

Compte Rendu - TP2 : Prise en main de l'ETL Apache Hop et population du Data Warehouse

Introduction

Ce TP a pour objectif de découvrir Apache Hop, un outil ETL (Extract, Transform, Load) open-source, et de l'utiliser pour peupler un Data Warehouse avec des données de ventes. Le travail consiste à créer des pipelines pour charger les tables de dimensions puis la table de faits, tout en respectant l'ordre logique de chargement.

1. Exploration de la documentation Apache Hop

Transformations identifiées :

Besoin	Transform utilisé	Description
Lire un CSV	CSV file input	Lit les données depuis un fichier CSV
Transformer les données	Select values	Sélectionne, renomme et change le type des colonnes
Rechercher des clés	Stream Lookup	Effectue des jointures en mémoire pour récupérer les clés étrangères
Charger dans la base	Table output	Insère les données dans une table PostgreSQL
Lire depuis la base	Table input	Lit les données depuis une table PostgreSQL
Trier les données	Sort rows	Trie les données selon un ou plusieurs champs

2. Chargement des tables de dimension

2.1 Dimension CHANNELS

Pipeline créé :

- **CSV file input** → Lecture du fichier channels.csv
- **Unique rows (hashset)** → Transform supprime les lignes dupliquées et ne conserve que les lignes uniques comme données d'entrée pour la transformation.
- **Table output** → Insertion dans la table channels

2.2 Dimension PRODUCTS

Pipeline créé :

- **CSV file input** → Lecture du fichier PRODUCTS.csv
- **Table output** → Insertion dans la table products

2.3 Dimension CUSTOMERS

Pipeline créé :

- **CSV file input** → Lecture du fichier `Customers.csv`
- **Unique rows (hashset)** → **Transform supprime les lignes dupliquées et ne conserve que les lignes uniques comme données d'entrée pour la transformation.**
- **Table output** → Insertion dans la table `customers`

2.4 Dimension TIMES

Pipeline créé :

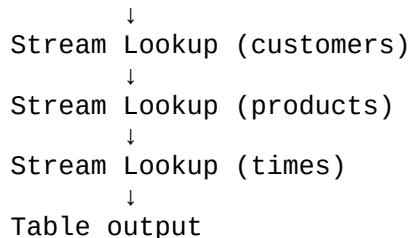
- **CSV file input** → Lecture du fichier `TIMES - TIMES.csv`
- **Sort rows** → Tri par date (`day_key`) en ordre croissant
- **Unique rows** → Enlève les duplication du clé primaire
- **Table output** → Insertion dans la table `times`

3. Chargement de la table de faits

3.1 Architecture du pipeline

Le pipeline de chargement de la table de faits `sales` est plus complexe car il nécessite de récupérer les clés étrangères des dimensions :

`CSV file input` →`user defiened java expression` → `Select values` → `Stream Lookup (channels)`



3.2 Détails des transformations

1. CSV file input :

- Lecture du fichier `SALES_FACTSN.csv`
- Contient les données brutes des ventes

2. User Defined Java Expression :

- Permet d'appliquer une expression Java personnalisée
- Utilisée pour transformer `DAY_KEY`
- Expression utilisée :

```
new java.sql.Timestamp(DAY_KEY.getTime() + (15L * 365 * 24 * 60 * 60 * 1000))
```

- Cette expression ajoute **15 ans** à la date contenue dans `DAY_KEY` et génère un nouvel objet `Timestamp`

3. Select values :

- Sélection des colonnes nécessaires
- Renommage si besoin
- **Conversion du type de day_key**

4. Stream Lookups (4 jointures) :

Chaque lookup récupère la clé primaire de la dimension correspondante :

Lookup	Table dimension	Champ de jointure	Clé récupérée
Stream lookup	channels	channel	channel_key
Stream lookup 3	customers	customer	cust_key
Stream lookup 4	products	product	prod_key
Stream lookup 2	times	day_key	day_key

4. Table output :

- Insertion dans la table sales