République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad Dahlab Blida

Faculté des sciences

Département d'informatique

Mémoire présenté par :

MOHAMED MAHMOUD Abdelkarim & Touil Abdelouahab

En vue d'obtenir du diplôme de master

Domaine : Mathématique et informatique

Filière : Informatique

Spécialité : Informatique

Option : Ingénierie de logiciel

Sujet :

SYSTÈME D'AIDE À LA DÉCISION : ENTREPÔT DE DONNÉES EXTENSIBLE

Soutenu le :

Devant le jury :

	Présidente
	Examinatrice
Mme. Chiki	Promotrice
Mme. Semar	Encadrante

Promotion

2018 / 2019

ملخص

اليوم ، تؤثر أنظمة صنع القرار بشكل كبير على إدارة المنظمات المختلفة ، فهي توفر دعماً كبيراً لإدارة خطط عملها والتعلم وتنفيذ الابتكارات الإضافية.

في الوقت الحالي ، يواجه الوضع الاقتصادي في الجزائر صعوبات في اتخاذ القرارات الجيدة. ويرجع ذلك إلى تنوع مصادر المعلومات التي انفجرت وأصبحت ضخمة ومعقدة. يثير هذا اهتمامًا بمستودعات البيانات وأنظمة التحليل التي تسمح بتحليل هذه البيانات الضخمة. يتمثل أحد الجوانب المهمة لنظام مستودع البيانات في قدرته على التطور وفقًا لاحتياجات المستخدم والمؤسسة ودمج المتطلبات الجديدة مع الحفاظ على تناسقها.

الهدف من العمل المقدم في هذه الورقة هو تصميم وتنفيذ مستودع بيانات قابل للتوسع قادر على مواكبة التطور المستمر في العالم وتحديداً الحقل الهيدروليكي في جنوب الجزائر. يتم ضمان هذا التطور من خلال إمكانية تكامل العديد من المستودعات عبر طرق العرض الملموسة وتصميم ETL في شكل خطط الإعتماد المتكونة من وظائف الاستخراج ومعالجة البيانات.

الكلمات المفتاحية : مستودع البيانات، KPI ، دعم القرار ، القابلية للتوسعة .

Résumé

Aujourd'hui, les systèmes décisionnels ont un impact majeur sur le gestion de différentes organisations.Ils apportent une aide non négligeable au pilotage de leurs plans d'actions, à l'apprentissage et à la réalisation d'innovations incrémentales. Actuellement, la situation économique en Algérie fait face à des difficultés pour la prise de bonnes décisions. Ceci est dû à la diversité des sources d'information qui se sont éclatées et devenues volumineuses et complexes. Cela évoque un intérêt aux entrepôts de données et des systèmes d'analyse qui permettent l'analyse de ces données massives. Un aspect important d'un système d'entrepôt de données est qu'il soit capable à évoluer en fonction des besoins de l'utilisateur et de l'organisation et d'intégrer de nouvelles exigences en maintenant sa cohérence. Le travail présenté dans se mémoire a pour objective La conception et l'implémentation d'un entrepôt de données extensible capable de s'adapter l'évolution continue du l'environnement et spécifiquement le domaine Hydraulique au sud de l'Algérie. Cette évolution est assurée par la possibilité de l'intégration de plusieurs entrepôts via les vues matérialisés et la conception de l'ETL en forme de schéma de dépendances composés de fonctions d'extraction et de traitement de données.

Mots clés : E ntrepôt de données, KPI, Aide à la décision, Extensibilité, .

Abstract

Today, decision-making systems have a major impact on the management of different organizations. They provide a significant support to the management of their action plans, learning and the realization of incremental innovations, Currently, the economic situation in Algeria faces difficulties in making good decisions. This is due to the diversity of information sources that have exploded and become bulky and complex. This evokes an interest in data warehouses and analysis systems that allow the analysis of this massive data. An important aspect of a data warehouse system is that it is able to evolve according to the needs of the user and the organization and to integrate new requirements while maintaining its consistency. The objective of the work presented in this dissertation is the study of decision-making systems (Business Intelligence) and the design of an extensible data warehouse capable of managing the continuous evolution of the world and specifically the hydraulic domain in southern Algeria. This evolution is ensured by the possibility of the integration of several warehouses via the materialized views and the design of the ETL in the form of a dependency scheme composed of extraction and data processing functions.

Key words: D ata Warehouse, KPIs, Decision Support, Extensibility, Business Intelligence.

DÉDICACES

Je dédie ce mémoire à Mes chers parents, que nulle dédicace ne puisse exprimer mes sincères sentiments, pour leur patience illimitée, leur encouragement continue, leur aide, en témoignage de mon profond amour et respect pour leurs grands sacrifices.

Mes chers amis Abdelkader, Ahmed, Imene, Anis, Djassem, Meriem, Amokrane, Mounir, Seifeddine pour leur grand amour et leur soutien qu'ils trouvent ici l'expression de ma haute gratitude.

Mon très cher petit frère Ramy.

Toute ma deuxième famille de AIESEC in Blida, vous m'inspirez et vous me poussez toujours en avant. Et surtout mon Binôme TOUIL Abdelouahab pour ses énormes efforts qui nous ont porté ici.

MOHAMED MAHMOUD Abdelkarim

DÉDICACES

Dédicace Je dédie ce modeste travail aux êtres qui me sont les plus chers, A mes chers parents, pour tous leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de mes études,

A Mes cheres grandes mères ,

A mes chers grand père allah yarhamhoum ,

A mes chers frères, Rafik, Yacine, Amina pour leur appui et leur encouragement„

A mes chers oncles et tantes A mes cousins et cousines ,

Toutes mes amis, particulièrement : Fouzi, Ahmed, Mohamed, Sohaib, Yaakoub, Akram, Islem, Abderraouf, Rimane, Nassiba, Ihcene, Imene, Rania ,

A mon binome Abdelkarim pour sa patience Que ce travail soit l'accomplissement de vos vœux tant allégués, et le fuit de votre soutien infaillible, Merci d'être toujours là pour moi.

TOUIL Abdelouahab

REMERCIEMENTS

Je voudrais dans un premier temps remercier ALLAH tout puissant qui nous a donné la force de mener à bon terme ce modeste travail. Je tiens a remercier mon encadreur au CDTA Mme.SEMAR, pour sa patience infinie, sa disponibilité et surtout ses judicieux conseils, qui ont contribué à alimenter ma réflexion et m'aidé dans le cadre du projet ainsi que ailleurs. Aussi, Mme.CHIKHI Ma promotrice a Saad dahleb Blida, pour ses efforts et toutes ses contributions et corrections au long du chemin de ce travail.

Je remercie également toute l'équipe pédagogique de l'université de Saad Dahleb et du CDTA et les intervenants professionnels responsables de ma formation, pour avoir assuré que j'y arrive ici.

Je tiens à témoigner toute ma reconnaissance aux personnes suivantes, pour leur aide dans la réalisation de ce mémoire :
Monsieur BENLEMIR Walid qui m'a beaucoup appris sur les défis à relever dans le monde des affaires. et les détails les plus techniques des technologies BI, Il a partagé ses connaissances et expériences dans ce milieu, tout en nous accordant sa confiance et une large indépendance dans l'exécution de missions valorisantes.

Madame CHIKHI Meriem pour son aide a la digitalision des données commerciales, ses conseils et son support tout au long du projet elle a été d'un grand soutien dans l'élaboration de ce projet.

Dr Youcef Touil , pour la quantité énorme d'investissement de temp et d'énergie consacrée pour notre projet, et le partage de sa précieuse expertice dans le domaine d'hydraulique

Messieurs ABABSIA Anis, ABDICHE Mohand Amokrane et BOUCHOUK Ahmed pour leurs aide dans un peut partout dans le travail du coté linguistique, implémentation, organisation, style de travail .. etc, ce travail ne serait pas possible sans vous.

Table des matières

Table des figures

Liste des tableaux

INTRODUCTION GÉNÉRALE

Aujourd'hui, les systèmes décisionnels ont un impact majeur sur le gestion de différentes organisations. Ils apportent une aide non négligeable au pilotage de leurs plans d'actions, à l'apprentissage et à la réalisation d'innovations incrémentales. En effet, ils constituent un moyen pour faciliter la mise en œuvre de stratégies gagnantes. Dans le domaine économique en particulier, les systèmes décisionnels ont une grande importance dans le développement de la société, offrant un appui majeur pour les décideurs de la politique économique. Le but étant de rendre plus performant le pilotage de l'économie nationale.

Actuellement, la situation économique en Algérie fait face à des difficultés pour la prise de bonnes décisions. Ceci est dû au paradigme des données massives. La conception d'un système décisionnel évolutif fait face à une série de défis techniques ; En premier lieu la récolte et l'intégration de données variées hétérogènes d'endroits distribués. Un des plus importants aspects d'un système d'entrepôt de données est sa capacité à évoluer en fonction des besoins de l'utilisateur et de l'organisation. Le schéma d'un entrepôt de données peut évoluer pour intégrer de nouvelles exigences. Cela aboutit à un traitement d'analyse ou d'une exploration de données, de modifications de schéma sur des sources d'informations, etc en assurant de maintenir la cohérence.

Il est très important que tout système de base de données avancé fournisse des outils et des techniques pour la prise en charge de l'évolution du schéma [21], [22]. Une première étude consistant à mesurer la fréquence d'évolution des schémas a été réalisée dans le cadre d'un système de gestion de la santé [22]. Les résultats de ce travail montrent que le nombre de relations a augmenté de 139% et le nombre d'attributs de 274% et que chaque relation a été modifiée. Le deuxième rapport [21] conclut que 59% des attributs sont modifiés en moyenne.

Un autre défi est assurer l'evolutivité du système d'extraction, de transformation et de chargement des données (ETL). Il extrait les données source des systèmes orientés-opération (Système de gestion de stock, des logiciels de comptabilité..)Et transforme les données pour résoudre divers types de problèmes d'hétérogénéité et charge les données transformées dans le système cible (par exemple, un entrepôt de données). Le processus ETL est généralement un processus indépendant reliant les deux systèmes. Il est souvent réalisé par l'outil tiers qui exécute le processus ETL dans une fenêtre temporelle spécifique. ce système doit aussi être conçu d'une façon qui peut évoluer a la grande échelle sans générer un montant ingérable de complexité.

Un autre des soucis dans l'implémentation d'un tel système, est la complexité de gérer un ETL a la grande echelle via les outils courants. L'approche graphique est complexe lorsque le nombre d'étapes que l'ETL doit suivre est important, lorsque le nombre de membres de l'équipe est grand et lorsque des problèmes se produisent dans ce système.

Dans le sud de l'algérie spécifiquement les principaux sources de l'eau sont des eaux fos-

siles ou nappes fossiles ; des eaux souterrains présents dans une réserve naturelle, dite aquifère, depuis une période qui excède le temps de la civilisation humaine La durée de régénération peut être de quelques siècles, mais elle est souvent de l'ordre de la dizaine de millénaires. Dans ce cadre, l'eau fossile a alors été piégée dans des conditions climatiques, physico-chimiques, ou géomorphologiques qui ne sont plus celles actuelles, et son renouvellement ne peut s'inscrire dans le cycle de l'eau actuel. La consommation d'eau fossile est donc définitive, le stock ne pouvant se renouveler, à l'échelle de temps humain. ; à ce titre, c'est une ressource non renouvelable. ce qui rend la bonne gestion de tels ressources et l'investissement économique dans ce domaine trés important.

A travers ce travail nous avons tenté de relever ces défis par la mise en en œuvre d'un mini-système d'aide à la décision, relatif au domaine de l'hydraulique. contenant un entrepôt de données extensible pouvant intégrer plusieurs entrepots dans un seul entrepot virtuel, et un système ETL evolutif conçu sous forme des diagrammes de dépendances composé de plusieurs fonctions evolutives. Implémenté via une approche basée sur le code(code-based), aulieu d'une approche graphique. Cela qui permet la gestion de la complexité de l'extensibilité a travers les bonnes pratiques de développement logiciel.

Notre système sera utilisé dans le but de faciliter l'investissement et de faciliter le positionnement des bases de forages sur une zone du Sud de l'Algérie.

Les contributions majeures de notre travail nous les avons structurés à travers ce mémoire dans quatre chapitres, comme suit :

- Le chapitre 1 : offre une vision globale à travers les concepts de bases sur le Business Intelligence, les entrepôts de données, ainsi qu'une étude et sélection des KPIs du domaine d'application de notre travail à introduire dans le système décisionnel.

- Le chapitre 2 : présente une étude état de l'art sur l'évolution des entrepôts de données (établir des critères de comparaison, synthèse et analyse des travaux) et une proposition d'une approche de conception d'un entrepôt extensible.

- Le chapitre 3 : Conception du système décisionnel.

- Le chapitre 4 : consiste en une étude comparative de l'aspect évolutif de différents approches et outils d'implémentation des entrepôt de données et l'Implémentation et application Business Intelligence sur un cas d'étude relatif au domaine hydraulique (positionnement des forages suivant plusieurs critères à savoir : le type de nappes d'eau, débit du forage, type d'utilisation du forage, ...)et par la suite la réalisation du mini système et ses tests.

Chapitre 1

CONTEXTE ET CONCEPTS DE BASE

1.1 Introduction

Ce chapitre servira pour introduire les concepts de base sur lesquels notre travail se base comme Business Intelligence, Entrepots de données, ETL et KPI Ainsi qu'une Etude sur les KPI du domaine choisi, le domaine d'hydraulique.

1.2 Business Intelligence

La Business Intelligence (BI) comprend les stratégies et les technologies utilisées par les entreprises pour l'analyse des données d'informations commerciales. [23] Les technologies BI fournissent des vues historiques, actuelles et prédictives des opérations commerciales. Les fonctions courantes des technologies d'intelligence d'entreprise comprennent mais ne sont pas limités à :

- la création de rapports
- le traitement analytique en ligne
- l'analyse
- l'exploration de données
- l'exploration de processus
- le traitement des événements complexes
- la gestion des performances de l'entreprise
- l'analyse comparative
- l'extraction de texte
- l'analyse prédictive et l'analyse normative.

Les technologies BI peuvent gérer de grandes quantités de données structurées et parfois non structurées pour aider à identifier, développer et créer de nouvelles opportunités commerciales et stratégiques. Ils visent à permettre une interprétation facile de ces données volumineuses. L'identification de nouvelles opportunités et la mise en œuvre d'une stratégie efficace basée sur des informations peuvent fournir aux entreprises un avantage concurrentiel et une stabilité à long terme. [24]

Les entreprises peuvent utiliser la BI pour prendre en charge un large éventail de décisions, qu'elles soient opérationnelles ou stratégiques. Les décisions opérationnelles de base incluent par exemple le positionnement ou la tarification du produit. Les décisions commerciales stratégiques impliquent des priorités, des objectifs et des orientations au niveau le plus large. Dans tous les cas, la BI est plus efficace lorsqu'elle combine des données issues du marché sur lequel une entreprise exerce ses activités (données externes) avec des données provenant de sources internes à l'entreprise, telles que des données financières et opérationnelles (données internes). Lorsqu'elles sont combinées, les données externes et internes peuvent fournir une image complète qui crée en réalité une "intelligence" qui ne peut être dérivée d'aucun ensemble de données unique.[25]

Parmi les innombrables utilisations, les outils de BI permettent aux organisations de mieux comprendre de nouveaux marchés, d'évaluer la demande et la pertinence des produits et services pour différents segments de marché et d'évaluer l'impact des efforts de marketing.[26] Les applications BI utilisent souvent des données recueillies à partir d'un entrepôt de données (DW) ou d'un data mart(magazin de données), et les concepts de BI et DW se combinent en tant que "BI / DW" [27] ou en tant que "BIDW". Un entrepôt de données contient une copie des données analytiques facilitant l'aide à la décision. Ces données sont souvent sous forme de KPIs (Key performance indicators ou Indicateurs de performance clés) qui représente les facteurs les plus importants dans la prise de décision.

1.3 Entrepôts de données

Un entrepôt de données (DW : Data warehouse ou DWH), également appelé entrepôt de données d'entreprise (EDW : Enterprise Data warehouse), est un système utilisé pour la création de rapports et l'analyse de données, et est considéré comme un composant essentiel du BI. Les DW sont des référentiels centraux de données intégrées provenant d'une ou de plusieurs sources hétérogènes. Ils stockent les données actuelles et historiques dans un emplacement unique. Ils sont utilisés pour créer des rapports analytiques pour les décideurs et même les travailleurs de l'ensemble de l'entreprise.

Les données stockées dans l'entrepôt sont téléchargées à partir des systèmes opérationnels (tels que le marketing ou les ventes). Les données peuvent passer par un magasin de données opérationnel et peuvent nécessiter un nettoyage des données pour des opérations supplémentaires afin de garantir la qualité des données avant leur utilisation dans le DW pour la génération de rapports. [28]

L'entrepôt de données typique basé sur l'extraction, la transformation, le chargement (ETL :extract, transform and load) (voir section 1.4 ETL) utilise des couches de mise en scène (staging), d'intégration de données et d'accès pour héberger ses fonctions clés. La couche intermédiaire ou la base de données intermédiaire stocke les données brutes extraites de chacun des systèmes de

données sources hétérogenes. La couche d'intégration intègre les ensembles de données disparates en transformant les données de la couche de stockage intermédiaire, stockant souvent ces données transformées dans une base de données de stockage de données opérationnelles (ODS). Les données intégrées sont ensuite déplacées vers une autre base de données, souvent appelée base de données d'entrepôt de données, dans laquelle les données sont organisées en groupes hiérarchiques, souvent appelés dimensions, et en faits et faits agrégés. La couche d'accès aide les utilisateurs à récupérer des données. [29]

1.3.1 Stockage de l'information

a. Faits

Un fait est une valeur, ou une mesure, qui représente un fait à propos de l'entité ou du système géré. c'est Les faits, sont dits au niveau brut, et souvent représentent des indicateurs de performances clés (KPIs) Une table de faits comprend les mesures, les métriques ou les faits d'un processus métier. elle est situé au centre d'un schéma en étoile ou d'un schéma en flocon de neige entouré de tables de dimensions. Lorsque plusieurs tables de faits sont utilisées, elles sont organisées comme un schéma de constellation de faits. Une table de faits comporte généralement deux types de colonnes : celles qui contiennent des faits et celles qui constituent une clé étrangère pour les tables de dimensions. La clé primaire d'une table de faits est généralement une clé composite composée de toutes ses clés étrangères. Les tables de faits contiennent le contenu de l'entrepôt de données et stockent différents types de me sures, telles que des mesures additives, non additives et semi-additives. Les tables de faits fournissent les valeurs (généralement) additives qui agissent comme des variables indépendantes par lesquelles les attributs dimensionnels sont analysés. Les tables de faits sont souvent définies par leur grain. Le grain d'une table de faits représente le niveau le plus atomique permettant de définir les faits. Le grain d'un tableau factuel des ventes peut être défini comme "volume des ventes par jour par produit par magasin". Chaque enregistrement de cette table de faits est donc uniquement défini par un jour, un produit et un magasin. Les autres dimensions peuvent être membres de cette table de faits (comme lieu / région), mais elles n'ajoutent rien à l'unicité des enregistrements de faits. Ces "dimensions de filiale" permettent des tranches supplémentaires de faits indépendants, mais fournissent généralement des informations à un niveau d'agrégation plus élevé (une région contient de nombreux magasins).

b. Approche dimensionnelle

L'approche dimensionnelle fait référence à l'approche de Ralph Kimball dans [16] dans laquelle il est indiqué que l'entrepôt de données doit être modélisé à l'aide d'un schéma modèle dimensionnel / étoile. L'approche normalisée, également appelée modèle 3NF (troisième forme normale), fait référence à l'approche de [30] dans laquelle il est indiqué que l'entrepôt de données doit être modélisé à l'aide d'un modèle E-R / modèle normalisé.

Dans une approche dimensionnelle, les données de transaction sont partitionnées en "faits", qui sont généralement des données de transaction numériques, et en "dimensions", qui sont les informations de référence qui donnent le contexte aux faits. Par exemple, une transaction de vente peut être divisée en faits tels que le nombre de produits commandés et le prix total payé

pour les produits, et en dimensions telles que la date de commande, le nom du client, le numéro de produit, le traitement de la livraison et la facturation. emplacements, et vendeur responsable de la réception de la commande.

L'un des principaux avantages d'une approche dimensionnelle est que l'entrepôt de données est plus facile à comprendre et à utiliser pour l'utilisateur. En outre, l'extraction des données de l'entrepôt de données a tendance à fonctionner très rapidement [8]. Les structures dimensionnelles sont faciles à comprendre pour les utilisateurs professionnels, car elles sont divisées en mesures / faits et en contexte / dimensions. Les faits sont liés aux processus opérationnels et au système opérationnel de l'organisation, tandis que les dimensions qui les entourent contiennent un contexte relatif à la mesure (Kimball, Ralph 2008). Un autre avantage offert par le modèle dimensionnel est qu'il n'implique pas à chaque fois une base de données relationnelle. Ainsi, ce type de technique de modélisation est très utile pour les requêtes d'utilisateur final dans un entrepôt de données.

Le modèle des faits et des dimensions peut également être compris comme un cube de données. Où les dimensions sont les coordonnées catégorielles dans un cube multidimensionnel, alors que le fait est une valeur correspondant aux coordonnées.[16] Les principaux inconvénients de l'approche dimensionnelle sont les suivants :

Pour préserver l'intégrité des faits et des dimensions, le chargement de l'entrepôt de données avec des données provenant de différents systèmes opérationnels est compliqué. Il est difficile de modifier la structure de l'entrepôt de données si l'organisation qui adopte l'approche dimensionnelle modifie sa manière de travailler.

1.4 ETL

Extraire, transformer, charger (ETL ou Extract Transform and Load) est la procédure générale de copie de données d'une ou de plusieurs sources dans un système de destination qui représente les données différemment de la ou des sources. Le processus ETL est devenu un concept populaire dans les années 1970 [31] et est souvent utilisé dans l'entreposage de données [32].

L'extraction de données consiste à extraire des données de sources homogènes ou hétérogènes ; la transformation des données traite les données en les nettoyant et en les transformant en un format / structure de stockage approprié aux fins de l'interrogation et de l'analyse ; enfin, le chargement de données décrit l'insertion de données dans la base de données cible finale, telles qu'un magasin de données opérationnel, un magasin de données ou un entrepôt de données. [33]

Un système ETL correctement conçu extrait les données des systèmes sources, applique des normes de qualité et de cohérence des données, les conforme afin que des sources distinctes puissent être utilisées ensemble, et fournit enfin les données dans un format prêt pour la présentation afin que les développeurs d'applications puissent créer des applications et les utilisateurs finaux. peut prendre des décisions. [34]

Comme l'extraction de données prend du temps, il est courant d'exécuter les trois phases en parallèle. Pendant l'extraction des données, un autre processus de transformation s'exécute pendant le traitement des données déjà reçues et le prépare au chargement tandis que le chargement des données commence sans attendre la fin des phases précédentes.

Les systèmes ETL intègrent généralement les données de plusieurs applications (systèmes), développées et prises en charge par différents fournisseurs ou hébergées sur du matériel infor-

matique distinct. Les systèmes distincts contenant les données d'origine sont fréquemment gérés et exploités par différents employés. Par exemple, un système de comptabilisation des coûts peut combiner des données de paie, de ventes et d'achats.

1.5 Indicateurs de performance clés(KPI)

Indicateur de performance clé (IPC ou Key performance Indicator KPI) est un type de mesure de la performance. [35] Les indicateurs de performance clés évaluent le succès d'une organisation ou d'une activité particulière (projets, programmes, produits et autres initiatives) dans laquelle elle s'engage.

Le succès est souvent simplement la réalisation répétée et périodique d'un certain niveau d'objectif opérationnel (par exemple, zéro défaut, 10/10 de satisfaction du client, etc.), d'autres fois, le succès est défini en termes de progression vers un objectif stratégique. Par conséquent, le choix des indicateurs de performance clés appropriés repose sur une bonne compréhension de ce qui est important pour l'organisation. Ce qui est considéré comme important dépend souvent de la façon dont le service mesure la performance - par exemple. les KPI utiles à financer seront différents des KPI affectés aux ventes.

Puisqu'il est nécessaire de bien comprendre ce qui est important, diverses techniques permettant d'évaluer la situation actuelle de l'entreprise et ses activités clés sont associées à la sélection d'indicateurs de performance. Ces évaluations conduisent souvent à l'identification d'améliorations potentielles, de sorte que les indicateurs de performance sont systématiquement associés aux initiatives d'amélioration de la performance. Un moyen très courant de choisir des indicateurs de performance clés consiste à appliquer un cadre de gestion tel que le tableau de bord prospectif. Identifier les indicateurs d'une organisation Les indicateurs de performance diffèrent des facteurs opérationnels et des objectifs (ou objectifs). Une école peut considérer le taux d'échec de ses élèves comme un indicateur clé de performance susceptible de l'aider à comprendre sa position dans la communauté éducative, alors qu'une entreprise peut considérer le pourcentage de revenu provenant de clients fidèles comme un indicateur de performance clé potentiel.

Les principales étapes de l'identification des indicateurs de performance clés sont les suivantes :

- Avoir un processus métier prédéfini.
- Avoir des exigences pour les processus métier.
- Avoir une mesure quantitative / qualitative des résultats et une comparaison avec les objectifs fixés.
- Enquêter sur les écarts et peaufiner les processus ou les ressources pour atteindre des objectifs à court terme.

Les indicateurs de performance clés (KPI) sont des moyens d'évaluer périodiquement les performances des organisations, des divisions et de leurs divisions, services et employés. En conséquence, les indicateurs de performance clés sont le plus souvent définis de manière compréhensible, significative et mesurable. Ils sont rarement définis de telle manière que leur réalisation

serait entravée par des facteurs considérés comme non contrôlables par les organisations ou les individus responsables. Ces indicateurs de performance sont généralement ignorés par les organisations.[36]

Les indicateurs de performance clés doivent suivre les critères SMART. Cela signifie que la mesure a un objectif spécifique pour l'entreprise, qu'il est mesurable d'obtenir réellement une valeur de l'indicateur de performance clé, que les normes définies doivent être atteignables, que l'amélioration d'un indicateur de performance clé doit être pertinente pour le succès de l'organisation, et enfin elle doit être échelonnée dans le temps, ce qui signifie que la valeur ou les résultats sont affichés pour une période prédéfinie et pertinente.

Pour être évalués, les KPI sont liés à des valeurs cibles, de sorte que la valeur de la mesure puisse être évaluée en fonction des attentes ou non.

1.6 Domaine d'application et Choix des KPI

Toute entreprise ou organisation se fixe des KPI à court et long terme, sans la prise de bonnes mesures et sans le suivi de ces dernières, ces entreprises sont amenées à traiter les symptômes d'un problème sans en comprendre la source. Il en va de même pour l'évaluation des réussites. L'association des objectifs fixés à un ensemble d'informations contribuant à l'appréciation d'une situation par le décideur, permet d'acquérir une vision claire des facteurs qui ont un impact sur l'atteinte d'objectifs. Le besoin de définir les bonnes mesures fait apparaître le terme (KPI) les indicateurs clés de performance.

Dans ce chapitre nous allons définir les (KPI), leurs utilités et en citer quelques exemples en Algérie ainsi que ceux utilisés dans notre travail.

1.6.1 Différents types de KPIs

Pour faciliter l'utilisation et mieux en cerner l'usage il est habituel de classer les indicateurs selon 3 catégories en relation avec le type d'information transmise et les attentes du décideur.

a. Anticipation

Un bon tableau de bord est aussi un instrument de prospective. Avec quelques indicateurs "d'anticipation", il permet de voir un peu plus loin que le bout de son écran et d'envisager avec une meilleure assise la situation actuelle.

b. Equilibration

Ce type d'indicateur de performance étroitement lié aux objectifs est un peu la boussole du décideur. Il informe sur l'état du système sous contrôle en relation avec les objectifs suivis.[37]

c. Alerte

Ce type d'indicateur signale un état anormal du système sous contrôle nécessitant une action, immédiate ou non. Un franchissement de seuil critique par exemple entre dans cette catégorie d'indicateur.[37]

1.6.2 Exemples sur les KPI (Key Performance Indicators) en Algérie

Il existe plusieurs types d'indicateurs de performance clés, et ceci en fonction du secteur d'activité et du service souhaité à suivre. Nous avons effectué une étude sur les différents KPI existants en Algérie, dans le domaine de l'économie, l'agriculture, la météo que nous exposons dans l'annexe (. . .) et en dessous nous détaillons ceux propres au cas d'étude sur lequel nous avons travaillé et qui est relatif au domaine de l'hydraulique et les forages des pompes à eau.

a. EAU (hydraulique)

Profondeur des précipitations

Moyenne à long terme (dans l'espace et dans le temps) des précipitations endogènes annuelles (produites dans le pays) en profondeur.

Indice national de pluie

0.98in0.0in Un indicateur, développé par la FAO, qui représente la qualité de la saison de croissance des cultures. Les résultats annuels de NRI prennent en compte les précipitations de cette année, les précipitations moyennes sur la période 1986-2000, le caractère saisonnier de la principale saison de croissance des cultures (distinction entre les hémisphères nord et sud) et les zones du pays plus humides. La médiane de chaque période de cinq ans est fournie. Note : en raison de différences méthodologiques, cette variable n'est pas comparable à la précipitation moyenne.

b. Précipitations

Gardons le même échantillon de wilaya et on calcule la précipitation moyenne par jour et la précipitation totale sur le mois

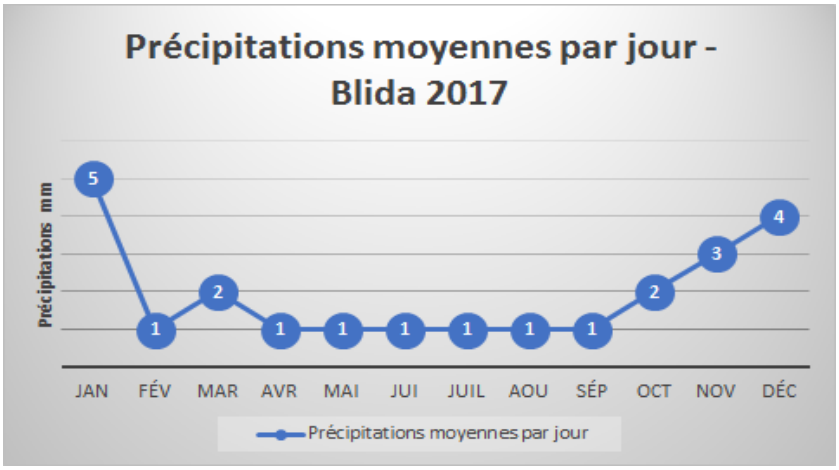


FIGURE 1.1 – Précipitations moyennes par jour - Blida

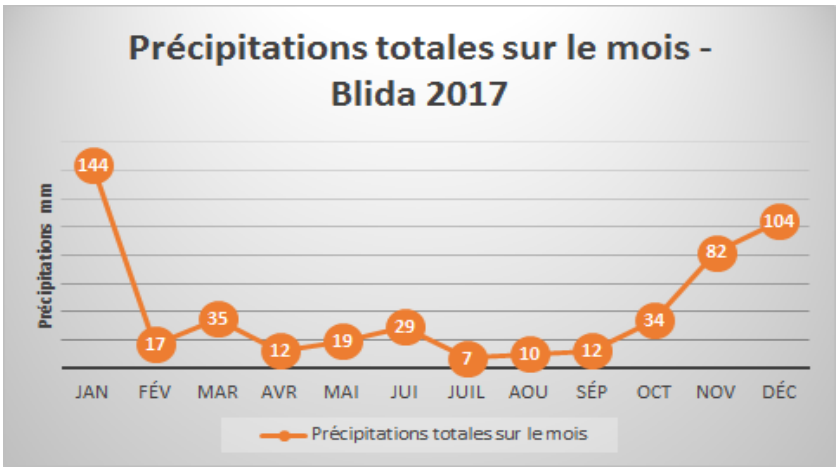


FIGURE 1.2 – Précipitations Totales sur le mois-Blida

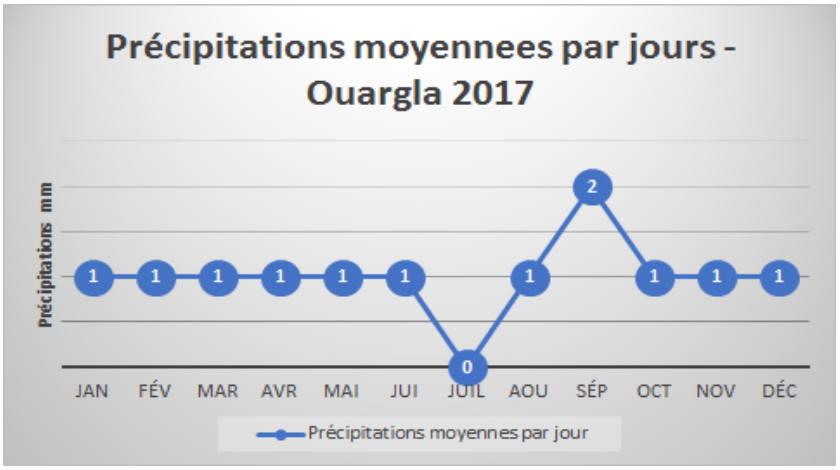


FIGURE 1.3 – Précipitations Moyennes par jours - Ourgla

c. Envasement des barrages

En Algérie, les 52 grands barrages reçoivent 32 millions de m3 de matériau solide annuellement. La répartition des barrages sur les cinq bassins hydrographiques indiquent clairement que les barrages de la région de Chéllif –Zahrez sont les barrages les plus menacés par le phénomène de l'envasement, puisque le taux de sédimentation annuel est de 0,75% (fig. 3). Ceci est dû à la forte érosion des bassins versants de la région, favorisé par la nature des sols et l'absence de boisement. Même pour les petits barrages, le taux de comblement évalué en 2002 dans le bassin hydrographique Chellif –Zahrez est de 16% de la capacité totale, il est beaucoup plus grand par rapport à celui des autres régions.

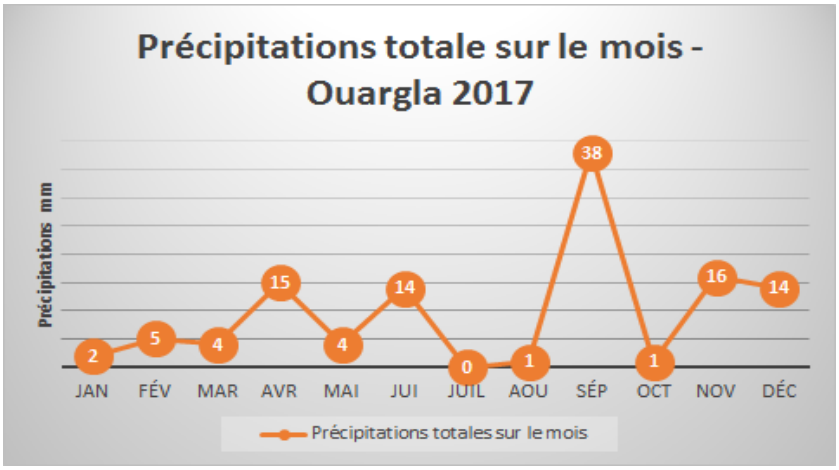


FIGURE 1.4 – Précipitations Totales sur le mois-Ourgla

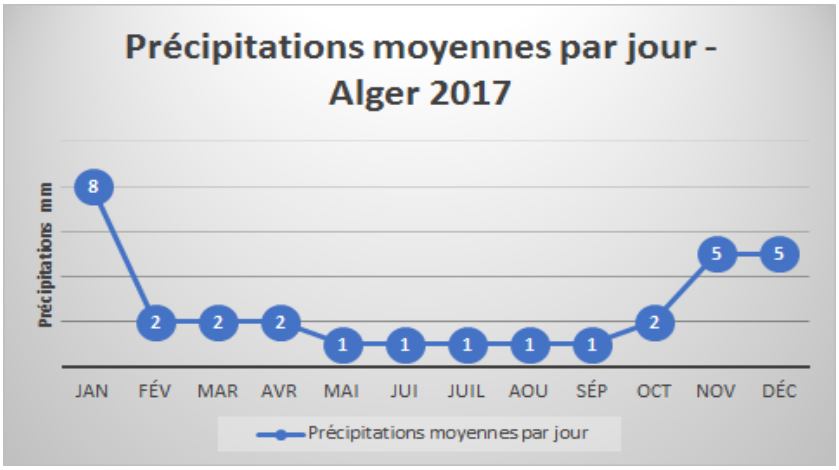


FIGURE 1.5 – Précipitations Moyennes par jours - Alger

d. Eaux souterraines

L'évaluation des ressources souterraines est basée sur les résultats des études des principales nappes du pays et des estimations basées sur une approche de l'infiltration de la pluie. Les données sur les eaux souterraines datent parfois de plusieurs années et nécessitent une actualisation, tandis que les nappes connues, uniquement par des estimations sur l'infiltration, devraient faire l'objet d'études complètes avec une modélisation des écoulements et l'établissement de bilans. Le tableau suivant est une répartition des ressources en eau souterraines par région.

tableau ici ! !

e. Les eaux superficielles

Les ressources en eau superficielle en Algérie sont actuellement évaluées entre 9,8 à 13,5 milliards de m3.

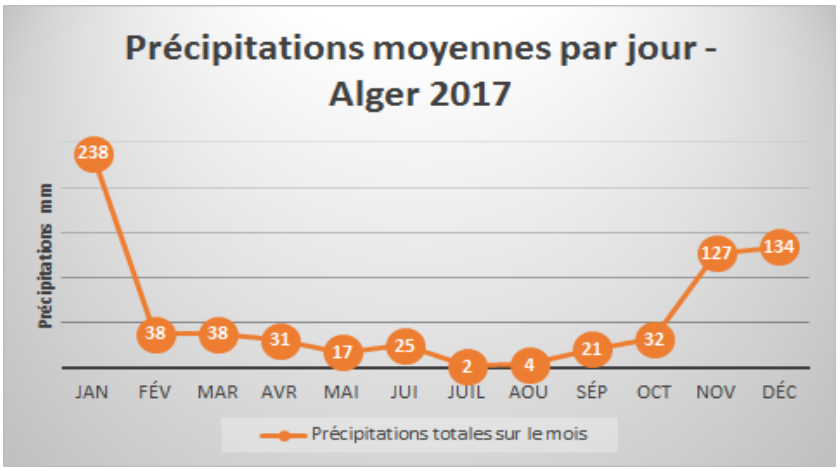


FIGURE 1.6 – Précipitations Totales sur le mois-Alger

le tableau suivant représente la répartition des ressources en eau superficielle par région.

tableau ici ! !

f. Les nappes superficielles des eaux

Les couches traversées comprennent des séries alternantes de sables, de graviers et d'argiles, au milieu desquels on rencontre ordinairement plusieurs nappes d'eau ascendantes ou jaillissantes, parfois saumâtres, alimentées par les infiltrations pluviales et les ruisseaux qui viennent se perdre dans ces terrains absorbants.

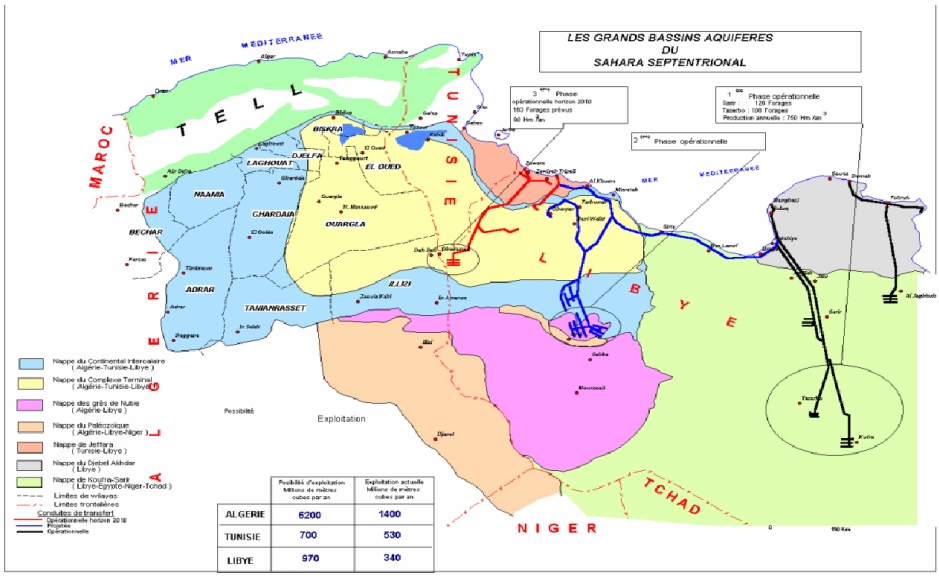


FIGURE 1.7 – Les grands bassins aquiferes du sahara septentrional [15]

1.6.3 Les nappes en Algérie

Il existe trois types de nappes en Algérie : les nappes phréatiques, nappes du continental intercalaire et complexe terminal :

1. **Nappes phréatiques**

   Cette nappe est la moins utilisée en Algérie, à cause de la pollution causée par la faible profondeur de 20 m maximum et parce qu'elle est très salée, aussi à cause de la faible quantité en eau dans ces nappes donc ça sert à rien de l'extraire, parmi les caractéristiques d'eau de ces nappes : une forte température de l'eau pouvant atteindre les 60° ou même plus, cela nécessite un équipement pour la refroidir.

2. **Nappes du Continental Intercalaire**

   Ces nappes sont les plus profondes avec 2000 m de profondeur et les plus utilisées avec une surface d'un million de Km^2 devisée entre l'Algérie, la Libye et la Tunisie avec une surface de 650000 km^2 en Algérie, l'eau de ces nappes se caractérise par la température basse par rapport à celle de la nappe phréatiques et les nappes du Complexe Terminal.

   Le Continental intercalaire est composé de nappe Albien, nappe Aptien et nappe Barrémien toutes ces nappes appartiennent à la nappe d'Albien.

3. **Nappes du Complexe Terminal**

Avec une surface de 300000 km^2 le deuxième plus profond type de nappes de profondeur atteindre 1100 m Complexe Terminal est composé de :

1. Nappe des sables contient la nappe Mio-Pliocène.
2. Premier imperméable contient la nappe Eocène évaporitique.
3. Nappe des calcaires (séneno-Eocène) contient la nappe Eocène carbonaté et le Sénonien carbonaté.
4. Deuxième imperméable contient la nappe sénonien lagunaire.
5. Nappe turonien.
6. Troisième imperméable contient la nappe cénomanien et la nappe Vraconien.

1.6.4 Les forages

Les forages des ouvrages permettant de capter les eaux souterraines (points d'eau), on peut les nommer aussi des puits ou des sources d'eau.

Les forages sont caractérisés par : le débit, niveau statique, niveau dynamique, la nappe exploité et le type d'utilisation.

1. **Le débit :**

   Le débit d'eau est la capacité d'un forage à fournir une certaine quantité d'eau, Il est calculé par unité de temps ; en nombre de litres par heure ou par minute, mais souvent se quantifie en m3/h.

   Les forages sont connus par trois débits :

   (a) Débit mobilisé : débit qui peut être donné par le forage.

   (b) Débit d'exploitation (débit de la pompe) : le débit pompé par le forage.

   (c) Débit utilisé : le débit consommé.

2. **Le niveau statique :**
   Le niveau statique (Ns) d'un puits ou d'un forage est la distance du sol par rapport à la surface de l'eau avant pompage.

3. **Le niveau dynamique :**

   Le niveau dynamique (Nd) d'un puits ou d'un forage est la distance du sol par rapport à la surface de l'eau pour un pompage à un débit donné.
   La différence entre le niveau dynamique et le niveau statique est appelée rabattement. (RM) est le rabattement maximal acceptable avant de stopper la pompe (la perte de puit).

4. **La nappe exploitée :**
   Ce paramètre

5. **Le type d'utilisation :**

Les eaux souterraines dans le Sahara sont la seule source d'alimentation en eau soit l'irrigation ou AEP (alimentation d'eau potable)

Un tableau qui représente les caractéristiques des forages d'irrigation de la palmeraie d'Ouargla est dans l'annexe (. . .).

1.7 Conclusion

Dans ce chapitre, nous avons défini les concepts de base en ce qui concerne BI, les entrepôts de données, et les indicateurs de performances clés. ces trois concepts fonctionnent ensemble dans les mêmes systèmes et offrent une très grande puissance et avantage pour le décideur. Ce chapitre était aussi une démonstration des indicateurs de performance clés utilisés en Algérie par les décideurs

Comme dans notre travail nous avons décidé de se concentrer sur le domaine de l'hydraulique, et ceci au niveau de la région sud, vu l'importance des décisions et les investissements dans ce domaine et dans cette région spécifiquement. Dans ce qui suit on étudiera les travaux existants qui traitent la problématique de l'extensibilité dans les entrepôts de données pour pouvoir implémenter ces KPI dans un systeme de BI extensible.

Chapitre 2

EXTENSIBILITÉ DANS LES ENTREPOTS DE DONNÉES

2.1 Introduction

La technologie DW a été développée pour intégrer des sources d'informations hétérogènes à des fins d'analyse. Les sources d'informations sont de plus en plus autonomes et changent souvent de contenu en raison de transactions perpétuelles (modifications de données) et peuvent changer de structure en raison de l'évolution constante du monde réel et des exigences des utilisateurs généralement et du monde économique spécifiquement. C'est un monde qui par nature évolue et grandit à une vitesse exponentielle. Gérer correctement tous les types de changements est très nécessaire. En fait, le DW, est considéré comme l'élément central des systèmes modernes de BI, il doit être mis à jour en fonction des différents types d'évolution des sources d'informations afin de refléter le monde réel soumis à l'analyse.

Dans ce deuxième chapitre nous proposerons une vue d'ensemble et une étude comparative entre différents travaux liés au problème de l'évolution de DW, et pour cela nous allons expliquer la méthode exploitée dans chaque travail et à la fin nous effectuons une comparaison entre les différentes approches.

2.2 Travaux sur l'extensibilité des entrepots de données

2.2.1 Travail de Lehner [1]

Lehner dans ce travail, considère les faits comme partie dynamique et les dimensions comme des entités statiques.

Le modèle multidimensionnel classique a été étendu en regroupant des attributs fonctionnellement dépendants dans des dimensions uniques, pour obtenir des dimensions orthogonales réelles, faciles à créer et à gérer au niveau de la conception du schéma.

Au cours de la phase d'analyse multidimensionnelle des données, cette technique donne des cubes de données imbriqués qui reflètent un processus de navigation intuitif en deux étapes :

1. Opérateurs de "drill-down" / “roll up"

   (a) Forage vers le bas (drill-down) : descendre dans la hiérarchie de la dimension (Ex. visualiser les ventes par mois que par année).

   (b) Forage vers le haut (drill-up, roll-up) : remonter dans la hiérarchie de la dimension (Ex. visualiser les ventes par année que par mois).

2. Des opérateurs "split" / de “merge” orientés vers la description sur des cubes de données.

   (a) Split : L’opérateur qui ajoute une fonctionnalité valide à l’ensemble des attributs dimensionnels.

   (b) Merge : L’opérateur de fusion supprime un attribut dimensionnel spécifique.

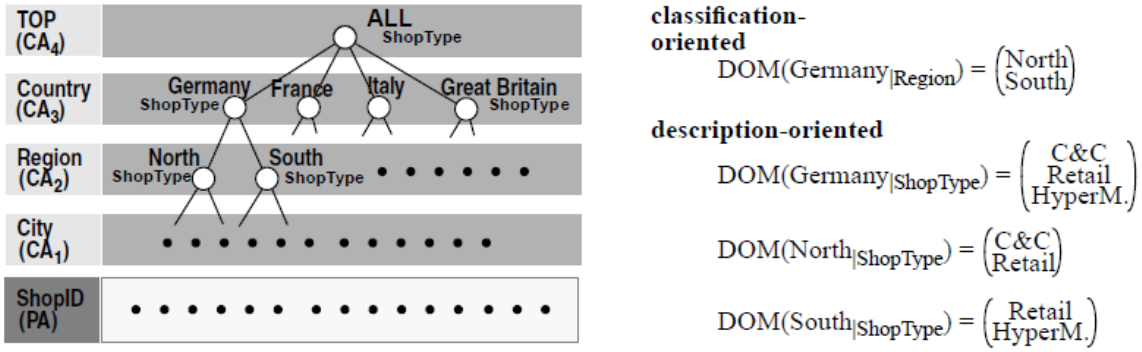


FIGURE 2.1 – Exemple de l’hiérarchie de classification et attributs dimensionnels [1]

Ainsi, leur MODÈLE DE DONNÉES MULTIDIMENSIONNELLES NICHÉES (Nested multidimensional data model) offre une grande souplesse de modélisation pendant la phase de conception du schéma et une restrictivité axée sur l’application pendant la phase d’analyse des données.

Cette approche conduit à des dimensions fonctionnellement indépendantes au niveau de la conception du schéma et à des cubes de données de faible dimension mais imbriqués au cours du processus d’analyse.

Lehner, critique sa propre approche en disant que bien que le modèle de données multidimensionnel imbriqué proposé permet la modélisation flexible de scénarios d’application statistique et scientifique, il reste quelques problèmes en suspens qui doivent être résolus : Du point de vue de la modélisation, la gestion des versions de structures dimensionnelles doit être prise en charge par le modèle de données. Etant donné que tous les modèles multidimensionnels gèrent

la dimension temporelle de la même manière que les autres dimensions, il convient de prendre en compte les caractéristiques particulières du temps et de transférer des mécanismes tels que le "temps de transaction / validité" à partir de différents modèles temporels. Du point de vue de la mise en œuvre, l'optimisation des requêtes basée sur la redondance, c'est-à-dire la prise en charge des pré-agrégats, reflète la nécessité majeure d'un processus d'analyse ad hoc efficace. Par conséquent, les travaux existants doivent être étendus au modèle de cubes de données imbriqués et sensibles au contexte. Néanmoins, ils pensent que cette approche de modélisation constitue une base adéquate pour résoudre tous ces problèmes.

2.2.2 Travail de Zhang & Rundensteiner [2]

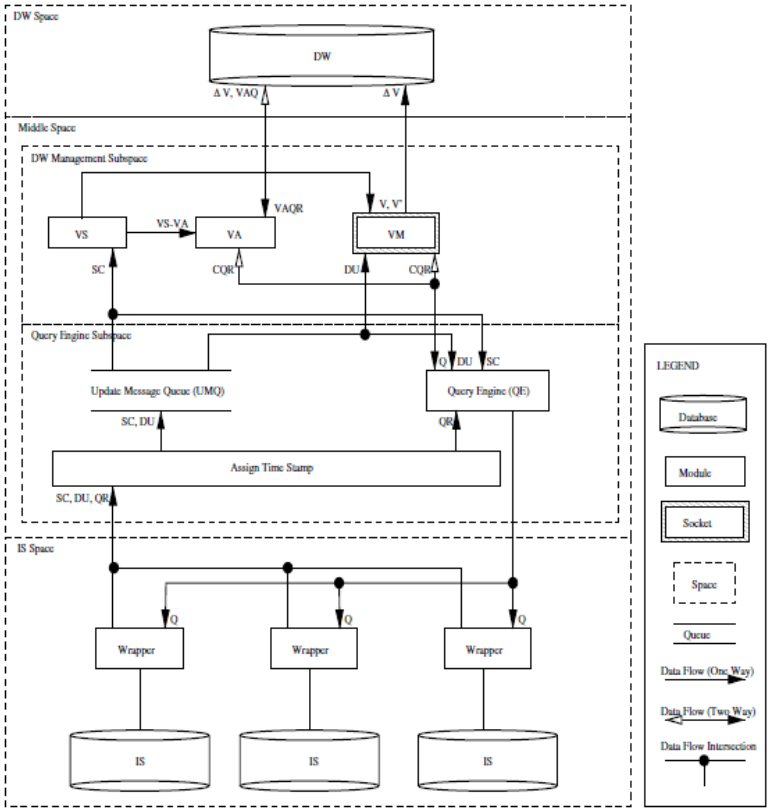
Dans des environnements dynamiques comme le web, les entrepôts de données doivent être mises à jour constamment, ce qui crée le problème de gestion de l'entrepôt de données sous mise à jour concurrentes, le système EVE (Evolvable View Environment, [38]) était le premier à traiter les changements de schémas non-concurrentes des sources de données.

Zhang et Rundensteiner traitent le problème de la concurrence des changements des schémas des différentes sources de données, et la concurrence des mises à jour de données avec les changements de schémas non concurrents. Dans ce travail un framework solution est proposé, appelé DyDa (Dynamic Data warehouse) qui résout les deux problèmes avec succès, en détectant les changements de schémas non concurrents, à travers les broken query scheme, et les conflits des mises à jour de données concurrentes à l'aide des timestamps.

L'architecture double couche de DyDa sépare les Mise à jours de données concurrentes et le traitement des changements de schémas sans imposer aucune restriction sur l'autonomie ou sur l'exécution concurrente des sources de données.

L'architecture du système DyDa est Illustrée dans la figure suivante :

DyDa surmonte la limitation de la seule approche précédente en abandonnant son hypothèse restrictive de coopération des systèmes d'information [39]



Meaning of Symbols Used in Figure 4.

Symbols	Meaning
V	View definition affected by either Schema Change (SC) or Data Update (DU).
V'	Evolved view definition of affected view V.
DW	Data warehouse.
ΔV	Incremental view extent of data warehouse, i.e., set of tuples to be inserted into or removed from the extent of view V.
CQR	Query result that is returned by QE.
VS-VA	All information VA module requires from VS module for view adaptation. It includes: V, V', Meta-knowledge, Synchronization-mapping.
VAQ	View Adaptation Query.
VAQR	View Adaptation Query Result.

FIGURE 2.2 – Architecture du Framework DyDa [2]

2.2.3 Travail de Bellahsene [3]

Bellahsene aborde les problèmes liés à l'évolution et à la maintenance des systèmes d'entreposage de données lorsque les sources de données sous-jacentes modifient leurs capacités de schéma. Ces modifications peuvent invalider les vues sur le système d'entreposage de données. Nous présentons une approche permettant d'adapter dynamiquement les vues en fonction des modifications de schéma survenant au niveau des relations source. Ce type de maintenance concerne à la fois le schéma et les données de l'entrepôt de données. Le principal problème consiste à éviter le recalcule de vues à partir de zéro, en particulier lorsque les vues sont définies à partir de plusieurs sources. Les données de l'entrepôt de données sont principalement utilisées dans la prise de décision organisationnelle et peuvent être stratégiques. Par conséquent, le schéma de l'entrepôt de données peut évoluer pour permettre la modélisation de nouvelles exigences résultant d'un traitement d'analyse ou d'exploration de données. Cette approche permet de prendre en charge l'évolution du schéma de l'entrepôt de données indépendamment des sources de données.

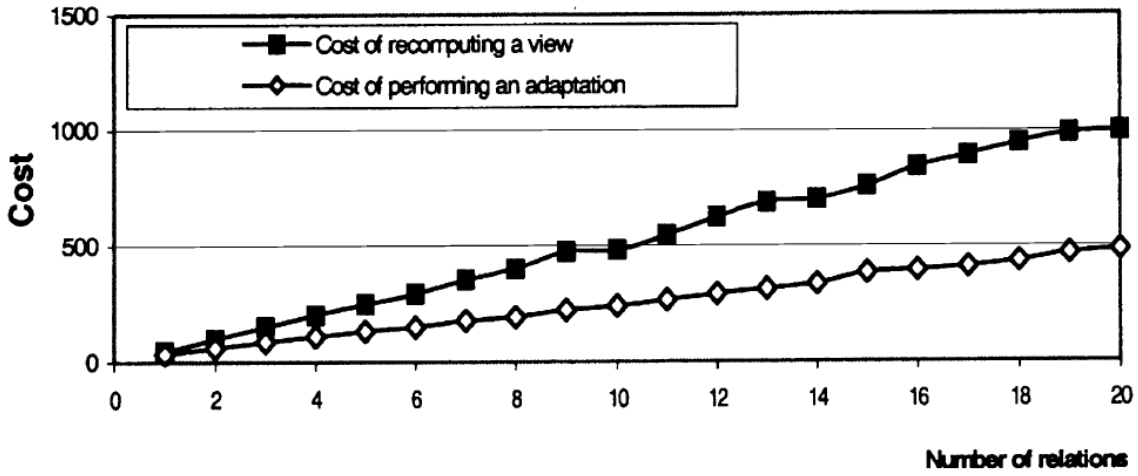


FIGURE 2.3 – Adaptation des vues aprés l'ajout d'attributs [3]

Comme on peut voir dans la figure ci-dessus, l'approche de Bellahcene surpasse celle du recalcule des vues. La raison en est que le recalcule doit accéder aux relations source afin d'effectuer les jointures impliquées dans la requête de vue, alors que cet approche doit simplement faire des recherches à partir de la matérialisation de la vue pour ajouter les nouvelles valeurs d'attribut. Dans ce cas, l'algorithme de Bellahcene d'adaptation n'a pas accès aux sources, de plus,l'approche prend en charge les modifications du schéma de l'entrepôt de données indépendamment des sources de données. Ces modifications ne doivent pas être propagées aux sources de données car elles sont autonomes. Enfin, un prototype validant cette approche a été implémenté sur le système de base de données Oracle V.7.3.

L'une des décisions les plus importantes lors de la conception d'un entrepôt de données est la sélection des vues matérialisées à gérer dans l'entrepôt. Le but est de sélectionner un ensemble approprié de vues qui minimise le temps de réponse total de la requête et/ou le coût de maintenance des vues sélectionnées, compte tenu d'un nombre limité de ressources telles que l'espace de stockage ou le temps total de maintenance des vues [40]

2.2.4 Travail de Taktak, Feki, & Zurfluh [4]

Dans ce travail, une démarche dirigée par les modèles a été proposée, pour automatiser la propagation de l'évolution du modèle de la source de données relationnelle vers l'entrepôt. Cette démarche est fondée sur deux modèles d'évolution ainsi qu'un

ensemble de règles de transformation formalisées en Query/View/Transformation. Un prototype logiciel nommé DWE (« Data Warehouse Evolution ») qui supporte cette démarche a été développé dans ce travail.

Deux modèles d'évolution décrivant à la fois les aspects structurels des données et comportementaux ont été proposées.

Parmi les règles de transformations, Celle d'ajout d'une table à la source de données, en illustrant à travers des interfaces de DWE l'effet de cette règle sur l'ED multidimensionnel assurant ainsi le passage entre les deux modèles.

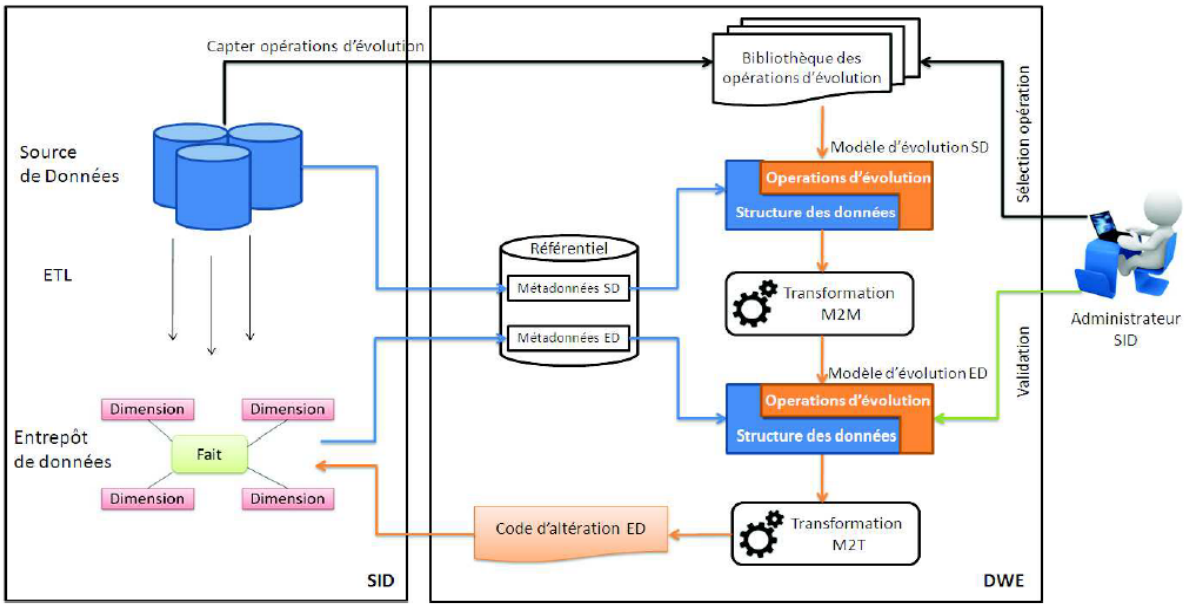


FIGURE 2.4 – Architecture du prototype DWE [4]

.

Ils ont abordé le problème d'évolution des systèmes d'information décisionnels en adoptant l'Architecture Dirigée par les Modèles (« MDA : Model Driven Architecture »). Ce choix est motivé par le fait que MDA présente un support souple et efficace pour la gestion des évolutions. En effet, le processus d'évolution d'un entrepôt permet d'épargner des efforts, de réduire les délais et les coûts et améliore la qualité lorsqu'il est traité à un niveau d'abstraction élevé, c'est-à-dire en utilisant des modèles. Ceci est particulièrement avantageux car MDA offre des mécanismes de transformations automatiques entre modèles situés à différents niveaux d'abstraction, contrairement aux approches classiques qui touchent directement le niveau implantation. De plus, MDA permet de fournir un support pour l'évolution, l'intégration, l'interopérabilité, l'adaptabilité, la portabilité et la réutilisabilité des systèmes d'information.

2.2.5 Travail de Guerrero [5]

Ce travail s'intéresse à l'évolution du schéma des entrepôts de données, Guerrero et Edgan-Ivàn ont réalisés une étude approfondie de l'état de l'art du domaine des entrepôts qui leur a permis d'identifier les besoins des utilisateurs en matière d'évolution, les contributions de ce travail est détaillé par la suite :

a. Un modèle multidimensionnel des données

Ce modèle est caractérisé par la séparation claire du schéma et de l'instance des dimensions et des cubes et par la complexité des dimensions, qui peuvent avoir de multiples hiérarchies alternatives, qui peuvent être déterminés par un ou plusieurs axes et peuvent contenir une ou plusieurs mesures.

16 Opérateurs d'évolution de schémas multidimensionnels ont été proposés, ces derniers comprennent la création de nouveaux schémas de dimensions et schémas de cube, la suppression d'ancien schémas, et la modification des schémas existants, ils assurent le passage des schémas d'un état cohérent à un autre.

b. Le langage MDL

MDL (Multidimensional Data Definition Language), un langage associé au modèle multidimensionnel de données qui a était proposé. comme un moyen de communication entre un gestionnaire d'entrepôt et ses utilisateurs. En utilisant des expressions MDL, un utilisateur peut créer et faire évoluer le schéma multidimensionnel de L'entrepôt. Les principales caractéristiques de MDL sont les suivantes :

- MDL autorise la création de schémas multidimensionnels complexes. II autorise en particulier la création clé des schémas de dimension avec plusieurs niveaux et de schémas de cube avec une ou plusieurs mesures.
- MDL autorise l'évolution clé des schémas. II permet plus précisément l'ajout, la suppression et la modification de schémas de dimension et de schémas de cube.

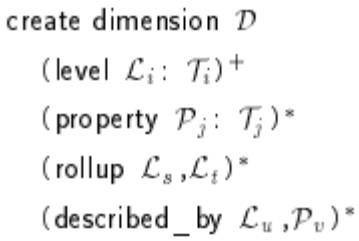


FIGURE 2.5 – Exemple en MDL [5]

Ci-dessus est un exemple d'une expression en ce langage, cette expression permet de créer un schéma de dimension.

2.2.6 Travail de Favre et al 2007 [6]

Dans ce travail, les auteurs ont proposé une approche d'évolution des entrepôts de données, qui permet à l'utilisateur d'introduire ses propres connaissances afin d'enrichir les possibilités

d'analyse de l'entrepôt. Les connaissances sont représentées par des règles «si alors» qui crée des axes d'analyse additionnels en générant de nouveaux niveaux de granularité dans les hiérarchies de dimension. Cette approche est fondée sur le modèle formel de l'entrepôt de données évolutif qui permet de gérer la mise à jour des hiérarchies de dimension. présentons un exemple de cas réel de la banque LCL ainsi que l'approche orientée utilisateur détaillée dans ce travail

a. Exemple :

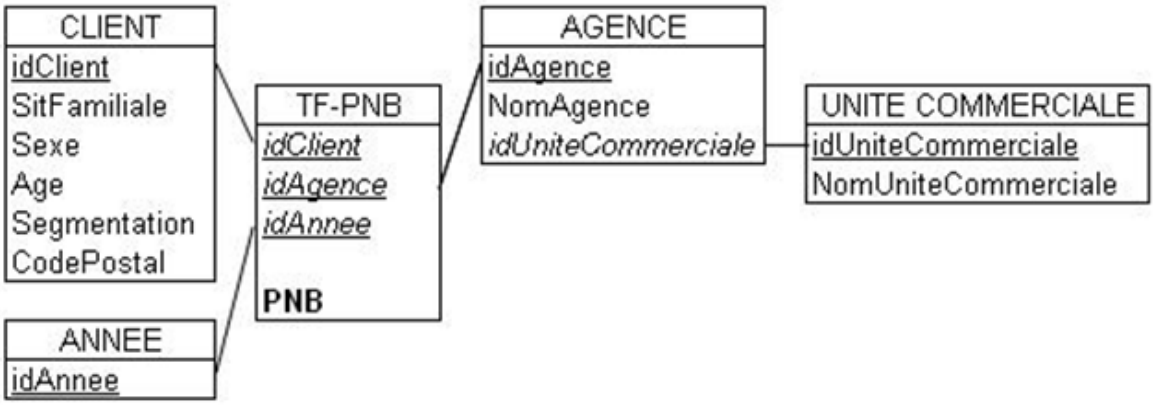


FIGURE 2.6 – Schéma multidimensionnel extrait du cas réel LCL [6]

Dans le but de décrire les évolutions possibles d'un modèle d'entrepôt de données, Les auteurs illustrent sur un exemple issu du cas réel de la banque LCL, les évolutions que peuvent subir un modèle et l'impact qu'elles ont en terme de cohérence des analyses.

Le PNB annuel (Produit Net Bancaire) correspond à ce que rapporte un client à l'établissement bancaire. Cette mesure est analysée selon les dimensions CLIENT, AGENCE et ANNEE Il est possible d'agréger les données selon le niveau UNITE_COMMERCIALE, qui est un regroupement d'agences

Supposons qu'un utilisateur veuille analyser les données selon le type d'agence ; il sait qu'il en existe trois : type « étudiant » pour les agences ne comportant que des étudiants, type «non résident» lorsque les clients ne résident pas en France, et le type «classique» pour les agences ne présentant pas de particularité. Ces informations n'étant pas présentes dans l'entrepôt, il est impossible pour lui d'obtenir une telle analyse. L'objectif est donc de proposer à L'utilisateur d'intégrer dans le schéma de l'entrepôt sa connaissance sur les types d'agence pour créer le niveau de granularité TYPE_AGENCE (Figure 2b).

b. L'approche orientée utilisateur

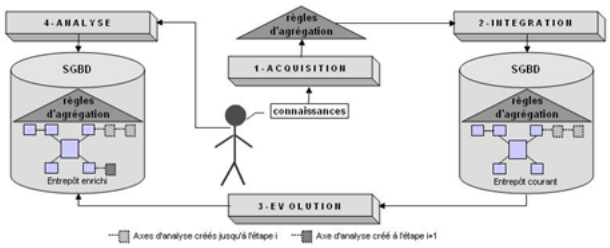


FIGURE 2.7 – L'architecture de l'approche orientée utilisateur [6]

Le schéma ci-dessus représente les étapes de l'approche orientée utilisateur qui se compose des phases suivantes :

1. une phase d'acquisition des connaissances utilisateurs sous la forme de règles d'agrégation.
2. une phase d'intégration de celles-ci pour les stocker dans le SGBD (Système de Gestion de Bases de Données).
3. une phase d'évolution du schéma dans laquelle a eu lieu la création de nouveaux niveaux de granularité au sein des hiérarchies de dimension à partir des règles d'agrégation.
4. une phase d'analyse sur le modèle qui évolue ainsi de façon incrémentale.

L'approche se base sur un modèle d'entrepôt évolutif à base de règles d'agrégation (R-DW), qui est composé d'une partie « fixe », correspondant au schéma de l'entrepôt qui répond à des besoins d'analyse globaux, et une partie « évolutive » définie par les règles d'agrégation créant de nouveaux niveaux de granularité dans les hiérarchies de dimension, répondant aux besoins d'analyse personnalisés.

2.2.7 Travail de Solodovnikova [7]

ce travail présente un framework d'entrepôt de données prenant en charge l'évolution de ce dernier. La structure est capable de gérer non seulement les modifications de sources de données, mais également les modifications directes dans un schéma d'entrepôt de données. Dans la structure, les versions de l'entrepôt de données sont prises en charge dans l'environnement de développement ainsi que dans les rapports de l'environnement utilisateur.

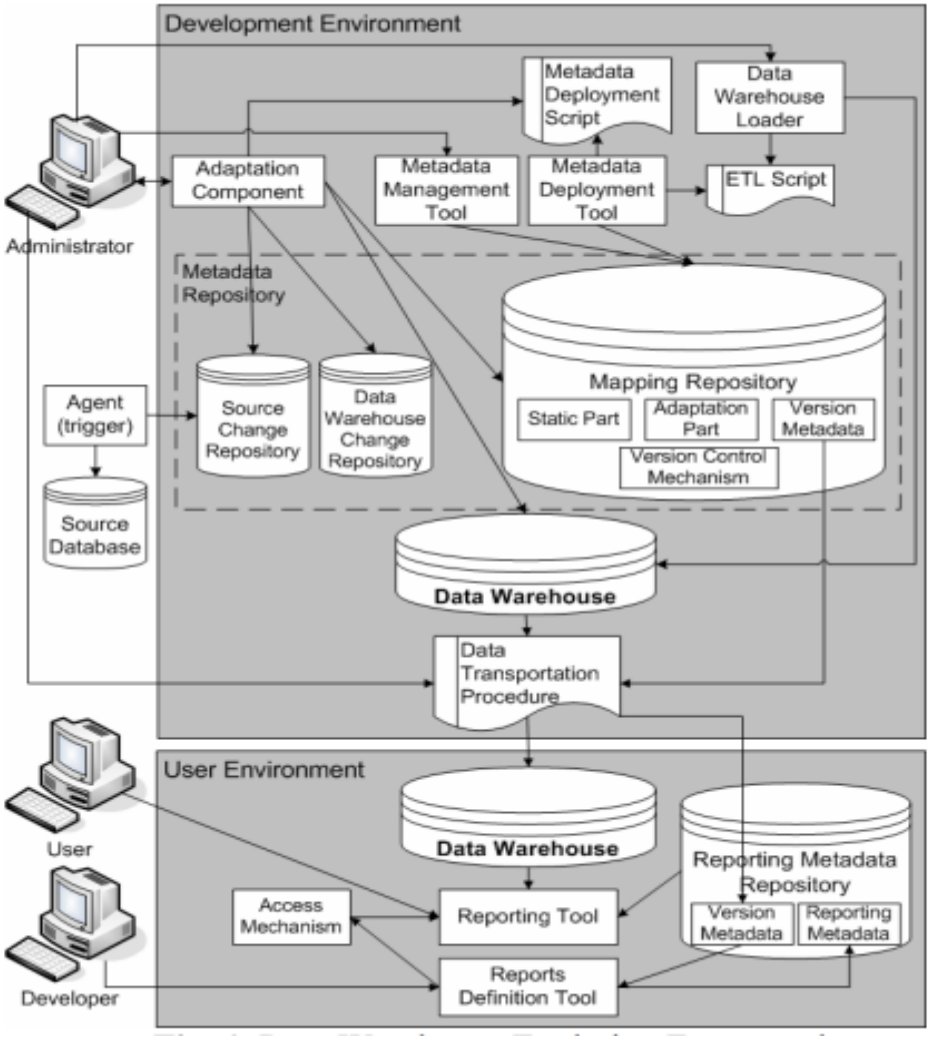


FIGURE 2.8 – Architecture du framework d'adaption des entrepots de données [7]

Ce framework a été développé en élargissant le framework d'adaptation de l'entrepôt de données. [41]

Contrairement au framework d'adaptation, le framework d'évolution est capable de gérer non seulement les modifications de sources de données, mais également les modifications directes d'un schéma d'entrepôt de données. La deuxième différence importante est le fait que, dans la structure d'évolution, les versions de l'entrepôt de données sont prises en charge dans l'environnement de développement ainsi que dans les rapports dans l'environnement utilisateur.

Le cadre proposé dans ce travail diffère des autres solutions de problèmes d'évolution des entrepôts de données présentés dans la littérature par le fait qu'il prend en charge de nombreux problèmes d'évolution à la fois, et non un seul problème. De plus, il est prévu de transformer le prototype développé du framework d'adaptation pour le rendre conforme au framework d'évolution susmentionné comme illustré dans la figure précédente.

2.2.8 Travail de Favre et al 2007[8]

Ce travail propose une solution à la personnalisation des analyses dans les entrepôts de données. Cette solution se base sur une évolution du schéma de l'entrepôt guidée par les utilisateurs. Il s'agit en effet de recueillir les connaissances de l'utilisateur et de les intégrer dans l'entrepôt de données afin de créer de nouveaux axes d'analyse. Quatre contributions majeures ont été proposées :

- la définition d'un modèle formel d'entrepôt de données évolutif, basé sur des règles «si-alors», que nous appelons règles d'agrégation. Ce modèle est composé d'une partie «fixe» et d'une partie «évolutive». La partie fixe est constituée de la table des faits et des tables de dimension qui lui sont directement reliées. La partie évolutive est composée d'un ensemble de hiérarchies de dimension qui sont mises à jour. Pour assurer la généricité de notre approche, nous proposons également un méta-modèle qui permet de décrire tout entrepôt de données évolutif.
- Le modèle évolutif est soutenu par une architecture qui comprend quatre modules
  - un module d'acquisition des connaissances utilisateurs sous forme de règles d'agrégation.
  - un module d'intégration des règles d'agrégation dans l'entrepôt de données.
  - un module d'évolution du schéma.
  - un module d'analyse permettant à l'utilisateur de réaliser des analyses sur le nouveau schéma.

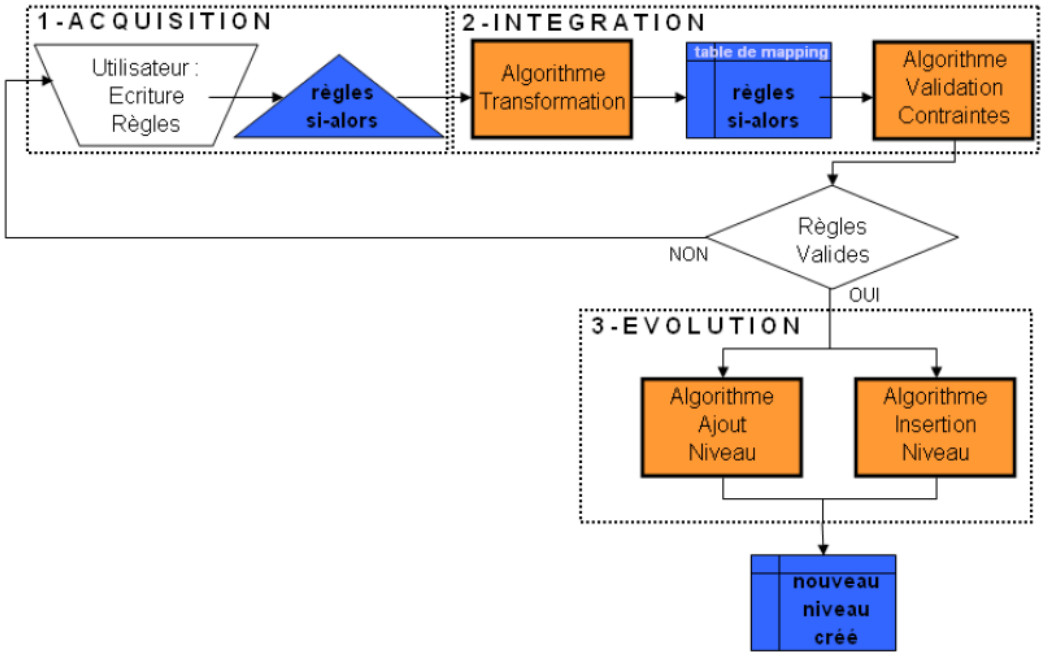


FIGURE 2.9 – Architecture du modele Evolutif de Favre [8]

- La proposition d'un modèle d'exécution comme décrit dans la figure suivante, avec l'approche rationnelle pour mettre en œuvre cette architecture globale. Il vise à gérer l'ensemble des processus liés à l'architecture globale. Il est fondé sur la transformation des règles d'agrégation en une table relationnelle de mapping qui permet le stockage, la vérification des règles et la création des niveaux de hiérarchie.

- l'évaluation de la performance du modèle d'entrepôt de données évolutif. Or, l'évaluation de la performance des modèles est généralement basée sur une charge (ensemble de requêtes utilisateurs). Lorsqu'un changement au niveau du schéma de l'entrepôt de données se produit, la charge doit être mise à jour. Dans ce contexte, ils proposent ici une méthode de mise à jour incrémentale de la charge.

Cette approche offre une flexibilité pour l'évolution du schéma de l'entrepôt puisque les hiérarchies de dimension sont mises à jour dynamiquement, une interaction accrue entre l'utilisateur et le système d'aide à la décision. L'utilisateur devient alors un réel acteur du processus décisionnel, dont le rôle va au-delà de la navigation dans les données selon le modèle initialement défini.

2.2.9 Travail de Favre et al 2009 [9]

Cette étude a comparé les méthodes de mise à jour du modèle par exemple [42], [43] et [5] qui ont utilisés les Opérateurs d'evolution, [44] et le travail de la même équipe [8] qui ont utilisés l'enrichissement des hiérarchies de dimensions et [3] et [45]qui ont été intéressés par la maintenance des vues.

Les auteurs ont comparé ces méthodes avec des méthodes de modélisation temporelle comme [46] qui ont utilisés les instances temporelles et [47] qui a proposé des Liens d'agrégation temporelles et enfin [3], [48] et [49]qui se sont beaucoup plus intéressés aux version temporelles

Dans tous les travaux qui suivent la modélisation temporelle, une extension à un langage SQL traditionnel est requise pour prendre en compte les particularités des approches d'analyse ou de chargement de données. Ces approches doivent être déployées lors de la phase de conception. Favre et al. indiquent que Certains systèmes commerciaux permettent déjà de suivre les modifications de données et de les interroger efficacement selon différents scénarios temporels. Par exemple, SAP-BW permet à l'utilisateur de choisir la version des hiérarchies pour une analyse [50]. Toutefois, la gestion des versions de schéma n'a été que partiellement explorée et aucun outil commercial dédié n'est disponible pour le concepteur. En outre, comme nous l'avons mentionné précédemment, le déploiement d'approches temporelles doit être envisagé au stade de la conception, et elles ne sont pas applicables aux entrepôts de données déjà déployés.

2.2.10 Travail de Zhou et al [10]

Ce travail explique L'utilisation du mécanisme des vues pour simuler les changements du schéma d'une base. Rappelons qu'une vue est définie par une abstraction sur le schéma réel de la base et que l'instance d'une vue est calculée à partir du schéma réel. L'idée est que les applications utilisent la base de données par l'intermédiaire des vues, une évolution du schéma est ainsi traduite par la création ou la modification d'une vue qui reflète la sémantique de cette évolution.

Les vues matérialisées Dans leurs formes la plus simple, sont des relations dérivées définies en termes de relations stockées dans la base. Une vue non matérialisée (vue virtuelle) est une spécification de programme (vue définition - intentions) qui génère des données à chaque appel. Une vue peut être matérialisée en stockant les résultats de la vue dans la base de données en tant qu'extension. En conséquence, outre son intention, une vue matérialisée contient également des données comme toute autre relation stockée. Ainsi, il est possible d'interroger davantage pour construire des vues sur vues ou de les regrouper collectivement pour créer des super-vues (superviews).

Il est important de noter que dans les environnements d'entreposage de données classiques, la mesure de la latence des données et la disponibilité des données d'entrepôt sont généralement assouplies. Cela signifie qu'il existe une période de temps pendant laquelle les données stockées dans l'entrepôt et les données provenant des sources d'informations sont incohérentes (c'est-à-dire, la réplication asynchrone) et que le DW reste indisponible pour les requêtes OLAP pendant le processus de maintenance de la vue (c'est-à-dire, refresh, où la sémantique transactionnelle n'est pas conservée). Dans les environnements d'entreposage de données distribués, il convient de spécifier à nouveau la mesure de la latence des données et la disponibilité des données réparties sur les sites. Ces critères doivent être utilisés par le module de niveau global afin de maintenir la cohérence inter site.

2.2.11 Travail de Gupta et al [11]

Dans ce travail, Les auteurs présentent une conception et une implémentation de bout en bout d'un système d'entreposage de données évolutif, géoréférencé, en temps quasi réel, appelé Mesa.

Mesa est un système d'entreposage de données analytiques hautement évolutif qui stocke des données de mesure critiques liées à l'activité de publicité Internet de Google. Mesa est conçu pour satisfaire un ensemble complexe et exigeant d'utilisateurs et de systèmes, comprenant l'ingestion et l'interrogation de données en temps quasi réel, ainsi que la haute disponibilité, la fiabilité, la tolérance aux pannes et l'évolutivité de données et requêtes volumineuses. Plus précisément, Mesa gère des pétaoctets de données, traite des millions de mises à jour de lignes par seconde et traite des milliards de requêtes qui extraient des milliards de lignes par jour. Mesa est géo-répliqué sur plusieurs centres de données et fournit des réponses cohérentes et répétables aux requêtes à faible latence, même en cas de défaillance d'un centre de données complet. Ce document présente le système Mesa et décrit les performances et l'échelle qu'il réalise.

L'architecture du système est composée de trois frameworks principaux

1. Le framework contrôleur / ouvrier(controller/worker framework), qui prend en charge les mise à jour et la maintenance (Figure 2.10)

2. framework de traitement de requêtes (figure 2.12)

3. Le framework de traitement de mise à jour dans un déploiement Mesa multi-centres de données(figure 2.11)

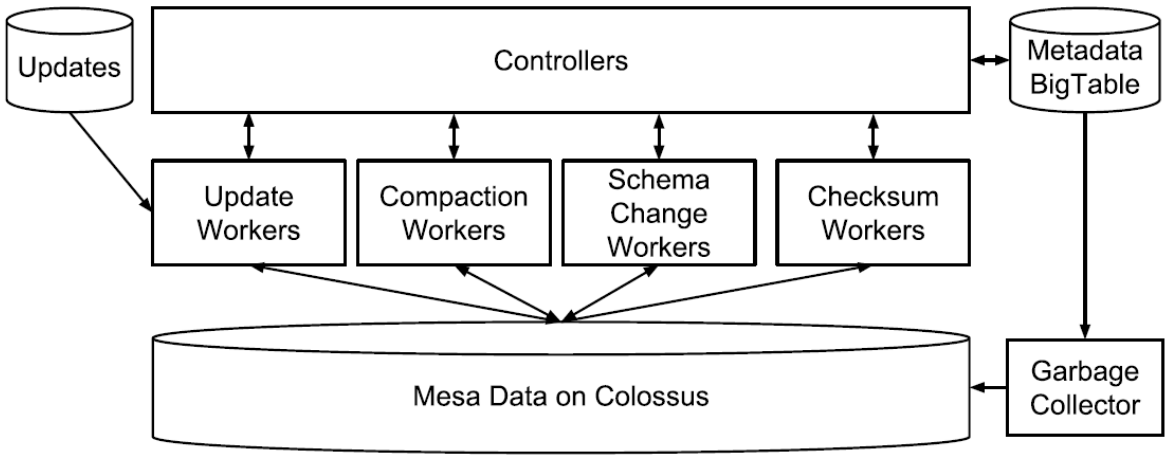


FIGURE 2.10 – Le framework de traitement de mise à jour dans un déploiement Mesa multi-centres de données[10]

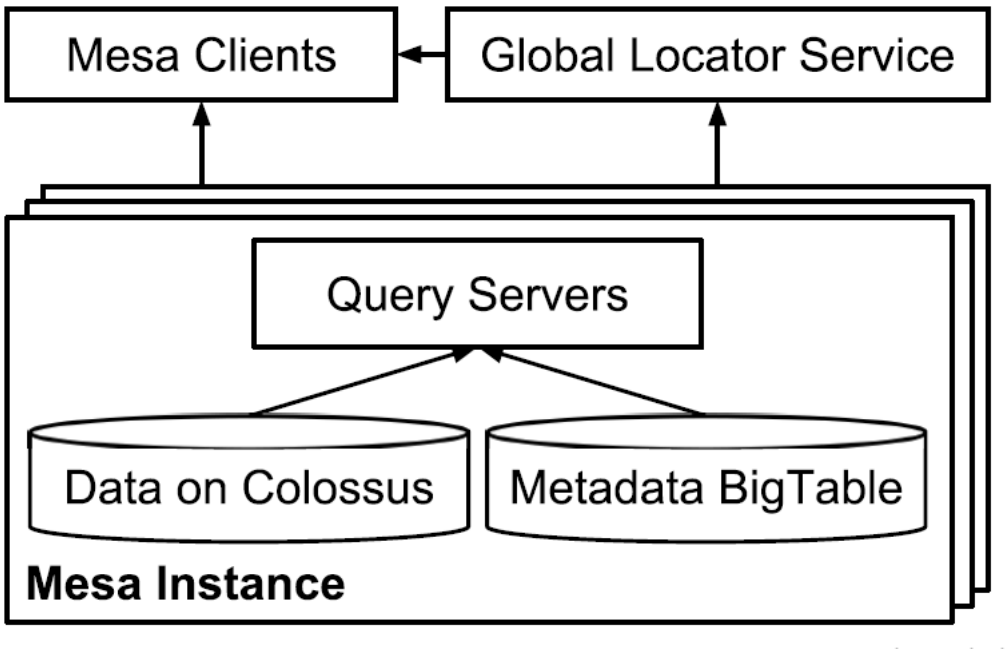


FIGURE 2.11 – Framework controleur[11]

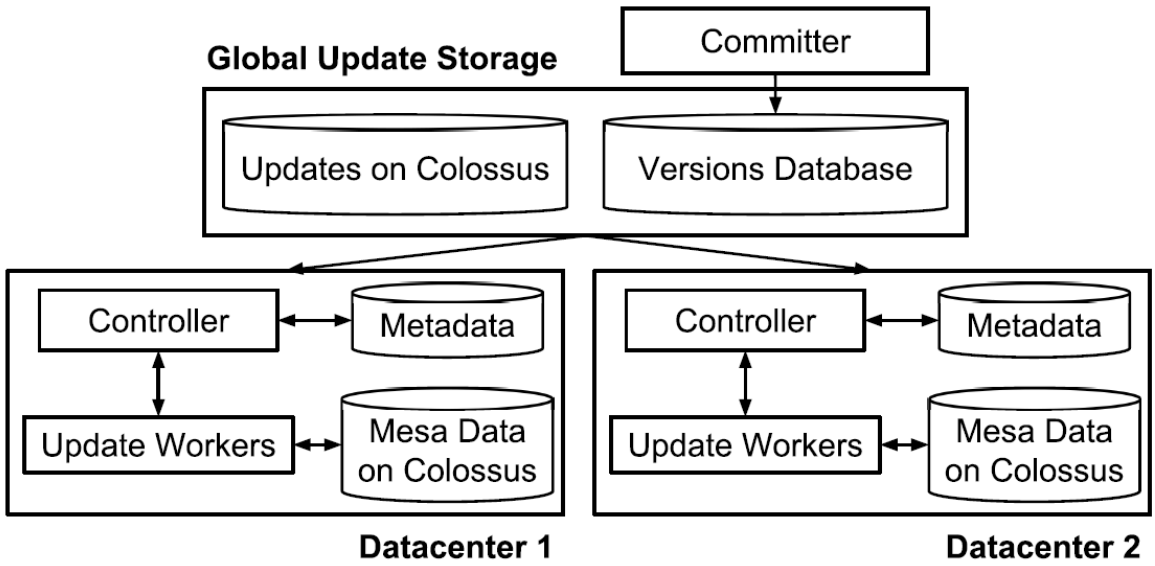


FIGURE 2.12 – Le framework de traitement de requetes[11]

La conception technique de Mesa s'appuie sur des idées de recherche fondamentales dans les domaines des bases de données et des systèmes distribués. En particulier, Mesa prend en charge les requêtes et les mises à jour en ligne tout en offrant des garanties de cohérence élevées et d'exactitude des transactions. Il atteint ces propriétés à l'aide d'une interface orientée traitement par lots, garantissant l'atomicité des mises à jour en introduisant un contrôle de version transitoire des données, éliminant ainsi le besoin de synchronisation basée sur le verrouillage des transactions de requête et de mise à jour. Mesa est géo-répliqué sur plusieurs centres de données pour une tolérance aux pannes accrue. Enfin, au sein de chaque centre de données, l'infrastructure contrôleur / travailleur de Mesa lui permet de répartir le travail et d'échelonner de manière dynamique les calculs requis sur un grand nombre de machines afin de fournir une évolutivité élevée. L'analyse en temps réel de vastes volumes de données générées en continu (de manière informelle, le «Big Data») s'est révélée être un défi important dans le contexte de la recherche et de la pratique de bases de données et de systèmes distribués. Une approche a consisté à utiliser des technologies matérielles spécialisées (par exemple, des machines massivement parallèles avec des interconnexions à grande vitesse et de grandes quantités de mémoire principale). Une autre approche consiste à exploiter les ressources du cloud avec un traitement parallèle par lots basé sur un paradigme de programmation similaire à MapReduce. Le premier facilite l'analyse de données en temps réel à un coût très élevé, tandis que le second sacrifie l'analyse de données récentes au profit d'un débit peu coûteux.

2.2.12 Travail de Golfarelli et Rizzi[12]

Ce travail passe en revue plus de 20 ans de recherche sur les systèmes d'entrepôts de données, depuis leurs premières implémentations relationnelles (encore largement adoptées dans les envi-

ronnements d'entreprise), jusqu'aux nouvelles architectures sollicitées par les scénarios BI 2.0 au cours de la dernière décennie et jusqu'aux défis passionnants. Maintenant posé par l'intégration avec les paramètres Big Data. La chronologie de la recherche est organisée en trois pistes interdépendantes : techniques, architectures et méthodologies.

Il montre dans la figure suivante la répartition par sujet et par piste des articles publiés dans les deux principaux forums spécifiques à la recherche sur le DW, à savoir la conférence de la DaWaK et l'atelier DOLAP, depuis 2013. Il apparaît clairement que les problèmes de traitement des requêtes et d'optimisation suscitent toujours l'intérêt de la communauté qu'une partie pertinente de la recherche est consacrée aux problèmes d'architecture.

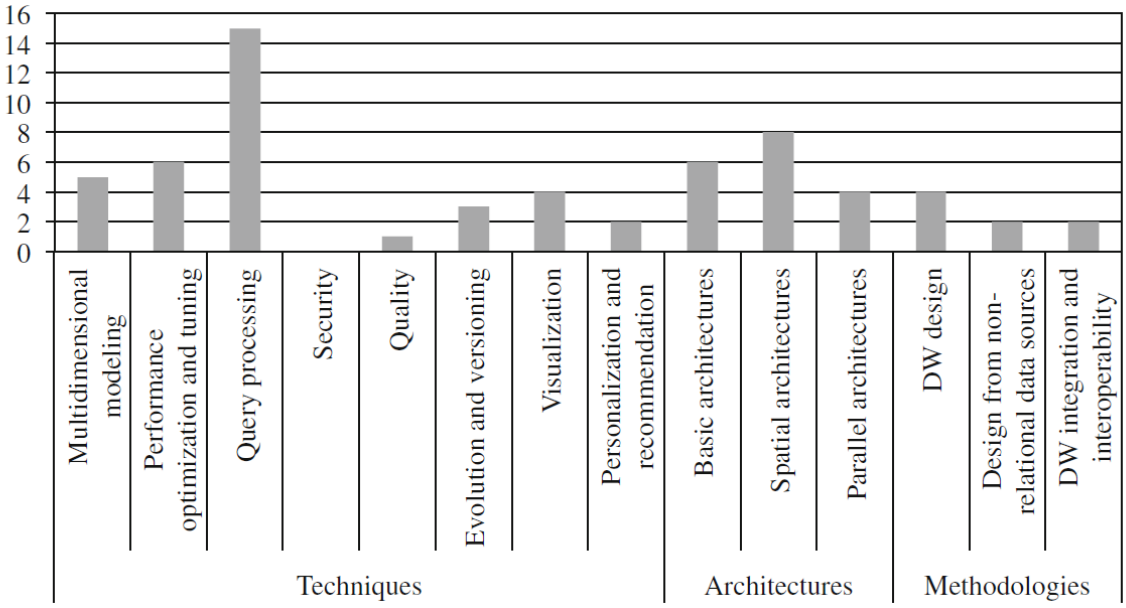


FIGURE 2.13 – Comparaison entre les différentes techniques, Architectures et Methodologies [12]

2.2.13 Travail de Larbi [13]

Dans ce travail, l'auteur répond aux problèmes d'évaluation et d'optimisation des requêtes en vue de spécifier les besoins des décideurs (requêtes classiques) surtout en cas où les besoins sont vagues ou comprennent des parties imprécises ou indéterminées (requêtes flexibles). Via la réécriture des requêtes, ils peuvent faire appel à la logique floue ou aux systèmes experts ou même aux ontologies pour pouvoir enlever l'ambiguïté.

L'auteur met l'accent sur l'évaluation précise des besoins des décideurs avant de choisir la base de production (Données sources) des systèmes décisionnels en général et dans la conception ascendante du SAD Système d'Aide à la Décision) en particulier, parce qu'en partant d'une mauvaise interprétation des besoins, toutes les conceptions du DW et les analyses OLAP(Online Analytical Processing) qui s'en suivent seront erronées.

Des solutions sont proposées à base d'ontologie, de logique floue pour évaluer l'imprécision des besoins décisionnels en vue d'améliorer la conception du datawarehouse et aussi pour affiner l'analyse OLAP entamée.

La solution ontologique est rigoureuse mais la proposition n'est qu'à son début, elle offre des mécanismes de raisonnement qui peuvent être exploitées pour l'identification des besoins (à base de requête).

Pour mettre en évidence le problème des besoins et pour proposer des solutions il a traité l'évaluation des risques, Parmis les cas :

- Gestion de projet intelligent et estimation du risque, explicitée dans le livre "Logique Floue" du [51]
- L'évaluation de risque dans le transport intelligent, qu'on l'a mis en application dans des solutions proposées (Solution structurelle, Solution ontologique et solution à base de systèmes experts), comme illustré dans les schéma suivant respectivement

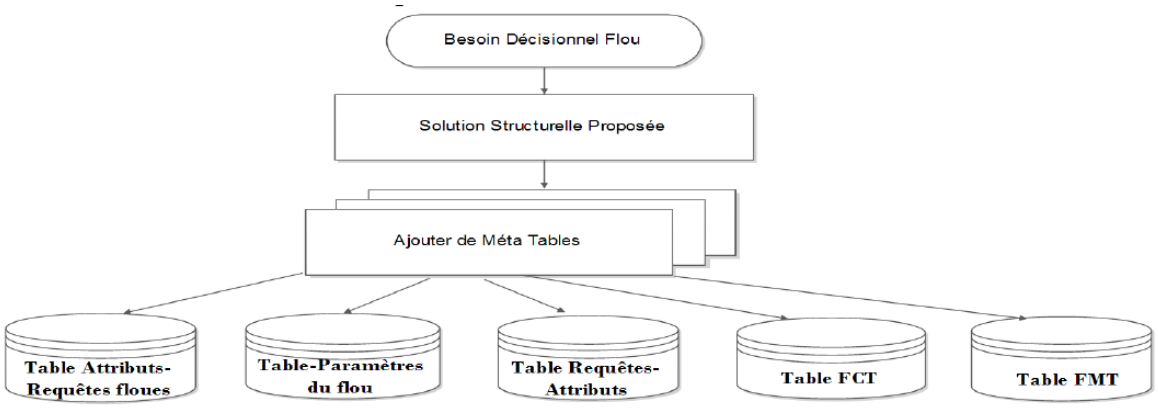


FIGURE 2.14 – Solution proposée par Larbi [13]

2.2.14 Travail de Matatallah 2018 [14]

Ce travail s'intéresse beaucoup plus à l'extensibilité physique, Ali élabore une étude comparative sur les performances de six solutions NoSQL(not only SQL) très répandues dans le marché, à savoir : MongoDB, CouchBase, Cassandra, HBase, Redis, OrientDB à l'aide du Benchmark YCSB(Yahoo ! Cloud Serving Benchmark.) Ce dernier étant un outil très connu pour sa puissance de test et utilisé dans plusieurs travaux d'évaluation des bases de données NoSQL à savoir YCSB. La finalité est d'apporter l'assistance et l'aide nécessaire aux acteurs intéressés de Big Data et de Cloud Computing pour d'éventuelles prises de décision sur le choix de la meilleure solution appropriée pour leurs entreprises.

Matallah dans son travail propose des remaniements internes dans l'architecture de l'implantation de référence des Clouds de stockage et de Big Data, à savoir HDFS(Hadoop Distributed File System) d'Hadoop. Ce dernier adopte un service de métadonnées séparé aux données avec

isolation et centralisation des métadonnées par rapport aux serveurs de stockage de données. c'est une approche qui permet d'améliorer le service de métadonnées d'HDFS, afin de maintenir la cohérence sans compromettre les performances des métadonnées en employant un parallélisme modéré des métadonnées, entre centralisation et distribution des métadonnées, dans le but d'accentuer la performance et l'extensibilité du modèle.

la structure proposée est schématisée dans la figure suivante :

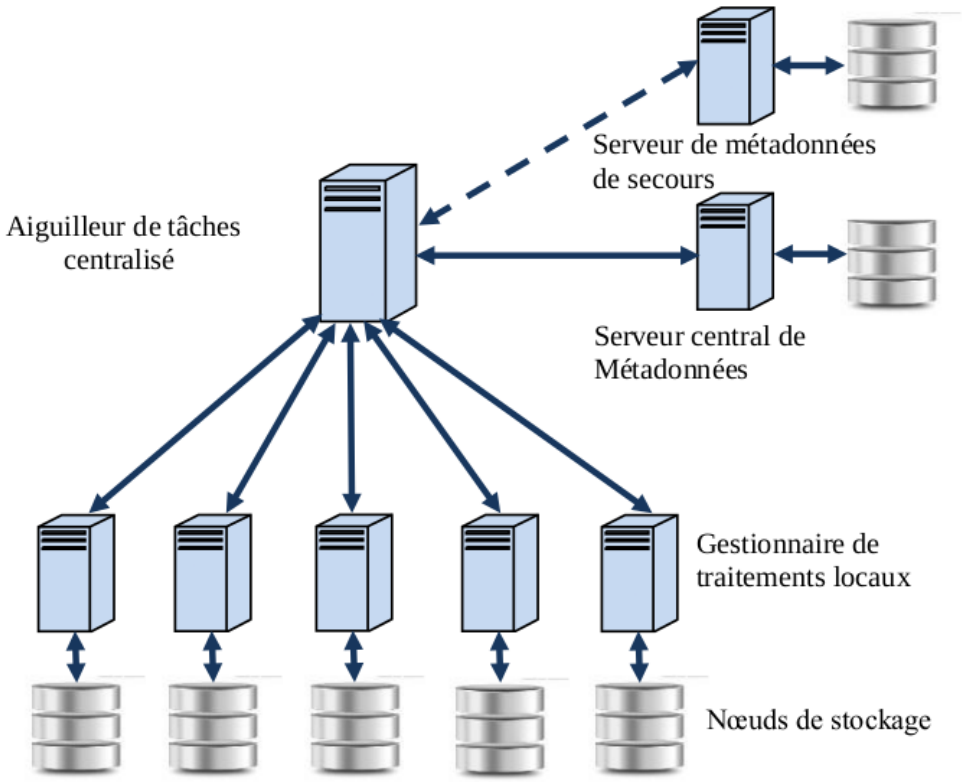


FIGURE 2.15 – Structure du modèle hybride de Matallah [14]

L'approche proposée, emploie un parallélisme modéré des métadonnées, en renforçant le parallélisme massif des données, qui est la clé de réussite des architectures modernes et délègue une partie de la responsabilité des métadonnées aux serveurs de stockage. Le parallélisme massif qui s'étend aux métadonnées peut affecter la fiabilité du service et la cohérence de ces informations. La centralisation du service des métadonnées dans une seule machine peut constituer un goulot d'étranglement qui peut dégrader les performances du système global dans le futur, vu l'accroissement massif très rapide des données qui engendre automatiquement un autre accroissement massif des métadonnées.

2.3 Synthèse et discussion

2.3.1 Approches utilisés par les traveaux existants

Cette partie présente un résumé des différentes approches sur l'extensibilité d'un entrepôt de données citées auparavant, et ceci à travers le tableau suivant, où les lignes représentent les travaux, et les colonnes sont les techniques et méthodes exploitées par chaque travail. X Indique les techniques(colonnes) exploites par chaque travail(lignes).

	Modèle Multidimmentionnel Classique	Opérateurs d'évolution	Mécanismes de Vues	Versions	Mise a jour des hiérarchies de dimentions	Théorie des transactions	Logique Floue	Les Ontologies	Réécriture des Requêtes
[1]	X								
[5]		X			X				
[3]		X	X						
[2]			X						
[11]			X	X		X			
[12]			X						
[13]							X	X	X
[8][6][9]					X				
[7]					X				
[4]		X							
[4]		X							

TABLE 2.1 – Comparaison entre les approches prises par de différents travaux

2.3.2 Critères de comparaison entre les approches

Dans ce qui suit pour évaluer les approches d'extensibilité nous utilisons les 3 catégories de critères de comparaison principales présentés dans [9] : fonctionnalités offertes, déploiement, et performance.

Les critères correspondant aux fonctionnalités offertes par les approches sont :

- historisation de dimension
- prise en compte les changements des schémas source
- approche axée sur l'utilisateur

Ces critères nous indiquent si une approche autorise des dimensions invariantes dans le temps, si l'analyse effectuée après l'évolution du schéma est cohérente et si l'approche est centrée sur l'utilisateur en ce qui concerne le processus de décision. Plus précisément, ce dernier point est lié à la personnalisation de l'entrepôt de données qui est relativement une nouvelle approche dans la communauté. Ces travaux concernent la visualisation de données basée sur les préférences et le profil de l'utilisateur. Par exemple, (Bellatreche, Giacometti, Marcel, Mouloudi, & Laurent, 2005) affinent les requêtes des utilisateurs en fonction de leurs préférences pour ne montrer que les données pertinentes.

Les critères relatifs au déploiement des approches sont les suivants :

- La nécessité de déployer la solution pendant la phase de conception
- la simplicité de déploiement

Le besoin de déployer la solution pendant la phase de conception détermine si l'approche doit être appliquée pendant la phase de conception de l'entrepôt de données. La simplicité du déploiement détermine si les outils existants doivent être adaptés.

Enfin, les critères de performance sont :

- le volume de données stockées
- le temps pour répondre aux requêtes d'analyse

Il existe pas mal d'objectifs d'entrepôts de données qui nécessitent de l'analyse En ligne, donc on souhaite que la solution soit rapide, aussi l'espace de stockage consommé est très important pour assurer l'Évolutivité de l'entrepôt après une longue période de temps qui fait qu'une quantité colossale de données est introduite au système.

Dans un monde trés volatil comme l'économie et l'investissement, quelques KPIs doivent être en temps réel rapidement lors du moment de la décision, et donc on prend en compte cette possibilité aussi comme critère d'evaluation.

Un des problèmes qui n'est pas traité dans ces travaux au même temps que l'évolution du Schéma est l'extensibilité du rafraîchissement de l'entrepôt ce qui correspond à un processus représenté par l'extensibilité la phase ETL qui consiste essentiellement à l'ajout de données provenant des sources, sans remettre en cause les données présentes dans l'entrepôt. Notre approche de découpage des ETL en des schémas de fonctions essaye de résoudre ce problème et donner un aspect évolutive au processus ETL au même temp que l'évolution du schéma

2.3.3 Discussion

La table 2.2 résume la comparaison entre les différentes approches par rapport aux deux grandes familles de critères de comparaison cités dans la section précédent chaque ✓indique que l'approche a une influence positive sur la critère.

Les entrepôts de données sont destinés à l'analyse en ligne ; par conséquent, la performance est un aspect crucial en termes de temps de réponse et de capacité de stockage.

Le courant basé sur les modèles multidimensionnels classiques [1] considère les dimensions comme partie statique et les faits comme partie dynamique, l'historisation est donc assurée par la dimension Temps.

Les autres dimensions sont Invariantes temporellement parlant. Cependant, dans la pratique, des changements peuvent se produire dans le schéma des dimensions ou sur le schéma global de l'entrepôt. En fait, le schéma a besoin d'évoluer à cause des changements dans la source de données ou des besoins d'analyse.

Les approches qui traitent de l'implication de l'utilisateur de différentes manières. Par exemple, [6] considèrent que l'évolution du modèle est déclenchée par l'utilisateur, [5] a créé un langage nommé MDL qui permet à l'utilisateur d'introduire des changements en gardant l'entrepôt dans un état cohérent, tandis que d'autres considèrent que l'évolution est déclenchée par l'administrateur de l'entrepôt de données. [13] a utilisé la logique floue pour gérer les nouveaux besoins des décideur, D'autre part, d'après C.Favre ces approches permettent de prendre en compte l'évolution des besoins d'analyse. mais ne permettent pas celle de l'évolution conjointe des sources de données et des besoins d'analyse. Les performances dans ces approches sont naturellement faibles, car l'humain est directement impliqué dans l'évolution de l'entrepôt, ces approches aussi ne prennent pas en compte l'évolution du schéma, ce qui est très nécessaire dans un domaine comme l'économie.

Le déploiement des approches basées sur la modélisation temporelle peut être difficile à cause du besoin de déployer ces dernières dans la phase de conception et le besoin des outils spéciaux pour le chargement des données et leur analyse.

Concernant les performances, les méthodes temporelles, ont besoin d'un montant colossal d'espace de stockage pour les méta-données, des labels temporelles et des versions, le temps de réponse est plus lent, et la réécriture des requêtes est nécessaire pour prendre en compte les différentes versions de l'entrepôt de données, ces critères de performances sont rarement discutées à part dans [46]

Un autre courant se base sur l'hypothèse qu'un entrepôt de données est un ensemble de vues matérialisés construites à partir des sources de données [3], [12] et [11] les auteurs se sont intéressés à la maintenance de vues pour propager l'évolution du modèle sur les cubes de données, représentés par des vues Dans [3], il s'agit de s'intéresser à la maintenance de vues matérialisées induite directement par une évolution des sources de données. Ainsi, il s'agit de ramener le problème de l'évolution des sources de données à celui de la maintenance des vues. Les mécanismes de vues augmentent les performances énormément dans les DWH, et peuvent être utilisés directement même avant leur matérialisation.

		Mise a jour de modèles			Modélisation temporelle		
		Opérateurs d'evolution	Enrichissement des hiérarchies	Mécanismes de vues	Instances	Liens d'agrégation	Liens d'agrégation
Caratéréstique	Historisation des Dimentions				✓	✓	✓
	Changements de schémas sources	✓		✓	✓	✓	✓
	Orienté Utilisateur		✓				✓
Mise en place	Mise en Oeuvre dés la conception	✓	✓	✓			
	Simplicité	✓	✓	✓			
Performances	Stockage relativement léger	✓	✓	✓			
	Temp de réponse rapide aux analyses	✓	✓	✓			
	Analyse en temp réel			✓			

TABLE 2.2 – Comparaison des approches par rapport aux critères choisis

2.4 Approche adoptée

Nous tenons à exprimer l'idée que, pour nous la problématique de l'évolution de modèle est bien différente de celle du rafraîchissement de l'entrepôt. Le rafraîchissement correspond à un processus représenté par l'extensibilité de la phase ETL et qui consiste essentiellement à l'ajout de données provenant des sources, sans remettre en cause les données présentes dans l'entrepôt, un aspect de la problématique qui n'a pas était adressé au même temps que l'évolution de l'architecture de l'entrepôt dans les travaux que nous avons étudié. Vu l'importance des performances et de la rapidité de l'entrepôt dans le domaine d'investissement on choisit dans notre travail une approche basée sur les vues matérialisées comme base pour l'extensibilité de l'entrepôt. Pour attaquer la problématique de l'extensibilité dans la phase ETL, Nous proposons une approche basée sur des schémas de dépendances ETL qui sont découpés en fonctions de traitement de données.

2.5 Conclusion

Dans ce chapitre nous fournissons une vision globale sur la problématique de l'extensibilité dans les entrepôts de données, les différentes perspectives des travaux et les solutions qu'ils ont été proposés au fil des années. Ces solutions ont fait face à la fois à l'évolution du schéma de l'entrepôt, l'évolution des besoins d'analyse de l'utilisateur, et même l'évolution physique et Big Data, Nous avons mené une étude comparative des travaux touchant à ce sujet, les travaux que

nous avons étudié apportent des solutions à certains aspects de cette problématique, par exemple, la maintenance des vues matérialisées permet d'assurer une certaine propagation de l'évolution des sources de données. Nous avons vu la fréquence d'utilisation de différentes techniques dans les travaux pour découvrir la tendance dans le domaine, et par la suite, nous avons pris ces différentes techniques ou approches, et on les a comparé selon les critères les plus importantes à l'utilisateur. Cette étude nous a mené a choisir une approche basée sur les vues matérialisés pour l'evolution du schéma. Et de proposer une approche basée sur des schémas de dépendances ETL qui sont découpés en fonctions de traitement de données Dans ce qui suit on présente notre approche dans une conception d'un système décisionnel du domaine de l'hydraulique, spécialisé dans la stratégie des forages d'eau dans la wilaya d'Ourgla, ainsi que l'implémentation de ce système dans une plate-forme BI.

Chapitre 3

CONCEPTION

3.1 Présentation du cycle de vie Kimball DW/BI

L'approche du cycle de vie de Kimball existe depuis des décennies. Les concepts ont été initialement conçus dans les années 1980 [52] par des membres du groupe Kimball et plusieurs collègues de Metaphor Computer Systems. Lorsqu'ils ont publié la méthodologie pour la première fois dans The Data Warehouse Lifecycle Toolkit en 1998 [53], on l'appelait alors le cycle de vie des dimensions de l'entreprise, car ce nom renforçait trois concepts fondamentaux

- Concentrez-vous sur l'ajout de valeur commerciale dans l'ensemble de l'entreprise.
- Structure dimensionnelle des données fournies.
- Développez de manière itérative la solution par incréments de cycle de vie gérables plutôt que d'essayer un produit livrable directement.

En revenant aux années 1990, la méthodologie était l'une des rares à mettre l'accent sur cet ensemble de principes fondamentaux. Le nom de Business Dimensional Lifecycle (Cycle de vie d'affaires dimensionnel) différenciait donc cette approche de celle du secteur. En 2008, lorsque ils ont publié la deuxième édition de la trousse à outils Lifecycle Data Warehouse, nous avions toujours une confiance absolue en ces concepts, mais le secteur avait évolué. Les principes étaient devenus les meilleures pratiques classiques vantées par beaucoup, nous avons donc condensé le nom officiel de la méthodologie pour simplement décrire le cycle de vie de Kimball.

En dépit des progrès spectaculaires réalisés dans les domaines de la technologie et de la compréhension au cours des deux dernières décennies, les constructions de base du cycle de vie de Kimball sont restées remarquablement constantes. Cette approche en matière de conception, de développement et de déploiement de solutions DW / BI est éprouvée. Il a été utilisé par des milliers d'équipes de projet dans pratiquement tous les secteurs, domaines d'application, fonctions commerciales et plates-formes technologiques. L'approche du cycle de vie de Kimball s'est avérée efficace encore et encore.

La figure 3.1 illustre l'approche du cycle de vie de Kimball[16] . La réussite des implémentations de DW / BI dépend de la fusion appropriée de nombreuses tâches et composants. il ne suffit pas d'avoir un modèle de données parfait ou une technologie de pointe. Le diagramme de cycle de vie est la feuille de route générale décrivant la séquence de tâches requises pour une conception, un développement et un déploiement efficaces.

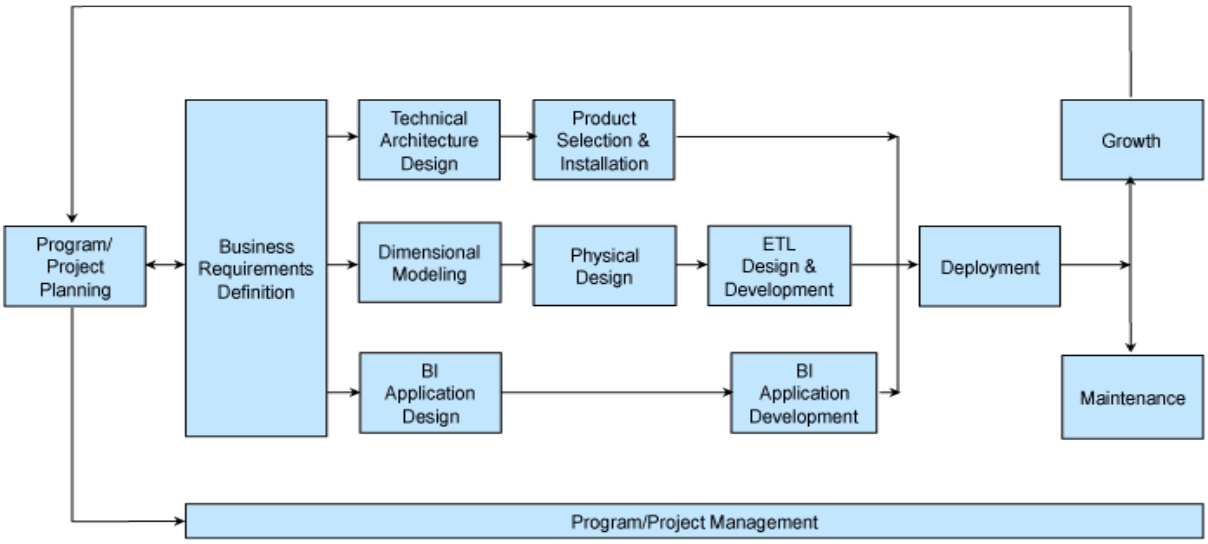


FIGURE 3.1 – Cycle de vie d'un projet BI/DW présenté par Kimball [16]

Suivre ce processus encourage la satisfaction des besoins du décideur à travers la bonne dérivation des besoins techniques, des besoins d'affaires et la création des documents d'architecture.

Pour faire cela il faut Illustrer les différents rôles au sein du système, et avoir une interaction constante avec les décideurs pour l'orientation et la validation, et pouvoir avoir une bonne idée de la direction dans laquelle le système va évoluer.

Dans la figure 3.1, les étapes du même niveau (vertical) peuvent progresser en parallèle.

3.1.1 Planification et gestion de programme / projet

Le premier encadré de la feuille de route est axé sur le lancement du programme / projet, y compris la portée, la justification et la dotation en personnel. Tout au long du cycle de vie, les tâches de gestion de programme et de projet en cours permettent de garder les activités sur la bonne voie.

3.1.2 Besoins du décideur

Déterminer les besoins décideur est une tâche clé dans le cycle de vie de Kimball, car ces résultats déterminent la plupart des décisions en amont et en aval. Les exigences sont collectées pour déterminer les facteurs clés ayant un impact sur l'entreprise en se concentrant sur ce que les utilisateurs professionnels font aujourd'hui (ou souhaitent faire à l'avenir), plutôt que de demander «que voulez-vous dans l'entrepôt de données ?». Les principales opportunités sont identifiées dans l'entreprise. , hiérarchisé en fonction de la valeur commerciale et de la faisabilité, puis des exigences détaillées sont rassemblées pour la première itération du développement du système DW / BI. Trois pistes de cycle de vie simultanées suivent la définition des exigences commerciales.

3.1.3 Piste de la technologie

Les environnements DW / BI exigent l'intégration de nombreuses technologies, magasins de données et métadonnées associées, Logiciels, Langages de programmation ... La piste technologique commence par la conception de l'architecture du système afin d'établir une liste complète des fonctionnalités nécessaires, puis par la sélection et l'installation de produits répondant à ces besoins.

3.1.4 Piste de données

La piste de données commence par la conception d'un modèle dimensionnel cible pour répondre aux besoins de l'entreprise, tout en prenant en compte les réalités de données sous-jacentes. Dans la modélisation dimensionnelle les données sont divisées en faits de mesure ou en dimensions descriptives. Les modèles dimensionnels peuvent être instanciés dans des bases de données relationnelles, appelées schémas en étoile, schémas en flocon de neige, schéma en constéllation d'étoiles ou des bases de données multidimensionnelles, appelées cubes OLAP selon le type de l'architecture. Quelle que soit la plate-forme, les modèles dimensionnels tentent d'atteindre deux objectifs simultanés : la facilité d'utilisation du point de vue des utilisateurs et la rapidité des performances des requêtes. La matrice de bus Enterprise Data Warehouse est un produit clé clé du cycle de vie de Kimball, qui représente les processus métier de base de l'entreprise et ses dimensions conformes communes. c'est un modèle de données qui garantit une intégration descendante de l'entreprise avec une livraison ascendante gérable en se concentrant sur un seul processus métier à la fois.

Le modèle dimensionnel est converti en une conception physique dans laquelle les stratégies de réglage des performances sont considérées, puis les défis de conception et de développement du système ETL sont résolus. Le cycle de vie décrit des sous-systèmes dans les flux d'extraction, de transformation et de chargement regroupés en quatre opérations principales : extraction des données de la source, nettoyage et conformité des transformations, fourniture des données à la couche de présentation et gestion des processus ETL d'arrière-plan.

3.1.5 Piste BI

Certains membres du projet sont immergés dans la technologie et les données, tandis que d'autres se concentrent sur l'identification et la construction d'un large éventail d'applications de BI, notamment les rapports standardisés, les requêtes paramétrées, les tableaux de bord, les tableaux de bord, les modèles analytiques, les applications d'exploration de données et les interfaces de navigation associées.

3.1.6 Déploiement, maintenance et croissance :

Les trois pistes du cycle de vie convergent lors du déploiement, réunissant la technologie, les données et les applications de BI. L'itération déployée entre dans une phase de maintenance, tandis que la croissance est prise en compte par la flèche vers la planification du projet pour la prochaine itération du système DW / BI. N'oubliez pas qu'un système DW / BI est un processus à long terme, pas un projet ponctuel.

Tout au long du cycle de vie de Kimball, un thème récurrent reconnaît que les concepteurs de DW / BI doivent constamment respecter les exigences du décideurs et les réalités sous-jacentes des données source, de la technologie et des ressources associées. Les équipes de projet qui se concentrent exclusivement sur les exigences (ou les réalités) isolément seront inévitablement confrontées à des risques importants en termes de livraison et / ou d'adoption commerciale.

3.1.7 L'application a notre projet

Dans ce chapitre, on détaille notre application du processus de cette conception étape par étape selon le processus précédent, quelques étapes concernent plutôt l'implémentation et sont détaillés dans le chapitre suivant.

3.2 Conception du Système BI/DW

3.2.1 Planification de projet/programme

Dans cette étape, nous avons établi les différentes échéances et les besoins pour pouvoir exécuter le projet, les différentes procédures de communication, et les politiques qu'on va suivre par la suite.

Des détails importants sont discutés comme les dates des réunions et leurs fréquence avec l'expert du domaine et le chef de projet. Ainsi que le canal de communication principal qui était l'email. La méthode de synchronisation du projet (google drive) et les différent outils utilisés pour le bon déroulement du projet.

3.2.2 Définition des besoins d'affaires

Dans ce qui suit, les besoins ont étaient dérivés après plusieurs séances de travail avec l'expert du domaine hydraulique,

Lors de la définition des besoins du décideur, il y a plusieurs facteurs à considérer pour pouvoir tailler l'entrepôt selon l'utilisateur, ces facteurs décident par la suite, l'architecture de l'entrepôt, les technologies utilisés, et de différents détails d'implémentation, le tableau 3.1 e les différents facteurs que nous avons considéré avant de définir les besoins :

a. Besoins du décideur

Pour prendre des décisions, le décideur doit être à tout moment, capable de consulter un tableau de bord, contenant une visualisation simple, compréhensible et à jour des Indicateurs de performances clés suivants :

Sources de données	Fichiers Excel de différentes entités
Quantité de données	En GB,
Latence des données	Mise a jour hebdomadaire, mais aussi sur demande
Nombre d'utilisateurs	1-50
Nature de taches des utilisateurs finaux	Consultation de KPIs, Cartographie, Système decisionnel, Evolution de l'entrepot
Objectifs de l'entrepot (Opérationnel/Stratégique)	Stratégique
Contraintes sur les données	Doivent respecter un modele précis

TABLE 3.1 – Caractéristiques de notre système BI

- Les couts d'installation forages :
  - Par Nappe
  - Par type (Potable/AEP/Mixte)
  - Par année
- Coût d'équipement de forage
  - Par Débit
  - Par Année
- Nombre de forages
  - Par année
  - Par zone
  - Par nappe exploitée
- Les besoins d'eau :
  - Par région
  - Par type d'utilisation (irrigation, eau potable)
  - Par année
- Débit moyen des forages
  - par Nappe exploitée
  - par Type d'utilisation

  - Par commune

- Durée de vie d'un forage
  - Par type
  - Par année
  - Par marque

Le décideur souhaite aussi pouvoir répondre aux questions d'affaires suivantes :

- Est-il possible d'implanter un nouveau forage dans une zone donnée ?
- Est il possible d'Implanter un forage dans les coordonnées (X,Y) ?
- Est-ce que le forage peut être exploité pour l'irrigation ou pour Adduction d'eau potable(AEP) ?
- Combien coûtera un forage approximativement de type T dans les coordonnées (X,Y) ?

3.2.3 Conception de l'architecture technique

Comme tout système BI, il est nécessaire que le système traite de différents aspects, la collecte de données, leur intégration,Extraction transformation et chargement (ETL) Stockage (entreposage) Analyse et visualisation, ces aspects doivent être traités de manière extensible et évolutive. Le système doit être capable d'évoluer avec le temps, et évoluer avec l'évolution des sources de données.

La figure 3.2 illustre comment fonctionne notre système

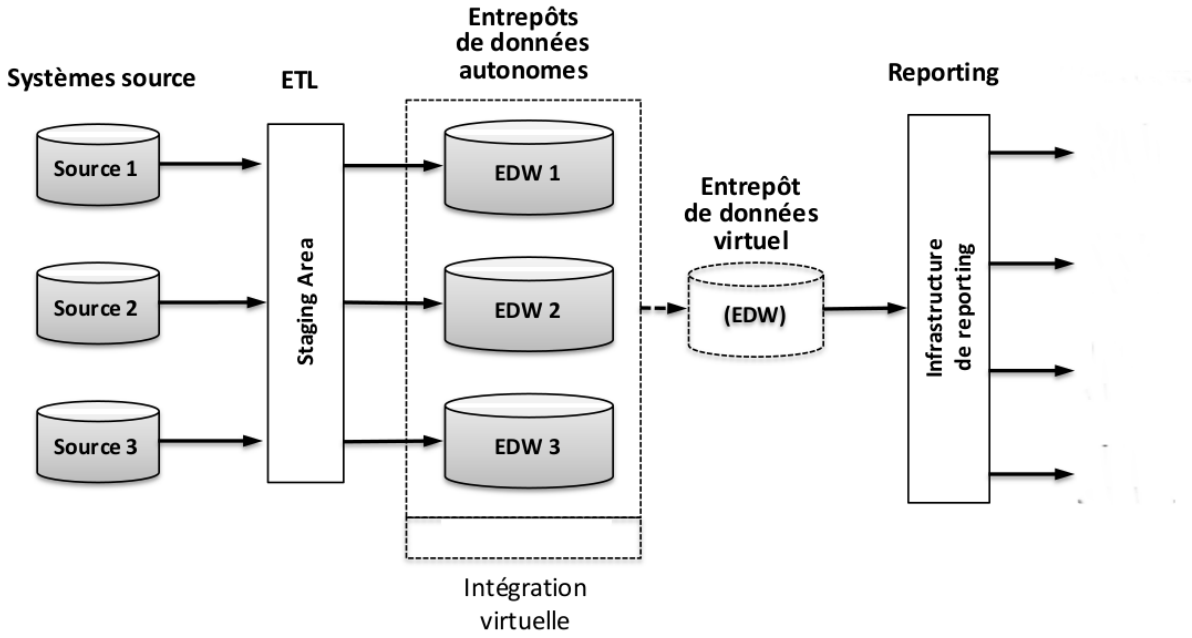


FIGURE 3.2 – Architecture du système

a. Sources de données

Les données de notre domaine d'application, parviennent des fichiers excel de différentes autorités dans le domaine (DRE Direction des Ressources d'eau), des factures, des données géographiques exportés des API web, Des données cartographiques..

3.2.4 Modélisation des données

Nous remarquons que les dimensions d'analyse de chaque fait sont différentes(revoir 2.2 Définition des besoins d'affaires pour plus de détails), pour accélérer l'analyse, minimiser les jointures, et faciliter l'accès aux différentes agrégations de données.Nous avons choisis l'architecture en constellation d'étoiles pour notre entrepôt initial.

L'extensibilité de ce Modèle de données est supportée par les vues matérialisés(voir la prochaine section)

Aprés plusieurs itérations de validation avec le décideur, le schéma dans la figure 3.3 représente le diagramme de notre modèle de données

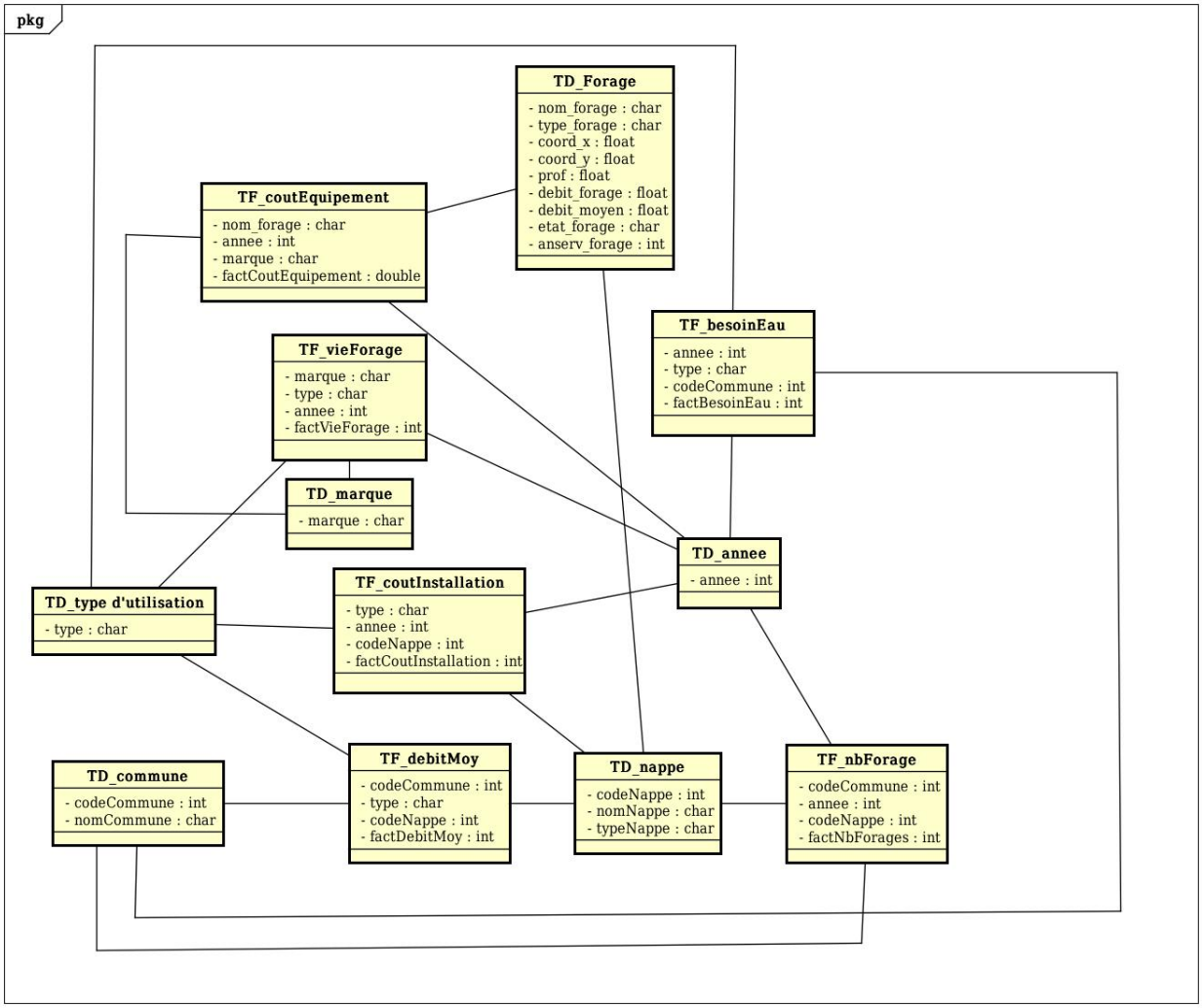


FIGURE 3.3 – Diagramme de classes

Le préfixe TF dans le nom d'une table, indique que cette table est une table de faits,

Le préfixe TD dans le nom d'une table, indique que cette table est une table de dimension,

Le préfixe fact dans le nom d'un attribut indique que cet attribut est un fait.

Chaque table a un maximum 3 dimensions pour ne pas trop complexer les agrégations de données et garder l'entrepôt à hautes performances.

3.2.5 Conception de l' ETL

Chaque entrepôt dans le système à un système ETL défini comme suit :

a. Intégration et fusion des données

Les données peuvent être introduit à la plate-forme, même directement par l'utilisateur à volonté. Via son interface, il peut facilement télécharger des fichiers Excel et CSV à la plate-forme et ils seront traités.

Ces données par la suite sont fusionnées et intégrés dans une grande zone de données par domaine, par exemple toutes les données concernant les forages sont ensemble, toutes données d'irrigation, avant d'initier le processus ETL.

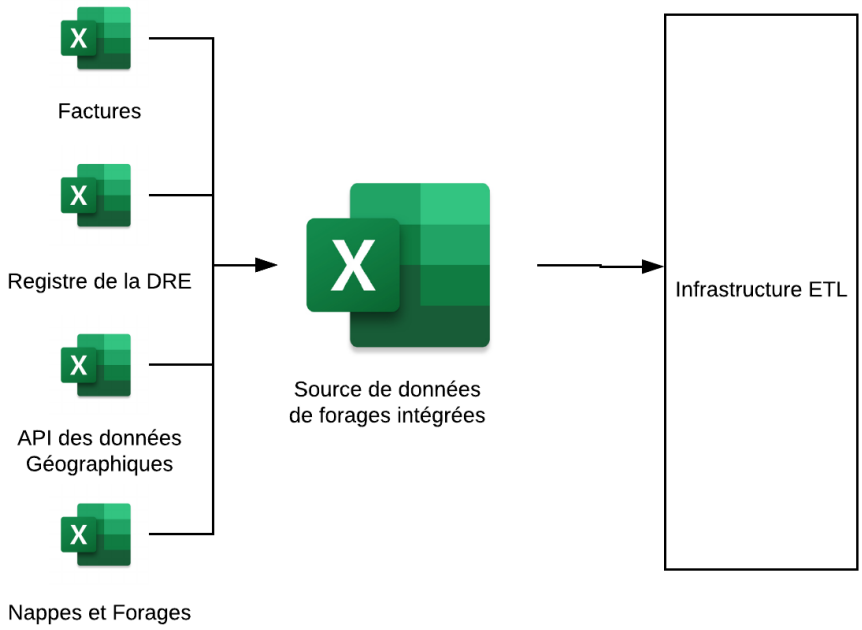


FIGURE 3.4 – Exemple de diagramme de dépendances ETL

b. Infrastructure ETL extensible

Pour chaque entrepôt dans le système, les données sont extraites via un système ETL(Extract, transform and Load) vers la zone de staging des données (DSA data staging area) ; une zone de stockage intermédiaire utilisée pour le traitement des données lors du processus d'extraction, de transformation et de chargement (ETL). Cette zone est utilisée pour la consolidation, l'alignement, la minimization ,la précalculation, et le nettoyage des données.

Les Diagrammes de Dépendances ETL

Le diagramme de dépendances ETL est utilisé pour représenter l'ordre d'extraction et de transformation de données sans causer des conflits ou des erreurs, c'est un graphe orienté, chaque

fleche sort de la fonction qui est nécessaire pour assurer le bon déroulement de la fonction successeur. Les fonctions sont conçus de tel façon que chaque fonction successeur fait appel aux fonctions prédécesseurs dans le diagramme de dépendances ETL. Cette façon de conception rend le débogage et la maintenance trés faciles, et meme l'extensibilité est assurée via la modification spécéfique des fonctions ETL. Pour donner un exemple, Le sous-diagramme dans la figure 3.4 illustre un des processus d'extraction de données dans notre système :

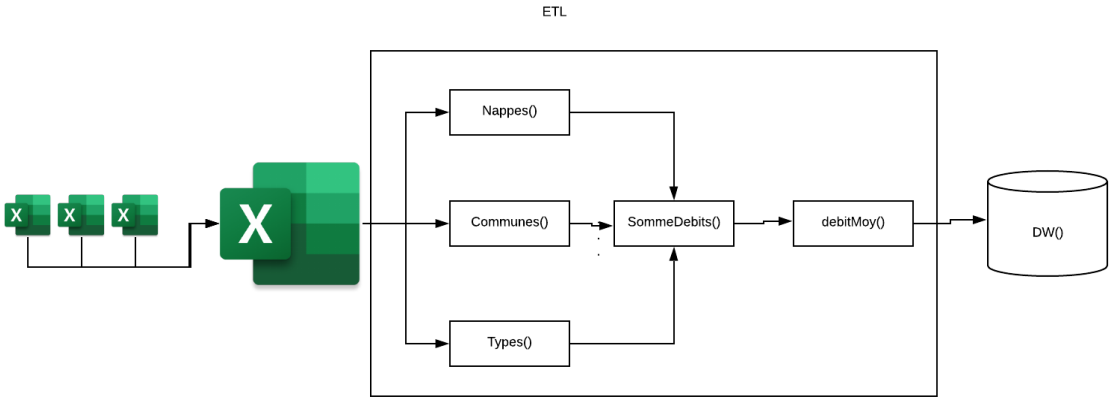


FIGURE 3.5 – Exemple de diagramme de dépendances ETL

La table de TF_debitMoy (table de fait de débit moyen) calcule le débit moyen par type d'utilisation, par Commune et par Nappe

D'abord, il faut extraire des tables de toutes les nappes, communes et types, puis calculer les sommes pour de différentes combinaisons de ces trois, pour enfin les diviser par le nombre de combinaisons pour avoir la moyenne.

Quelques données sont extraites directement et non pas calculés, c'est souvent le cas pour les dimensions

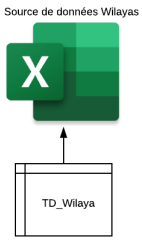


FIGURE 3.6 – Exemple de diagramme de dépendances ETL : extraction directe

c. Gestion de l'extensibilité au niveau de l'ETL

Pour chaque nouveau domaine ajouté, l'administrateur peut configurer un nouveau processus ETL via la conception de nouveaux schémas ETL similaires à ceux d'avant.

Cette approche des fonctions ETL indépendantes permet de facilement Intégrer de nouveaux entrepôts de données la façon décrite précédemment, elle permet aussi l'extensibilité Inter-Entrepôts de données, c'est à dire on peut enrichir les données d'un entrepôt à partir d'un autre. En prenant le deuxième entrepôt comme une source de données pour la fonction ETL.

Notre approche aussi permet aussi de facilement déboguer les problèmes, en les réglant directement au niveau de la fonction qui ne fonctionne pas correctement.

On peut très facilement retirer une source, modifier les calculs

d. Exemple : Extensibilité Inter-Entrepôts dans la phase d'ETL

Dans l'entrepôt Initial, Nous avons une table de faits TF_BesoinEau, cette table permet d'analyser et d'estimer les besoins d'eau par région, par année, et par type d'utilisation. Dans le but de prévoir l'insuffisance de l'eau extraite dans la région, et repérer les opportunités d'investir dans l'installation de nouveaux forages.

Il existe deux types d'utilisation : Irrigation/Adduction d'eau potable

Initialement, pour calculer le besoin d'eau potable par région notre entrepôt suppose l'utilisation moyenne d'un citoyen estimée par l'expert du domaine à 150L par citoyen. Et donc pour calculer le besoin d'eau d'une commune par jour, on multiplie le nombre de citoyens de cette commune par 150L/j.

Cela est illustré dans le schéma ETL suivant :

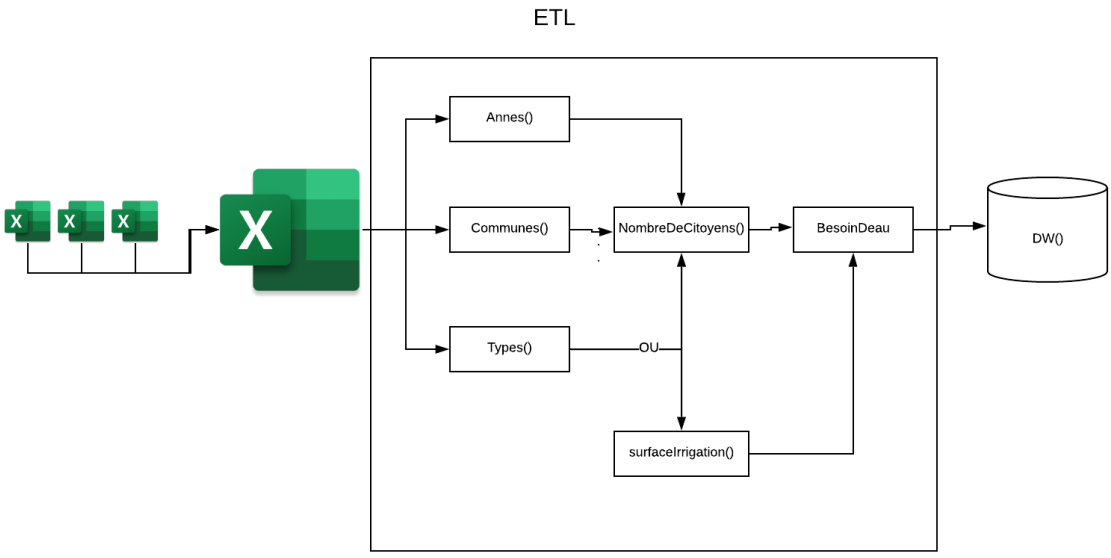


FIGURE 3.7 – Schéma ETL de calcul des besoins d'eau

L'expert du domaine remarque que la consommation d'un citoyen ordinaire est très différente de celle d'un citoyen Nomade, une catégorie de citoyens qui consiste en une partie importante de la population de Ourgla. Donc on décide d'étendre l'entrepôt des forages pour récupérer des données de la Direction de la planification et de l'urbanisme de Ourgla

concernant le type des citoyens de la région.

Comme ça on peut avoir des résultats d'estimation de besoin d'eau plus fiables et plus proches de la réalité. Et on peut calculer le besoin d'eau des citoyens nomade différemment des citoyens ordinaires, sachant que la consommation d'un citoyen Nomade est estimée a 50L/j.

On obtient donc le nouveau Schéma ETL suivant :

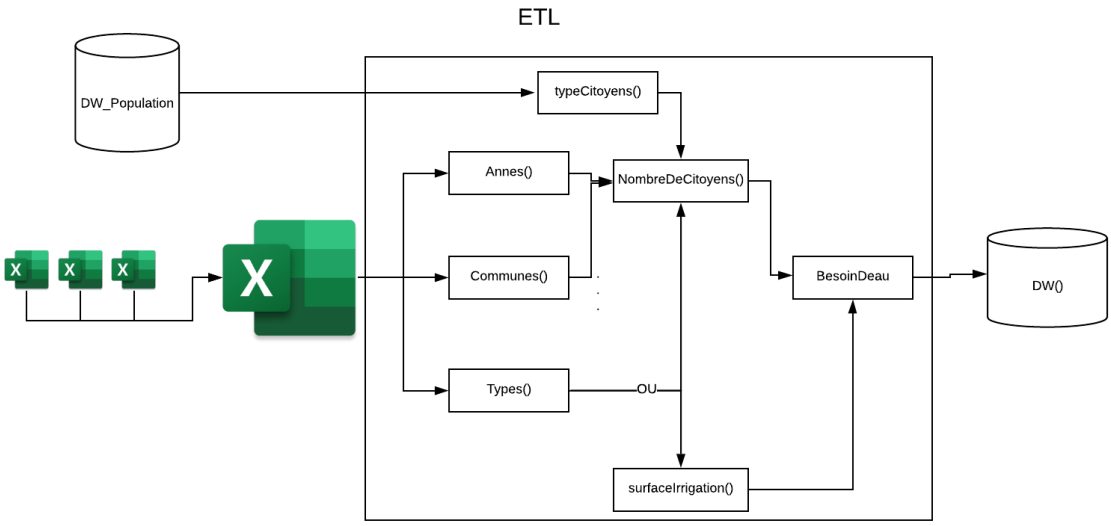


FIGURE 3.8 – Schéma ETL des besoins d'eau étendu

3.2.6 Cas d'utilisation

on peut résumer les cas d'utilisation du système dans le diagramme UML de la figure 3.8

Dans le but d'expliquer les différents cas d'utilisation on va regrouper les cas d'utilisation en 3 groupes :

1. Gérer l'entrepot
2. Aide a la décision
3. Gestion des roles

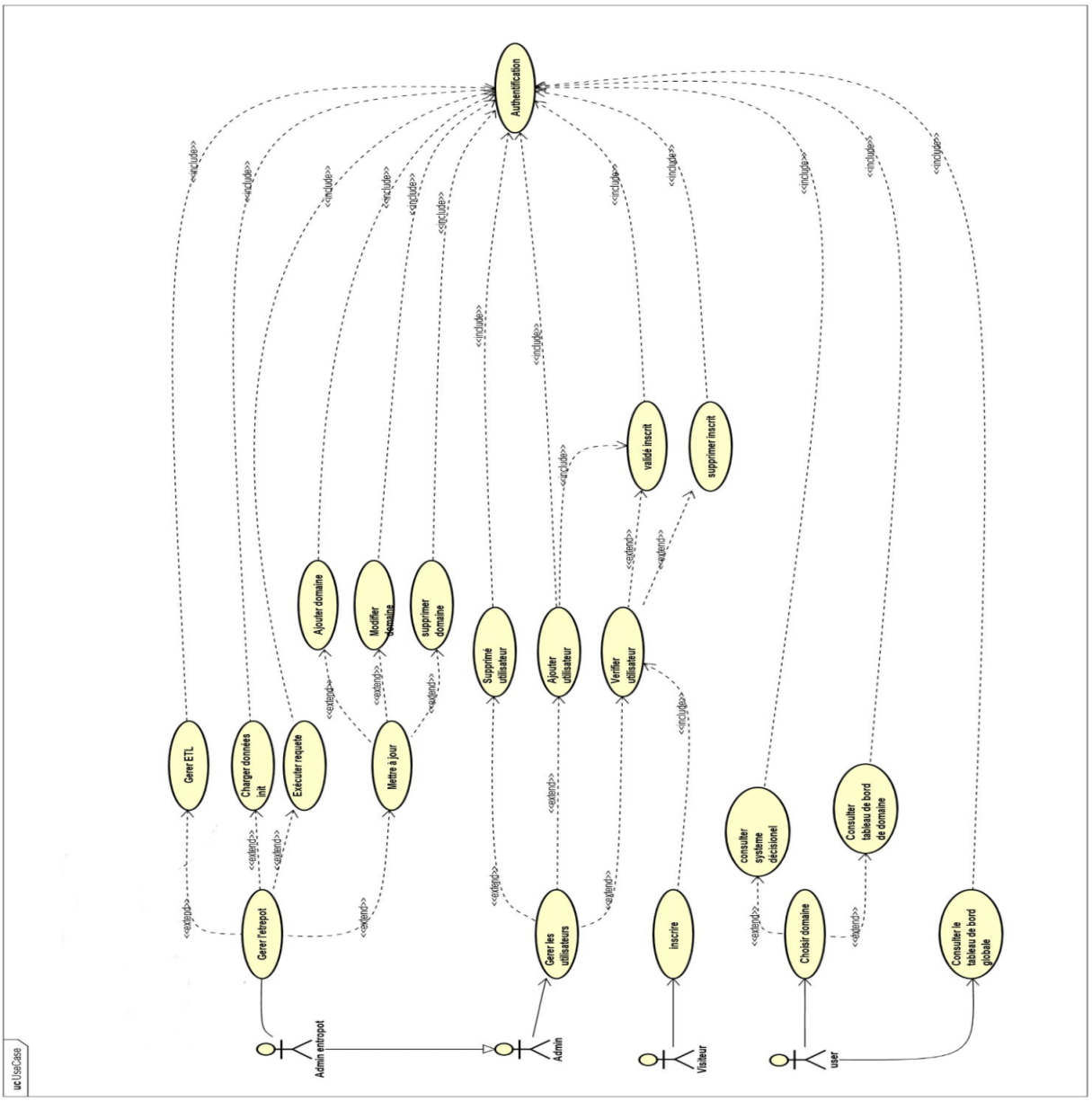


FIGURE 3.9 – Diagramme de cas d'utilisation

a. Gérer l'entrepôt

Ces cas d'utilisation sont accessible a l'administrateur de l'entrepôt, il peut :

- Charger les données initiaux : Lorsque L'entrepôt est vide, l'utilisateur peut charger les données, le chargement initiales dois par exemple, ignorer les étapes de vérification si les données existent déjà.
- Exécuter une requête SQL personnalisée : Cela offre a l'administrateur plus de contrôle et

de flexibilité, pour des but de débogage, d'évolution, de test et de modification demandée par les clients.

- Mettre a jour l'entrepôt : Modifier les différentes propriétés ou ajouter une intégration avec un nouvel entrepôt
- Et de gérer le processus ETL, pour configurer les différentes fonctions d'ETL.

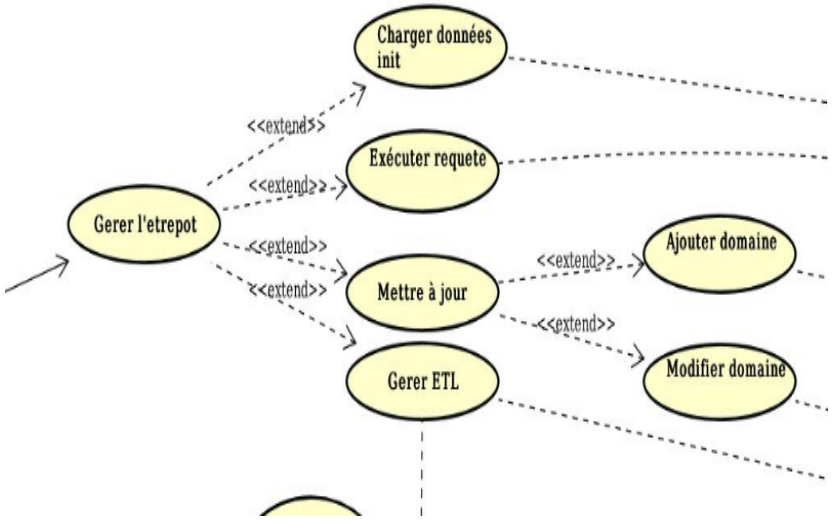


FIGURE 3.10 – Cas d'utilisation : Gérer l'entrepot

b. Aide a la décision

Les cas d'utilisation principaux de l'utilisateur final, Il peut consulter le tableau de bord de son domaine, ce domaine est spécifié lors de la création du compte= !,

. Seul l'administrateur a l'accès a tout les talbeau de bord. Un tableau de bord lui permet de visualiser de différents KPIs dans le but d'aide àla décision, Notre plate-forme aussi contient un système décisionnel qui peut directement choisir la meilleure décision pour le décideur à travers différents calculs, comme spécifié dans les besoins ci dessous.

Il peut aussi manuellement télécharger des données vers l'entrepôt sur demande en dehors du processus ETL habituel, à tout moment, pour gérer les imprévus.

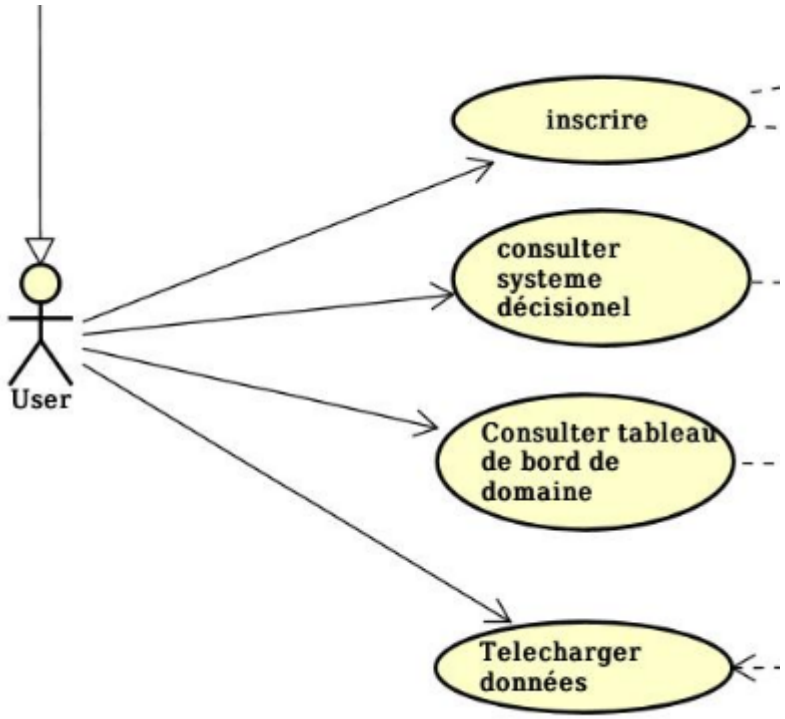


FIGURE 3.11 – Cas d'utilisation : Consulter le système BI

c. Gérer les utilisateurs

Notre Système est sécurisé par l'authentification de différents rôles, une fois un utilisateur inscrit sur la plate-forme, L'administrateur doit vérifier son compte avant qu'il puisse utiliser la plate-forme, il peut aussi refuser son inscription.

L'administrateur du système a l'accès pour ajouter manuellement ou supprimer des comptes existants.

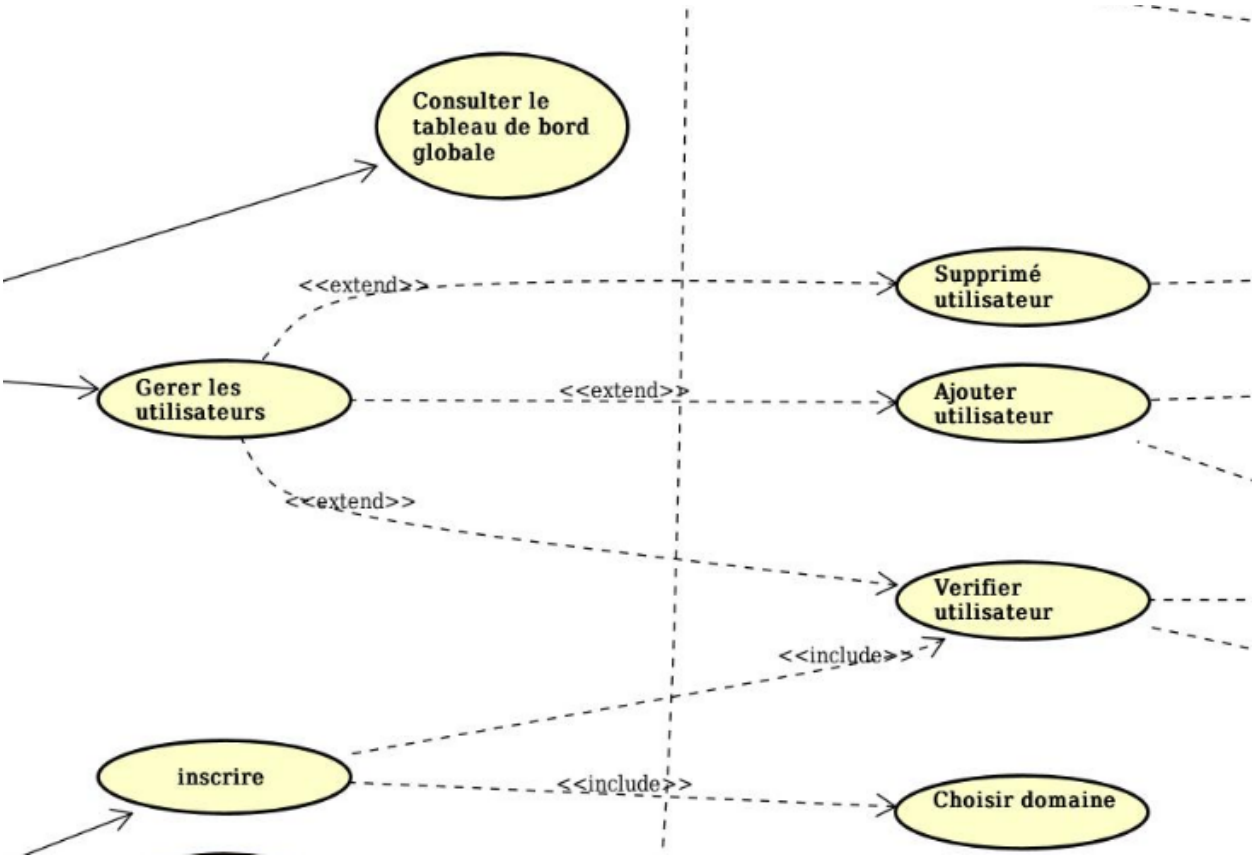


FIGURE 3.12 – Cas d'utilisation : Gestion d'utilisateurs

3.2.7 Entrepôt de données virtuel

a. Vues Matérialisés

Dans les entrepôts de données, les vues matérialisées sont utilisées pour pré calculer et stocker des données agrégées telles que la somme des ventes. Les vues matérialisées dans ces environnements sont généralement appelées résumés car elles stockent des données résumées. elles peuvent également être utilisés pour pré calculer des jointures avec ou sans agrégations. Ainsi, une vue matérialisée est utilisée pour éliminer les frais généraux associés à des jointures ou à des agrégations coûteuses pour une classe de requêtes importante [54]. Les données d'un entrepôt

peuvent être perçues comme un ensemble de vues matérialisées générées conformément aux exigences de l'utilisateur spécifiées dans les requêtes générées par rapport aux informations contenues dans l'entrepôt [55] Les exigences et les contraintes des utilisateurs changent fréquemment dans le temps, ce qui peut faire évoluer les données et afficher les définitions stockées dans un entrepôt de données de manière dynamique. Les exigences actuelles sont modifiées et des exigences nouvelles et innovantes sont ajoutées afin de prendre en compte les derniers scénarios commerciaux. En fait, les données conservées dans un entrepôt ainsi que ces vues matérialisées doivent également être mises à jour et gérées de manière à pouvoir traiter les modifications des sources de données ainsi que les exigences définies par les utilisateurs. Les sources d'information intégrées dans l'entrepôt de données sont de nature dynamique, c'est-à-dire qu'elles doivent et peuvent se transformer ou évoluer en termes d'instances et de schémas. Pour plusieurs raisons mentionnées par Thakur et Gosain dans leurs travail [55]

- Conditions ambiguës ou insuffisantes pendant la phase de développement [56]
- Modification des besoins pendant la phase opérationnelle de l'entrepôt de données, ce qui entraîne une évolution structurelle de l'entrepôt de données [57]
- Réorganisation du schéma de l'entrepôt de données pendant la phase opérationnelle de l'entrepôt de données à la suite de différentes solutions de conception décidées [57]
- De nouveaux besoins utilisateur ou du métier apparaissent ou de nouvelles versions doivent être créées [58].
- Des révisions périodiques sont effectuées afin d'éliminer les erreurs et les redondances [56]
- L'entrepôt de données doit être adapté à tout changement survenant dans les sources de données sous-jacentes.[58], [57]

Donc l'évolution des vues peut être déclenché par trois évènements :

- Les modifications effectuées directement sur l'entrepôt de données Gérée par l'expert du domaine et l'administrateur de l'entrepôt
- Les modifications effectuées au niveau des sources de données. Géré au niveau des fonctions ETL
- L'ajout d'un nouvel entrepôt de données, gérée par l'administrateur

Exemple

Supposons que l'administrateur veut créer une vue qui s'intéresse spécifiquement aux pompes françaises, la vue matérialisée sera définie comme suivant :

```
CREATE MATERIALIZED VIEW Pompes_Francaises
As
query : Select A.DESIGNATION, F.NUMSUP, F.NOM, A.PRIXU
From Achats A, Fournisseurs F
Where  A.DESIGNATION = 'POMPE' and F.PAYS = ' France ' and
End Fompes_Francaises;
```

La plupart des langages de requêtes des systèmes de gestion de bases de données, contiennent des opérateurs trés puissants pour gérer ses modifications comme dbms_mview.refresh dans oracle ou REFRESH MATERIALIZED VIEW dans postgresSQL. Qui permettent d'adapter la nouvelle vue aux changements dans les schémas sources.

On compte exploiter ces mécanismes pour avoir une extensibilité dans notre système en intégrant de nouvel entrepôts. Comme illustré dans la figure 3.12, on étend notre système en ajoutant les données du nouvel entrepot directement dans des vues matérialisées, créant un "Entrepot de données virtuel". Décider si et comment ces nouvelles données sont pertinentes par rapport aux applications BI, reste la responsabilité de l'expert du système/Administrateur de l'entrepôt de données.

b. Architecture Extensible

Les données de tout les entrepôts sont intégrés via les vues matérialisés, La pertinence de ces attributs aux applications BI est largement dépendante du context et nécessite l'avis d'un expert du domaine. La figure 3.12 présente l'architecture Extensible de notre entrepot

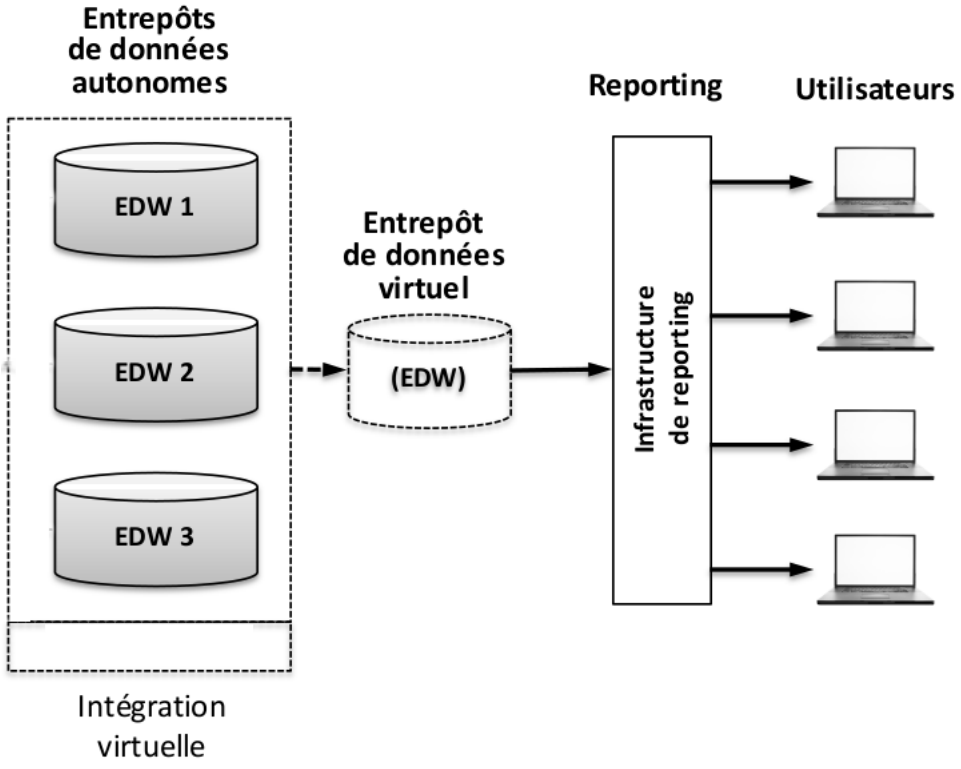


FIGURE 3.13 – Intégratin des données via les entrepots de données

3.2.8 Conception des applications BI

Un tableau de bord est un outil de reporting utilisé pour analyser de grands volumes de données afin de permettre aux utilisateurs d'enquêter sur les tendances, de prédire les résultats et de découvrir des informations. Les tableaux de bord analytiques sont plus courants dans les outils de BI car ils sont généralement développés et conçus par des analystes de données. Les tableaux de bord analytiques incluent souvent des fonctionnalités décisionnelles avancées telles que l'exploration en profondeur et l'interrogation ad hoc.

Ils sont utilisés pour surveiller l'état des indicateurs de performance clés (KPI) et sont généralement utilisés par les dirigeants. Les données derrière un tableau de bord stratégique sont mises à jour de manière récurrente, mais à des intervalles moins fréquents que celles d'un tableau de bord opérationnel. Les tableaux de bord stratégiques peuvent être consultés une fois par jour et aident les dirigeants à prendre les bonnes décision ou d'être en alerte dans le cas de risque, ou de repérer des opportunités d'investissement.

a. L'audience

L'utilisateur principal de notre plate-forme est l'investisseur, Il n'a pas de formation approfondie en informatique et en sciences de données.

Donc notre Tableau de bord doit présenter un affichage visuel des informations les plus importantes nécessaires pour atteindre un ou plusieurs objectifs. consolidées et organisées sur un seul écran afin que l'information puisse être surveillée en un coup d'œil. Et donner l'accès a l'utilisateur pour plus tard de manuellement plonger dans les détails.

Cette Information sur nos utilisateurs nous permet de choisir la technique "Personalization rather than Customization" [59],

"Customization" est faite par l'utilisateur. Un système peut permettre aux utilisateurs de personnaliser ou de modifier l'expérience pour répondre à leurs besoins spécifiques en configurant la présentation, le contenu ou les fonctionnalités du système.

Ce qu'on applique c'est la "Personalization" qui est faite par le système lui-même.Un Identifiant système est défini pour identifier les utilisateurs et leur fournir le contenu, l'expérience ou les fonctionnalités correspondant à leur rôle. C'est a dire, on présente les informations jugés plus importantes au l'utilisateur, et on lui présente uniquement les informations concernant son domaine uniquement.

b. Choisir la bonne représentation pour les données

Pour réussir à présenter efficacement vos données, Nous devons sélectionner les diagrammes adaptés à notre projet, à notre public et à votre objectif(l'aide a la décision).

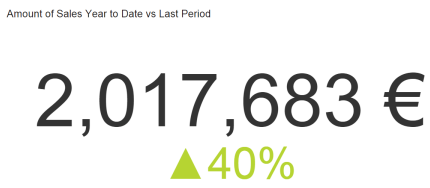


FIGURE 3.14 – Exemple de charte de chiffres[17]

Charte de chiffres particulièrement efficaces lorsqu'on souhaite présenter un aperçu immédiat et interactif d'un indicateur de performance clé, dans notre système il s'agisse d'un KPI de nombre de forages, et de besoins d'eau.

FIGURE 3.15 – Exemple d'une carte interactive pour la prévision de la météo[18]

Cartographie Tout d'abord, les cartes ont fière allure, ce qui signifie qu'elles inspireront la participation à une réunion ou à une présentation du conseil. Deuxièmement, une carte est un moyen rapide, facile et compréhensible de présenter des ensembles d'informations géographiques volumineux ou complexes à diverses fins. On utilisera les cartes pour montrer des informations comme les emplacement des forages, les données sur la populations . . .

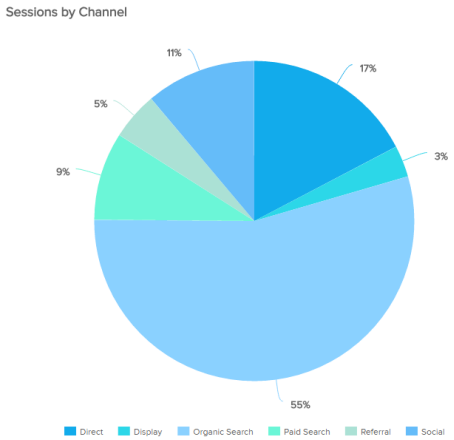


FIGURE 3.16 – Exemple de camambert [17]

Le camembert Forme un outil de visualisation utile qui fournit des métriques importantes dans un format facile à suivre. Les camemberts s'avèrent particulièrement utiles pour démontrer la composition proportionnelle d'une certaine variable sur une période de temps statique. En tant que tels, les camemberts constitueront un élément précieux de notre, on utilisera cela pour montrer les besoins d'eau par type d'utilisation, et les besoins d'eau par type de population.

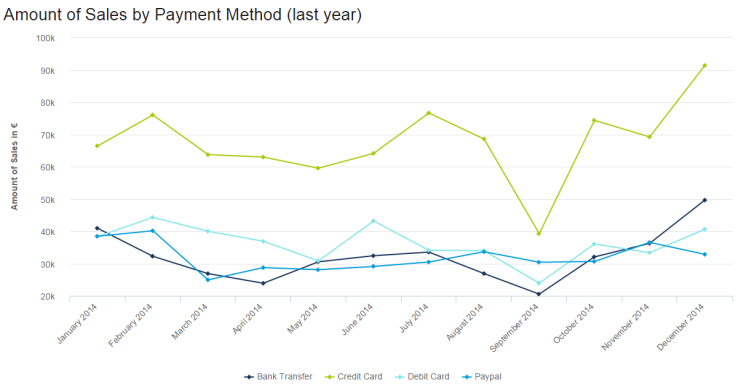


FIGURE 3.17 – Exemple de graphe de lignes [17]

Graphe de lignes Lorsqu'on présente un changement sur une période donnée avec plus qu'une petite poignée d'informations, un graphique linéaire est un moyen efficace de visualisation. De plus, les lignes facilitent le tracé de plusieurs séries ensemble.

C'est la façon classique et la plus efficace pour afficher l'évolution de différents KPIs et ce sera la base de notre visualisation ;

3.2.9 Théorie des couleurs

La plus simple de nos techniques de visualisation de données sélectionnées - la sélection du bon agencement de couleurs pour nos ressources de présentation contribuera à améliorer considérablement nos efforts.

Les principes de la théorie des couleurs [60] auront un impact notable sur le succès général de notre modèle de visualisation. Cela dit, nous devons toujours essayer de garder votre jeu de couleurs cohérent tout au long de vos visualisations de données, en utilisant des contrastes clairs pour distinguer les éléments (par exemple, les tendances positives en vert et les tendances négatives en rouge).

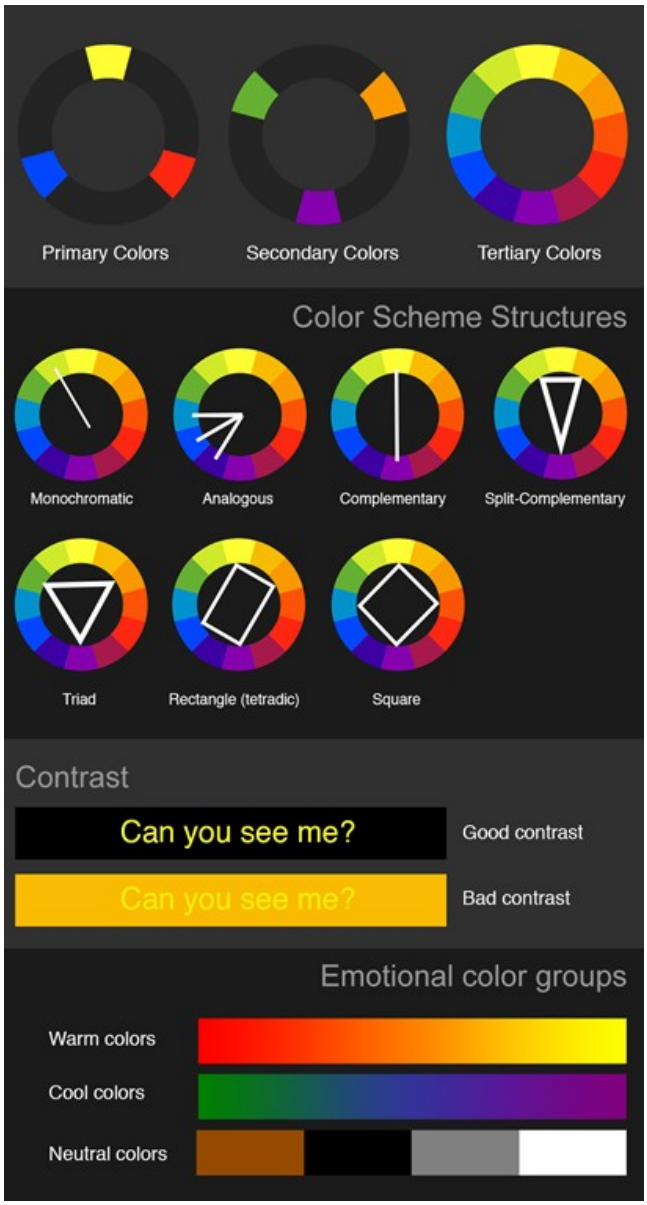


FIGURE 3.18 – Quelques exemples des principes de la théorie de couleurs [17]

En règle générale, On utilisera le rouge, le vert, le bleu et le jaune pour être reconnus et déchiffrés facilement, et pour les contrastes on utilisera un choix de couleurs Triad par rapport a la couleur choisie dans la figure spécifiée(comme dans la figure précédente), pour mettre en évidence les composants les plus importants.

3.2.10 Inclure des Comparisons

On inclus dans notre plate-forme plusieurs KPIs visualisés dans la même figure pour Laisser le décideur avoir un perspective global.

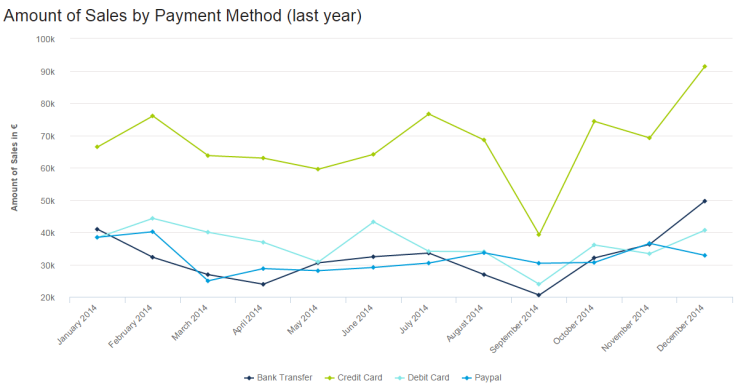


FIGURE 3.19 – De multiples graphes dans le meme visuel pour encourager la comparaison [17]

3.3 Mode de déploiment

Dans cette section, nous allons voir comment sont construits la plupart des frameworks BI grâce au modèle MVT, nous aborderons ensuite les spécificités du fonctionnement de notre plateforme et comment les éléments de notre platforme s'articulent autour du modèle MVT, que nous introduirons également. En dernier lieu, nous expliquerons le système de projets et d'applications, qui permet une séparation nette, propre et précise du code.

3.3.1 Modele MVC

Lorsque nous parlons de frameworks qui fournissent une interface graphique à l'utilisateur (soit une page web contenant des chartes de données et des cartes comme dans notre cas, soit l'interface d'une application graphique classique, comme celle de votre traitement de texte par exemple), nous parlons souvent de l'architecture MVC. Il s'agit d'un modèle distinguant plusieurs rôles précis d'une application, qui doivent être accomplis. Comme son nom l'indique, l'architecture (ou « patron ») Modèle-Vue-Contrôleur est composé de trois entités distinctes, chacune ayant son propre rôle à remplir.[61]

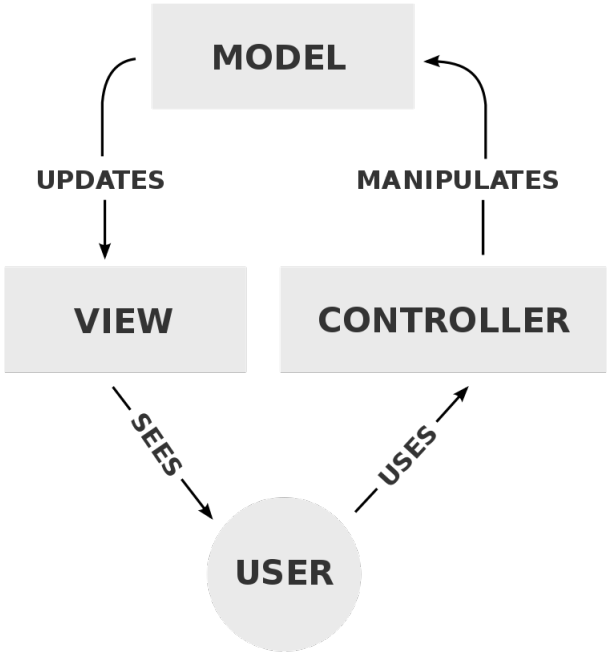


FIGURE 3.20 – Architecture du modèle MVC [19]

Tout d'abord, le modèle représente une information enregistrée quelque part, le plus souvent dans une base de données. Il permet d'accéder à l'information, de la modifier, d'en ajouter une nouvelle, de vérifier que celle-ci correspond bien aux critères (on parle d'intégrité de l'information), de la mettre à jour, etc. Il s'agit d'une interface supplémentaire entre le code et la base de données, mais qui simplifie grandement les choses.

Ensuite la vue qui est, comme son nom l'indique, la visualisation de l'information. C'est la seule chose que l'utilisateur peut voir. Non seulement elle sert à présenter une donnée, mais elle permet aussi de recueillir une éventuelle action de l'utilisateur (un clic sur un lien, ou la soumission d'un formulaire par exemple). Typiquement, un exemple de vue est une page web, ni plus, ni moins.

Finalement, le contrôleur prend en charge tous les événements de l'utilisateur (accès à une page, soumission d'un formulaire, etc.). Il se charge, en fonction de la requête de l'utilisateur, de récupérer les données voulues dans les modèles. Après un éventuel traitement sur ces données, il transmet ces données à la vue, afin qu'elle s'occupe de les afficher. Lors de l'appel d'une page, c'est le contrôleur qui est chargé en premier, afin de savoir ce qu'il est nécessaire d'afficher.

3.3.2 Le modèle MVT

Cette Architecture diffère légèrement de l'architecture MVC classique, elle reprends les définitions de modèle et de vue que nous avons vues, et en introduit une nouvelle : le template (voir figure suivante). Un template est un fichier HTML, aussi appelé en français « gabarit ». Il sera récupéré par la vue et envoyé au visiteur ; cependant, avant d'être envoyé, il sera analysé et exé-

cuté par le framework, comme s'il s'agissait d'un fichier avec du code. Notre plate-forme dois permettre, dans le code web, d'afficher des variables, d'utiliser des structures conditionnelles (if/else) ou encore des boucles (for), et surtout elle dois pouvoir afficher une visualisation des KPIs

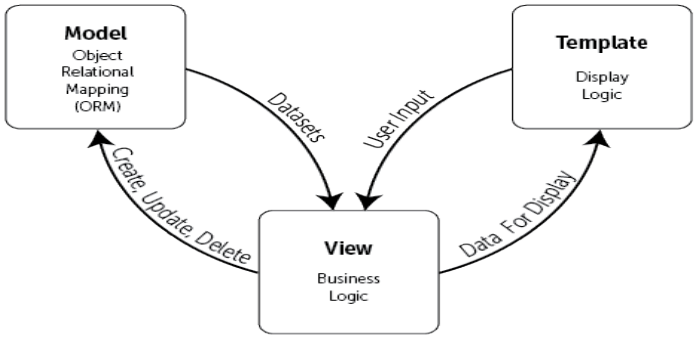


FIGURE 3.21 – Architecture du modèle MVT [20]

On distingue quatre points qu'on dois gérer dans la phase de développement :

- Le routage des requêtes, en fonction de l'URL ;
- La représentation des données dans l'application, avec leur gestion (ajout, édition, suppression...), c'est-à-dire les modèles ;
- L'affichage de ces données et de toute autre information au format HTML, c'est-à-dire les templates ;
- Enfin le lien entre les deux derniers points : la vue qui récupère les données et génère le template selon celles-ci.

Si le template est un fichier HTML classique, un modèle en revanche sera écrit sous la forme d'une classe où chaque attribut correspondra à un champ dans la base de données. Notre plate-forme se chargera ensuite de créer la table correspondante dans la base de données, et de faire la liaison entre la base de données et les objets de notre classe. Non seulement il n'y a plus besoin d'écrire de requêtes pour interagir avec la base de données, même s'il y a toujours la possibilité de le faire, mais en plus ce modèle propose la représentation de chaque entrée de la table sous forme d'une instance de la classe qui a été écrite. Il suffit donc d'accéder aux attributs de la classe pour accéder aux éléments dans la table et pouvoir les modifier, ce qui est très pratique !

Enfin, la vue est une simple fonction, qui prend comme paramètres des informations sur la requête (s'il s'agit d'une requête GET ou POST par exemple), et les paramètres qui ont été donnés dans l'URL. Par exemple, si l'identifiant ou le nom d'une certaine charte a été donné dans l'URL hydro.com/visualisations/NiveauDynamiqueForage, la vue récupérera NiveauDynamiqueForage comme titre et cherchera dans la base de données les données correspondantes à afficher. Suite à quoi la vue générera le template avec la bonne visualisation et le renverra à l'utilisateur.

3.3.3 Intégration

Toutes les composants de la plate-forme seront intégrés dans ce framework. Et donc sera développé sous forme de plusieurs applications (ETL, Les visualisations du tableau de bord, et le système cartographique) ayant chacune un ensemble de vues, de modèles et de schémas d'URL. ces applications sont réutilisables dans d'autres projets, puisque chaque application est indépendante.

3.4 Conclusion

Dans cette section, Nous avons présenté le cycle de vie BI proposé par Kimball, et nous avons détaillé son application a notre projet étape par étape, nous avons maintenant une conception de notre système BI qui sera utilisée pour l'implémentation.

Dans ce qui suit, nous allons faire une étude comparative entre les différentes méthodes d'implémentation d'un système DW/BI, et présenter notre implémentation du système conçu dans ce chapitre.

Chapitre 4

IMPLEMENTATION

4.1 Introduction

Ce chapitre présente les différentes étapes suivi pour l'implémentation de notre système BI/DW suivant toujours le cycle de vie de Kimball [16], ce chapitre élaborera une etude comparative entre les différents paradigmes d'implémentation des systèmes BI/DW, les outils possibles pour le choix et installation de produits, la coenception physique, puis l'implémentation de l'ETL extensible et L'implémentation des applications BI puis ensuite présenter l'intégration de toutes ces composants dans une plateforme BI Web.

4.2 Choix et Installation des produits

On souhaite avoir un système qui nous permettra d'extraire les données facilement, tout le système dois utiliser le meme type de mécanismes d'extraction et d'intégration, on souhaite aussi pouvoir établir des relations entre plusieurs sources et des agrégations.

Le principal défi ici, est de maintenir la cohérence et l'exactitude des données face aux changements et aux évolutions de l'entrepôt, on souhaite avoir un système qu'on peut facilement personnaliser du coté implémentation, sans casser les autres processus.

A un certain point aussi, la complexité devient un défi, le nombre de tables augmente énormément et ça devient difficile a gérer cette dernière, il doit être facile d'expliquer toutes ces tables et fonctions et comment les utiliser.

Le choix des techniques et outils d'implémentation est très important pour prévoir la gestion de ces défis, et peut faire la différence dans l'extensibilité (ou non) de l'entrepôt.

4.2.1 Paradigme D'implémentation

Dans cette section, on va explorer la différence entre les paradigmes d'implémentation de systèmes BI/DW ; les outils "basés sur les clics" (comme Talend, Pentaho ..), les scripts de flux de données, et les grandes plate-formes comme Apache Airflow.

a. Outils Click Based

La façon traditionnelle de faire cette derniere est d'utiliser un outil click-based(basé sur les clics) comme talend

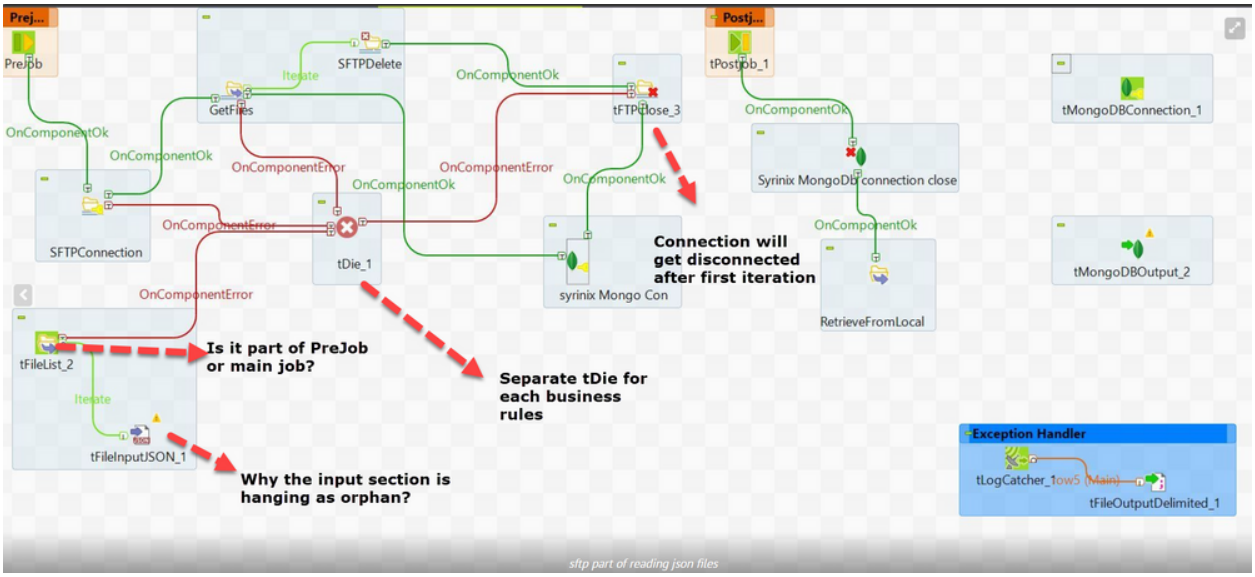


FIGURE 4.1 – Complexité d'un des outils click based a la grande échelle (pentaho)

Dans un outil pareil on crée des flux de données visuellement, qu'on peut modifier, laisser des commentaires et c'est tres facile a mettre en place. Ces outils contiennent des mécanismes prédéfinis pour se connecter à plusieurs bases, des outils prêts pour la manipulation des données, et de leurs visualisation. Ils offrent un gain énorme en temp et efforts qu'il faut investir pour créer et déployer de tels systèmes.

Il contient aussi des outils de visualisation hautement dynamiques. Par exemple Power BI peut visualiser les données en utilisant différents visuels et des options personnalisables élevées pour chaque graphique.

Dr. Martin Loetzsch dans [62] recommande fortement d'éviter cela à grande échelle lorsqu'on vise l'extensibilité, lorsque le système grandis ça devient très complexe a déboguer lorsque nous avons des défaillances, parce qu'on dois cliquer sur tous les noeuds et tester et retester a chaque tentative de correction pour pouvoir repérer où est l'erreure sans avoir une vue globale sur ce qui ce passe. C'est aussi trés difficile a changer,parce que seul la personne qui a créé le réseau qui comprends bien le schéma, c'est très difficile de répartir ce genre de tâches sur une équipe par exemple.

"L'une des principales tendances de l'entreposage de données et de l'ingénierie des données est la transition des outils basés sur des clics vers l'utilisation de code pour la définition de flux de données. De nos jours, la grande majorité des projets commencent avec un ensemble de scripts ou avec des plate-formes telles que Luigi ou Apache Airflow, ces deux derniers devenant clairement les principaux acteurs. Au cours des 6 dernières années [62]

b. L'approche Hard-Coding (scripts)

Elle peut être aussi simple qu'avoir quelques fichiers SQL, des fichiers BASH et des script PYTHON, et lorsque nos données ne sont pas le résultat d'exécution d'un schéma dans un outil basé sur les clics, mais un résultat de l'exécution du code, on peut appliquer plusieurs différentes bonne pratiques de développement de logiciels. [63]

C'est beaucoup plus facile de comprendre le raisonnement lorsque c'est du code bien formaté, on peut l'inspecter, on peut utiliser une variété d'outils de débogage de codes disponibles dans le marché [64], on peut profiter des logiciels de gestion de versions décentralisés comme github [65], et donc on peut le mettre dans un environnement de mise en scène, le tester, avant de commettre les changements.

Pouvoir coder aussi offre une très grande flexibilité qui n'est pas présente dans des outils préconstruits. On peut d'une meilleure façon inclure de nouvelle opération et gérer les exceptions.

Python comme langage est supporté par la plus grande communauté en ce qui concerne sciences de données et manipulation de données, et donc on peut trouver une très grande quantité de bibliothèques et de la documentation bien détaillée dans le domaine.

c. Apache Airflow

Apache Airflow d'un autre coté (actuellement en statut d'incubateur, ce qui signifie qu'il n'a pas encore été approuvé par Apache Software Foundation) est un système d'automatisation et de planification de flux de données. Il peut être utilisé pour créer un pipeline de données pour alimenter un entrepôt de données et (avec un peu de codage) pour développer des processus ETL réutilisables et paramétrables. Bien qu'il soit utilisé dans le processus ETL, Airflow n'est pas un outil interactif ETL.

Airflow aussi dépends largement d'une connexion internet fonctionnelle, ce qui n'est pas très avantageux pour nous, on préfère avoir dans la région du sud, un système indépendant de la connexion internet, pour plus de stabilité.

Plus la complexité de la configuration d'un système tel que airflow est plus adapté a un système de l'échelle de petabytes [62],

Voici un tableau qui résume notre comparaison entre les différentes implémentation d'un système BI, l'évaluation de ces paradigmes est faite basé sur [62] et nos propres tests de ces derniers.

Critere	 Scripts Python	 Outils Click Based	 Apache Airflow
Open Source	✓	Généralement	✓
Visualisation de l'ETL	Code	Graphique	Hybride
Compatibilité avec les sources de données	Toutes	Un ensemble prédéfini de type de sources les plus connues	Toutes
Facile à mettre en place	✓✓✓	✓✓✓✓✓	✓
Flexibilité	✓✓✓✓✓	✓	✓✓✓
Scalabilité	Scalable	Complex	Trés scalable
Débogage	Relativement Facile	Complex	Trés complex
Hors connexion	Selon l'implémentation	Généralement	Non
Mise en scène(staging) avant le déploiement	✓		
Outils de débogage	✓		✓

TABLE 4.1 – Comparaison entre les systèmes d'implémentation de systèmes BI

Pour les raisons mentionnées auparavant, on décide d'utiliser python pour créer notre propre plate-forme BI personnalisé et extensible.

4.2.2 Outils et bibliothèques

Les bibliothèques Python comprennent plusieurs fonctionnalités qui permettent à l'utilisateur d'évaluer et d'analyser les ensembles de données et de produire des résultats effectifs. Le langage de programmation Python est devenu un outil robuste et puissant pour l'analyse de données à l'aide de ces bibliothèques. Une de ces puissante bibliothèques est Pandas

Puisque notre système vas être implémenté en python, on utilise le framework Django, pour pouvoir Implémenter notre système dans une plate-forme BI web, Django est un framework de développement web open source en Python. Il a pour but de rendre le développement web 2.0 simple et rapide. Pour cette raison, le projet a pour slogan « Le framework web pour les perfectionnistes sous pression ». Développé en 2003 pour le journal local de Lawrence (Kansas), Django a été publié sous licence BSD à partir de juillet 2005. accéssible d'aprés tout les appa-

reils mobiles et Desktop. Ce n'est pas le seul dans sa catégorie, nous pouvons compter d'autres frameworks Python du même genre comme web2py, TurboGears, Tornado ou encore Flask. Il a cependant le mérite d'être le plus exhaustif, d'automatiser un bon nombre de tâches et de disposer d'une très grande communauté.

Django est créé en 2003 dans une agence de presse, Lawrence Journal-World, qui devait développer des sites web complets dans des laps de temps très courts (d'où l'idée du framework). En 2005, cette agence décide de proposer Django au grand public, le jugeant assez mature pour être réutilisé n'importe où. Trois ans plus tard, la fondation Django Software est créée par les fondateurs du framework afin de pouvoir maintenir celui-ci et la communauté très active qui l'entoure.

Aujourd'hui, Django est devenu très populaire et est utilisé par des sociétés du monde entier, telles qu'Instagram, Pinterest, et même la NASA !

Ce framework est composé de trois parties distinctes :

- Un langage de gabarits flexible qui permet de générer du HTML, XML ou tout autre format texte ;
- Un contrôleur fourni sous la forme d'un « remapping » d'URL à base d'expressions rationnelles ;
- Une API d'accès aux données est automatiquement générée par le cadre compatible CRUD. Inutile d'écrire des requêtes SQL associées à des formulaires, elles sont générées automatiquement par l'ORM.

Il est important que ce soit le cas pour le décideur pour plusieurs raisons ; Le décideur n'a pas forcément une formation approfondie en informatique, et donc il ne faut pas créer un systeme qui nécessite beaucoup de configuration de d'installation de la part de l'utilisateur final, une plate-forme web vas fonctionner sans installation et sans beaucoup de pré-requis de l'appareil. L'installation et la configuration se fera au niveau du serveur a la place. Notre plate-forme dois être Implémenté de façon responsive et dois pouvoir s'adapter aux naviguateurs et aux appareils. Pour système d'exploitation on choisit linux a cause de sa sécurité, il est aussi plus facile de configurer et installer les bibliothèques sous ce système et on trouve qu'il offre un environment de dévleloppement plus propre. Pour système de gestion de base de données(SGBD) relationnel on choisit PostgresSQL, ce SGBD offre beaucoup d'outils compatibles avec les vues et les vues matérialisés ce qui nous permettera d'implémenter notre approche de l'extensibilité en ajoutant d'autres entrepots, Postgres est compatible avec Python et Django. On utilise une variété d'autre outils offerts par la communauté de Data Science pour python pour faciliter l'implémentation de notre approche ETL, faciliter l'extraction, la manipulation et l'organisation de données, et meme le chargement :

a. Pandas

La Bibliothèque de pandas est utilisée pour la manipulation de données basée sur la structure de données NumPy elle fournit également diverses fonctions dans l'analyse de la finance, des statistiques, des sciences sociales. Cette bibliothèque propose des outils permettant de transformer

les données brutes en ensembles de données utiles. Il fournit également plusieurs fonctions pour accéder, indexer, fusionner ou regrouper des données facilement.

b. Numpy

Numpy est une bibliothèque Python, qui prend en charge de grands tableaux multidimensionnels et matrices, ainsi qu'une vaste collection de fonctions mathématiques de haut niveau pour opérer sur ces tableaux. Il contient entre autres : un puissant objet matriciel à N dimensions, des fonctions sophistiquées (de diffusion), une algèbre linéaire utile, une transformée de Fourier et des capacités de nombres aléatoires ...

c. Psycopg2

est l'adaptateur de base de données PostgreSQL le plus populaire pour le langage de programmation Python. Ses principales caractéristiques sont l'implémentation complète de la spécification Python DB API 2.0 et la sécurité des threads (plusieurs threads peuvent partager la même connexion). Il a été conçu pour des applications fortement multithreads qui créent et détruisent de nombreux curseurs et génèrent un grand nombre de «INSERT» ou de «UPDATE» simultanés. Toutes ces bibliothèques sont inclus dans

d. Annaconda

l'environnement de développment python Annaconda, qui est un tout-en-un pour les outils de ce language, cela guaranti la bonne installation et configuration de bibliothèques et nous evitera les problemes reliés a cela.

4.2.3 Visualisation

Pour la visualisation de données, on avait plusieur choix :

a. Chartjs

Ce plugin permet d'insérer des graphes dans les plate-formes Web. Il inclut 6 types (courbe, barre, camembert, radar, doughtnut) ainsi qu'une solution pour les vieilles versions d'Internet Explorer. Basé sur HTML5, Chart.js utilise le javascript et repose sur le tag HTML5 canvas. Il est compatible avec tous les navigateurs modernes et offre un support polyfills pour IE7/8. Il est aussi responsive par défaut. Simple et flexible Chart.js ne dépend pas d'une autre librairie et il est assez léger une fois compressé.

b. ChartIT

Django Chartit est une application Django qui permet de créer facilement des graphiques à partir des données de votre base de données. Les graphiques sont rendus à l'aide des bibliothèques

JavaScript Highcharts et jQuery. Les données de votre base de données peuvent être tracées sous forme de graphiques linéaires simples, de graphiques en colonnes, de graphiques en aires, de diagrammes de dispersion et de nombreux autres types de graphiques. Les données peuvent également être tracées sous forme de graphiques croisés croisés dynamiques où les données sont regroupées et / ou pivotées selon une ou plusieurs colonnes spécifiques.

c. Cubesviewer

CubesViewer est une application et une bibliothèque HTML5 visuelles et réactives permettant d'explorer et de visualiser différents types de jeux de données.CubesViewer est divisé en deux parties : l'application côté client et une application Web facultative côté serveur (celle-ci), qui ajoute des fonctionnalités telles que l'enregistrement et le partage de vues.CubesViewer peut être utilisé pour l'exploration et l'audit de données, la génération de rapports, la conception et l'intégration de graphiques, et en tant qu'application d'analyse (simple) à l'échelle de l'entreprise.

d. D3js

D3 est une bibliothèque puissante avec une tonne d'utilisations. Nous nous interressons a une application particulièrement convaincante de D3 : la cartographie. D3.js peut lier n'importe quelle donnée arbitraire à un DOM (Document Object Model), puis appliquer des transformations au document pilotées par JavaScript, CSS, HTML et SVG. Le résultat peut être une simple sortie HTML ou des graphiques SVG interactifs avec un comportement dynamique comme des animations, des transitions et des interactions. Toutes les transformations de données et les rendus sont effectués côté client, dans le navigateur.

e. Geodjango

GeoDjango signifie Geographic Django. Il s'agit d'un framework qui fait partie de Django et peut être utilisé pour créer des applications utilisant des données géospatiales ou géographiques. Ces applications peuvent être des systèmes SIG (système d'information géographique) à part entière ou de simples applications basées sur la localisation Web.

L'intérêt de GeoDjango réside dans son intégration étroite avec l'ORM de Django, qui vous permet de manipuler des données spatiales dans des bases de données géospatiales telles que PostGIS, Oracle ou PostgresSQL, ou dans toute autre base de données à partir d'une API de haut niveau, qui permet de résumer les différences entre les différentes implémentations et les complexités associées aux requêtes spatiales.

f. Résumé

On choisit Chartjs pour la visualisation de données et la comparaison des chartes graphiques de différents types, a cause de sa capabilité de gérer une trés grande quantité de données dans une plate-forme Web et la simplicité de sa configuration et haute compatibilité avec Django, Pour

la cartographie 3djs offre tout ce que nous avons besoin, mais la complexité de la configration d'un tel outil nous a poussé a utiliser GeoDJango l'utilisation de ce dernier est plus simple, et le projet vu le lapse de temps que nous avons est plus réalisable.

4.3 Conception physique

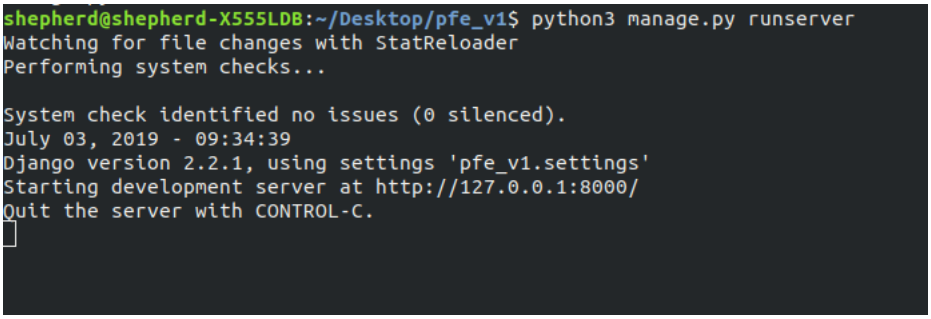


FIGURE 4.2 – Le serveur Python en Exécution

Notre système s'exécute sur une seule machine dans un environnement virtuel Django qui nous permettra d'exécuter des scripts python dans une plate-forme WEB, Cette plate-forme peut être accédée directement de n'importe quel appareil. Les données aussi sont centralisés dans le même serveur, Le serveur Django (figure 4.2) et PostgresSQL (figure 4.3) doivent être en exécution pour que les application

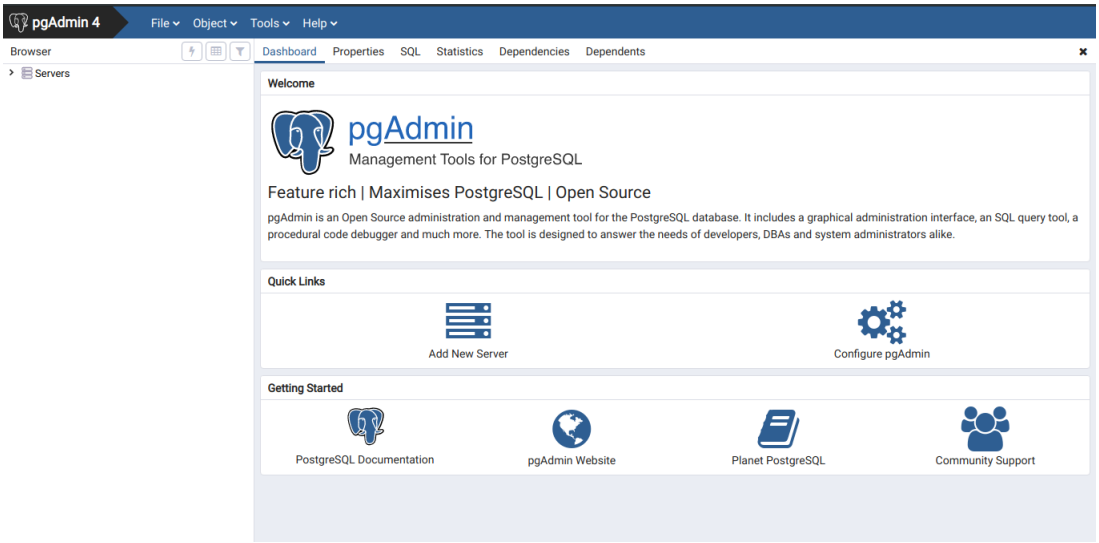


FIGURE 4.3 – Interface d'accueil du SGBD Postgres : pgAdmin

4.4 Implémentation du Système ETL

Dans cette section on détaillera les techniques que nous avons utilisé pour implémenter notre approche d'ETL extensible, ainsi que son intégration dans la plate-forme BI.

4.4.1 Modèle d'extraction de données

a. Entete

Pour effectuer l'ETL correctement dans notre plate-forme, et puisque pas toutes les données sont digitalisées dans le domaine. Nous avons choisi de donner a l'utilisateur des modèles de fichiers Excels dans lesquels il peut remplir ses données, ces fichiers Excels contiennent des données supplémentaires qui peuvent être altérés pour modifier la configuration de l'ETL. L'entete Contiens les informations suivantes :

1. La Pertinence de l'attribut au système : Un attribut marqué 1 (pertinent) sera traité par le système, un attribut marqué 0 (non-pertinent) sera complètement ignoré.
2. Le Nom de l'attribut source : Le nom de l'attribut dans la source de données, représente souvent la signification de l'attribut, et peut être utilisé pour la visualisation ou le débogage, utile aussi pour l'utilisateur lorsqu'il remplis les données dans le fichier
3. Le Nom de l'attribut cible : Le nom de l'attribut cible dans l'entrepôt de données utilisé pour correctement formuler les requêtes
4. Le Nom de la table(ou de fonction) cible : Dans le cas ou on charge directement l'attribut dans une table dans la base de données, cette propriété prendra la valeur du nom de la table en question, sinon elle prendera le nom de la fonction qui l'utilise pour les calculs
5. Indicateur si l'attribut représente une clé : les attributs clés traités d'une manière différente dans la pluspart des fonctions, et ce sont eux qui sont prises en compte dans les fonctions comme "count"

L'exemple d'entête de fichiers est dans la figure 4.4

4.4.2 Implémentation des fonctions ETL

a. Etape1 : connexion et chargement du fichier

La premeire étape nécessaire est de se connecter et de charger les données, on utilise la bibliothèque psycopg2 pour maintenir des connexions à plusieurs bases de données en parallèle.

1	CODE_COMMUNE	NOM_COMMUNE	NOM_FORAGE	TYPE_NAPPE	Zone_Desservie	X	Y
2	1	1	1	1	1	1	1
3	code_commune	nom_commune	nom_forage	type_nappe	zone_desservie	coord_x	coord_y
4	td_commune	td_commune	td_forage	td_nappe	td_forage	td_forage	td_forage
5	CLE						

FIGURE 4.4 – Exemple du modèle de données offert a l'utilisateur

```
# Connection
def Connection (user,password,database):
        try:
                con = psycopg2.connect(host = "127.0.0.1", database
= database, user = user, password = password, port = "5432")
                print ("*** Vous etes maintenant connecté ***")
                return con
        except (Exception, psycopg2.Error) as error :
                print(error)

# Read excel file
def ReadEXCEL(path):
        df = pd.read_excel(path)
        return df
```

FIGURE 4.5 – Les foncitons permettant la connexion et chargement du fichier

b. Etape2 : Chargement des dimentions/Mise a jour

L'étape qui suit c'est bien l'étape de l'extraction des données, les spécifications exactes du processus dépendent largement de la fonction ETL concernée, la figure exemple d'une fonction ETL générique permettant l'extraction de données qui sont extraites directement vers les tables de l'entrepôt (souvent le cas pour les dimensions).

De la même façon la mise a jour est effectuée lorsqu'il existe plusieurs sources contenant les mêmes informations pour une des sources, celle qui a cette information comme clé en priorité. et pour toutes les autres elles exécutent la fonction de mise à jour, pour ajouter celle qui ne sont pas prises en compte.

Ces deux fonctions "outils" sont utilisés dans les différentes fonction ETL pour extraire les

```
# Insert fonctions isintance
def Insert(connection, table, val,attrib):
        cursor = connection.cursor()
        if isintance(val,str):
                cursor.execute("insert into {}".format(table)+"
values ('{}') ".format(val)+" ON CONFLICT ({}) DO
NOTHING;".format(attrib))
        else :
                cursor.execute("insert into {}".format(table)+"
values ({}) ".format(val)+" ON CONFLICT ({}) DO
NOTHING;".format(attrib))
```

FIGURE 4.6 – Les fonctions permettant la connexion et chargement du fichier

```
# Update fonction
def Update(connection, table, attrib,val,cle,cleval):
        cursor = connection.cursor()
        querystr = "update {}".format(table)+" set
{}".format(attrib)+" = '{}'".format(val)+" where {}".format(cle)+"
= {}".format(cleval)+";"
        if isintance(val,str):
                if isintance(cleval,str):
                        cursor.execute("update {}".format(table)+"
set {}".format(attrib)+" = '{}'".format(val)+" where {}".format(cle)
+" = '{}' ;".format(cleval))
                else:
                        cursor.execute("update {}".format(table)+"
set {}".format(attrib)+" = '{}'".format(val)+" where {}".format(cle)
+" = {} ;".format(cleval))
        else:
                if isintance(cleval,str):
                        cursor.execute("update {}".format(table)+"
set {}".format(attrib)+" = {}".format(val)+" where {}".format(cle)+"
= '{}' ;".format(cleval))
                else:
                        cursor.execute("update {}".format(table)+"
set {}".format(attrib)+" = {}".format(val)+" where {}".format(cle)+"
= {} ;".format(cleval))
```

FIGURE 4.7 – Les fonctions permettant la connexion et chargement du fichier

données des fichiers source de données, les calculs, les agrégations, les moyennes, sont calculés selon la logique nécessaire pour la fonction. (voir etape3)

Le déroulement de l'algorithme est le suivant :

- Le curseur pointe sur le debut du document.
- Il récupère les informations dans l'entête, Il charge les clés.

- La bibliothèque psycopg2 nous permet d'exécuter des requêtes sur PostgresSQL directement et la a structure de Django en MVT(voir facilite l'écriture de requêtes SQL de chargement de données personnalisées (instructions complètes) ou simplement de clauses WHERE personnalisées en tant que paramètres.
- Dans le cas ou on a besoin d'effectuer des traitements, ces données sont chargées dans un dataframe pour être manipulés via Python directement.

L'ETL est effectué périodiquement (Mise a jour hebdomadaire par défaut) un script Bash est déclenché par le serveur, ce script lancera tout les script python nécessaires pour effectuer cette opération.

La bibliothèque psycopg2 nous permet d'exécuter des requêtes sur PostgresSQL directement, La structure de Django facilite l'écriture de requêtes SQL personnalisées (instructions complètes) ou simplement de clauses WHERE personnalisées en tant que paramètres personnalisés ..

Dans cas ou les données doivent être extraites directement, il n'y a pas de besoin de passer par Pandas, les données sont chargés directement.

c. Etape3 : Traitement des données et calcul des faits

Pour les faits, plusieurs transformations, corrections, agrégations, jointures et calculs doivent être faits pour réaliser correctement le processus ETL et alimenter les tables de faits.

On exploite la puissance de la bibliothèque Pandas ici pour manipuler les tables sous forme de Dataframes [66] et utiliser la logique python pour faire les opérations nécessaires sur ces données.

Toutes les requêtes SQL de l'anatomie suivante peuvent être traduis en code Pandas vu les fonctionnalités qu'il offre :

```
SELECT... FROM... WHERE...
GROUP BY... HAVING...
ORDER BY...
LIMIT... OFFSET...
```

Donc aprés le raffinement du dataframe qu'on veut charger dans l'entrepot de données, on utilise SQLAlchemy pour insérer directement le Dataframe vers la Base de données pour charger les données, le raffinement consiste en suppression des données non pertinentes, suppression des index de lignes, résoudre les problemes uniformité des données, group by, count.

Dans ce qui suit, quelques exemples des implémentations des fonctions ETL :

TF_Forages Cette table est utilisée pour stocker le Nombre de forages

- Par année

- Par zone
- Par nappe exploitée

La fonction principale générant ces données la est définie comme dans la figure 4.8 Remar-

```
def tf_nbforages(dataframe, engine):
    try:
        #Create
        engine = create_engine('
            postgresql+psycopg2://postgres:123456@localhost:5432/dw_hydro')
        df = pd.read_excel('DataSourceH.xlsx', index_col=False)
        data = df.groupby([communes().codecommune(),nappes().codenappe(),annees()]
            ).size().to_frame('nbforage').reset_index()

        data.to_sql('tf_nbforage',engine, if_exists='append', index = False)

    except (Exception, psycopg2.Error) as error :
        print ( error)
```

FIGURE 4.8 – Fonction d'ETL de la table TF_Forages

quons que la fonction utilise des fonctions communes(), nappes(), annees(), cela est due a la dépendance de de la fonction TF Forages des données générés par ces fonctions.

L'ETL est effectué périodiquement (Mise a jour hebdomadaire par défaut) un script Bash est déclenché par le serveur, ce script lancera tout les script python nécessaires pour effectuer l'ETL.

Pour l'interface de L'ETL one-shot(sur demande) elle est trés simple, L'utilisateur n'a pas besoin de savoir le fonctionement interne des fonctions ETL. Il choisis un domaine de données, le fichier template est affiché pour ce domaine dans le cas ou il a des données non digitalisées.

Il a un champ pour insérer l'adresse du fichier EXCEL contenant des données, et un boutton «parcourir... » pour selectionner le fichier a partir de l'explorateur des fichiers du système. Lancer L'ETL est un boutton qui permet d'exécuter l'opération. Une barre de chargement est affichée. S'il y a des erreurs ils sont affichées a l'utilisateur. Sinon « Succés » sera affiché a l'utilisateur. Juste après l'opération d'ETL, quelques KPIs d'alerte sont consultés pour envoyer des notifications a l'utilisateur.

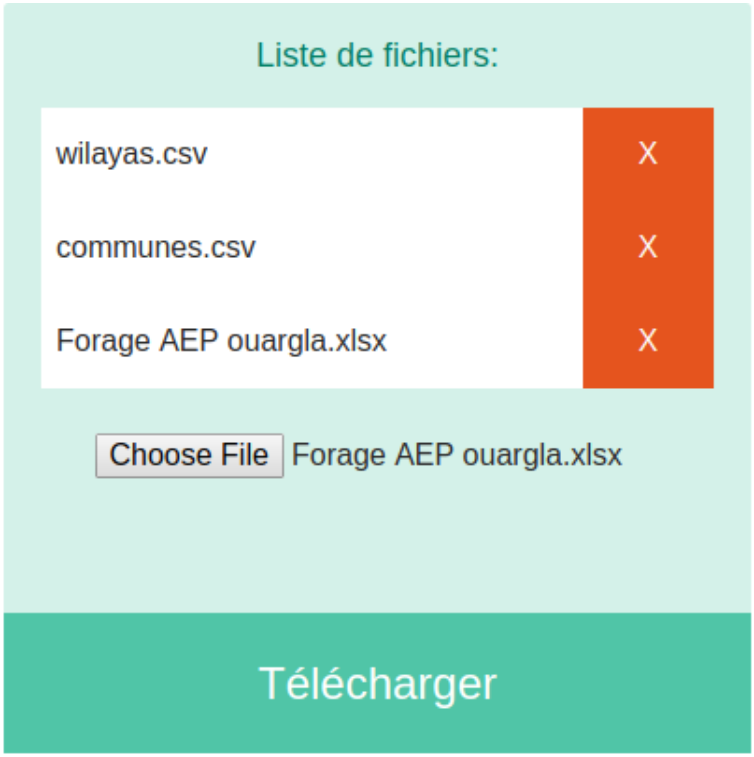


FIGURE 4.9 – Interface ETL one shot

4.5 Implémentation des applications BI

4.5.1 Tableau de Bord

Le tableau de bord principal consiste de plusieurs visualisations de données interactive des KPIs qui interressent l'utilisateur (selon les besoin définis dans le chapitre précédent), l'outil de visualisation principal utilisé est Chartjs. Chartjs récupere les données exportées des Vues Matérialisés des entrepôts sous format json. Il affiche a partir des données récupérées des Chartes organisées par ordre d'importance, selon l'utilisateur, et en prenant en considération l'espace limité. Les données sont visualisées différemment, le type de charte a afficher est selon la nature de la donnée, la quantité de données, et les préférences de l'expert.

Chartjs est notre outil principal de visualisation, c'est un plugin javascript du coté client, il ne dois pas donc avoir accès a la BDD, les données sont récupérés de la base de données par Django envoyés au coté client via Jquery. sous forme des fichiers JSON.

Pour donner un exemple d'une charte qui peut être Interprétée par Chartjs voici la figure

```
class ChartData(APIView):
    autentification_classes = [Dw_hydro]
    permission_classes=[]
    def get(self, request, format=None){
        content = json.loads(response.content)
        data ={
            'année' = content.['anne']
            'commune' = content.['commune']
            'nappe' = content.['nappe']
            'nb_forages' = content.['nb_forages']
        }
    }
    return Response(data)
```

FIGURE 4.10 – Récupérer les données d'un fichier JSON

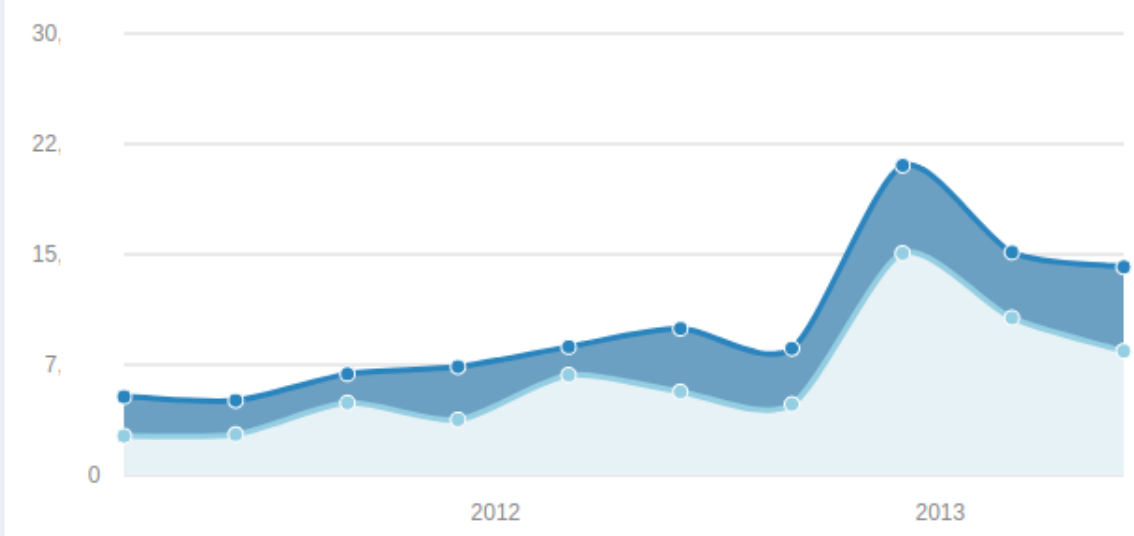


FIGURE 4.11 – Nobre de forages Aïn Beida en comparaison avec El Borma

4.5.2 Système Cartographique pour les Forages

On utilise GeoDjango pour aider l'utilisateur positionner les forages, en visualisant la zone géographique et les forages existants. la structure des cartes est définie sous format json et les forages sont définis comme des coordonnées X,Y dans la base de données Postgres, cette cartographie permettera l'aide a la décision pour le positionnement des forages.

Chaque forage a un Rayon d'influence, c'est le rayon duquel le forage peut extraire de l'eau, les rayons d'influence de deux forages ne doivent pas intersecter sinon cela rendera l'eau inexploitable et consommera l'eau a une vitesse qui peut détruire la pompe.

```
{% load leaflet_tags %}
{% leaflet_css %}
{% leaflet_js %}

<div>
    <h1>{{ city.name }}</h1>
    {% leaflet_map "main" callback="map_init" %}
</div>

<script type="text/javascript">
    function map_init(map, options) {
        // get point lat and lon
        var lon = "{{ city.geometry.x }}";
        var lat = "{{ city.geometry.y }}";

        // zoom to point & add it to map
        map.setView([lat, lon], 12);
        L.marker([lat, lon]).addTo(map);
    }
</script>
```

FIGURE 4.12 – Récupérer les emplacements des forages et placer les marques

Dans notre carte on marque les forages par des cercles de 500m de diametre (Rayon d'influence) comme estimé par l'expert , dont le centre est l'endroit du forrage (X,Y).

L'utilisation du forage est marque comme Bleu pour les forages d'AEP (Aduction d'eau potable), et rouge pour les forages utilisés pour l'Irrigation. Violet signifie une utilisation Mixte (AEP + Irrigation)

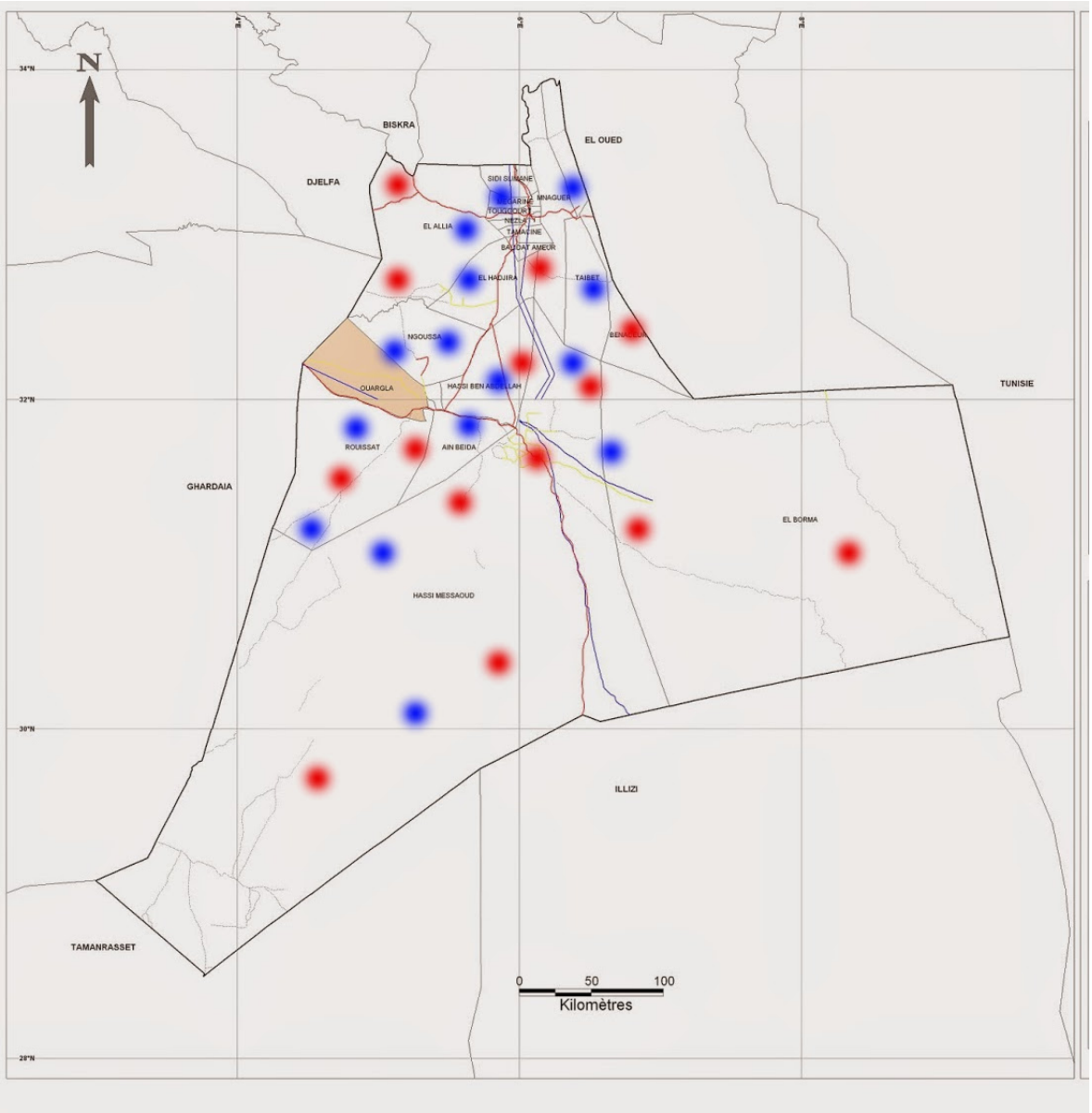


FIGURE 4.13 – Forages de Ouargla

4.6 Déploiement

Chaque outil était développé et testé Indépendemment suivant la philosophie de Django, Toutes les applications doivent être "pluggable", les applications réunis tous outils en un seul endroit, notre plate-forme BI, la figure 4.9 est sont interface.

Le principal but de cette plate-forme est de démontrer comment notre approche d'entrepôt de données extensible peut être utilisée dans un système d'aide a la décision, spécifiquement dans le domaine d'hydraulique.

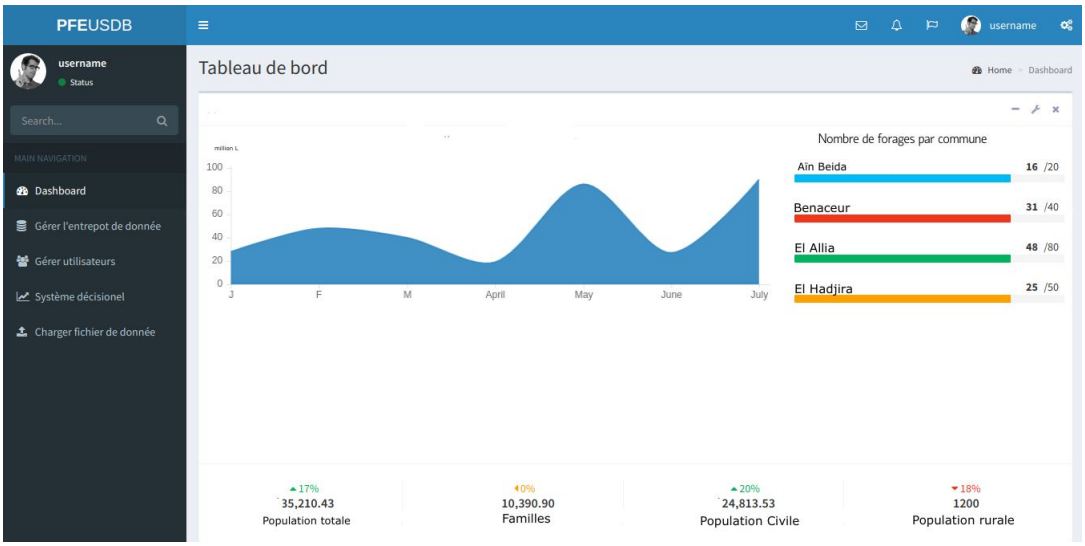


FIGURE 4.14 – L'écran d'accueil dans la plate-forme

4.6.1 Menu latéral

Les différentes utilisations de notre Plateforme sont affichés dans un menu latéral, ce menu ouvre de différentes applications Django que Nous avons développé, les utilisateurs peuvent avoir ou pas l'accès a ces onglets selon leurs rôle.

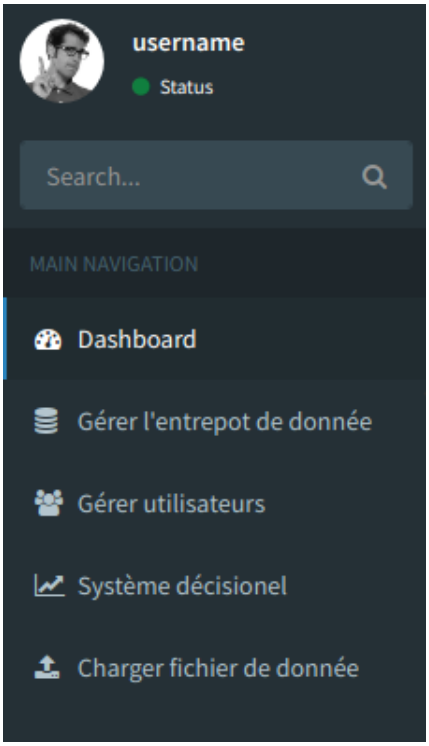


FIGURE 4.15 – Le menu latéral

4.6.2 Menu d'entête

le menu d'entête affiche les messages reçus dans la plate forme, et les notifications, qui sont des messages déclenchés par certains KPIs d'alerte.

Exemple : Niveau Dynamique = Niveau Statique

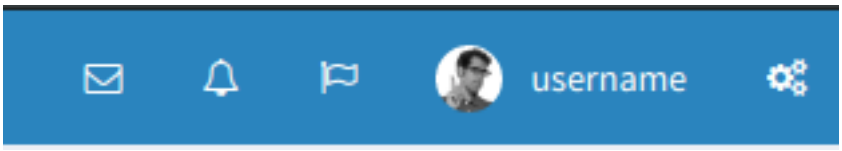


FIGURE 4.16 – Le menu d'entête

4.6.3 Intégration des applications

Comme indiqué dans la partie conception, nous utilisons le modèle MVT, Model view template, la figure 4.13 indique la structure des fichiers dans notre projet utilisant ce modèle avec Django

Merci a Django, on peut définir des fonctions dynamique(figure 4.14) personnalisant les schémas et leurs couleurs dans les templates HTML, ceci nous sert pour modifier la couleur d'une charte dans le cas qu'un KPI atteins un niveau d'alerte

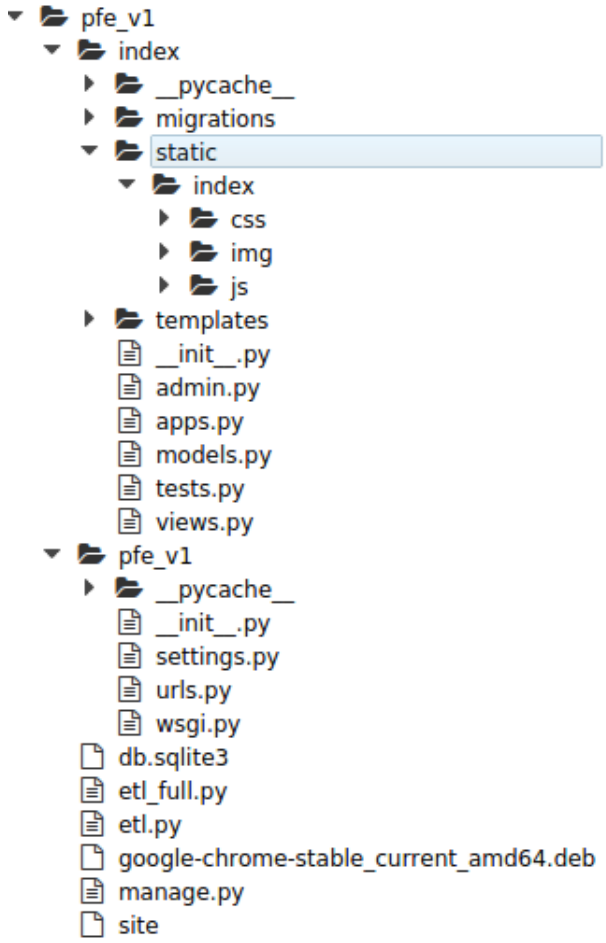


FIGURE 4.17 – La structure des fichiers du projet

```
<script>
{% block jquery %}
var endpoint = '/api/chart/data/'
var defaultData = []
var labels = [];
$.ajax({
    method: "GET",
    url: endpoint,
    success: function(data){
        labels = data.labels
        defaultData = data.default
        setChart()
    },
    error: function(error_data){
        console.log("error")
        console.log(error_data)
    }
})
```

FIGURE 4.18 – La configuration de Jquery

```
function setChart(){
    var ctx = document.getElementById("myChart");
    var ctx2 = document.getElementById("myChart2");
    var myChart = new Chart(ctx2, {
    type: 'bar',
    data: {
        labels: labels,
        datasets: [{
            label: 'Nombre de Forages',
            data: defaultData,
            backgroundColor: [
                'rgba(255, 99, 132, 0.2)',
                'rgba(54, 162, 235, 0.2)',
                'rgba(255, 206, 86, 0.2)',
                'rgba(75, 192, 192, 0.2)',
                'rgba(153, 102, 255, 0.2)',
                'rgba(255, 159, 64, 0.2)'
            ],
            borderColor: [
                'rgba(255,99,132,1)',
                'rgba(54, 162, 235, 1)',
                'rgba(255, 206, 86, 1)',
                'rgba(75, 192, 192, 1)',
                'rgba(153, 102, 255, 1)',
                'rgba(255, 159, 64, 1)'
            ],
```

FIGURE 4.19 – changement des couleurs de cartes dans HTML via django

CONCLUSION

Ce mémoire a été réalisé dans le cadre d'une convention entre l'université de Saad dahleb Blida - Blida 1- et le Centre de développement des technologies avancées - CDTA - Le but du projet étant de développer un système d'aide à la décision extensible dans l'investissement économique en Algérie.

Ainsi nous avons proposé une approche d'évolution de l'entrepôt de données qui prends en compte aussi le rafraîchissement ETL, cette approche consiste a concevoir le processus ETL en tant que schéma de dépendances et le traduire vers des fonctions qui peuvent être personnalisés facilement et qui permettent de gérer les problèmes facilement, cette approche est implémentée en utilisant directement les langages de programmation, en utilisant les bonnes pratiques de développement logiciel, Aussi l'extensibilité est dans le sens ou on utilise les mécanismes de vues matérialisés pour gérer l'extensibilité entre plusieurs entrepôts créant un "entrepôt virtuel" qui consolide toutes les données nécessaires. Les contributions majeurs de notre travail sont :

- Étude sur l'état de l'art sur l'évolution des entrepôts de données (établir des critères de comparaison, synthèse et analyse des travaux),
- Proposition d'une approche de conception d'un entrepôt évolutive.
- Étude et sélection des KPIs à introduire dans le système décisionnel.
- Conception du système décisionnel extensible.
- Étude comparative entre plusieurs paradigmes et outils d'implémentation de systèmes BI/DW
- Implémentation et application sur un cas d'étude relatif au domaine hydraulique (positionnement des forages suivant plusieurs critères à savoir : le type de nappes d'eau, débit du forage, type d'utilisation du forage, ...) et d'un Tableau de bord BI pour aide a la décision. sous forme d'une plate-forme en Django(python).

Notre approche permet une meilleur d'évolution de l'entrepôt a grande échelle, et evitera une grande quantité de problemes de complexité. Elle nécessite toujours le suivi d'un expert, pour faire un jugement sur la pertinence des modifications effectués et l'intégration de ces derniers dans les applications BI.

Notre travail evoque pas mal de perspectives trés intéressantes, D'abord, on veut élaborer un système de gestion de roles complet pour cette plateforme, ainsi que développer les autres cas d'utilisation secondaires.

Avec l'extention de l'entrepôt, le nombre de KPIs à prendre en considération simultanément deviens trés grand, et donc la capacité du décideur à utiliser le système pour faire les bonnes décisions est réduite.

Aussi, quelques décisions économiques nécessitent des données qui sont confidentiels, et qui ne peuvent pas être transmises a l'administrateur du système (comme le domaine de la santé). On veut se concentrer sur l'implémentation de l'intelligence ar pour l'aide a la décision dans notre système, on veut créer des applications IA qui font l'apprentissage continu sur les sources données (même confidentiels) sans les divulger. Pour enfin proposer des décisions a l'utilisateur. ce système résoudra aussi le problème de la croissance de nombre de KPIs a prendre en considération et le probleme de confidentialité au meme temps.

On propose aussi en perspectives de viser l'extensibilité physique, et adapter notre approche a un système distribué.

Pour l'extensibilité des vues, on peut l'améliorer en utilisant les solutions propsés dans [55]. i.e inclure des tâches d'Adaptation de vues automatique, Selection des vues a matérialiser automatiquement et enfin Maintenance des vues automatique.

Bibliographie

[1] W. Lehner, "Modeling large scale olap scenarios," in *International Conference on Extending Database Technology*. Springer, 1998, pp. 151–167.

[2] X. Zhang and E. A. Rundensteiner, "Dyda : Dynamic data warehouse maintenance in a fully concurrent environment," in *International Conference on Data Warehousing and Knowledge Discovery*. Springer, 2000, pp. 94–103.

[3] Z. Bellahsene, "Schema evolution in data warehouses," *Knowledge and Information Systems*, vol. 4, no. 3, pp. 283–304, 2002.

[4] S. Taktak, J. Feki, and G. Zurfluh, "Modélisation des transformations pour l'évolution de modèles multidimensionnels," 2016.

[5] E.-I. B. Guerrero, "Infrastructure adaptable pour les entrepots de données," Ph.D. dissertation, Université Joseph-Fourier-Grenoble I, 2002.

[6] C. Favre, "Évolution de schémas dans les entrepôts de données : mise à jour de hiérarchies de dimension pour la personnalisation des analyses," Ph.D. dissertation, Université Lumière-Lyon II, 2007.

[7] D. Solodovnikova, "Data warehouse evolution framework." in *SYRCoDIS*, 2007.

[8] C. Favre, F. Bentayeb, and O. Boussaid, "Evolution et personnalisation des analyses dans les entrepôts de données-une approche orientée utilisateur." in *INFORSID*, vol. 7, 2007, pp. 308–323.

[9] ——, "A survey of data warehouse model evolution," in *Handbook of Research on Innovations in Database Technologies and Applications : Current and Future Trends*. IGI Global, 2009, pp. 129–136.

[10] L. J. Zhou, H. J. Geng, and M. S. Xu, "Materialized view selection in the data warehouse," in *Applied Mechanics and Materials*, vol. 29. Trans Tech Publ, 2010, pp. 1133–1138.

[11] A. Gupta, F. Yang, J. Govig, A. Kirsch, K. Chan, K. Lai, S. Wu, S. G. Dhoot, A. R. Kumar, A. Agiwal *et al.*, "Mesa : Geo-replicated, near real-time, scalable data warehousing," *Proceedings of the VLDB Endowment*, vol. 7, no. 12, pp. 1259–1270, 2014.

[12] M. Golfarelli and S. Rizzi, “From star schemas to big data : 20

+

years of data warehouse research,” in *A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years*. Springer, 2018, pp. 93–107.

[13] A. LARBI *et al.*, “Conception des entrepôts de données : Evaluation des besoins flexibles,” Ph.D. dissertation, 2018.

[14] M. Houcine, “Vers un nouveau modèle de stockage et d‘accès aux données dans les big data et les cloud computing,” Ph.D. dissertation, 2018.

[15] M. Besbes, B. Abdous, B. Abidi, A. Ayed, M. Bachta, M. Babasy, B. B. Baccar, D. El Batti, Y. B. Salah, M. B. Charreton *et al.*, “Système aquifère du sahara septentrional gestion commune d’un bassin transfrontière,” *La Houille Blanche*, no. 5, pp. 128–133, 2003.

[16] R. Kimball and M. Ross, *The data warehouse toolkit : the complete guide to dimensional modeling*. John Wiley & Sons, 2011.

[17] “How can i use bibtex to cite a web page ? - tex - latex stack exchange,” https://tex.stackexchange.com/questions/3587/how-can-i-use-bibtex-to-cite-a-web-page, (Accessed on 07/08/2019).

[18] “Prévisions météo pour ouargla,” https://fr.weather-forecast.com/locations/Ouargla/forecasts/latest, (Accessed on 07/08/2019).

[19] “Model–view–controller - wikipedia,” https://en.wikipedia.org/wiki/Model-view-controller, (Accessed on 07/08/2019).

[20] “Django’s structure – a heretic’s eye view - python django tutorials,” https://djangobook.com/mdj2-django-structure/, (Accessed on 07/08/2019).

[21] S. Marche, “Measuring the stability of data models,” *European Journal of Information Systems*, vol. 2, no. 1, pp. 37–47, 1993.

[22] D. Sjøberg, “Quantifying schema evolution,” *Information and Software Technology*, vol. 35, no. 1, pp. 35–44, 1993.

[23] N. Dedić and C. Stanier, “An evaluation of the challenges of multilingualism in data warehouse development,” 2016.

[24] R. Gaardboe and T. Svarre, “Business intelligence success factors : A literature,” *Journal of Information Technology Management*, vol. 29, no. 1, p. 1, 2018.

[25] F. Coker, *Pulse : Understanding the vital signs of your business*. BookBaby, 2015.

[26] S. Grandhi, R. Chugh *et al.*, “The value of business intelligence tools : Aligning business intelligence governance with corporate governance,” 2013.

[27] B. Golden, *Amazon web services for dummies*. John Wiley & Sons, 2013.

[28] S. Hammoudi, L. A. Maciaszek, M. M. Missikoff, O. Camp, and J. Cordeiro, *Enterprise Information Systems : 18th International Conference, ICEIS 2016, Rome, Italy, April 25–28, 2016, Revised Selected Papers*. Springer, 2017, vol. 291.

[29] "Ijca - optimization of data warehousing system : Simplification in reporting and analysis," https://www.ijcaonline.org/proceedings/icwet/number9/2131-db195, (Accessed on 07/08/2019).

[30] B. Inmon, "Dw 2.0 ; architecture for the next generation of data warehousing," *Information Management*, vol. 16, no. 4, p. 8, 2006.

[31] "What is etl ? | sas," https://www.sas.com/en_us/insights/data-management/what-is-etl.html, (Accessed on 07/08/2019).

[32] M. J. Denney, D. M. Long, M. G. Armistead, J. L. Anderson, and B. N. Conway, "Validating the extract, transform, load process used to populate a large clinical research database," *International journal of medical informatics*, vol. 94, pp. 271–274, 2016.

[33] S. Zhao, "What is etl ? (extract, transform, load) | experian," https://www.edq.com/blog/what-is-etl-extract-transform-load/, 2017, (Accessed on 07/08/2019).

[34] S. B. Kimball, *Heber C. Kimball : Mormon Patriarch and Pioneer*. University of Illinois Press, 1986.

[35] C. T. Fitz-Gibbon, *Performance indicators*. Multilingual Matters, 1990, vol. 2.

[36] D. Parmenter, *Key performance indicators : developing, implementing, and using winning KPIs*. John Wiley & Sons, 2015.

[37] A. Fernandez, *Les nouveaux tableaux de bord des managers : Le projet Business Intelligence clés en main*. Editions Eyrolles, 2013.

[38] A. J. Lee, A. Nica, and E. A. Rundensteiner, "The eve approach : View synchronization in dynamic distributed environments," *IEEE Transactions on knowledge and data engineering*, vol. 14, no. 5, pp. 931–954, 2002.

[39] E. A. Rundensteiner, A. Koeller, and X. Zhang, "Maintaining data warehouses over changing information sources," *Communications of the ACM*, vol. 43, no. 6, pp. 57–57, 2000.

[40] A. Gupta, I. S. Mumick, and K. A. Ross, *Adapting materialized views after redefinitions*. ACM, 1995, vol. 24, no. 2.

[41] D. Solodovnikova and L. Niedrite, "Data warehouse adaptation after the changes in source schemata," in *Proc. of the 7th Intl. Baltic Conference on Databases and Information Systems*, 2006, pp. 52–63.

[42] C. A. Hurtado, A. O. Mendelzon, and A. A. Vaisman, "Maintaining data cubes under dimension updates," in *Proceedings 15th International Conference on Data Engineering (Cat. No. 99CB36337)*. IEEE, 1999, pp. 346–355.

[43] M. Blaschka, C. Sapia, and G. Höfling, "On schema evolution in multidimensional databases," in *International Conference on Data Warehousing and Knowledge Discovery*. Springer, 1999, pp. 153–164.

[44] J.-N. Mazón, J. Trujillo, and J. Lechtenbörger, "A set of qvt relations to assure the correctness of data warehouses by using multidimensional normal forms," in *International Conference on Conceptual Modeling*. Springer, 2006, pp. 385–398.

[45] C. Cunningham, I.-Y. Song, and P. P. Chen, "Data warehouse design to support customer relationship management analysis," *Journal of Database Management (JDM)*, vol. 17, no. 2, pp. 62–84, 2006.

[46] R. Bliujute, S. Saltenis, G. Slivinskas, and G. Jensen, "Systematic change management in dimensional data warehousing," Citeseer, Tech. Rep., 1998.

[47] A. O. Mendelzon and A. A. Vaisman, "Temporal queries in olap," in *Proceedings of the 26th International Conference on Very Large Data Bases*. Morgan Kaufmann Publishers Inc., 2000, pp. 242–253.

[48] J. Eder, C. Koncilia, and T. Morzy, "The comet metamodel for temporal data warehouses," in *International Conference on Advanced Information Systems Engineering*. Springer, 2002, pp. 83–99.

[49] E. Annoni, F. Ravat, O. Teste, and G. Zurfluh, "Modélisation intégrée de la dynamique des systèmes d'information décisionnels," in *EDA*, 2008.

[50] H. Fu and B. Fu, *SAP BW : a step-by-step guide*. Addison-Wesley Professional, 2002.

[51] P. Bosc and O. Pivert, "Sqlf : a relational database language for fuzzy querying," *IEEE transactions on Fuzzy Systems*, vol. 3, no. 1, pp. 1–17, 1995.

[52] R. Kimball, *The data warehouse toolkit : practical techniques for building dimensional data warehouses*. John Wiley & Sons, Inc., 1996.

[53] R. Kimball, L. Reeves, W. Thornthwaite, M. Ross, and W. Thornwaite, *The Data Warehouse Lifecycle Toolkit : Expert Methods for Designing, Developing and Deploying Data Warehouses with CD Rom*, 1st ed. New York, NY, USA : John Wiley & Sons, Inc., 1998.

[54] "Data warehousing with materialized views," https://www.csee.umbc.edu/portal/help/oracle8/server.815/a67775/ch1.htm, (Accessed on 07/09/2019).

[55] G. Thakur and A. Gosain, "A comprehensive analysis of materialized views in a data warehouse environment," *IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 2, no. 5, 2011.

[56] D. Sahpaski, G. Velinov, B. Jakimovski, and M. Kon-Popovska, "Dynamic evolution and improvement of data warehouse design," in *2009 Fourth Balkan Conference in Informatics*. IEEE, 2009, pp. 107–112.

[57] B. Bebel, J. Eder, C. Koncilia, T. Morzy, and R. Wrembel, "Creation and management of versions in multiversion data warehouse," in *Proceedings of the 2004 ACM symposium on Applied computing*. ACM, 2004, pp. 717–723.

[58] B. Bębel, Z. Krolikowski, and R. Wrembel, "Formal approach to modelling a multiversion data warehouse," *Bulletin of the Polish Academy of Sciences : Technical Sciences*, 2006.

[59] T. Bakusevych, "Rules for better dashboard design," 2017.

[60] E. Jacobson and W. Ostwald, *Color harmony manual*. Container Corporation of America Chicago, 1948.

[61] J. M. Lucassen and S. H. Maes, "Mvc (model-view-controller) based multi-modal authoring tool and development environment," Feb. 7 2006, uS Patent 6,996,800.

[62] "code.talks commerce special 2018," https://now.spryker.com/code.talks-commerce-special-2018, (Accessed on 07/09/2019).

[63] S. McConnell, *Code complete*. Pearson Education, 2004.

[64] A. Zeller, *Why programs fail : a guide to systematic debugging*. Elsevier, 2009.

[65] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Social coding in github : transparency and collaboration in an open software repository," in *Proceedings of the ACM 2012 conference on computer supported cooperative work*. ACM, 2012, pp. 1277–1286.

[66] "pandas.dataframe — pandas 0.24.2 documentation," https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html, (Accessed on 07/09/2019).