

---

---

# Création d'un data warehouse, processus ETL et analyse

---

---

PROJET DATA WAREHOUSE

MANAL EZ-ZAHER  
ANTHONY MENGHI  
MEHDI GHOUAM

21 NOVEMBRE 2024

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Définition de l'objectif d'analyse</b>	<b>2</b>
<b>3</b>	<b>Conception du Data Warehouse</b>	<b>2</b>
<b>4</b>	<b>Création du processus ETL avec Apache Hop</b>	<b>4</b>
4.1	Extraction des données . . . . .	5
4.1.1	Extraction depuis PostgreSQL . . . . .	5
4.1.2	Extraction depuis MongoDB . . . . .	5
4.2	Transformation des données . . . . .	6
4.2.1	Décomposition des documents JSON (MongoDB) . . . . .	6
4.2.2	Nettoyage et filtration des données . . . . .	6
4.2.3	Enrichissement des données . . . . .	6
4.2.4	Jointures et alignement des données . . . . .	6
4.2.5	Gestion des clés étrangères . . . . .	6
4.3	Chargement des données . . . . .	6
4.3.1	Chargement des tables dimensionnelles . . . . .	6
4.3.2	Chargement de la table des faits . . . . .	6
4.3.3	Utilisation d'Apache Hop . . . . .	7
<b>5</b>	<b>Création d'un cube OLAP et Requêtes analytiques</b>	<b>7</b>
5.1	Création du cube OLAP dans IcCube . . . . .	7
5.1.1	Étapes de création du cube . . . . .	7
5.2	Requêtes analytiques MDX . . . . .	8
5.2.1	Nombre de cartes par type d'abonnement . . . . .	9
5.2.2	Nombre de cartes par collection . . . . .	9
5.2.3	Nombre de cartes par adresse email . . . . .	9
5.2.4	Nombre de cartes par date . . . . .	9
5.3	Conclusion . . . . .	9

# 1 Introduction

Ce rapport détaille les étapes clés de la conception et de la mise en œuvre d’une solution analytique basée sur un Data Warehouse. Le projet vise à répondre à des objectifs analytiques précis, tels que l’analyse de la distribution des cartes au sein d’une collection de flashcards, en utilisant des données provenant de sources différentes.

Pour cela, nous avons structuré les données selon un modèle en étoile, mis en place un processus ETL pour leur intégration, construit un cube OLAP avec IcCube et formulé des requêtes analytiques avancées en MDX. Ces étapes permettent une exploration détaillée et croisée des données pour fournir des insights exploitables.

## 2 Définition de l’objectif d’analyse

Notre objectif analytique est de comprendre la distribution et les caractéristiques des cartes au sein d’une collection. Plus précisément, nous cherchons à analyser les données selon différents axes et attributs pour répondre aux questions suivantes :

- Combien de cartes sont présentes par collection ?
- Quelle est la répartition des cartes selon les différents types d’abonnement ?
- Combien de cartes sont associées à chaque adresse e-mail unique ?
- Quelle est la distribution des cartes selon leurs dates de création ou d’activation ?

Ces analyses permettront de mieux comprendre les tendances et les comportements liés aux cartes, en offrant une vision globale et détaillée de leur répartition.

## 3 Conception du Data Warehouse

Le Data Warehouse est structuré en un schéma en étoile pour répondre aux objectifs analytiques définis. Ce schéma comprend une table de faits et plusieurs tables de dimensions, comme suit :

### Table de faits

La table de faits centralise les informations clés issues des deux sources de données (PostgreSQL et MongoDB) pour effectuer les analyses.

```
1 CREATE TABLE fact_table (  
2     ticket_id INT,  
3     subscription_id INT,  
4     mobile_id INT,  
5     user_id INT,  
6     collection_id INT,  
7     card_count INT  
8 );
```

- `ticket_id` : Identifiant des tickets (source MongoDB).
- `subscription_id` : Identifiant des abonnements (source PostgreSQL et MongoDB).
- `mobile_id` : Identifiant des sessions mobiles (source MongoDB).
- `user_id` : Identifiant des utilisateurs (lien avec `dim_users`, source PostgreSQL).
- `collection_id` : Identifiant des collections (source PostgreSQL).
- `card_count` : Nombre de cartes par collection (source PostgreSQL).

## Tables de dimensions

### Dimension Collection

Contient les informations sur les collections issues de PostgreSQL.

```

1 CREATE TABLE dim_collection (
2     collection_id INT,
3     owner_id INT,
4     title VARCHAR(255),
5     description VARCHAR(255),
6     visibility BOOLEAN,
7     color VARCHAR(255),
8     icon VARCHAR(255),
9     created_at TIMESTAMP,
10    updated_at TIMESTAMP,
11    card_count INT
12 );

```

### Dimension Utilisateurs

Contient les informations des utilisateurs issues de PostgreSQL.

```

1 CREATE TABLE dim_users (
2     user_id INT,
3     email VARCHAR(255),
4     username VARCHAR(255),
5     bio TEXT,
6     terms_accepted_at TIMESTAMP,
7     updated_at TIMESTAMP,
8     created_at TIMESTAMP
9 );

```

### Dimension Mobile

Contient les informations issues des sessions mobiles provenant de MongoDB.

```

1 CREATE TABLE dim_mobile (
2     mobile_id INT,

```

```

3      date DATE,
4      clicks INT,
5      feature_name VARCHAR(255),
6      premium_features_usage INT,
7      session_duration INT,
8      ai_card_modifications INT,
9      user_id INT
10 );

```

### Dimension Abonnement

Contient les données d'abonnement des utilisateurs provenant de PostgreSQL et MongoDB.

```

1 CREATE TABLE dim_subscription (
2     subscription_id INT,
3     action_type VARCHAR(255),
4     subscription_type VARCHAR(255),
5     date DATE,
6     user_id INT
7 );

```

### Dimension Tickets

Contient les données des tickets de support provenant de MongoDB.

```

1 CREATE TABLE dim_ticket (
2     ticket_id INT,
3     user_id INT,
4     date DATE,
5     category VARCHAR(255),
6     status VARCHAR(255)
7 );

```

## 4 Création du processus ETL avec Apache Hop

Dans ce projet, nous avons utilisé Apache Hop pour concevoir, implémenter et exécuter un processus ETL. L'objectif était d'extraire les données des sources PostgreSQL et MongoDB, de les transformer, puis de les charger dans le Data Warehouse.

La base de données PostgreSQL contient les informations structurées sur les utilisateurs, les collections et les cartes. Tandis que la base de données MongoDB contient les interactions utilisateurs sous forme de documents JSON semi-structurés.

Le processus ETL (Extraction, Transformation, Chargement) est structuré en trois grandes étapes : extraction, transformation et chargement (voir [1](#)).

## 4.1 Extraction des données

L'extraction des données consiste à récupérer les informations depuis les deux sources de données principales afin de les préparer pour la transformation. Le Data Warehouse se connecte aux bases de données PostgreSQL et MongoDB.

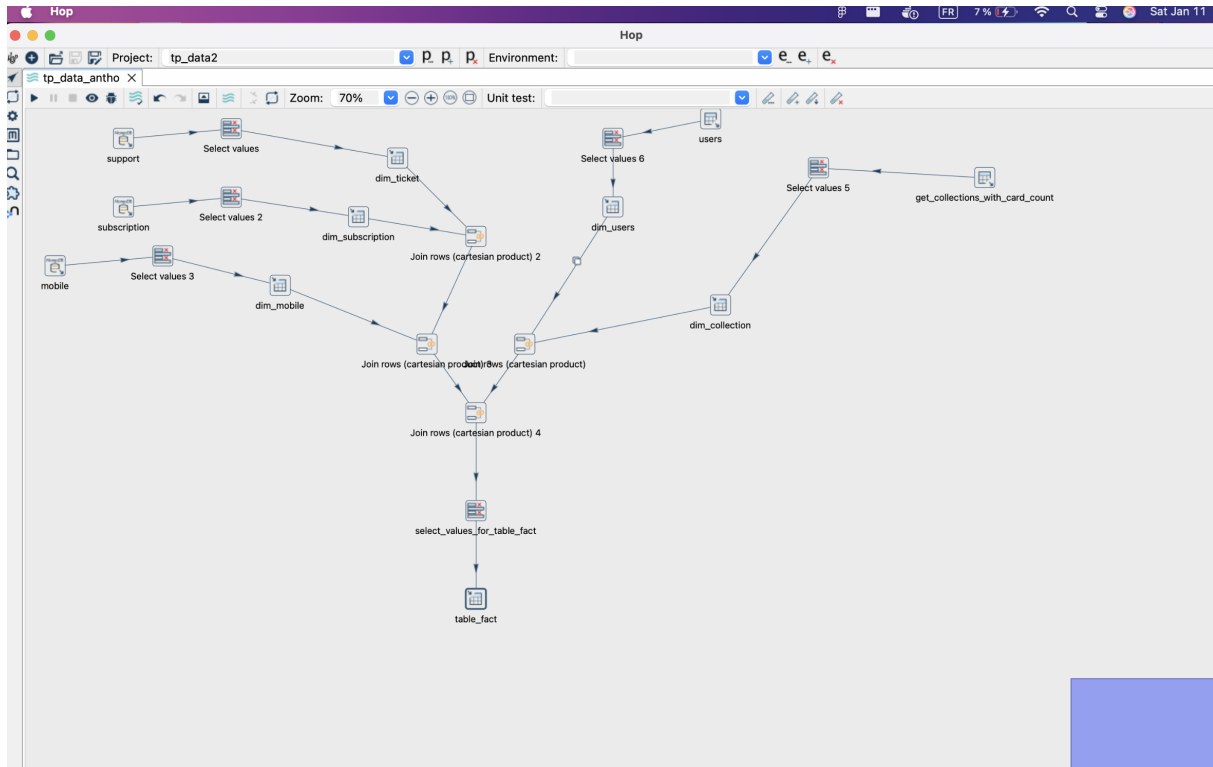


Figure 1: ETL Apache hop

### 4.1.1 Extraction depuis PostgreSQL

Les données nécessaires à la création des dimensions et de la table de faits sont extraites des tables suivantes :

- **users**
- **collections**
- **cards**

### 4.1.2 Extraction depuis MongoDB

Les collections suivantes ont été ciblées pour extraire les données pertinentes :

- **website** : Données sur les sessions web (IP, clics, durée de session, etc.).
- **mobile** : Données des sessions mobiles (clics sur des fonctionnalités, utilisation des fonctionnalités premium, etc.).
- **support** : Informations sur les tickets.
- **subscription** : Données des abonnements des utilisateurs.

## 4.2 Transformation des données

Les données extraites sont transformées afin de s'adapter au modèle en étoile défini pour le Data Warehouse. Les étapes principales de cette transformation sont décrites ci-dessous.

### 4.2.1 Décomposition des documents JSON (MongoDB)

- Extraction des champs imbriqués, comme `feature_clicks` (clics par fonctionnalité).
- Conversion des structures JSON en colonnes relationnelles.

### 4.2.2 Nettoyage et filtration des données

- Suppression des champs inutiles ou redondants.
- Normalisation des valeurs (par exemple, formatage des dates et des chaînes de caractères).

### 4.2.3 Enrichissement des données

- Calcul de colonnes dérivées, telles que le total des cartes par collection ou le nombre de modifications de cartes par utilisateur.

### 4.2.4 Jointures et alignement des données

- Association des informations provenant de PostgreSQL et MongoDB à l'aide des identifiants communs (`user_id`, `collection_id`, etc.).
- Création des dimensions (`dim_users`, `dim_collection`, etc.) et de la table des faits (`fact_table`) conformément au modèle défini.

### 4.2.5 Gestion des clés étrangères

- Génération et mapping des clés étrangères pour garantir l'intégrité des relations dans le Data Warehouse.

## 4.3 Chargement des données

Une fois les données transformées, elles sont chargées dans le Data Warehouse, en suivant les étapes ci-dessous :

### 4.3.1 Chargement des tables dimensionnelles

Les tables dimensionnelles, telles que `dim_users`, `dim_collection` et `dim_mobile`, sont chargées en premier dans PostgreSQL.

### 4.3.2 Chargement de la table des faits

La table de faits `fact_table` est alimentée ensuite, en utilisant les clés des dimensions pour maintenir l'intégrité référentielle.

### 4.3.3 Utilisation d'Apache Hop

Apache Hop a été utilisé pour gérer le processus de chargement :

- **Extraction** : Les connecteurs PostgreSQL et MongoDB ont été configurés pour récupérer les données des deux bases.
- **Transformation** : Les étapes JSON Input, Filter Rows et Join Rows ont permis de nettoyer, enrichir et formater les données.
- **Chargement** : Les tables de dimensions et la table des faits ont été écrites directement dans PostgreSQL grâce à l'étape Table Output.

Grâce à ce processus ETL, les deux sources de données ont été intégrées dans un Data Warehouse structuré en étoile. De cette façon, nous pouvons répondre efficacement aux objectifs analytiques, tels que le suivi des collections, des abonnements et des interactions utilisateurs.

## 5 Création d'un cube OLAP et Requêtes analytiques

### 5.1 Création du cube OLAP dans IcCube

Pour structurer les données comme convenus dans les objectifs de la partie 1, un cube OLAP nommé `tpcube` a été créé dans IcCube. Ce cube repose sur les dimensions et les mesures dérivées de la table des faits et des dimensions disponibles dans le Data Warehouse.

#### 5.1.1 Étapes de création du cube

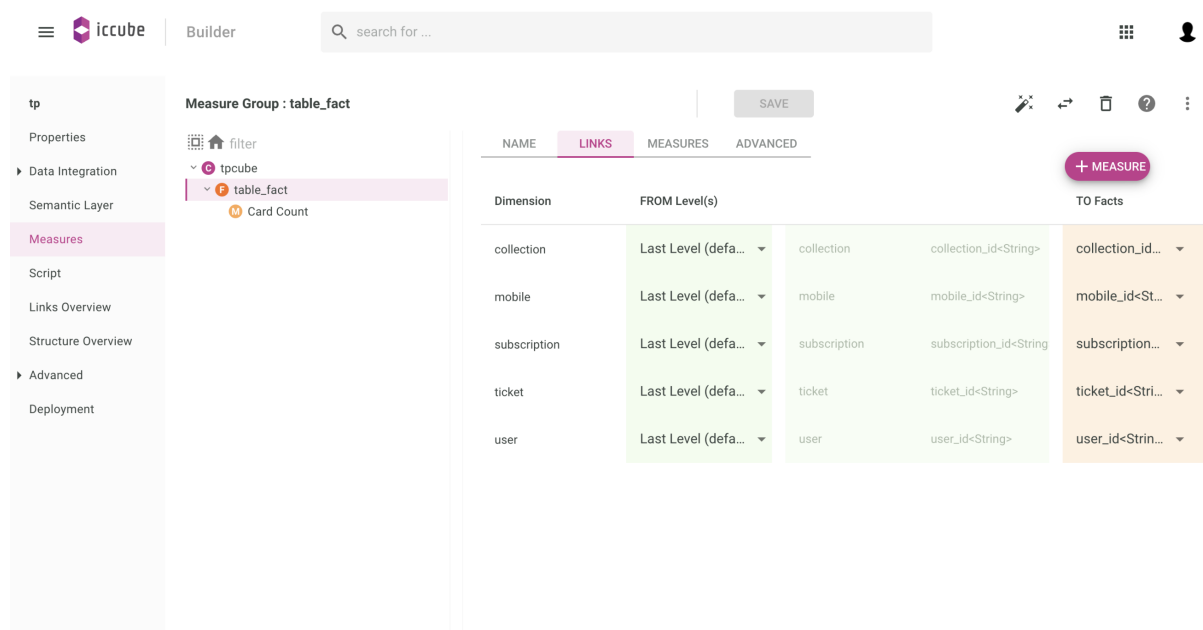


Figure 2: Mesures sur Iccube

### Définition des mesures



- **Card Count** : Mesure principale représentant le nombre de cartes (voir 2).
  - Calculée à partir de la colonne `card_count` de la table des faits.

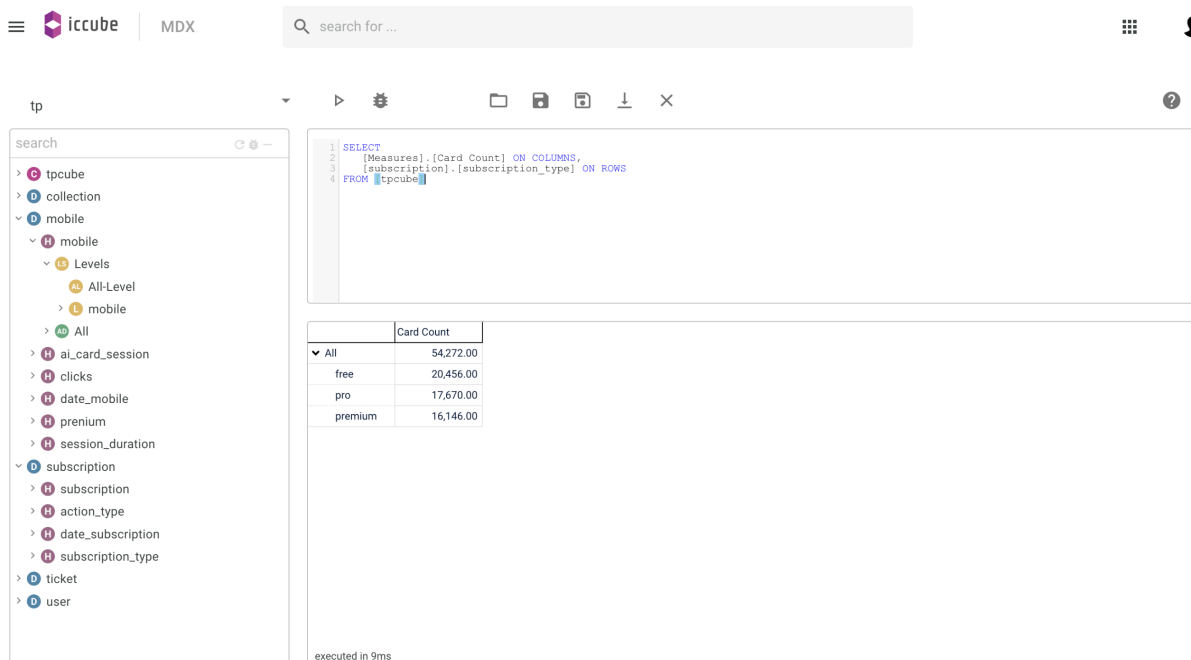
## Création des dimensions

- **Collection** : Basée sur la dimension `dim_collection`, permettant de regrouper les cartes par collection.
- **Abonnement (Subscription)** : Basée sur `dim_subscription`, permettant une analyse par type et date d'abonnement.
- **Utilisateur (User)** : Basée sur `dim_users`, pour analyser les cartes par adresse email.

## Configuration dans IcCube

- Les mesures et dimensions ont été définies via l'interface graphique d'IcCube.
- Un fichier XML de configuration a été généré et déployé pour activer le cube.
- Les relations entre la table des faits et les dimensions ont été configurées en respectant les clés primaires et étrangères.

## 5.2 Requêtes analytiques MDX



The screenshot shows the IcCube web interface for MDX queries. On the left is a tree view of dimensions: tp, collection, mobile, mobile (with sub-levels All-Level and mobile), All, ai\_card\_session, clicks, date\_mobile, premium, session\_duration, subscription (with sub-levels subscription, action\_type, date\_subscription, subscription\_type), ticket, and user. The main area displays an MDX query:

```
SELECT
[Measures].[Card Count] ON COLUMNS,
[subscription].[subscription_type] ON ROWS
FROM [tpcube]
```

Below the query editor, a table shows the results of the query:

	Card Count
All	54,272.00
free	20,456.00
pro	17,670.00
premium	16,146.00

At the bottom, it indicates "executed in 9ms".

Figure 3: Requête MDX

Plusieurs requêtes MDX ont été conçues pour analyser les données et répondre aux objectifs définis (voir 3)

### 5.2.1 Nombre de cartes par type d'abonnement

Cette requête permet de visualiser le nombre total de cartes regroupées par type d'abonnement.

```
1 SELECT
2     [Measures].[Card Count] ON COLUMNS,
3     [subscription].[subscription_type] ON ROWS
4 FROM [tpcube]
```

### 5.2.2 Nombre de cartes par collection

Cette requête liste le nombre de cartes pour chaque collection.

```
1 SELECT
2     [Measures].[Card Count] ON COLUMNS,
3     [collection].[collection] ON ROWS
4 FROM [tpcube]
```

### 5.2.3 Nombre de cartes par adresse email

Cette requête compte le nombre de cartes associées à chaque utilisateur, identifié par son adresse email.

```
1 SELECT
2     [Measures].[Card Count] ON COLUMNS,
3     [user].[email] ON ROWS
4 FROM [tpcube]
```

### 5.2.4 Nombre de cartes par date

Cette requête regroupe les cartes en fonction de leur date d'abonnement.

```
1 SELECT
2     [Measures].[Card Count] ON COLUMNS,
3     [subscription].[date_subscription] ON ROWS
4 FROM [tpcube]
```

## 5.3 Conclusion

Le cube OLAP `tpcube` a été conçu pour permettre une analyse croisée efficace des données grâce aux dimensions et mesures définies. Les requêtes MDX illustrent la capacité du cube à répondre directement aux objectifs analytiques définis en phase 1. Ces analyses offrent des perspectives précises sur les collections, les abonnements, les utilisateurs et les tendances temporelles.