVE une table externe  userProfile_hive_ext permettant
-- de voir les données nosql a déjà été crée.

-- Création de la table externe userProfile_hive_ext dans la base Oracle.
-- Cette table externe pointe sur la table externe HIVE.
-------------------------------------------------------------------------------------------------------------------------------------------------------------------
drop table userProfile_hive_ext;

CREATE  TABLE userProfile_hive_ext (
id number(8),
name varchar2(40),
address varchar2(100),
dnaiss  varchar2(12),
zip_code varchar2(12))
ORGANIZATION EXTERNAL (TYPE ORACLE_HIVE  DEFAULT DIRECTORY   ORACLE_BIGDATA_CONFIG ACCESS PARAMETERS
(com.oracle.bigdata.tablename=default.userProfile_hive_ext )
) REJECT LIMIT UNLIMITED;
table créé
```

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données Oracle Nosql via Big data SQL

▪ Etape 4: Consultation de tables externes Oracle SQL

```
-------------------------------------------------------------------------------------------------------------------------------------------------------
-- Consultation des lignes de la table NoSQL Oracle USERPROFILE via  la table externe userProfile_hive_ext dans Oracle qui pointe
-- vers  la table userProfile_hive_ext dans HIVE. Cette dernière pointe vers la table physique USERPROFILE dans la base Oracle Nosql.
-------------------------------------------------------------------------------------------------------------------------------------------------------
col name format a12
col address format a37
col dnaiss format a12
col zip_code format a12

SQL> SELECT * FROM userProfile_hive_ext;

ID   NAME         ADDRESS                              DNAISS ZIP_CODE
5    Malula       61 Avenue du port a Vallauris        06/11/1957   06220
1     joe         12 rue du Congres a Valbonne         11/11/1960   06560
3     john        13 Avenue de Marseille a Cassis      11/11/1962    13260
4     Alumu       60 Traverse des escaliers a Valbonne 11/08/1948   06560
2     jack        11 rue du Begonias a Vallauris       13/11/1955   06220
6    Kasadi       5 rue des pretres a Mougins          11/09/1980   06250
6 rows selected.
```

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢ Accès aux données HDFS via Big data SQL

- **Mise en place par étapes**
  - ✓ **Etape 1 : création fichiers HDFS**
  - ✓ **Etape 2** : Création de tables externes HIVE sur un fichier HDFS
  - ✓ **Etape 3** : Création de tables externes Oracle Sql pour une table externe Hive
  - ✓ **Etape 4** : Consultation de tables externes Oracle SQL

  - ✓ **Accès direct** aux fichiers HDFS depuis Oracl SQL hors étapes 2 et 3

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données HDFS via Big data SQL

▪ Etape 1 : création fichiers HDFS

-- Création d'une directorie hadoop
hdfs dfs -mkdir /employe

oracle@bigdatalite gmDiversTests]$ **hdfs dfs -mkdir /employe**
[oracle@bigdatalite gmDiversTests]$

-- Contenu du fichier employe.txt
KING | PARIS | 11/11/1989
KONG | YONGKONG | 11/11/1995
BLECK | BRISTOI | 12/12/2000

-- Ajout d'un fichier dans hdfs
hdfs dfs -put employe.txt /employe

-- Vérification de l'ajout.
[oracle@bigdatalite gmDiversTests**]$ hdfs dfs -ls /employe**
Found 1 items
-rw-r--r-- 1 oracle supergroup 74 2015-12-27 06:19 /employe/employe.txt

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢ Accès aux données HDFS via Big data SQL

▪ Etape 2 : Création de tables externes HIVE sur un fichier HDFS

```
-------------------------------------------------------------------------------------------------------------------------------------------------------------------
-- Accès indirecte via hive. Création de la table externe emp_hive_hadoop_ext dans HIVE puis dans la base DWH Oracle 12c. La table externe
--  12c doit pointer vers la table externe HIVE. La table externe HIVE pointe sur le fichier texte employe.txt de hadoop hdfs
-------------------------------------------------------------------------------------------------------------------------------------------------------------------
0: jdbc:hive2://localhost:10000> CREATE EXTERNAL TABLE emp_hive_hadoop_ext (name string,  address string, dnaiss  string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE LOCATION 'hdfs:/employe';
No rows affected (1.1 seconds)

0: jdbc:hive2://localhost:10000> SELECT * FROM emp_hive_hadoop_ext ;
+-------------------------+-------------------------------+
|name         | address          | dnaiss           |
+-------------------------+-------------------------------+
| KING        |  PARIS           |  11/11/1989      |
| KONG        |  YONGKONG        |  11/11/1995      |
| BLECK       |  BRISTOl         |  12/12/2000      |
+-------------------------+-------------------------------+
4 rows selected (4.513 seconds)
```

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données HDFS via Big data SQL

▪ Etape 3: Création de tables externes Oracle Sql pour une table externe Hive

-- Création de la table externe **emp_hive_hadoop_ext** dans la base DWH/Data lake Oracle. Cette table externe pointe sur la table externe
-- HIVE de même nom. qui elle même pointe sur **le fichier texte employe.txt** de hadoop hdfs

sql> connect system/welcome1

```
DROP TABLE  emp_hive_hadoop_ext;

CREATE  TABLE emp_hive_hadoop_ext (name varchar2(40), address varchar2(100), dnaiss  varchar2(12))
ORGANIZATION EXTERNAL (
TYPE ORACLE_HIVE
DEFAULT DIRECTORY   ORACLE_BIGDATA_CONFIG
ACCESS PARAMETERS (
com.oracle.bigdata.tablename=default.emp_hive_hadoop_ext
)
)
REJECT LIMIT UNLIMITED;
table créé.
```

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données HDFS via Big data SQL

▪ Etape 4: Consultation de tables externes Oracle SQL

-- Consultation des lignes du **fichier hadoop employe.txt** via la table externe **emp_hive_hadoop_ext** dans Oracle qui pointe vers
-- la table externe **emp_hive_hadoop_ext** dans **HIVE**. Cette dernière pointe vers le fichier texte **hadoop employe.txt.**

```
col name format a12
col address format a37
col dnaiss format a12

SELECT * FROM emp_hive_hadoop_ext;
```

NAME	ADDRESS	DNAISS
KING	PARIS	11/11/1989
KONG	YONGKONG	11/11/1995
BLECK	BRISTOl	12/12/2000

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

Accès aux données HDFS via Big data SQL

Accès direct aux fichiers HDFS depuis Oracle SQL hors étapes 2 et 3

```
-- Création de la table externe employe_hadoop_ext dans la base DWH. Oracle 12c. Cette table externe pointe sur le employe.txt
-- dans hadoop  hdfs. Les directories : ORACLE_BIGDATA_CONFIG et ORA_BIGDATA_CL_bigdatalite sont supposée avoir été créées

sql> connect system/welcome1

drop table employe_hadoop_ext;
CREATE  TABLE employe_hadoop_ext (name varchar2(40), address varchar2(100), dnaiss  varchar2(12))
ORGANIZATION EXTERNAL (
TYPE ORACLE_HDFS
DEFAULT DIRECTORY   ORACLE_BIGDATA_CONFIG
ACCESS PARAMETERS (
com.oracle.bigdata.fileformat:TEXTFILE
com.oracle.bigdata.overflow:{"action":"truncate"}
com.oracle.bigdata.erroropt:{"action":"setnull"}
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|') LOCATION ('hdfs:/employe/employe.txt') );
```

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Accès aux données HDFS via Big data SQL

- **Accès direct aux fichiers HDFS depuis Oracle SQL hors étapes 2 et 3**

```
-- Consultation des lignes du fichier hadoop employe.txt via la table externe employe_hadoop_ext dans Oracle 12c. Cette dernière
-- pointe directement vers le fichier texte hadoop hdfs employe.txt
set linesize 200
col name format a50
col address format a40
col dnaiss format a12

SELECT * FROM employe_hadoop_ext;
```

NAME	ADDRESS	DNAISS
KING |	PARIS |	11/11/1989
KONG |	YONGKONG |	11/11/1995
BLECK |	BRISTOl |	12/12/2000

Attention : Les délimiteurs ne sont pas reconnus. Bug ?
Un traitement spécifique peut être fait sur la base SQL pour reconstituer les colonnes.

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢Consultation de données multi-sources via Oracle Big Data SQL

- **Consultation des données nosql oracle et hdfs depuis la base Oracle SQL via les tables externes**

Sql> connect pdbadmin/password

Col name format A20
Col address format A40
Col dnaiss format A12
SELECT name, address, dnaiss
FROM userProfile_hive_ext
UNION ALL
SELECT name, address, dnaiss
FROM emp_hive_hadoop_ext;

```
NAME                 ADDRESS                                  DNAISS
-------------------- ---------------------------------------- ------------
Kasadi               5 rue des pretres a Mougins              11/09/1980
joe                  12 rue du Congres a Valbonne             11/11/1960
jack                 11 rue du Begonias a Vallauris           13/11/1955
Alumu                60 Traverse des escaliers a Valbonne     11/08/1948
john                 13 Avenue de Marseille a Cassis          11/11/1962
Malula               61 Avenue du port a Vallauris            06/11/1957
KING                  PARIS                                    11/11/1989
KONG                  YONGKONG                                 11/11/1995
BLECK                 BRISTOl                                  12/12/2000

9 rows selected.
```

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

ESTIA INSTITUTE OF TECHNOLOGY

UNIVERSITÉ CÔTE D'AZUR

➢ Accès à ORACLE depuis R

- Vous avez **choisi ORACLE comme FRONTAL de votre lac** de données
- Vous **avez construits des matrices grâce à des requêtes BIG DATA SQL** sur le Frontal
- Vous avez **choisi R comme outil d'Analyse** et vous souhaitez utiliser les matrices obtenues avec BIG DATA SQL (Oracle Big Data SQL)
- **Comment faire** ?

Accès à ORACLE depuis R

Prérequis

- **R doit être installé**. Il faut s'appuyer la version compatible avec le Driver spécifique Oracle. Exemple

https://cran.r-project.org/bin/windows/base/old/3.6.0/

Lancer R. Vous obtenez le prompt : >

- **Le driver R spécifique du SGBD cible** doit être installé. Ici ROracle

http://www.oracle.com/technetwork/database/database-technologies/r/roracle/downloads/index.html

> **setwd**('D:/1agm05092005/5Logiciels_new/R/R_Oracle_Zip')

> **install.packages**('ROracle_1.3-2.zip', repos = NULL) # Le zip doit être dans le dossier cité

Accès à ORACLE depuis R

Prérequis

- **Le driver R générique DBI (DataBase Inteface)** des SGBD doit être installé.
- Télécharger le package DBI ici : https://cran.biotools.fr/src/contrib/DBI_1.1.3.tar.gz

```
> setwd('D:/1agm05092005/5Logiciels_new/R/DBIGZ')   # placer le .gz dans ce dossier
> install.packages("DBI_1.1.3.tar.gz");
```

Module M4.6, section 4 : Construction de lacs de données virtuel et/ou physique avec Big Data SQL Oracle

➢ Accès à ORACLE depuis R

- **Connexion à Oracle et prise en main**

 # 3.1 chargement de la librairie

```
> library('ROracle')
```

 # 3.2 chargement du driver

```
> drv <- dbDriver("Oracle")
```

 # 3.3 Nom de la chaine de connexion à la base

```
> ORCL <-"(DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST = 134.59.152.111)(PORT =
8822))(CONNECT_DATA =(SERVER = DEDICATED)(SERVICE_NAME = orcl)))"
```

 # 3.4 Connexion à la base

```
> con <- dbConnect(drv, username = "MOPOLO", password = "Password", dbname=ORCL)
```

➢ Accès à ORACLE depuis R

- **Interrogation de la base de données Oracle depuis R**

 # Exécution d''une requête SQL multi-sources

> rs <- **dbSendQuery**(**con**, "SELECT name, address, dnaiss FROM **userProfile_hive_ext** UNION ALL SELECT name, address, dnaiss FROM **emp_hive_hadoop_ext**");

 # Affichage du résultat

> **fetch(rs);**

	NAME	ADDRESS	DNAISS
1	Kasadi	5 rue des pretres a Mougins	11/09/1980
2	joe	12 rue du Congres a Valbonne	11/11/1960
3	jack	11 rue du Begonias a Vallauris	13/11/1955
4	Alumu	60 Traverse des escaliers a Valbonne	11/08/1948
5	john	13 Avenue de Marseille a Cassis	11/11/1962
6	Malula	61 Avenue du port a Vallauris	06/11/1957
7	KING	PARIS	11/11/1989
8	KONG	YONGKONG	11/11/1995
9	BLECK	BRISTOl	12/12/2000

Module M4.6, section 4 : QUIZ

- **Question 1 : Quel est le frontal Big data SQL dans l'architecture Big Data mise en œuvre dans cette section**
  - A: Sql server de Microsoft
  - B: SGBD SQL Oracle
  - C: DB2 d'IBM
  - D: Hadoop HIVE
  - E: Hadoop Hawq
  - F: Hadoop Impala

- **Question 2** : **Les données du frontal Big data SQL dans l'architecture Big Data mise en œuvre dans cette section sont physiquement en totalité dans**
  - A: HDFS
  - B: MongoDB
  - C: La base de données frontal Big data Sql
  - D: Oracle NOSQL
  - E: HDFS, MongoDB, La base de données frontal Big data Sql, Oracle NOSQL, Hive
  - F: Oracle SQL

Module M4.6, section 4 : QUIZ

➢ **Question 3** : **A partir du frontal Big data SQL dans l'architecture Big Data mise en œuvre dans cette section, cochez les requêtes SQL possibles**

- A: Il est possible d'effectuer des mises à jour distribuées depuis le frontal du lac
- B: Il est possible d'effectuer consultations multi-sources depuis le frontal
- C: Il est de créer des tables n'importe ou depuis le frontal
- D: Il est possible d'effectuer des jointures multi sources depuis le frontal

➢ **Question 4** : **A partir du frontal Big data SQL dans l'architecture Big Data mise en œuvre dans cette section, il est possible de brancher des outils d'analyse tel que R. Cochez ce qui est juste**

- A: Il est possible d'effectuer depuis R des mises à jour distribuées sur le frontal du lac
- B: Il est possible d'effectuer depuis R des consultations multi-sources sur le frontal
- C: Il est de créer des tables depuis R sur le frontal
- D: Il est possible d'effectuer depuis R des jointures sur le frontal
- E: R doit avoir été installé et configuré correctement

Module M4.6 : Bilan et Exercices

- Bilan
- Exercices

Module M4.6 : Références Web

- [R1] “Exploring CouchDB: A document-oriented database for Web applications”, Joe Lennon, Software developer, Core International. http://www.ibm.com/developerworks/opensource/library/os-couchdb/index.html
- [R2] “Graph Databases, NOSQL and Neo4j” Posted by Peter Neubauer on May 12, 2010 at: http://www.infoq.com/articles/graph-nosql-neo4j
- [R3] “Cassandra vs MongoDB vs CouchDB vs Redis vs Riak vs HBase comparison”, Kristóf Kovács. http://kkovacs.eu/cassandra-vs-mongodb-vs-couchdb-vs-redis
- [R4] “Distinguishing Two Major Types of Column-Stores” Posted by Daniel Abadi onMarch 29, 2010 http://dbmsmusings.blogspot.com/2010/03/distinguishing-two-major-types-of_29.html
- [R5] Bases de données : Big Data et NoSQL http://administration-systeme.blogspot.fr/2013/10/bases-de-donnees-big-data-et-nosql.html

Module M4.6 : Références Web

- [R6] “MapReduce: Simplified Data Processing on Large Clusters”, Jeffrey Dean and Sanjay Ghemawat, December 2004. https://static.googleusercontent.com/media/research.google.com/fr//archive/mapreduce-osdi04.pdf
- [R7] “Scalable SQL”, ACM Queue, Michael Rys, April 19, 2011 http://queue.acm.org/detail.cfm?id=1971597
- [R8] “a practical guide to noSQL”, Posted by Denise Miura on March 17, 2011 at http://blogs.marklogic.com/2011/03/17/a-practical-guide-to-nosql/
- [R9] Visual Guide to NoSQL Systems http://blog.nahurst.com/visual-guide-to-nosql-systems
- [R10] Infos sur Map Reduce. http://en.wikipedia.org/wiki/MapReduce
- [R11] WEBRANKINFO : site d’informations et de statistiques sur le WEB
  http://www.webrankinfo.com/dossiers/googles
  http://www.webrankinfo.com/dossiers/facebook
- [R12] Echo Système HADOOP

http://hadoop.apache.org/#What+Is+Apache+Hadoop%3F

Module M4.6 : Références Web

- [R13] Hadoop Vive Tutorial
  https://www.tutorialspoint.com/hive/hive_create_table.htm

- [R14] Site web Hadoop Hive
  https://hive.apache.org/

- [R15] Wiki Apache Hive user documentation
  https://cwiki.apache.org/confluence/display/Hive/Home#Home-UserDocumentation

- [R16] HPL/SQL documentation (procédure stockées)
  https://cwiki.apache.org/confluence/pages/viewpage.action?pageId=59690156

- [R17] Structured vs Unstructured Data: 5 Key Differences, by Mark Smallcombe
  https://www.integrate.io/blog/structured-vs-unstructured-data-key-differences/

- [R18] Semistructured Data, Peter Buneman, Department of Computer and Information Science, University of Pennsylvania
  https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.117.5869&rep=rep1&type=pdf

Module M4.6 : **Bibliographie**

- [B1] HADOOP in practice, Alex HOLMES, Editions Manning Publications, 2012

- [B2] Principles of Big Data
  Preparing, Sharing, and Analyzing Complex Information,
  Par Jules J. Berman, Edition Morgan Kaufmann, 2013

- [B3] "LES BASES DE DNNÉES NOSQL: COMPRENDRE ET METTRE EN ŒUVRE"
  Edition Eyrolles 2013

- [B4] Building the Data Warehouse; W.H. INMON, Éditeur : John Wiley & Sons (19 septembre 2005), ISBN-10 : 6610279713, ISBN-13 : 978-6610279715