**Universite Cadi Ayyad**

**Ecole Supérieur de Technologie – Safi**

**Département Informatique**

**RAPPORT DE PROJET**

***Conception d’un Data Warehouse aéroportuaire, Reporting et Tableaux de bord avec Analyse décisionnelle par Intelligence Artificielle***

Spécialité : **Ingénierie des systèmes d’information et réseaux**

Réalisé par :

* **Mr. TAIRARI Zakaria**
* **Mr. HASSIB Hicham**

Encadré par :

* **Mme. MOUNIR Ilham**

**2025-2026**

Sommaire

[1. Présentation générale 4](#_Toc218687273)

[2. Objectifs métiers 4](#_Toc218687274)

[3. Systèmes sources (OLTP) 5](#_Toc218687275)

[3.1 Système de Gestion des Vols (PostgreSQL) 5](#_Toc218687276)

[3.2 Système de Gestion des Passagers (MySQL) 8](#_Toc218687277)

[3.3 Système des Boutiques & Restaurants (CSV, JSON) 9](#_Toc218687278)

[4. Architecture du Data Warehouse 10](#_Toc218687279)

[4.1 Vue d’ensemble 10](#_Toc218687280)

[4.2 Data Warehouse (Modèle Dimensionnel) 10](#_Toc218687281)

[5. Modèle décisionnel (Schéma en étoile) 11](#_Toc218687282)

[5.1 Dimensions 11](#_Toc218687283)

[5.2 Mesures 12](#_Toc218687284)

[5.3 Tables de Faits 13](#_Toc218687285)

[5.4 Data Marts 13](#_Toc218687286)

[5.5 Data Warehouse schema 16](#_Toc218687287)

[6. Processus ETL (Extract – Transform – Load) 17](#_Toc218687288)

[6.1 Phase d’Extraction (Extract) 17](#_Toc218687289)

[6.2 Phase de Transformation (Transform) 17](#_Toc218687290)

[6.3 Phase de Chargement (Load) 18](#_Toc218687291)

[6.4 ETL avec Pentaho Data Integration (PDI) 19](#_Toc218687292)

[Étape 1 : Connexion aux sources de données 20](#_Toc218687293)

[Étape 2 : Extraction des données 21](#_Toc218687294)

[Étape 3 : Transformation des données 24](#_Toc218687295)

[Étape 4 : Chargement des Tables 29](#_Toc218687296)

[6.5 Gestion des erreurs et journalisation 30](#_Toc218687297)

[6.6 Avantages de l’utilisation de PDI 30](#_Toc218687298)

[7 Reporting et Tableaux de Bord (Power BI) 31](#_Toc218687299)

[7.1 Objectif du Reporting Décisionnel 31](#_Toc218687300)

[7.2 Outil de Business Intelligence Utilisé 32](#_Toc218687301)

[7.3 Connexion au Data Warehouse 33](#_Toc218687302)

[7.4 Modèle de Données dans Power BI 34](#_Toc218687303)

[7.5 Mesures et Calculs DAX 35](#_Toc218687304)

[7.6 Annual Report Dashboard 36](#_Toc218687305)

[7.7 Flights Performance Dashboard 37](#_Toc218687306)

[7.8 Commercial / Passenger / Shops Dashboard 38](#_Toc218687307)

[8 Analyse avancée par Data Mining 39](#_Toc218687308)

[8.1 Introduction à la phase de Data Mining 39](#_Toc218687309)

[8.2 Objectif du Data Mining dans ce projet 39](#_Toc218687310)

[8.3 Description des données utilisées 39](#_Toc218687311)

[8.4 Explication de l’algorithme utilisé 40](#_Toc218687312)

[Partie 1 : Méthode du coude (Elbow method) 41](#_Toc218687313)

[Partie 2 : Algorithme K-Means 43](#_Toc218687314)

[8.5 Analyse et interprétation des résultats 45](#_Toc218687315)

[Conclusion 46](#_Toc218687316)

# 1. Présentation générale

L’Aéroport International Marrakech–Menara souhaite mettre en place un système décisionnel (Data Warehouse) afin d’améliorer la performance opérationnelle, la gestion des passagers, la ponctualité des vols, et les revenus commerciaux.

L’aéroport possède plusieurs systèmes opérationnels distincts (OLTP), chacun utilisant sa propre base de données et ses propres formats. Ces systèmes ne communiquent pas bien entre eux, ce qui rend difficile :

* l’analyse des flux de passagers,
* le suivi des retards,
* l’optimisation des terminaux et comptoirs d’enregistrement,
* l’analyse des revenus des boutiques,
* le reporting stratégique.

Le Data Warehouse servira de source unique de vérité (SSOT).

# 2. Objectifs métiers

Le Data Warehouse devra répondre aux questions stratégiques suivantes :

**Trafic passagers**

* Combien de passagers transitent par jour / mois / année ?
* Quels sont les créneaux horaires les plus chargés ?
* Quels terminaux sont saturés ?

**Performance des vols**

* Quels vols et compagnies sont les plus ponctuels ?
* Quelle est la durée moyenne des retards ?
* Quels sont les facteurs liés aux retards (destination, compagnie, terminal) ?

**Revenus commerciaux**

* Quelles boutiques génèrent le plus de revenus ?
* Quels types (Duty Free, restauration, mode…) performent le mieux ?
* Quels jours / mois génèrent le plus de ventes ?

# 3. Systèmes sources (OLTP)

## 3.1 Système de Gestion des Vols (PostgreSQL)

**Table : airlines**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| airline\_id | INT PK | Identifiant airline |
| airline\_code | VARCHAR(5) | Code IATA (AT, FR, AF…) |
| airline\_name | VARCHAR(100) |  |
| country | VARCHAR(50) |  |
| alliance | VARCHAR(50) | SkyTeam, Oneworld… |

**Table : aircrafts**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| aircraft\_id | INT PK | Identifiant aircraft |
| registration\_number | VARCHAR(20) |  |
| aircraft\_type | VARCHAR(50) | Boeing 737, Airbus A320… |
| manufacturer | VARCHAR(50) |  |
| seat\_capacity | INT |  |
| airline\_id | INT FK(Airlines) |  |

**Table : terminals**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| terminal\_id | INT PK | Identifiant terminal |
| terminal\_code | VARCHAR(5) | T1, T2… |
| capacity | INT |  |
| terminal\_type | ENUM | Domestic, International |

**Table : gates**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| gate\_id | INT PK | Identifiant gate |
| gate\_code | VARCHAR(5) | A12, B03 |
| terminal\_id | INT FK(terminals) |  |
| gate\_type | ENUM | Domestic, International |

**Table : airports**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| airport\_id | INT PK | Identifiant aeroport |
| airport\_code | VARCHAR(5) | Code IATA |
| airport\_name | VARCHAR(100) |  |
| city | VARCHAR(100) |  |
| country | VARCHAR(50) |  |
| timezone | VARCHAR(50) |  |
| latitude | DECIMAL(9, 6) |  |
| longitude | DECIMAL(9, 6) |  |

**Table : flights**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| flight\_id | INT PK | Identifiant vol |
| flight\_number | VARCHAR(20) | AT402 |
| airline\_id | INT FK(Airlines) |  |
| aircraft\_id | INT FK(Aircrafts) |  |
| origin\_airport\_id | INT FK(Airports) |  |
| destination\_airport\_id | INT FK(Airports) |  |
| scheduled\_departure | DATETIME |  |
| actual\_departure | DATETIME |  |
| scheduled\_arrival | DATETIME |  |
| actual\_arrival | DATETIME |  |
| gate\_id | INT FK(Gates) |  |
| terminal\_id | INT FK(Terminals) |  |
| flight\_status | ENUM | Scheduled, Boarding, Departed, Arrived, Delayed, Cancelled |
| delay\_code | ENUM | Weather, Technical, Crew, ATC |
| cancellation\_reason | TEXT |  |
| passenger\_count | INT |  |

## 3.2 Système de Gestion des Passagers (MySQL)

**Table : passengers**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| passenger\_id | INT PK | Identifiant passager |
| first\_name | VARCHAR(80) |  |
| last\_name | VARCHAR(80) |  |
| birth\_date | DATE |  |
| nationality | VARCHAR(50) |  |
| gender | ENUM | Sexe (Female, Male) |
| email | VARCHAR(100) |  |
| phone | VARCHAR(50) |  |
| passenger\_type | ENUM | Tourist, Business |
| passport\_number | VARCHAR(20) |  |

**Table : checkins**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| checkin\_id | INT PK | Identifiant checkin |
| passenger\_id | INT FK(Passengers) |  |
| flight\_id | INT FK(from Flighs system) |  |
| checkin\_time | DATETIME |  |
| terminal\_id | INT FK(from Flighs system) |  |
| desk\_number | VARCHAR(10) |  |
| seat\_number | VARCHAR(10) |  |
| boarding\_pass\_number | VARCHAR(20) |  |
| baggage\_count | INT |  |
| baggage\_weight | DECIMAL(5,2) |  |
| checkin\_status | ENUM | Completed, Cancelled |

## 3.3 Système des Boutiques & Restaurants (CSV, JSON)

**CSV : shops.csv**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| shop\_id | INTEGER | PK |
| shop\_name | STRING |  |
| shop\_category | ENUM | Fashion, Food… |
| terminal\_id | INTEGER | Reference terminal |
| floor | INTEGER | 0 / 1 / 2 |
| opening\_hour | TIMESTAMP |  |
| closing\_hour | TIMESTAMP |  |
| monthly\_rent | DECIMAL(10, 2) |  |

**JSON : sales\_transactions.json**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| sale\_id | INTEGER | PK |
| sale\_number | STRING |  |
| shop\_id | INTEGER | Reference shop |
| sale\_datetime | DATETIME |  |
| total\_price | DECIMAL |  |
| tax\_price | DECIMAL |  |
| discount\_price | DECIMAL |  |
| payment\_method | STRING | Card, Cash |
| customer\_type | STRING | Passenger, Staff, Visitor |

# 4. Architecture du Data Warehouse

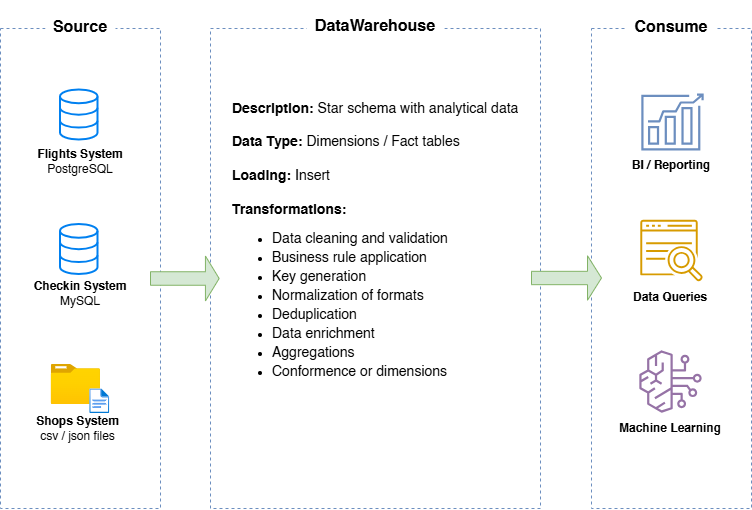
## 4.1 Vue d’ensemble

Le Data Warehouse pour le projet de l’Aéroport International Marrakech–Menara est conçu pour intégrer les données provenant de plusieurs systèmes opérationnels (vols, passagers, boutiques, sécurité) dans un référentiel centralisé à des fins analytiques.

L’architecture suit désormais une approche ETL, où les données sont chargées depuis les systèmes sources vers les tables de dimensions et de faits.

Cette approche simplifie le processus ETL tout en garantissant la cohérence et la disponibilité des données pour l’analyse.

**Diagramme de l’Architecture DW :**



## 4.2 Data Warehouse (Modèle Dimensionnel)

**Objectif :**

Cette couche contient les tables de dimensions (dim\_\*) et les tables de faits (fact\_\*) formant un schéma en étoile. Les données sont chargées depuis les systèmes sources, prêtes pour l’analyse et le reporting.

**Caractéristiques :**

* **Type de données :** Tables physiques (dimensions et faits)
* **Type de chargement :** Insert direct depuis les systèmes sources

**Transformations :**

* Application des règles métiers et calculs
* Génération de clés substituts (surrogate keys)
* Jointures et mappings directs à partir des données sources
* Agrégations et enrichissements dans les tables de faits
* Conformité des clés de dimension à travers les domaines

# 5. Modèle décisionnel (Schéma en étoile)

## 5.1 Dimensions

**dim\_time** (**time\_sk**, day, month, quarter, semester, year)

**dim\_airline** (**airline\_sk**, airline\_id, airline\_code, airline\_name, airline\_country, airline\_alliance)

**dim\_aircraft** (**aircraft\_sk**, aircraft\_id, aircraft\_registration\_number, aircraft\_type, aircraft\_manufacturer, aircraft\_seat\_capacity)

**dim\_flight** (**flight\_sk**, flight\_id, flight\_number, flight\_status, flight\_delay\_code, flight\_cancellation\_reason)

**dim\_airport** (**airport\_sk**,airport\_id, airport\_code, airport\_name, airport\_city, airport\_country, airport\_timezone, airport\_latitude, airport\_longitude)

**dim\_gate** (**gate\_sk**, gate\_id, gate\_code, gate\_type)

**dim\_terminal** (**terminal\_sk**, terminal\_id, terminal\_code, terminal\_capacity, terminal\_type)

**dim\_passenger** (**passenger\_sk**, passenger\_id, passenger\_nationality, passenger\_gender, passenger\_birth\_date, passenger\_email, passenger\_phone, passenger\_type, passernger\_passport\_number)

**dim\_checkin** (**checkin\_sk**, checkin\_id, checkin\_desk\_number, checkin\_seat\_number, checkin\_boarding\_pass\_number)

**dim\_shop** (**shop\_sk**, shop\_id, shop\_name, shop\_category, shop\_location, shop\_floor, shop\_opening\_hour, shop\_closing\_hour, shop\_monthly\_rent)

**dim\_sale** (**sale\_sk**, sale\_id, sale\_number, sale\_datetime, sale\_payment\_method, sale\_customer\_type)

## 5.2 Mesures

**Fact\_flights :**

|  |  |  |
| --- | --- | --- |
| **Measure** | **Type** | **Description** |
| delay\_departure\_minutes | INT | actual\_departure - scheduled\_departure |
| delay\_arrival\_minutes | INT | actual\_arrival - scheduled\_arrival |
| passenger\_capacity | INT |  |
| passenger\_onboard | INT |  |
| flight\_duration | INT | actual\_arrival - actual\_departure |
| occupancy\_rate | DECIMAL | passenger\_onboard / passenger\_capacity |

**Fact\_checkins :**

|  |  |  |
| --- | --- | --- |
| **Measure** | **Type** | **Description** |
| passenger\_count | INT |  |
| baggage\_count | INT |  |
| baggage\_weight | DECIMAL |  |
| checkins\_completed | INT |  |
| checkins\_cancelled | INT |  |

**Fact\_shop\_sales :**

|  |  |  |
| --- | --- | --- |
| **Measure** | **Type** | **Description** |
| sales\_price\_total | DECIMAL |  |
| sales\_count | INT |  |
| tax\_amount | DECIMAL |  |
| discount\_amount | DECIMAL |  |
| net\_sales | DECIMAL | sales\_total + tax - discount |
| average\_sale\_value | DECIMAL | Sales\_price\_total / sales\_count |

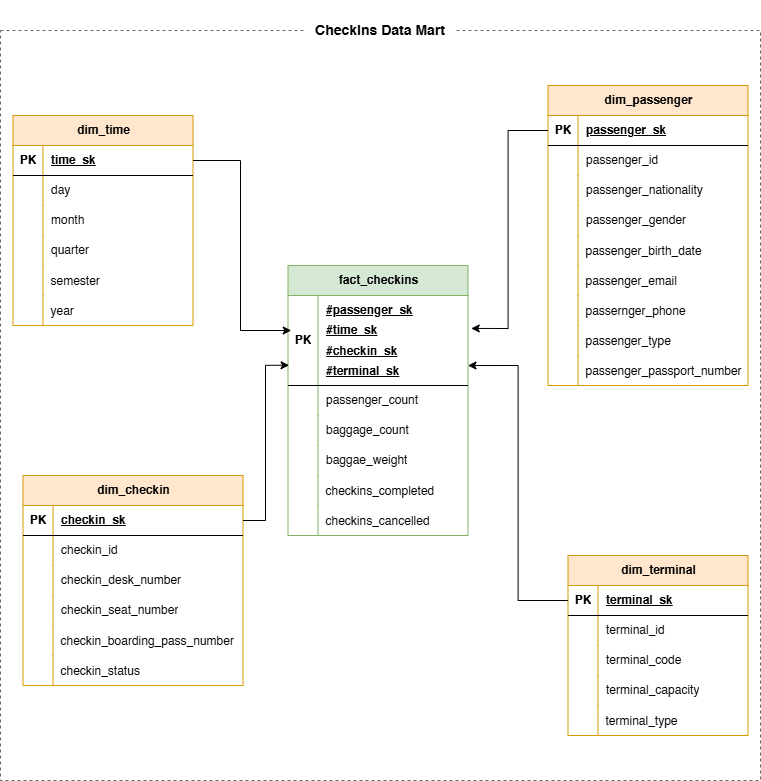
## 5.3 Tables de Faits

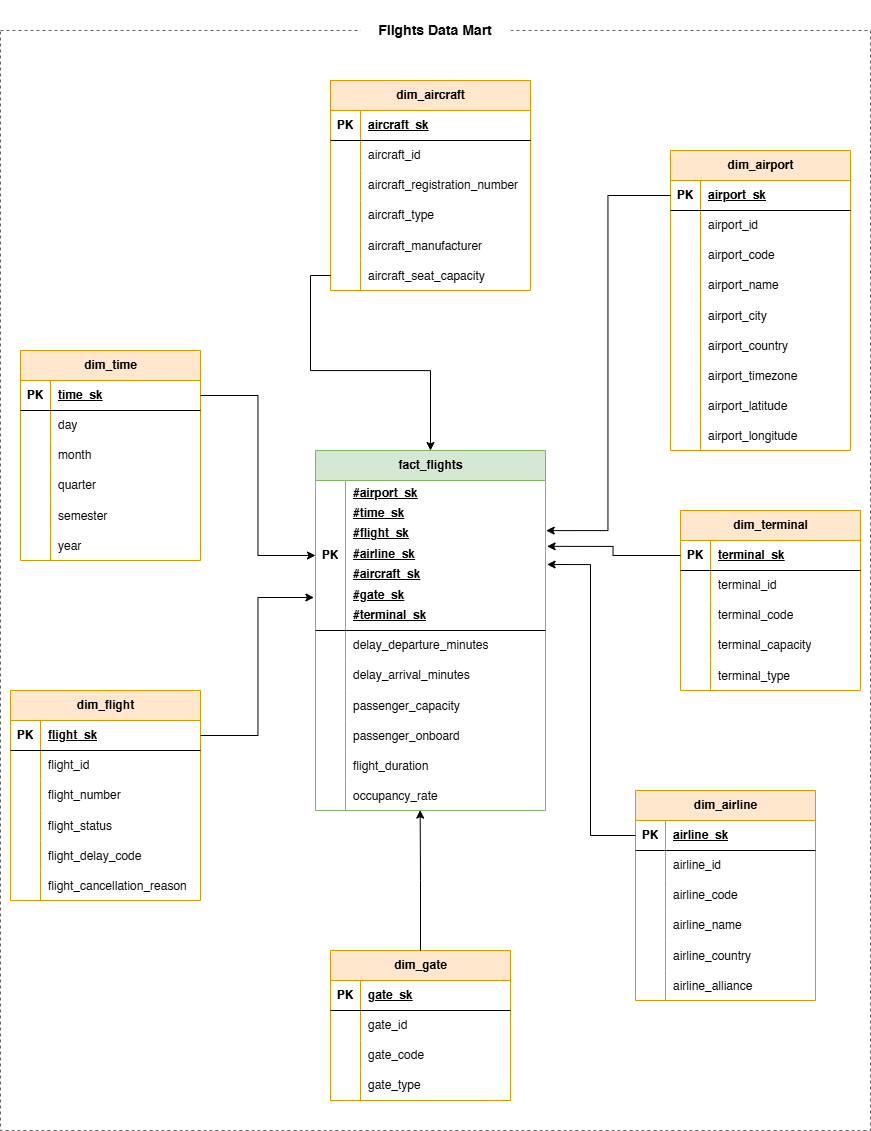
**1. fact\_checkins (#passenger\_sk, #time\_sk, #checkin\_sk, #terminal\_sk**, passenger\_count, baggage\_count, baggage\_weight, checkins\_completed, checkins\_cancelled)

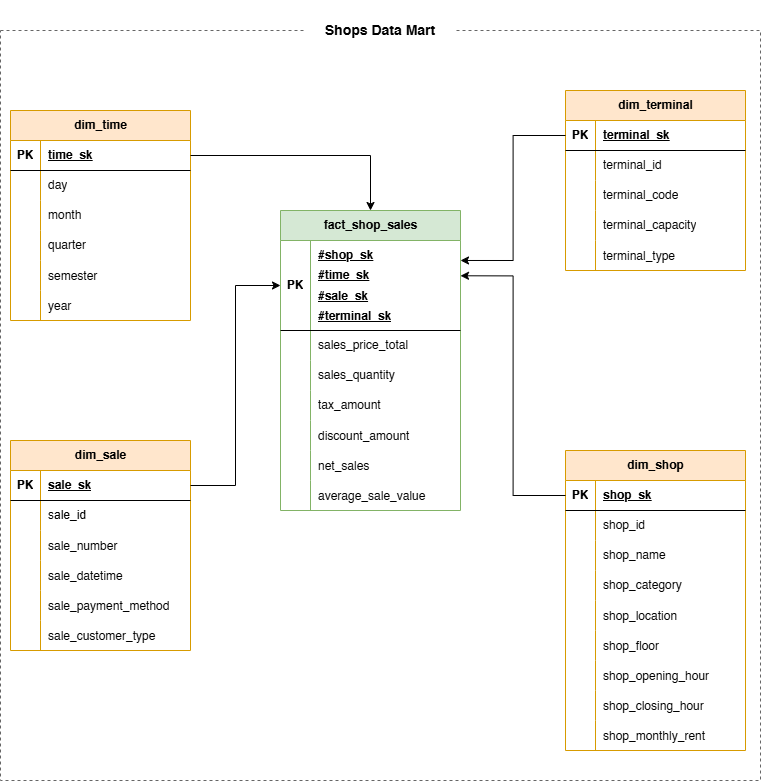
**2. fact\_flights** **(#airport\_sk, #time\_sk, #flight\_sk, #airline\_sk, #aircraft\_sk, #gate\_sk, #terminal\_sk**, delay\_departure, delay\_arrival, passenger\_capacity, passenger\_onboard, flight\_duration, occupancy\_rate)

**3. fact\_shop\_sales** (**#shop\_sk, #time\_sk, #sale\_sk, #terminal\_sk** sales\_price\_total, sales\_quantity, tax\_amount, discount\_amount, net\_sales, average\_sale\_value)

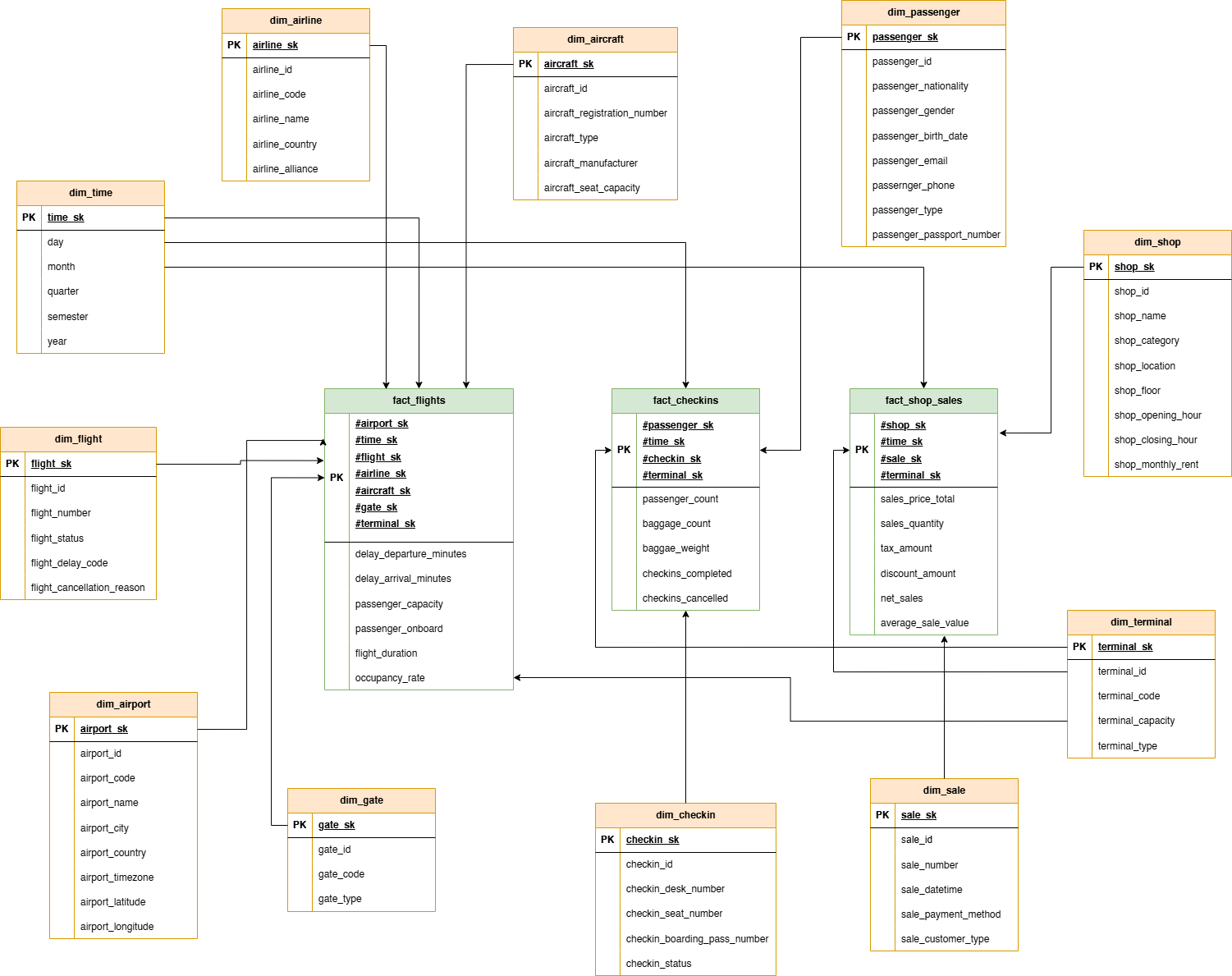
## 5.4 Data Marts







## 5.5 Data Warehouse schema



# 6. Processus ETL (Extract – Transform – Load)

Le processus ETL constitue un élément fondamental du système de data warehousing. Il a pour objectif principal de collecter les données issues des systèmes opérationnels, de les nettoyer et transformer, puis de les charger dans l’entrepôt de données afin de permettre l’analyse décisionnelle, le reporting et les techniques de data mining.

## 6.1 Phase d’Extraction (Extract)

La phase d’extraction consiste à récupérer les données brutes provenant de plusieurs systèmes opérationnels liés aux activités aéroportuaires. Ces systèmes représentent différents domaines fonctionnels de l’aéroport :

* Système de gestion des vols
* Système de check-in
* Système commercial

En raison de l’indisponibilité de données réelles, des données synthétiques ont été générées tout en respectant des règles réalistes afin de garantir la cohérence et la crédibilité des analyses.

## 6.2 Phase de Transformation (Transform)

La phase de transformation vise à nettoyer, harmoniser et enrichir les données afin de les rendre conformes au modèle de l’entrepôt de données.

**Nettoyage des données**

Les principales opérations de nettoyage réalisées sont :

* Suppression des valeurs nulles ou incohérentes
* Correction des formats de date et d’heure
* Élimination des doublons

**Normalisation et standardisation**

Afin d’assurer l’uniformité des données :

* Les formats de date sont standardisés
* Les libellés textuels (compagnies, terminaux, nationalités) sont homogénéisés
* Les unités de mesure sont rendues cohérentes

**Gestion des clés**

Dans l’entrepôt de données, les clés substituts (surrogate keys) sont utilisées à la place des clés métier issues des systèmes opérationnels.

* Chaque dimension possède une clé substitut unique
* Les clés métier sont conservées à titre informatif
* Cette approche améliore la performance et facilite l’évolution du modèle

**Gestion de la dimension Temps**

Une dimension Temps est générée automatiquement afin de permettre les analyses temporelles. Cette dimension est indépendante des systèmes sources et constitue un élément central de l’analyse décisionnelle.

## 6.3 Phase de Chargement (Load)

**Chargement des dimensions**

Les tables de dimensions sont chargées en premier afin de garantir l’intégrité référentielle. Chaque enregistrement est identifié par une clé substitut.

**Chargement des tables de faits**

Après le chargement des dimensions, les tables de faits sont alimentées

Les tables de faits contiennent :

* Des mesures quantitatives (montant des ventes, nombre de check-ins, nombre de vols)
* Des clés étrangères vers les dimensions

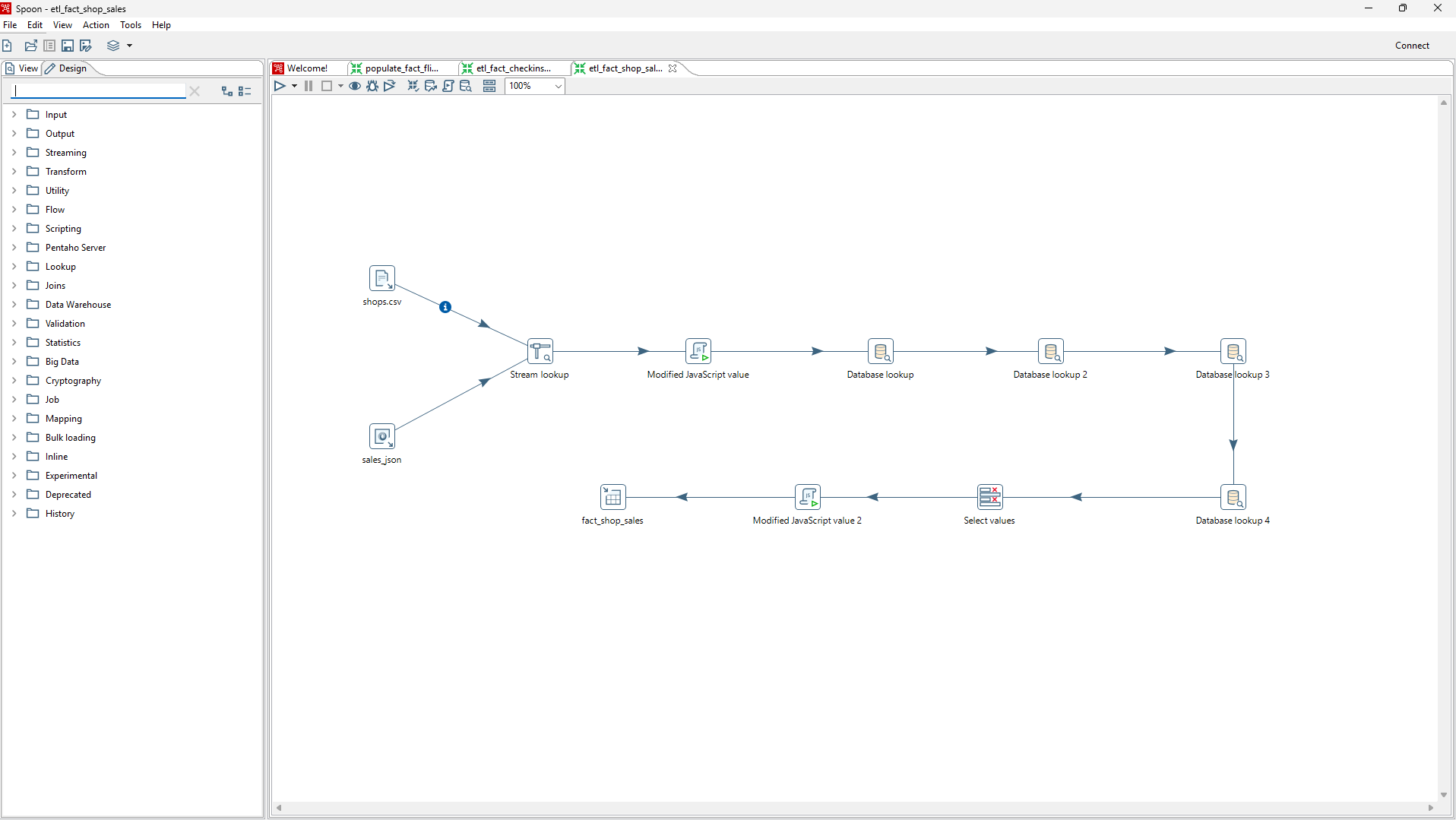
## 6.4 ETL avec Pentaho Data Integration (PDI)

L’implémentation du processus ETL a été réalisée à l’aide de Pentaho Data Integration (PDI), également connu sous le nom de Spoon. PDI est un outil ETL open source permettant de concevoir graphiquement des flux de données pour l’extraction, la transformation et le chargement des données vers un entrepôt de données.

Une transformation représente un flux de traitement des données définissant comment les données sont extraites, transformées et chargées.  
Une transformation est constituée de composants appelés “steps”, reliés entre eux par des liens de données (hops).

Les données circulent ligne par ligne, chaque step appliquant une opération spécifique avant de transmettre les résultats au step suivant. Ce modèle correspond à une architecture en pipeline, favorisant la performance et la modularité.

**Interface avec exemple de transformation:**

****

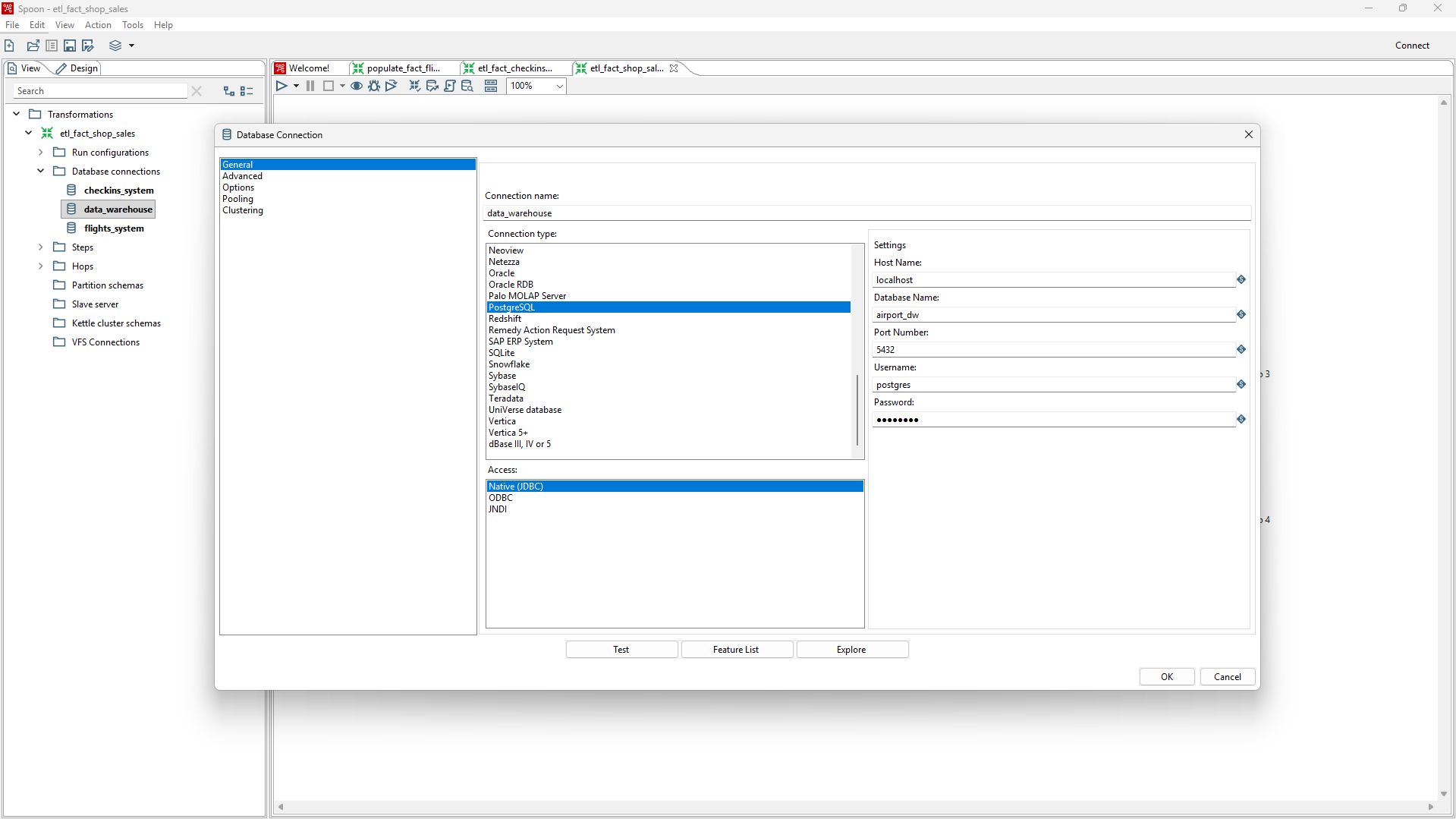
### Étape 1 : Connexion aux sources de données

Dans PDI, des connexions JDBC ont été configurées afin d’accéder :

* Aux bases de données opérationnelles
* À la base de données de l’entrepôt de données

Ces connexions garantissent un accès sécurisé et réutilisable aux différentes sources.

**Exemple de connexion a une base de données :**

****

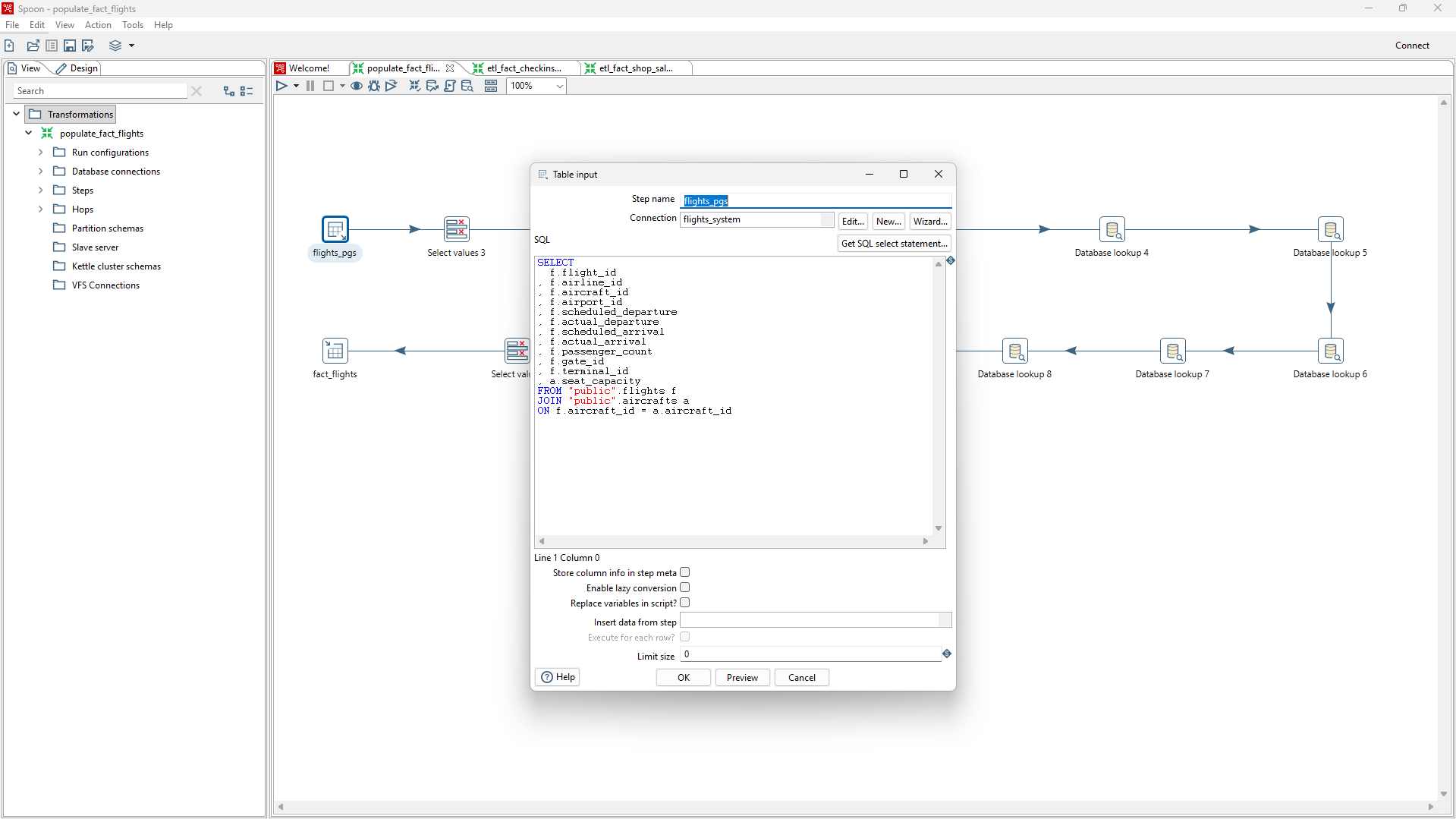
### Étape 2 : Extraction des données

L’extraction est réalisée à l’aide des composants PDI suivants :

* **Table Input** : pour exécuter des requêtes SQL sur les bases sources
* **CSV Input**: pour extraire les données depuis un fichier CSV
* **JSON Input**: pour extraire les données depuis un fichier Json

Les données extraites sont ensuite transmises vers les étapes de transformation.

**Exemple de Table Input :**

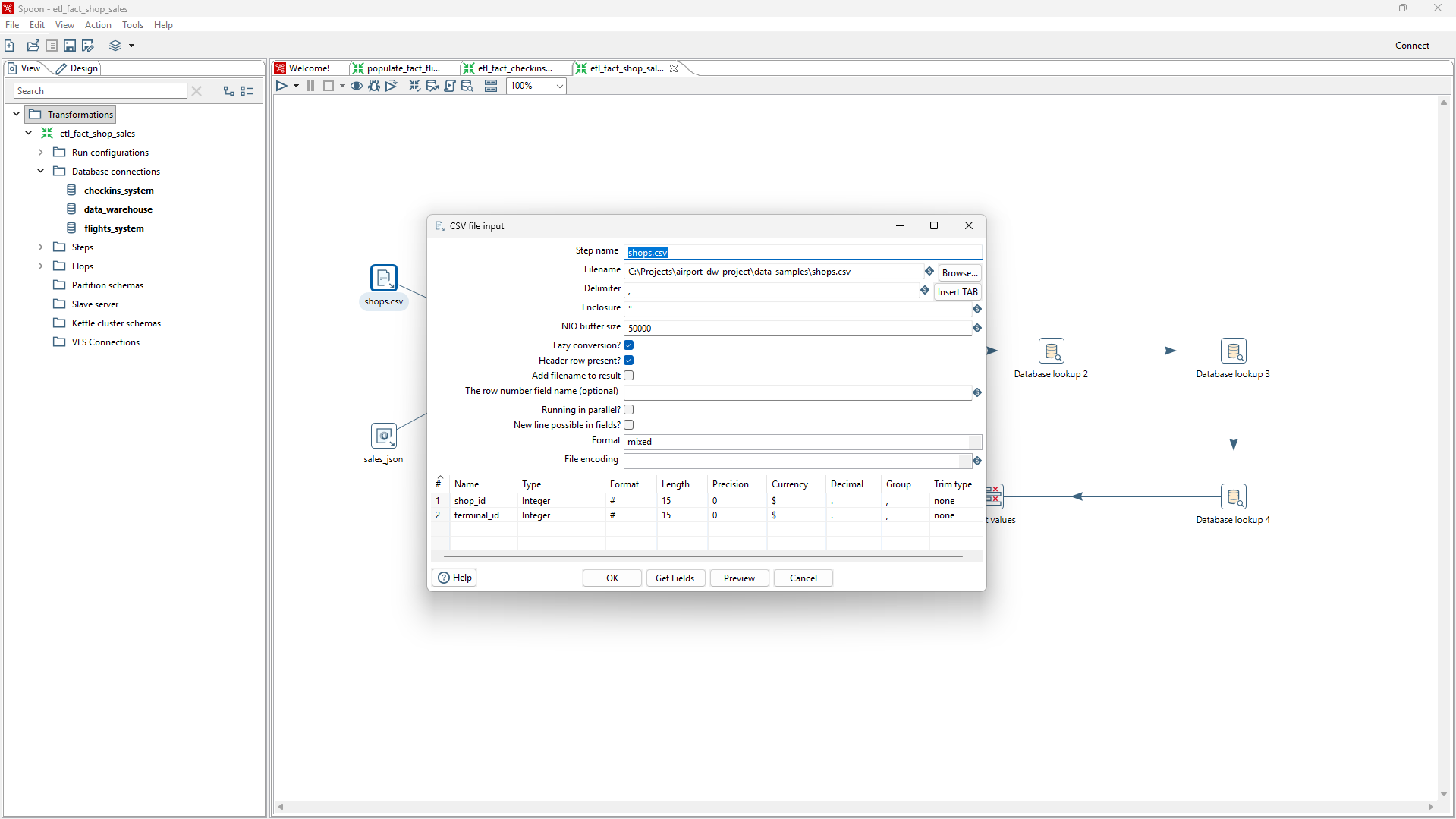
****

**Fonctionnement général**

1. Connexion à la base de données via JDBC
2. Exécution d’une requête SQL définie par l’utilisateur
3. Lecture des résultats ligne par ligne
4. Envoi des données vers le step suivant du flux

Les données sont transmises sous forme de flux de lignes, ce qui permet un traitement en pipeline.

**Exemple de CSV Input :**

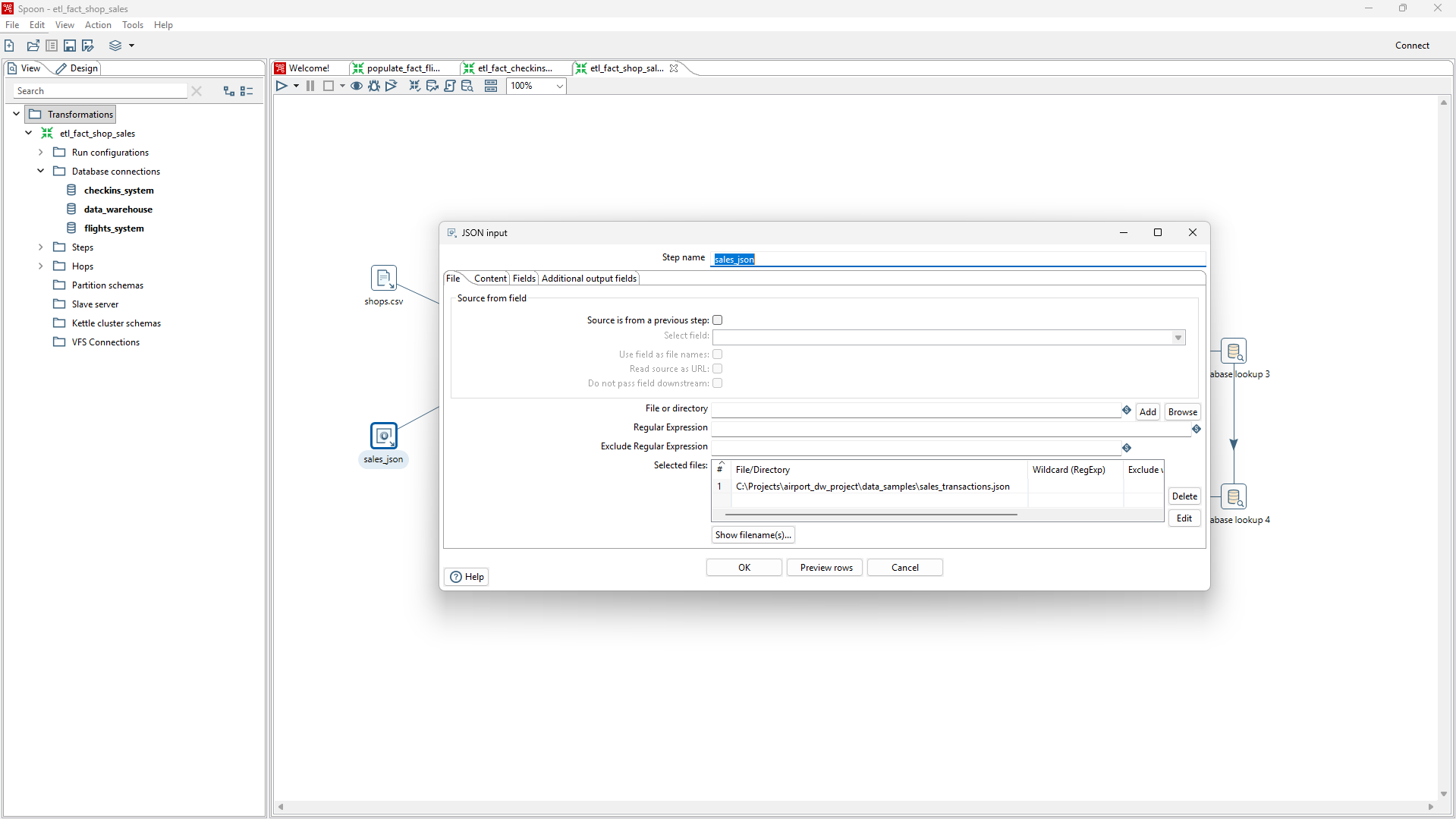
****

**Fonctionnement général**

1. Lecture du fichier CSV depuis le système de fichiers
2. Interprétation de la structure du fichier (séparateur, encodage)
3. Découpage de chaque ligne en colonnes
4. Conversion des valeurs en types de données exploitables
5. Envoi des données vers les étapes suivantes de la transformation

Les données sont traitées ligne par ligne, de manière similaire aux données extraites d’une base relationnelle.

**Exemple de Json Input :**



**Fonctionnement général**

1. Lecture du fichier ou du flux JSON
2. Analyse de la structure hiérarchique (objets, tableaux)
3. Extraction des champs définis via des chemins JSON
4. Transformation des données JSON en lignes tabulaires
5. Transmission des données au pipeline ETL

Chaque élément du fichier JSON est converti en une ou plusieurs lignes de données selon la structure définie.

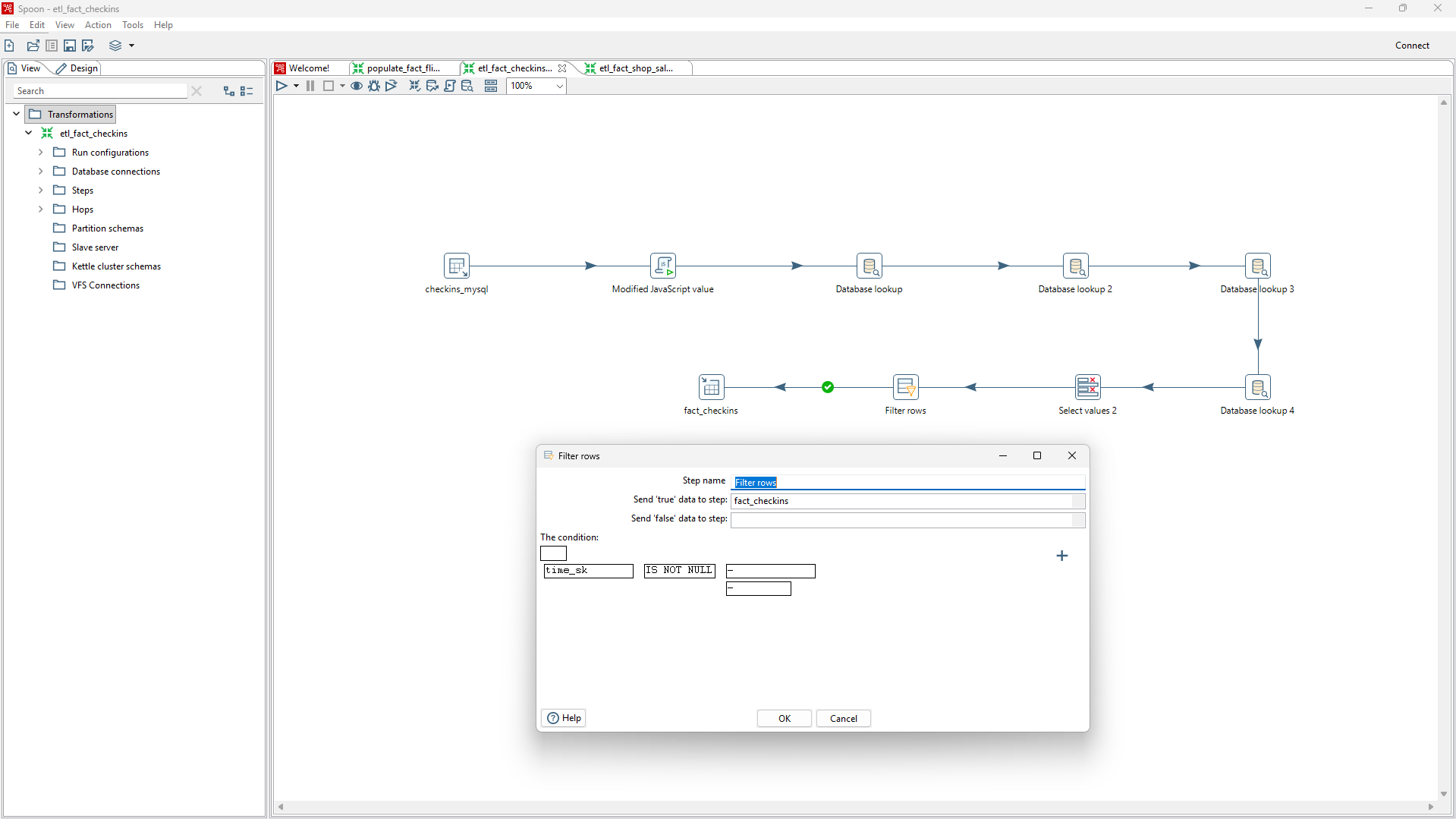
### Étape 3 : Transformation des données

Les transformations appliquées incluent :

**a) Nettoyage des données**

* **Filter Rows** : élimination des enregistrements invalides
* **Select Values** : suppression des champs inutiles

**Exemple de Filter Rows :**



**Fonctionnement général**

1. Définition de conditions logiques : basées sur les valeurs des colonnes.
2. Évaluation de chaque ligne : chaque enregistrement est testé par rapport aux conditions définies.
3. Routage des lignes :
   * Les lignes valides continuent vers les étapes suivantes.
   * Les lignes non conformes peuvent être redirigées vers un flux d’erreur ou ignorées.

**Exemple de Select Values :**

****

**Fonctionnement général**

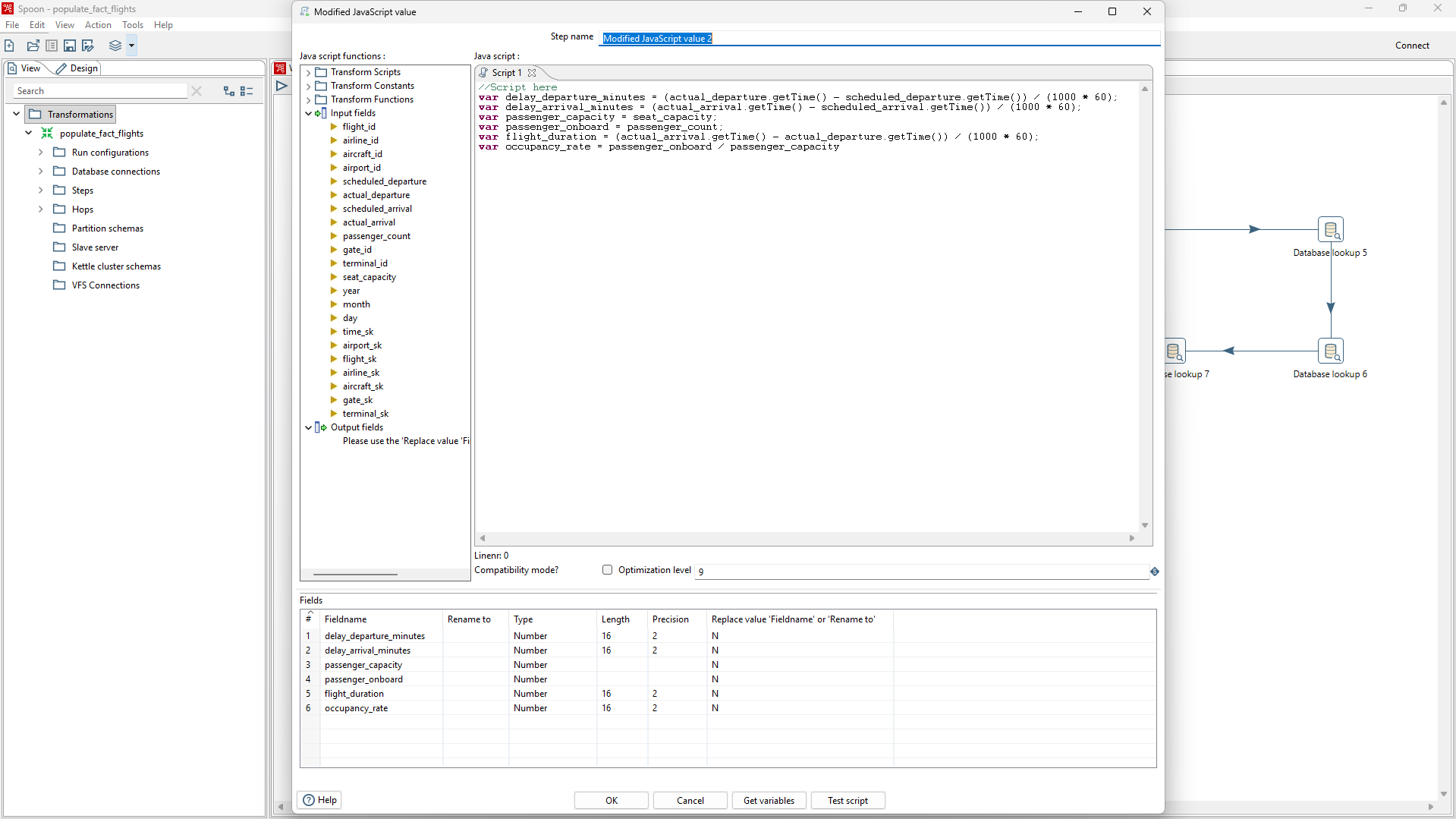
1. Sélection des colonnes : choisir uniquement les champs utiles pour le data warehouse.
2. Renommage des colonnes : harmoniser les noms pour correspondre au modèle de l’entrepôt.
3. Changement de type : convertir les données vers les types requis
4. Suppression des colonnes inutiles : réduire le volume de données et simplifier le flux.

Les données passent intactes d’un step à l’autre, seules les colonnes ou leur structure sont modifiées.

**b) Normalisation et formatage**

* **Modified Javascript Value :** appliquer des transformations personnalisées sur les données d’un flux en utilisant le langage JavaScript.

**Exemple de Modified Javascript Value :**



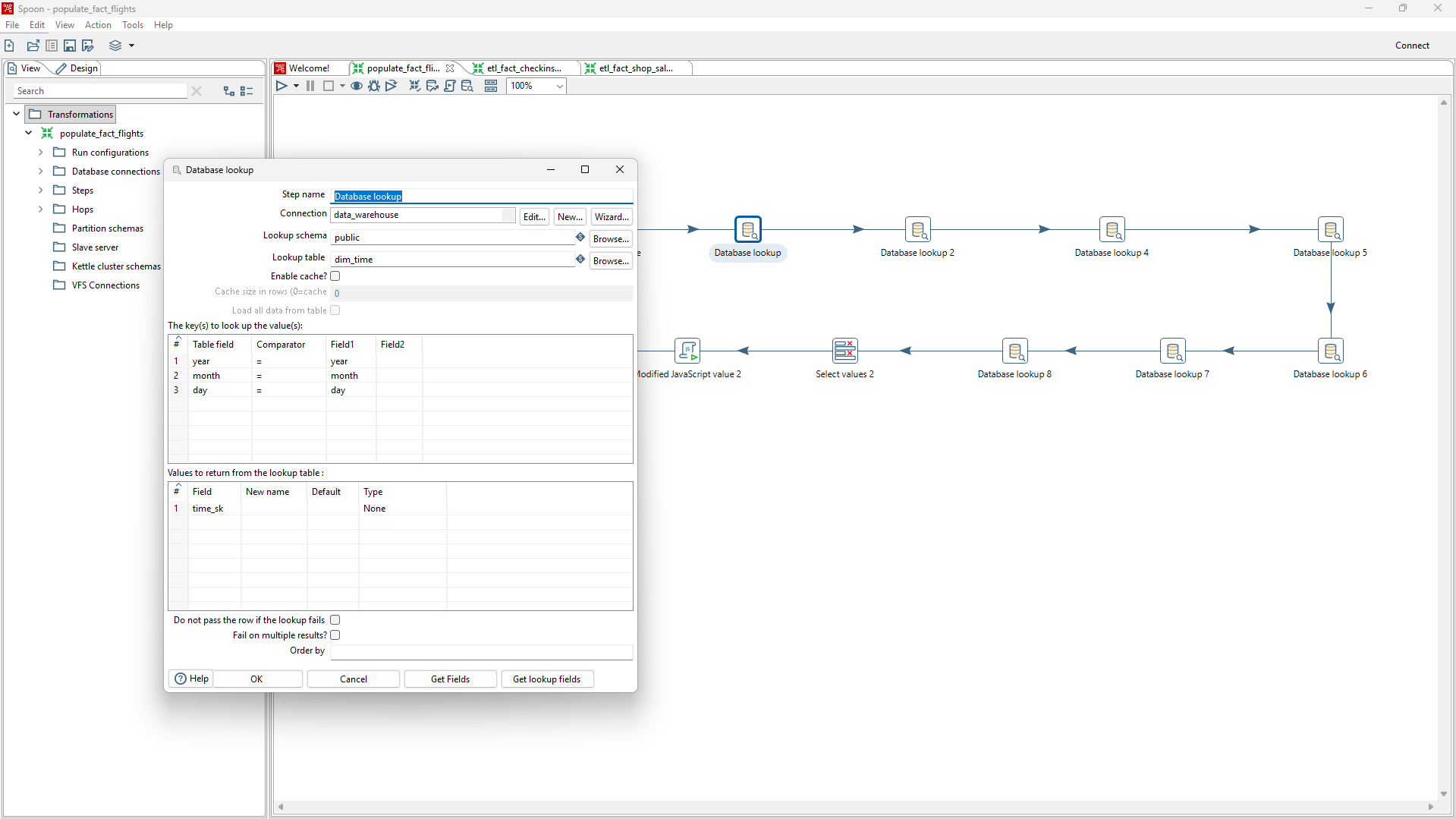
**Fonctionnement général**

1. Lecture des lignes de données : chaque ligne du flux est traitée individuellement.
2. Exécution du script JavaScript : les colonnes sont lues, modifiées ou combinées selon la logique définie.
3. Création ou modification de champs : de nouvelles colonnes peuvent être générées, ou les colonnes existantes modifiées.
4. Transmission de la ligne transformée : au step suivant du flux.

**c) Gestion des clés substituts**

* **Database Lookup** : le mapping et récupération des clés substituts depuis les dimensions
* **Stream Lookup**: la recherche des valeurs dans un autre flux de données

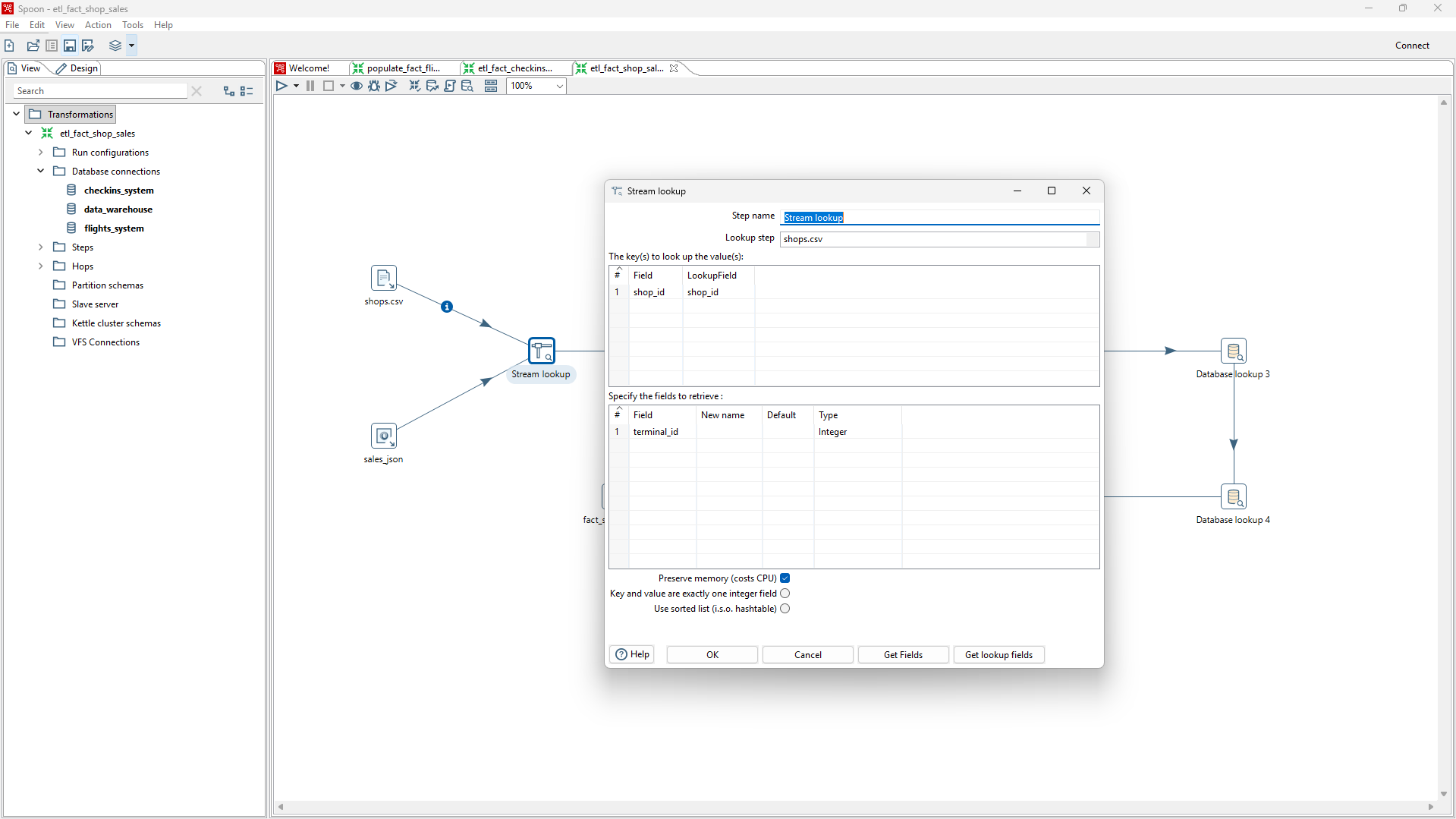
**Exemple de Database Lookup :**



**Fonctionnement général**

1. Lecture d’une ligne du flux principal
2. Utilisation de la ou des clés pour interroger la table de référence
3. Extraction des valeurs correspondantes
4. Ajout des champs récupérés à la ligne du flux
5. Transmission de la ligne enrichie au step suivant

**Exemple de Stream lookup :**



**Fonctionnement général**

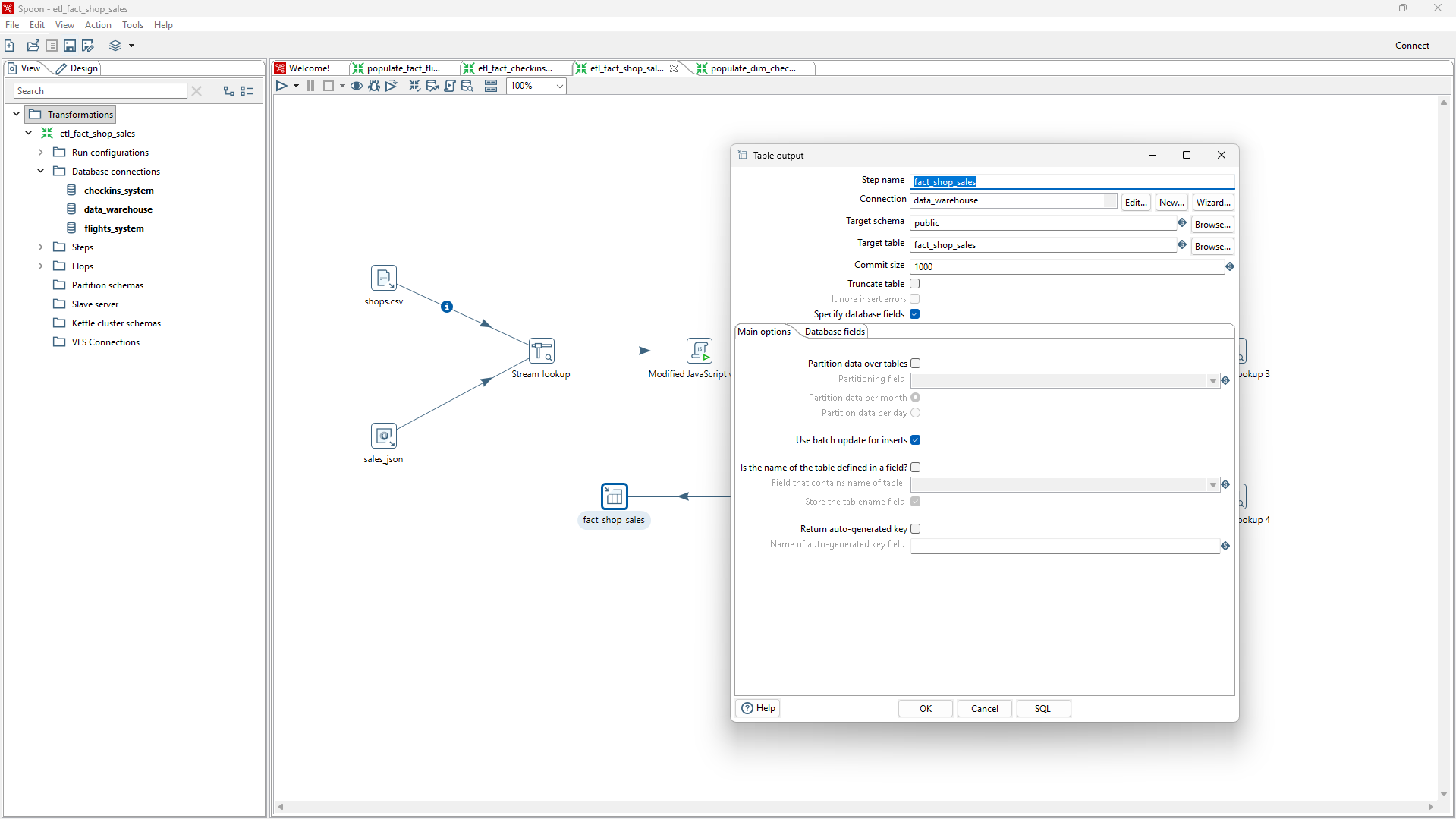
1. Lecture d’une ligne du flux principal
2. Recherche de la correspondance dans le flux secondaire basé sur une ou plusieurs clés
3. Récupération des valeurs correspondantes du flux secondaire
4. Ajout des champs récupérés à la ligne du flux principal
5. Transmission de la ligne enrichie au step suivant

### Étape 4 : Chargement des Tables

Les dimensions sont chargées en priorité à l’aide de :

* **Table Output** :charger le flux de données transformées dans les dimensions/faits

**Exemple de Table Output :**



**Fonctionnement général**

1. Connexion à la base de données via une connexion JDBC préconfigurée
2. Mapping des colonnes du flux vers les colonnes de la table cible
3. Insertion des lignes dans la table
4. Gestion des erreurs et des doublons selon la configuration :

* Ignorer les erreurs
* Rediriger les lignes erronées vers un flux séparé
* Arrêter la transformation en cas d’erreur critique

Les données sont traitées ligne par ligne, mais le step peut être configuré pour effectuer des inserts par lots, améliorant ainsi la performance sur de grands volumes.

## 6.5 Gestion des erreurs et journalisation

PDI permet la mise en place de :

* Logs d’exécution
* Tables de journalisation
* Détection des erreurs de chargement
* Redirection des lignes erronées

Ces mécanismes facilitent le suivi et la maintenance du processus ETL.

## 6.6 Avantages de l’utilisation de PDI

* Interface graphique intuitive
* Séparation claire entre logique métier et technique
* Réutilisabilité des transformations
* Adapté aux projets académiques et professionnels

L’utilisation de Pentaho Data Integration a permis de mettre en œuvre un processus ETL structuré, fiable et évolutif, assurant l’alimentation correcte de l’entrepôt de données et la disponibilité de données de qualité pour le reporting et le data mining.

# 7 Reporting et Tableaux de Bord (Power BI)

## 7.1 Objectif du Reporting Décisionnel

Le reporting décisionnel constitue la phase finale du système décisionnel mis en place dans ce projet.  
Son objectif principal est de transformer les données stockées dans le Data Warehouse en informations claires, synthétiques et visuelles, facilitant ainsi la prise de décision.

À travers des tableaux de bord interactifs, les décideurs de l’aéroport peuvent analyser :

* “Le trafic des passagers”
* “La performance et la ponctualité des vols”
* “Les revenus générés par les boutiques commerciales”

Le reporting agit comme un lien entre les données techniques et les besoins métiers.

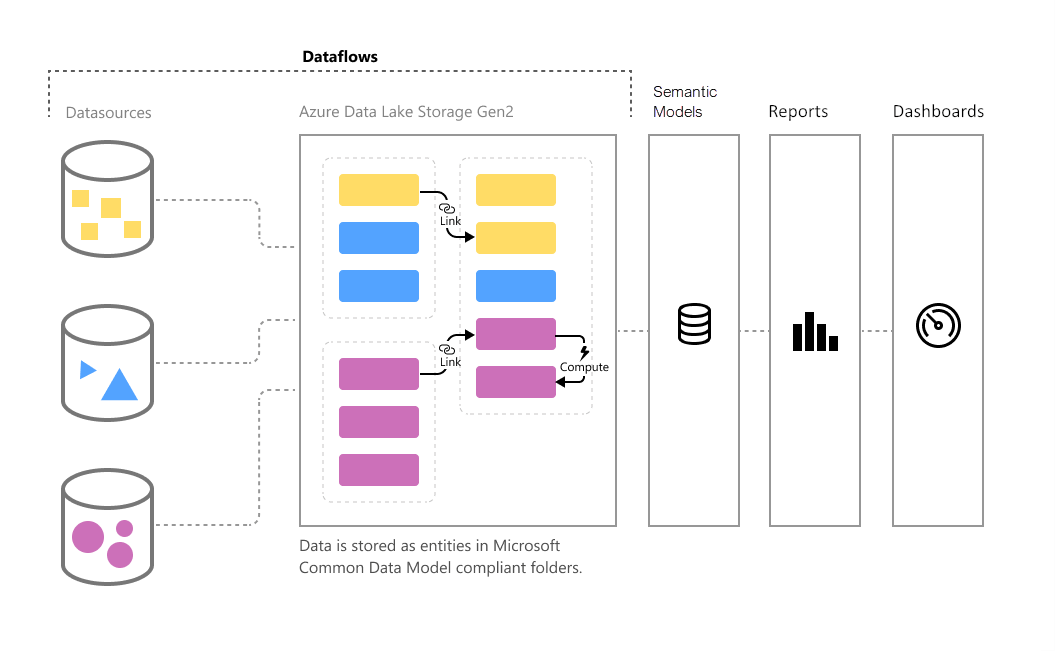


Figure 7.1 : Rôle du reporting dans le système décisionnel

## 7.2 Outil de Business Intelligence Utilisé

L’outil Microsoft Power BI a été choisi pour la réalisation du reporting et des tableaux de bord.

Power BI permet :

* “Une connexion directe au Data Warehouse”
* “La modélisation des données”
* “La création de mesures DAX”
* “La conception de tableaux de bord interactifs”

Cet outil est largement utilisé en milieu professionnel pour l’analyse décisionnelle.



Figure 7.2 : Interface de l’outil Power BI Desktop

## 7.3 Connexion au Data Warehouse

Power BI est connecté directement à la base de données PostgreSQL contenant le Data Warehouse.  
Cette connexion permet d’exploiter les tables de faits et de dimensions sans duplication des données, garantissant ainsi la cohérence et la fiabilité des analyses.

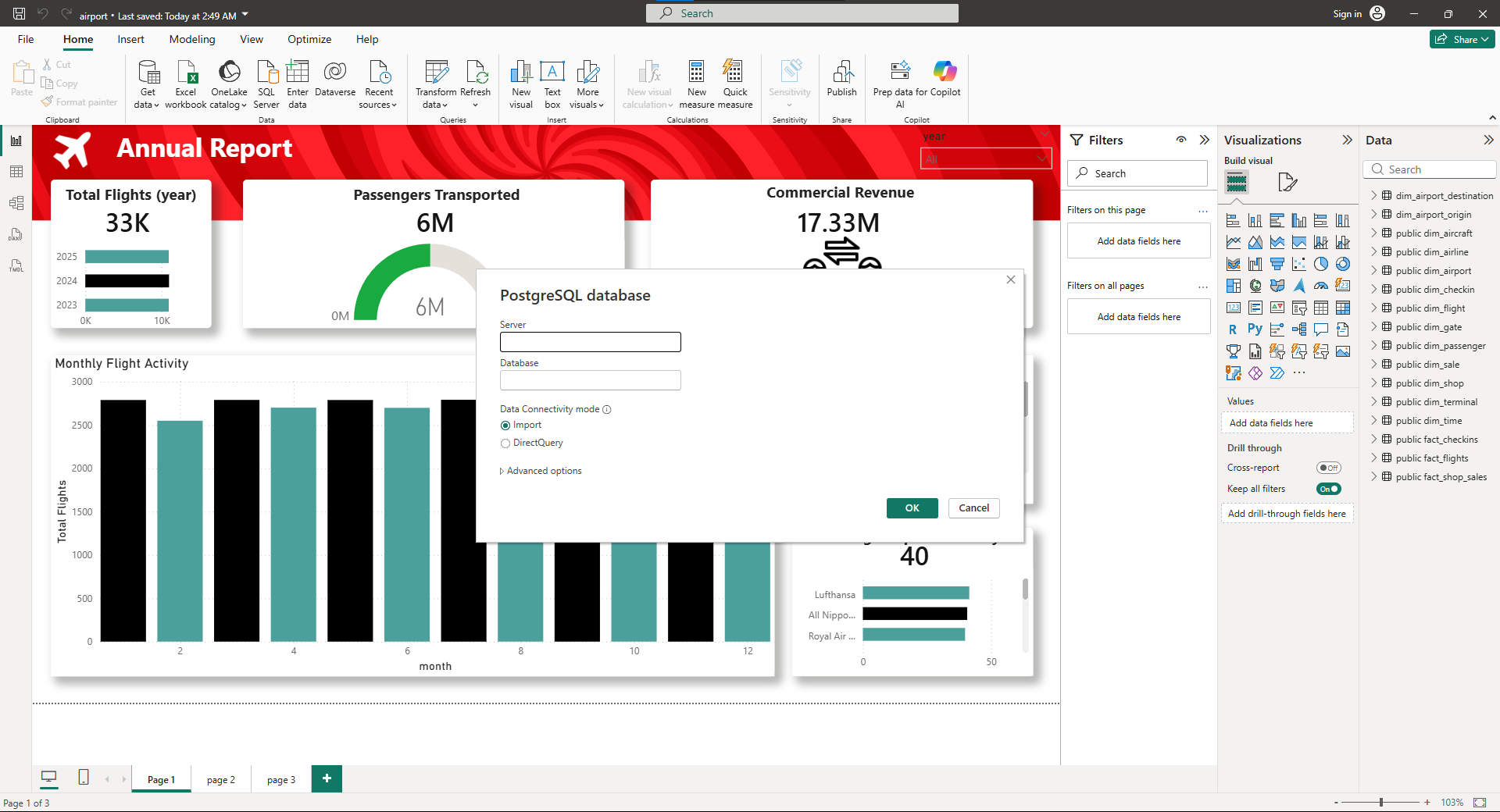


Figure 7.3 : Connexion de Power BI au Data Warehouse PostgreSQL

## 7.4 Modèle de Données dans Power BI

Après l’importation des données, un modèle relationnel a été mis en place dans Power BI.

Ce modèle suit un schéma en étoile, avec :

* Des tables de faits représentant les événements (vols, check-ins, ventes)
* Des tables de dimensions représentant les axes d’analyse (temps, terminal, compagnie, boutique)

Les relations reposent sur des clés substituts afin d’améliorer les performances et la lisibilité du modèle.

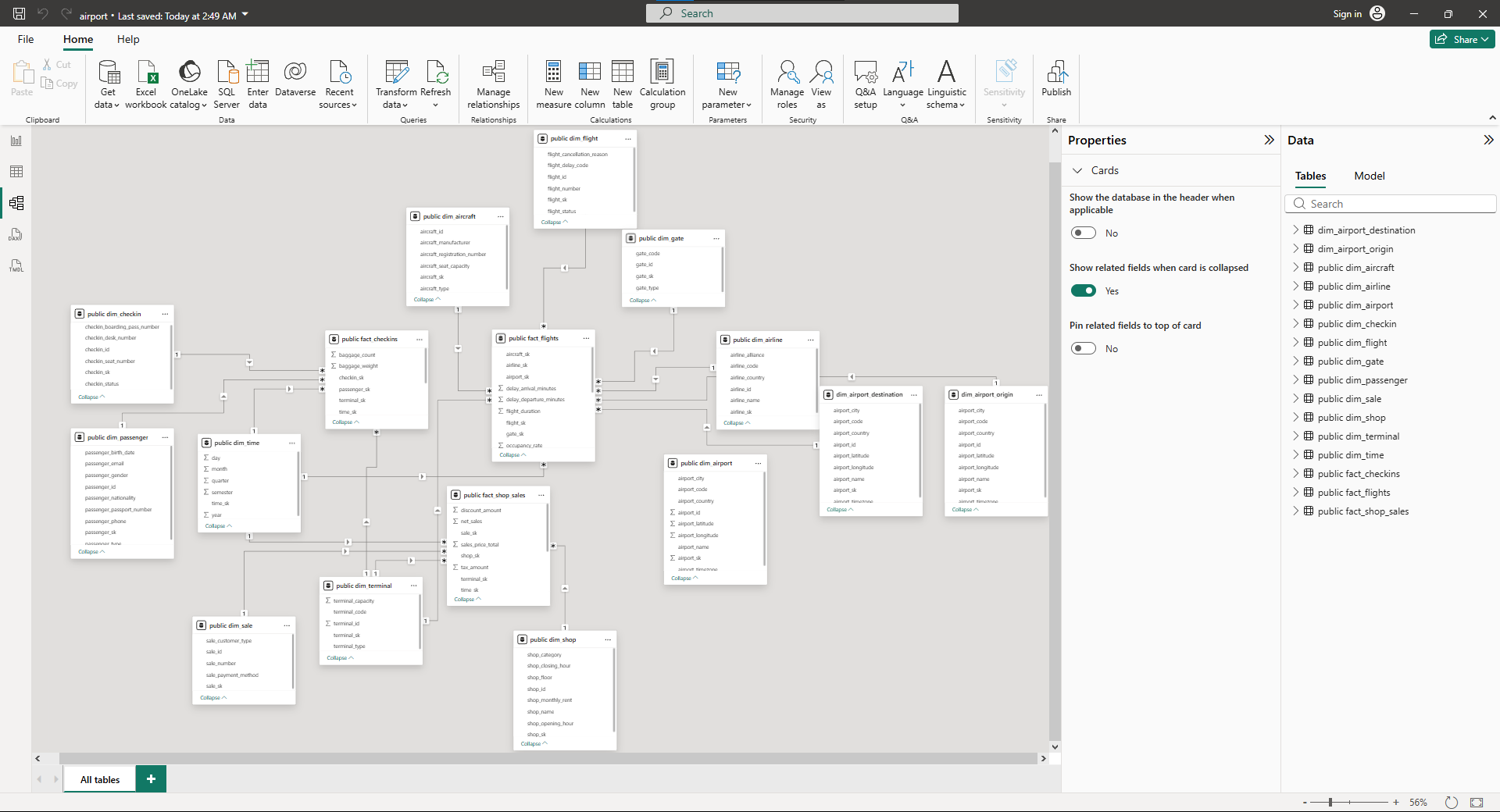


Figure 7.4 : Modèle dimensionnel (schéma en étoile) dans Power BI

## 7.5 Mesures et Calculs DAX

Les indicateurs clés de performance (KPI) sont calculés à l’aide du langage DAX (Data Analysis Expressions).

DAX permet de créer des mesures dynamiques qui s’adaptent aux filtres et aux interactions des tableaux de bord.

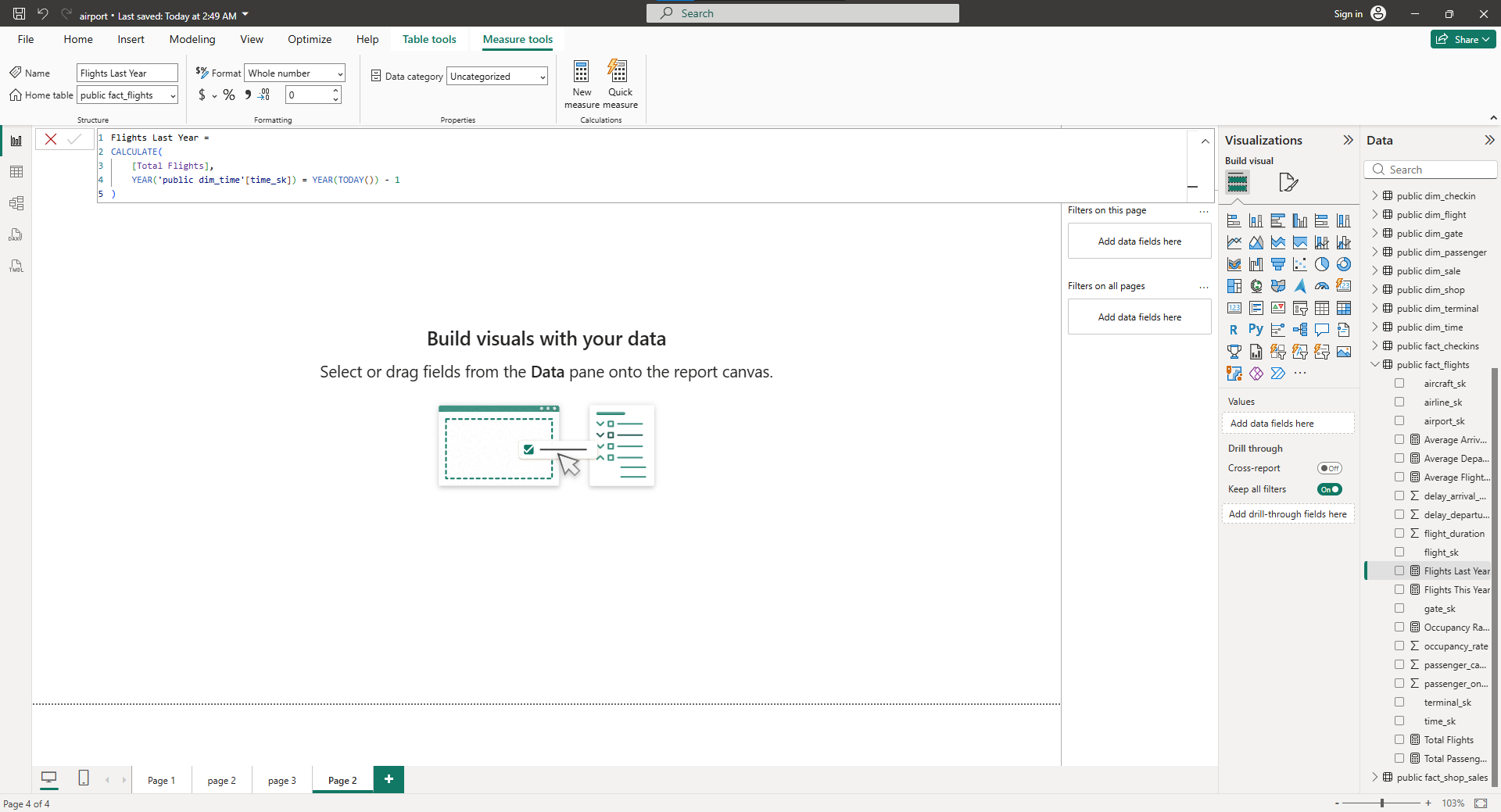


Figure 7.5 : Création de mesures DAX dans Power BI

Total Passengers =

SUM ( fact\_flights[passenger\_onboard] )

Average Departure Delay =

AVERAGE ( fact\_flights[delay\_departure\_minutes] )

Total Revenue =

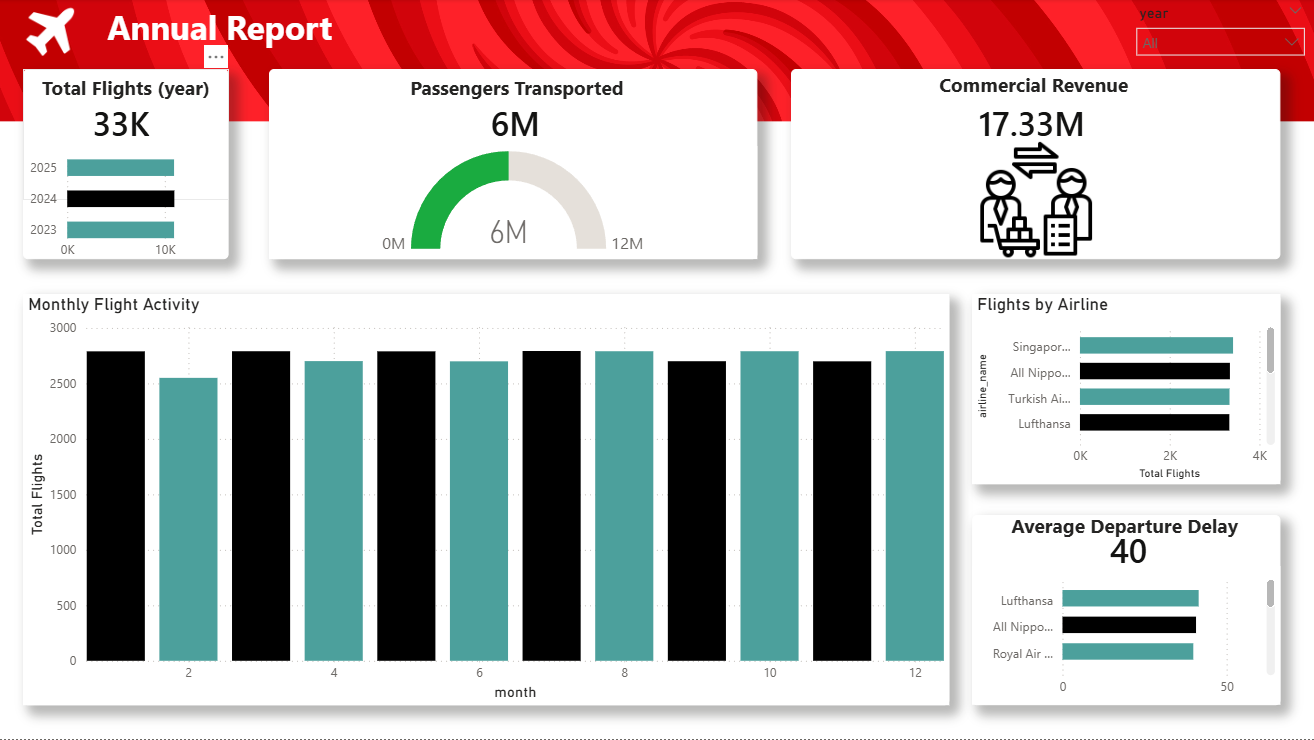
SUM ( fact\_shop\_sales[net\_sales] )

## 7.6 Annual Report Dashboard

Ce tableau de bord annuel fournit une vue globale et synthétique de l’activité de l’aéroport sur une période donnée.

Il est destiné principalement aux décideurs et à la direction, afin de leur offrir une vision consolidée des principaux indicateurs de performance.

Le tableau de bord regroupe des indicateurs clés tels que le volume total de passagers, le nombre de vols, les performances globales ainsi que les tendances annuelles.  
Il permet d’identifier rapidement l’évolution de l’activité aéroportuaire et de comparer les performances d’une année à une autre.



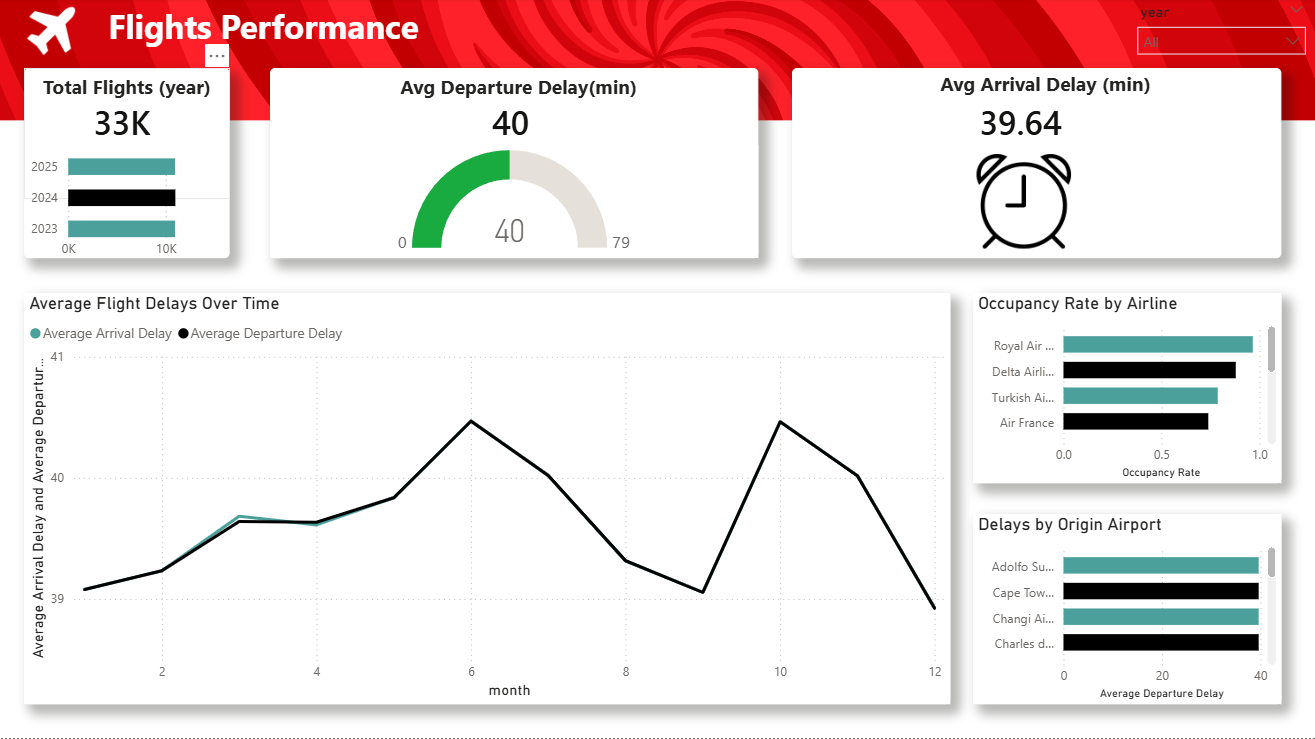
7.6 Global Dashboard

## 7.7 Flights Performance Dashboard

Ce tableau de bord est dédié à l’analyse de la performance opérationnelle des vols.

Il permet d’évaluer la ponctualité des vols, d’identifier les retards et d’analyser leurs causes principales.

Grâce à ce tableau de bord, les responsables opérationnels peuvent suivre les indicateurs de retard par compagnie aérienne, par destination ou par période, et ainsi mettre en place des actions correctives visant à améliorer la qualité du service et la ponctualité des vols.



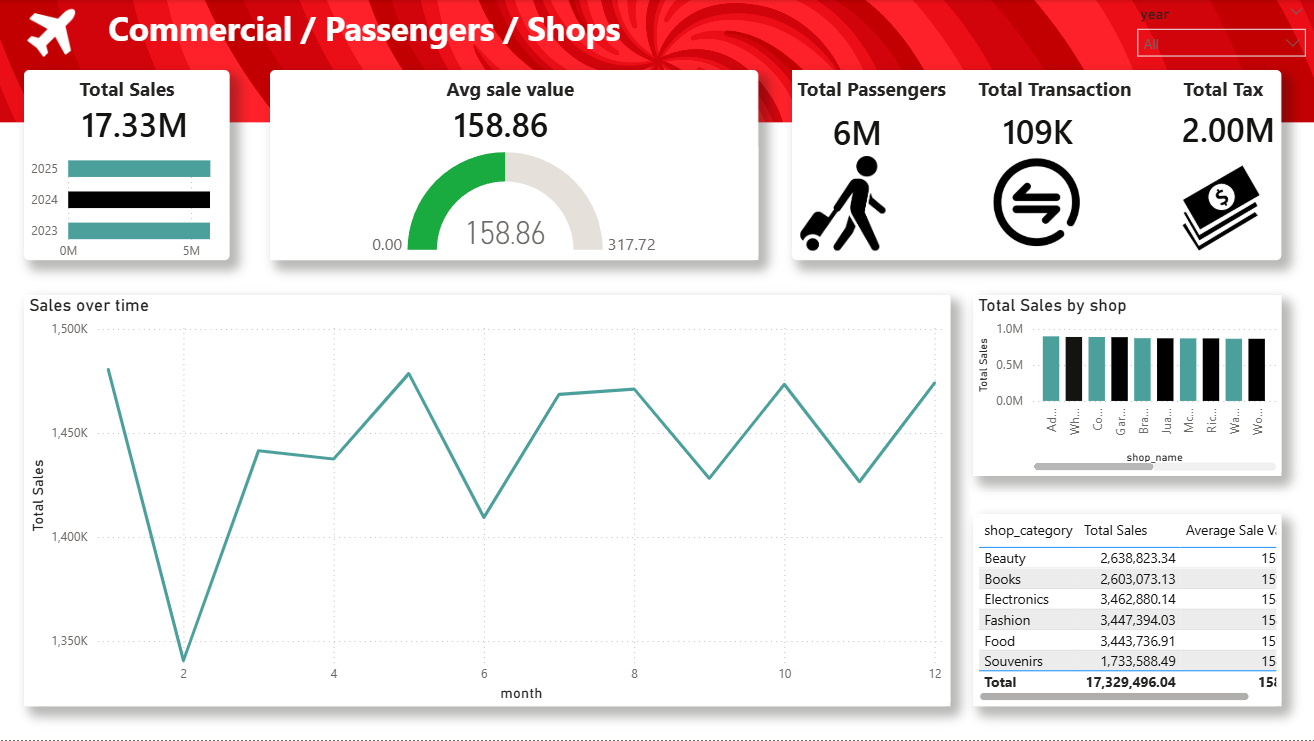
7.7 Flights Performance Dashboard

## 7.8 Commercial / Passenger / Shops Dashboard

Ce tableau de bord regroupe les indicateurs liés aux activités commerciales et au comportement des passagers au sein de l’aéroport.

Il permet d’analyser les performances des boutiques, le chiffre d’affaires généré, ainsi que l’impact du trafic passagers sur les ventes commerciales.

Ce tableau de bord aide les responsables commerciaux à identifier les points de vente les plus performants, les périodes de forte activité et les opportunités d’optimisation de l’offre commerciale et de l’implantation des boutiques.



7.8 Business & Revenue Dashboard

# 8 Analyse avancée par Data Mining

## 8.1 Introduction à la phase de Data Mining

Après la mise en place du Data Warehouse et du processus ETL, une phase de Data Mining a été réalisée afin d’extraire des connaissances à forte valeur ajoutée à partir des données stockées.

Contrairement aux outils de Business Intelligence qui offrent une analyse descriptive (rapports, tableaux de bord), le Data Mining permet d’aller plus loin en identifiant des tendances, des comportements et des regroupements naturels au sein des données.  
Dans ce projet, le Data Mining est appliqué pour analyser la performance des boutiques de l’aéroport à partir des données de ventes.

## 8.2 Objectif du Data Mining dans ce projet

L’objectif principal de cette phase est de :

* Segmenter les boutiques selon leur performance commerciale
* Identifier des profils de boutiques similaires
* Aider à la prise de décision stratégique

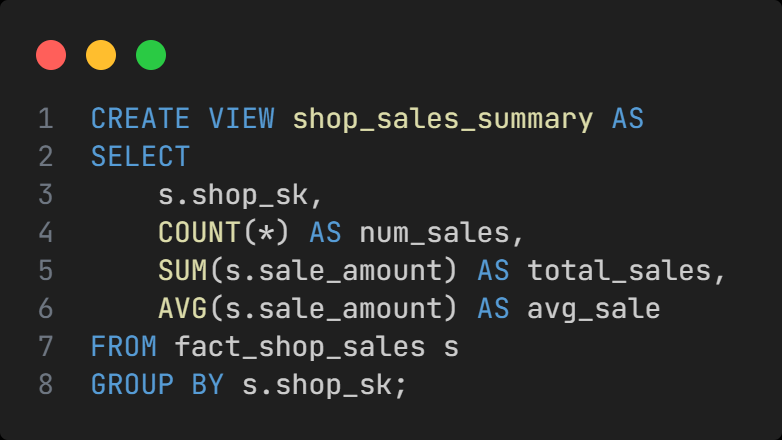
Pour cela, un algorithme de clustering K-Means a été utilisé.

## 8.3 Description des données utilisées

Les données utilisées proviennent du Data Warehouse, à travers la vue shop\_sales\_summary.  
Cette vue contient des données agrégées par boutique, notamment :

* shop\_sk : identifiant de la boutique
* num\_sales : nombre total de ventes
* total\_sales : chiffre d’affaires total
* avg\_sale : montant moyen par vente

Ces indicateurs permettent de décrire le profil de ventes de chaque boutique.

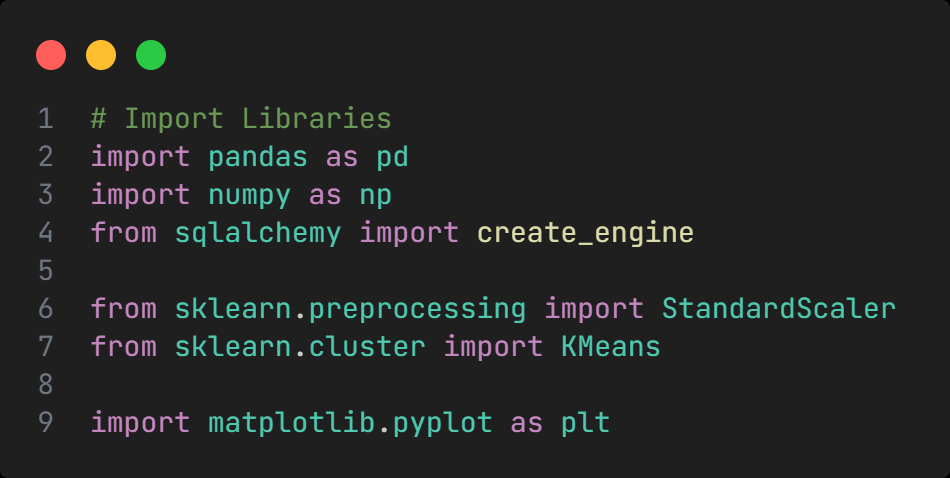


## 8.4 Explication de l’algorithme utilisé

**Bibliothèques utilisées**

Plusieurs bibliothèques Python ont été utilisées afin de réaliser la phase de Data Mining :

* **pandas** : utilisée pour la manipulation et l’analyse des données sous forme de DataFrame.
* **numpy** : utilisée pour les opérations numériques.
* **sqlalchemy** : permet d’établir une connexion entre Python et la base de données du Data Warehouse.
* **scikit-learn** :
  + StandardScaler : pour la normalisation des données.
  + KMeans : pour l’algorithme de clustering.
* **matplotlib** : utilisée pour la visualisation des résultats (graphiques).

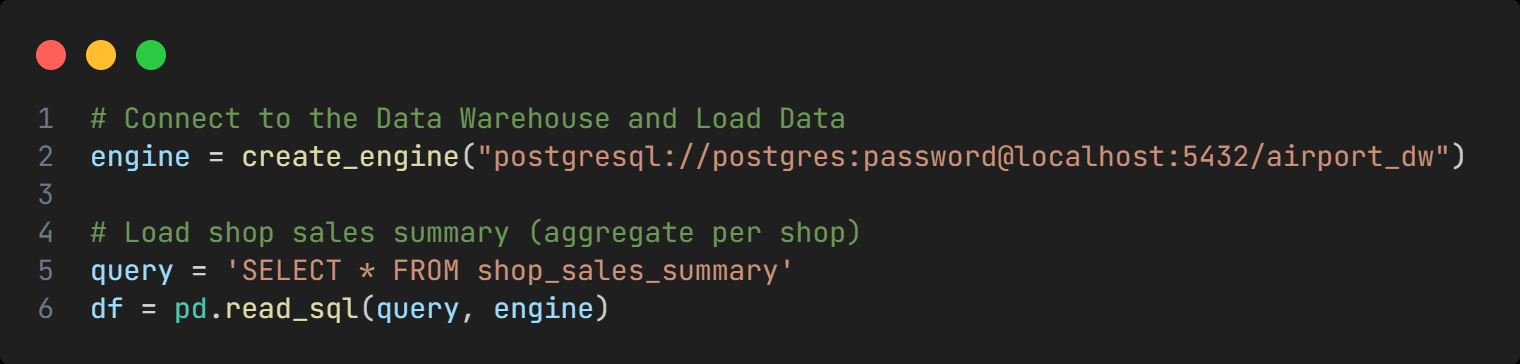
Ces bibliothèques permettent de réaliser une analyse complète allant de l’extraction des données jusqu’à l’interprétation visuelle des résultats.

**Chargement des données depuis le Data Warehouse**

Le script établit une connexion avec la base de données PostgreSQL contenant le Data Warehouse à l’aide de sqlalchemy.

Les données sont extraites depuis la vue shop\_sales\_summary, qui contient des indicateurs agrégés par boutique, puis chargées dans un DataFrame pandas.

Cette étape permet de travailler directement sur des données propres et structurées issues du système décisionnel.



### Partie 1 : Méthode du coude (Elbow method)

**Préparation des données**

Avant d’appliquer la méthode du coude :

* La colonne shop\_sk est supprimée car il s’agit d’un identifiant technique sans valeur analytique.
* Les variables numériques (num\_sales, total\_sales, avg\_sale) sont normalisées à l’aide de StandardScaler.

La normalisation est indispensable pour que toutes les variables aient la même importance dans le calcul des distances.

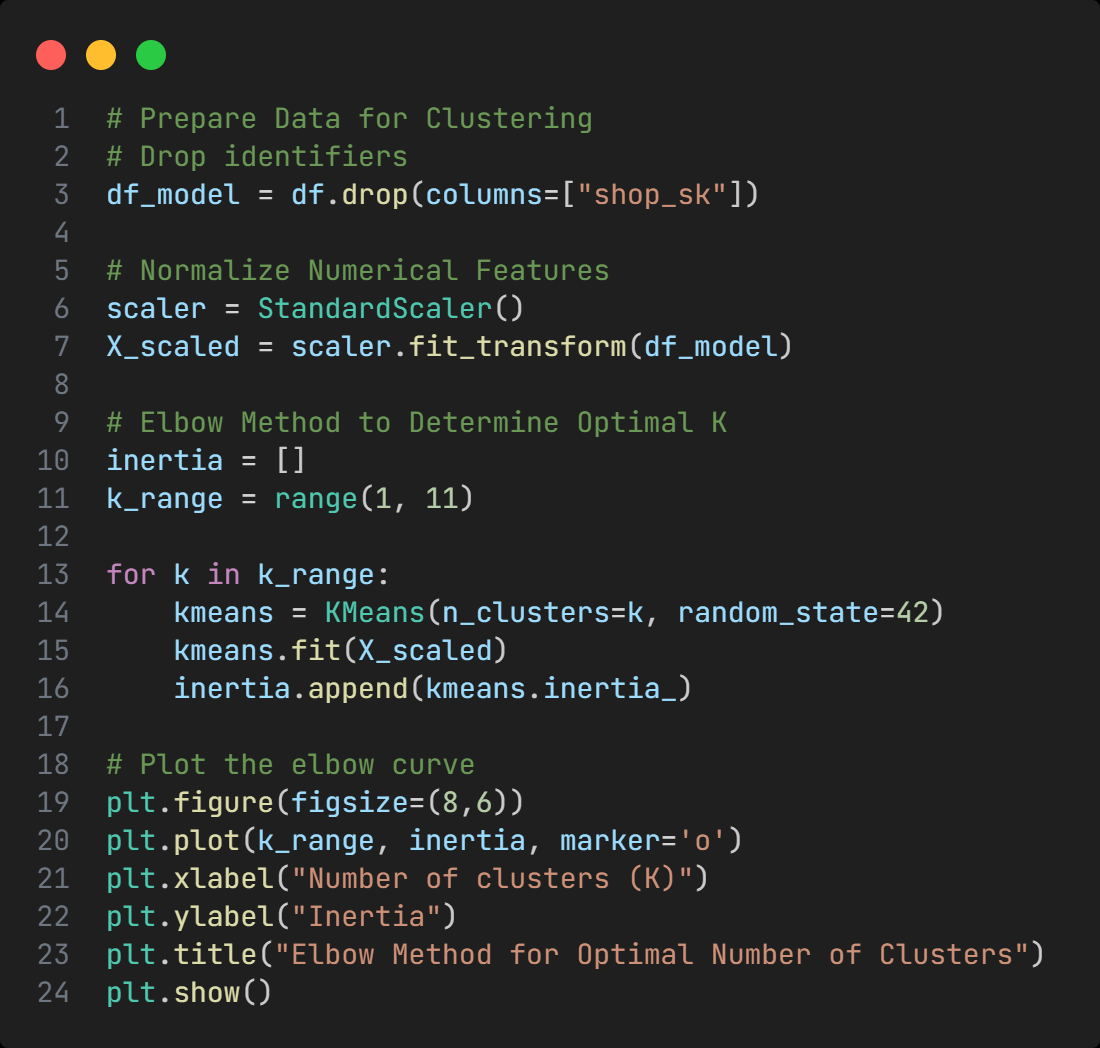
**Principe de la méthode du coude**

La méthode du coude permet de déterminer le nombre optimal de clusters (K).

Principe :

* L’algorithme K-Means est exécuté pour plusieurs valeurs de K.
* Pour chaque K, on calcule l’inertie (somme des distances intra-clusters).
* L’objectif est d’identifier le point où la diminution de l’inertie devient moins significative.

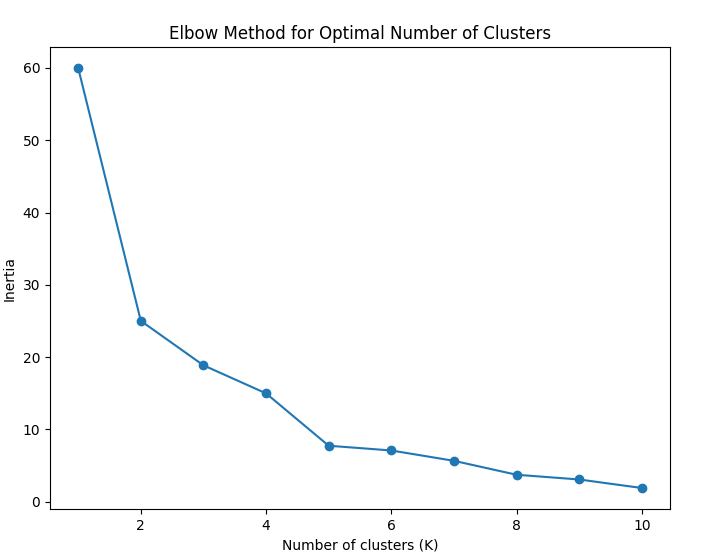
**Implémentation de la méthode du coude**

****

Dans le code :

* Les valeurs de K testées vont de 1 à 10.
* Pour chaque valeur :
  + K-Means est entraîné sur les données normalisées.
  + L’inertie est stockée dans une liste.
* Les résultats sont visualisés à l’aide d’un graphique.

Le graphique obtenu montre un changement notable à K = 2, ce qui correspond au point du coude.

****

### Partie 2 : Algorithme K-Means

**Principe de l’algorithme K-Means**

L’algorithme K-Means est un algorithme de clustering non supervisé qui vise à :

* Regrouper les données en *K* clusters
* Minimiser la distance entre les points d’un même cluster
* Maximiser la séparation entre les clusters

Chaque cluster est représenté par un centroïde, recalculé à chaque itération.

**Application de K-Means dans le projet**

Après avoir déterminé le nombre optimal de clusters :

* K est fixé à 2
* L’algorithme K-Means est appliqué sur les données normalisées
* Chaque boutique est affectée à un cluster

Le résultat est ajouté au DataFrame sous forme d’une nouvelle colonne cluster.



**Résultats du clustering**

Les données sont ensuite regroupées par cluster afin de calculer la moyenne des indicateurs numériques.

Cette étape permet de :

* Comparer les clusters entre eux
* Identifier les caractéristiques principales de chaque groupe

Cluster summary:

num\_sales total\_sales avg\_sale

cluster

0 5529.00 872646.79750 157.836561

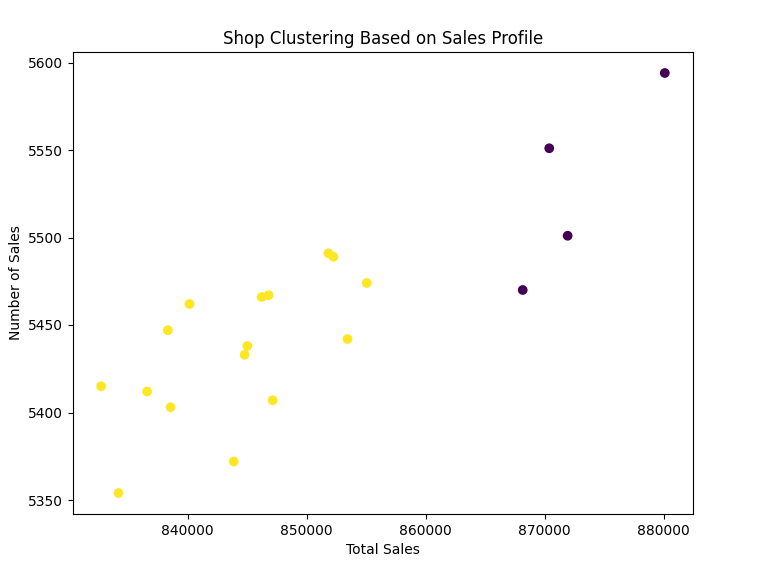
1 5435.75 844187.23125 155.305020

**Visualisation des clusters**

Un graphique de dispersion est utilisé pour représenter :

* L’axe X : chiffre d’affaires total
* L’axe Y : nombre de ventes
* La couleur : le cluster

Cette visualisation confirme la séparation des boutiques selon leur performance commerciale.

****

## 8.5 Analyse et interprétation des résultats

**Résumé des clusters**

Les résultats montrent l’existence de deux clusters distincts :

| **Cluster** | **Nombre de ventes** | **Chiffre d’affaires** | **Vente moyenne** |
| --- | --- | --- | --- |
| 0 | Élevé | Élevé | Élevée |
| 1 | Moyen | Moyen | Moyenne |

**Interprétation des clusters**

**Cluster 0 : Boutiques à forte performance**

* Chiffre d’affaires plus important
* Montant moyen par vente plus élevé
* Forte activité commerciale

Ces boutiques représentent les points de vente stratégiques de l’aéroport.

**Cluster 1 : Boutiques à performance moyenne**

* Résultats légèrement inférieurs
* Potentiel d’amélioration identifié

Ces boutiques peuvent bénéficier :

* d’actions marketing ciblées
* d’une meilleure visibilité
* d’une optimisation de l’offre commerciale

**Rapport du Data Mining à la prise de décision**

Cette analyse permet :

* Une meilleure compréhension des performances commerciales
* Une segmentation claire des boutiques
* Un soutien à la prise de décision stratégique

Le Data Mining complète ainsi efficacement les outils de Business Intelligence traditionnels.

# Conclusion

Ce projet a permis de mettre en œuvre une architecture décisionnelle complète, allant de l’intégration des données jusqu’à l’extraction de connaissances à forte valeur ajoutée.  
À travers la conception et la réalisation d’un Data Warehouse, les données issues de différentes sources ont été centralisées, nettoyées et structurées afin de garantir leur fiabilité et leur cohérence.

Le processus ETL, réalisé à l’aide de Pentaho Data Integration (PDI), a joué un rôle essentiel dans la qualité du système décisionnel. Il a permis l’extraction, la transformation (nettoyage, normalisation, enrichissement) et le chargement des données vers le Data Warehouse, assurant ainsi une base solide pour l’analyse.

Sur cette base décisionnelle, des outils de Business Intelligence ont été utilisés afin de produire des tableaux de bord et des indicateurs clés, offrant une vision globale et synthétique de l’activité.  
Cependant, l’apport principal du projet réside dans la phase de Data Mining, qui a permis d’aller au-delà de l’analyse descriptive.

Grâce à l’application de techniques de clustering (K-Means) sur les données issues du Data Warehouse, il a été possible d’identifier des profils distincts, notamment en matière de performance commerciale des boutiques de l’aéroport. Cette analyse a permis de dégager des tendances, de segmenter les données et de fournir un véritable support à la prise de décision stratégique.

En conclusion, ce projet illustre l’importance de la complémentarité entre Data Warehousing, ETL, Business Intelligence et Data Mining. Il démontre comment des données brutes peuvent être transformées en connaissances exploitables, contribuant ainsi à une meilleure compréhension des activités et à une optimisation des décisions au sein de l’organisation.