UNIVERSITÉ
CAEN
NORMANDIE

CC2 Base De Données

Créer - Alimenter un entrepôt de données

Noms et Prénoms de l'étudiant :

DIALLO Fatoumata Mbalou

SOW Ismael

Numéro de l'étudiants :

22013282 - 22014919

Parcours : M1 Science de données

Sous la supervison :

ZUNUTINI Bruno

Octobre 2023

0.1 Question 1

Proposer une modélisation en étoile des données, en prenant en compte les axes d'analyse décrits ci-dessus. Indication : On pourra répéter les données correspondant aux villes, départements, régions, etc. pour les lieux de réalisation des prestations et pour les lieux de résidence des clients.

Réponse 1 :

- **DIMDate :** Cette dimension représente l'information sur les dates. Elle décompose une date en divers composants tels que : l'année, le trimestre, le mois ...., permettant l'analyse des dates.
- **DIMLieu :** Cette dimension contient des informations géographiques. Elle inclut des détails sur des lieux spécifiques, tels que le code postal, le nom du département, la région... Elle est utile pour des analyses basées sur la situation geographique.
- **DIMPrestation :** Cette dimension représente les types et les prestations vendus. Elle contient des informations comme le code de la prestation, le nom, et la catégorie. Elle permet de catégoriser et d'analyser les différents types de prestation.
- **DIMClient :** Cette dimension est axée sur les clients. Elle inclut des informations personnelles et géographiques sur le client, comme le nom, l'adresse, le code postal.... Cette dimension permet d'effectuer des analyses centrées sur le client.
- **Ventes (Table de faits) :** Cette table contient les mesures et les clés étrangères reliant les dimensions associées. Elle permet d'analyser les ventes en fonction de divers axes comme la date, le lieu, la prestation et le client.

Classes

1. Dimension "DIMDate"

- **Attributs :**
  - `id_date: INTEGER`
  - `datetimeSQL: TEXT`
  - `année: INTEGER`
  - `trimestre: INTEGER`
  - `mois: INTEGER`
  - `nom_du_mois: TEXT`
  - `jour_du_mois: INTEGER`
  - `jour_semaine: TEXT`
  - `heure: INTEGER`
  - `minute: INTEGER`

2. Dimension "DIMLieu"

- **Attributs :**
  - id_lieu: INTEGER
  - insee: TEXT
  - zip: TEXT
  - name: TEXT
  - slug: TEXT
  - departments_id: TEXT
  - departments_code: TEXT
  - departments_name: TEXT
  - departments_slug: TEXT
  - region_id: INTEGER
  - Code: TEXT
  - region_name: TEXT
  - region_slug: TEXT

3. Dimension "DIMPrestation"

- **Attributs :**
  - id_prestation: INTEGER
  - code_prestation: TEXT
  - nom: TEXT
  - catégorie: TEXT

4. Dimension "DIMClient"

- **Attributs :**
  - id_client1: INTEGER
  - id_client: INTEGER
  - num_client: INTEGER
  - nom: TEXT
  - prenom: TEXT
  - adresse: TEXT
  - code_code: TEXT
  - ville: TEXT
  - departments_id: TEXT
  - departments_code: TEXT
  - departments_name: TEXT

- departments_slug: TEXT
- region_id: INTEGER
- Code: TEXT
- region_name: TEXT
- region_slug: TEXT

5. Table de Faits : "Ventes"

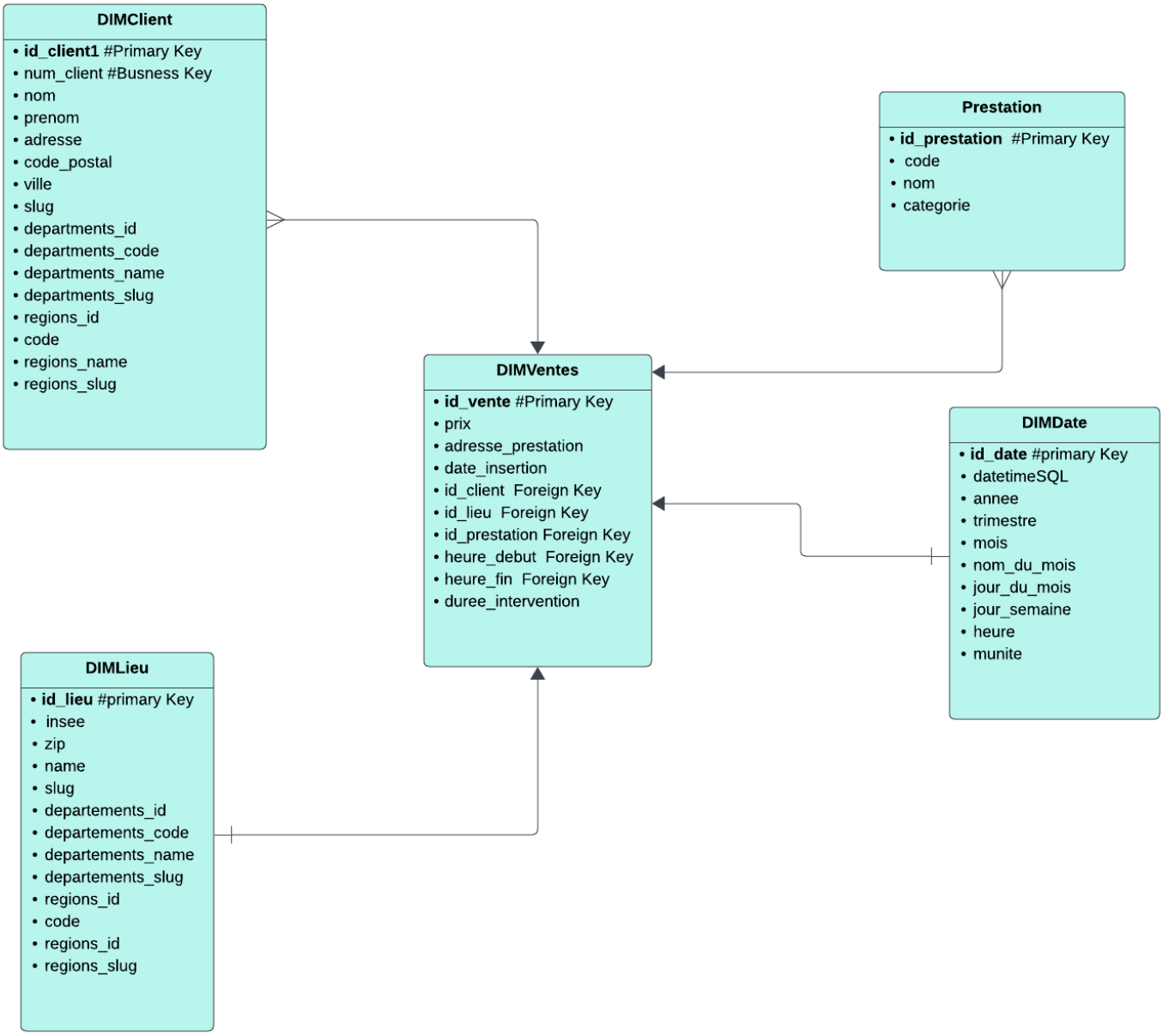
- **Attributs :**
  - id_vente: INTEGER
  - prix: REAL
  - adresse_prestation: TEXT
  - date_insertion: TEXT
  - id_client: INTEGER
  - id_lieu: INTEGER
  - id_prestation: INTEGER
  - heure_debut: INTEGER
  - heure_fin: INTEGER
  - duree_intervention: INTEGER

0.2 Question 2

Déduire de la modélisation un schéma relationnel pour un entrepôt de données. Veiller à insérer les éléments nécessaires pour que les données soient résilientes aux changements (adresses des clients), et pour permettre la traçabilité des données.

Réponse 2 :

la modelisation ci-dessous : A noté que DIMVentes répresente la table de FAITS



Tables de dimensions et de faits

1 TALEND

1.1 Exercice 3

Écrire les scripts SQL permettant de créer les tables correspondant à votre modélisation dans une base de données SQLite. Exécuter ces scripts et les conserver. Le datawarehouse est désormais prêt à recevoir les données.

Réponse 3 : Les scripts SQL de la creation des se trouver dans le fichier "**Table.sql**".

1.2 Exercice 4

Créer un job Talend pour alimenter la dimension correspondant aux dates, et un pour alimenter celle correspondant aux lieux de réalisation des prestations.

Réponse 4 : les job dates et lieux sont respectivement dans le dossier talend (lieu, date)

Job date : Ce job Talend commence par établir une connexion à une base de données avec **tDBConnection_1**. Il lit des données depuis une source externe nommée "date.csv", les transforme via **tMap_1**, puis les insère dans la base de données avec **tDBOutput_1**. Si tout se passe bien, il les commit avec **tDBCommit_1**. En cas d'erreur, la connexion est fermée avec **tDBClose_1**.

Job lieu : Ce Job établit une connexion à une base de données via **tDBConnection_1**. Il récupère des données d'une source nommée "geography.csv" via le fichier delimité , les traitent avec **tMap_1**, et les enregistrent dans la base de données via **tDBOutput_1**. En cas de succès, **tDBCommit_1** valide les données. Si une erreur survient, la connexion est interrompue avec **tDBClose_1**.

1.3 Exercice 5

Créer un job Talend pour alimenter les autres dimensions du datawarehouse, en les transformant autant que rendu nécessaire par la structure de votre datawarehouse.

Réponse 5 : job prestation et client sont respectivement dans le dossier talend (client, prestation).

Job prestation : Ce job Talend initie une connexion à une base de données via **tDBConnection_1**. Il extrait des données d'une source appelée " prestaion.csv" , les transforme à travers **tMap_1**, et les sauvegarde dans la base de données grâce à **tDBOutput_1**. Si l'opération réussit, les modifications sont confirmées par **tDBCommit_1**. Sinon, en cas d'erreur, **tDBClose_1** clôture la connexion.

Job client : Ce job Talend établit deux connexions de base de données via **tDBConnection_1** provenant oprationnel_data.db et **tDBConnection_2** provenant de notre base de données (sqlLite créer) . Lit les données de la première avec `tDBInput_1`, les transforme avec **tMap_1**, puis les écrit dans la deuxième via **tDBOutput_1**. Si le processus se déroule correctement, il les commit avec **tDBCommit_1** ; sinon, il ferme les deux connexions avec **tDBClose_1** et **tDBClose_2**.

1.4 Exercie 6

Créer un job Talend permettant d'alimenter la table de faits, à partir des données de la table Ventes, en les transformant à nouveau autant que nécessaire

Réponse 6 : le job vente se trouvre dans le dossier talend "vente".

Job ventes : Ce job démarre par une connexion à la base de données "operational_data.db" via **tDBConnection**. Les données principales proviennent du composant **Input_ventes** qui contient de la table vente. Ces données sont ensuite enrichies à travers plusieurs jointures (**tMap** composants) en utilisant des données supplémentaires depuis diverses sources comme **InputClients**, **date1**, **date2**, **lieux** et **prestation**. Chaque **tMap**

exécute une jointure, générant des flux intermédiaires . Une fois les transformations terminées, les données résultantes sont sauvegardées dans la base de données via **tDBOutput_1**. Les confirmations de transaction et les clôtures de connexion sont gérées par **tDBCommit_1** et les composants **tDBClose**. En cas de succès, le job utilise **OnComponentOk**, tandis qu'en cas d'erreur, il passe par **OnComponentError**.

1.5 Conclusion :

Dans l'exercice, quatre jobs Talend sont créés pour alimenter différentes dimensions et la table de faits d'un datawarehouse. Les jobs pour les dimensions "date" , "lieu" et "Prestation" récupèrent et transforment les données depuis des fichiers CSV avant de les stocker dans la base de données. tandis que le job "client" établit deux connexions de base de données pour lire et écrire les données. Enfin, le job "ventes" effectue des jointures complexes pour enrichir les données avant de les insérer dans la table de faits. Tous les jobs gèrent les confirmations de transaction et les clôtures de connexion.

2 OLAP

Réponse global sur Olap : Afin d'optimiser les opérations OLAP, notre entrepôt de données a été transformé en un fichier XML structuré. Cette conversion permet une manipulation plus aisée des données pour des analyses multidimensionnelles. Les requêtes spécifiques pour ces analyses, numérotées de 1 à 11, sont regroupées dans un dossier séparé intitulé "exemples". Ce dossier sert de référentiel pour tous les de requêtes OLAP demander.