MP-i

LIVRE BLANC TECHNIQUE

ETL en 2026 : comprendre, choisir et mettre en œuvre

Patterns, outils et architectures — du script Python au data stack cloud-native, avec un focus sur l'ETL géospatial

Un tour d'horizon complet de l'univers ETL en 2026 : des fondamentaux aux outils open source, cloud, géospatiaux (QGIS, FME, GDAL) et ETL-light. Guide décisionnel et architectures de référence inclus.

Mattieu Pottier

Consultant en transformation digitale — SIG, géomatique, Web, ERP Odoo

MP-i — Mattieu Pottier Indépendant · mp-i.pro

Version 1.1.1 — Avril 2026

SOMMAIRE

Préface

Il y a une scène que j'ai vécue des dizaines de fois, sous des formes différentes. Un géomaticien reçoit chaque lundi un fichier de la base adresse nationale mise à jour. Avant de pouvoir l'utiliser dans son SIG, il passe une heure à renommer les colonnes selon la convention interne, harmoniser les valeurs texte, supprimer les trente colonnes inutiles sur trente-huit, filtrer les adresses de son périmètre, et faire la jointure avec le fichier des IRIS. La semaine suivante, il recommence exactement les mêmes opérations. Un intégrateur Odoo reçoit un export CSV de l'ancien ERP du client, passe deux jours à nettoyer les doublons, à reformater les champs date, à mapper les références produits vers le nouveau référentiel. Un développeur web mapping attend que son collègue lui livre un GeoJSON "propre" avant de pouvoir commencer à coder, parce que la source contient des géométries invalides et des attributs mal normalisés.

Aucun de ces trois protagonistes ne se dit qu'il fait de l'ETL. Et pourtant, c'est exactement ce qu'il fait — extraire une donnée d'une source, la transformer pour qu'elle convienne à son usage, la charger dans sa destination. Extract, Transform, Load. ETL.

Ce document est né de ce constat : l'ETL est partout, mais il est rarement nommé, rarement pensé comme une discipline à part entière, et presque jamais doté des bons outils dans les organisations de taille intermédiaire — PME, collectivités, bureaux d'études, équipes SIG de quelques personnes. On bricole, on scripte, on copie-colle, on espère que ça tient. Et ça tient, jusqu'au jour où ça ne tient plus.

J'ai voulu écrire le document que j'aurais aimé avoir quand j'ai commencé à travailler sérieusement sur ces sujets. Un tour d'horizon honnête de ce qu'est un ETL, de comment le concept a évolué, de ce que les outils font réellement — et de ce qu'ils ne font pas. Sans marketing, sans hype autour du "modern data stack", mais sans nostalgie non plus pour les vieux outils visuels qui tournent depuis vingt ans dans les DSI.

Ce livre blanc couvre à la fois le monde généraliste — Python, dbt, Airbyte, n8n — et le monde géospatial — GDAL, QGIS, FME. Ces deux univers s'ignorent souvent alors qu'ils partagent les mêmes problèmes fondamentaux : une donnée en entrée qui ne convient pas, une donnée en sortie qui doit correspondre à une cible précise, et un pipeline à construire entre les deux. Ce document essaie de faire le pont.

Mattieu Pottier — MP-i · Mattieu Pottier Indépendant

Introduction

La donnée brute n'est jamais prête à l'emploi. Elle arrive dans le mauvais format, dans le mauvais système de coordonnées, avec des champs mal nommés, des doublons, des valeurs nulles là où on attend des entiers. Entre la source et la destination, il y a toujours un travail de préparation — un travail souvent invisible, souvent manuel, souvent répété. Ce travail a un nom : l'ETL.

Pourquoi ce sujet est stratégique en 2026

La data est au cœur de tous les projets SI, SIG et ERP. Mais sa qualité conditionne directement celle des décisions prises à partir d'elle. Un tableau de bord construit sur des données mal transformées est pire qu'une absence de tableau de bord : il donne une illusion de précision là où il y a de l'approximation. Un projet QGIS alimenté par des données dans le mauvais système de projection produit des analyses fausses. Une migration Odoo menée sans pipeline ETL rigoureux se traduit par des données corrompues dans le nouvel ERP dès le premier jour de production.

En 2026, trois tendances renforcent l'importance de maîtriser l'ETL. La première est la montée du cloud : les données vivent désormais dans des entrepôts cloud (Snowflake, BigQuery, Redshift) dont la puissance de calcul change fondamentalement l'approche de la transformation. La deuxième est la généralisation du géospatial dans les SI métier : de plus en plus de projets qui n'étaient pas des projets SIG intègrent une dimension spatiale, ce qui introduit les problèmes spécifiques de l'ETL géospatial dans des équipes qui n'y sont pas préparées. La troisième est la démocratisation des outils open source : dbt, Airbyte, QGIS Processing, GDAL — des outils de qualité professionnelle, gratuits, accessibles à des équipes sans budget data engineering dédié.

Maîtriser l'ETL, c'est reprendre le contrôle sur sa donnée. C'est passer du bricolage à l'infrastructure.

Le périmètre de ce document

Ce livre blanc couvre les patterns ETL/ELT/reverse ETL, les outils généralistes open source et cloud, les outils géospatiaux, et un guide décisionnel pour choisir selon son contexte. Il est volontairement orienté praticien : chaque outil est présenté avec ses forces réelles et ses limites concrètes, pas avec le discours marketing de son éditeur.

Ce document ne couvre pas le data governance et les catalogues de données, les outils de BI et de visualisation (Tableau, Metabase, Grafana), les pipelines streaming temps réel de type Apache Kafka, ni l'architecture des data lakehouses au niveau avancé. Ces sujets mériteraient chacun un document dédié.

Profils de lecture

TAB. 1 — COMMENT LIRE CE DOCUMENT SELON VOTRE PROFIL

PROFIL	PARTIES PRIORITAIRES	OBJECTIF
Géomaticien / chargé SIG	Parties 1, 2 et 4, Partie 5 (Guide décisionnel)	Comprendre pourquoi QGIS n'est pas toujours le bon outil ETL et quand FME ou GDAL s'imposent
Développeur / architecte SI	Parties 2, 3 et 4, Parties 5 et 6 (Guide & Architectures)	Choisir la bonne stack ETL selon la volumétrie, le budget et les compétences disponibles
Intégrateur Odoo / chef de projet ERP	Partie 1, Partie 3 (§ Python, n8n), Partie 5 (Guide décisionnel)	Comprendre comment structurer un pipeline ETL pour une migration ou une intégration ERP
DSI / Décideur technique	Introduction, Parties 5, 6 et 7, Conclusion	Évaluer les options et les coûts avant de décider d'un investissement outil

Partie 1 — Bref historique : comment l'ETL est devenu incontournable

Pour comprendre pourquoi l'ETL s'est imposé comme une discipline à part entière, il faut revenir à l'origine du problème qu'il résout : la fragmentation des données dans les systèmes d'information.

Dans les années 1970 et 1980, les données d'entreprise vivent dans des silos applicatifs étanches. Chaque système — paie, stocks, comptabilité, production — possède sa propre base, son propre format, son propre modèle de données. Lorsqu'une direction veut un rapport consolidé, il faut extraire manuellement les données de chaque silo, les réconcilier à la main et les agréger dans un tableur ou un rapport imprimé. C'est de l'ETL sans le nom, réalisé par des équipes informatiques avec des scripts JCL sur mainframe IBM.

La naissance du concept comme discipline formelle intervient à la fin des années 1980, portée par l'explosion des ERP — SAP, Oracle — et l'émergence des entrepôts de données décisionnels (data warehouses). Les entreprises veulent centraliser leurs données opérationnelles dans un référentiel unique pour piloter leur activité. En 1991, Bill Inmon — considéré comme le père du data warehouse — fonde Prism Solutions et lance le Prism Warehouse Manager, l'un des premiers outils ETL dédiés à l'alimentation des entrepôts de données. Au début des années 1990, Informatica et IBM DataStage s'imposent comme les outils de référence. Ces solutions sont lourdes, coûteuses, réservées aux grandes entreprises disposant d'équipes IT dédiées. Le monde SIG, lui, développe en parallèle ses propres outils de conversion : FME (Safe Software) voit le jour en 1996 pour répondre aux besoins de traduction de formats géospatiaux, initialement entre CAD et GIS.

1970–1990	1991–1996	1996–1999	2010–2014	2016–2026
Batch & silos	**ETL comme discipline**	**ETL géospatial**	**Cloud warehouses**	**ELT & ope source**
JCL IBM · mainframes · données fragmentées	Prism Solutions 1991 · Informatica · DataStage	FME Safe Software 1996 · GDAL Frank Warmerdam 1998	BigQuery 2010 · Redshift 2012 · Snowflake 2014	dbt 2016 · Airbyt 2020 · accor dbt+Fivetran oc 2025

Le tournant décisif arrive avec le cloud. En 2010, Google annonce BigQuery — un entrepôt de données serverless capable de requêter des téraoctets en quelques secondes. Amazon Redshift suit en novembre 2012 (disponibilité générale en février 2013), Snowflake lance sa plateforme publiquement en octobre 2014. Ce changement est fondamental : la puissance de calcul devient élastique et disponible à la demande. Il n'est plus nécessaire de transformer les données avant de les charger, car le warehouse peut absorber les données brutes et les transformer en interne. L'ELT — Extract, Load, Transform — est né, inversant l'ordre des deux dernières étapes de l'ETL classique.

Cette révolution cloud donne naissance à une nouvelle génération d'outils. En mars 2016, Tristan Handy et son équipe chez Fishtown Analytics publient le premier commit de dbt (data build tool) — un outil de transformation SQL-first dans le warehouse, avec version control et tests automatisés. Le "modern data stack" s'impose progressivement comme la référence pour les nouvelles architectures data : Airbyte ou Fivetran pour l'ingestion, Snowflake ou BigQuery pour le stockage, dbt pour la transformation, Apache Airflow pour l'orchestration. En octobre 2025, dbt Labs et Fivetran signent un accord définitif de fusion (clôture attendue mi à fin 2026, sous réserve d'approbation réglementaire), confirmant la consolidation du marché autour de ce paradigme.

Dans le monde géospatial, l'évolution est différente mais converge vers les mêmes problèmes. GDAL, initié par Frank Warmerdam en 1998, devient le socle universel de la conversion de formats géospatiaux — intégré silencieusement dans QGIS, FME, PostGIS, ArcGIS. QGIS développe son cadre Processing au fil des versions 2.x et 3.x, offrant aux géomaticiens un environnement visuel pour enchaîner des traitements. FME, de son côté, élargit progressivement son périmètre au-delà du géospatial pour devenir une plateforme d'intégration tous formats, spatial et non-spatial.

En 2026, les deux mondes se rejoignent : les données géospatiales entrent dans les data stacks cloud via GeoParquet et DuckDB, et les géomaticiens se retrouvent confrontés aux mêmes questions de pipeline, de transformation et d'orchestration que les data engineers.

💡 **Ce que ce préambule nous apprend**

L'ETL n'est pas une technologie récente — c'est une discipline qui a plus de trente ans. Ce qui a changé, c'est l'ordre des étapes (ETL → ELT), les coûts (du mainframe à l'open source), et la démocratisation des outils. Le problème fondamental — la donnée brute n'est jamais prête — est toujours le même.

Partie 2 — Comprendre l'ETL : définitions, patterns et critères de choix

> ***Un ETL n'est pas un outil — c'est une discipline qui en utilise plusieurs. Confondre les deux est la première source d'erreur dans le choix d'une stack data.***
>
> *— Mattieu Pottier, MP-i*

Avant de parler d'outils, il faut poser un cadre conceptuel commun. Le terme "ETL" est souvent utilisé de façon générique pour désigner toute forme d'intégration de données — ce qui crée des confusions réelles dans le choix des outils. Cette partie définit précisément les concepts, distingue les trois patterns principaux et donne les critères de choix qui structureront la lecture des parties suivantes.

2.1 Anatomie d'un ETL : Extract, Transform, Load

Un pipeline ETL est composé de trois étapes distinctes, chacune avec ses propres enjeux techniques.

Extract — l'extraction consiste à se connecter à une ou plusieurs sources de données et à en lire le contenu. La source peut être une base de données relationnelle (PostgreSQL, MySQL, SQL Server), un fichier (CSV, Excel, JSON, Shapefile, GeoPackage), une API REST, un service cloud (S3, Google Sheets), un système ERP (Odoo, SAP) ou un flux temps réel. L'extraction est souvent l'étape la plus délicate : les sources sont rarement conçues pour être lues par un système tiers, les API ont des limites de débit, les bases de données de production ne doivent pas être surchargées, et les formats propriétaires nécessitent des adaptateurs spécifiques.

Transform — la transformation est l'étape centrale, celle qui donne à l'ETL sa valeur réelle. Elle peut être légère — renommer des colonnes, caster des types, formater des dates — ou lourde — jointures entre plusieurs sources, agrégations, déduplication, application de règles métier complexes, reprojection de coordonnées géographiques, calculs d'indicateurs. C'est ici que la logique métier est encodée. C'est aussi ici que les erreurs ont le plus d'impact : une règle de transformation incorrecte se propage silencieusement jusqu'à la destination, corrompant les analyses en aval.

Load — le chargement consiste à écrire les données transformées dans la destination cible. La destination peut être un data warehouse (Snowflake, BigQuery), une base de données métier (PostGIS, PostgreSQL), un fichier (GeoParquet, GeoJSON, CSV), une API ou un outil SaaS. Le chargement doit gérer les conflits : que faire si un enregistrement existe déjà en cible ? Écraser, ignorer, fusionner, historiser ? Ces règles de chargement (upsert, append, truncate-and-load, CDC) sont aussi importantes que les règles de transformation.

Extract

Connexion à la source, lecture des données brutes. La source dicte les contraintes : débit, format, authentification, fraîcheur.

Transform

Nettoyage, normalisation, conversion, enrichissement, jointures. C'est ici que la logique métier est encodée — et que les erreurs ont le plus d'impact.

Load

Écriture dans la destination cible avec gestion des conflits : upsert, append, truncate, CDC. Les règles de chargement sont aussi critiques que les transformations.

2.2 ETL, ELT, reverse ETL — les trois patterns

Ces trois acronymes désignent trois architectures distinctes qui répondent à des contextes différents. Les confondre conduit à choisir le mauvais outil.

L'ETL classique transforme les données avant de les charger dans la destination. L'infrastructure de transformation est externe au warehouse — c'est le moteur ETL lui-même (Talend, Informatica, FME, un script Python) qui porte la charge de calcul. Ce pattern est adapté lorsque les données brutes ne doivent pas être stockées telles quelles — pour des raisons réglementaires notamment (données personnelles, RGPD), ou lorsque la destination est une base opérationnelle à faible tolérance aux données sales. Son principal inconvénient est qu'il nécessite une infrastructure de transformation dédiée, séparée du warehouse.

L'ELT inverse les deux dernières étapes : les données sont chargées brutes dans le warehouse, puis transformées en interne par le moteur de calcul du warehouse lui-même. Ce pattern est devenu dominant pour les nouveaux projets cloud depuis 2014–2016, précisément parce que

les warehouses cloud (Snowflake, BigQuery, Redshift) disposent d'une puissance de calcul SQL élastique et bon marché. Charger du brut est rapide ; transformer dans le warehouse exploite les ressources déjà payées. dbt est l'outil de référence pour la couche T de l'ELT : il transforme les données brutes en modèles SQL versionnés, testés et documentés.

Le reverse ETL est le pattern le plus récent. Il part du warehouse — qui contient des données transformées, enrichies, agrégées — et les réinjecte dans les outils métier opérationnels : CRM (Salesforce, HubSpot), ERP (Odoo), outils marketing, applications internes. Le reverse ETL "active" les données analytiques dans les outils où travaillent les équipes métier. Census est l'outil de référence de ce segment.

TAB. 2 — LES TROIS PATTERNS ETL EN UN COUP D'ŒIL

CRITÈRE	ETL CLASSIQUE	ELT	REVERSE ETL
Ordre des étapes	Extract → Transform → Load	Extract → Load → Transform	Warehouse → Transform → Outil métier
Où se fait la transformation	Moteur externe	Dans le warehouse	Dans le warehouse + outil cible
Infrastructure requise	Moteur ETL dédié	Warehouse cloud suffisant	Warehouse + outil reverse ETL
Adapté pour	Conformité RGPD, bases opérationnelles	Analytics, data warehouse, BI	Activation de la donnée analytique
Outils types	Talend, FME, Informatica, Python	Airbyte + dbt + Snowflake	Census, Hightouch
Tendance 2026	En recul sur les nouveaux projets	Standard de facto	En croissance

À retenir

ETL et ELT ne sont pas des synonymes. En 2026, ELT domine les nouveaux projets sur infrastructure cloud. L'ETL classique reste pertinent pour les contraintes réglementaires et les bases opérationnelles. Le reverse ETL est une troisième discipline à part entière, souvent ignorée.

2.3 Les critères de choix d'un outil ETL

Avant de choisir un outil, quatre questions doivent trouver une réponse claire. Ces questions structurent le guide décisionnel en fin de document.

"01"

Quelle est la volumétrie des données ?

"Moins de 100 000 lignes : un script Python ou n8n suffit. Entre 100 000 et 10 millions : dbt + Airbyte ou FME. Au-delà : infrastructure cloud (AWS Glue, Databricks) ou FME Flow en mode distribué. La volumétrie est le premier filtre — utiliser une solution enterprise sur 10 000 lignes est du sur-dimensionnement."

"02"

Les données ont-elles une composante géospatiale ?

"Si oui, les outils généralistes (dbt, Airbyte, n8n) ne gèrent pas les opérations spatiales — reprojection, intersection, buffer, jointure spatiale. Il faut GDAL, PyQGIS, ou FME. Si la transformation n'implique aucune opération spatiale, QGIS et FME n'apportent rien de plus qu'un script Python."

"03"

Quelles sont les compétences disponibles en interne ?

"No-code / faible technique : n8n, FME Form (visuel), QGIS Graphical Modeler. SQL maîtrisé : dbt est le choix naturel. Python maîtrisé : pandas + SQLAlchemy couvrent la majorité des cas. L'outil le plus puissant est inutile s'il n'est pas adopté par l'équipe."

"04"

Quel est le budget disponible ?

"Zéro budget licence : Python, GDAL, QGIS, dbt Core, Airbyte Community sont entièrement gratuits. Budget SaaS modéré : Airbyte Cloud, dbt Cloud, n8n Cloud. Budget enterprise : FME (pricing sur devis depuis 2023, ordre de grandeur ~1 000–1 500 $/an pour FME Form), Informatica, AWS Glue. Le coût caché à ne pas oublier : le temps de développement et de maintenance."

→ **La règle d'or**

Aucun outil ETL n'est universellement meilleur qu'un autre. Le meilleur outil est celui qui correspond à la volumétrie, aux compétences et au budget de l'équipe — et qui sera réellement maintenu dans le temps.

Partie 3 — Les outils généralistes

Les outils ETL généralistes traitent tout type de données : relationnelle, fichier, API, cloud — sans contrainte spatiale particulière. Ils se répartissent en cinq familles selon leur paradigme, leur niveau technique requis et leur positionnement marché. Cette partie les présente dans un ordre allant du plus bas niveau (Python) au plus abstrait (ETL-light), en passant par les ETL visuels historiques, le modern data stack et les plateformes cloud.

3.1 Python : l'ETL invisible

Python est de loin l'outil ETL le plus répandu en 2026 — souvent sans que ses utilisateurs le désignent comme tel. Un script qui lit un CSV avec pandas, nettoie les données, applique des règles métier et les charge dans une base PostgreSQL via SQLAlchemy est un pipeline ETL fonctionnel. Aucun outil dédié n'est nécessaire pour cela.

Le cœur d'un pipeline Python ETL repose sur quelques bibliothèques fondamentales. **pandas** gère l'ingestion et la transformation de données tabulaires avec une API expressive et bien documentée. **SQLAlchemy** fournit une couche d'abstraction SQL pour les connexions aux bases relationnelles. **Requests** ou **httpx** couvrent l'extraction depuis des APIs REST. Pour les données géospatiales, **Fiona** et **Shapely** complètent le dispositif.

PYTHON

```python
import pandas as pd
from sqlalchemy import create_engine

# Extract
df = pd.read_csv("export_clients.csv", encoding="utf-8")

# Transform
df.columns = df.columns.str.lower().str.replace(" ", "_")
df["date_creation"] = pd.to_datetime(df["date_creation"], dayfirst=True)
df = df.dropna(subset=["email"]).drop_duplicates(subset=["email"])

# Load
engine = create_engine("postgresql://user:pass@localhost/odoo_db")
df.to_sql("res_partner_import", engine, if_exists="replace", index=False)
```

La limite principale du pipeline Python natif est l'absence d'infrastructure : pas de scheduler intégré, pas d'interface de monitoring, pas de gestion robuste des reprises sur erreur. C'est là qu'interviennent les orchestrateurs. **Apache Airflow** (démarré en octobre 2014 chez Airbnb par Maxime Beauchemin, open-sourcé en juin 2015, Apache Top-Level Project depuis janvier 2019) est le standard de facto pour l'orchestration Python. Il définit les pipelines comme des DAGs (Directed Acyclic Graphs) en Python, avec scheduling, monitoring et reprise sur erreur. La version 3.0, sortie en avril 2025, introduit une architecture modulaire et un support natif des workloads ML/AI. **Prefect** est une alternative plus moderne, plus simple à déployer en self-hosted, qui gagne du terrain auprès des équipes qui trouvent Airflow trop lourd.

> 💡 **Python ETL — quand c'est le bon choix**
>
> Python est le bon choix lorsque les transformations sont complexes et spécifiques au métier, que l'équipe maîtrise déjà Python, et que les volumes restent inférieurs à quelques millions de lignes. Au-delà, les coûts de maintenance du code et les problèmes de performance orientent vers des outils dédiés.

3.2 ETL visuels historiques : Talend, Apache Hop, FME

Les ETL visuels proposent une interface graphique pour construire des pipelines par glisser-déposer, sans écrire de code. Ils ont dominé le marché de l'intégration de données des années 2000 aux années 2010.

Talend a été fondé en 2005 à Suresnes par Bertrand Diard et Fabrice Bonan — deux Français qui créent le premier éditeur commercial open source spécialisé en intégration de données. Talend Open Studio est lancé en octobre 2006 : une interface Eclipse qui génère du code Java à partir de pipelines conçus graphiquement. Il connaît une large adoption dans les DSI européennes et les collectivités. En 2023, Talend est racheté par Qlik pour 1,5 milliard de dollars. En janvier 2024, Talend Open Studio est officiellement discontinué. La communauté a depuis forké le projet sous le nom **Talaxie**, qui continue à recevoir des correctifs de sécurité.

Apache Hop (Hop Orchestration Platform) est né fin 2019 comme fork de Kettle/Pentaho Data Integration, lui-même l'un des pionniers des ETL visuels open source (Kettle existe depuis 2002, intégré dans la suite Pentaho). Apache Hop modernise l'architecture de Kettle tout en conservant sa logique de pipeline visuel. Il est aujourd'hui le successeur naturel de Talend Open Studio pour les équipes qui veulent un ETL visuel open source maintenu.

FME (Safe Software, 1996) occupe une position à part dans cette catégorie. Initialement conçu pour la traduction de formats géospatiaux CAD → GIS, FME est aujourd'hui une plateforme d'intégration "All-Data, Any-AI" qui traite aussi bien des fichiers JSON, des bases relationnelles, des APIs REST, des flux BIM et des données spatiales. Il est présenté en détail dans la Partie 4 pour son excellence géospatiale, mais il appartient également à cette catégorie d'ETL visuels enterprise — avec 800+ transformers out-of-the-box, un environnement d'authoring (FME Form) et un serveur d'automatisation (FME Flow).

COMMUNAUTAIRE

Talaxie (ex-Talend OS)

2006 (fork 2024) · Java/Eclipse

Successeur communautaire de Talend Open Studio. Interface visuelle, génération de code Java. Bon pour les migrations depuis des projets Talend existants.

Legacy · Migration

OPEN SOURCE

Apache Hop

2019 · Java

Fork modernisé de Kettle/PDI. Pipeline visuel, orchestration, nombreux connecteurs. Alternative sérieuse à Talend pour les ETL visuels open source.

Visuel · Orchestration

COMMERCIAL

FME Form

1996 · Safe Software

ETL visuel enterprise, spatial et non-spatial. 800+ transformers. Référence pour les projets géospatiaux complexes. Pricing sur devis.

Géospatial · Enterprise

3.3 Modern data stack : dbt, Airbyte, Meltano

Le "modern data stack" désigne l'architecture ELT qui s'est imposée comme référence pour les nouveaux projets data depuis 2018–2020. Sa logique : séparer les responsabilités — un outil pour l'ingestion, un warehouse pour le stockage et le calcul, un outil pour la transformation.

dbt (data build tool) est l'outil de transformation SQL-first de référence. Son premier commit date de mars 2016, issu de Fishtown Analytics (rebaptisée dbt Labs en 2021). dbt ne fait que la transformation — le "T" de l'ELT. Il permet d'écrire des transformations SQL sous forme de modèles versionnés, testés et documentés, avec gestion des dépendances via un DAG implicite. Le 13 octobre 2025, dbt Labs et Fivetran ont signé un accord définitif de fusion en all-stock (clôture attendue mi à fin 2026), créant une entité combinée d'environ 600 M$ d'ARR — signal d'une consolidation majeure du modern data stack.

Airbyte est la plateforme d'ingestion ELT open source de référence. Fondé en janvier 2020 par Michel Tricot et John Lafleur, Airbyte se positionne sur la résolution du problème des connecteurs "longue traîne" : les 80 % de sources que les outils propriétaires n'adressent pas parce qu'elles sont trop peu répandues. En 2026, Airbyte propose plus de 600 connecteurs pré-construits, un SDK Python pour créer des connecteurs personnalisés, et supporte le CDC (Change Data Capture) pour la réplication incrémentale. Disponible en self-hosted (open source) ou en cloud managé.

Meltano est un orchestrateur open source construit autour du standard Singer (protocole de connecteurs créé par Stitch). Il permet de composer des pipelines ELT à partir de taps (sources) et targets (destinations) Singer existants, avec gestion des configurations en YAML et CI/CD intégré. Plus orienté DevOps que les deux précédents.

Airbyte	Ingestion (Extract + Load) — 600+ connecteurs, CDC, self-hosted ou cloud	`Gratuit (self-hosted)` `Cloud sur devis`
dbt Core	Transformation SQL-first dans le warehouse — modèles, tests, documentation, DAG	`Gratuit (open source)`
Apache Airflow	Orchestration — scheduling, DAGs, monitoring, reprise sur erreur	`Gratuit (open source)` `Astronomer Cloud sur devis`
Snowflake / BigQuery / PostgreSQL	Warehouse — stockage et calcul de transformation	`0 € (PostgreSQL self-hosted)` `Plusieurs centaines $/mois (cloud)`

💡 **La stack ELT de référence en 2026**

Pour un nouveau projet data sans contrainte géospatiale : Airbyte (ingestion) + PostgreSQL ou Snowflake (warehouse) + dbt (transformation) + Airflow (orchestration). Chaque composant est remplaçable indépendamment. C'est la force du paradigme moderne par rapport aux ETL monolithiques.

FORCES

- Modularité et remplaçabilité de chaque brique
- Open source dominant (dbt Core, Airbyte, Airflow)
- Communauté et écosystème massifs
- Standards portables (SQL, YAML, DAG)

FAIBLESSES

- Complexité d'orchestration multi-outils
- Compétences SQL + Python + DevOps requises
- Coût cloud élastique difficile à prévoir
- Maintenance répartie sur plusieurs projets

OPPORTUNITÉS

- Convergence avec le géospatial via DuckDB et GeoParquet
- Reverse ETL pour réinjecter dans Odoo et CRM
- Intégration IA/ML native dans Airflow 3.0
- Standards Iceberg pour la portabilité warehouse

MENACES

- Consolidation du marché (fusion dbt+Fivetran)
- Risque de fermeture progressive du tooling open source
- Dépendance aux warehouses cloud propriétaires
- Augmentation des coûts post-fusion à anticiper

3.4 Cloud et SaaS : Fivetran, AWS Glue, Azure Data Factory

Les plateformes cloud d'intégration offrent des pipelines entièrement managés, sans infrastructure à maintenir. Elles correspondent aux organisations qui veulent de la fiabilité clé-en-main au prix d'un coût récurrent et d'une dépendance au fournisseur.

Fivetran est le leader des connecteurs ELT managés. Il automatise l'extraction et le chargement depuis des centaines de sources (SaaS, bases de données, APIs) vers les warehouses cloud, avec CDC natif et zéro maintenance des connecteurs. En octobre 2025, Fivetran a signé un accord définitif de fusion avec dbt Labs (clôture attendue mi à fin 2026, sous réserve d'approbation réglementaire) — une consolidation qui rapprocherait les deux étapes E+L et T dans une seule offre. Pricing basé sur le volume de lignes synchronisées, ce qui peut devenir coûteux à grande échelle.

AWS Glue est un service ETL serverless sur AWS. Il utilise Apache Spark sous le capot pour les transformations à grande échelle, avec une intégration native aux services AWS (S3, Redshift, Athena, RDS). Il supporte Python et Scala. Son principal atout est l'absence d'infrastructure à gérer ; son principal défaut est une complexité de configuration et un coût qui peut surprendre sur les workloads intensifs.

Azure Data Factory est l'équivalent Microsoft, optimisé pour l'écosystème Azure. Il excelle dans l'orchestration et le mouvement de données entre systèmes hétérogènes dans un contexte Microsoft (SQL Server, Dynamics, SharePoint, Azure Synapse). Pour les transformations lourdes, il est généralement combiné avec Azure Synapse Analytics ou Databricks.

> ⚠️ **Le coût caché du SaaS ETL**
>
> Les plateformes cloud ETL facturent à la ligne synchronisée, à l'heure de compute ou au connecteur. Pour des volumes faibles à moyens, le coût est prévisible. Pour des ingestions fréquentes sur de gros volumes, la facture peut décupler rapidement. Toujours modéliser le TCO (coût total) avant de choisir une plateforme SaaS.

3.5 ETL-light : n8n, Make

n8n et Make (ex-Integromat) occupent un segment distinct : l'automatisation légère de flux de données, orientée déclenchement événementiel. Ils ne sont pas des ETL au sens strict — ils n'ont pas de gestion de lineage, de reprise robuste sur erreur, ni de capacité de transformation lourde. Mais pour les équipes PME avec des volumes limités et des besoins d'intégration simples, ils remplissent efficacement le rôle d'ETL-light.

n8n est open source (self-hostable) et propose une interface visuelle pour enchaîner des blocs de traitement. Il supporte des transformations en JavaScript, des appels d'API, des connexions aux bases de données et des déclenchements webhook. Sa version cloud est disponible en SaaS. Il est particulièrement adapté pour des pipelines Odoo → tableur, API tierce → PostgreSQL, ou formulaire web → base de données — dès lors que les volumes restent inférieurs à quelques dizaines de milliers de lignes et que la fiabilité absolue n'est pas critique.

Make est le concurrent SaaS de n8n, avec un catalogue de connecteurs prêts à l'emploi plus large mais sans option self-hosted.

N8n n'est pas un ETL — ne pas le confondre

Utiliser n8n pour migrer 500 000 clients depuis un ancien ERP vers Odoo est une erreur de conception. Sans gestion de reprise sur erreur et sans lineage, un pipeline n8n qui échoue au milieu d'une migration laisse les données dans un état incohérent. Pour les migrations critiques ou les volumes > 100K lignes, utiliser Python + SQLAlchemy ou un ETL dédié.

Résumé Partie 3

Cinq familles d'outils, cinq positionnements distincts : Python pour la flexibilité maximale, les ETL visuels pour les équipes non-développeurs sur des transformations complexes, le modern data stack (dbt + Airbyte) pour les architectures ELT cloud-native, les plateformes SaaS pour la fiabilité clé-en-main, et les outils ETL-light pour les intégrations légères en PME. Le choix se fait sur la volumétrie, les compétences et le budget — pas sur la réputation des outils.

Partie 4 — ETL géospatial : spécificités et outils

L'ETL géospatial partage les mêmes fondamentaux que l'ETL généraliste — extraire, transformer, charger — mais ajoute trois dimensions de complexité que les outils généralistes ne gèrent pas nativement : les systèmes de référence de coordonnées, les formats spécifiques, et les opérations de transformation spatiale. Cette partie explore ces spécificités et présente les outils dédiés.

4.1 Pourquoi la donnée géospatiale pose des problèmes spécifiques

Une donnée géospatiale n'est pas qu'un tableau avec des colonnes de coordonnées. Elle porte avec elle un système de référence de coordonnées (CRS) qui détermine comment interpréter ces coordonnées. Le Lambert 93 (EPSG:2154) est le système de référence légal français pour les données terrestres. Le WGS84 (EPSG:4326) est le système des GPS et des APIs cartographiques web. Servir des données Lambert 93 à une application qui attend du WGS84 produit des géométries décalées de plusieurs centaines de mètres, parfois de plusieurs kilomètres selon la région. Ce n'est pas une petite erreur — c'est une erreur qui invalide toute analyse spatiale.

Au problème de projection s'ajoutent la multiplicité des formats (Shapefile, GeoJSON, GeoPackage, GeoParquet, GeoTIFF, DXF, KML, WKT, WKB…) et les opérations de transformation proprement spatiales. Une jointure spatiale — "attribuer à chaque point la zone administrative dans laquelle il se trouve" — n'existe pas dans pandas ou dans dbt. Elle nécessite un index spatial et des fonctions géométriques. C'est pourquoi les ETL généralistes sont insuffisants dès que la transformation implique une opération sur la géométrie elle-même.

"CRS"
Systèmes de référence — des milliers d'EPSG, une erreur de projection invalide tout

"200+"
Formats supportés par GDAL — chaque format porte ses propres contraintes

"800+"
Opérations spatiales dans PostGIS — intersection, buffer, jointure, interpolation…

4.2 GDAL/OGR : le socle universel

GDAL (Geospatial Data Abstraction Library) est la bibliothèque fondatrice de l'interopérabilité géospatiale open source. Créée en 1998 par Frank Warmerdam, elle fournit une interface unifiée pour lire et écrire plus de 200 formats de données géospatiales raster et vecteur. GDAL est la couche basse sur laquelle s'appuient QGIS, FME, PostGIS, GRASS GIS, et presque tous les outils géospatiaux open source. La version GDAL 2.0 fusionne la bibliothèque raster GDAL et la bibliothèque vecteur OGR en un seul projet unifié.

`ogr2ogr` est l'outil CLI de référence pour les conversions et reprojections en ligne de commande. Il permet de convertir un Shapefile en GeoPackage, de reprojeter de Lambert 93 vers WGS84, de filtrer par attribut, de découper par emprise géographique — le tout en une seule commande.

BASH

```
# Convertir un Shapefile Lambert 93 en GeoPackage WGS84
ogr2ogr \
  -f GPKG communes_wgs84.gpkg \
  communes_l93.shp \
  -s_srs EPSG:2154 \
  -t_srs EPSG:4326

# Filtrer les communes de plus de 10 000 habitants
ogr2ogr \
  -f GeoJSON grandes_communes.geojson \
  communes_wgs84.gpkg \
  -where "population > 10000"
```

Pour les pipelines Python, **Fiona** expose l'API OGR en Python natif, et **Rasterio** fait de même pour le raster. **GeoPandas** combine Fiona, Shapely et pandas pour offrir un DataFrame géospatial manipulable avec la syntaxe pandas habituelle. Ces trois bibliothèques sont les briques de base de tout pipeline ETL géospatial en Python.

💡 **GDAL — le composant invisible**

Si QGIS ouvre un fichier, c'est GDAL qui le lit. Si FME exporte en GeoJSON, c'est GDAL qui l'écrit. Si rasterio lit un GeoTIFF en Python, c'est GDAL sous le capot. Maîtriser ogr2ogr directement, c'est accéder à la couche la plus puissante et la plus portable de l'écosystème géospatial — sans dépendance à aucun outil graphique.

4.3 QGIS comme ETL : Processing Toolbox, Graphical Modeler, PyQGIS

QGIS dispose d'un cadre de traitement (Processing Framework) qui lui confère de réelles capacités ETL géospatiales. Ce cadre expose quatre modes d'usage complémentaires.

La **Processing Toolbox** donne accès à plus de 400 algorithmes natifs QGIS, auxquels s'ajoutent les algorithmes des providers tiers (GRASS GIS, SAGA GIS, GDAL, R). Chaque algorithme peut être exécuté en mode interactif ou en mode batch — ce dernier permettant d'appliquer le même traitement à N datasets successivement, sans relancer l'algorithme manuellement.

Le **Graphical Modeler** permet de chaîner des algorithmes Processing en un workflow visuel, sans écrire de code. On définit les entrées, les sorties, et les enchaînements entre algorithmes. Le modèle est sauvegardé et rejouable — c'est le niveau d'automatisation ETL le plus accessible pour un géomaticien non-développeur.

`qgis_process` est l'interface CLI de Processing. Elle permet d'exécuter un algorithme ou un modèle Graphical Modeler depuis la ligne de commande, sans ouvrir l'interface graphique QGIS. C'est la brique qui permet l'intégration dans un pipeline automatisé — cron, script shell, CI/CD.

```bash
# Reprojection en ligne de commande avec qgis_process
qgis_process run native:reprojectlayer \
  -- INPUT=communes.gpkg \
  TARGET_CRS=EPSG:4326 \
  OUTPUT=communes_wgs84.gpkg
```

PyQGIS est l'API Python complète de QGIS. Elle permet de charger des couches, d'appeler des algorithmes Processing, de lire et modifier des attributs, et d'exécuter des transformations complexes avec la pleine puissance de Python. Les scripts PyQGIS peuvent fonctionner en mode standalone — sans interface graphique — ce qui les rend intégrables dans des pipelines ETL automatisés.

1 Processing Toolbox

- ✓ Interface graphique, 400+ algorithmes
- ✓ Batch processing natif
- ✓ Accessible sans compétences Python

2 Graphical Modeler

- ✓ Workflow visuel sans code
- ✓ Sauvegardable et rejoué
- ✓ Idéal pour les géomaticiens non-développeurs

3 qgis_process CLI

- ✓ Exécution hors interface graphique
- ✓ Intégrable dans cron / scripts shell
- ✓ Pont entre QGIS et l'automatisation système

4 PyQGIS

- ✓ API Python complète
- ✓ Mode standalone possible
- ✓ Transformations complexes et logique métier

4.4 QGIS hors géospatial : fausse bonne idée

QGIS peut techniquement ingérer un fichier CSV sans géométrie, le transformer via le Calculateur de champ, et l'exporter dans un autre format. Il est donc théoriquement utilisable comme ETL généraliste. En pratique, c'est une fausse bonne idée — pour des raisons structurelles.

QGIS n'a pas de scheduler natif. Pour automatiser l'exécution d'un pipeline QGIS sans interaction humaine, il faut passer par `qgis_process` en CLI et un outil externe (cron, Airflow) — ce qui sort du périmètre de QGIS et réintroduit une dépendance à d'autres outils. QGIS ne dispose pas de gestion structurée des erreurs dans les pipelines Processing : si un algorithme échoue, le pipeline s'arrête sans mécanisme de reprise automatique. Les logs sont visibles dans l'interface mais ne sont pas structurés pour une exploitation programmatique. Enfin, QGIS n'a aucun connecteur natif vers des sources non-géospatiales : APIs REST, bases NoSQL, fichiers Parquet non-spatiaux, services cloud — tout cela nécessite du code Python ou un outil dédié.

La règle est simple et sans exception : **QGIS comme ETL est pertinent uniquement lorsque la transformation implique une opération spatiale**. Reprojection, buffer, intersection, jointure spatiale, calcul de surface, découpage par emprise — ces opérations justifient QGIS. Nettoyer un CSV de clients, dédupliquer une liste de prospects, transformer des données Odoo — ce n'est pas le bon outil. Un script Python de 20 lignes avec pandas fera le travail plus proprement, plus vite, et sera maintenable sans QGIS installé sur la machine d'exécution.

> ⚠️ **La règle QGIS ETL**
>
> Utiliser QGIS dans un pipeline ETL uniquement si au moins une étape de transformation implique une opération spatiale. Si toutes les transformations sont attributaires (renommage, calcul, filtre, jointure sur identifiant), Python + pandas est plus adapté, plus portable et plus maintenable.

4.5 FME : l'excellence géospatiale et au-delà

FME (Feature Manipulation Engine) est développé par Safe Software, société canadienne basée à Surrey, en Colombie-Britannique. Le produit FME est lancé en 1996 — initialement pour répondre à un contrat du gouvernement canadien sur le monitoring de la déforestation, qui nécessitait de traduire des formats CAD en formats GIS. C'est cette origine géospatiale qui explique pourquoi FME reste, en 2026, la référence absolue pour les transformations de données géospatiales complexes.

Mais FME n'est plus un outil purement géospatial. Sa documentation officielle le positionne comme une plateforme d'intégration "All-Data, Any-AI" : il traite des JSON, XML, bases relationnelles, APIs REST, fichiers BIM (IFC), flux IoT, données SAP — au même titre que des Shapefile ou des GeoPackage. Ses 800+ transformers out-of-the-box couvrent des cas d'usage bien au-delà du SIG.

L'architecture FME se compose de deux produits complémentaires. **FME Form** est l'environnement d'authoring visuel : on y construit les "workspaces" en reliant visuellement des readers, transformers et writers. La version 2026.1, sortie en mars 2026, introduit le live feature caching (visualisation en temps réel des données pendant l'exécution du workspace), le support natif de Microsoft Fabric, de Databricks avec types spatiaux natifs, et DuckDB v1.4.3. **FME Flow** (anciennement FME Server) est le serveur d'automatisation : il exécute les workspaces selon des schedules, en réponse à des événements, ou via API — avec gestion des logs, des alertes et du monitoring.

Les forces de FME par rapport à QGIS sont concrètes : gestion d'erreur native par workspace, logs complets et structurés, reprise sur échec configurable, connecteurs non-géospatiaux natifs, performance sur gros volumes, et interface de monitoring centralisée dans FME Flow. Son point faible principal reste le coût : depuis 2023, Safe Software ne publie plus de grille tarifaire et le pricing s'effectue sur devis. Selon les fourchettes mentionnées par les revendeurs et la communauté, une licence FME Form se situe en ordre de grandeur autour de 1 000 à 1 500 $/an, avec FME Flow facturé séparément en fonction du dimensionnement — un investissement justifiable pour des organisations qui font de l'intégration de données géospatiales complexes à titre régulier, difficilement justifiable pour un usage occasionnel.

QGIS + PYQGIS

Open Source

- ✓ Gratuit, communauté active
- ✓ Idéal pour les opérations spatiales ponctuelles
- ✓ Automatisable via qgis_process + cron
- ✓ Pas de gestion d'erreur structurée
- ✓ Pas de connecteurs non-géospatiaux natifs

VS

FME FORM + FLOW

Commercial

- → Gestion d'erreur et logs natifs
- → 800+ transformers spatiaux et non-spatiaux
- → FME Flow pour l'automatisation enterprise
- → Monitoring centralisé et alertes
- → Pricing sur devis (~1 000–1 500 $/an pour FME Form)

Résumé Partie 4

L'ETL géospatial ajoute trois couches de complexité à l'ETL généraliste : CRS, formats spécifiques, et opérations spatiales. GDAL/OGR est le socle universel, invisible mais indispensable. QGIS est pertinent dès qu'une opération spatiale est impliquée — et seulement dans ce cas. FME est la référence enterprise pour les pipelines géospatiaux complexes, avec des capacités qui débordent largement sur le non-spatial. Le choix entre les deux se fait sur le budget, la criticité des pipelines et la régularité des besoins.

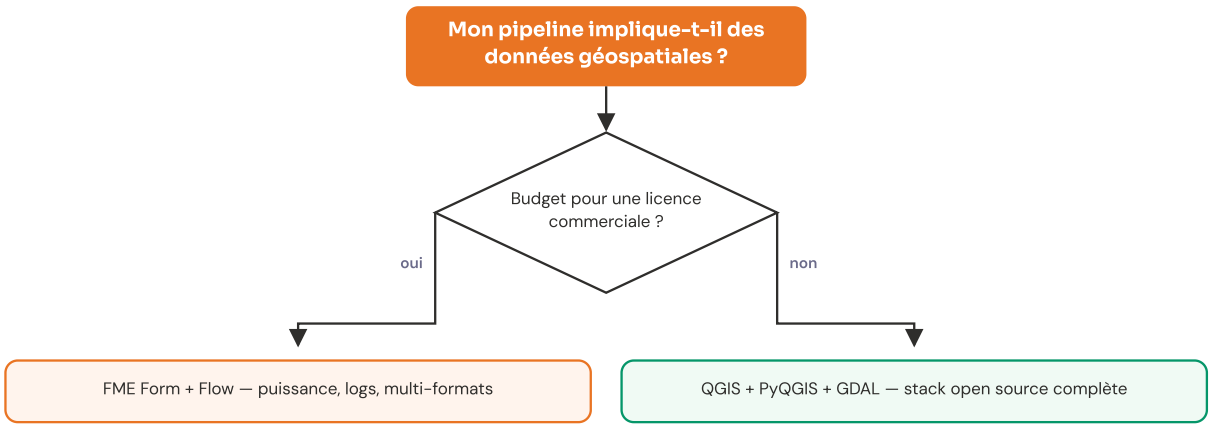
Partie 5 — Guide décisionnel

Les parties précédentes ont décrit les outils. Cette section aide à choisir — en partant de la situation réelle, pas du catalogue des outils disponibles. Trois niveaux de lecture : un arbre de décision pour le premier filtre, une matrice d'usage pour affiner, et des scénarios types pour les cas les plus courants.

Arbre de décision

L'arbre de décision se lit en deux temps. La première question est éliminatoire : **y a-t-il du géospatial ?** Si oui, deux options selon le budget. Si non, on bascule sur la chaîne généraliste, où la volumétrie puis les compétences orientent vers le bon outil.

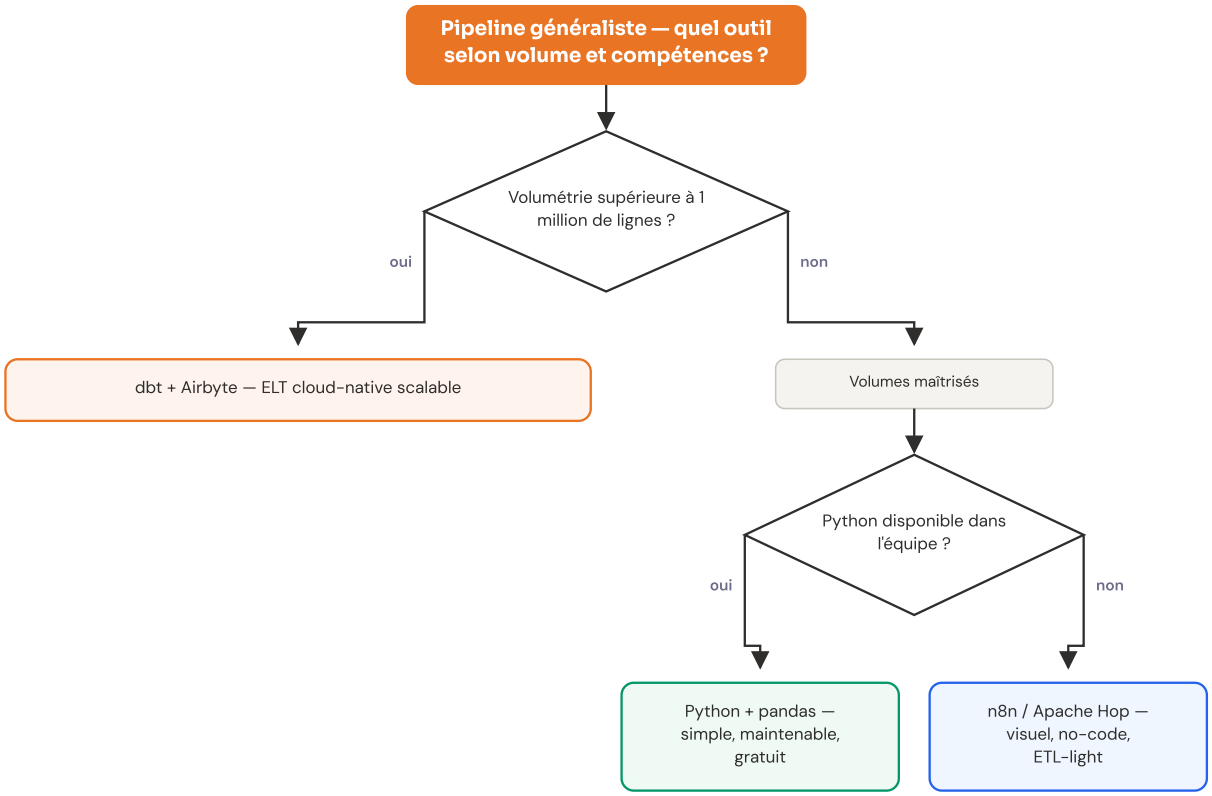
ÉTAPE 1 — PIPELINE AVEC DONNÉES GÉOSPATIALES ?



Si pas de géospatial : passer à l'étape 2

Si la réponse à la question initiale est « non, pas de géospatial », l'arbre continue ci-dessous avec la chaîne de décision généraliste.

ÉTAPE 2 — PIPELINE GÉNÉRALISTE (SANS GÉOSPATIAL)



Matrice d'usage

TAB. 3 — QUEL OUTIL SELON VOTRE CONTEXTE — SYNTHÈSE 2026

CRITÈRE	PYTHON/PANDAS	DBT + AIRBYTE	FME	QGIS PROCESSING	N8N
Géospatial natif	❌ Via GeoPandas	❌ Non	✅ Référence	✅ Natif	❌ Non
Volumétrie < 100K	✅	✅	✅	✅	✅

CRITÈRE	PYTHON/PANDAS	DBT + AIRBYTE	FME	QGIS PROCESSING	N8N
Volumétrie > 1M	⚠️ Selon RAM	✅ Cloud scalable	✅	⚠️ Limité	❌
Compétences requises	Python	SQL + DevOps	Visuel / no-code	SIG + Python	No-code
Scheduler intégré	❌ Airflow externe	✅ Via Airflow	✅ FME Flow	❌ cron externe	✅ Natif
Gestion erreurs	⚠️ À coder	✅ Native dbt	✅ Native FME	❌ Limitée	⚠️ Basique
Coût licence	Gratuit	Gratuit (OS)	Sur devis	Gratuit	Gratuit (OS)
Migration ERP	✅ Adapté	✅ Adapté	✅ Adapté	❌ Non adapté	⚠️ < 100K lignes

Pièges classiques à éviter

Au-delà du choix d'outil, certaines erreurs récurrentes invalident un projet ETL — quel que soit l'outil retenu. Ces pièges sont systématiquement observés sur les projets PME et collectivités, et leur coût de remédiation augmente exponentiellement avec le temps avant détection.

RISQUE	PROBA...	IMPACT	MITIGATION
Reprojection oubliée ou erronée	Élevée	🔴 Critique	Documenter le CRS source et cible dans le pipeline · tests systématiques sur des points de contrôle connus
n8n utilisé hors périmètre (>100K lignes, pipeline critique)	Moyenne	🔴 Critique	Définir un seuil de bascule vers Python+SQLAlchemy · pas de migration ERP en n8n

RISQUE	PROBA…	IMPACT	MITIGATION
Coût SaaS sous-estimé (Fivetran, dbt Cloud, Snowflake)	**Élevée**	🟠 Élevé	Modéliser le TCO sur 24 mois · prévoir un plan B self-hosted
QGIS comme ETL généraliste (sans opération spatiale)	**Moyenne**	🟡 Moyen	Règle stricte : QGIS uniquement si transformation spatiale · sinon Python+pandas
Absence de tests sur les transformations	**Élevée**	🔴 Critique	dbt tests pour SQL · pytest pour Python · validation sur jeu de données de référence
Pipeline non versionné dans Git	**Élevée**	🟠 Élevé	Tout artefact (SQL, YAML, scripts) versionné · pas de modification directe en production
Pas de stratégie de reprise sur erreur	**Moyenne**	🔴 Critique	Idempotence des étapes · checkpoints et reprise ciblée · alertes sur échec

Scénarios types

1 Géomaticien collectivité — budget zéro

- ✓ Données géospatiales, reprojections fréquentes
- ✓ Équipe de 1 à 3 personnes, budget logiciel nul
- ✓ Stack : QGIS Graphical Modeler + qgis_process + cron
- ✓ Python + GeoPandas pour les cas complexes

2 Data engineer PME — data warehouse

- ✓ 500K à 5M lignes, sources multiples (ERP, SaaS, CSV)
- ✓ SQL maîtrisé, pas de budget SaaS élevé
- ✓ Stack : Airbyte self-hosted + PostgreSQL + dbt Core + Airflow
- ✓ Coût infra : 30–80 €/mois (VPS)

3 Intégrateur Odoo — migration ERP

- ✓ Import de données depuis un ancien ERP (SAP, Sage, Dynamics)
- ✓ Mapping de champs complexe, règles de nettoyage métier
- ✓ Stack : Python + pandas + SQLAlchemy + scripts de validation
- ✓ Routes ETL préparées en amont, exécution le jour J

4 Organisation multi-sources — géospatial + métier

- ✓ Données géospatiales ET données métier (BDD, Excel, API)
- ✓ Pipelines critiques, besoin de monitoring et alertes
- ✓ Stack : FME Flow (géospatial) + dbt + Airbyte (non-spatial)
- ✓ Budget : licence FME + infra cloud

Partie 6 — Architectures de référence

Les architectures présentées ici sont des références concrètes, calibrées sur trois profils courants dans l'audience MP-i. Elles ne sont pas des prescriptions universelles — chaque projet a ses contraintes propres — mais des points de départ validés sur le terrain.

LÉGÈRE

Stack ETL open source — PME et collectivités

15-40 €/mois (VPS)

Volumes < 500K lignes · Budget zéro licence · Équipe technique limitée

Python + pandas · SQLAlchemy · GDAL/ogr2ogr · PostgreSQL · cron

INTERMÉDIAIRE

Stack ELT data warehouse

50-200 €/mois (infra self-hosted) ou 300-800 €/mois (cloud managé)

Volumes 500K–10M lignes · SQL maîtrisé · Nouveaux projets data

Airbyte (self-hosted) · PostgreSQL ou Snowflake · dbt Core · Apache Airflow

GÉOSPATIALE ENTERPRISE

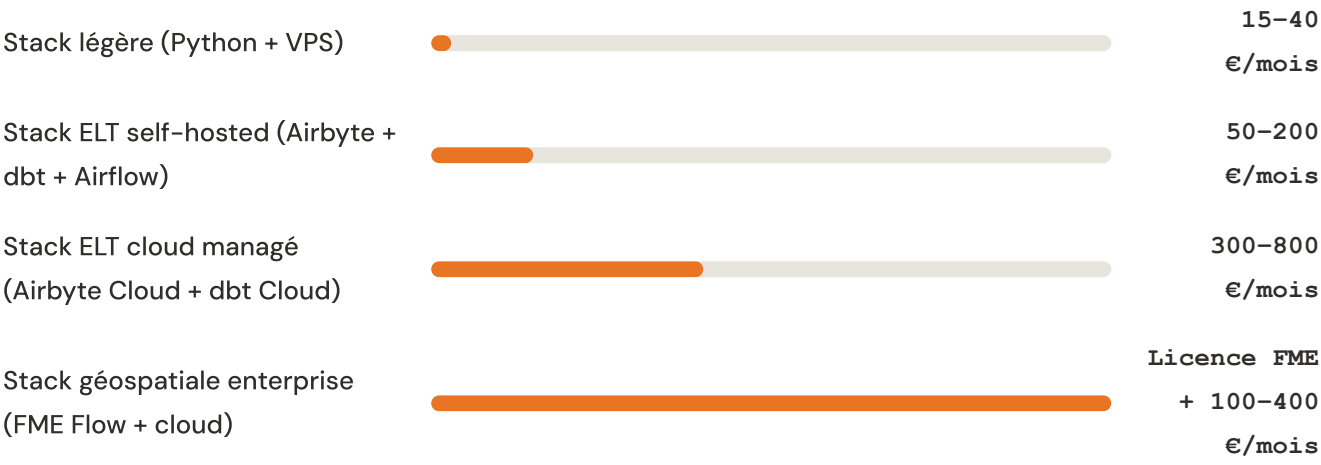
Stack FME + cloud pour pipelines géospatiaux critiques

Licence FME sur devis + infra cloud 100-400 €/mois

Données géospatiales complexes · Pipelines automatisés · Criticité élevée

FME Form + FME Flow · PostGIS · S3 / Cloudflare R2 · dbt (couche non-spatiale)

Comparaison des coûts



Le coût invisible : le temps de développement

La comparaison de coûts ci-dessus ne prend en compte que les coûts d'infrastructure et de licence. Le temps de développement initial et de maintenance est souvent le poste dominant — en particulier pour les stacks open source qui demandent davantage de configuration. Un pipeline n8n prend 2 heures à mettre en place ; un pipeline dbt + Airbyte + Airflow prend 2 à 5 jours. Les deux sont légitimes, à des échelles différentes.

Partie 7 — Tendances et horizon 2027

L'écosystème ETL évolue vite. Les six tendances suivantes méritent une attention particulière dans les 12 à 18 prochains mois.

🔄

ELT comme nouveau standard

En 2026, ELT domine tous les nouveaux projets sur infrastructure cloud. L'ETL classique subsiste pour les contraintes réglementaires (PII non stocké brut) et les bases opérationnelles on-premise. Pour tout nouveau projet data warehouse, partir directement en ELT.

🤝

Consolidation du modern data stack

L'accord de fusion dbt Labs + Fivetran signé en octobre 2025 (clôture attendue 2026) signale que le marché se concentre. Les outils de la couche ingestion (EL) et transformation (T) tendent à se rapprocher dans des offres intégrées. Les prochaines années verront probablement d'autres consolidations.

🗺️

ETL géospatial cloud-native

GeoParquet + DuckDB + PMTiles remplacent progressivement les pipelines Shapefile → PostGIS pour les cas de consultation. La frontière entre ETL géospatial et data engineering généraliste se réduit. Les géomaticiens qui maîtrisent dbt et DuckDB seront mieux positionnés.

IA dans les pipelines

FME 2026 intègre des connecteurs IA (MCPCaller, Azure AI, Hugging Face). dbt expérimente la génération de modèles assistée. Ces fonctionnalités restent en phase d'adoption — prometteuses pour la détection d'anomalies et la suggestion de transformations, pas encore matures pour remplacer la logique métier.

Reverse ETL en croissance

Injecter les données analytiques du warehouse dans les outils métier (Odoo, Salesforce, HubSpot) est un besoin croissant. Census et Hightouch sont les acteurs principaux. Pour les intégrateurs Odoo, c'est un levier à surveiller : enrichir Odoo depuis un data warehouse sans développement custom.

Python partout, no-code nulle part

Même les outils "no-code" comme FME et n8n renforcent leur support Python. La frontière entre visuel et code s'estompe — non pour remplacer Python, mais pour l'intégrer là où il est le plus puissant. La compétence Python reste l'investissement le plus durable dans l'écosystème ETL.

Recommandation prospective

Deux investissements durables quelle que soit l'évolution du marché : maîtriser dbt pour la transformation SQL-first dans les warehouses cloud, et maîtriser GDAL/PyQGIS pour les pipelines géospatiaux. Ces deux compétences traverseront les consolidations d'éditeurs et les changements de paradigme — elles adressent des problèmes fondamentaux qui ne disparaissent pas.

Conclusion

L'ETL n'est pas une technologie récente, ni un sujet réservé aux grandes organisations dotées d'équipes data engineering. C'est une discipline fondamentale — extraire une donnée, la transformer pour qu'elle convienne à son usage, la charger dans sa destination — qui concerne toute organisation qui manipule de la donnée, quelle que soit sa taille.

Ce livre blanc a voulu montrer trois choses. La première : le paysage ETL est vaste et fragmenté, mais il se structure en cinq familles cohérentes selon la volumétrie, les compétences et le budget. Choisir un outil ETL n'est pas une question de mode — c'est une question d'adéquation au contexte réel. La deuxième : l'ETL géospatial n'est pas une niche exotique. Il partage les mêmes fondamentaux que l'ETL généraliste, avec trois dimensions supplémentaires que les outils non-spatiaux ne gèrent pas. GDAL, QGIS Processing et FME répondent à des besoins réels et distincts — et savoir lequel choisir est une compétence différenciante. La troisième : la donnée brute ne sera jamais prête à l'emploi. Ce problème ne disparaîtra pas. Les outils évolueront, les paradigmes (ETL, ELT, reverse ETL) se succéderont, mais le travail de préparation restera — et mieux il est structuré, plus il est reproductible, maintenable et fiable.

Pour aller plus loin sur les applications concrètes de ces principes, trois articles de blog MP-i développent les cas d'usage métier en détail : la préparation de données géospatiales avant QGIS, les croisements et agrégations en amont du SIG, et la préparation d'une migration ERP vers Odoo via des routes ETL structurées. Ces articles sont disponibles sur blog.mp-i.pro.

CRITIQUE	Définir la volumétrie et la présence ou non de données géospatiales avant de choisir un outil — ces deux critères éliminent 80 % des options.
IMPORTANT	Investir sur dbt et GDAL/PyQGIS comme compétences fondamentales durables — elles traverseront les consolidations d'éditeurs.
UTILE	Explorer le reverse ETL pour réinjecter les données analytiques dans Odoo ou les outils métier — un levier sous-exploité en PME.

Références et ressources

Documentation officielle

- → GDAL/OGR — documentation et utilitaires CLI : gdal.org ↗
- → QGIS Processing Framework : docs.qgis.org ↗
- → dbt Core — documentation officielle : docs.getdbt.com ↗
- → Apache Airflow — documentation : airflow.apache.org ↗
- → FME — documentation Safe Software : docs.safe.com ↗

Outils et plateformes

- → Airbyte — plateforme ELT open source : airbyte.com ↗
- → Apache Hop — ETL visuel open source : hop.apache.org ↗
- → n8n — automatisation légère self-hosted : n8n.io ↗
- → Prefect — orchestration Python moderne : www.prefect.io ↗
- → Talaxie — fork communautaire de Talend Open Studio : www.talaxie.com ↗

Lectures complémentaires

- → The history and future of the data ecosystem — dbt Labs : www.getdbt.com ↗
- → FME 2026.1 — notes de version : community.safe.com ↗
- → GeoParquet — spécification : geoparquet.org ↗

Glossaire

CDC (Change Data Capture)

Technique d'ingestion qui ne transfère que les modifications (insertions, mises à jour, suppressions) depuis la dernière synchronisation, plutôt que de recharger l'intégralité des données. Réduit la charge sur les sources et améliore la fraîcheur des données en cible.

CRS (Coordinate Reference System)

Système de référence de coordonnées — définit comment les coordonnées géographiques d'une donnée spatiale sont interprétées. En France, le CRS légal pour les données terrestres est le Lambert 93 (EPSG:2154). Les APIs cartographiques web utilisent WGS84 (EPSG:4326).

DAG (Directed Acyclic Graph)

Graphe orienté acyclique — représentation des dépendances entre tâches dans un pipeline ETL ou un workflow d'orchestration. Utilisé par Apache Airflow et dbt pour définir l'ordre d'exécution des tâches sans cycle.

dbt (data build tool)

Outil de transformation SQL-first open source. Gère la couche "T" de l'ELT — les transformations dans le warehouse. Supporte le versioning, les tests automatisés et la documentation. Fondé en 2016, aujourd'hui standard de facto de l'analytics engineering.

ELT (Extract, Load, Transform)

Pattern d'intégration où les données sont d'abord chargées brutes dans le warehouse, puis transformées en interne par le moteur de calcul du warehouse. Dominant sur les nouveaux projets cloud en 2026.

ETL (Extract, Transform, Load)

Pattern d'intégration classique où les données sont transformées avant d'être chargées dans la destination. L'infrastructure de transformation est externe au warehouse. Adapté aux contraintes réglementaires (données personnelles non stockées brutes).

FME (Feature Manipulation Engine)

Plateforme d'intégration de données développée par Safe Software (Canada, 1996). Couvre les données spatiales et non-spatiales. FME Form est l'environnement d'authoring visuel, FME Flow le serveur d'automatisation. 800+ transformers out-of-the-box.

GDAL (Geospatial Data Abstraction Library)

Bibliothèque open source fondatrice de l'interopérabilité géospatiale, créée en 1998 par Frank Warmerdam. Supporte 200+ formats raster et vecteur. Utilisée en sous-couche par QGIS, FME, PostGIS. ogr2ogr est son outil CLI de référence.

Modern data stack

Architecture ELT cloud-native composée d'outils spécialisés et interopérables : outil d'ingestion (Airbyte, Fivetran) + warehouse cloud (Snowflake, BigQuery, Redshift) + outil de transformation (dbt) + orchestrateur (Airflow). Standard de facto pour les nouveaux projets data depuis 2020.

Reverse ETL

Pattern qui réinjecte les données transformées d'un warehouse dans les outils métier opérationnels (CRM, ERP, marketing). Activation de la donnée analytique dans les outils où travaillent les équipes métier. Outils principaux : Census, Hightouch.

ogr2ogr

Utilitaire CLI de GDAL/OGR pour la conversion et la reprojection de données vecteur géospatiales. Supporte 200+ formats en lecture et écriture. Outil de référence pour les conversions batch sans interface graphique.

PyQGIS

API Python de QGIS. Permet d'accéder aux couches, algorithmes Processing et fonctions QGIS depuis des scripts Python, en mode interactif ou standalone (sans interface graphique).

qgis_process

Outil CLI de QGIS qui permet d'exécuter des algorithmes Processing ou des modèles Graphical Modeler depuis la ligne de commande, sans ouvrir l'interface graphique. Brique d'intégration dans les pipelines automatisés.

Historique des révisions

VERSION	DATE	AUTEUR	MODIFICATIONS
v1.0.0	25 avril 2026	Mattieu Pottier	Version initiale
v1.1.0	26 avril 2026	Mattieu Pottier	Promotion du « Bref historique » en Partie 1 et renumérotation complète des parties (1 à 7) · ajout SWOT du modern data stack · ajout section « Pièges classiques » (risks) · ajout quote d'ouverture Partie 2 · correction date accord dbt+Fivetran (octobre 2025) · reformulation pricing FME · alignement bio auteur · correction SVG arbre de décision
v1.1.1	26 avril 2026	Mattieu Pottier	Remplacement du SVG inline « diagram decision » (composant non reconnu par le renderer) par deux blocs `flowchart` natifs successifs : étape 1 géospatial, étape 2 généraliste

_

À propos de l'auteur

Mattieu Pottier

Consultant indépendant en transformation digitale — MP-i · Mattieu Pottier Indépendant

Expert SIG, architecture système et Odoo. Spécialisé dans les ERP (Odoo V18/V19), les systèmes d'information géographique (QGIS, développement Python, PostGIS), et les applications web (FastAPI, MapLibre GL, WordPress). Auteur du blog technique mp-i.pro. mp-i.pro