

Document d'architecture



Projet : Amazon Review Analysis

Auteur : Dyhia TOUAHRI

Date : 15 novembre 2025

Version 1.0

Table des matières

| | | |
|--------|--|----|
| 1. | Contexte et objectifs..... | 3 |
| 1.1. | Objectifs du Système | 3 |
| 1.2. | Périmètre..... | 3 |
| 2. | Les données | 4 |
| 2.1. | Formats de Données | 4 |
| 2.2. | Modèle de Données et Relations..... | 5 |
| 2.2.1. | Table Review | 5 |
| 2.2.2. | Table PRODUCT | 5 |
| 2.2.3. | Table CATEGORY | 5 |
| 2.2.4. | Table ORDERS | 5 |
| 2.2.5. | Table PRODUCT_REVIEWS | 6 |
| 2.2.6. | Table REVIEW_IMAGES | 6 |
| 2.2.7. | Relation entre les tables et cardinalités | 6 |
| 2.3. | Technologies Choisies..... | 6 |
| 2.4. | Proposition de stockage massif..... | 7 |
| 2.5. | Procédures de sélection et d'extraction..... | 8 |
| 2.5.1. | Procédure d'Extraction PostgreSQL → S3 | 8 |
| 2.6. | Extraction S3, nettoyage de données et stockage dans Snowflake | 9 |
| 2.6.1. | Nettoyage des données | 9 |
| 2.6.2. | Stockage dans Snowflake | 10 |
| 2.6.3. | Traitement des Données Rejetées vers MongoDB et logs | 11 |
| 2.7. | Analyse des données (Machine Learning)..... | 11 |
| 2.7.1. | Algorithme de pondération | 11 |
| 2.7.2. | Algorithme zero-shot | 12 |
| 2.8. | Schéma complet de l'architecture | 12 |
| 3. | SÉCURITÉ ET CONFORMITÉ | 13 |
| 3.1. | Mesures de Sécurité..... | 13 |
| 3.2. | Conformité RGPD et CCPA..... | 13 |

1. Contexte et objectifs

Ce projet vise à développer une solution automatisée capable de classifier les avis clients d'Amazon et d'identifier les retours les plus pertinents. Actuellement, ces tâches sont effectuées manuellement par les Business Analysts, ce qui représente une charge de travail importante et peu efficiente.

La présente documentation a pour objectif de définir l'architecture de la solution proposée. Elle décrit les composants nécessaires pour stocker, traiter et analyser les données transactionnelles, tout en assurant leur qualité, leur sécurité et leur conformité aux exigences réglementaires.

1.1. Objectifs du Système

Objectifs fonctionnels

Chaque jour, les données sont extraites depuis une base PostgreSQL transactionnelle, puis transformées et enrichies à l'aide de différentes métriques, telles que la longueur des textes, la présence d'images ou encore l'historique des commandes. Une fois nettoyées, elles sont chargées dans Snowflake afin de permettre leur analyse. L'ensemble du processus bénéficie d'une traçabilité complète grâce à MongoDB, qui conserve les logs, les métadonnées ainsi que les données rejetées. Enfin, une qualité de données supérieure à 99 % est garantie grâce à des mécanismes de détection et d'isolation des anomalies.

Objectifs non-fonctionnels

Le temps de traitement de bout en bout est inférieur à dix minutes, avec un objectif ambitieux de cinq minutes. Le coût de traitement reste maîtrisé, à moins de 0,10 \$ pour mille avis analysés. La solution est conçue pour être pleinement scalable, capable de gérer plus d'un million d'avis sans nécessiter de refonte majeure. Elle offre également une grande fiabilité, avec un taux d'échec inférieur à 1 %, assuré par des mécanismes de retry et de recovery.

1.2. Périmètre

Dans le périmètre

Dans le périmètre, le projet inclut l'extraction de six tables sources — product, category, review, product_reviews, review_images et orders — suivie de transformations basées sur des jointures SQL et des enrichissements de données. Les jeux de données sont ensuite validés et nettoyés grâce à Great Expectations, puis chargés dans Snowflake pour leur exploitation analytique. L'ensemble du pipeline est supervisé via des mécanismes de logging et de monitoring dans MongoDB, et orchestré à l'aide d'Apache Airflow. Une interface utilisateur, Streamlit App dans Snowflake, permet enfin de visualiser les résultats

au travers d'un tableau de bord dédié et une application Streamlit permet de requêter les avis les plus pertinents en fonction de l'acheteur.

Hors périmètre

En dehors du périmètre, le projet n'inclut pas l'analyse en temps réel sous forme de streaming, ni les traitements NLP avancés tels que l'analyse de sentiment ou l'intégration de modèles de machine learning. La gestion des images, incluant leur stockage ou leur traitement, ne fait pas non plus partie des fonctionnalités prévues.

2. Les données

| Table | Volume | Description | Rôle dans le pipeline |
|-----------------|--------|------------------------|--|
| product | 42858 | Catalogue produits | Table de référence produits principale |
| category | 2 | Catégories de produits | Lookup pour enrichissement |
| review | 111322 | Avis clients | Table transactionnelle principale |
| product_reviews | 111322 | Liaison produit-avis | Table de jointure |
| review_images | 119382 | URLs d'images d'avis | Données semi-structurées |
| orders | 222644 | Historique commandes | Enrichissement comportemental |

- Volume total : 607 630 enregistrements.
- Poids estimé : ~500 MB (format CSV compressé).
- Croissance attendue : +10% mensuel.

2.1. Formats de Données

Données structurées

- Format source : Tables relationnelles PostgreSQL (types natifs : VARCHAR, INTEGER, TIMESTAMP, TEXT)
- Format intermédiaire : CSV UTF-8 sur S3 (séparateur : ;)
- Format cible : Table Snowflake avec types optimisés (TIMESTAMP_NTZ, BOOLEAN)

Données semi-structurées

- URLs d'images stockées en TEXT
- Métadonnées JSON dans MongoDB (logs, rejets)

Encodage : UTF-8 systématique pour compatibilité internationale

2.2. Modèle de Données et Relations

2.2.1. Table Review

| Colonne | Type | Contraintes | Description |
|---------------------|--------------|--------------|------------------------------|
| REVIEW_ID | INTEGER | PK, NOT NULL | Identifiant unique de l'avis |
| BUYER_ID | INTEGER | FK, NOT NULL | Référence vers BUYER |
| DESC | TEXT | NOT NULL | Texte complet de l'avis |
| TITLE | VARCHAR(150) | NOT NULL | Titre de l'avis |
| RATING | INTEGER | CHECK (1-5) | Note de 1 à 5 étoiles |
| SELLER_PRODUCT_FLAG | BOOLEAN | NOT NULL | 0=Produit, 1=Vendeur |

2.2.2. Table PRODUCT

| Colonne | Type | Contraintes | Description |
|-------------|---------------|--------------|----------------------------|
| P_ID | INTEGER | PK, NOT NULL | Identifiant unique produit |
| P_NAME | VARCHAR(200) | NOT NULL | Nom du produit |
| DESC | TEXT | NULL | Description détaillée |
| PRICE | DECIMAL(10,2) | NOT NULL | Prix unitaire |
| QTY | INTEGER | NOT NULL | Quantité en stock |
| CATEGORY_ID | INTEGER | FK, NOT NULL | Référence vers CATEGORY |

2.2.3. Table CATEGORY

| Colonne | Type | Contraintes | Description |
|-------------|--------------|------------------|-----------------------|
| CATEGORY_ID | INTEGER | PK, NOT NULL | Identifiant catégorie |
| NAME | VARCHAR(100) | NOT NULL, UNIQUE | Nom de la catégorie |
| DESC | TEXT | NULL | Description catégorie |

2.2.4. Table ORDERS

| Colonne | Type | Contraintes | Description |
|-------------|-------------|--------------|-------------------------|
| ORDER_ID | INTEGER | PK, NOT NULL | Identifiant ORDER |
| BUYER_ID | VARCHAR(40) | FK, NOT NULL | Référence vers BUYER |
| DISCOUNT_ID | INTEGER | FK, NOT NULL | Référence vers DISCOUNT |
| PAYMENT_ID | INTEGER | FK, NOT NULL | Référence vers PAYMENT |
| ORDER_DATE | DATE | NOT NULL | Date de la commande |

2.2.5. Table PRODUCT_REVIEWS

| Colonne | Type | Contraintes | Description |
|-----------|-------------|--------------|---------------------|
| P_ID | VARCHAR(10) | FK,NOT NULL | Identifiant PRODUCT |
| REVIEW_ID | INTEGER | FK, NOT NULL | Identifiant REVIEW |

2.2.6. Table REVIEW_IMAGES

| Colonne | Type | Contraintes | Description |
|--------------|--------------|-------------|--------------------|
| REVIEW_ID | INTEGER | FK,NOT NULL | Identifiant review |
| REVIEW_IMAGE | VARCHAR(250) | NOT NULL | Url image |

2.2.7. Relation entre les tables et cardinalités

- BUYER (1 : N) REVIEW.
- REVIEW (N : N) PRODUCT via PRODUCT_REVIEWS.
- PRODUCT (N : 1) CATEGORY.
- ORDERS (N : 1) BUYER.
- ORDERS (0 : N) DISCOUNT.
- ORDERS (N : 1) PAYMENT.
- REVIEW (1 : N) REVIEW_IMAGES.

La hiérarchie de catégories est limitée (2 catégories dans la base actuelle).
Unicité garantie par les clés primaires ((REVIEW_ID, BUYER_ID, P_ID))

2.3. Technologies Choisies

AWS S3

Le choix d’AWS S3 se justifie par son niveau exceptionnel de durabilité, garantissant une conservation fiable des données à long terme. Le service offre également une scalabilité pratiquement illimitée, permettant de gérer des volumes croissants sans reconfiguration. Son coût reste maîtrisé, avec un tarif standard d’environ 0,023 \$ par gigaoctet et par mois. De plus, S3 s’intègre nativement avec Snowflake via les External Stages, ce qui facilite les flux de données. Enfin, les lifecycle policies permettent d’automatiser l’archivage et l’optimisation des coûts selon la durée de rétention des données.

Snowflake

Snowflake est retenu pour son architecture cloud-native qui sépare le stockage et la puissance de calcul, offrant une flexibilité optimale pour adapter les ressources selon les besoins. La plateforme propose de l’auto-scaling afin de s’ajuster automatiquement au volume de requêtes. Elle gère nativement les formats semi-structurés grâce au type VARIANT, facilitant

la manipulation de JSON. Snowflake permet également un partage de données sécurisé entre équipes ou partenaires.

MongoDB

MongoDB 7.0 est choisi pour sa flexibilité de schéma, particulièrement adaptée à des logs dont la structure peut varier dans le temps. La base de données offre des performances élevées en écriture, essentielles pour absorber un flux important d'événements. Son pipeline d'agrégation permet d'effectuer des requêtes complexes et puissantes de manière efficace. MongoDB intègre également des mécanismes natifs de réplication et de haute disponibilité, assurant la robustesse et la continuité du service.

| Technologie | Avantages | Inconvénients |
|---------------|---|--|
| PostgreSQL | Standard industrie, SQL ACID, open-source | Scalabilité limitée (vertical) |
| AWS S3 | Scalabilité infinie, 99,99% SLA, coût bas | Latence réseau |
| Snowflake | Séparation compute/storage, auto-scaling, Time Travel | Coût variable (pay-per-use) |
| MongoDB | Flexibilité schéma, haute performance écriture | Pas de transactions ACID multi-documents |
| Airflow | Standard orchestration, UI riche, communauté large | Courbe apprentissage |
| Python/pandas | Écosystème riche, facile à maintenir | Performance limitée (GIL) |

Les alternatives

| Besoin | Choix retenu | Alternatives considérées | Raison du choix |
|----------------|--------------------|--------------------------------|-------------------------------|
| Orchestration | Airflow | Prefect, Dagster | Standard industrie, UI |
| Data Lake | S3 | Azure Blob, GCS | Intégration Snowflake |
| Data Warehouse | Snowflake | Redshift, BigQuery, Databricks | Meilleur rapport qualité/prix |
| Processing | pandas | Spark, Dask | Volume actuel < 1 GB |
| Validation | Great Expectations | dbt tests, custom | Framework mature |

2.4. Proposition de stockage massif

Pattern architectural : Medallion Architecture (Bronze → Silver → Gold)

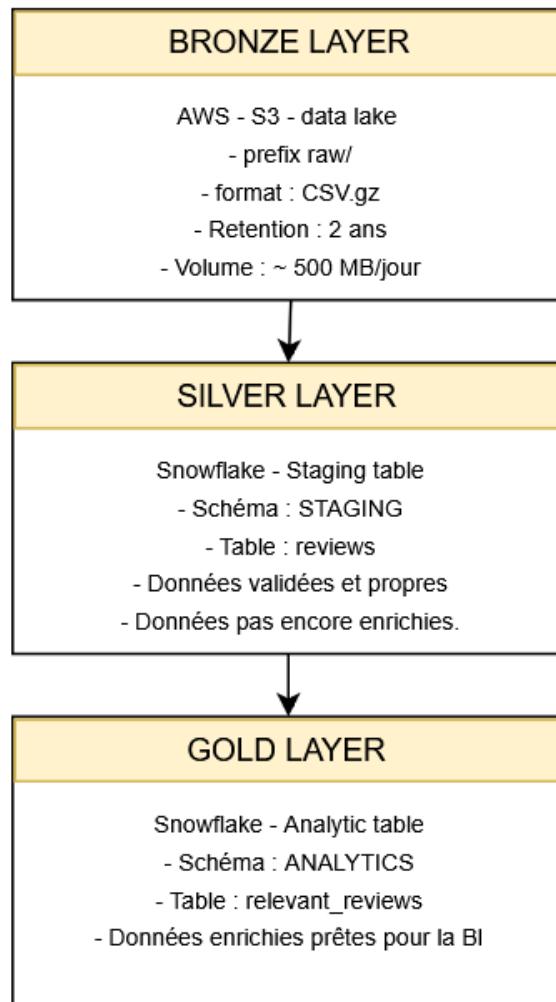


Figure 1 : Architecture en médaillon

2.5. Procédures de sélection et d'extraction

2.5.1. Procédure d'Extraction PostgreSQL → S3

Outil : Script Python extract_to_s3.py orchestré par Airflow DAG

Gestion de la Compatibilité des Données

Problématiques identifiées et solutions :

| Problématique | Impact | Solution implémentée |
|---------------|---|---|
| Encodage | Caractères spéciaux (émojis, accents) corrompus | UTF-8 forcé à l'extraction + validation |
| Valeurs NULL | Inconsistance NULL vs chaîne vide | Préservation NULL explicite (na_rep="") |

| | | |
|------------------|----------------------------|--|
| Dates/Timestamps | Formats locaux (timezone) | Normalisation ISO 8601 (YYYY-MM-DD HH:MM:SS) |
| Décimales | Précision flottants (prix) | Arrondi à 2 décimales, format US (point) |
| Délimiteurs | Virgules dans descriptions | Échappement CSV standard (quotes) |

Règles de rétention

- Archive S3 Glacier après 90 jours

2.6. Extraction S3, nettoyage de données et stockage dans Snowflake

2.6.1. Nettoyage des données

Après l'extraction et la centralisation des données brutes dans Amazon S3, une étape de pré-traitement et de validation est effectuée dans le pipeline afin de garantir la qualité, la cohérence et l'exploitabilité des données avant leur stockage définitif et leur utilisation pour l'analyse.

Cette étape comprend plusieurs phases :

1/ Jointure des données (Data Enrichment)

Les différentes tables chargées depuis S3 sont jointes à l'aide d'une requête SQL.

Les tables principales concernées sont :

- review
- product_reviews
- product
- category
- review_images
- orders

La jointure permet d'obtenir une vue enrichie contenant notamment :

- l'identifiant du client (buyer_id)
- l'identifiant et le nom du produit (p_id, product_name)
- la catégorie du produit
- le texte de l'avis (title, description)
- la note (rating)
- la longueur du texte (text_length)
- la présence d'une image associée à l'avis (has_image)
- l'existence d'une commande liée à l'avis (has_orders)

Ce dataset sert de base de référence pour les étapes suivantes (nettoyage + analyse).

2/ Détection et rejet des données non conformes

Un mécanisme systématique de détection d'anomalies est implémenté pour séparer les données propres des données rejetées.

Les règles de rejet sont :

- Duplicata : Suppression des doublons sur la clé review_id.
- Champs obligatoires manquantes : rejet si review_id ou rating ne sont pas renseignés.
- Note invalide : rejet si la note est inférieure à 1 ou plus grande que 5.
- Description vide : rejet de la ligne si le champ description est vide ou est null.

Chaque ligne rejetée est stockée avec : l'identifiant de l'avis, la raison de rejet, la date de rejet, la ligne originale complète.

Ce mécanisme permet d'avoir une traçabilité complète, de pouvoir faire des analyses sur la qualité de données et d'avoir une amélioration continue du pipeline.

2.6.2. Stockage dans Snowflake

Les données nettoyées sont ensuite chargées dans Snowflake, qui sert d'entrepôt de données principal pour la plateforme d'analyse.

La connexion est sécurisée à l'aide de variables d'environnement :

- SNOWFLAKE_USER
- SNOWFLAKE_PASSWORD
- SNOWFLAKE_ACCOUNT
- SNOWFLAKE_DATABASE
- SNOWFLAKE_SCHEMA
- SNOWFLAKE_ROLE

Les données sont stockées dans la table : STAGING.REVIEWS

Structure des données insérées

Les champs principaux insérés sont :

- review_id
- buyer_id
- p_id
- product_name
- category
- title
- description
- rating
- text_length
- has_image
- has_orders
- ingestion_timestamp (date de chargement)
- pipeline_version (version du pipeline)

2.6.3. Traitement des Données Rejetées vers MongoDB et logs

Critères de rejet

- Enregistrements avec clé primaire NULL
- Valeurs hors plage (ex: rating = 0 ou 6)
- Duplicata (review_id en double)
- Champs obligatoires NULL (buyer_id, rating, p_id)
- Incohérences de types (texte dans champ numérique)
- Champs Description NULL

Règles de rétention

- Conservation des rejets : **90 jours**

Stockage des données rejetées

Les données rejetées ne sont pas perdues. Elles sont envoyées vers MongoDB dans la collection : amazon_reviews.rejected_reviews

Ces données pourront être utilisées pour :

- des audits qualité
- des statistiques sur les causes de rejet
- une réintégration future après correction

Stockage des logs

Des métadonnées sur chaque exécution sont également enregistrées dans MongoDB, dans : amazon_reviews.pipeline_metadata, contenant :

- nombre de lignes traitées
- nombre de lignes propres
- nombre de lignes rejetées
- nombre de lignes insérées dans Snowflake

2.7. Analyse des données (Machine Learning)

Les données stockées dans Snowflake sont ensuite utilisées pour effectuer une analyse intelligente des avis clients, basée sur deux approches complémentaires.

2.7.1. Algorithme de pondération

Un score de pertinence est calculé pour chaque avis, en combinant plusieurs critères pondérés :

- Longueur du texte (text_length)
- Présence d'une image (has_image)
- Achat confirmé (has_orders)
- Note extrême (1 ou 5)
- Mots utilisés (analyse lexicale)

Ce score permet d'identifier les avis les plus utiles et de mettre en avant les commentaires les plus fiables et enfin de prioriser certains avis pour les analyses métier.

2.7.2. Algorithme zero-shot

Pour comprendre le contenu des avis sans avoir besoin de données annotées, un modèle Zero-Shot pré-entraîné est utilisé.

Les catégories retenues sont :

- product quality or satisfaction
- product defect or damaged item
- delivery issue or shipping delay
- customer service or support

Chaque avis est automatiquement classifié dans l'une de ces catégories en fonction de son contenu textuel.

Cette approche permet d'avoir une classification automatique sans entraînement spécifique, une adaptation rapide à de nouveaux projets et une vision synthétique des principaux problèmes ou points forts.

2.8. Schéma complet de l'architecture

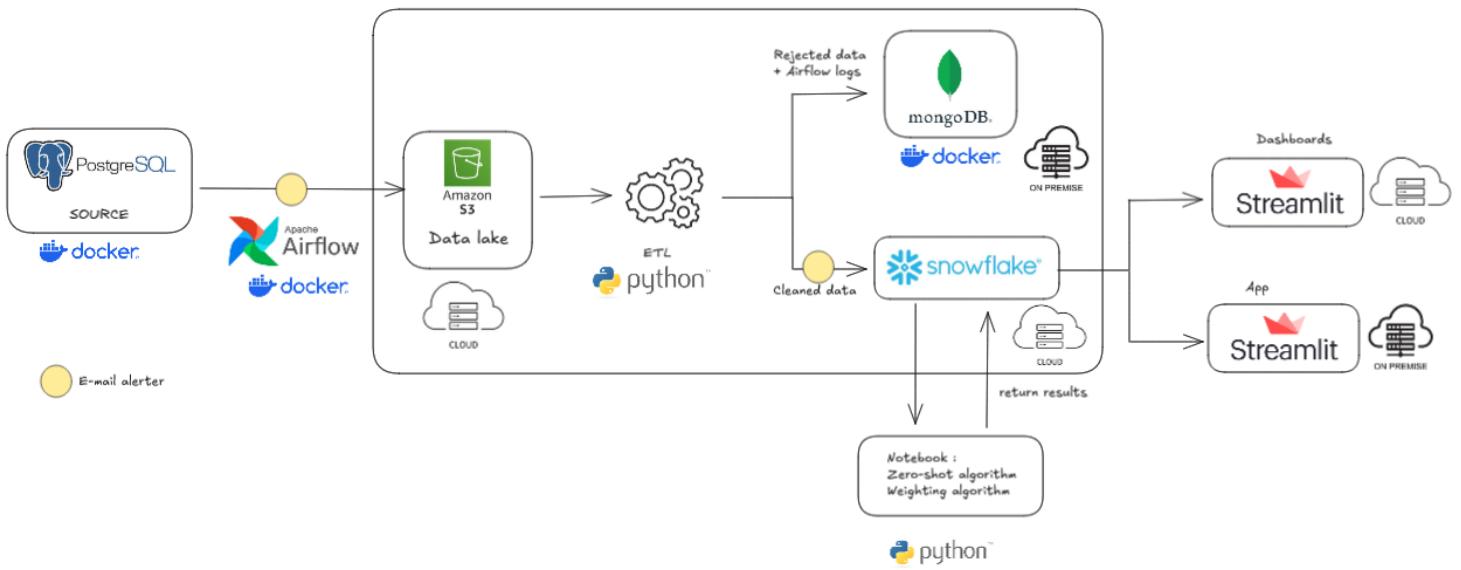


Figure 2 : schéma complet de la solution analytique

3. SÉCURITÉ ET CONFORMITÉ

3.1. Mesures de Sécurité

L'architecture mise en place repose sur un ensemble de mécanismes destinés à garantir la sécurité des données tout au long de leur cycle de vie. Ces mesures couvrent le réseau, le stockage, la gestion des accès et l'anonymisation des données sensibles.

Le tableau ci-dessous synthétise les principaux dispositifs mis en œuvre :

| Niveau | Mesure | Implémentation |
|------------------|-------------------------------|---|
| Stockage | Encryption at rest | S3 (AES-256), Snowflake (automatique) |
| Accès | Principe de moindre privilège | Compte PostgreSQL read-only, IAM Roles AWS, Snowflake role par function (dev, business analyst...). |
| Authentification | Credentials management | .env en environnement dev, AWS Secrets Manager en production |
| Anonymisation | Hachage PII | SHA-256 sur buyer_id |

Ces mesures permettent de réduire la surface d'attaque, garantir la confidentialité des données, prévenir les accès non autorisés et assurer la conformité avec les normes de sécurité industrielles.

3.2. Conformité RGPD et CCPA

RGPD

Afin de se conformer au Règlement Général sur la Protection des Données (RGPD), plusieurs principes réglementaires ont été intégrés dans le pipeline de données.

Le tableau suivant décrit les exigences RGPD applicables et les mécanismes associés.

| Article RGPD | Exigence | Implémentation |
|--------------|--------------------------|--|
| Article 5 | Minimisation des données | Extraction uniquement des champs strictement nécessaires |
| Article 4(5) | Pseudonymisation | Hachage SHA-256 des buyer_id |
| Article 17 | Droit à l'oubli | Procédure de suppression dans S3 + Snowflake (Time Travel & purge) |
| Article 32 | Sécurité du traitement | Chiffrement end-to-end, contrôle d'accès granulaire |

CCPA

La solution respecte également les principes du California Consumer Privacy Act (CCPA), notamment en ce qui concerne la transparence, le droit d'accès et le contrôle des données des utilisateurs californiens.

Même si le CCPA est moins strict que le RGPD, des mesures spécifiques sont intégrées dans le pipeline pour en assurer la conformité.

| Exigence CCPA | Description | Implémentation |
|---------------------|---|--|
| Right to Know | Droit de savoir quelles données sont collectées | Catalogue des données + documentation des champs collectés |
| Right to Delete | Possibilité de supprimer les données personnelles | Procédure d'effacement dans S3 + Snowflake |
| Right to Opt-Out | Possibilité pour l'utilisateur d'exclure l'usage de ses données | Flag d'exclusion transféré dans les zones de staging |
| Data Minimization | Limitation de la collecte des données personnelles | Sélection uniquement des champs nécessaires dans l'ETL |
| Security Safeguards | Protection contre l'accès non autorisé | Hash SHA-256, IAM roles |