République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Larbi Tébessi –Tébessa-
Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie
Département : Mathématiques et informatique

MEMOIRE DE MASTER
Domaine: Informatique
Filière: Mathématiques/Informatique
Option: Réseaux et sécurité informatique

Thème:

L'intégration des Big Data dans le processus ETL (Extrat, Transform, Laod)

Présenté par:
BENMEHANIA Houria
BENSAKTA Latifa

Devant le jury:

M. Ridda LAOUAR	Professeur	Université Larbi Tebessi	Président
Louardi BRADJI	MCA	Université Larbi Tebessi	Rapporteur
Ahmed AHMIM	MAB	Université Larbi Tebessi	Examinateur

Date de soutenance: 29/05/2016

Note :……. Mention :……

ملخص

Big Data، حرفيا البيانات الضخمة هو مصطلح انجليزي يستخدم لوصف مجموعات البيانات التي تصبح كبيرة بحيث يصبح من الصعب العمل مع إدارة أدوات قواعد البيانات التقليدية. عندما نتكلم هناك عدد قليل من غيغا بايت (10^9 بايت)، والآن هناك صيانة بدلا تيرابايت (10^{12} بايت) من بيتابايت (10^{15} بايت) من إكسا بايت (10^{18} بايت) وحتى زيتابايت (10^{21} بايت).

في هذه الطلبات الجديدة للمعلومات من حيث الحجم، والتقاط وتخزين، والبحث، وتبادل وتحليل وتصور البيانات يجب إعادة تعريف. آفاق التعامل مع البيانات الضخمة هائلة، لاسيما بالنسبة الرأي السياسي أو تحليل الاتجاهات الصناعية، وعلم الجينوم، والكفاح ضد الأمن.

لقد عملنا على تصميم وسيلة لإيجاد حل لدمج الضخمة ، وقدمنا العمارة التي تقوم على عملية ETL التقليدية القائمة على مخطط XML في الاستخراج.

الكلمات المفتاحية

التكامل، البيانات الكبيرة، ETL، استخراج، تحويل، تحميل، XML.

Abstract

Big Data, literally massive data is an English term used to describe data sets that become so large that it becomes difficult to work with the management of conventional database tools. When we speak there are few gigabytes (10^9 bytes), there is now maintenance rather Terabyte (10^{12} bytes) of petabytes (10^{15} bytes) of exabytes (10^{18} bytes) and even zettabytes (10^{21} bytes).

In these new orders of magnitude, capture, storage, search, sharing, analysis and visualization of data must be redefined. Prospects dealing with massive data is enormous, particularly for political opinion or analysis of industrial trends, genomics, the fight against security.

We worked on the design of a method of solution for the integration of Big Data, and we presented an architecture that is based on a traditional ETL process based on the XML schema at the extraction.

Key words

Integration, Big Data, ETL, Extraction, Transformation, Loading, XML.

Résumé

Les Big Data, littéralement de données massives sont un terme anglais employé pour décrire les ensembles de données qui deviennent si grands qu'il devient difficile les travailler avec la gestion de bases de données conventionnelle d'outils. Quand nous parlons il y a peu de gigaoctets (10^9 octets), là est maintenant des Terabyte d'entretien plutôt (10^{12} octets), des petabytes (10^{15} octets) des exabytes (10^{18} octets) et même des zettabytes (10^{21} octets).

Dans ces nouveaux ordres de grandeur, la capture, le stockage, la recherche, partager, l'analyse et la visualisation des données doivent être redéfinie. Les perspectives traitant des données massives sont énormes, en particulier pour l'opinion ou l'analyse politique des tendances industriels, la génomique, le combat contre la sécurité.

Nous avons travaillé sur la conception d'une méthode de solution pour l'intégration de Big Data, ainsi, nous avons présenté une architecture qui repose sur un processus ETL traditionnel basé sur le schéma XML au niveau de l'extraction.

Mot clés

Intégration, Big Data, ETL, Extraction, Transformation, Chargement, langage XML.

Dédicace

Tout d'abord, je tiens à remercier Dieu tout puissant clément et Miséricordieux de m'avoir donné la force et la diligence providentielle d'avoir élaboré ce projet d'étude.

A mes chers parents

Je suis aujourd'hui ici grâce à votre patience et vos innombrables sacrifices. Qu'Allah, le tout puissant, vous préserve et vous procure la santé et la longue vie afin que je me puisse vous faire des prières salvatrices de reconnaissance et de remerciement.

A mes chères sœurs

Aucune dédicace ne saurait exprimer profondément ce que

Je rme sens envers vous, soient le grand merci et la grande affection que j'ai pour vous

A mes beaux frères et mes neveux Adem, Ahmed habib el rahemane, zakaria, abdel hafidhe et le petit Abdel djalil

A mes chères amies et mes collègues de promotion et de la faculté de science économie et sciences commercial et de gestion.

Et à tous ce qui m'ont enseigné moi au long de ma vie scolaire.

Sans oublier ma binôme et collègue « HOURIA »

BENSAKTA.L

Dédicace

Avant tous je remercie ALLAH

«الحمد و الشكر لله»

Je dédie cet humble travail qui est le fruit d'une pleine d'obstination et de persévérance :

- ***A mon cher père**, pour ses efforts, sa bienveillance et son sacrifice, qu'il n' jamais cessé de consentir pour ma réussite et mon bonheur.*
- ***A ma chère mère**, à qui je doit tout et sans, son soutien, son encouragement et surtout son amour, je n'aurais jamais pu être ce que je suis maintenant.*
- *A mes chères sœurs, mon frère Hamza, et mes amies Amel et Jihen les plus proches de mon cœur à qui je souhaite beaucoup de réussite dans la vie.*
- *A mes chères amies et cousine qui ont partagé la vie avec moi : Houda , Meriem, Mirette, Khawla, et ma chère tata Souad*
- *Tous ceux qui m'aiment et à tous ceux que j'aime.*
- *A toute ma famille d'Alger, Annaba et Tébessa.*
- *Sans oublier ma binôme et collègue « LATIFA »*

BENMHANIA.H

Table de matière

Chapitre 3

Liste de tableaux

Liste des figures

Liste d'abréviation

OLTP	online transaction processing
OLAP	On Line Analytical Processing
SQL	Structured Query Language
SGBD	*système de gestion de base de données*
IBM	International Business Machines Corporation
Rcfile	Record Columnar File
CSV	Comma-separated values
JSON	JavaScript Object Notation
SSIS	Ressources SQL Server Integration Services
CRM	customer relationship management
SOA	L'architecture orientée services
JDBC	Java Data base Connectivity
SGML	Standard Generalized Markup Language

Liste des Descriptions

ERP	**(Enterprise Ressource Planning)** tableau de bord décisionnel: On utilise parfois également le terme de *Executive Information System*, ou EIS, pour faire référence à un système décisionnel spécifiquement destiné à la Direction Générale, et donc présentant des informations stratégiques très consolidées.
DSA	(**Data Staging Area)** est une zone de stockage intermédiaire utilisé pour le traitement des données lors de l'extraction, de transformation et de chargement (ETL). La zone de transit de données se trouve entre la source de données et la cible de données, qui sont souvent des entrepôts de données
OLTP	En informatique et plus particulièrement dans le domaine des BD, le traitement transactionnel en ligne (en anglais *online transaction processing*, abrégé en *OLTP*) est un type d'application informatique qui sert à effectuer des modifications d'informations en temps réel.
OLAP	En informatique, et plus particulièrement dans le domaine des BD, le traitement analytique en ligne (anglais *On Line Analytical Processing*, OLAP) est un type d'application informatique orienté vers l'analyse sur-le-champ d'informations selon plusieurs axes, dans le but d'obtenir des rapports de synthèse
ODBC	Sigle d'*Open Database Connectivity:* est un intergicielle qui permet à une application informatique, par un procédé unique, de manipuler plusieurs bases de données
Snapshot	*Le snapshot enregistre les modifications apportées au volume logique cible*
Legacy	Logiciel que l'on veut garder en état de marche malgré le coût et la difficulté de trouver des pièces de rechange.
Timestamp	(Informatique) Horodatage
CDC	(**Changing Data Capture)** La main de la modification la saisie des données
SK	(**Surrogate Key)**Une clé de substitution dans BD est un identifiant unique pour soit une entité dans le monde modélisé ou un objet.
SCD	(**Slowly Changing Dimension)** En informatique décisionnelle, on parle d'une dimension à évolution lente (slowly changing dimension) lorsqu'une dimension peut subir des changements de description des membres.
SKP	(**Surrogate Key Pipeline)** Une clé de substitution (Surrogate key) est une clé non intelligente utilisée afin de substituer la clé naturelle (Business Key) qui provient des systèmes opérationnels. La clé naturelle est en général composée de plusieurs colonnes.
Open Linked	L'open data ou donnée ouverte est une donnée numérique d'origine publique ou privée.
Noyaux CPU	Un exo-noyau ou exokernel est un type de système d'exploitation dont l'architecture logicielle est dite modulaire.

La boucle for each	Est en informatique une structure de contrôle de programmation permettant de réaliser une boucle associée à une variable entière ou un pointeur qui sera incrémentée à chaque itération
emitIntermediate()	Sur l'instance MapReduce. Cette méthode stocke l'article dans la liste des articles avec pour label soit publié (published) ou non publié (unpublished).
Un workflow	Anglicisme pour flux de travaux, est la représentation d'une suite de tâches ou opérations effectuées par une personne, un groupe de personnes, un organisme, etc.
Rcfile	Est une structure de placement de données qui détermine la façon de stocker des tables relationnelles sur les clusters informatiques. Il est conçu pour les systèmes utilisant le cadre de MapReduce.
JSON	Est un format de données textuelles dérivé de la notation des objets du langage JavaScript. Il permet de représenter de l'information structurée comme le permet XML par exemple.
SerDe	Un sérialiseur / désérialiseur est une paire de blocs fonctionnels couramment utilisés dans les communications à haute vitesse pour compenser limitée entrée / sortie.
DDL	Un langage de définition de données ou la description des données de langue (DDL) est une syntaxe similaire à un langage de programmation informatique pour définir des structures de données, en particulier les schémas de base de données.
Le théorème CAP	Ou CDP, aussi connu sous le nom de théorème de Brewer dit qu'il est impossible sur un système informatique de calcul distribué de garantir en même temps (c'est à dire de manière synchrone) les trois contraintes Cohérence, Disponibilité, Tolérance au partitionnement
SAP Data Services	Offre une solution de classe entreprise unique pour l'intégration de données, la qualité des données, le profilage des données, et le traitement de données de texte qui vous permet d'intégrer, de transformer, améliorer, et de livrer des données fiables aux processus critiques de l'entreprise.
DataStage	IBM InfoSphere DataStage intègre des données sur de multiples systèmes en utilisant une structure parallèle à hautes performances.
Le CRM	Est un ensemble de systèmes permettant d'optimiser la relation qu'entretient la marque avec ses clients, afin de les fidéliser et d'augmenter le chiffre d'affaires de l'entreprise par client.
GUI	Outils de constructions d'interfaces graphiques
Serveur MPP	MPP (massivement de traitement parallèle) est le traitement coordonné d'un programme par plusieurs processeurs qui travaillent sur différentes parties du programme, à chaque processeur en utilisant son propre système d'exploitation et de la mémoire.
Wrapper	En génie logiciel, adaptateur (ou wrapper) est un patron de conception (design pattern) de type structure (structural). Il permet de convertir l'interface d'une classe en une autre interface que le client attend.

W3C	(Le World Wide Web Consortium) : est un organisme de standardisation à but non lucratif.
SGML	Standard Generalized Markup Language (« langage de balisage généralisé normalisé » -) est un langage de description à balises, de norme ISO (ISO 8879:1986).
Oracle9i	Oracle Data base est un système de gestion de base de données relationnelle (SGBDR).
FileMaker	Est un logiciel de gestion de bases de données développé par une filiale d'Apple appelée FileMaker Inc.
Matching	Cette phase crée des groupes correspondant. se déroule en deux étapes. Cette phase comprend les tables maîtres d'intégration (MI), où les doublons sont marqués.

Introduction générale

La croissance rapide des données distribuées sur l'Internet et aux entreprises qui sont arrivées depuis plusieurs sources (des capteurs utilisés pour collecter les informations climatiques ou bien des messages sur les médias sociaux facebook, gmail…, images numériques et vidéos publiées en ligne, enregistrements transactionnels d'achat en ligne Amazon, eBay…, des signaux GPS de téléphones mobiles…) la définition de ces données Massives indiquant que Big Data est un volume élevé, à grande vitesse, et / ou des actifs de haute information diverses qui produit beaucoup d'intérêts dans des systèmes d'intégration de donnée et qui nécessitent de nouvelles formes de traitement pour permettre une prise de décision améliorée, les organisations de tous les secteurs public et privé ont pris la décision stratégique de transformer les données en gros avantage concurrentiel.

Le défi consistant à extraire la valeur de Big Data est similaire à bien des égards au problème de la distillation Business Intelligence à partir de données transactionnelles séculaire. Au cœur de ce défi est le processus utilisé pour extraire des données provenant de sources multiples hétérogènes, le transformer pour répondre à vos besoins analytiques, et le charger dans un entrepôt de données pour une analyse ultérieure, un processus connu sous le nom "Extract, Transform & Load" (ETL). La nature des données volumineuses exige que l'infrastructure pour ce processus puisse évoluer de manière rentable.

Notre contribution, par la présente étude, consiste à proposer une architecture pour les entreprises désirant à intégrer des donnés massives structurés ou non structurés pour la fin d'analyse, dans le but d'augmenter leur potentiel concurrentiel.

L'architecture proposée est une architecture hybride qui réunit l'architecture traditionnelle (ETL, datawarehouse) basée sur le langage XML.

Notre mémoire est organisé en quatre chapitres, on a commencé par une introduction général, où on a défini l'informatique décisionnelle et sa relation avec le processus traditionnel de traitement de données ETL pour la prise de décision, on a décrit par la suite le phénomène de Big Data, ses technologies et ses domaines d'application, et au troisième chapitre on a parlé sur l'importance d'intégration des données massives hétérogènes, ses approches, l'approche d'intégration ETL, et les problèmes d'intégration des Big Data, enfin dans le dernier chapitre on a proposé une solution améliorée pour intégrer les données massives en parallèle et en temps réel a l'aide d'un processus ETL basé sur les schémas XML.

CHAPITRE I

Processus ETL (Extraction, Transformation, Chargement)

Chapitre1

1. Introduction

Toutes les entreprises accumulent de grandes quantités de données au travers de leur système (**OLTP, OLAP**…). Mais ces données ne deviennent des informations pertinentes pour les décideurs que lorsqu'elles permettent de répondre à des questions sur l'entreprise et son évolution. Pour être aidés dans leurs choix, les décideurs ont besoin de mesurer l'activité de l'entreprise à l'aide d'indicateurs de performance. En l'absence d'outils **d'informatique décisionnelle**, l'acquisition et l'analyse des données absorbent, parfois de façon stérile, le temps des décideurs. Or, le décideur de la nouvelle économie dispose d'un temps de lus en plus réduit.

Un environnement **BI** (Business Intelligence) inclus presque toujours une solution **ETL** (**Extract Transform Load**) pour extraire, transformer et charger les données source dans un entrepôt de données, et c'est ça ce que nous allons détailler dans ce premier chapitre.

2. Informatique décisionnelle

2.1. Définition

L'informatique décisionnelle ou bien la **Business Intelligence** désigne les moyens, les outils et les méthodes qui permettent de collecter, consolider, modéliser et restituer les données d'une entreprise en vue de fournir une aide à la décision aux managers. Le terme anglais est « **Business Intelligence** ». Une application de ce genre exécute la capture, l'analyse et le stockage de données provenant de plusieurs sources hétérogènes qui peuvent être des Enterprise Ressource Planning (ERP), des bases de données ou d'autres entrepôts de données. Traditionnellement, un **entrepôt de données** est utilisé comme source d'information par les décisionnaires. La Business Intelligence s'insère dans l'architecture du système d'information d'une entreprise.

Pour pouvoir obtenir une vision synthétique de l'objet de l'analyse avancée, une application de Business Intelligence utilise un entrepôt de données comme source d'informations. **L'entrepôt de données** a comme principale mission de filtrer, croiser et reclasser les informations grâce aux outils **ETL (Ectract Transform Load)**, afin de mettre à disposition des générateurs d'analyse des données de niveau global. C'est-à-dire, des données pertinentes directement en rapport avec l'objet de l'analyse réalisée. Ces données proviennent, dans la plupart des situations, des bases de données relationnelles. L'image de la situation de l'entreprise, ou le résultat produit par des générateurs d'analyse, se présente, soit sous forme de rapport statistique, soit sous forme de tableau de bord décisionnel. Les informations affichées dans un report ou dans un tableau de bord décisionnel sont utilisées pour vérifier l'alignement de la stratégie de l'entreprise et assister la direction de l'organisme, d'où le nom d'**informatique décisionnelle**. **[1]**

2.2. Processus d'informatique décisionnelle

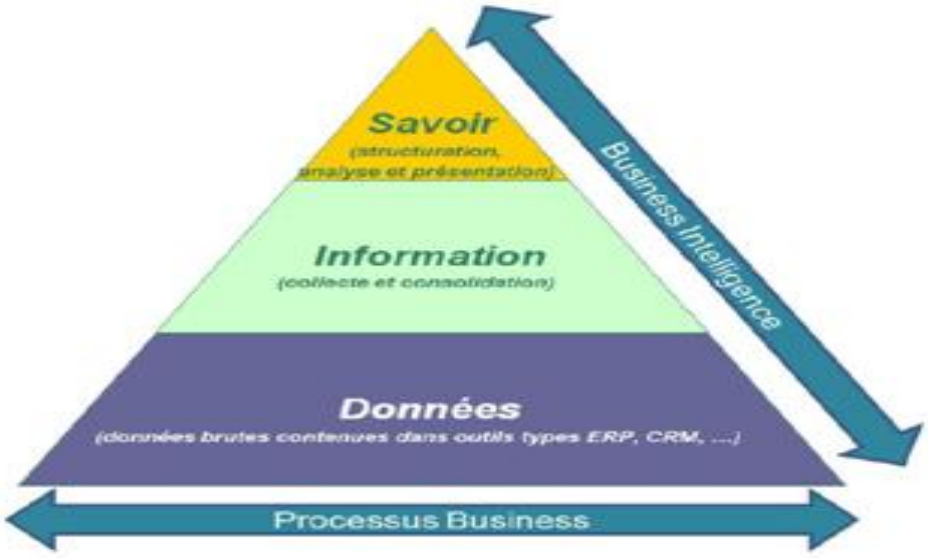


Figure 1.1 – Pyramide modélisant le processus de BI [1]

2.3. Les avantages de l'informatique décisionnelle

Business Intelligence (BI) est crucial pour la gestion efficace et l'emploi des données. Il est apparu dans la dernière partie du 20ième siècle et est devenue une partie intégrante du processus de prise pour les entreprises prudentes qui cherchent à faire usage d'une multitude de données relatives au service à la clientèle, les stocks, les prix, et bien plus encore la décision.

L'intégration des données est une caractéristique essentielle de toute solution de **BI** et peut tenir compte des facteurs qui affectent effectivement les préoccupations spécifiques du marché, tels que les données démographiques des clients, des conditions économiques et des environnements de marché. Ces analyses prédictives permettent aux entreprises d'affiner les processus d'affaires en préparation pour l'avenir. On peut soutenir que l'adaptation individuelle des processus d'affaires est une distinction essentielle entre les entreprises dans le même secteur. Ils utilisent souvent des technologies et des produits similaires, mais les personnaliser façons très différentes.

BI facilite la structure d'entreprise appropriée en fournissant un assortiment de données qui détaille ses besoins. Il permet d'identifier les zones qui ont un manque ou un surplus d'attention, d'accélérer la rationalisation des ressources de l'entreprise qui peuvent aider à la productivité. Solutions de **BI** compétitifs permettent la mise à jour immédiate, accordant aux utilisateurs les informations les plus récentes sur les processus de l'entreprise et leurs effets. **[6]**

2.4. Outils et fonctionnalités de Business Intelligence

Comme le terme Business Intelligence (**BI**) se réfère principalement à la collecte des processus, des technologies et des outils utilisés pour recueillir, stocker, accéder et analyser les données sur une entreprise, il y a un certain nombre d'outils utiles qu'une entreprise pourrait approcher de construire ses Business Intelligence (**BI**) capacités. Selon Robert J. Thierauf (2001, p. 163) un mélange de BI des outils pourraient inclure **l'entrepôt de données**, l'exploration de données, **OLAP** les sources des données, **les outils d'extraction, transformation et chargement ETL**, et ainsi de suite. Avec la mise à disposition de la table de pivot, Tabs Cross, calculs personnalisés, Assistant Requête Expertise, mise en page Commentaires et caractéristiques analytiques comme le classement, le filtrage, le tri, le groupe et ainsi de suite (maia-intelligence.com 2011), Business Intelligence (**BI**) est la réception d'un renforcée et une plus grande popularité parmi les entités commerciales et de nombreuses industries. Dans ce qui suit, nous allons détailler les deux outils essentiels de **l'informatique décisionnelle**. **[7]**

3. Entrepôts de données

3.1. Définition de l'entrepôt de données

Un entrepôt est un ensemble de données de différentes sources de l'entreprise centralisé et homogénéisé. Les systèmes ciblés pour des fins d'analyse seront incorporés à l'entrepôt pour ne former qu'un seul et unique système et ainsi d'aider les preneurs de décision en leur permettant d'accéder rapidement à l'information via des outils adaptés.

Selon SEN-SINHA, **un entrepôt de données** est: **orienté-sujet**, **intégré**, variant dans **le temps** et **non volatile**. Cette définition est dérivée de celle du créateur du terme «**entrepôt de données**» monsieur Bill Inmon.

L'entrepôt de données permet de consolider deux types de données: les données détaillées et les données agrégées. Les données détaillées représentent les transactions courantes qui doivent être chargées quotidiennement dans l'entrepôt. Leur **granularité** de chargement est journalière. Les données agrégées correspondent aux besoins d'analyse des utilisateurs. La fréquence de chargement dépend du niveau de granularité qui peut être mensuel, trimestriel ou annuel selon le besoin d'analyse requis. **[2]**

3.2. Les caractéristiques de l'entrepôt de données

L'entrepôt de données n'est pas une simple copie des données de production. Il est organisé et structuré.

Père du concept, Bill Immon dans son livre "Building the Data Warehouse" (John Wiley and Son 1996) le décrit ainsi :

« Subject oriented, integrated, nonvolatile, time variant collection of data in support of management decisions »

- **Orienté sujet**

  Au cœur du **Data warehouse,** les données sont organisées par thème. Les données propres à un thème, les ventes par exemple, seront rapatriées des différentes bases **OLTP** de production et regroupées.

- **Intégré**

  Les données proviennent de sources **hétérogènes** utilisant chacune un type de format. Elles sont **intégrées** avant d'être proposées à utilisation

- **Non volatile**

  Les données ne disparaissent pas et ne changent pas au fil des traitements, au fil du temps (Read-Only).

- **Historisé**

  Les données non volatiles sont aussi horodatées. On peut ainsi visualiser l'évolution dans le temps d'une valeur donnée.
  Le degré de détail de l'archivage est bien entendu relatif à la nature des données. Toutes les données ne méritent pas d'être archivées. **[8]**

Figure 1.2 –Data warehouse

3.3. La modélisation de l'entrepôt de données dans le niveau conceptuel

Afin de comprendre le niveau conceptuel de la modélisation d'un entrepôt de données, on va définir deux concepts : le concept de fait et le concept de dimension.

Concept de fait : Un fait représente un sujet d'analyse. Il est constitué de plusieurs mesures relatives au sujet traité. Ces mesures sont numériques et généralement valorisées de façon continue.

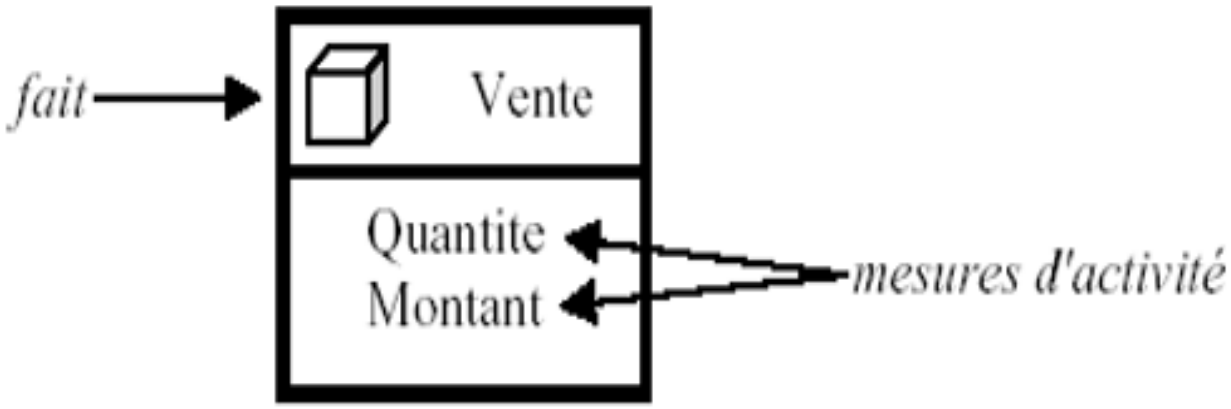


Figure 1.3 –Table des faits

Concept de dimensions : La dimension est le critère suivant lequel on souhaite évaluer, quantifié, qualifier le fait.

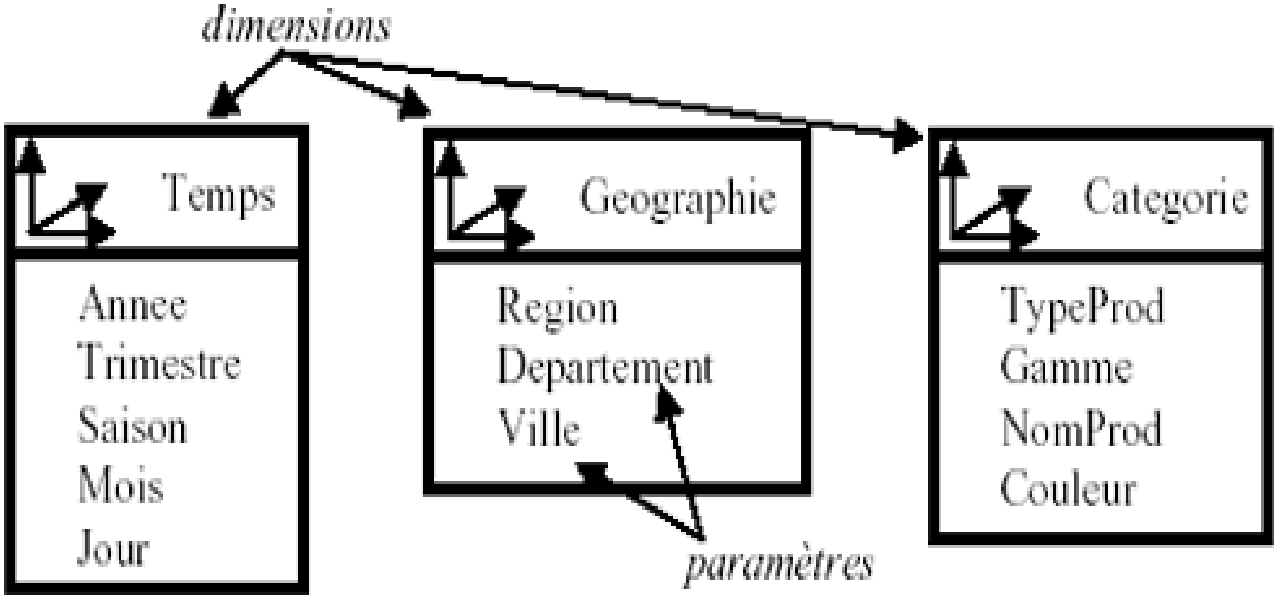


Figure 1.4 –Dimension

On part du principe que les données sont des faits à analyser selon plusieurs dimensions. Il est ainsi possible de réaliser une structure de données simple qui correspond à ce besoin de modélisation multidimensionnelle. Cette structure est constituée du fait central et des dimensions.

Au niveau logique cela peut se traduire par trois modèles différents : **en étoile,** en **flocon de neige**. **[9]**

3.3.1. Modèle en étoile

Le centre est la table des faits, et les branches en sont les dimensions. Pour une dimension il existe plusieurs faits. La structure est dissymétrique : la table des faits est énorme et les tables des dimensions sont petites. Les faits sont généralement numériques alors que les dimensions sont qualitatives. Une table de fait centrale et des dimensions, les dimensions n'ont pas de liaison entre elles. **[9]**

- **Avantages:**
  1. Facilité de navigation
  2. Nombre de jointures limité
- **Inconvénients:**
  1. Redondance dans les dimensions
  2. Toutes les dimensions ne concernent pas les mesures

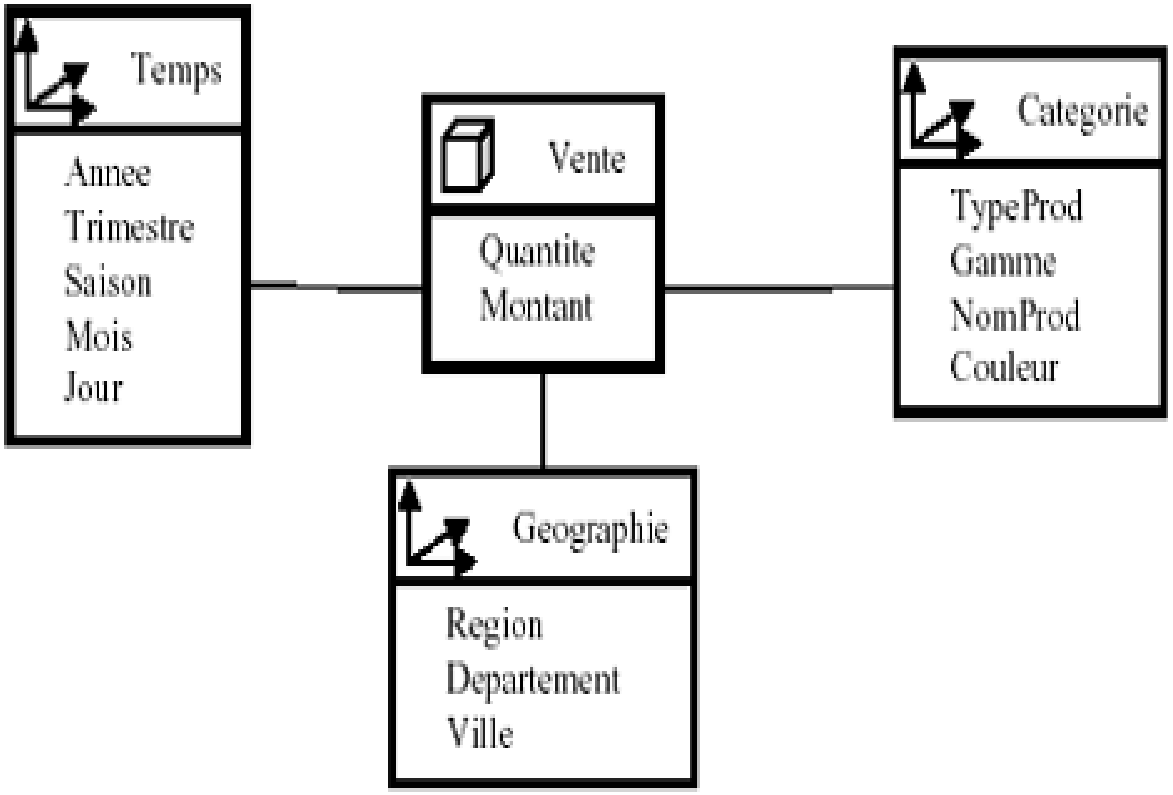


Figure 1.5 –Modèle en étoile

3.3.2. Modèle en flocon

Le principe est le même que pour le modèle en étoile, mais en plus les dimensions sont décomposées en sous hiérarchies. Le but est d'économiser ainsi de la place. Cela permet également d'instaurer une hiérarchie au sein des dimensions. Cela engendre par contre une complexification du modèle. La table de dimension de niveau hiérarchique le plus bas est reliée à la table de fait. On dit qu'elle a la granularité la plus fine. **[9]**

- **Inconvénients**:
  1. Modèle plus complexe (jointure)
  2. Requêtes moins performantes
- **Avantages:**
  1. Normalisation des dimensions
  2. Économie d'espace disque.

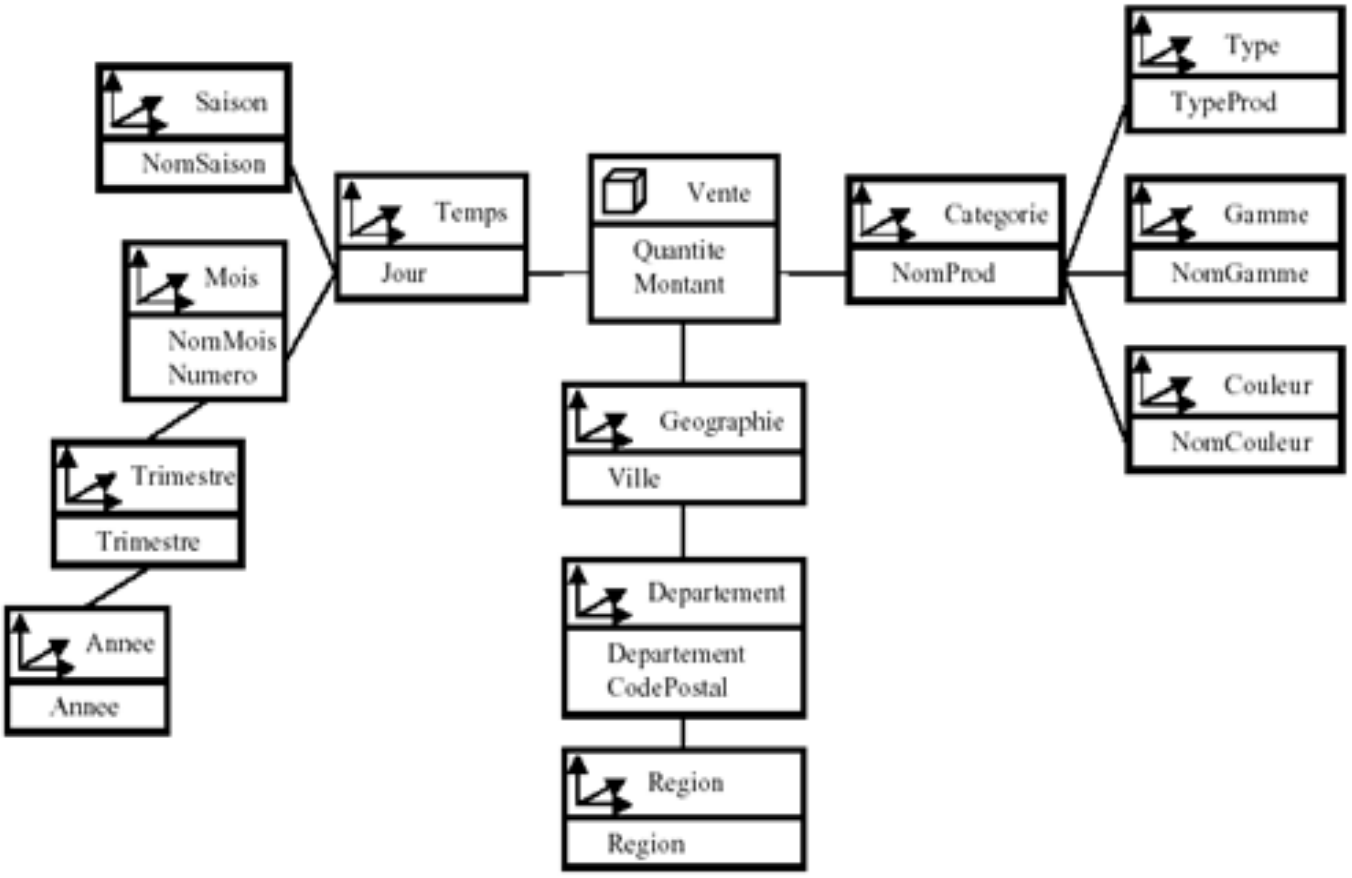


Figure 1.6 –Modèle en flocon

3.4. L'alimentation des données dans l'entrepôt de données :

L'alimentation des données à partir des bases de production est une phase primordiale d'un **entrepôt de données**. Des outils logiciels sont alors nécessaires pour **intégrer** les données dans **l'entrepôt de données**. On parle d'**ELT (Extract, Transform, Load)**. Les **phases de l'alimentation** d'un entrepôt de données sont les suivantes :

A. **Découverte des données** : Il s'agit d'identifier dans les systèmes sources les données à importer dans l'**entrepôt de données**. Il faut prendre les données les plus judicieuses. Un mauvais choix peut considérablement compliquer les phases suivantes de l'alimentation.

B. **Extraction des données** : Il s'agit de collecter les données utiles dans les systèmes de production. Il faut identifier les données ayant été modifiées afin d'importer le minimum de données dans l'entrepôt de données.

C. **Transformation des données** : Il faut rendre les données cohérentes avec la structure d'entrepôt de données. On d'applique alors des filtres sur les données. Il peut être nécessaire de convertir le format des données (EBCDIC vers ASCII par exemple) ou d'harmoniser les formats de dates (jj/mm/aaaa). Il faut également associer les champs source avec les champs cibles. Un champ source « adresse » pourra ainsi par exemple être décomposé en « numéro », « rue », « code postal », « ville » ou l'inverse. Enfin des données des systèmes de production doivent être agrégées ou calculées avant leur chargement.

D. **Chargement des données** : C'est la dernière phase de l'alimentation d'un **entrepôt de données**. Il s'agit d'insérer les données au sein d'entrepôt de données. C'est une phase délicate car les quantités de données sont souvent très importantes. **[10]**

Alimentation \ mise à jour d'entrepôt de données :

Entrepôt mis à jour régulièrement on a besoin d'un outil permettant d'automatiser les chargements dans **l'entrepôt des données** venant de multiple bases de données on a donc besoin d'un **ETL (Extract, Transform, Load)**

4. Processus d'ETL (Extracting – Transforming – Loading)

La majorité des systèmes d'information d'entreprise sont de nature **hétérogène** car le ou les systèmes d'information de l'entreprise s'élaborent le plus souvent sur de longues périodes. Bien que la standardisation des échanges entre les divers outils informatique avance à grands pas, la disparité des formats des données en circulation est toujours en réalité. C'est le principal obstacle technologique aux échanges d'informations.

Avant d'être utilisable, les données de l'entreprise doivent être mise en forme, Nettoyées et consolidées. Les outils **ETL (Extract Transform Load)** permettent d'automatiser ces traitements et de gérer les flux de données qui alimentent les entrepôts de données.

Les outils **ETL** font référence à plusieurs opérations qui s'articulent autour de trois axes majeurs :

- **Extraction**
- **Transformation**
- **Chargement.**

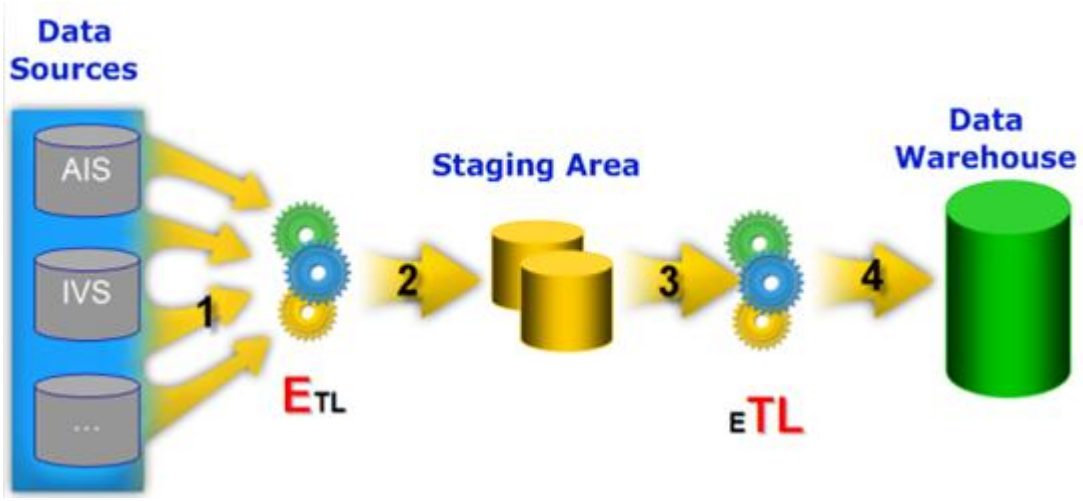


Figure 1.7 –Fonctionnement de processus ETL

4.1. Définition

Le **processus ETL** est un ensemble de tâches classées en trois catégories. Les tâches **d'extraction(E)** permettent de se connecter sur les données les plus pertinentes. Les tâches de **transformation (T)** préparent les données en vue de leur affecter des propriétés en termes de qualité, de format et de pertinence. Enfin, les tâches de **chargement (L),** basées sur un schéma de mappage, permettent de charger les données dans l'entrepôt de données (**DW** : **Data Warehouse**). La couche inférieure de la figure 1.8 présente la partie statique qui montre les sources de données, le **Data Staging Area (DSA)** vers lequel sont rapatriées les données pour être préparées et le **DW** représentant la destination finale des données. Dans la couche supérieure, sont représentées l es trois phases d'**ETL** à savoir **extraction, transformation** et **chargement**. **[3]**

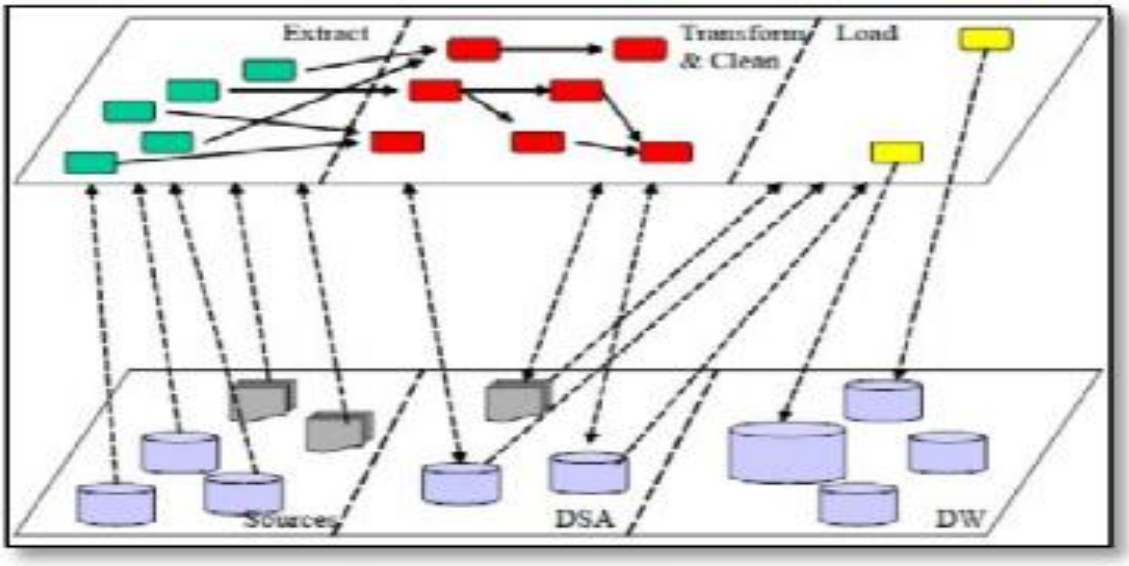


Figure 1.8 –Environnement d'un processus ETL

4.2. Les avantages ELT

- **Maximiser** la performance
- **Efficace** : pas de technologie intermédiaire
- **Rapide** : dialogue en langage natif avec les acteurs mis en jeu
- **Simplifier et rationaliser** l'architecture
- **Non intrusif** : aucun système supplémentaire à installer
- **Distribué** : la charge peut être lissée sur l'ensemble du système
- **Optimiser** l'utilisation des technologies déjà en place
- **Rentabilité** : la puissance disponible des systèmes existants est utilisée
- **La connaissance et la maîtrise** des systèmes par le client sont mutualisées. **[11]**

4.3. Les phases ETL

Pendant le **processus d'ETL**, les données sont extraites d'une base de données **OLTP**, transformées pour correspondre au schéma **d'entrepôt de données**, et chargé dans la base de données de **l'entrepôt de données**. De nombreux DW contiennent également des données provenant de systèmes **non-OLTP**, tels que les fichiers texte, les systèmes existants, et des feuilles de calcul. **ETL** est souvent une combinaison complexe de processus et de la technologie qui consomme une partie importante des efforts de développement de **l'entrepôt de données** et nécessite les compétences des analystes d'affaires, les concepteurs de bases de données et les développeurs d'applications. **Le processus ETL** n'est pas un événement ponctuel. Comme sources de données changent l'entrepôt de données sera mise à jour périodiquement. En outre, comme entreprise modifie le **système DW** doit changer - afin de maintenir sa valeur comme un outil pour les décideurs, à la suite de ce que l'**ETL** change et évolue également. Les processus **ETL** doivent être conçus pour faciliter la modification. Un système **ETL** solide, bien conçu et documenté est nécessaire à la réussite d'un projet d'entrepôt de données. Un système d'**ETL** se compose de trois étapes fonctionnelles consécutives: **extraction, transformation et chargement**:

4.3.1. Extraction

La première étape de tout scénario d'**ETL** est l'extraction de données. L'étape **d'extraction ETL** est chargée d'extraire des données à partir des systèmes source. Chaque source de données a son ensemble distinct de caractéristiques qui doivent être gérés afin d'extraire efficacement les données pour le **processus ETL**... **[4]**

Pour mieux détaillé :

❖ **Considérations pratiques:**

- Identifier les sources de données et leurs structures.

- Décider, pour chaque source, si l'extraction est faite à la main (ex: script) ou à l'aide d'un outil;
- Choisir, pour chaque source, la fenêtre temporelle durant laquelle sera faite l'extraction;
- Déterminer la séquence des tâches d'extraction;
- Déterminer Comment gérer les exceptions.

❖ **Identification des sources:**

- Énumérer les items cibles (métriques et attributs de dimension) nécessaires à **l'entrepôt de données**;
- Pour chaque item cible, trouver la source et l'item correspondant de cette source;
- Si plusieurs sources sont trouvées, choisir la plus pertinente;
- Si l'item cible exige des données de plusieurs sources, former des règles de consolidation;
- Si l'item source referme plusieurs items cibles (ex: un seul champ pour le nom et l'adresse du client), définir des règles de découpage;
- Inspecter les sources pour des valeurs manquantes.

➢ **Types d'extraction des données :**

a) Extraction complète:

Capture l'ensemble des données à un certain instant (*snapshot* de l'état opérationnel);

- Normalement employée dans deux situations:
  1. Chargement initial des données;
  2. Rafraîchissement complet des données (ex: modification d'une source).
- Peut êtrebtrès coûteuse en temps (ex: plusieurs heures/jours).

b) Extraction incrémentale:

- Captureuniquementlesdonnéesquiontchangéesouontétéajoutéesdepuisladernièreextraction;
- Peut être faite de deux façons:
  1. Extraction temps-réel;
  2. Extraction différée (en lot). **[5]**

b).1. Extraction temps-réel :

S'effectue au moment où les transactions surviennent dans les systèmes sources.

i. Capture à l'aide du journal des transactions :

A. Utilise les logts de transactions de la BD servant à la récupération en cas de panne ;
B. Aucune modification requise à la BD ou aux sources ;
C. Doit être fait avant le rafraichissement périodique du journal ;
D. Pas possible avec les systèmes legacy ou les sources à base de fichiers (il faut une BD journalisée)

ii. **Capture à l'aide de triggers :**

A. Des procédures déclenchées (triggers) sont définies dans la BD pour recopier les données à extraire dans un fichier de sortie ;

B. Meilleur contrôle de la capture d'événement ;
C. Exige de modifier les BD sources ;
D. Pas possible avec les systèmes legacy ou les sources à base de fichiers.

iii. Capture à l'aide des applications sources :

A. Les applications sources sont modifiées pour écrire chaque ajout et modification de données dans un fichier d'extraction ;
B. Exige des modifications aux applications existences ;
C. Entraine des couts additionnels de développement et de maintenance ;
D. Peut être employé sur des systèmes legacy et les systèmes à base de fichiers. **[5]**

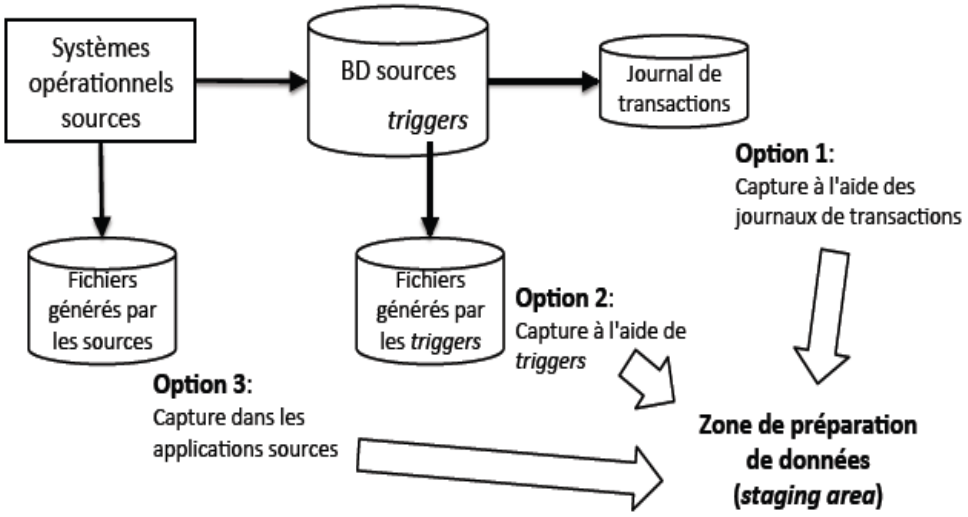


Figure 1.9 –Extraction en temps réel

b).2. Extraction différée :

Extrait tous les chargements survenus durant une période donnée (ex : heure, jour, semaine, mois).

i. Capture basée sur les Times tamps :

A. Une estampille (*times tamp*) d'écriture est ajoutée à chaque ligne des systèmes sources;
B. L'extraction se fait uniquement sur les données dont le *times tamp* est plus récent que la dernière extraction;
C. Fonctionne avec les systèmes *legacy* et les fichiers plats, mais peut exiger des modifications aux systèmes sources;
D. Gestion compliquée des suppressions.

ii. Capture par comparaison de fichiers :

A. Compare deux *snapshots* successifs des données sources;
B. Extrait seulement les différences (ajouts, modifications, suppressions) entre les deux *snapshots*;

C. Peut être employé sur des systèmes *legacy* et les systèmes à base de fichiers, sans aucune modification;
D. Exige de conserver une copie de l'état des données sources;
E. Approche relativement coûteuse. **[5]**

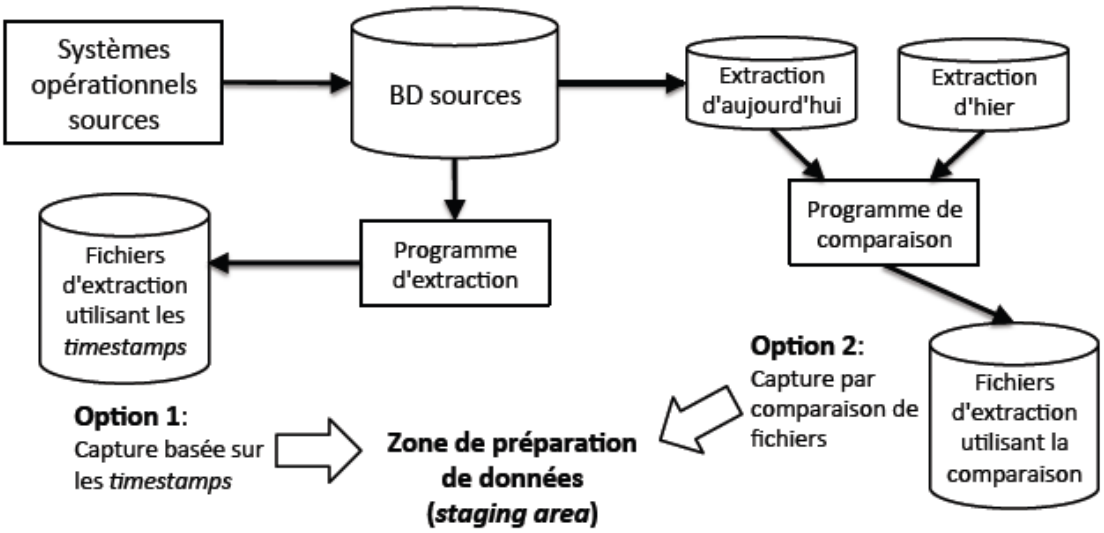


Figure 1.10 –Extraction différée

4.3.2. Transformation

La deuxième étape dans tout scénario d'**ETL** est la transformation des données. L'étape de **transformation** a tendance à faire un peu de nettoyage et conforme sur les données entrantes pour obtenir des données précises qui sont correctes, complètes, cohérentes, et sans ambiguïté. Ce processus comprend le nettoyage des données, la transformation et l'intégration. Il définit la granularité des tables de fait, les tableaux de dimensions, schéma **DW (star ou flocon de neige**), des faits dérivés, changeant lentement les dimensions, les tables de fait. Toutes les règles de **transformation** et les schémas résultants sont décrits dans le référentiel de métadonnées. **[4]**

- **Types de transformation:**
  1. Révision de format:
     - Ex: Changer le type ou la longueur de champs individuels.
  2. Décodage de champs:
     - Consolider les données de sources multiples

     Ex: ['homme','femme'] vs ['M','F']vs[1,2].
     - Traduire les valeurs cryptiques

     Ex: 'AC', 'IN', 'SU' pour les statuts actifs, inactifs et suspendus.
  3. Pré-calcul des valeurs dérivées:
     Ex: profit calculé à partir de ventes et coûts.
  4. Découpage de champs complexes:
     Ex: extraire les valeurs *prénom*, *second Prénom* et *nom Famille* à partir d'une seule chaîne de caractères *nom Complet*.

5. Fusion de plusieurs champs:
   Ex: information d'un produit
   – Source1:code et description;
   – Source2: types de forfaits;
   – Source3:coût.
6. Conversion de jeu de caractères:
   Ex: EBCDIC(IBM) vers ASCII.
7. Conversion des unités de mesure:
   Ex: impérial à métrique.
8. Conversion de dates:
   Ex: '24FEB2011' vs '24/02/2011' vs '02/24/2011'.
9. Pré-calcul des agrégations:
   Ex: ventes par produit par semaine par région.
10. Déduplication:
    Ex: Plusieurs enregistrements pour un même client. **[5]**

4.3.3. Chargement

Chargement des données à la structure multidimensionnelle cible est l'étape finale **ETL**. Dans cette étape, on extrait et transformé les données sont écrites dans les structures tridimensionnelles réellement accessibles par les utilisateurs finaux et les systèmes d'application. **Chargement** étape comprend deux tables de dimension de chargement et de tables de chargement d'information. **[4]**

- **Types de chargement:**

i. Chargement initial:

Fait une seule fois lors de l'activation de l'entrepôt de données;

A. Les indexes et contraintes d'intégrité référentielle (clé étrangères) sont normalement désactivés temporairement;
B. Peut prendre plusieurs heures.

ii. Chargement incrémental:

Fait une fois le chargement initial complété;

A. Tient compte de la nature des changements (ex: SCDType1, 2ou 3);
B. Peut être fait en temps-réel ou en lot.

iii. Rafraîchissement complet:

A. Employé lorsque le nombre de chargements rend le chargement incrémental trop complexe ;
B. Ex : lorsque plus de 20% des enregistrements sont changé depuis dernier chargement.

iv.Considération additionnelles:

A. Faire les chargements en lot dans une période creuse (entrepôt de données non utilisé);
B. Considérer la bande passante requise pour le chargement;
C. Avoir un plan pour évaluer la qualité des données chargées dans l'entrepôt;
D. Commencer par charger les données des tables de dimension. **[5]**

4.4. Classification des travaux sur le processus ETL

Nous proposons, dans cette section, une classification des travaux sur l'**ETL** sous forme de deux approches en se basant sur la parallélisassions de celui-ci : « approche classique » et « approche basée sur le modèle MR ».

4.4.1. Processus ETL classique

Nous considérons que le **processus ETL** est classique lorsque celui-ci s'exécute sur un serveur en une seule instance (une seule exécution en même temps) et où les données sont de taille modérée. **[3]**

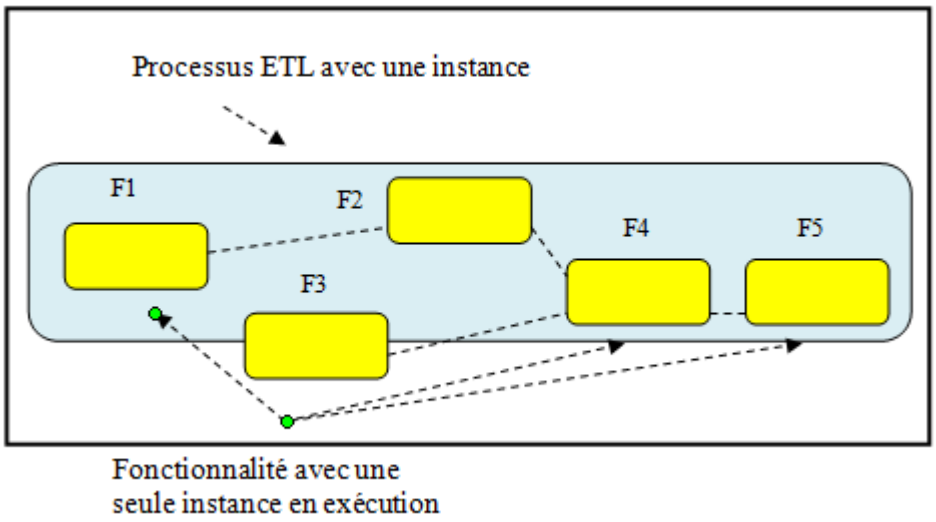


Figure 1.11 –Processus ETL classique

Dans ce contexte, seules les fonctionnalités d'**ETL** indépendantes peuvent s'exécuter en parallèle. Une fonctionnalité d'**ETL**, telles que Changing Data Capture (**CDC**), Surrogate Key (**SK**), SlowlyChanging Dimension (**SCD**), Surrogate Key Pipeline (**SKP**), est une fonction de base qui prend en charge un aspect particulier dans un **processus ETL**. La fonctionnalité **CDC**, par exemple, permet dans la phase d'extraction d'un **processus ETL** d'identifier, dans les systèmes sources, les tuples affectés par des changements afin de les capturer et les considérer dans le rafraichissement Dans la figure précédente, nous remarquons que les fonctionnalités F1 et F3 ou F2 et F3 peuvent s'exécuter en parallèle.

4.4.2. Processus ETL basé sur le modèle MapReduce (MR)

Le processus **ETL**, avec un schéma classique, ne pourra pas faire face à l'**intégration** de **données massives**. Le paradigme **MR** permet de partitionner de gros volumes de données dont chaque partition sera soumise à une instance du processus **ETL**.

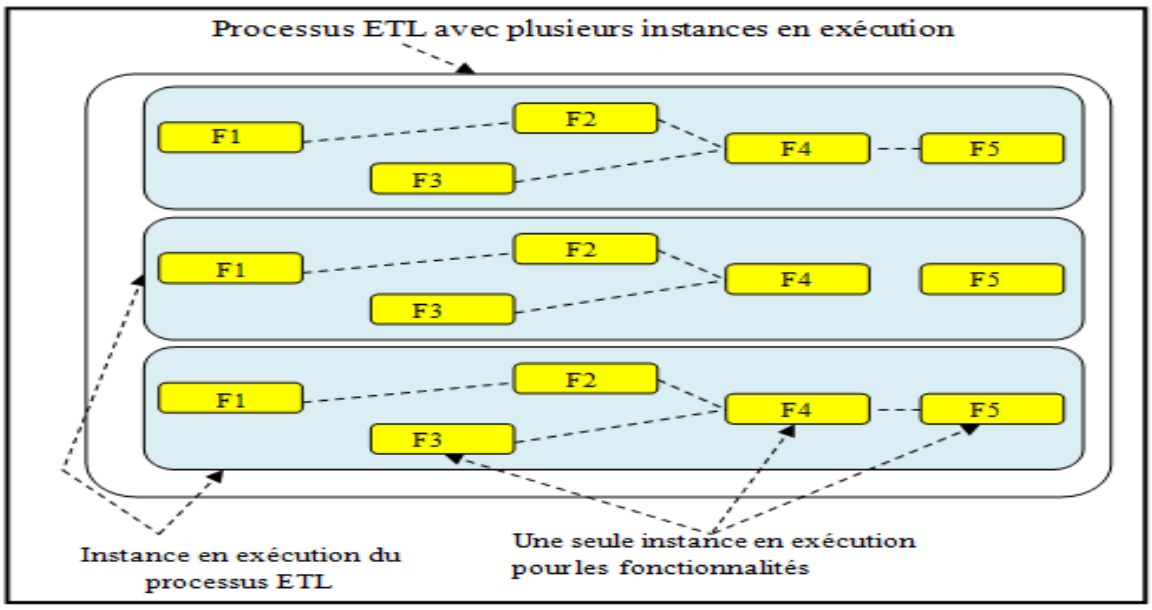


Figure 1.12 –Processus ETL basé sur le modèle MR

Comme le montre la figure précédente, plusieurs instances du processus **ETL** s'exécutent en parallèle où chacune d'elles traitera les données de sa partition dans une phase appelée **Map**. Les résultats partiels seront fusionnés et agrégés, dans une phase Reduce, pour obtenir des données prêtes à être chargées dans le **DW**. **[3]**

4.5. Le future No ETL (Not Only ETL)

ETL existe parce que les systèmes traditionnels ne pouvaient pas gérer à la fois **OLTP** et **OLAP** dans un seul système et de fournir de bonnes performances pour les deux. **ETL** sur Hadoop ouvre la porte à une nouvelle façon de penser **ETL** car il modifie la structure des coûts autour exploité **Big Data**. **ETL** est devenue critique car elle a permis de stockage hiérarchisé où les données à partir d'applications opérationnelles pourraient être gérées et manipulées dans un entrepôt de données, au lieu de surcharger le système frontal. Un système de **Hadoop** typique peut être un substitut à un entrepôt de données, mais pas pour les applications opérationnelles.

Maintenant, les progrès en mémoire des technologies telles que Spark peuvent rendre obsolètes **ETL** en exécutant les applications **OLTP** et **OLAP** à partir de la même plate-forme. Ceci élimine le besoin d'extraire ou de la charge de données, comme toutes les applications pourraient tirer d'une instance de **HDFS** et juste être transformé pour adapter le format de la base de données cible. Cela pourrait prendre les processus **ETL** d'aujourd'hui des heures et des jours, à quelques secondes, permettant aux applications et aux analystes de bénéficier de données en temps quasi réel qui est livré de façon transparente tout au long de la journée. **[12]**

- **Pourquoi NoETL?**

ETL est une étape intermédiaire, et à chaque étape d'**ETL** vous pouvez introduire des erreurs et des risques:

- ETL peut perdre des données
- ETL peut dupliquer les données après le basculement outils ETL peuvent coûter des millions de dollars.
- ETL diminue le débit.
- ETL augmente la complexité du pipe-line.

Un bon signe que vous rencontrez ETL il est écrit des fichiers intermédiaires. Fichiers ont eu leur journée sous le soleil, mais ne sont pas type ful et certaines personnes préconisent le remplacement d'un système de fichiers avec une meilleure API complètement.

Nous dirions, en quelque sorte, **ETL** est la conduite sous Unix, et **NoETL** est un objet tapé dans PowerShell. Nous utiliserons la progression du niveau de Kafka sur l'Unix pour bus de message et de le comparer avec Oracle. Si vous stockez vos données transactionnelles dans Oracle, qui est encore une chose à des données financières pour les seules raisons de conformité, vous devez chercher périodiquement ces données et de les utiliser en aval. **[13]**

5. Conclusion

Dans ce chapitre, nous avons discuté de l'environnement **BI (Business Intelligence)** et les entrepôts et leur gestion de stockage de données. Nous avons fermé le chapitre en identifiant la solution **ETL (Extract Transform Load)** pour extraire, transformer et charger les données source dans un entrepôt de données.

Le stockage des sources de données à une vitesse croissante, dans ce cas ont mis les systèmes d'information (SI) face au défi du **Big Data**. Ces données sont caractérisées par, en plus de leur volumétrie et la vitesse avec laquelle elles sont générées, **une hétérogénéité** plus importante suite à l'émergence de nouvelles structures de données. Dans le chapitre suivant, vous obtenez la solution de **extraction** et le nettoyage, la **transformation** ainsi le **chargement** des sources de **données massives (Big Data)**.

CHAPITRE II

Les Big Data (Les données massives)

Chapitre 2

1. Introduction

Dans chaque jour, nous générons 2,5 trillions d'octets de données, 90% de ces données dans le monde ont été créées au cours des deux dernières années. Les données sont arrivées depuis plusieurs sources y'en a qui sont des capteurs utilisés pour collecter les informations climatiques ou bien des messages sur les médias sociaux (facebook, gmail…), images numériques et vidéos publiées en ligne, enregistrements transactionnels d'achat en ligne (Amazon, eBay…), des signaux GPS de téléphones mobiles…
Ces données appelées des **Big Data** ou les **Données Massives**, et c'est ça ce que nous allons détailler dans ce deuxième chapitre.

2. Concept Big Data

Le terme « **grandes données** » a été la première fois présenté au monde de calcul par Roger Magoulas de media d'O'Reilly en 2005 afin de définir une grande quantité de données que les techniques traditionnelles de gestion des données ne peuvent pas contrôler et traiter en raison de la complexité et de la taille de ces données. Une étude sur l'évolution de grandes données comme recherche et sujet scientifique prouve que le terme « grandes données » était présent dans la recherche commençant par les années 1970 mais a été comportée en publications en 2008.

De nos jours le grand concept de données est traité de différents points de vue couvrant ses implications dans beaucoup de domaines. Selon Mike 2,0, la norme ouverte de source pour la gestion de l'information, de grandes données sont définie par sa taille, comportant une grande, complexe et indépendante collecte des données des ensembles, chacune avec le potentiel d'agir l'un sur l'autre. En outre, un aspect important de grandes données est le fait qu'il ne peut pas être manipulé avec des techniques standard de gestion des données dues à la contradiction et à l'imprévisibilité des combinaisons possibles.

Dans une définition plus simple nous considérons comme étant de grandes données une expression qui comporte différents ensembles de données très de grand, fortement complexe, non structuré, organisé, stocké et traité utilisant des méthodes spécifiques et des techniques utilisées pour des processus d'affaires.

Il y a beaucoup de définitions sur de **Big Data** circulant autour du monde, mais nous considérons que le plus important est celui que chaque chef donne à ses une données de société.

La manière dont de **Big Data** sont définis a l'implication dans la stratégie des affaires. Chaque chef doit définir le concept afin d'apporter l'avantage compétitif pour la société. **[14]**

3. Les caractéristiques des Big Data

3.1. Le volume

Le volume se rapporte à l'immense quantité de données a produit de chaque seconde. Pensez juste à tous les emails, messages de Twitter, clips de photo, vidéo et données de sonde que nous produisons et partageons chaque seconde. Nous ne parlons pas des Terabyte, mais des zettabytes ou des brontobytes des données. **[20]**

3.2. La vitesse

La vitesse se rapporte à la vitesse à laquelle de nouvelles données sont produites et à la vitesse autour derrière la à laquelle les données se déplacent. Pensez juste aux messages sociaux de media allant viraux en quelques minutes, **la vitesse** à laquelle des transactions de carte de crédit sont examinées pour assurer les activités frauduleuses ou les millisecondes où elle prend des systèmes de commerce pour analyser les réseaux sociaux de media pour prendre des signaux que des décisions de déclencheur pour acheter ou vendre des actions. La grande technologie de données nous permet maintenant d'analyser les données tandis qu'elle est produite sans la mettre jamais dans des bases de données. **[20]**

3.3. La variété

La variété se rapporte aux différents types de données que nous pouvons maintenant employer. Dans le passé nous nous sommes concentrés sur les données structurées qui s'insèrent d'une manière ordonnée dans des tables ou des bases de données relationnelles telles que des données financières (par exemple, sous-produit ou région de ventes). En fait, 80 pour cent des données du monde sont maintenant non structurés et ne peuvent pas donc facilement être mis dans des tables ou apparenté base de données-pensez aux photos, aux ordres visuels ou aux mises à jour sociales de media. Avec la grande technologie de données nous pouvons maintenant armer les types différés de données comprenant des messages, des conversations sociales de media, des photos, des données de sonde, la vidéo ou des enregistrements de voix et les apporter ainsi que des données plus traditionnelles et plus structurées. **[20]**

3.4. La véracité

La véracité se réfère au désordre de la fiabilité des données. Avec de nombreuses formes de données importantes, la qualité et la précision sont moins contrôlables, pour les postes exemple Twitter avec hashtags, les abréviations, les fautes de frappe et le langage familier. La technologie **Big Data** et d'analyse nous permet maintenant de travailler avec ces types de données. **Les volumes** font souvent le manque de qualité ou l'exactitude.

Mais tous les volumes de données rapide mouvement de **la variété** et de **la véracité** différente doivent être transformés en valeur! Ceci est la raison pour laquelle la valeur est celle de V données volumineuses qui importe le plus. **[20]**

3.5. La valeur

Si nous produisons tellement des données, alors quelle quantité de lui est réellement utile ? La compréhension de **la valeur** de ces données est cruciale de sorte que le processus puisse être optimisé. L'investissement dans de grandes solutions de données est cher ainsi il est très important de s'assurer que les données étant extraites sont précieuses pour la société. **[20]**

3.6. Variabilité

Il y a plusieurs significations potentielles pour **la variabilité.** Les données sont-elles cohérentes en termes de disponibilité ou intervalle du reportage ? Dépeignent-elles exactement l'événement rapporté ? Quand les données contiennent beaucoup de valeurs extrêmes qu'elles présentent un problème statistique pour déterminer quoi faire avec des ces

« annexe » évaluent et si elles contiennent un nouveau et important signal ou sont juste des données bruyantes. **[W11]**

3.7. Validité

Comme **Big Data** la véracité est la question de la signification de validité est les données correctes et précises pour l'usage prévu. Les données clairement valides sont principales à prendre les bonnes décisions. Phil Francisco, VP de gestion du produit d'**IBM** a parlé au sujet de la grande stratégie et des outils des données d'IBM qu'elles offrent d'aider avec **la véracité** de données et **la validité**. **[20]**

3.8. Volatilité

La grande **volatilité** de données se rapporte à de quelle longueur sont les données valides et à combien de temps devraient elles être stockées. En ce monde des données en temps réel vous devez déterminer à quel point n'est des données plus concernant l'analyse actuelle. Les Big Data traitent clairement des questions au delà de volume, de variété et de vitesse à d'autres soucis comme la véracité, **la validité** et **la volatilité**. Pour entendre parler des autres grandes tendances et présentation de données suivez le grand sommet d'innovation de données sur le Twitter. [**20]**

4. L'origine des données

Les données traitées par le **Big Data** proviennent notamment :

- du Web: les textes, les images, les vidéos, et tous ce qui peut figurer sur les pages Web.
- plus généralement, de l'internet et des objets communicants: réseaux de capteurs, journaux des appels en téléphonie;
- des sciences: génomique, astronomie, physique subatomique, etc.;
- données commerciales
- données personnelles (ex: dossiers médicaux);
- données publiques (open data).

Ces données sont localisées généralement dans les **Data Warehouses** d'entreprises ou chez les fournisseurs du **Cloud**. Ce qui facilite leur traitement en utilisant des méthodes adaptées à ces architectures. On peut citer par exemple l'algorithme **MapReduce**. **[15]**

5. Les principaux acteurs

Les acteurs sont les organismes qui « produisent » du **Big Data**, et dans la plupart des cas ce sont leurs fondateurs. « Les principales innovations du domaine trouvent leur origine chez les leaders du Web : Google (MapReduce et Big Table), Amazon (Dynamo, S3), Yahoo! (Hadoop, S4), Facebook (Cassandra, Hive), Twitter (Storm, FlockDB), LinkedIn (Kafka, SenseiDB, Voldemort), Live Journal (Memcached), etc…

La Fondation Apache est ainsi particulièrement active dans ce domaine, en lançant ou en recueillant plus d'une dizaine de projets, matures ou en incubation: **Hadoop**, **Hbase**, **Hive**,

Pig, **Cassandra**, **Mahout**, **Zookeeper**, Kafka, Flume, Hama, Giraph, etc. Outre les sociétés du Web, le secteur scientifique et plus récemment les promoteurs de l'Open Data (et de sa variante, l'Open Linked Data, issu du Web Sémantique), sont également historiquement très ouverts à l'Open Source, et ont logiquement effectué des contributions importantes dans le domaine du Big Data. **[15]**

6. L'importance des Big Data

L'importance principale de **Big Data** consiste en la possibilité d'améliorer l'efficacité dans le cadre de l'utilisation d'un grand volume de données, de type différent.

Si **Big Data** est correctement défini et utilisé en conséquence, les organisations peuvent obtenir une meilleure vue sur leur activité principale donc à l'efficacité dans différents domaines tels que les ventes, l'amélioration du produit manufacturé et ainsi de suite.

Les **Big Data** peuvent être utilisées efficacement dans les domaines suivants:

- Dans la technologie de l'information afin d'améliorer la sécurité et le dépannage en analysant les tendances dans les journaux existants;
- Dans le service à la clientèle en utilisant des informations provenant des centres d'appels afin d'obtenir le motif de la clientèle et accroître ainsi la satisfaction du client par les services personnalisation;
- Dans l'amélioration des services et des produits à travers l'utilisation du contenu des médias sociaux. En connaissant les préférences des clients potentiels de l'entreprise : peut modifier son produit afin de répondre à une plus grande surface de personnes;
- Dans la détection de la fraude dans les transactions en ligne pour toute l'industrie;
- Dans l'évaluation des risques en analysant les informations des transactions sur le marché financier.

Tout ce ci orientera les entreprises vers une économie centrée sur la donnée. **[14]**

7. Le stockage des Big Data

- **Cloud Computing**

L'un des éléments les plus importants dans le nouveau grand paradigme de données est la disponibilité croissante du stockage et des applications pour les opérations commerciales plus petites nuages. Systèmes de cloud computing permettent le stockage à distance et l'accès aux données de l'entreprise grâce à des technologies de connexion sécurisées. Dans la pratique générale, le nuage permet aux membres du personnel pour effectuer les données de travail et d'accès à tout moment et de partout avec une connexion Internet sécurisée. Cela peut augmenter les bénéfices de productivité et d'augmenter dans les entreprises de toutes tailles. Dans le contexte des grandes données, cependant, les systèmes de cloud peuvent permettre le stockage de données véritablement évolutives pour les grands projets et les besoins de tous les jours. Cela peut rendre l'analyse de données grande une possibilité abordable pour de nombreuses petites entreprises, sans les ressources nécessaires pour maintenir un espace de serveur sur site pour ces tâches d'analyse. **[21]**

Figure 2.2 –Les Big Bata dans le Cloud Computing

8. Une technologie des Big Data

- **Hadoop**

➢ **Définition**

Hadoop fournit un stockage pour **Big Data** à un coût raisonnable Stockage Big Data en utilisant le stockage traditionnel peut être coûteux. Hadoop est construit autour du matériel de base.

Par conséquent, il peut fournir assez grand stockage pour un coût raisonnable. **Hadoop** a été utilisé dans le domaine à l'échelle de Peta octet. Une étude réalisée par **Cloudera** a suggéré que les entreprises dépensent généralement autour de 25000 $ à 50 000 $ dollars par octet de tera par an. Avec Hadoop ce coût tombe à quelques milliers de dollars par octet de tera par an. Et le matériel devient moins cher et moins cher ce coût continue de baisser.

En utilisant la solution fournie par Google, Doug Cutting et son équipe ont développé un projet Open Source qu'il est le **Hadoop**.

Hadoop exécute des applications en utilisant l'algorithme **MapReduce**, où les données sont traitées en parallèle avec les autres. En bref, **Hadoop** est utilisé pour développer des applications qui pourraient effectuer une analyse statistique complète sur d'énormes quantités de données. **[16]**

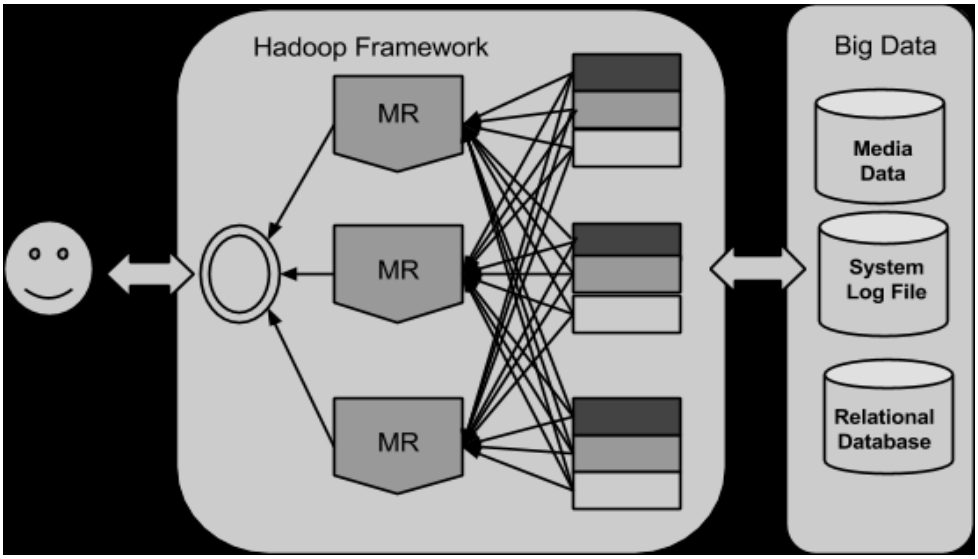


Figure 2.2 –Architecture d'un Hadoop avec les Big Data

Hadoop est un framework open source Apache écrit en java qui permet le traitement de grands ensembles de données réparties entre des grappes d'ordinateurs à l'aide de modèles de programmation simples. L'application **framework Hadoop** fonctionne dans un environnement qui fournit le stockage distribué et de calcul entre les grappes d'ordinateurs. **Hadoop** est conçu pour évoluer à partir serveur unique à des milliers de machines, offrant à chaque calcul et le stockage local. **[16]**

➢ Les avantages de Hadoop

Hadoop permet à l'utilisateur d'écrire rapidement et les systèmes de test distribué. Il est efficace, et il distribue automatiquement les données et le travail à travers les machines et à son tour, utilise le parallélisme sous-jacent des noyaux CPU.

- ✓ **Hadoop** ne repose pas sur le matériel pour fournir la tolérance aux pannes et la haute disponibilité, plutôt bibliothèque **Hadoop** lui-même a été conçu pour détecter et gérer des défaillances au niveau de la couche application.
- ✓ Les serveurs peuvent être ajoutés ou supprimés de la grappe dynamique et **Hadoop** continue à fonctionner sans interruption.
- ✓ Un autre grand avantage de **Hadoop** est qu'en plus d'être open source, il est compatible sur toutes les plates-formes, car il est basé sur Java. **[16]**

➢ Les composants de Hadoop

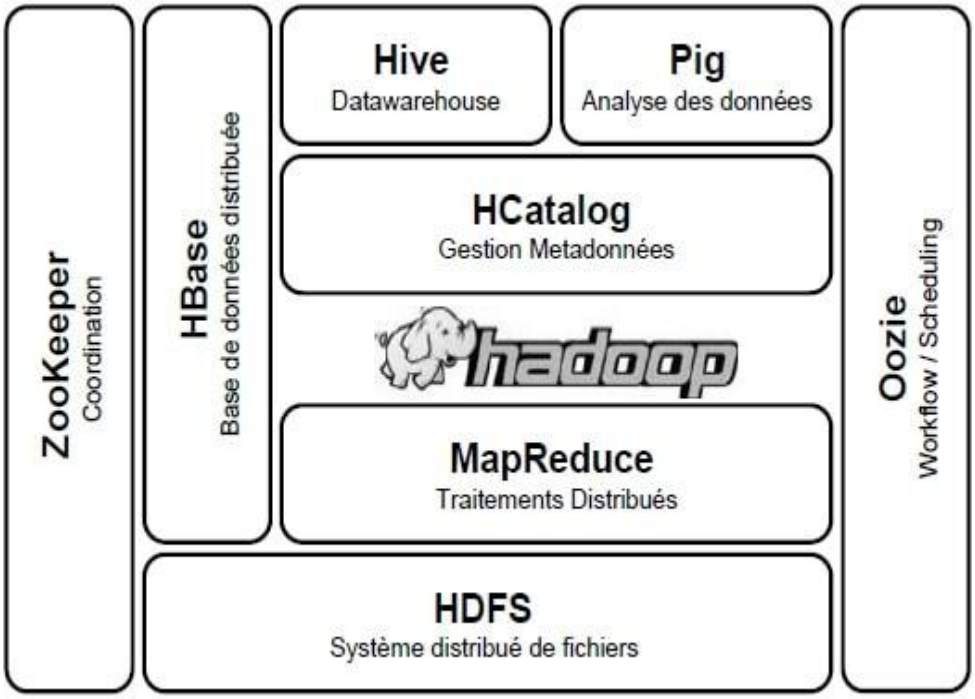


Figure 2.3 –L'architecture des composants Hadoop

❖ Système de fichiers distribué de Hadoop (HDFS)

Le système de fichiers distribué de **Hadoop** est, car le nom suggère évidemment, des systèmes de fichiers distribué modelés après le système de fichiers de Google.

Comme mentionné ci-dessus lui est le composant distribué primaire de stockage de données employé par d'autres applications de **Hadoop**, mais peut également servir de système de fichiers autonome. **HDFS** rend la distribution aux utilisateurs complètement transparents.

Aux utilisateurs il fournit la vue d'un système de fichiers traditionnel au-dessus d'une hiérarchie des annuaires et des dossiers. Seulement intérieurement il trace et distribue des dossiers à différents nœuds dans le groupe et traduit des opérations d'annuaire et de dossier aux opérations réparties sur plusieurs nœuds.

Généralement **HDFS** prévoit le stockage de coût bas, c'est bon pour lire séquentiellement les blocs de données entiers, ainsi pour les tâches qui exigent l'accès aux données séquentiel ou conduisent des opérations au-dessus d'une gamme étendue de points de repères séquentiels. **[17]**

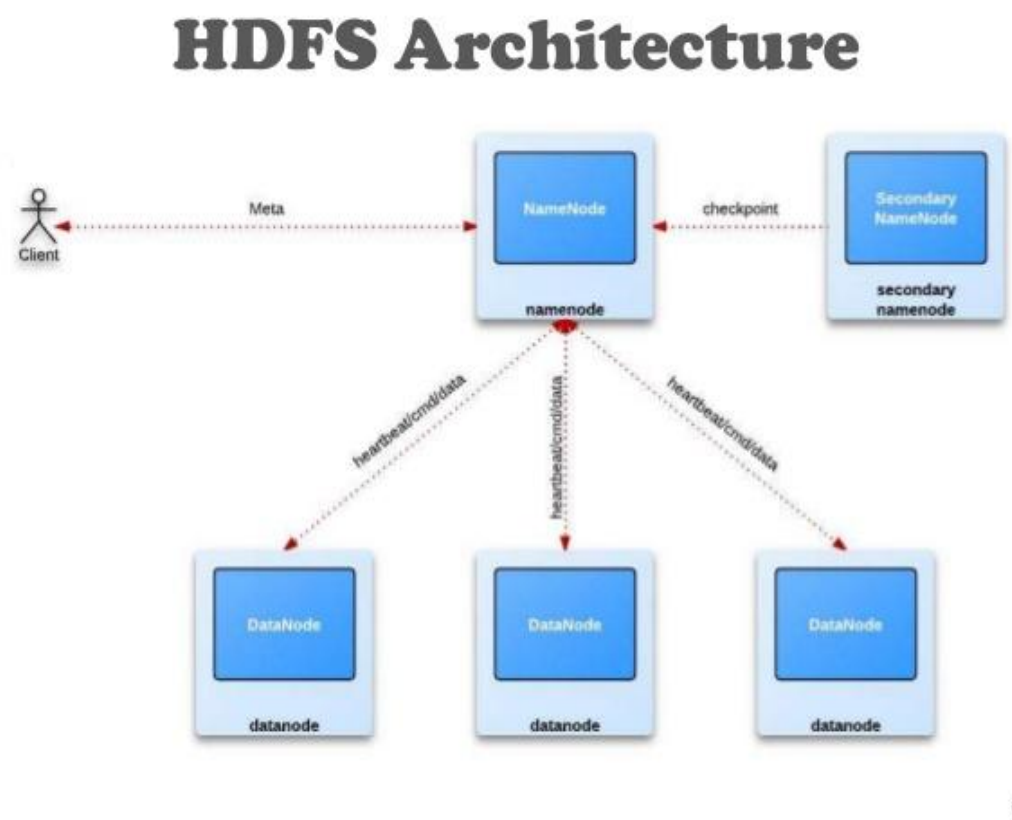


Figure 2.4 –Architecture d'un HDFS

❖ Hadoop MapReduce

MapReduce est le second composant majeur d'**Hadoop** qui gère la répartition et l'exécution des requêtes sur les données stockées par le Cluster.

MapReduce est un cadre modèle et le logiciel de programmation d'abord développé par Google (**MapReduce** papier de Google a présenté en 2004).

Destiné à faciliter et simplifier le traitement des grandes quantités de données en parallèle sur de grandes grappes de matériel de base d'une manière fiable, tolérant aux pannes :

- Pétaoctets de données
- Des milliers de nœuds

Le traitement informatique se produit à la fois:

- Les données non structurées: système de fichiers
- Données structurées: base de données.

Un programme **MapReduce** peut se résumer à deux fonctions **Map ()** et **Reduce ()**. **[18]**

- **La première, MAP**, va transformer les données d'entrée en une série de couples clef /valeur. Elle va regrouper les données en les associant à des clefs, choisies de telle sorte que les couples clef/valeur aient un sens par rapport au problème à résoudre. Par ailleurs, cette opération doit être parallélisable: on doit pouvoir découper les données

d'entrée en plusieurs fragments, et faire exécuter l'opération MAP à chaque machine du cluster sur un fragment distinct. **[18]**

- **La seconde, REDUCE**, va appliquer un traitement à toutes les valeurs de chacune des clefs distinctes produites par l'opération **MAP**. Au terme de l'opération **REDUCE**, on aura un résultat pour chacune des clefs distinctes. Ici, on attribuera à chacune des machines du cluster une des clefs uniques produites par **MAP**, en lui donnant la liste des valeurs associées à la clef. Chacune des machines effectuera alors l'opération **REDUCE** pour cette clef. **[18]**

Considérons l'exemple suivant :

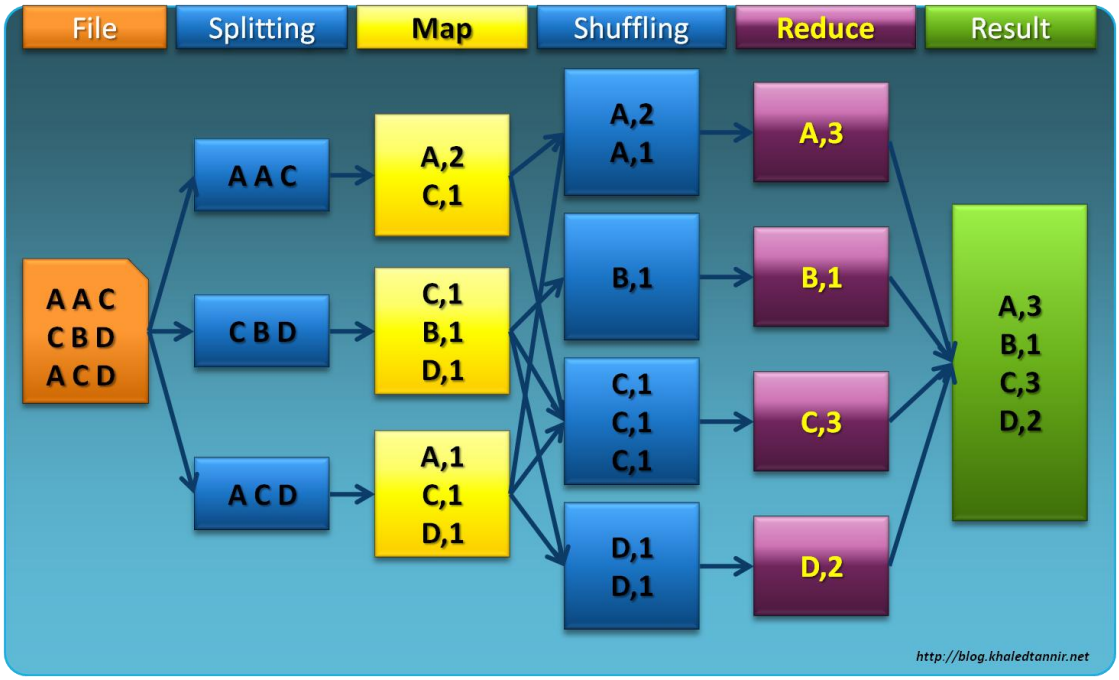


Figure 2.5 –Exemple d'un MapReduce

Dans cet exemple, nous avons un fichier en entrée contenant trois lignes dont chacune contient trois lettres (AAC, CBD, ACD).

La figure illustre les différentes étapes d'exécution de l'algorithme (de gauche à droite). Ces étapes sont :

1. File
2. Splitting
3. Map
4. Shuffling
5. Reduce
6. Result

✓ L'étape File : on lit le fichier en entrée et on initialise les différents 'Workers' MapReduce.

✓ L'étape Splitting : on distribue les données à traiter sur les différents nœuds du cluster de traitement.

- ✓ L'étape Map : on effectue le compte de chacune des lettres et ceci en local sur chaque nœud du cluster de traitement.
- ✓ L'étape Suffling : on regroupe toutes les lettres ainsi que leur compte à partir de tous les nœuds de traitement.
- ✓ L'étape Reduce : on effectue le cumule de toutes les valeurs de chaque lettre.
- ✓ L'étape result : on agrège tous les résultats des différentes étapes Reduce et on retourne le résultat final. **[22]**

➢ Les autres composants de Hadoop

❖ HBase

Apache **Hbase** est une base de données **Hadoop**, distribuée et évolutive Big data orienté colonne. C'est un schéma clairsemé, distribué persistant et multidimensionnel. Le schéma est indexé par une clé d'enregistrement, une clé colonne et un horodatage. Chaque valeur dans ce schéma est une matrice d'octets. **Hbase** utilise un modèle de données très similaire à **BigTable**. Les utilisateurs enregistrent les lignes de données dans des tables labellisés. Un enregistrement de données a une clé triable et un nombre arbitraire de colonnes. La table est enregistrée de manière clairsemée, ainsi la ligne dans la même table peut avoir un nombre très varié de colonne selon les préférences de l'utilisateur. **[23]**

❖ Hive

Hive est un composant **ETL** (**extraction, transformation, chargement**) pour **Hadoop** facilitant la remontée des données, les requêtes ad-hoc et l'analyse de larges ensembles de données avec les systèmes de fichier compatible **Hadoop**. C'est un middleware permettant d'accéder aux données et requêtes utilisant un langage s'apparentant à du SQL appelé HiveQL. Par ailleurs ce langage autorise aussi les programmeurs map/reduce à connecter leur mapping/réduction customisé quand il est possible de l'exprimer dans HiveQL. Il simplifie le requêtage et la gestion de large ensemble de données monté sur du stockage distribué.

Reposant sur l'architecture **Hadoop** il permet l'outillage pour la transformation, l'extraction, et le chargement de données. Il s'adapte à une grande variété de format et permet d'accéder aux fichiers enregistrés en format **HDFS** et dans d'autre système de stockage tels qu'Hbase. Il permet de proposer un simple requêtage de type SQL, appelé HiveQL, ce qui permet les utilisateurs familiers avec Map/Reduce d'être capable de connecter leur map/reduce pour exécuter de rapides requêtes. **[23]**

❖ Oozie

Oozie est un système d'ordonnancement de workflow capable de gérer les jobs **Hadoop**. Les coordinateurs **Oozie** sont des jobs récurrents de workflow déclenchés par le temps (fréquence) et la disponibilité des données.

Oozie est intégré avec le reste de la pile **Hadoop** pour soutenir plusieurs types de jobs Hadoop (comme Java **Map-Reduce**, Streaming **Map-Reduce**, **Pig**, **Hive**) ainsi que des jobs spécifiques du système (telles que des programmes Java et des scripts shell). **[23]**

❖ Zookeeper

Zookeeper est un service centralisé pour maintenir, configurer, informer et proposer une distribution synchronisé et proposer des services groupés. Tous ces types de service sont utilisés sous une forme ou une autre par les applications distribuées. **[23]**

❖ **HCatalog**

HCatalog prend en charge la lecture et l'écriture de fichiers dans tous les formats pour lesquels un Ser De Hive (sérialiseur-désérialiseur) peut être écrit. Par défaut, **HCatalog** soutient RCfile, CSV, JSON, et les formats de fichiers de séquence. Pour utiliser un format personnalisé, vous devez fournir les Input Format, Output Format et Sarde.

HCatalog est construit au-dessus du Metastore **Hive** et intègre des composants de la ruche DDL. **HCatalog** fournit lire et écrire des interfaces pour **Pig** et **MapReduce** et utilise l'interface de ligne de commande Hive pour la délivrance de la définition des données et des commandes d'exploration de métadonnées. Il présente également une interface REST pour permettre des outils externes accès à la ruche DDL (Data Definition Language) opérations, telles que «create table» et «décrire la table".

HCatalog présente une vue relationnelle de données. Les données sont stockées dans des tables et ces tables peuvent être placées dans des bases de données. Les tables peuvent aussi être divisées en une ou plusieurs touches. Pour une valeur donnée d'une clé (ou un ensemble de touches), il y aura une partition qui contient toutes les lignes avec cette valeur (ou un ensemble de valeurs). **[24]**

9. La manipulation des Big Data

En quelques années, le **volume des données** brassées par les entreprises a considérablement augmenté. Émanant de sources diverses (transactions, comportements, réseaux sociaux, géo localisation...), elles sont souvent structurées autour d'un seul point d'entrée, la clé, et susceptibles de croître très rapidement. Autant de caractéristiques qui les rendent très difficiles à traiter avec des outils classiques de gestion de données. Par ailleurs, l'analyse de grands volumes de données, ce qu'on appelle le Big Data, défi e également les moteurs de bases de données traditionnels.

C'est pour répondre à ces différentes problématiques que sont nées les bases de données **NoSQL** (**Not Only SQL**), sous l'impulsion de grands acteurs du Web comme Facebook ou Google, qui les avaient développées à l'origine pour leurs besoins propres. Grâce à leur flexibilité et leur souplesse, ces bases non relationnelles permettent en effet de gérer de gros volumes de données hétérogènes sur un ensemble de serveurs de stockage distribués, avec une capacité de montée en charge très élevée. Elles peuvent aussi fournir des accès de paires clé valeur en mémoire avec une très grande célérité. Réservées jusqu'à peu à une minorité, elles tendent aujourd'hui à se poser en complément du modèle relationnel qui dominait le marché depuis plus de 30 ans.

• NoSQL (Not Only SQL)

NoSQL c'est un terme qui désigne une catégorie de systèmes de gestion de base de données destinés à manipuler des bases de données volumineuses pour des sites de grande audience. Les bases de données **NoSQL** permettent de traiter les données d'une façon distribuée, ces basses de données qui sont scalables. **[19]**

➢ **Les avantages du NoSQL**

✓ Leurs performances ne s'écroulent jamais quel que soit le volume traité. Leur temps de réponse est proportionnel au volume ;

- ✓ Elles se migrent facilement. En effet, contrairement aux **SGBDR** classiques, il n'est pas nécessaire de procéder à une interruption de service pour effectuer le déploiement d'une fonctionnalité impactant les modèles des données ;
- ✓ Elles sont facilement évolutives. A titre d'exemple, le plus gros cluster de **NoSQL** fait 400 To, tandis qu'Oracle sait traiter jusqu'à une vingtaine de Téraoctet (pour des temps de réponse raisonnable). **[19]**

➢ **Les Caractéristiques des bases de données NoSQL**

Afin de garantir l'intégrité des données, la plupart des systèmes de base de données classiques sont basées sur des transactions. Ceci assure la cohérence des données dans toutes les situations de gestion des données.

Ces caractéristiques transactionnelles sont également connues comme ACIDE (atomicité, cohérence, isolement, et longévité).

Cependant, le mesurage hors des systèmes Acide-conformes a montré pour être un problème. Les conflits surgissent entre les différents aspects de facilement disponible dans les systèmes distribués qui ne sont pas entièrement solubles - connu comme théorème de PAC :

i. **Cohérence forte** : tous les clients voient la même version des données, même sur des mises à jour à l'ensemble de données - par exemple au moyen du biphasé commettez le protocole (transactions de XA), et l'ACIDE,

ii. **Facilement disponible :** tous les clients peuvent toujours trouver au moins une copie des données priées, même si certaines des machines dans un groupe sont vers le bas,

iii. **Séparation-tolérance :** tout le système garde sa caractéristique même lorsqu'étant déployé sur différents serveurs, transparents au client. Le Chapeau-théorème postule que seulement deux des trois aspects différents du mesurage sont peuvent être réalisés entièrement en même temps. **[19]**

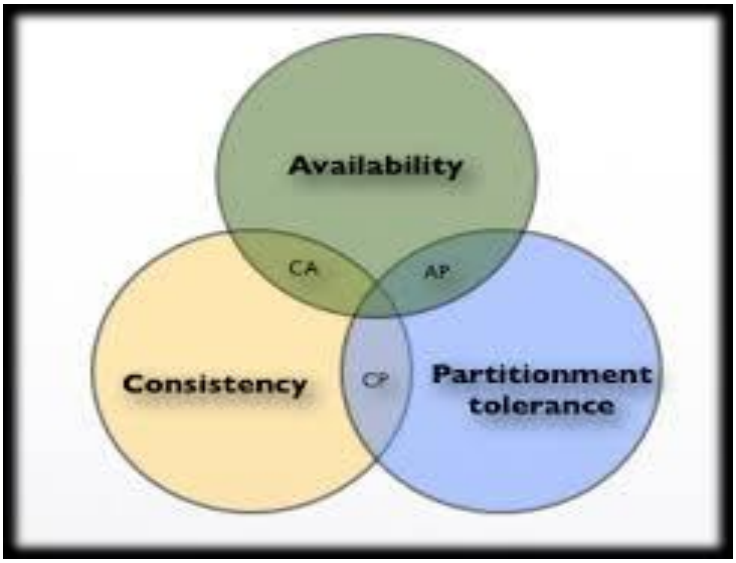


Figure 2.6 –Caractéristiques de base de données NoSQL

- **Les types des bases NoSQL**

On distingue actuellement 4 types de bases **NoSQL**:

1) **Stockage de couples (clé, valeur)**
2) **Bases de documents**
3) **Bases orientées colonnes**
4) **Bases de graphes**

1) Stockage de couples (clé, valeur) :

La base est une table de hachage distribuée. On dispose en général de 4 opérations :

a. **Create :** créer un nouveau couple (clef, valeur). La valeur est n'importe quel objet.
b. **Read :** lire un objet connaissant sa clef
c. **Update :** mettre `a jour l'objet associe à une clef
d. **Delete :** supprimer un objet connaissant sa clef. **[19]**

Car	
Key	Attributes
1	Make: Nissan Model: Pathfinder Color: Green Year: 2003
2	Make: Nissan Model: Pathfinder Color: Blue Color: Green Year:2005 Transmission: Auto

Figure 2.7 –Stockage de couples (clé, valeur)

2) bases de données de documents

Inspiré par Lotus Notes, les bases de données de documents ont été, comme leur nom l'indique, conçus pour gérer et stocker des documents. Ces documents sont encodés dans un format d'échange de données standard tels que XML, JSON (Javascript Option Notation) ou BSON (Binary JSON).

Contrairement aux magasins simples valeurs-clés décrites ci-dessus, la colonne de valeur dans les bases de données de document contient des données semi-structurées - attribut spécifiquement paires nom / valeur. Une seule colonne peut accueillir des centaines de ces attributs, et le nombre et le type d'attributs enregistrés peuvent varier d'une ligne à.

Aussi, contrairement à de simples magasins de valeurs-clés, les clés et les valeurs sont entièrement consultables dans des bases de documents. **[19]**

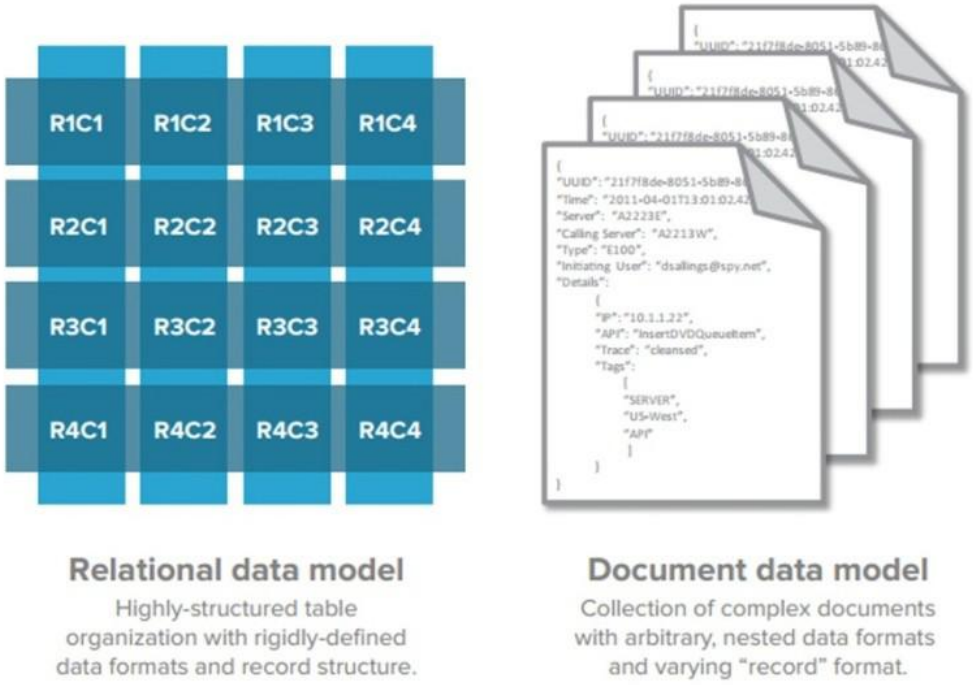


Figure 2.8 –Base de documents

3) Large-colonne (ou Colonne-famille)

Comme des bases de données de document, les magasins de Large-colonne (ou Colonne-famille) (ci-après WC/CF) utilisent une structure de données distribuée et orientée colonne qui adapte à des attributs multiples par clé.

Tandis que les magasins d'un certain WC/CF ont une ADN de Clé-valeur (par exemple, Cassandra inspiré par la dynamo), plus sont modelés après les Bigtable de Google, le système réparti interne Google de stockage de données de Pétaoctets-échelle développé pour sa recherche indexent et d'autres collections comme des finances de Google Earth et de Google. Ceux-ci cadre replient généralement le système de fichiers distribué pas simplement de Google des Bigtable de stockage de données de structure, mais de Google (GFS) et de **MapReduc**e traitement en simultanéité aussi bien, comme cela est le cas pour **Hadoop**, qui comporte le système de fichiers de **Hadoop** (**HDFS**, basé sur GFS) + **Hbase** + **MapReduce**…**[19]**

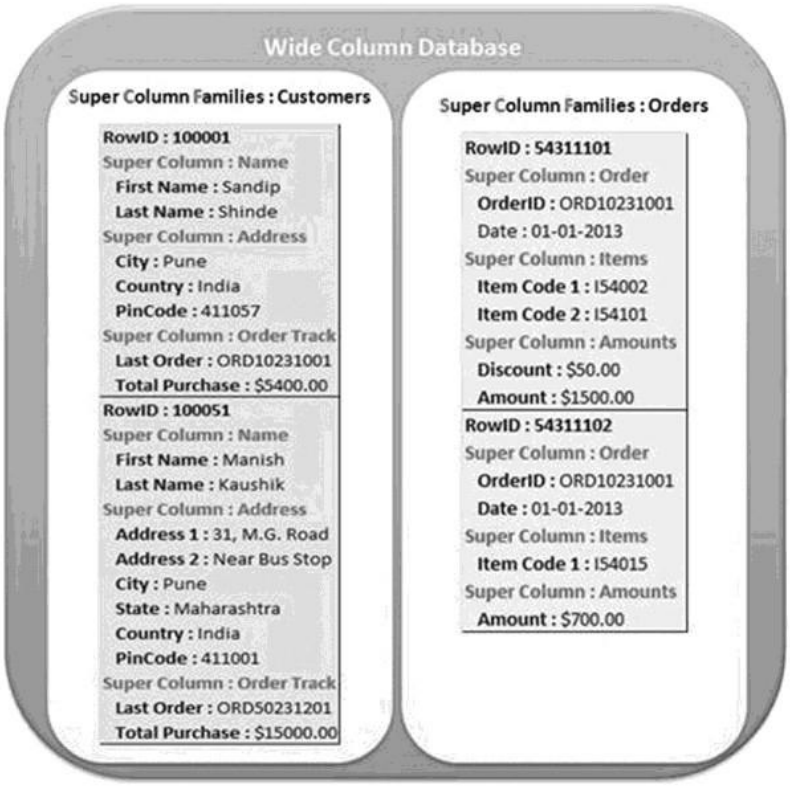


Figure 2.9 –Bases orientées colonnes

4) Bases de données de graphes

Bases de données de graphes remplacent les tables relationnelles avec des graphiques relationnels structurés de paires clé-valeur interconnectés.

Ils sont des bases de données orientées objet à même que les graphiques sont représentés en tant que réseau de nœuds (objets conceptuels), les relations de nœuds orienté objet et les propriétés (attributs d'objets exprimés sous forme de paires clé-valeur). Ils sont le seul des quatre types **NoSQL** discutées ici qui se préoccupent des relations, et leur attention sur la représentation visuelle de l'information rend plus humaine de l'environnement que **NoSQL**. **[19]**

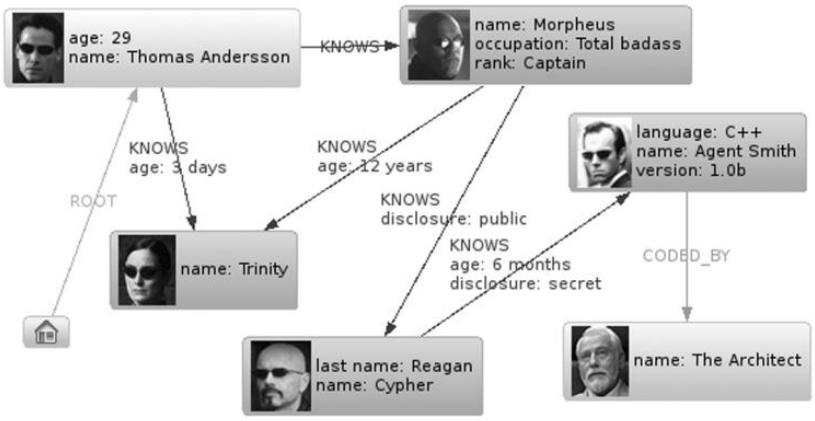


Figure 2.10 –Bases de graphes

- **Les principales bases de données NoSQL**

(A) Cassandra

Cassandra est le projet open source qui découle de la technologie de stockage Facebook, à l'origine il a été écrit spécifiquement pour répondre à la croissance explosive de cette entreprise. Il est assez complexe à configurer, mais il permet d'adresser toutes les situations où la performance et le traitement de la volumétrie est critique. Cassandra est écrite en JAVA.

Il est Bon pour les problématiques de cluster géant et de performance, par contre il est complexe à configurer. Elle est difficilement dissociable d'une structure en cluster à plusieurs nœuds. Par conséquent, Cassandra n'est pas un produit idéal pour commencer dans le NoSQL. **[25]**

(B) mongoDB

La plus populaire des bases **No-SQL** est écrite en C et n'utilise pas de machine virtuelle JAVA. Cette base est idéale pour débuter car elle est à la fois polyvalente et simple. Aussi à l'aise pour le stockage massif de données que pour les développements rapides orienté web. Elle possède également une documentation de premier ordre. **[25]**

10. Conclusion

L'expression **Big Data** Les données peuvent être liées à des définitions multiples, chacun d'eux se concentrant sur une caractéristique différente; Il est important de souligner que Big Data n'identifient une nouvelle technologie, mais un ensemble de technologies et de pratiques déjà existantes qu'une entreprise doit appliquer afin d'améliorer ses connaissances et obtenir plus d'informations sur les produits, les intervenants et les services, donc dans ce chapitre, nous avons présenté les concepts de base de la technologie «Big Data» afin de préciser autant que possible, car il comprend un grand nombre de concepts et d'outils.

L'ingestion rapide, le stockage et le traitement de gros volumes de données nécessite une infrastructure rentable qui peut évoluer avec la quantité de données et la portée de l'analyse. La plupart des organisations avec des systèmes de plates-formes typiquement traditionnelles données relationnelles de gestion de base de données couplés à des entrepôts de données d'entreprise en utilisant **ETL** outils pour **l'intégration** trouvent que leur infrastructure existante est soit techniquement incapable ou financièrement irréalisable pour le stockage et l'analyse de gros volumes de données.

CHAPITRE III

L'intégration des données

Chapitre 3

1. Introduction

L'accroissement des données produites par les entreprises, les particuliers, les scientifiques et les acteurs publics, couplé au développement d'outils informatiques, offre de nouvelles perspectives d'analyses Ces dernières on des répercussions importantes en termes de recherche et développement ou d'amélioration des services et de leur gestion.

Les méthodes traditionnelle **d'intégration** de données ont montre leur limitent, c'est pourquoi des nouvelles technologies ont vu le jour **Hadoop** et **NoSQL…** ont été précurseur dans les concepts qui forment le **Big Data**, mais beaucoup d'autres technologies ont vu le jour par la suite. Ces technologies exploitent de nouvelles idées ou explorent plus encore, des concepts déjà connus. Elles sont le signe d'un nouveau marché qui s'est ouvert et continu à s'ouvrir.

Dans ce chapitre nous allons citer les derniers travaux **d'intégration** en utilisant le processus **ETL** ainsi notre problématique.

2. Intégration des données

L'intégration des données demeure plus que jamais un des plus gros défis dans le monde de **l'intelligence d'affaires**. Bien qu'elle soit une composante charnière de l'environnement de **BI**, l'architecture **d'intégration** est trop souvent négligée et son développement est régulièrement plus ou moins improvisé par des équipes peu coordonnées.

Ce laisser-aller fait grimper en flèche les efforts de développement et les difficultés de maintenance, sans compter les problèmes récurrents de stabilité et de rendement qui peuvent compromettre la disponibilité des données.

Les architectes **d'intégration** et les développeurs **ETL** utilisent systématiquement les meilleures pratiques de l'industrie pour l'**extraction**, la **transformation** et le **chargement** de les données. Ces pratiques sont incluses dans une standardisation stricte qui assure l'uniformité, la qualité et la prédictibilité des solutions implantées, et ce, quels que soient le niveau de complexité et les contraintes de votre environnement. **[31]**

Plusieurs de ces pratiques sont indépendantes d'un environnement spécifique tandis que d'autres tiennent compte des spécificités des solutions technologiques utilisées. Nos équipes de développement et nos pratiques de solutions d'**intégration** vous sont offertes dans les environnements suivants :

- SAP Data Services
- **Microsoft** SSIS
- Datastage
- Informatica. **[31]**

3. Les objectifs d'intégration

Plus particulièrement, l'intégration de données doit fournir

- ✓ Un accès (requêtes, éventuellement mises-à-jour) – uniforme (comme si c'était une seule BD homogène)
- ✓ A des sources (pas seulement des BD)
- ✓ Multiples (déjà deux est un problème)
- ✓ Autonomes (sans affecter leur comportement, indépendant des autres sources ou du système d'intégration)
- ✓ Hétérogènes (différents modèles de données, schémas)
- ✓ Structurées (ou semi-structurées). **[26]**

4. Les Enjeux d'intégration

4.1. Dans l'entreprise

– Données dispersées dans une grande variété de sources **hétérogènes**:

- internes à l'entreprise (protégées)
- externes, chez des fournisseurs, des partenaires ou des clients

– Objectif « **business intégration** »: accès efficace, facile et sûr à ces données

– Études:

- IBM: « pour 1$ dépensé pour une application, 5-9$ sont dépensés pour assurer son **intégration** »
- Gartner: « plus de 40% des budgets IT sont dépensés en **intégration** »
- Morgan Stanley: « **l'intégration de données** est devenue la priorité n°1 des entreprises avant le e-business et le CRM » **[26]**

4.2. Grand public

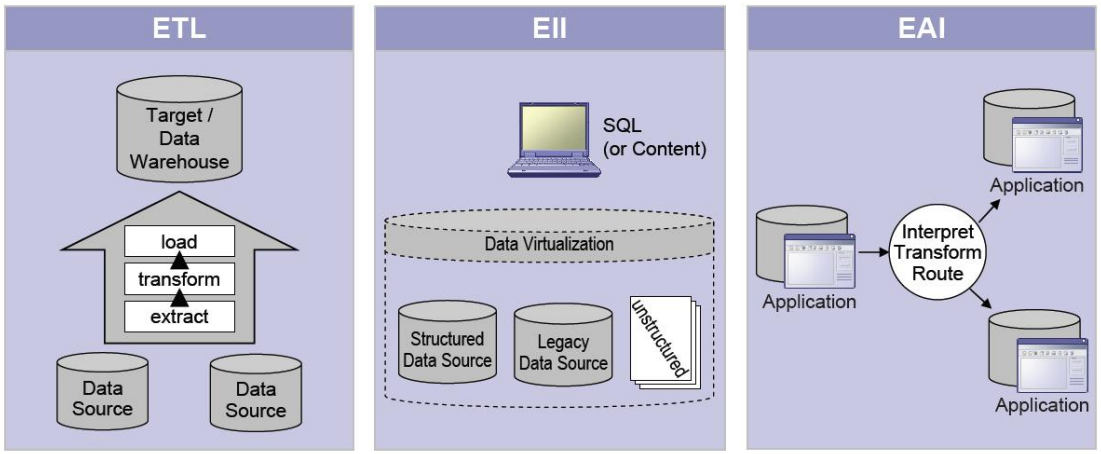
– Accès simple, rapide et efficace aux informations disponibles sur le web

- Texte/HTML, images, vidéo
- XML, fils RSS, cartes
- Le web caché
- Services web

– Commerce électronique: comparateurs de prix, intégration de magasins en ligne. **[26]**

5. Approches d'intégration

Il existe trois approches générales pour l'intégration de données,



Extract, Transform and Load
- Intégration et livraison des données en lot
- Transformations appliquées sur les données

Enterprise Information Intergration
- Fédération de données provenant de plusieurs sources
- Accès temps-réel aux données
- Données structurées ou semi-structurées

Enterprise Application Intergration
- Processus d'intégration des données d'applications
- Basé sur l'échange de messages sur un bus commun

Figure 3.1 –Les approches d'intégration

5.1. Entreprise Information Intégration (EII)

- **Définition**

L'intégration de l'information d'entreprise (EII) est un logiciel qui combine des données et l'information d'entreprise dans une interface unique de surveillance de données où des données sont exprimées par l'intermédiaire de la représentation uniforme. **EII** consolide un grand groupe de points d'émission de données distincts dans une ressource d'utilisateur et de système. **[32]**

EII s'est développé comme industrie, mais n'a pas encore atteint son potentiel maximum.

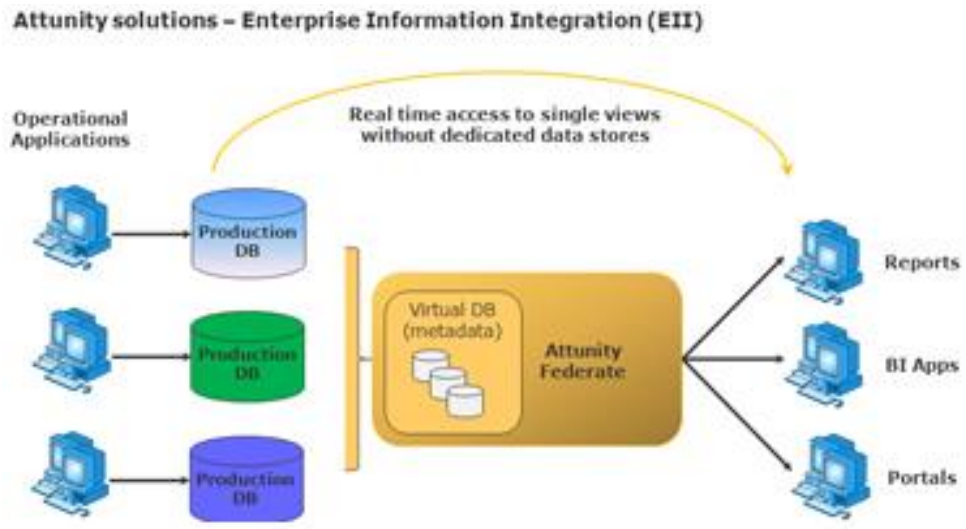


Figure 3.2 –Enterprise Information Integration (EII)

- **Caractéristiques:**

– Fournit une vue unifiée des données de l'entreprise, où les sources de données forment une fédération.

– Les sources de données dispersées sont consolidées à l'aide d'une BD virtuelle, de manière transparente aux applications utilisant ces données.

– Toute requête à la BD virtuelle est décomposée en sous-requêtes aux sources respectives, dont les réponses sont assemblées en un résultat unifié et consolidé.

– Permet de consolider uniquement les données utilisées, au moment où elles sont utilisées (source data pulling).

– Le traitement en-ligne des données peut cependant entraîner des délais importants. **[5]**

- **Avantages:**

– Accès relationnel à des sources non-relationnelles;

– Permet d'explorer les données avec la création du modèle de l'entrepôt de données.

– Accélère le déploiement de la solution.

– Peut être réutilisé par le système **ETL** dans une itération future.

– Aucun déplacement de données. **[5]**

- **Inconvénients:**

– Requiert la correspondance des clés d'une source à l'autre.

– Consolidation des données plus complexe que dans l'**ETL**.

– Surtaxe les systèmes sources.

– Plus limité que l'**ETL** dans la quantité de données pouvant être traitée.

– **Transformations** limitées sur les données.

– Peut consommer une grande bande passante du réseau. **[5]**

5.2. Entreprise Application Intégration (EAI)

- **Définition**

EAI (**intégration d'applications d'entreprise**) est un terme informatique d'entreprise pour les plans, les méthodes et les outils visant à moderniser, consolider et coordonner les-applications informatiques dans une entreprise. **[33]**

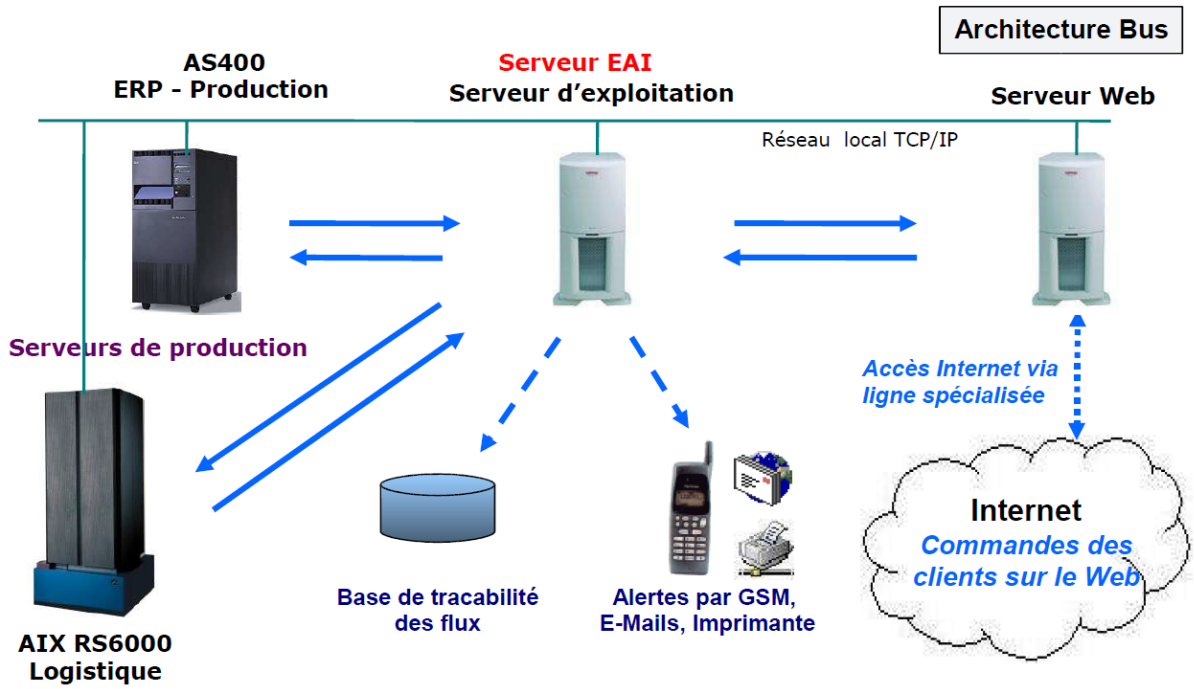


Figure 3.3 –Architecture (EAI) Exemple

- **Caractéristiques:**

– Approche permettant de fournir à l'entrepôt des données provenant des sources (source data pushing).

– Repose sur l'intégration et le partage des fonctionnalités des applications sources à l'aide d'une architecture SOA.

– Généralement utilisé en temps réel ou en semi-temps réel (NearReal Time).

– L'**EAI** ne remplace pas le processus **ETL**, mais permet de simplifier ce dernier. **[5]**

- **Avantages:**

– Facilite l'interopérabilité des applications.

– Permet l'accès en (quasi) temps-réel.

– Ne transfère que les données nécessaires.

– Contrôle du flot de l'information. **[5]**

- **Inconvénients:**

– Support limité aux transformations et agrégations des données.

– Taille des transactions limitée (en nombre de lignes).

– Développement complexe.

– Gestion complexe de l'intégrité sémantique des données (e.g.règles d'affaires).

– Utilise la bande passante du réseau durant les heures de pointe. **[5]**

5.3. Extract, Transform and Load (ETL)

(Il est défini dans le chapitre 1)

Figure 3.4 –Architecture ETL

• **Caractéristiques:**

– Permet la consolidation des données à l'aide des trois opérations suivantes:

• **Extraction:** identifier et extraire les données de sources ayant subi une modification depuis la dernière exécution.

• **Transformation**: appliquer diverses transformations aux données pour les nettoyer, **les intégrer** et les agréger.

• **Chargement**: insérer les données transformées dans **l'entrepôt** et gérer les changements aux données existantes.

– Traite normalement de grandes quantités de données en lots cédulés.

– Est surtout utilisé avec les entrepôts de données et les comptoirs de données. **[5]**

• **Avantages:**

– Optimisé pour la structure de **l'entrepôt de données**.

– Peut traiter de grandes quantités de données dans une même exécution (traitement en lot).

– Permet des transformations complexes et agrégations sur les données.

– La cédule d'exécution peut être contrôlée par l'administrateur.

– La disponibilité d'outils GUI sur le marché permet d'améliorer la productivité.

– Permet la réutilisation des processus et transformations (ex:packages dans SSIS). **[5]**

• **Inconvénients:**

– Processus de développement long et coûteux.

– Gestion des changements nécessaire.

– Exige de l'espace disque pour effectuer les transformations (staging area).

– Exécuté indépendamment du besoin réel.

– Latence des données entre la source et l'entrepôt.

– Unidirectionnel (des sources vers l'entrepôt de données). **[5]**

5.4. Comparaison entre les approches d'intégration

	ETL	EII	EAI
Flot de données	Unidirectionnel (sources à l'entrepôt)	Bidirectionnel	Bidirectionnel
Mouvement De données	Lots cédulés	Au Moment de la requête	Déclenché par la transaction
Latence	Journalier à mensuel	Temps-réel	Quasi temps-réel
Transformations/ agrégations des données	Grande capacité	Moyenne capacité	Faible capacité
Volume Des données	Grand (millions Ou milliards de lignes)	Moyen (10,000– 1,000,000 de lignes)	Petit (100-1000 lignes)

Tableau3.1 : Comparaison entre les approches d'intégration

- **Exemples de produits commerciaux**

• Outils ETL:

– Oracle Warehouse Builder;
– IBM Infosphere Information Server;
– **Microsoft** SQL Server Integration Services (SSIS);
– SAS Data Integration Studio. **[5]**

• Outils EAI:

– IBM WebSphere Message Broker;
– Microsoft BizTalk Server;
– Oracle SOA Suite. **[5]**

• Outils EII:

– SAP BusinessObjects Data Federator;
– IBM WebSphere Federation Server. **[5]**

6. Les étapes d'intégration avec l'ETL

Nous distinguons deux niveaux dans la construction des **entrepôts de données**. Le premier niveau correspond à la construction des sources de données opérationnelles, et de l'entrepôt de données global. Le second niveau englobe tous les entrepôts de données locaux. La raison de cette distinction est, qu'à chaque niveau, sont associées différentes étapes de traitement et différentes difficultés techniques.

Au premier niveau, le processus de construction est décomposé en quatre étapes principales, qui sont : (1) **l'extraction** des données des sources de données opérationnelles,

(2) **la transformation** des données aux niveaux structurel et sémantique, (3) **Chargement** des données, et (4) **le stockage** des données intégrées dans le système cible.

La figure suivante résume l'enchaînement de ces étapes de traitement.

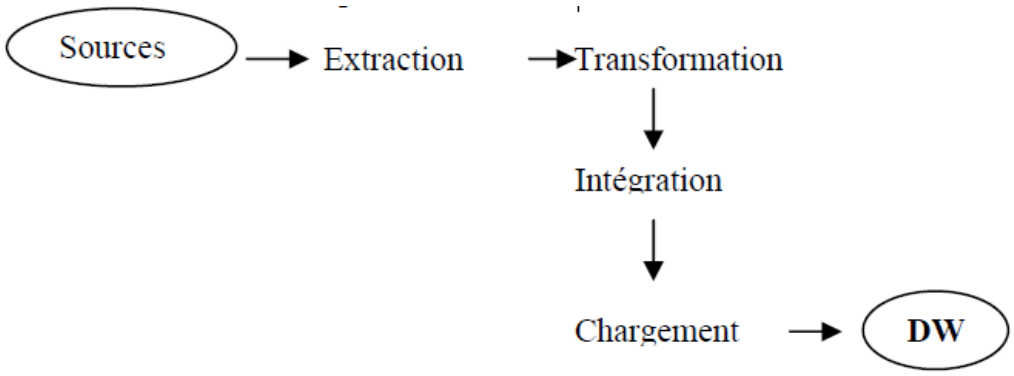


Figure 3.5 –Etapes de traitement du premier niveau de construction d'un entrepôt de données

Notez cependant que cette décomposition est seulement logique.

L'étape **d'extraction** et une partie de l'étape de **transformation** peuvent être groupées dans le même composant logiciel, tel qu'un « wrapper » ou un outil de migration de données. L'étape d'**intégration** est souvent couplée avec des possibilités de transformation de données riches dans un même composant logiciel, qui, habituellement, réalise **le chargement** dans **l'entrepôt de données**. **[27]**

Toutes les étapes de traitement peuvent aussi être groupées dans un même logiciel, comme par exemple un système multi base. Quand les étapes **d'extraction** et **d'intégration** sont séparées, les données nécessitent d'être stockées entre les deux. Ceci peut être fait en utilisant un média par source ou un média pour toutes les sources. Une vue opérationnelle typique de ces composants est donnée par la figure suivante.

Les composants logiciels sont représentés par des rectangles. Les ellipses désignent des stockages intermédiaires des résultats de l'étape **d'extraction/transformation**. Toutes les données qui sont en entrée du composant intégration utilisent le même modèle de représentation de données. Finalement, un « wrapper » est associé à chaque source, fournissant ainsi une interface API à la source.

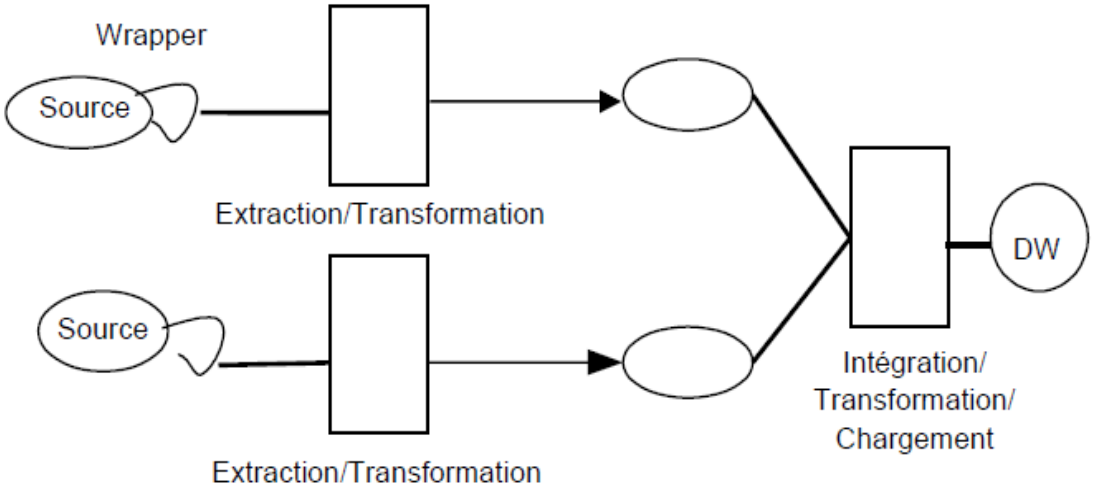


Figure 3.6 –Vue opérationnelle des composants utilisés pour la construction d'entrepôts de données

Au second niveau, le processus de construction comporte trois étapes distinctes, qui sont : (1) **l'extraction de données** à partir d'une base de données (**entrepôt de données** local ou global), (2) **le calcul des données** dérivées pour l'entrepôt de données local cible, et (3) **le stockage** des résultats dans l'entrepôt de données local.

L'étape d'extraction est un cas particulier de celle du premier niveau car les données de **l'entrepôt** sont stockées dans une base de données. A l'opposé, dans le premier niveau, **l'extraction** peut concerner des sources de données arbitraires, comme des fichiers par exemple. Le calcul des données dérivées est assez spécifique car il peut impliquer des requêtes complexes avec agrégats. **[27]**

7. Les types d'intégration

Le type **d'intégration** réalisé dans la conception d'un **entrepôt de données** est celui que l'on réalise dans le domaine de l'intégration d'information, qui a été exploré dans différents domaines comme :

- les bases de données,
- les systèmes d'information coopératifs,
- les systèmes d'information globaux,
- la représentation des connaissances.

Une première classification des différentes approches repose sur le contexte **d'intégration**, et par conséquent, le type des entrées/sorties du **processus d'intégration**, et le but du processus lui-même. Nous distinguons **l'intégration** de schémas, **l'intégration** de données virtuelle, et **l'intégration** de données matérialisée.

7.1. Intégration de schémas: Dans ce cas, l'entrée de **l'intégration** est un ensemble de schémas sources, et la sortie est un schéma de données correspondant à la représentation intensionnelle réconciliée de tous les schémas en entrée. L'entrée comporte également la spécification de la façon d'associer les schémas des données sources à des parties du schéma résultant (cible).

7.2. Intégration de données virtuelle (médiateurs):L'entrée est un ensemble de **données sources**, et la sortie est une spécification décrivant la façon de fournir un accès global et unifié aux sources dans le but de satisfaire certains besoins en information, sans interférer avec l'autonomie des sources.

7.3. Intégration de données matérialisée: Comme dans le cas précédent, l'entrée est un ensemble de **données sources**, mais ici la sortie est un ensemble de données représentant une vue réconciliée des sources, à la fois au niveau intensionnel et au niveau extensionnel. **[27]**

8. Les techniques d'intégration des données massives

L'importance de la **grande intégration** des données a conduit à une quantité importante de la recherche au cours des dernières années sur les thèmes de la cartographie de schéma, couplage d'enregistrements et de fusion de données pour faire face aux nouveaux défis auxquels sont confrontés les **grands intégration** de données. Nous allons présenter un résumé de ces techniques :

8.1. BDI : Cartographie de schéma

Le schéma traçant dans un système **d'intégration** de données se rapporte

i. réant un schéma (global) négocié,

ii. identifiant les tracés entre le schéma (global) négocié et les schémas locaux des points d'émission de données pour déterminer ce que (des ensembles de) attribue contiennent la même information. Les efforts tôt en **intégrant** un grand nombre de sources ont impliqué d'intégrer des données du Web profond. D'après les derniers travaux **(Xin Luna Dong, 2013)**, ils ont proposé deux types de solutions. Le premier est de construire des tracés entre les formes de Web (interfaces pour questionner le Web profond) en tant que des moyens de répondre à une question de Web au-dessus de toutes les sources profondes de Web. Le deuxième est de ramper et d'indexer les données profondes de Web. Des efforts plus récents incluent des données structurées d'extraction et d'intégration des tables de Web et des listes de Web.

Le nombre de sources augmente également la variété des données. Les systèmes traditionnels d'**intégration** de données exigent un schéma significatif traçant l'effort avant que le système puisse être employé, est ainsi évidemment infaisable quand **l'hétérogénéité** est à l'échelle de BDI. L'idée fondamentale des systèmes de dataspace est de fournir des services de meilleur-effort tels que la recherche par mot-clé simple au-dessus des points d'émission de données disponibles au début, et évolue graduellement des tracés de schéma et améliore la qualité de recherche au fil du temps. **[28]**

8.2. BDI: couplage d'enregistrements

Le couplage d'enregistrements se réfère à la tâche d'identifier les dossiers qui se rapportent à la même entité logique entre les différentes sources de données, surtout quand ils ne partagent pas un identifiant commun à travers les sources de données. Il a toujours mis l'accent sur le lien entre un ensemble statique d'enregistrements structurés qui ont le même schéma.

i. les points d'émission de données tendent à être **hétérogènes** en leur structure et beaucoup de sources (par exemple, bips, courriers de blog) fournissent des données non structurées des textes,

ii. les points d'émission de données sont dynamiques et sans interruption évoluant. Ces caractéristiques font le couplage d'enregistrements contestant en particulier dans BDI. Quand il y a un grand nombre de sources et un de large volume de données, les approches traditionnelles de couplage d'enregistrements deviennent inefficaces et inefficaces dans la pratique. Pour adresser la dimension de volume, **(Xin Luna Dong, 2013)**, ils ont proposé de nouvelles techniques pour permettre au couplage d'enregistrements parallèle utilisant **MapReduce**. Celles-ci incluent les techniques pour le blocage adaptatif et les techniques qui équilibrent la charge parmi différents nœuds.

Quand les points d'émission de données sont dynamiques et sans interruption évolution, l'application de couplage d'enregistrements à partir de zéro pour chaque mise à jour devient exorbitante. Pour aborder l'aspect de vitesse, on a proposé des techniques de groupement par accroissement. **[28]**

8.3. BDI: Fusion de données

La fusion de données se réfère à la résolution des conflits à partir d'une collection de sources et de trouver la vérité qui reflète le monde réel.

Contrairement à la cartographie de schéma et couplage d'enregistrements, la fusion de données est un nouveau domaine qui a émergé récemment. Sa motivation est exactement la véracité des données: le Web a rendu facile de publier et diffuser de fausses informations sur de multiples sources et il est donc essentiel de séparer le grain de l'ivraie pour présenter des données de haute qualité.

Pour répondre à ces défis de véracité liés, les techniques ont été proposées pour trouver les vérités simples ou multiples de valeurs contradictoires. Ces techniques ont été étendues pour gérer le volume des données (fusion de données en ligne), la vitesse des données (découverte de la vérité pour les données dynamiques), et la variété des données (combinant couplage d'enregistrements et de fusion de données). **[28]**

9. Les derniers travaux d'intégration

Base de connaissances Construction via Collaboration de masse:

Notre travail a été inspiré par plusieurs travaux récents qui tentent de tirer parti du volume important des utilisateurs des **Big Data** pour construire des bases de connaissances et de données de soutien technique **(Richardson & Domingos 2003)**;**(Richardson, Aggarwal, et Domingos 2003)**, L'idée de base de ces travaux est d'avoir les utilisateurs contribuent faits et des règles dans un langage spécifié.

Construction d'intégration de données Systems:

La construction manuelle et l'entretien des **systèmes d'intégration** de données est très intensive en main et sujette aux erreurs. Il y a eu de nombreux travaux sur la réduction des coûts de main-d'œuvre des tâches spécifiques pendant le processus de construction, tels que le schéma correspondant **(Rahm et Bernstein, 2001)** et la construction d'emballage (par exemple, **(Kushmerick, Weld, & Doorenbos 1997)**;**(Ashish & Knoblock 1997**), mais peu

de travaux sur un effort systématique pour aborder la réduction des coûts pour l'ensemble du processus, à l'exception de **(Rosenthal et al 2001)**; **(Rosenthal Seligman 2001)**.

Schéma correspondant:

De nombreux travaux ont été menés sur le schéma correspondant, un problème fondamental dans l'intégration des schémas. Certains travaux récents comprennent **(Milo & Zohar 1998)**; **(Palopoli, Sacca, & Ursino 1998)**;**(Li & Clifton 2000; Madhavan, Bernstein, & Rahm 2001)**; **(Doan, Domingos, & Halevy 2001)**; **(Yan et al 2001)**; **(Kang & Naughton 2003)**; **(He & Chang 2003)**; **(Madhavan et al 2003)**; **(Rahm et Bernstein, 2001**). **[29]**

Ces travaux emploient des règles travaillés manuellement et techniques d'apprentissage, avec une certaine interaction humaine limitée, machine pour découvrir les correspondances sémantiques.

L'intégration des données massives avec l'approche ETL :

L'une des premières contributions dans le domaine **d'intégration** avec **l'ETL** est celle de **Vassiliadis et al. (2002)** ; Il s'agit d'une approche de modélisation basée sur un formalisme graphique non standard ; ARKTOS II étant le prototype mis en œuvre. **Trujillo et Luján-Mora (2003)** se sont intéressés, eux aussi, à la modélisation de **l'ETL** dans une approche plus globale basée sur des choix standards, **Le travail de Liu et al. (2011)** s'intéresse aux performances de l'**ETL** face au **Big Data** et adopte le modèle **MR**. Cette approche a été implémentée dans un prototype appelé **ETLMR**, version **MR** du prototype **PygramETL(cf. Thomson et Pederson, 2009a)**.**Bala et Alimazighi (2013)** ont proposé une modélisation de l'**ETL** pour le **Big Data** selon le paradigme **MR** en adoptant le formalisme de **Vassiliadis et al. (2002)** enrichi par des notations graphiques pour modéliser les spécificités du modèle **MR**. **Oliveira et Belo (2013)** ont proposé une approche de modélisation qui consiste en une vue résumée du processus **ETL** et adopte le modèle Reo **(cf. Arbab, 2003)**, Récemment,**(Myriam Lamolle, Amar Zerdazi, 2005)**ont proposé une approche **d'intégration** de données **hétérogènes** avec la définition d'un modèle de médiation structurelle et sémantique, pour l'extraction et la modélisation des connaissances à partir de données sources, appelé hyper**schéma XML**.

10. Intégration et analyse de données en temps réel

Les pressions résultant des demandes des clients et de la compétitivité liée à la nouvelle économie ont créé une demande insatiable pour une **intégration** et analyse, en temps réel, de l'information. Il n'est plus acceptable pour les décideurs de prendre des décisions en se basant sur des données datant de plus d'une semaine, voir même d'une journée. Les employés, les décideurs, les clients et tous les partenaires économiques ont besoin d'accéder à l'information quand elle est pertinente.

La possibilité d'accéder à temps et de façon simple à des données pertinentes au moyen d'outils d'interrogation et d'analyse est fondamentale pour les organisations qui souhaitent être compétitives. Cependant, avec la prolifération d'environnements **hétérogènes** qui doivent être intégrés à des systèmes d'aide à la décision, à des **entrepôts de données**, etc., les défis sont nombreux. Les données clients, données financières, données de navigations – constituent un avantage considérable sous réserve qu'elles soient intégrées et utilisées pour

faciliter les échanges entre partenaires économiques. Une solution au problème de **l'intégration de données en temps réel** constituera une étape importante vers l'exploitation effective des possibilités de l'Internet dans le domaine de l'aide à la décision.

Le traitement et **l'intégration** de gros volumes de données sur le Web posent des problèmes épineux comme le montrent les résultats de tests effectués sur un Pentium III, 700 MHz, 1 Go Ram et 100 Mbit Ethernet.

Ainsi, dans le cas du WebHouse par exemple, le problème majeur reste celui de concevoir et de développer des agrégateurs incrémentaux efficaces. Des solutions à ce problème **d'intégration de données** pourraient conduire à terme à unifier proprement les différents services. **[27]**

11. Les problèmes d'intégration des données massives

Les entrepôts de données ont été conçus pour l'aide à la décision. Ils intègrent les informations en provenance des différents systèmes transactionnels. L'ensemble des données, y compris leur historique, est utilisé pour faire des calculs prévisionnels, des statistiques ou pour établir des stratégies de développement et d'analyses des tendances.

L'intégration de bases de données massives de ces systèmes est un problème complexe**.** C'est pourtant une tâche que les entreprises peuvent difficilement éviter si elles veulent mettre en route de nouvelles applications ou réorganiser le système d'information existant pour une meilleure productivité. En effet, plusieurs problèmes doivent être pris en compte pendant la conception de systèmes d'intégration. Ces problèmes résultent **de l'hétérogénéité** des données. **[30]**

Cette **hétérogénéité** est dûe au fait que les sources de données ont été conçues indépendamment par des concepteurs différents, ceci explique le fait que les données relatives à un même sujet sont représentées différemment sur des systèmes d'information distincts. Cette **hétérogénéité** provient des choix différents qui sont faits pour représenter des faits du monde réel dans un format informatique. En effet, les données des sources sont structurellement indépendantes mais sont toujours supposées relever de domaines similaires.

L'intégration de données massives a conduit à de nombreux problèmes :

11.1. Grands volumes de données

Plus les utilisateurs professionnels d'aujourd'hui veulent avoir accès à des données plus granulaires (par exemple, les transactions) et plus d'années d'histoire à travers les zones les plus soumises que jamais auparavant. Sans surprise, les entrepôts de données explosent dans la taille et la portée. Données Terabyte entrepôts une rareté il y a plusieurs années sont maintenant monnaie courante fée. Les plus grands entrepôts de données dépassent 50 téraoctets de données brutes. **[30]**

11.2. Hétérogénéité structurelle ou schématique

Elle provient quand les sources adoptent différents modèles de données, structures de données ou schémas, par exemple les modèles de bases de données relationnelles ou orientées objets. De nombreux travaux concernant ce type **d'hétérogénéité** ont été proposés dans les contextes des bases de données fédérées et des multi-bases de données.

11.3. Hétérogénéité sémantique

Elle est dûe aux conflits **sémantiques dans les termes**, les expressions, etc., qui sont adoptés par différents schémas de données mais exprimés de diverses manières. Autrement dit, elle est dûe aux différentes interprétations pour les objets du monde réel. En effet, les sources de données ont été conçues indépendamment par des concepteurs différents ayant des objectifs applicatifs différents. Chacun peut donc avoir un point de vue différent sur le même concept. L'interopérabilité sémantique de données présente un défi majeur dans le processus d'´elaboration des systèmes d'intégration. **[30]**

Avant de passer au quatrième chapitre, il est important de définir la nature du problème à résoudre.

12. Notre problématique

Le développement exponentiel d'échanges d'informations a mis à jour les difficultés pour retrouver l'information pertinente désirée par un utilisateur final En effet, les informations sont représentées et stockées dans une multitude de sources de données distribués et cela de façon très hétérogène. Les sources de données ont pour but principal de fournir des capacités d'interrogation et non d'exportation des données.

- Comment peut-on **intégrer** cette Sources ?
- Comment peut-on **extraire** et **transformer** cette Sources ?
- Comment **extraire** et **transformer** les données pour faire l'étape **d'intégration** plus facile et plus fiable ?

13. Conclusion

Les données ne sont d'aucune utilité si aucun être humain ne peut en tirer parti ou si elles ne sont pas exploitées dans un système automatisé conçu par des êtres humains. **L'intégration** du **Big Data** vise à simplifier autant que possible l'accès aux données, leur compréhension et leur exploitation.

Les fruits d'une intégration réussie du **Big Data** sont les avantages issus de l'utilisation des données. Réduction des délais, élimination des goulots d'étranglement dus au manque de compétences et fluidité des interactions permettent aux entreprises de gagner en rapidité et en efficacité.

CHAPITRE V

L'intégration des Big Data dans le processus ETL

Chapitre 4

1. Introduction

Le développement exponentiel de transformation et de stockage d'informations a mis à jour les difficultés pour retrouver l'information à cherchée par un utilisateur. Ces informations sont représentées et stockées dans plusieurs sources de données distribué d'une façon très **hétérogène**.

Nous proposons dans cette **contribution** une implémentation de l'intégration de schémas **XML** basé la sur le modèle [Myriam Lamolle, AmarZedazii] (**Intégration** de Bases de données hétérogènes par une modélisation conceptuelle **XML)** en créant un modèle d'intégration des **Big data** avec le processus **ETL** basé sur un schéma **XML** sources des bases de données à fédérer**.**

2. Technologies XML

XML, qui signifie *extensible Markup Language* (traduisible par langage de balisage extensible, ou encore langage **à** balises extensible), est un langage constitue de balises, ou de *tags.*

Mis au point par le ***XML Working Grou*psous** la direction de la *World Wide Web Consortium (W3C)* des **1996,**il est destiné **à** la structuration des documents, En **1998,**les spécifications ***XML*** sont **devenues** des recommandations ***XML*** dérive du *SGML*, défini en **1986** par le standard *ISO 8879.*La recommandation du *W3C*indique d'ailleurs que le but de ***XML*** est de permettre au *SGML* générique d'être transmis, reçu et traite sur le Web, Le but du *SGML* était de dissocier complètement le contenu du document de sa présentation, tout en lui ajoutant une structure de description ne laissant aucune ambiguïté aux éléments de contenu. Par exemple :

<description> un exemple </description> ne décrit que ce que représente le texte, c'est-à-dire, qu'il s'agit d'une description sans aucun souci de présentation.

Bien que ces langages soient proches, ***XML*** est un langages beaucoup plus puissant et diversifie dans ses applications, en **effet,** les balises *HTML* sont définies et arrêtées par le *W3C,* alors que celles du ***XML*** sont limitées par le champ d'application qu'on pourrait y trouver, II nous appartient de créer nos balises, en fait, ***XML*** peut être vu comme un métalangage, Du fait de son extensibilité, ***XML*** possède la capacité de décrire n'importe quel domaine de données dont on saura la structure et le vocabulaire **à** employer,

Pour résume, ***XML*** permet de répondre a un tas de critères :

- ✓ ***XML*** peut être utilisé sans difficulté sur Internet.
- ✓ II soutient une grande variété d'applications.
- ✓ II facilite l'écriture de programmes traitants des documents ***XML*.**
- ✓ Les documents ***XML*** sont lisibles par **l'homme** et raisonnablement claires.
- ✓ La conception du ***XML*** est rapidement exécutable.
- ✓ La facilite de créer des documents ***XML***.

- ➢ Le langage peut s'auto d'écrire ;
- ➢ Le langage peut être étendu par lui-même : Les langages dérives du ***XML*** sont définis en ***XML***, ce qui permet de gérer tous les domaines d'application;

- Le contenu d'un document peut être lisible dans n'importe quel éditeur de texte et être compréhensible par une personne **n'ayan**t aucune connaissance particulière;
- Le langage permet la description arborescente des données afin d'apporter par l'intermédiaire des arbres des structures de données effacées;
- Toute application munie d'un parseur texte peut utiliser le document ***XML***.

3. *XML* et les bases de données

XML est en passe de **devenir** le standard utilisé systématiquement par les entre- prises pour échanger des données dans des documents structures, que ce soit en interne, **avec** des partenaires commerciaux ou dans des applications accessibles **à** tous sur internet, Toutefois, ces données sont **souvent** stockées dans un format de voltigeur autre que le ***XML*** par exemple, dans une base de données relationnelle, II faut donc mettre en place un processus de conversion des données du format actuel de la base en documents ***XML***, puis du ***XML*** au format utilise pour le traitement des données, cela se traduit par un plus long traitement des données,

Des outils sont disponibles **avec** les bases de données relationnelles pour traiter ce genre de tache, comme *Oracle9i* et *IBM DB2*. Ils **peuvent** traduire en ***XML*** les données, structures ou non, Toutefois, si ces outils sont utilisés pour traduire des données en direct lors d'un échange ***XML***, cela risque d'augmenter les temps de traitement des transactions ***XML*** et de ralentir également les autres applications qui ascendant **a** la base de données relationnelles.

4. Les bases de données XML basées sur un modèle

Plutôt que de stocker un document ***XML*** en tant que texte, ces bases de données construisent un modèle objet interne du document et stockent ce modèle. La manière dont le modèle est stocke dépend de la base. Certains produits stockent le modèle dans une base relationnelle ou orientée objet.

D'autres bases utilisent un format de stockage propriétaire adapte **à** leur modèle, Les bases ***XML*** natives basées sur un modèle et construites sur d'autres bases possèdent **vrais semblablem**ent des performances similaires **à** ces bases sous-jacentes lors de la recherche des documents, pour la raison évidente qu'elles reposent sur ces systèmes pour **retrouver** les données.

5. Data Warehouse et XML

Les systèmes de gestion de données (*SGBD*) ***XML*** actuels présentent globalement des performances nettement inférieures à celles des *SGBD* relationnels, Cependant, stocker des données ***XML***, notamment si dies présentent des structures complexes et irrégulières, dans une base relationnelle **n'est** pas trivial, Leur interrogation nécessite également des opérations couteuses pour traduire les requêtes du langage *XQuery* en *SQL*, puis les résultats obtenus sous forme de relation en documents ***XML***.

Pour qu'une entreprise sache si doit mettre en place une base de données ***XML*** native, doit réfléchir **à** celles, parmi ses sources de données, qui peuvent nécessiter une traduction ***XML***.

Plusieurs sources de données **a** utiliser des applications ***XML***, une base de données ***XML*** native est intéressante, Par ailleurs, die permettra de réduire l'influence du traitement ***XML*** sur les autres applications si vous **avez** beaucoup de données a traiter, Les bases de données ***XML*** **intégrant** des interfaces de programmations d'application (API) que peuvent être utilisées pour **intégrer** les données ***XML*** **avec** les applications,

Les bases de données ***XML*** natives permettent d'accéder aux sources de données (base de données relationnelle ou données dans le système de fielders par exemple) en utilisant les méthodes standardises d'accès aux bases de données, comme *JDBC* et *ODBC*.

En éclair, pratiquement toutes les bases de données, de *File Maker*, *MySQL*, et *Post-grésil* **à** *Oracle*, *DB*Set *Sybase*, sont accessibles en source ou en cible pour les traductions ***XML***.

Les bases de données ***XML*** natives disponibles aujourd'hui en sont encore **à** leurs débuts. La plupart des utilisateurs qui ont déjà adopté ces technologies travaillent dans les secteurs de la finance ou de la production, et utilisent principalement les bases de données ***XML*** natives pour accélérer les vitesses de transaction.

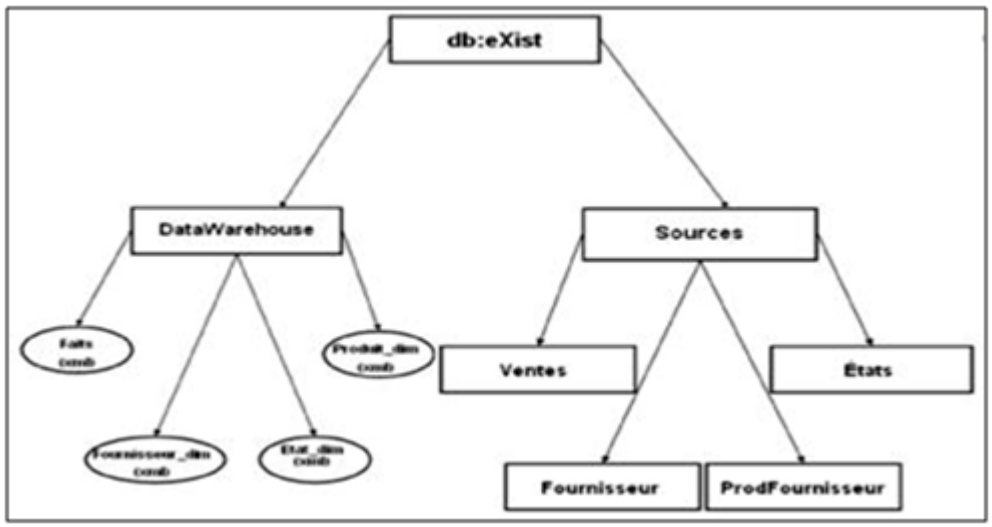


Figure 4.1 – Schéma du système de stockage basée sur XML

6. XSLT (eXtensible Stylesheet Language Transformations)

XSLT ou eXtensible Stylesheet Language Transformations est une technologie qui permet de transformer les informations d'un document XML vers un autre type de document comme un autre document XML ou encore une page web, fichiers multimédia. C'est d'ailleurs ce dernier cas que nous aborderons au cours de cette partie.

Comme toutes les technologies que nous avons abordées dans ce tutoriel, XSLT est un standard du W3C depuis 1999 pour sa première version et 2007 pour sa seconde version. Comme d'autres technologies que nous avons vues jusqu'ici, les documents XSLT sont écrits à l'aide d'un langage de type XML [W3C].

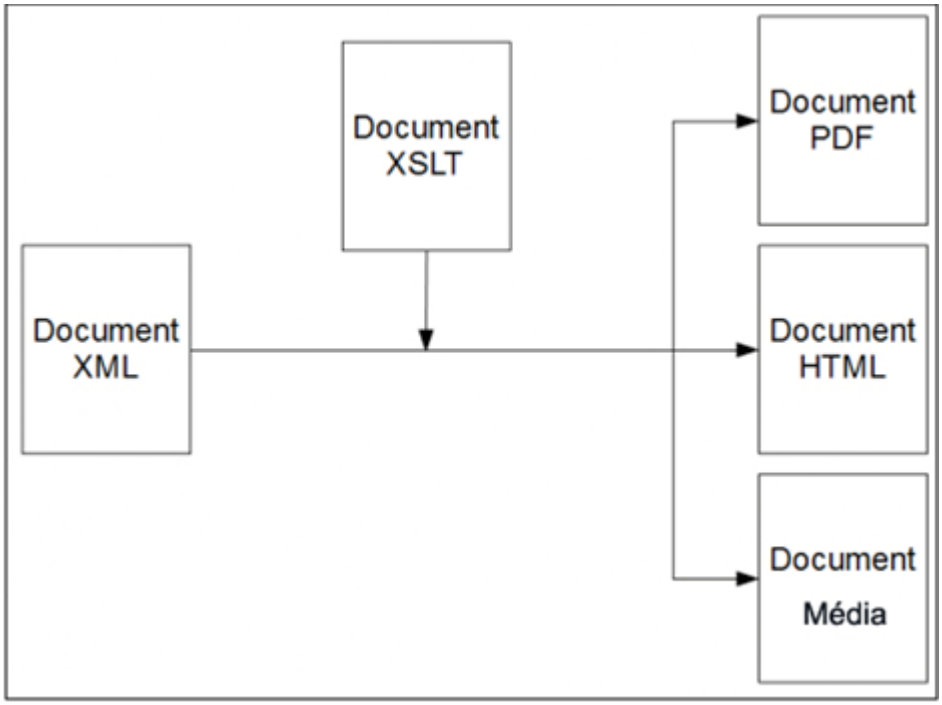


Figure 4.2 –Principe d'une transformation XSLT

7. modèle (MyriamLamolle, AmarZedazi)

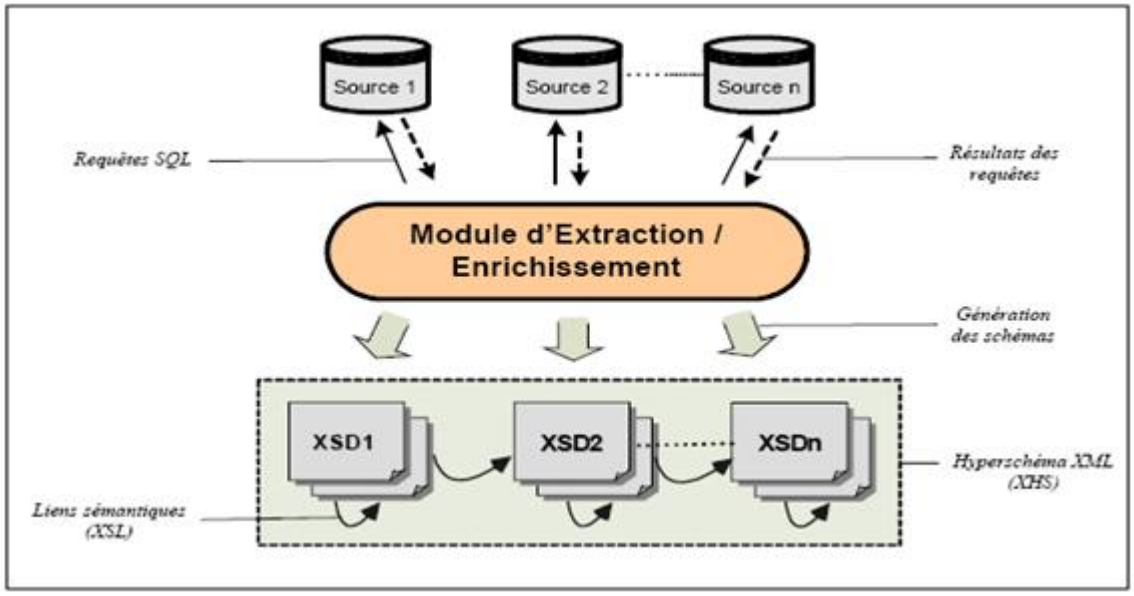


Figure 4.3 –Architecture d'intégration via XML. Modèle

Dans la figure:

La phase d'**extraction** homogénéise la représentation des différents schémas des BDs à **intégrer** en l'exprimant sous la forme d'une définition de schéma **XML** (i.eXSDi). **[34]**

Une fois cette **transformation** faite **[35],** chaque concept (dans les fichiers XSD) est représente selon deux dimensions à savoir une dimension structurelle et une dimension sémantique. Dans cette phase, la partie sémantique des schémas **XML** extraits est affinée par l'adjonction de méta connaissances soient déduites du catalogue des données, soient précisées par les experts de la BD.

Ces deux dimensions sont prises en compte dans le calcul du degré de similitude de concepts (la détection des conflits sémantiques et logiques dans les schémas **XML** lors de la phase de *matching)* qui nous permet de mettre en place les opérateurs de transformation (appelés liens sémantiques sur la figure 1) **[36].** Il est à noter que la recherche des « *liens semantiques entre schemas XSD »* ne rentre pas dans le cadre du présent article.

Le schéma **XML** issu de cette phase est compose de trois types d'élément à savoir :

- ***Concept:*** est l'élément le plus important dans **le schéma XSD.** Il peut engendrer des propriétés et/ou d'autres concepts imbriques, et porter des relations avec d'autres concepts.

- ***Relation*** *:* correspond aux associations structurelles et/ou sémantiques existant entre deux ou plusieurs concepts. Chaque relation possède éventuellement un sens de "lecture" (cardinalités [0,1] ou [1,1]) et un rôle

- ***Propriété*** *:* est l'élément le plus "basique" dans le schéma XSD. Pour chaque propriété, nous définissons son nom, son type et également ses occurrences dans le concept ou la relation, ses contraintes (unicité, renseigne, constante, etc.).

Au final, l'hyper schéma **XML** est constitué de l'ensemble des schémas des BDs sous forme XSD *étendu* et de l'ensemble des opérateurs de transformation d'un schéma a un autre sous forme XSL **[34].** L'**extraction** des différents schémas suit des règles précises pour les dimensions statique et dynamique d'un concept

8. Vue générale sur l'architecture proposée

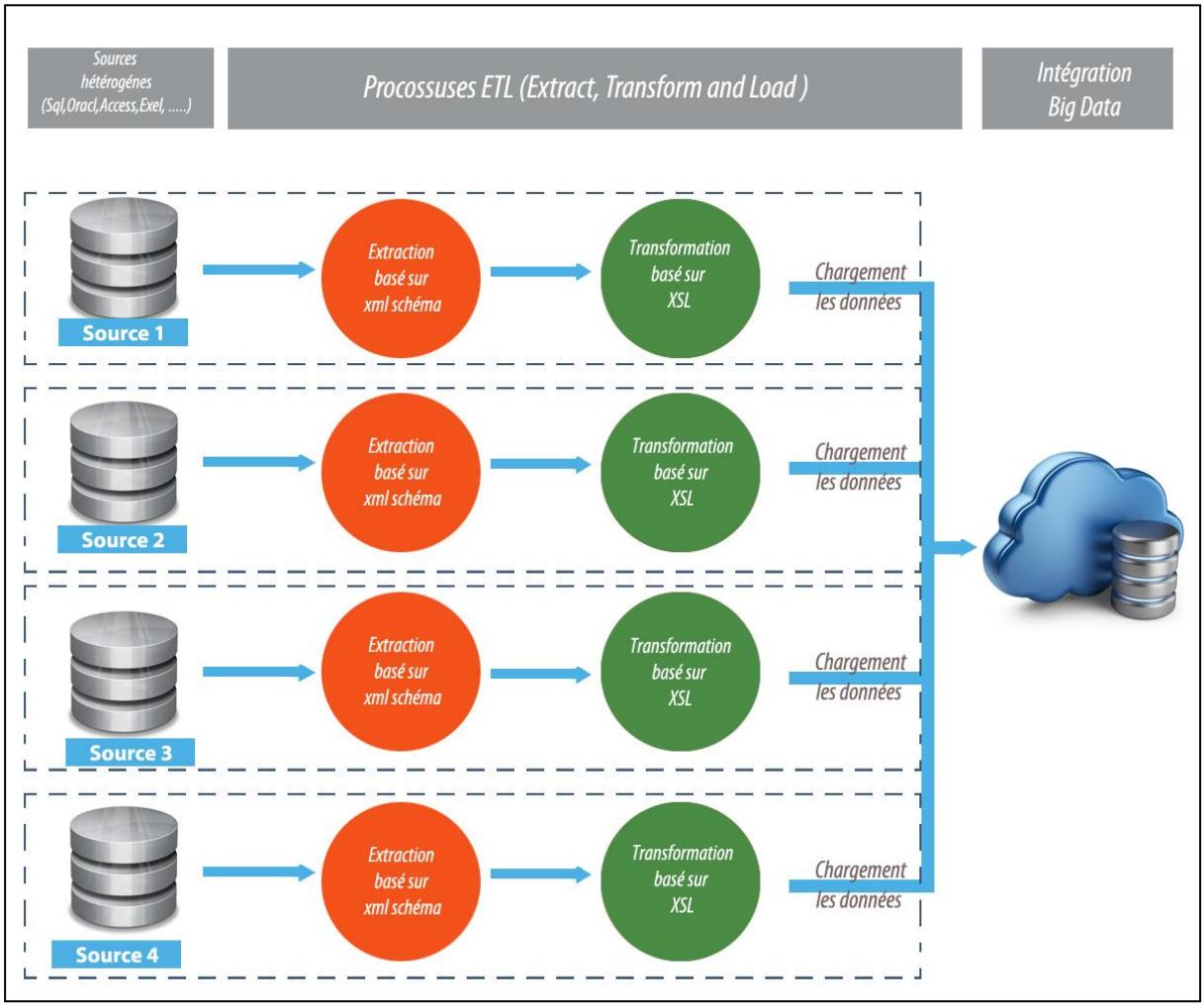


Figure 4.4 –Model d'intégration basée sur Parelle ETL et technologie XML

- ✓ Dans notre architecture, nous avons basé uniquement sur la partie semi-structurelle et logique de **la source de données**.

- ✓ Les données seront présentées lors de la phase d'**extraction** selon deux dimensions (partie semi structurale et la partie logique).

- ✓ Pour chaque source de données le **processus ETL** sera effectué en parallèle et l'intégration au temps réel.

- **Les différentes étapes**
  - **Sources Hétérogènes**

Les sources de données ne sont pas du même type ou la même structure de données qui est le système extraire des données à partir d'une source de données **hétérogènes** (SQL, Oracle, Access, Exl,...)

Exemple:

Source 01: base de donnée de type **SQL**

Etudient (Matricule, Nom, Prenom, NomAR, PrenomAR, Sex, Moyennebac, willaya)

Attribut	**Type**
Matricule	Int
Nom	Varchar
Prénom	Varchar
NomAr	Varchar
PrenomAr	Varchar
Sex	Varchar
Moyenne bac	Float
Willaya	Varchar

Tableau 4.1 : Base de donnée de type SQL

Source 02:base de donnée de type **Excel**

Person (NumIns, Name, Fname, ArName, ArFname, mf, note, state)

Attribut	**Type**
NumIns	Number
Name	Standard
Fname	Standard
ArName	Standard
ArFname	Standard
Mf	Standard
Moyennebac	Nombre
Willaya	Text

Tableau 4.2 : Base de donnée de type Excel

Source 03: base de donnée de type **Accès**

Bacheliers (Nins, Nom, Prénom, ArName, ArFname, sex, note, state)

Attribut	**Type**
Nins	Text
Nom	Text
Prenom	Text
ArName	Text
ArFname	Text
Mf	Text
Moyennebac	Numérique
Willaya	Text

Tableau 4.3 : Base de donnée de type Accès

Source 04: base de donnée de type **Oracle**

Bac_Note(#Matr_Etd, Module, Note)

Attribut	**Type**
Matr_Etd	nvarchar
Module	nvarchar
Note	Float

Tableau 4.4 : Base de donnée de type Oracle

- **Processus ETL**

❖ Extraction :

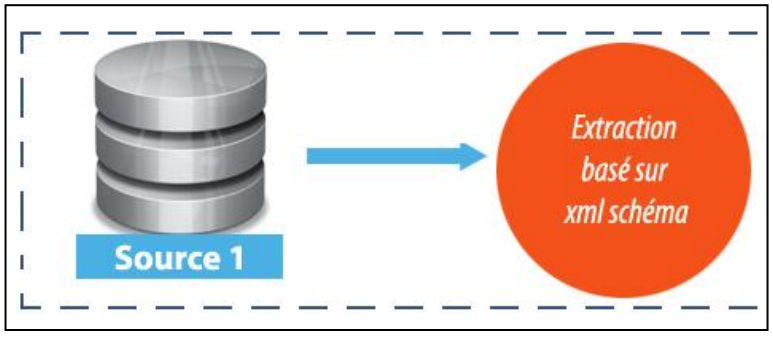


Figure 4.4 –La phase d'extraction

- Les règles d'**extraction** basé sur la structure de la base de données source les attributs et les types logique de ces attributs.

- les données extraites seront placés dans fichier **XML** basé sur un schéma **XSD**.

  Exemple :

Soit le schéma partiel d'une base de données sources dont le modèle relationnel est:

Etudient (Matricule, Nom, Prenom, NomAR, PrenomAR, Sex, Moyennebac, willaya)

Schéma XSD :

```
<xsd:complexTypename="Etudient"    type="Table_Etudient"...
/>
<xsd:attributename="Matricule" type="xsd:positiveInteger".../>

<xsd:attributename="Nom" type="xsd:string".../>

<xsd:attributename="Prenom" type="xsd:string".../>

<xsd:attributename="NomAr" type="xsd:string".../>

<xsd:attributename="PrenomAr" type="xsd:string".../>

<xsd:attributename="Sex" type="xsd:string".../>

<xsd:attributename="Moyenne" type="xsd:decimal".../>

<xsd:attributename="Willaya" type="xsd:string".../>
```

XML :

```
<Etudient>
  <Matricule>344875564</Matricule>

<Nom>EtudNom</Nom>

<Prenom>EtudPrenom</Prenom>

<NomAr>EtudNomAr</NomAr>

<PrenomAr>EtudPrenomAr</PrenomAr>

<Sex>EtudSex</Sex>

<Moyenne>EtudMoyenne</Moyenne>

<Willaya>EtudWillaya</Willaya>

</Etudient>
```

❖ Transformation :

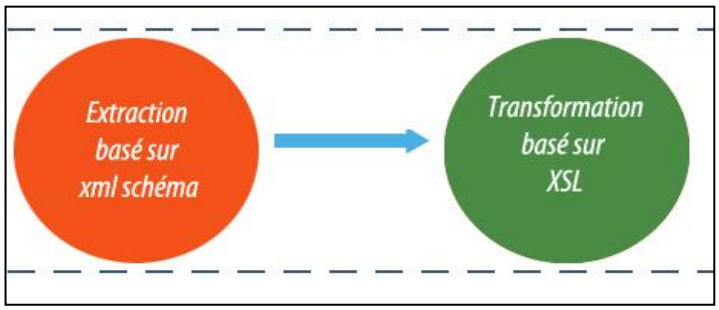


Figure 4.6 –La phase de transformation

- Les règles de transformation basée sur la structure de la base de données ciblent les attributs et les types logique de ces attributs, son type d'association avec d'autres attributs.
- Les attributs sont fortement pris en considération lors de la phase de transformation, comme la clé primaire, attributs de type uniques … etc.

- Les données extraites sont générées sous forme même que la base de données cible ou de la forme du cadre, qui sera présenté pour, telle que : vidéo, audio, forme des calculs statistiques …. etc.

- La phase de transformation exploité par les règles de passage XSLT (eXtended Stylesheet Language Transformations) Il faut créer un document XSL qui permettra de définir comment un document XML va être transformé. Nous nous appliquerons à ces transformations uniquement vers la forme de les données représentes ou stocké.

Exemple :

Pour la base de donnée cible (base de donnée de type Sql) et la présentation sous forme html :

Structure de la base de données :

Table 01 Person:

Attribut	**Type**
Matricule	Int
Nom	Nvarchar
NomAr	Nvarchar
Sex	Nvarchar
Moyenne	Real
Willaya	Nvarchar

Tableau 4.5 : Table Personne

Table 02 Notes:

Attribut	**Type**
Matricule	Int
Module	Nvarchar
Note	Real

Tableau 4.6 :Table Note

Table 03 Présentation de donnée :

Attribut	**Type**
Matricule	Nvarchar
DonneePresent	Nvarchar

Tableau 4.7 : Présentation de donnée

Schéma XSD table 01 :

<**xsd:complexType**name=**"Etudient**" type="Table_Etudient"... />

<**xsd:attribute**name="**Matricule**" type="xsd:positiveInteger" **use="required" unique="true"**/>

<**xsd:attribute**name="Nom" type="xsd:string"/>

<**xsd:attribute**name="NomAr" type="xsd:string"./>

<**xsd:attribute**name="Sex" type="xsd:string".../>

<**xsd:attribute**name="Moyenne" type="xsd:decimal".../>

<**xsd:attribute**name="Willaya" type="xsd:string".../>

Schéma XSD table02 :

<**xsd:complexType**name=**"Note**" type="Table_Note"/>

<**xsd:attribute**name="**Matricule**" type="xsd:positiveInteger"/>

<**xsd:attribute**name="Module" type="xsd:string".../>

<**xsd:attribute**name="Note" type="xsd:decimal".../>

Schéma XSD table03 :

<**xsd:attribute**name="**Matricule**" type="xsd:positiveInteger"/>

<**xsd:attribute**name="DonneePresent" type="xsd:string".../>

Transformation XML avec les régales XSLT table 03 :

Transformation la transformation des données à la forme de présentation en html

Pour la table 03 :

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" encoding="iso-8859-1" doctype-public="-//W3C//DTD XHTML 1.0
Transitional//FR" doctype-system="http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"/>
<xsl:template match="/">
<html>
..<head>
..<title>Liste des etudiants</title>
..</head>
..<body>
..<h1>Etat des heures</h1>
..<xsl:for-each select="Table_Etudient ">
....<h2>Matricule<xsl:value-of select="Matricule"/></h2>
....<h2>Nom<xsl:value-of select="Nom"/></h2>
....<h2>Nom Arabe<xsl:value-of select="NomAr"/></h2>
....<h2>Sex<xsl:value-of select="Sex"/></h2>
....<h2>Moyenne de BAC<xsl:value-of select=" Moyenne "/></h2>
....<h2>Willaya de Bac<xsl:value-of select="Willaya"/></h2>
....<xsl:for-each select="Note">
......<h3>Notes</h3>
......<table border="3">
........<tr>
..........<td>Module<xsl:value-of select="Module"/></td>
..........<td>Note<xsl:value-of select="Note"/></td>
........</tr>
......</table>
....</xsl:for-each>
..</xsl:for-each>
..</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

❖ Chargement (Load) de données:

- Le chargement des données est basée sur les règles d'insertion et la jointure de la base de données cible ou la forme de présentation des données ;

- Les erreurs de chargement et les exceptions sont résolus par le système d'intégration des données de la base cible ou la forme de présentation telle que Sql Exeption Class de Microsoft et Oracle Named System Exceptions pour la redondance des données et la jointure automatique.

9. Implémentation

Dans note contribution nous avons utilisé le Visual Studio .NET une application d'intégration d'une base de données massives des étudiants qui ont eu leur baccalauréat d'années 2015 en appliquant un entreposage de données (Etablir un classement des bacheliers de chaque wilaya respectivement) en utilisant un processus ETL basé sur le langage XML.

9.1. Visual Studio .NET

Est un jeu complet d'outils de développement permettant de générer des applications Web ASP, des services Web XML, des applications bureautiques et des applications mobiles. Visual Basic .NET, Visual C++ .NET, Visual C# .NET et Visual J# .NET utilisent tous le même environnement de développement intégré (IDE, integrated development environment), qui leur permet de partager des outils et facilite la création de solutions faisant appel à plusieurs langages. Par ailleurs, ces langages permettent de mieux tirer parti des fonctionnalités du .NET Framework, qui fournit un accès à des technologies clés simplifiant le développement d'applications Web ASP et de services Web XML.

9.2. Traits marquants de Visual Studio .NET

9.2.1. Windows Forms

Windows Forms est la nouvelle plate-forme de développement d'applications Microsoft Windows basée sur le .NET Framework. Elle met à votre disposition un ensemble complet de classes extensibles et orientées objet, avec lequel vous pourrez développer des applications Windows élaborées. En outre, Windows Forms peut faire office d'interface utilisateur locale dans une solution distribuée multicouche. Pour plus d'informations, consultez Introduction aux Windows Forms.

9.2.2. Services Web XML

Les services Web XML sont des applications qui peuvent recevoir des demandes et des données au format XML via le protocole HTTP. Il est possible d'accéder aux services Web XML à partir de n'importe quel langage, modèle de composant ou système d'exploitation dans la mesure où ces services ne sont pas tributaires d'une technologie de composant ou d'une convention d'appel d'objet particulières. Dans Visual Studio .NET, vous pouvez rapidement créer et inclure des services Web XML en utilisant Visual Basic, Visual C#, JScript, les extensions managées pour C++ ou ATL Server. Pour plus d'informations, consultez Programmation du Web avec les services Web XML.

9.2.3. Prise en charge du langage XML

Le langage XML (Extensible Markup Language) fournit une méthode pour décrire des données structurées. XML est un sous-ensemble du langage SGML qui est optimisé pour la livraison de données sur le Web. Le World Wide Web Consortium (W3C) définit les normes XML de telle sorte que les données structurées soient uniformes et indépendantes des applications. Visual Studio .NET prend totalement en charge XML et met à votre disposition le Concepteur XML pour faciliter la modification des données au format XML et la création de schémas XML. Pour plus d'informations, consultez Schémas et données XML et Concepteur XML.

9.2.4. Le .NET Framework

Le .NET Framework est un environnement offrant plusieurs langages pour générer, déployer et exécuter des services et des applications Web XML. Il se compose de trois éléments principaux :

- Common Language Runtime Malgré son nom, le runtime joue un rôle aussi bien au moment de l'exécution d'un composant que dans les expériences de la phase de développement. Au moment de l'exécution du composant, le runtime est chargé de gérer les allocations de mémoire, de démarrer et d'arrêter les threads et les processus, de mettre en application la stratégie de sécurité et de satisfaire les dépendances que le composant est susceptible d'entretenir avec d'autres composants. Au moment du développement, le rôle du runtime évolue légèrement : comme il automatise un grand nombre de tâches (par exemple celles occasionnées par la gestion de la mémoire), le runtime est vécu par le développeur comme très simple, surtout en comparaison avec COM, tel qu'il se présente aujourd'hui. En particulier, des fonctionnalités telles que la réflexion réduisent considérablement la quantité de code qu'un développeur doit écrire pour transformer le code de niveau BLL (Business Logic Layer) en un composant réutilisable.

- Classes de programmation unifiées Le .NET Framework fournit aux développeurs un ensemble de bibliothèques de classes (API) unifié, orienté objet, hiérarchique et extensible. Actuellement, les développeurs C++ utilisent les classes MFC (Microsoft

Foundation Class) et les développeurs Java les classes WFC (Windows Foundation Class). Le .NET Framework unifie ces modèles disparates et permet aux programmeurs Visual Basic et JScript d'accéder aussi à des bibliothèques de classes. En créant un jeu d'API commun à tous les langages de programmation, le Common Language Runtime permet l'héritage, la gestion des erreurs et le débogage interlangage. Tous les langages de programmation, depuis JScript jusqu'à C++, ont un accès analogue au .NET Framework, et les développeurs ont toute liberté quant au choix du langage qu'ils souhaitent utiliser.

- ASP.NET ASP.NET repose sur les classes de programmation du .NET Framework et fournit un modèle d'application Web doté d'un jeu de contrôles et d'une infrastructure facilitant la génération d'applications Web ASP. ASP.NET inclut un jeu de contrôles qui encapsule des éléments d'interface utilisateur HTML communs, tels que les zones de texte et les menus déroulants. Ces contrôles s'exécutent sur le serveur Web, mais présentent leur interface utilisateur au navigateur sous forme de code HTML. Sur le serveur, les contrôles exposent un modèle de programmation orienté objet qui offre au développeur Web toute la richesse de la programmation orientée objet. ASP.NET fournit également des services d'infrastructure, tels que la gestion des états de session et le recyclage de processus, qui contribuent à réduire encore plus la quantité de code qu'un développeur doit écrire et à augmenter la fiabilité des applications. Par ailleurs, ASP.NET utilise les mêmes concepts pour permettre aux développeurs de fournir le logiciel sous forme de service. À l'aide de fonctionnalités des services Web XML, les développeurs ASP.NET peuvent écrire leur code BLL (Business Logic Layer) et utiliser l'infrastructure ASP.NET pour fournir ce service via SOAP.

9.3. Création les interfaces et les formes

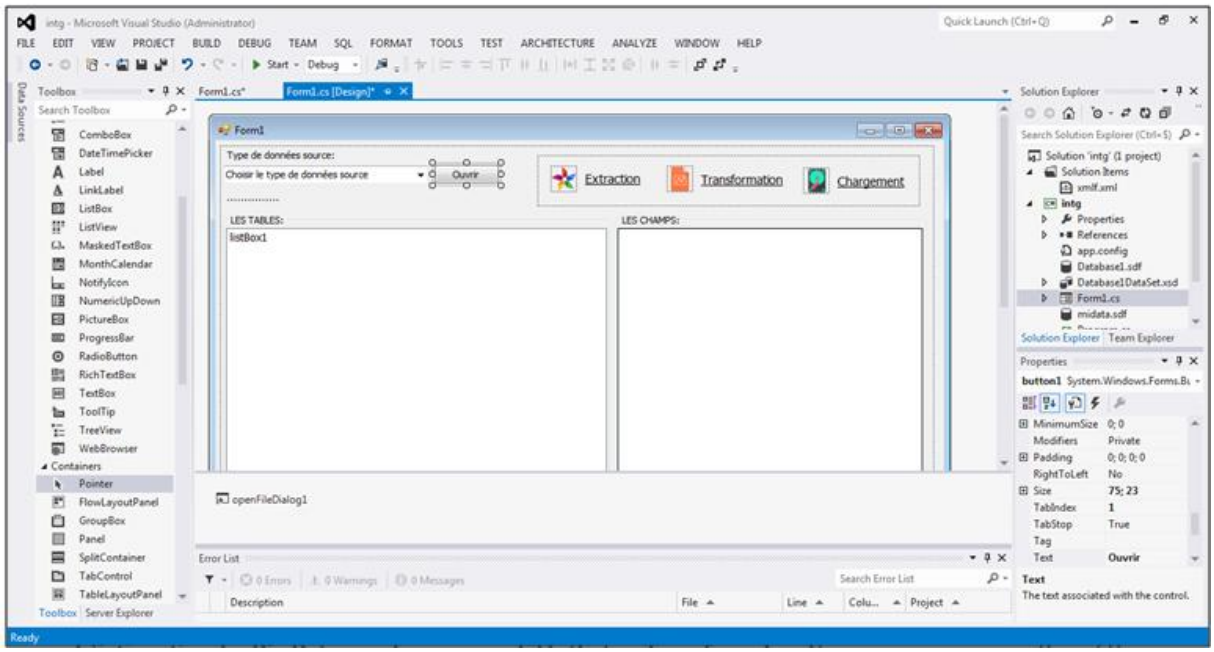


Figure 4.7 –Création la forme principale

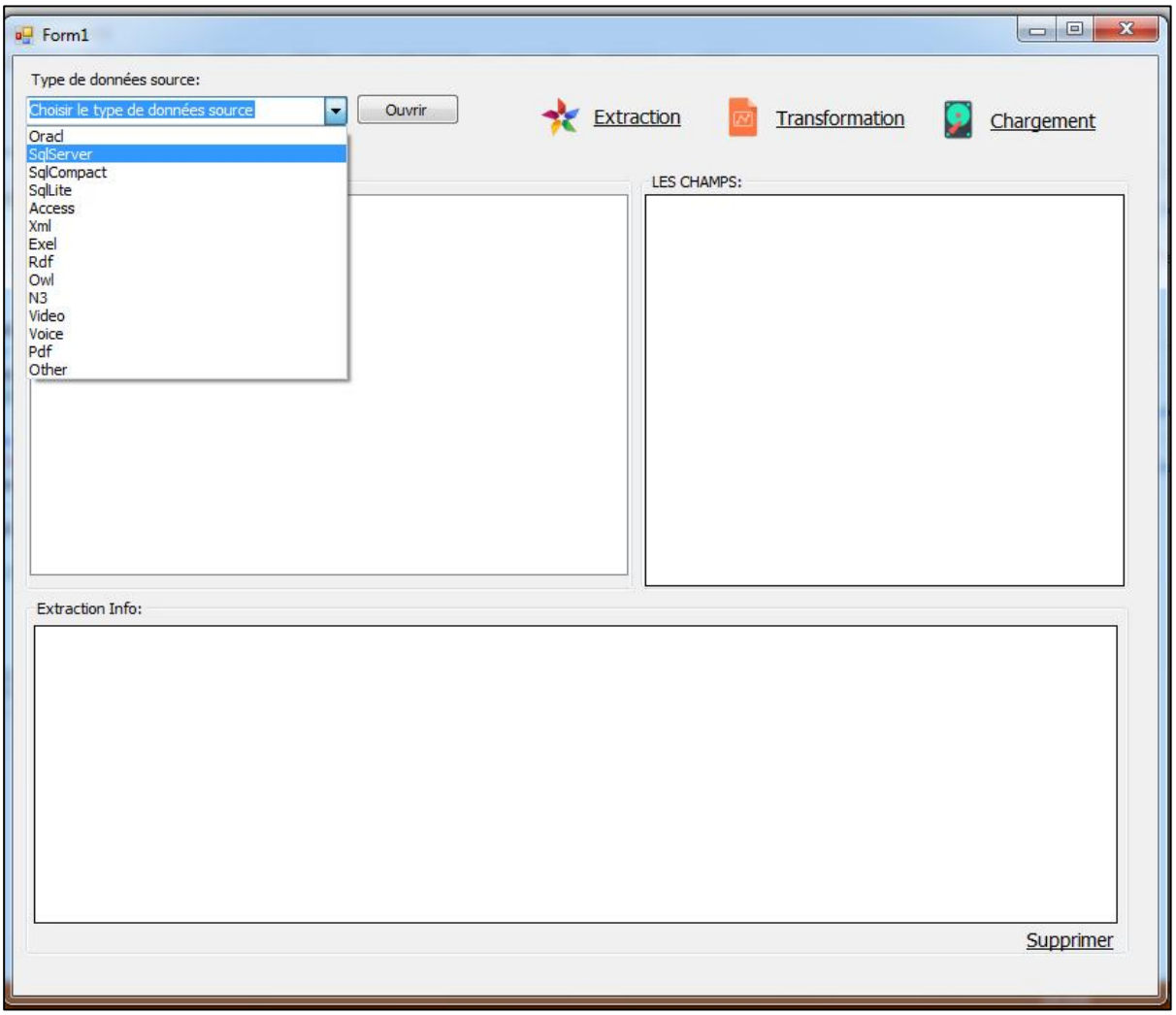


Figure 4.8 –Forme Principale « choisir Type BD Source »

On peut choisir les bases de données à intégrer en cliquant sur le bouton ouvrir et sélectionner les bases de données sources.

Code sélectionne « type BD source et les tables» :

```
DataTable schema;
                dc.clear_dataf();
OleDbConnection    cn    =    newOleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=" + files + " ;");
                cn.Open();

                schema = cn.GetOleDbSchemaTable(OleDbSchemaGuid.Columns, newObject[] {
null, null,listBox1.SelectedItem.ToString() });

int i=0;
foreach (DataRow dr in schema.Rows)
                {
                    dc.add_datafield(dr["COLUMN_NAME"].ToString(),
schema.Columns[i].DataType.Name.ToString());
```

```
                i++;
            }
OleDbConnection cnn = dc.creat_connection();
string cmdstr = "SELECT dataf as [Champ],dataft as [TypeChamp] from dataf";
OleDbCommand cmd = newOleDbCommand(cmdstr, cnn);
OleDbDataAdapter adt = newOleDbDataAdapter();
            adt.SelectCommand = cmd;
DataSet ds2 = new System.Data.DataSet();
            adt.Fill(ds2, "dataf");
            dataGridView2.DataSource = ds2.Tables[0];
cn.Close();

                    cnn.Close();
```

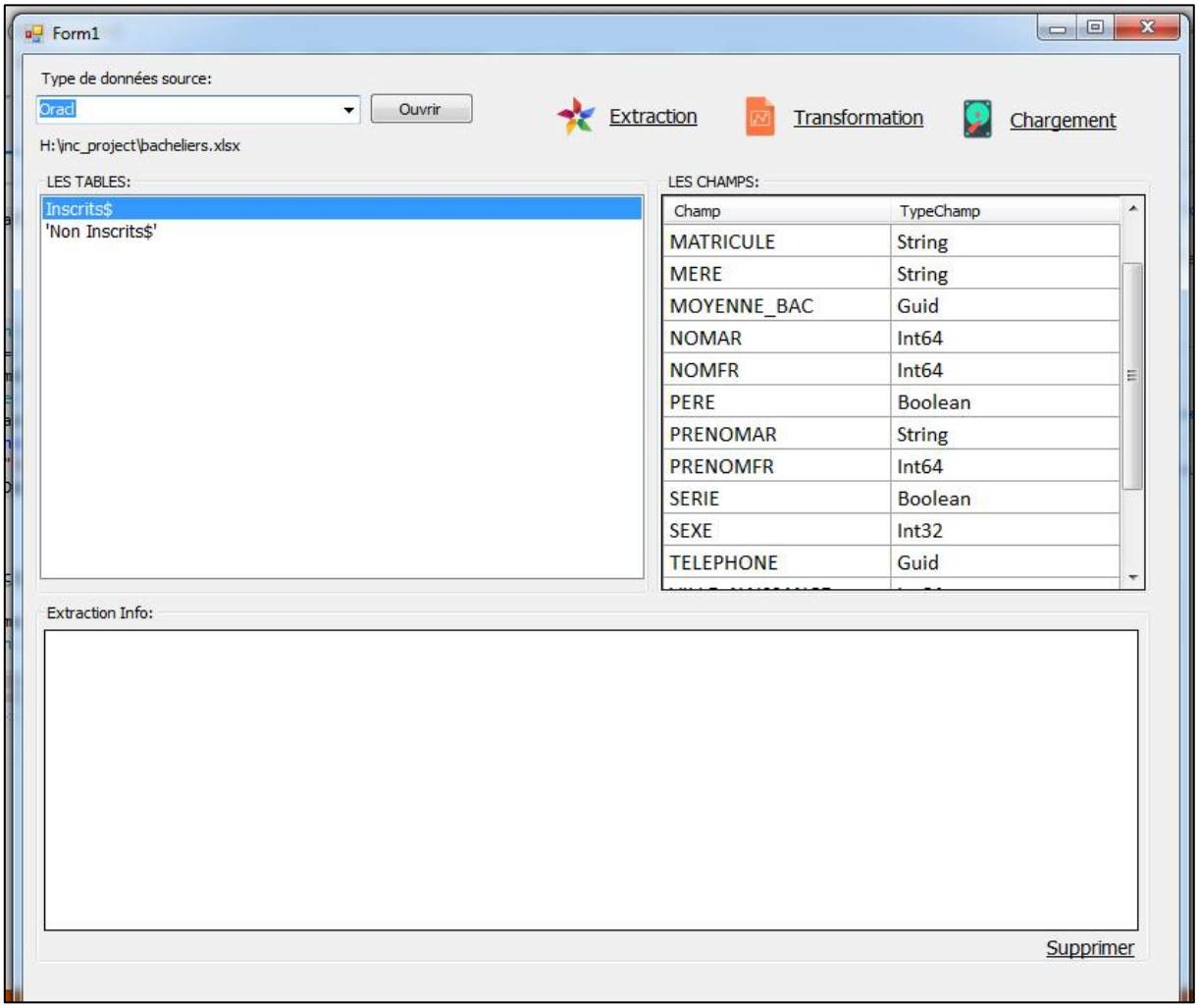


Figure 4.9 –Forme « «choisir les champs »

Code « choisir les champs » :

```
DataTable schema;
OleDbConnection    cn    =    newOleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=" + files + " ;");
                    cn.Open();
                    schema = cn.GetOleDbSchemaTable(OleDbSchemaGuid.Tables, null);
String[] exelsheet = newstring[schema.Rows.Count];
int i = 0;
foreach (DataRow row in schema.Rows)
                    {
                        listBox1.Items.Add(row["TABLE_NAME"].ToString());
i++;
                    }
                           cn.Close();
```

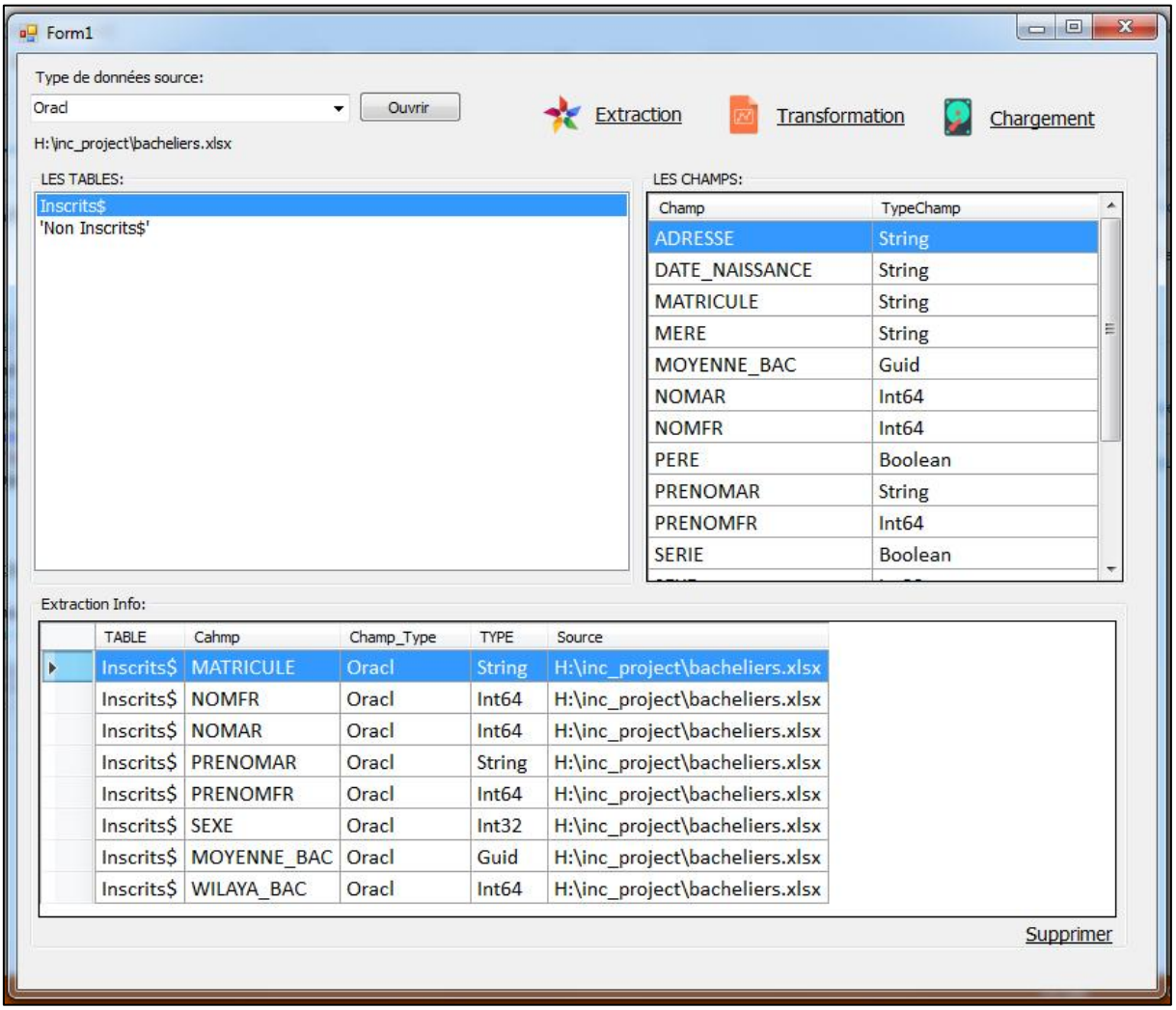


Figure 4.10 –"sélection les champs (Exemple)"

Extraction :

- Dans l'extraction les données extraites seront placés dans fichier **XML** basé sur un schéma **XSD** prêt pour la transformation et le chargement dans l'entrepôt de données.

Ficher XSD source01 (Exemple):

```
<?xmlversion="1.0"encoding="utf-8"?>
<xs:schematargetNamespace="http://tempuri.org/XMLSchema.xsd"
elementFormDefault="qualified"
xmlns="http://tempuri.org/XMLSchema.xsd"
xmlns:mstns="http://tempuri.org/XMLSchema.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:complexTypename="Inscription">
<xs:attributename="Matricule"type="xs:positiveInteger"/>
<xs:attributename="NomFR"type="xs:string"/>
<xs:attributename="PrenomFR"type="xs:string"/>
<xs:attributename="NomAr"type="xs:string"/>
<xs:attributename="PrenomAr"type="xs:string"/>
<xs:attributename="Sex"type="xs:string"/>
<xs:attributename="Moyenne"type="xs:decimal"/>
<xs:attributename="Willaya"type="xs:string"/>
</xs:complexType>
</xs:schema>
```

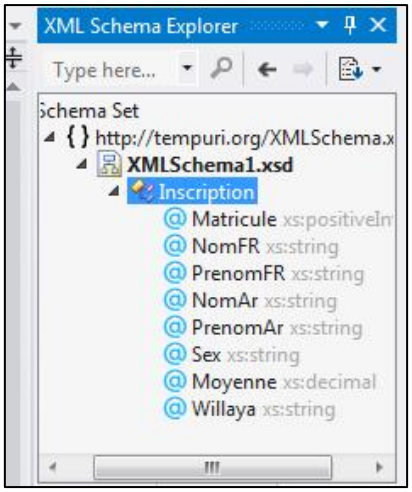


Figure 4.11 –"FicherSchema1.xsd de fichier XML (Exemple)"

Figure 4.12 –"Fichier XML données (extrait)"

Transformation et chargement :

Transformer les types d'attributs d'après les types d'attributs dans notre entrepôt de données par exemple l'attribut « moyenne » était présenté dans la base de données source SQL en FLOAT nous avons transformé ce type d'attribut en DECIMAL.

Ficher XSD schéma d'intégration :

```
<?xmlversion="1.0"encoding="utf-8"?>
<xs:schematargetNamespace="http://tempuri.org/XMLSchema.xsd"
elementFormDefault="qualified"
xmlns="http://tempuri.org/XMLSchema.xsd"
xmlns:mstns="http://tempuri.org/XMLSchema.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:attributename="Matricule"type="xs:positiveInteger"/>
<xs:attributename="Nom"type="xs:string"/>
<xs:attributename="NomAr"type="xs:string"/>
<xs:attributename="Sex"type="xs:string"/>
<xs:attributename="Moyenne"type="xs:decimal"/>
<xs:attributename="Willaya"type="xs:string"/>
</xs:schema>
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<Person xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Etudient>
        <Matricule>31000002</Matricule>
        <NomFR>DIHMI AHMED</NomFR>
        <NomAr>أحمد ديهمي</NomAr>
        <Sex>MASCULIN</Sex>
        <Moyenne>11.15</Moyenne>
        <Willaya>11</Willaya>
        </Etudient>
    <Etudient>
        <Matricule>31000006</Matricule>
        <NomFR>DOUAKHA OUSSAMA</NomFR>
        <NomAr> أسامة دواخة  </NomAr>
        <Sex>MASCULIN</Sex>
        <Moyenne>10.59</Moyenne>
        <Willaya>11</Willaya>
    </Etudient>
```

Figure 4.13 – Ficher XML d'intégration

Une présentation des données avec le langage HTML (page web) en utilisant XSLT :

```xml
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:msxsl="urn:schemas-microsoft-com:xslt" exclude-result-prefixes="msxsl"
>
    <xsl:output method="html" indent="yes"/>

    <xsl:template match="@* | node()">
      <html>
        <head>
          <title>Liste des etudiants</title>
        </head>
         <body>
           <h1></h1>
           <xsl:for-each select="Table_Etudient ">
              <h2>
              Matricule<xsl:value-of select="Matricule"/>
             </h2>
              <h2>
              Nom<xsl:value-of select="Nom"/>
             </h2>
              <h2>
              Nom Arabe<xsl:value-of select="NomAr"/>
             </h2>
              <h2>
              Sex<xsl:value-of select="Sex"/>
             </h2>
              <h2>
              Moyenne de BAC<xsl:value-of select="Moyenne "/>
             </h2>
              <h2>
```

Figure 4.14 – Ficher XSLT pour la présentation sous forme HTML

bac: Query(C:\User...intg\Database1.sdf) | XSLTFile1.xslt* | XMLSchema2.xsd* | XMLSchema2.xsd [De

Matricule	NomFR	NomAR	Sex	Moyenne	Willaya
31000006	DOUAKHAOUS...	أسامةدواخة	MASCULIN	10.59	11
31000008	MENTADJIASMA	أسماءمنتاجي	FEMININ	10.54	11
31000009	ANBI AHMEDA...	أسماءعنبي أحمد	FEMININ	10.50	11
31000010	SALMIASMA	أسماءسالمي	FEMININ	10.78	11
31000013	ZELFAOUM KEL...	أم كلثومزالغة	FEMININ	11.04	11
31000015	ATTAFIAMINE	أمينعطافي	MASCULIN	11.94	11
31000017	AG MATALLA...	أمينةأق معطا الله	FEMININ	12.00	11
31000021	SOULILYES	إلياسسول	MASCULIN	10.33	11
31000036	MOULGARA AS...	اسماءمولقارة	FEMININ	11.54	11
31000048	TOUDJIDINE	الدينالتوجي	MASCULIN	11.06	11
31000060	TOUDJICHERIF	الشريفالتوجي	MASCULIN	11.65	11
31000062	BIGALICHEIKH	الشيخبيقالي	MASCULIN	10.75	11
31000067	BENAISSAEMB...	امباركةبن عيسى	FEMININ	10.89	11
31000077	BEN MESSAOU...	ايمانبن مسعود	FEMININ	11.20	11

Figure 4.15 – Base de données intégrée

En fin nous avons obtenue une base de données massive intégrée (données cibles) chargé dans un entrepôt de donnée avec un classement des bacheliers de chaque wilaya respectivement.

10. Conclusion

Dans ce chapitre nous avons proposé une architecture d'intégration avec la définition d'une présentation standard moulée semi-structurelle d'une base de données massives avec un processus ETL traditionnel basé sur le langage XML qui facilite les phases d'intégration (Extraction, Transformation et Chargement) ainsi permet le parallélisme et en temps réel dans un but d'assurer une intégration dans un temps plus réduit et un langage XML facile pour l'implémentation et rapide pour l'exécution ainsi c'est un langage supporté par toutes les plate-forme.

Conclusion générale

Les volumes de données se développent plus rapidement que jamais avant et par l'année 2020, environ 1,7 méga octets de nouvelle information seront créés chaque seconde pour chaque être humain sur la planète. D'ici là, notre univers numérique accumulé des données se développera de 4,4 zettabyets aujourd'hui à environ 44 zetta bytes, ou de 44 trillion de giga octets. Chaque seconde nous créons de nouvelles données. Par exemple, nous exécutons 40.000 questions de recherche chaque seconde (sur seul Google), qui lui fait 3,5 recherches par jour et 1,2 recherche trillion par an. Ce problème d'amélioration de stockage des données produit beaucoup d'intérêts dans des systèmes d'intégration de donnée, pour permettre une prise de décision améliorée à l'aide d'un système d'aide à la décision BI en utilisant le processus ETL pour intégrer ces Big Data.

Notre idée est une modification de l'approche de A.ZERDAZI en 2015 « Intégration de Bases de données hétérogènes par une modélisation conceptuelle XML » pour faire une intégration des données en parallèle et en temps réel en basant sur le schéma XML au niveau de la phase d'extraction dans un objectif de :

- ✓ Faire une intégration dans une durée plus réduite grâce à l'intégration en parallèle ;
- ✓ Le langage XML prendre en charge tout les formats de domaine d'intégration : structuré, semi-structuré et non-structuré par l'utilisation des technologies des Big Data et de l'ETL ;
- ✓ La combinaison des technologies XML, ETL et Big Data permet d'obtenir une architecture plus performante que celles existantes ;
- ✓ XML permet de représenter des données indépendamment de leurs présentations et de leurs stockages.

Enfin, l'objectif principal visé reste comme une perspective à valider et améliorer l'architecture ainsi améliorer le temps d'intégration.

Bibliographie

Thèses :

[1] Rodrigo FREITAS PAIXÃO ; « La Business Intelligence est-elle adaptée au monde des PME? »; Travail réalisé en vue de l'obtention du diplôme HES ; 24/11/2006.

[2] M.CHANTAL DENIS ; « conception et réalisation d'un entrepôt de données institutionnel dans une perspective de support à la prise de décision » ; mémoire présenté à l'université du québec à trois-rivières, 24/05/2012.

[15] ACEF MOHAMED; « utilisation du modele mapreduce dans les differents systemes nosql: etude comparative », thése Pour l'Obtention du Diplôme de Post Graduation Spécialisée, 14/02/2015.

[30] Lahmar Fatima ; « Une approche Hybride d'intégration de sources de données hétérogènes dans les datawarehouses », THESE doctorat Canstantine 2011.

[36] Rapport de la théorie des catégories à la représentation des connaissances. Thèse de doctorat, Université Pierre et Marie Curie, Paris VI, (1988).

Articles de revue :

[3] B.Mahfoud, O.Boussaid, Z. Alimazighi, F.Bentayeb ; « PF-ETL : vers l'intégration de données massives dans les fonctionnalités d'ETL » ; conférence paper · mai 2014.

[18] Casey McTaggart ; **«** hadoop mapreduce » ; conférence présentation cadre orienté objetn CSCI 5448; 31/03/2011.

[28] D.Srivastava, « Big Data Integration », Revue X.L.Dong, 2013.

[34] XML Schema, W3C Recommendation: XML Schema Primer, W3 Consortium, Availableat http://www.w3.org/TR /xmlschema-0, (2001).

Les journaux :

[4] Shaker H, A.Sappagh ; « A proposed model for data warehouse ETL processes » ; Journal of King Saud University-Computer and Information Sciences ; 16/06/2011.

[14] Marinela Mircea ; « Perspectives on Big Data and Big Data Analytics » ; Journal vol. III Data base Systems, 07/01/2013.

Les livres électroniques :

[5] Christian Desrosiers ; « Entrepôts de données et intelligence d'affaires » ; cours Département de génie logiciel et des technologies de l'information ; 09/03/2015.

[16] K. Panigrahi ; « hadoop big data analysis », Livre, 06/01/2015.

[17] Shv, Hairong, SRadia, Chansler ; « The Hadoop Distributed File System » ;livre ; 09/04/2010 ; 978-1-4244-7153-9/10/$26.00 ©2010 IEEE.

[19] Meyer Léonard, « L'AVENIR DU NoSQL », Livre VOL III ; 2014.

[26] Dan VODISLAV , « l'intégration des données » ; Cours IED Université de Cergy-Pontoise Master Informatique M1 2015.

[27] Mohand-Saïd Hacid et Chantal Reynaud , « L'intégration de sources de données » , Cours Université Claude Bernard Lyon 1 2005.

[29] AnHai Doan Robert McCann , « Building Data Integration Systems A Mass Collaboration Approach », livre 2014 .

[35] Ontario , « Schema Evolution in Heterogeneous Database Architectures », « A Schema Transformation Approach. In Proc », Livre CAiSE'02, Toronto, Canada 2002

Les sites :

[6] http://www.dataversity.net/an-introduction-to-business-intelligence-the-benefits/

[7] https://www.chinaabout.net/advantages-and-disadvantages-of-business-intelligence-bi-with-relevance-to-business-performance/
[8] http://www.piloter.org/business-intelligence/datawarehouse.htm

[9] http://igm.univ-mlv.fr/~dr/XPOSE2005/entrepot/datawarehouse.html
[10] http://www.supinfo.com/articles/single/1460-data-warehouse-entrepot-donnees
[11] http://www.piloter.org/business-intelligence/ETL.htm
[12] //www.splicemachine.com/stuck-in-the-middle-the-future-of-data-integration-is-no-etl/
[13] http://noetl.org/
[20] http://www.ibmbigdatahub.com/blog/why-only-one-5-vs-big-data-really-matters
[21] hhttp://cloudtweaks.com/2013/11/cloud-computing-makes-democratization-of-big-data-possible/
[22] http://blog.khaledtannir.net/2011/05/mapreduce-2eme-partie/#.VuhEn33hDIV
[23] https://sites.google.com/site/selhconsultant/hadoop-composants.
[24] http://stackoverflow.com/questions/22533814/what-is-use-of-hcatalog-in-hadoop.
[25] http://www.technologies-ebusiness.com/langages/les-bases-no-sql.
[31] http://www.agiledss.com/services/developpement-implantation/integration-de-donnees.html.
[32] https://www.techopedia.com/definition/4273/enterprise-information-integration-eii
[33] http://searchsoa.techtarget.com/definition/EAI.