



PROJET CLOUD HEALTHCARE UNIT

Livrable 1

Résumé

Cette documentation consiste en l'étude de l'architecture Big Data à mettre en place, dans le cadre d'une transformation digitale majeure initiée par le groupe CHU, ayant pour objectif d'exploiter la quantité considérable de données générées par les systèmes de gestion de soins



Table des matières

I)	Contexte.....	3
II)	Schéma de l'architecture Big Data mise en place.....	4
	Les données sources.....	4
	Le chargement des données	5
	La transformation des données	5
	L'exploitation des données	5
III)	Le modèle de données.....	6
IV)	Les jobs d'alimentation.....	7
	Dimension Patient.....	7
	Dimension Professionnel	7
	Dimension Diagnostic	9
	Dimension Localisation.....	10
	Dimension Temps.....	11
	Faits.....	13
V)	Hiérarchie du data Lake.....	17
VI)	Variables de contexte	19
VII)	Conclusion.....	21



Table des Illustrations

Figure 1 Modèle Conceptuel de Données en étoile du Projet CHU	6
Figure 2 Job d'alimentation de la table Patient.....	7
Figure 3 Requête d'extraction des données de Patient	7
Figure 4 Job d'alimentation de la dimension professionnel.....	8
Figure 5 Requête de récupération des professionnels	8
Figure 6 Mapping des professionnels du fichier délimité.....	8
Figure 7 Mapping des établissements de santé	9
Figure 8 Jointure entre les établissements et les id de professionnels	9
Figure 9 Jointure entre les établissements de santé et les professionnels.....	9
Figure 10 Job d'alimentation des diagnostics.....	10
Figure 11 Requête des récupérations des diagnostics.....	10
Figure 12 Mapping des diagnostics.....	10
Figure 13 Mapping final des diagnostics	10
Figure 14 Job d'alimentation de la dimension Localisation	11
Figure 15 Job d'alimentation de la dimension Temps	12
Figure 16 Requête de sélection des dates de consultations.....	12
Figure 17 Mapping de changement de format des dates	13
Figure 18 Mapping de l'atomisation des dates.....	13
Figure 19 Ensemble des dossiers stockant les dimensions dans le Data Lake.....	17
Figure 20 L'arborescence des dossiers stockant la dimension Patient dans le Data Lake ..	17
Figure 21 L'arborescence des dossiers stockant la dimension Temps dans le Data Lake....	18
Figure 22 L'arborescence des dossiers stockant la dimension Localisation dans le Data Lake	18
Figure 23 Création du contexte de connexion à la BDD PostgreSQL.....	19
Figure 24 Ensemble des variables de contexte concernant la BDD PostgreSQL	19
Figure 25 Ensemble des variables de contexte concernant le cluster HDFS	20
Figure 26 Schémas de données des localisations.....	20



I) Contexte

Le Groupe CHU (Cloud Healthcare Unit) a reconnu l'intérêt et la nécessité d'une transformation numérique majeure. Il a été demandé à notre département de l'aider à construire son propre entrepôt de données, ce qui permettrait au groupe de tirer parti des grandes quantités de données générées par le système de gestion des soins.

L'objectif est de pouvoir répondre à une variété de besoins d'accès et d'analyse des utilisateurs. À cette fin, le Groupe CHU attend :

- Une solution complète de modèles, d'outils et d'architecture pour extraire et stocker des données, qui peuvent ensuite être explorées et visualisées selon différents critères.
- Une solution pour intégrer les données de fichiers distribués dans une seule source persistante.
- Identifier les besoins des utilisateurs (praticiens, responsables d'établissements) en analyse de données pour une utilisation directe dans le suivi des patients au niveau national et à long terme
- Des conseils sur les outils d'intégration, les logiciels de stockage et de visualisation adaptés, et l'exploration de données entièrement sécurisée pour faciliter une meilleure prise de décision.

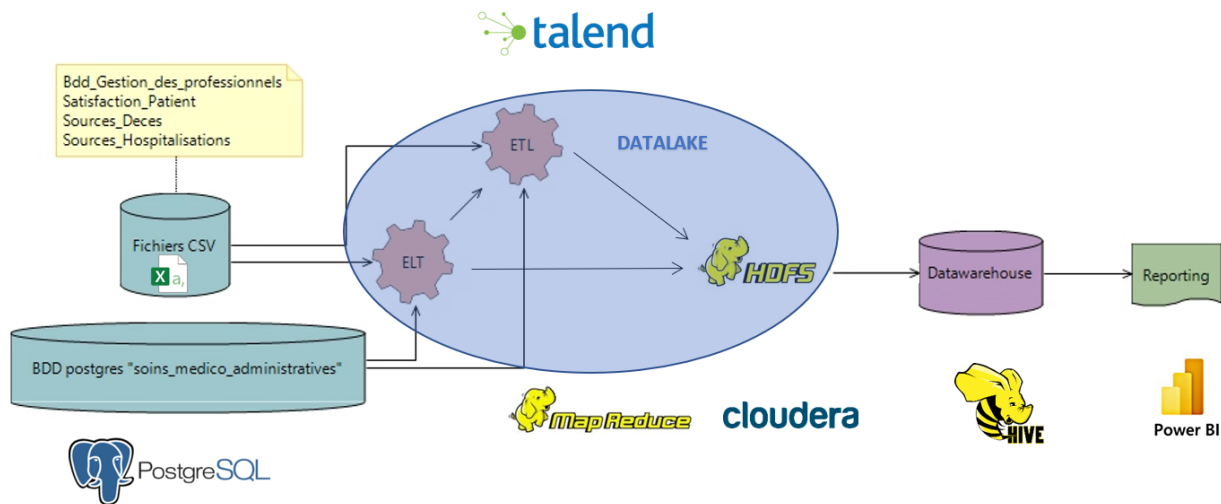
Cette étude tente de relever tous ces défis dans le secteur de la santé. Par conséquent, des solutions doivent être développées pour répondre aux exigences d'infrastructure des données médicales, telles que le rapport coût/efficacité, la sécurité, la résilience et l'évolutivité.

Les grandes phases envisagées du projet sont les suivantes :

- Etude de l'architecture à mettre en place
- Modélisation conceptuelle des données
- Implémentation physique des données
- Exploitation, mesure de performance, optimisations

Ce premier livrable comprend l'étude de l'architecture à mettre en place ainsi que la modélisation conceptuelle des données.

II) Schéma de l'architecture Big Data mise en place



Les données sources

Il s'agit de données provenant d'historiques des systèmes de gestion des soins, des fichiers de satisfaction et de décès. Ils contiennent des données d'exploitation sur plusieurs années.

Voici un descriptif des sources de données :

- Une BDD PostgreSQL qui gère les soins-medico-administratives des patients, contenant notamment :
 - Une liste des patients des établissements de santé français,
 - Une liste des consultations prodigués à cette liste de patients,
 - Les diagnostics réalisés sur ces patients lors des consultations,
 - Une liste des professionnels de santé en France.
- Des fichiers CSV :
 - Des notes de satisfactions émises par des patients sur différents établissements de santé, provenant de différents secteurs professionnels du médical, de l'année 2015 à 2020,
 - L'ensemble des décès en France recensés par l'INSEE de 2010 à 2021,
 - Des informations sur les établissements de santé de France,
 - La liste des hospitalisations en France de 2015 à 2021,
 - Des informations sur l'activité et le statut des professionnels de santé en France.



Le chargement des données

La première étape consiste en la récupération des données depuis leurs différentes sources, pour les stocker de manière unifiée sur *HDFS* (Hadoop Distributed File System) et en sélectionnant les champs appropriés. Cela passe par la mise en place de plusieurs jobs d'alimentation, en utilisant des outils comme *Talend* pour la création des jobs, *mapReduce* pour le traitement, *Cloudera* pour le stockage des données sortantes.

La transformation des données

Une fois les données de la source chargées dans l'emplacement choisi, cette étape consiste à les nettoyer, les combiner en vue de les préparer pour un modèle d'utilisation spécifique.

Les données sont alors stockées dans un entrepôt de données, à l'aide de l'outil *Hive*.

L'exploitation des données

La dernière étape est celle de l'exploitation et l'analyse des données transformées, afin de pouvoir répondre à la variété des besoins et exigences d'accès et d'analyses des utilisateurs. *Microsoft Power BI* est une solution adaptée pour cette étape.

III) Le modèle de données

Voici un rappel des besoins finaux :

- Taux de **consultation** des patients dans un **établissement** X sur une **période** Y
- Taux de **consultation** des patients par rapport à un **diagnostic** X sur une **période** Y
- Taux global **d'hospitalisation** des **patients** dans une **période** donnée Y
- Taux **d'hospitalisation** des **patients** par rapport à des **diagnostics** sur une **période** donnée
- Taux **d'hospitalisation/consultation** par sexe, par âge
- Taux de **consultation** par **professionnel**
- Nombre de **décès** par **localisation** (région) et sur **l'année** 2019
- Taux global de **satisfaction** par **région** sur **l'année** 2020

D'après les différents éléments mis en évidence ci-dessus, nous avons choisi de mettre en place 5 dimensions, pour permettre une analyse structurée et obtenir des mesures (faits) cohérentes :

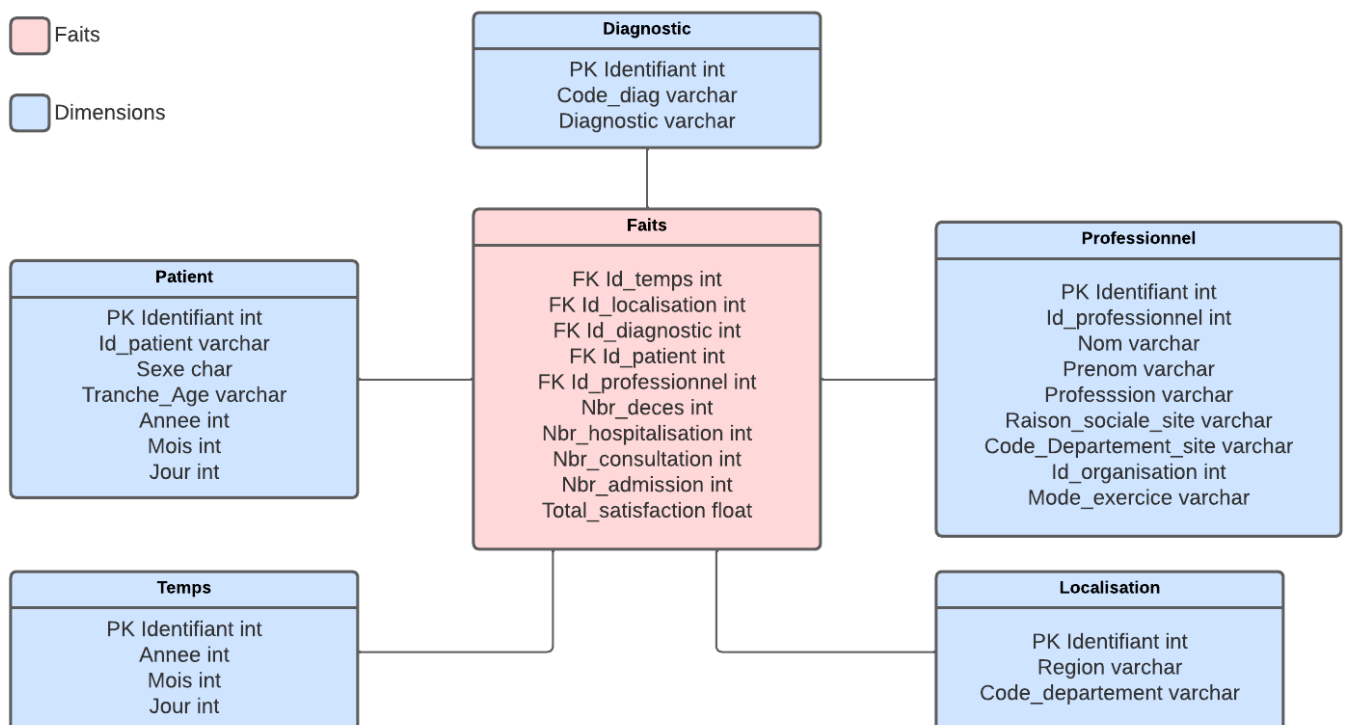


Figure 1 Modèle Conceptuel de Données en étoile du Projet CHU

IV) Les jobs d'alimentation

Dimension Patient

Vis-à-vis de l'alimentation de la table Patient, l'objectif est de réunir l'ensemble des champs nécessaires au suivi d'un patient à partir de la table Patient de la base de données PostgreSQL. Voici ci-dessous le job développé sous Talend.

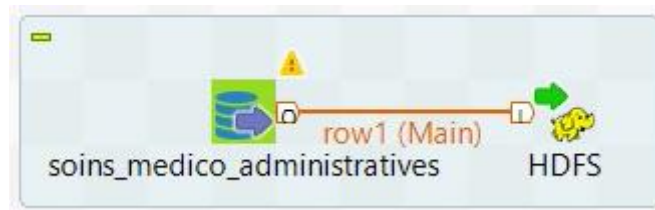


Figure 2 Job d'alimentation de la table Patient

La requête nous permet d'obtenir directement les champs nécessaires. A savoir, la génération d'une clé primaire, l'identifiant client originel qui permettra de joindre les consultations, le sexe du patient sous la forme 'F' ou 'M', la tranche d'âge du patient sous la forme d'un entier définissant la dizaine (0 -> 0-9 ans, 1 -> 10-19 ans, etc.) et finalement l'ensemble de la date de naissance du patient atomisé. La date de naissance nous permettra de calculer l'âge du patient au moment des consultations, des décès ou des hospitalisations par exemple. Afin de comprendre concrètement la requête, la voici ci-dessous.

```
SELECT
ROW_NUMBER() OVER (ORDER BY "soins_medico_administratives"."public"."Patient"."Id_patient") AS Id,
"soins_medico_administratives"."public"."Patient"."Id_patient",
UPPER(SUBSTRING("soins_medico_administratives"."public"."Patient"."Sexe" FROM 1 FOR 1)) AS Sexe_patient,
TRUNC((EXTRACT(year FROM CURRENT_DATE) - EXTRACT(year FROM TO_DATE("soins_medico_administratives"."public"."Patient"."Date", 'MM/DD/YYYY')))/ 10, 0) AS Tranche_age,
EXTRACT(day FROM TO_DATE("soins_medico_administratives"."public"."Patient"."Date", 'MM/DD/YYYY')) AS Jour_naissance_patient,
EXTRACT(month FROM TO_DATE("soins_medico_administratives"."public"."Patient"."Date", 'MM/DD/YYYY')) AS Mois_naissance_patient,
EXTRACT(year FROM TO_DATE("soins_medico_administratives"."public"."Patient"."Date", 'MM/DD/YYYY')) AS Annee_naissance_patient
FROM "soins_medico_administratives"."public"."Patient"
```

Figure 3 Requête d'extraction des données de Patient

Cette dimension sera alimentée tous les mois, via cette tâche ELT, afin d'obtenir un bon équilibre entre données à jour et puissance de calcul d'alimentations.

Dimension Professionnel

Pour ce qui est des professionnels, nous nous baserons sur les tables *Professionnel_de_sante* de la BDD *PostGreSql* et les fichiers délimités *activite_professionnel_sante*, *etablissement_sante* et *professionnel_sante*. Cette combinaison de fichier nous permet de récupérer tous les champs pertinents liés à un professionnel à savoir, une clé primaire, un Id de professionnel, un Nom, un Prénom, un Id d'organisation, un libelle d'organisation, un code départemental d'organisation, une profession et un mode d'exercice. Voici le job associé :

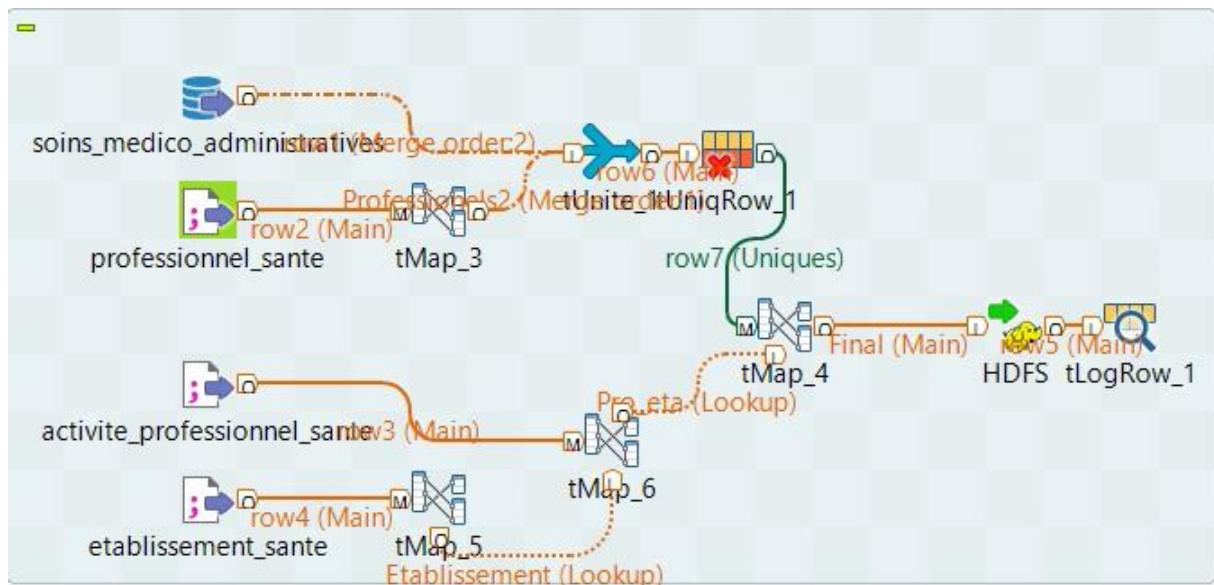


Figure 4 Job d'alimentation de la dimension professionnel

On commence donc par la récupération et la formalisation de l'ensemble des sources de données. Pour celles de la BDD elles sont récupérées à l'aide d'une requête PG-SQL.

SELECT

```
"soins_medico_administratives"."public"."Professionnel_de_sante"."Identifiant",
"soins_medico_administratives"."public"."Professionnel_de_sante"."Profession",
"soins_medico_administratives"."public"."Professionnel_de_sante"."Nom",
"soins_medico_administratives"."public"."Professionnel_de_sante"."Prenom"
FROM "soins_medico_administratives"."public"."Professionnel_de_sante"
```

Figure 5 Requête de récupération des professionnels

Ensuite on réalise une union avec le composant *tUnite* entre les professionnels de santé de *PostGreSQL* et ceux du fichier délimité après avoir mappé l'ensemble des données du fichier afin de faciliter l'union. On supprime ensuite les doublons avec *tUniqRow*.



Figure 6 Mapping des professionnels du fichier délimité

En parallèle on réalise la jointure entre la table *activite_professionnel_sante* et *etablissement_sante* afin de pouvoir lier les établissements à tous les professionnels par la suite. Pour réaliser cette jointure on utilise des composant *tMap* afin de reformuler les données des établissements mais aussi pour réaliser la jointure en elle-même sur le champ *id_organisation*.

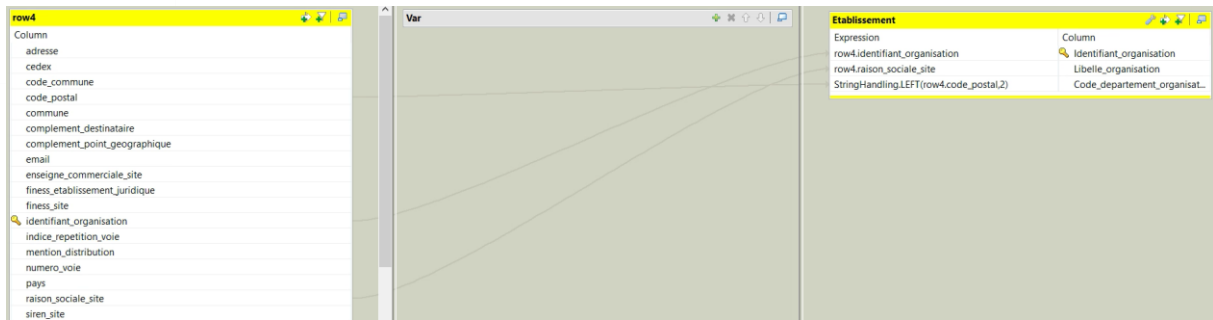


Figure 7 Mapping des établissements de santé



Figure 8 Jointure entre les établissements et les id de professionnels

Finalement, on réalise la jointure finale toujours à l'aide d'un composant *tMap*. On profite aussi de ce *tMap* pour insérer une clé primaire et filtrer les champs afin de ne conserver que les données pertinentes. Voici la jointure finale :

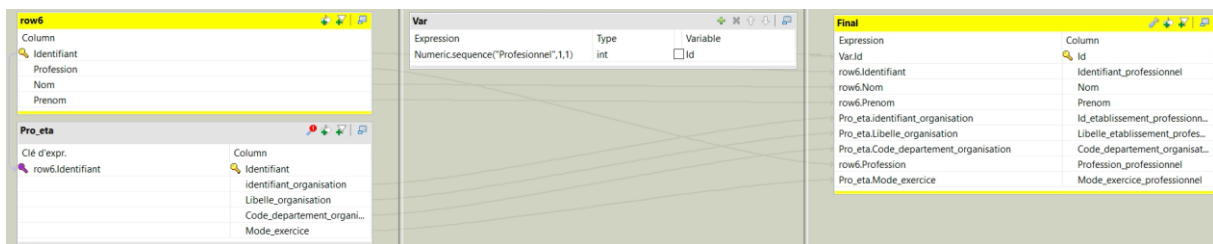


Figure 9 Jointure entre les établissements de santé et les professionnels

Cette dimension sera alimentée tous les mois, via cette tâche ELT, afin d'obtenir un bon équilibre entre données à jour et puissance de calcul d'alimentation.

Dimension Diagnostic

Le but cette dimension est de regrouper l'ensemble des diagnostics établis présents dans les données sources. Pour ce faire, nous nous appuyons sur les tables *diagnostic* de la BDD et le fichier délimité *hospitalisations*. L'objectif est d'obtenir chaque diagnostic avec un code diagnostic et son libellé.



Figure 10 Job d'alimentation des diagnostics

On commence donc par récupérer l'ensemble des données et par en formaliser certaines afin de faciliter les opérations futures. Voici la requête PG_SQL ainsi que le mapping associé à la récupération des données.

```
SELECT
  "soins_medico_administratives"."public"."Diagnostic"."Code_diag",
  "soins_medico_administratives"."public"."Diagnostic"."Diagnostic"
FROM "soins_medico_administratives"."public"."Diagnostic"
```

Figure 11 Requête des récupérations des diagnostics



Figure 12 Mapping des diagnostics

Ces deux opérations nous permettent de réaliser une union de l'ensemble des diagnostics présents dans les données initiales. Ensuite, les données sont regroupées par union, puis filtrées afin de supprimer les doublons et finalement incorporées au système *HDFS* en passant au travers *tMap* qui permet l'ajout d'une clé primaire.

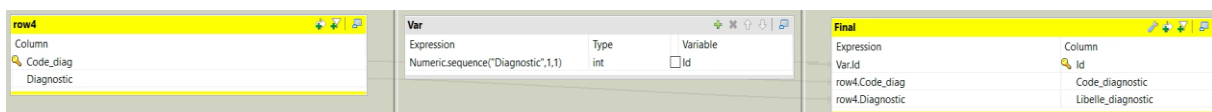


Figure 13 Mapping final des diagnostics

Cette dimension sera alimentée tous les mois, via cette tâche ELT, afin d'obtenir un bon équilibre entre données à jour et puissance de calcul d'alimentation.

Dimension Localisation

Concernant la table localisation, l'objectif est de lister toutes les régions de France ainsi que leurs départements associés. Pour récupérer toutes les informations nécessaires, la source de données provient des fichiers csv qui listent les personnes décédées par années. Voici le schéma du job :

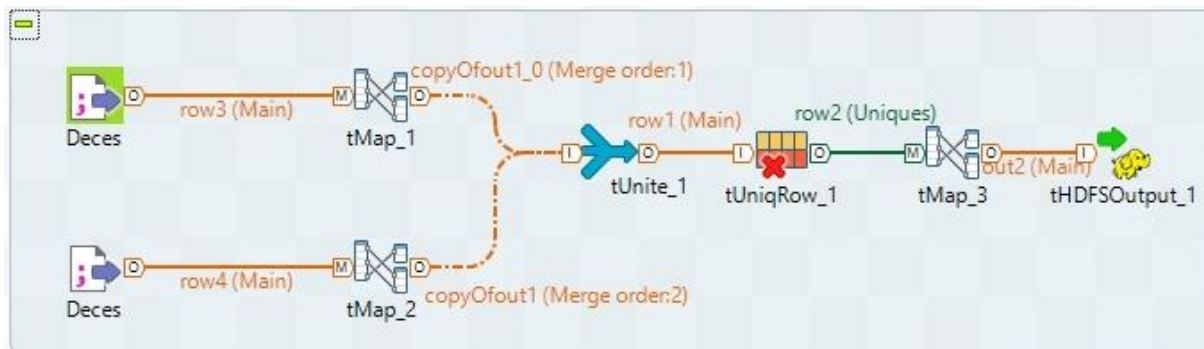


Figure 14 Job d'alimentation de la dimension Localisation

La première étape consiste en la segmentation du code postal des communes, pour ne récupérer que les 2 premiers chiffres, autrement dit le numéro du département. Le nom actuel de la région est aussi gardé, le reste n'est pas pris en compte. Cette étape est effectuée pour les communes de naissance et pour les communes de décès, dans l'optique d'avoir toutes les données nécessaires.

Cette dimension ne présente pas de fréquence d'alimentation identifiable et elle ne sera donc alimentée qu'une seule fois, jusqu'à ce qu'une modification soit nécessaire, via cette tâche ELT, dans le cadre où la composition des régions viendrait à changer comme ce fut le cas dans le cadre de la loi n°2015-29 du 16 janvier 2015.

Dimension Temps

Afin de collecter l'ensemble des dates utilisés au sein des données sources nous allons nous servir de l'ensemble des dates de décès présentes dans les fichiers décès, toutes les dates des consultations de *PostGreSq*/et finalement celles du fichier des hospitalisations. Ces dates nous permettront de couvrir l'ensemble des faits à enregistrer. Pour plus de précision voici le job développé :



Figure 15 Job d'alimentation de la dimension Temps

Pour ce qui est des décès on commence par filtrer l'ensemble des fichiers afin de ne conserver que les dates de décès. Ensuite on récupère les dates de consultations à l'aide de la requête ci-dessous.

```
SELECT
  "soins_medico_administratives"."public"."Consultation"."Date"
FROM "soins_medico_administratives"."public"."Consultation"
```

Figure 16 Requête de sélection des dates de consultations

Finalement, le format des dates d'hospitalisations est changé à l'aide du mapping ci-dessous afin de mieux correspondre aux autres données.

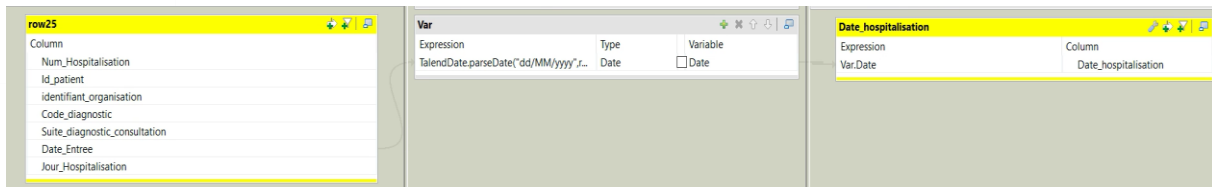


Figure 17 Mapping de changement de format des dates

L'ensemble des traitements en amont nous permet de faire une union de l'ensemble des dates. Ensuite, on utilise le composant *tUniqRow* afin de supprimer les doublons puis nous enchaînons avec un tri des dates afin de rendre la table plus lisible. Finalement nous réalisons un dernier mapping afin d'atomiser les dates avant leur mise en mémoire dans le *HDFS* et de créer une clé primaire.

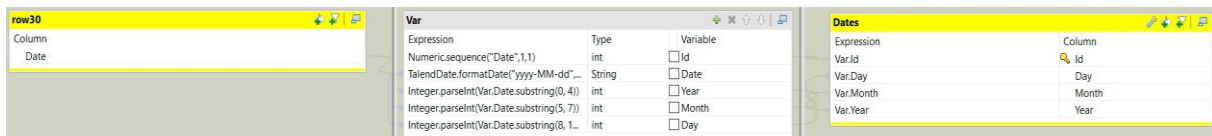


Figure 18 Mapping de l'atomisation des dates

Cette dimension ne présente pas de fréquence d'alimentation identifiable et elle ne sera donc alimentée qu'une seule fois, jusqu'à ce qu'une modification soit nécessaire, via cette tâche ELT, dans le cadre où nous voudrions analyser de nouvelles données, antérieures ou ultérieures au jeu de données actuel.

Faits

À la différence des dimensions, la table des faits n'a pas encore été formalisée sous Talend. Cependant, la version Talend n'est pas encore demandée lors de ce premier livrable. Nous allons donc dans cette partie décrire les étapes constituant le job qui nous permettra de peupler la table de l'ensemble de faits. Cette table sera constituée de plusieurs clés étrangères qui référeront aux différentes dimensions précédemment chargées. Ce mécanisme permet de fournir par la suite une table qui permet de réaliser les différentes mesures demandées. Ces mesures portent sur le nombre de décès, le nombre d'hospitalisations, le nombre de consultations et la satisfaction. Au sein de la table de fait, chaque ligne est unique et représente donc une clé primaire en soi. D'autre part chaque champ doit être rempli pour chacune des lignes, il est donc normal que certaines données soient dédoublées.

La table de faits est composée des champs, *id_patient*, *id_professionnel*, *id_diagnostic*, *id_localisation*, *id_temps*, *nombre_admission*, *nombre_hospitalisation*, *nombre_consultation*, *nombre_deces* et finalement *note_satisfaction*. L'ensemble des clés étrangères permettent de lier les dimensions à chaque fait. Ensuite, pour les champs utiles on retrouve :

- Nombre_admission, qui nous permettra de compter le nombre d'admissions en centre hospitalier. Il correspond en réalité au premier jour de chaque hospitalisation.
- Nombre_hospitalisation, qui correspond aux nombres de jours d'hospitalisations de chaque patient. On retrouvera donc une hospitalisation par jour passé à l'hôpital pour un patient. Il peut être supérieur à 1 si un patient a été plusieurs fois hospitalisé dans une même journée, par un même professionnel et à la suite d'un même diagnostic.
- Nombre_consultation, qui représentera le nombre de consultations d'un patient par un professionnel de santé pour une date et un diagnostic précis.
- Nombre_décès, sera exprimé par jours et par régions sur chacune des lignes correspondantes au travers d'un entier également.
- Note_satisfaction, permettra d'obtenir la note de satisfaction par région et par années. Nous retrouverons donc une note sur l'ensemble des lignes qui correspondra à une localisation et un temps.

L'ensemble de ces champs et l'ensemble de la structure nous permettront de calculer simplement l'ensemble des mesures à partir d'une table unique. Effectivement si on souhaite obtenir le nombre de morts par jours et par régions il suffit de grouper la table par dates et par régions afin d'obtenir ces deux métriques.

Pour ce qui est de la construction de la table de faits nous allons maintenant détailler le raisonnement qui nous permettra de réaliser l'ensemble de cette table.

1 Construction des mesures annexes


1.1 Groupement des décès par régions et jours

L'objectif de cette table est d'obtenir la somme des décès en France pour chaque jours et régions. Cette table sera ensuite jointe au reste par le biais de la date et de la localisation afin d'obtenir un compte des décès sur chaque ligne de la table des faits.

Pour la réaliser, on commence par filtrer les colonnes des tables de décès afin de conserver uniquement la date de décès, le code du département de décès et la région de décès si présente. Ensuite, on réalise l'union de toutes ces données avant de les grouper par régions et par jours afin d'en faire le compte. Ainsi on obtient une table qui comporte les champs, jour, mois, année, région et nombre_décès.

1.2 Groupement des satisfactions par établissements et par années

Après l'obtention de l'ensemble des décès, il est maintenant question d'obtenir une table similaire pour les satisfactions. Afin de l'obtenir, on commence par réaliser le calcul de la satisfaction moyenne de chaque établissement pour chaque année. Pour cela, il suffit de suivre la méthode de calcul de chaque année fournie en fichier PDF. Ensuite, on réalise l'union du résultat en ajoutant l'année afin d'obtenir une table qui comporte la satisfaction moyenne par région et par année. On obtient donc la table



satisfaction qui comprend les champs région, année et note que l'on pourra joindre par la suite.

2 Construction de la table principale

2.1 Construction des hospitalisations et des admissions

Maintenant que nos deux mesures annexes sont faites nous pouvons nous concentrer sur le principal à savoir les hospitalisations. Pour construire cette table, on se base sur le fichier *hospitalisations* qui cumule l'ensemble des informations nécessaires aux jointures des différentes dimensions.


1. On commence donc par extraire l'ensemble des données de cette table. Ensuite on vient joindre la dimension diagnostic à l'aide de *code_diagnostic* pour récupérer la première clé étrangère.
2. Par la suite on peut récupérer le patient à l'aide de l'*id_patient*.
3. On peut ensuite associer le temps avec la *Date_entree*.
4. Finalement on joint la dimension localisation en passant par la région de l'établissement de santé correspondant. Effectivement, on peut joindre les établissements de santé de la table *etablissement_sante* par le biais de leur identifiant.
5. On renseigne la valeur '1' pour le nombre d'admission lors du premier jour d'hospitalisation et on dédouble la ligne pour chaque jour d'hospitalisation en comptant à chaque fois une hospitalisation et zéro admission.
6. Finalement, on peut venir joindre le compte de décès avec les régions et les dates et les satisfactions avec les établissements et les années.
7. On termine par filtrer les champs ce qui nous donne une table correspondant aux attentes du modèle.

À l'aide de cette organisation, nous sommes capables de compter rapidement les hospitalisations ainsi que les admissions.

2.2 Construction des consultations

Pour réaliser cette table nous allons suivre le même raisonnement que pour celle des hospitalisations. Effectivement, l'objectif reste le même, à savoir glaner les différentes clés étrangères à intégrer à la table des faits tout en proposant une méthode de mesure rapide des différentes consultations.

- On se basera donc sur la table consultation présente au sein de la BDD. Après avoir extrait les données, nous commencerons par joindre la dimension Patient à l'aide de *id_patient*. Cette jointure nous donne la clé étrangère de la dimension patiente.
- On ajoute ensuite la dimension Professionnel de santé à l'aide de *id_prof_sante* présent sur chaque diagnostic. On récupère donc ici la clé du professionnel.
- On peut ensuite récupérer la dimension Diagnostic, à l'aide du *code_diagnostic*. On récupère donc la clé étrangère des diagnostics.

- 
- Ensuite, on joint la dimension Temps à l'aide du champ présent dans la table consultation de la BDD, à savoir *Date*.
 - On récupère la localisation du diagnostic à l'aide de la région de l'établissement de santé présent dans la dimension Professionnel de santé. On joint donc d'abord les établissements à l'aide de leur id puis les régions à l'aide de leur libellé. On obtient donc la dernière clé étrangère.
 - On entre le nombre de consultations sur chaque ligne consultation à savoir '1' et '0' sur les hospitalisations et les admissions.
 - Finalement, on joint encore une fois les tables nombre_décès et note_satisfaction, comme on le fait pour les hospitalisations, puis on filtre chaque champ ce qui nous donne une table qui correspond au schéma.

On finalise la table des faits en réalisant une union des deux parties décrites plus haut ce qui nous donne la table de fait finale et complète. Pour ce qui est de la table de faits, elle sera alimentée tous les mois par cette tâche qui représente donc un ETL. Cette grande table nous permettra par la suite de répondre à l'ensemble des besoins clients exprimés en amont, avec parfois même plus de granularité. Effectivement bien qu'elle soit illisible en soit, elle contient pourtant l'ensemble des données nécessaire aux calculs des différentes mesures.

V) Hiérarchie du data Lake

L'ensemble des données stockées dans l'environnement HDFS est divisé en 5 dossiers, chacun représentant la dimension qu'il contient.

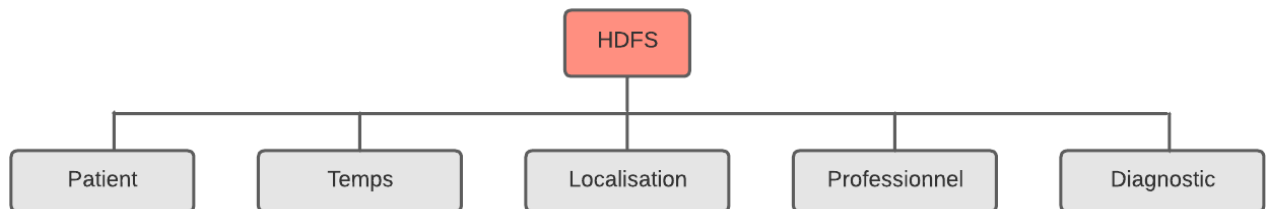


Figure 19 Ensemble des dossiers stockant les dimensions dans le Data Lake

Parmi les 5 dossiers, 3 bénéficieront d'un ensemble de sous-dossiers, afin de partitionner les jeux de données lors du stockage. Cette granularité permettra une diminution du temps d'analyse des données, sans pour autant impacter la quantité de données stockées.

Ainsi, la dimension Patient sera partitionnée par Sexe, puis par tranche d'âge de 10 en 10 car une analyse du client nécessite une analyse par Sexe et par âge. La fourchette des tranches d'âge sera définie par le jeu de données traité, comme la tranche d'âge sera générée à partir de l'âge réel du patient lors du Job de chargement de la dimension Patient.

Cette tranche de 10 en 10 a été décidée à la suite de l'observation d'analyses médicales existantes faites par la DREES ou l'INSEE.

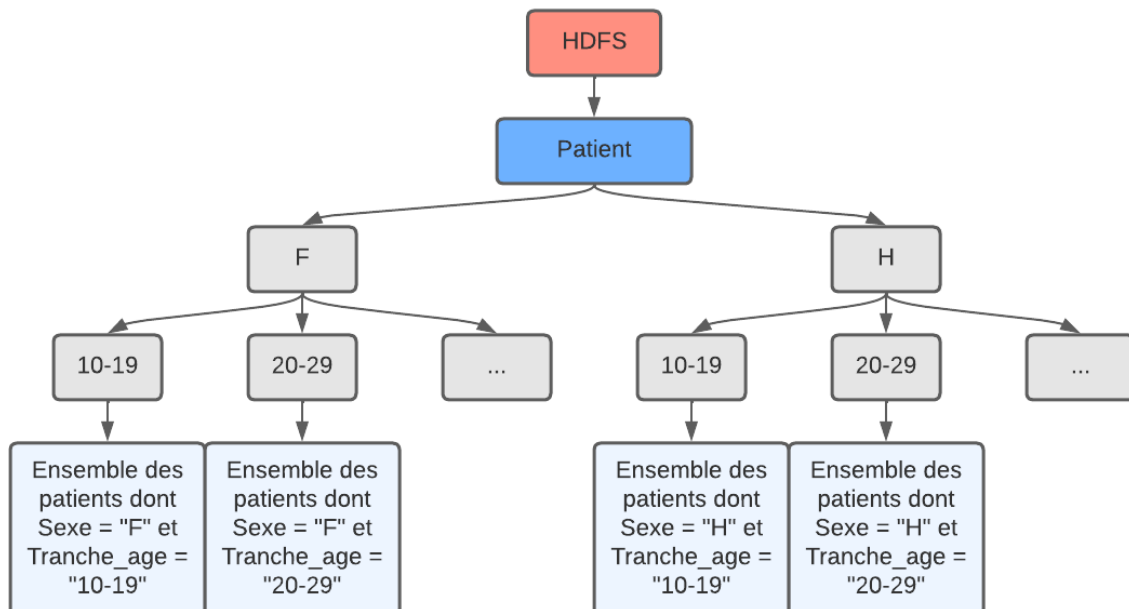


Figure 20 L'arborescence des dossiers stockant la dimension Patient dans le Data Lake

La dimension Temps sera divisée par année, compte-tenu du fait que 2 analyses du client concernent des années précises.

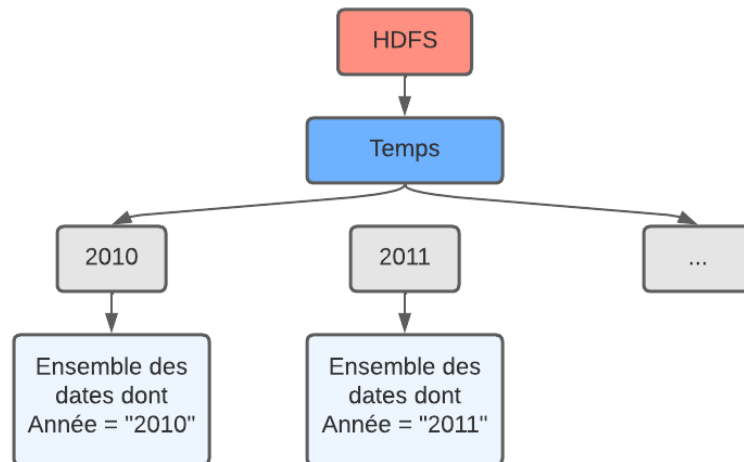


Figure 21 L'arborescence des dossiers stockant la dimension Temps dans le Data Lake

La dernière dimension qui sera partitionnée est la dimension Localisation. Elle sera divisée par le champ *Region*, car 2 analyses du client nécessitent une analyse par régions.

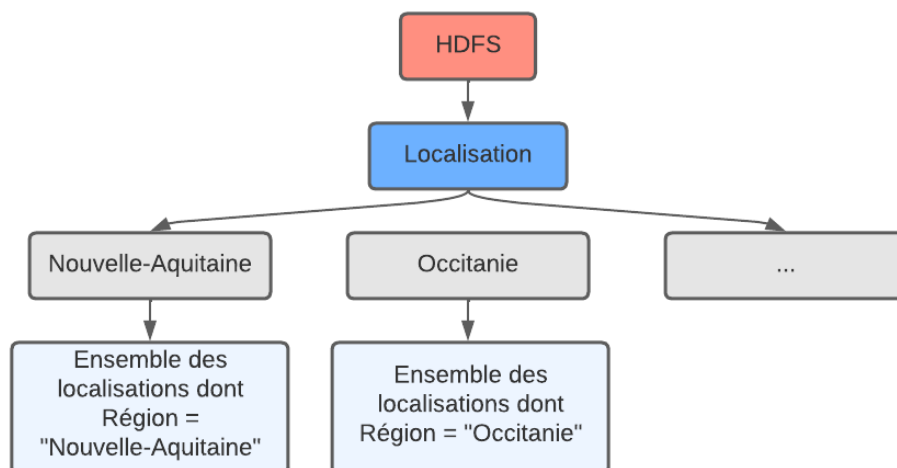


Figure 22 L'arborescence des dossiers stockant la dimension Localisation dans le Data Lake

VI) Variables de contexte

Les variables de contexte sont utilisées pour la connexion à la base de données PostgreSQL :

Connexion à la base de données

Mettre à jour la connexion à une base de données - Etape 2/2

Type de BdD: PostgreSQL

Version de la base de données: v9 and later

Chaîne de caractères de connexion: jdbc:postgresql://localhost:5432/soins_medico_administratives?

Identifiant: context.postgres_Login

Mot de passe: context.postgres_Password

Serveur: context.postgres_Server

Port: context.postgres_Port

Base de données: context.postgres_Database

Schéma: context.postgres_Schema

Paramètres supplémentaires: context.postgres_AdditionalParams

Tester la connexion

Exporter en tant que contexte

Revenir au contexte précédent

Figure 23 Création du contexte de connexion à la BDD PostgreSQL

Pour éviter de remplir manuellement les champs de connexions, les variables de contexte de PostgreSQL permettent de stocker les valeurs à remplir, pour pouvoir par exemple être réutilisées si besoin. Voici la valeur des différentes variables du contexte *PostgreSQL* :

5	[-] postgres (from repository context)			
6	postgres_AdditionalParams	String		
7	postgres_Database	String		soins_medico_administrativ
8	postgres_Login	String		postgres
9	postgres_Password	Password		****
10	postgres_Port	String		5432
11	postgres_Schema	String		public
12	postgres_Server	String		localhost

Figure 24 Ensemble des variables de contexte concernant la BDD PostgreSQL

De plus, les variables de contexte sont également utilisées au niveau du cluster Cloudera pour le stockage *HDFS*. Voici les valeurs des différentes variables du contexte *HDFS*, utilisé à la fin des différents jobs d'alimentation :

	Name	Type	Comment	Default
				Value
1	⊞ HDFS (from repository context)			
2	myhadopcluster_hdfs_HdfsFileSe	String ▼		"_"
3	myhadopcluster_hdfs_HdfsRowS	String ▼		"\n"
4	myhadopcluster_hdfs_HdfsUser	String ▼		cloudera

Figure 25 Ensemble des variables de contexte concernant le cluster HDFS

Enfin, les composants « Référentiel » permettent également de réutiliser les schémas de données, comme c'est le cas pour le job d'alimentation des localisations :

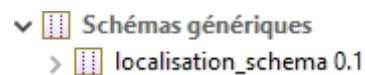


Figure 26 Schémas de données des localisations

Le schéma est ici enregistré en tant que métadonnée et il est ensuite réutilisé dans plusieurs composants tout au long du job.



VII) Conclusion

Nous avons réalisé les deux premières grandes phases de ce projet que sont l'étude de l'architecture et la modélisation des données. Plus précisément, ce livrable nous a permis de détailler :

- L'architecture Big Data qui sera mise en place, comprenant un business Model ainsi que l'ensemble des outils utilisés,
- Le modèle de données en étoile qui nous servira de référentiel lors du stockage des dimensions de nos données analytiques,
- L'ensemble des jobs et traitements automatisés qui nous permettront de charger les données existantes dans nos dimensions et de construire notre table de faits,
- L'organisation du stockage de nos dimensions au sein du Data Lake,
- Les variables contextuelles d'applications, nous servants à mutualiser la déclaration de certaines variables communes aux jobs.

L'ensemble de ces composants, modèles de données et processus sont la garantie de pertinence des analyses futures, en permettant d'éviter les potentielles erreurs tout en étant assuré de répondre aux différents besoins.