DE1 — Rapport Final de Projet

Auteurs : Couzinet Lorenzo & Rabahi Enzo

Cours : Data Engineering I (2025–2026)

1 - Cas d’usage et Dataset

L'objectif est d'analyser la sécurité routière aux États-Unis pour aider les autorités de transport (DOT) et les actuaires d'assurance à identifier les zones à haut risque et les tendances temporelles des accidents.

Origine, Taille et Schéma du Dataset :

- **Source :** "US Accidents (2016 - 2023)" via Kaggle (Auteur : Sobhan Moosavi).
- **Taille :** Environ 7,7 millions d'enregistrements (~3 Go en CSV brut).
- **Schéma :** Le dataset contient 46 colonnes. Les champs clés retenus pour ce pipeline sont :
    - ID (String) : Identifiant unique.
    - Severity (Integer) : Échelle de gravité (1-4).
    - Start_Time (Timestamp) : Date et heure de l'accident.
    - State (String) : Code de l'État US.
    - Weather_Condition (String) : Météo (Pluie, Neige, etc.).

Problèmes Identifiés (Known Issues) :

- **Qualité des données :** Fort taux de valeurs nulles dans les colonnes météo.
- **Typage :** Le fichier brut est entièrement en chaînes de caractères (String), nécessitant un casting explicite pour l'analyse temporelle et numérique.

2 - Système et SLOs

Matériel et Configuration Spark :

- **Environnement :** Cluster Spark Local Standalone (Single Node).
- **Version Spark :** Apache Spark 3.x avec PySpark.
- **Configuration Spécifique :** os.environ["SPARK_LOCAL_IP"] = "127.0.0.1" a été forcé pour résoudre les erreurs réseaux Netty/Py4J en local.
- **Format de stockage :** Parquet utilisé pour les couches Bronze, Silver et Gold (Compression Snappy par défaut).

SLOs (Objectifs de Niveau de Service) :

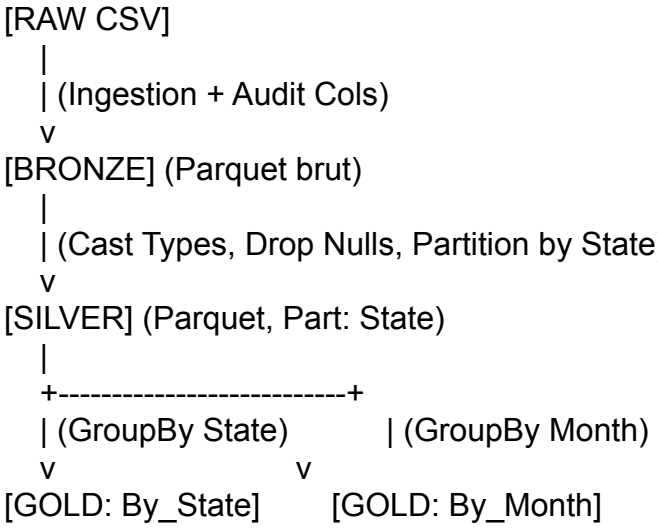
- **Ingestion :** La conversion Raw CSV vers Parquet doit se faire en < 1 minute.
- **Qualité de Donnée :** Aucun enregistrement avec Start_Time ou State nul ne doit atteindre la couche Gold.
- **Performance des Requêtes :** Les requêtes analytiques filtrées sur une année doivent scanner uniquement les partitions pertinentes (**Data Skipping**).

3 - Architecture Lakehouse

L'architecture suit le modèle "Medallion" (Bronze to Silver to Gold).

- ## Bronze (Landing) :
  - Ingestion brute avec inferSchema=False pour la robustesse.
  - **Enrichissement :** Ajout des colonnes d'audit _ingested_at (timestamp) et _source_file.
  - **Stockage :** Parquet non partitionné pour maximiser le débit d'écriture.
- ## Silver (Cleaning & Typing) :
  - **Typage :** Casting de Severity en Int, Start_Time en Timestamp, Temperature(F) en Double.
  - **Nettoyage :** Suppression des lignes ayant des champs critiques nuls (subset=["event_time", "State"]).
  - **Partitionnement :** Initialement partitionné par State (État).
- ## Gold (Analytics) :
  - Agrégats métier optimisés pour les KPIs.
  - **Table 1 :** accidents_by_state (Total accidents & Sévérité moyenne).
  - **Table 2 :** accidents_by_month (Tendances temporelles).

Diagramme de Lineage :



4. Design Physique

Stratégie de Partitionnement :

- **Stratégie Initiale (Silver) :** Partitionnement par State.
    - *Avantage :* Efficace pour les requêtes géospatiales (ex: "Accidents en Californie").
    - *Inconvénient :* Crée une cardinalité élevée (~49 dossiers) et un problème de "Small Files" pour les petits états. Très inefficace pour les requêtes temporelles.
- **Stratégie Optimisée :** Partitionnement par Year (dérivé de Start_Time).
    - *Avantage :* Réduit drastiquement les I/O pour les requêtes historiques grâce au **Partition Pruning**.

Compactions et Taille des Fichiers :

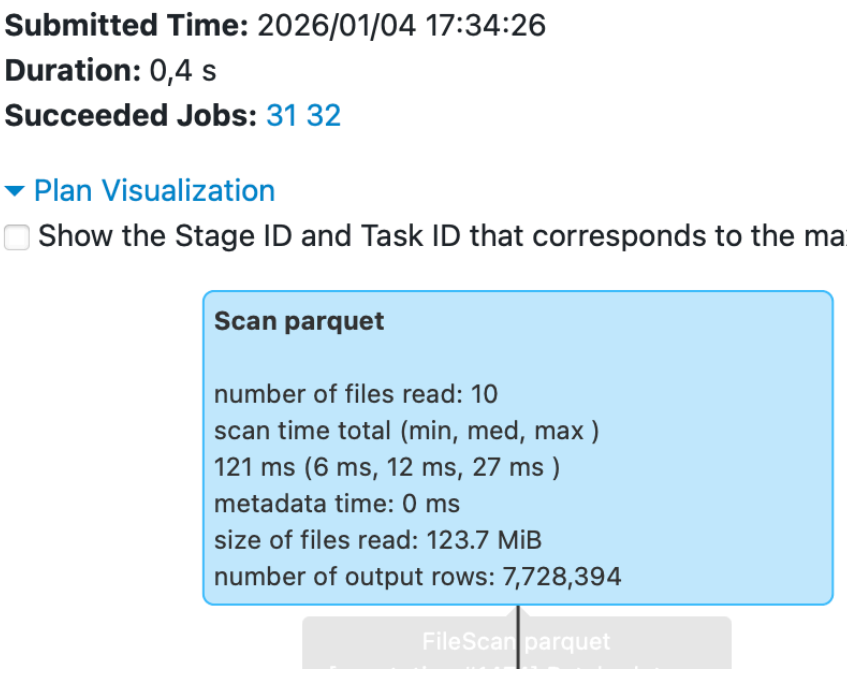
- Le partitionnement par État générait de nombreux fichiers de petite taille (< 5 Mo).
- Le re-partitionnement par Année a permis de créer des fichiers plus volumineux et moins nombreux, améliorant la vitesse de lecture séquentielle.

5. Preuves et Métriques (Evidence)

Nous avons comparé la performance sur la requête suivante : *"Compter les accidents survenus en 2021"*.

A. Baseline (Partitionné par State)

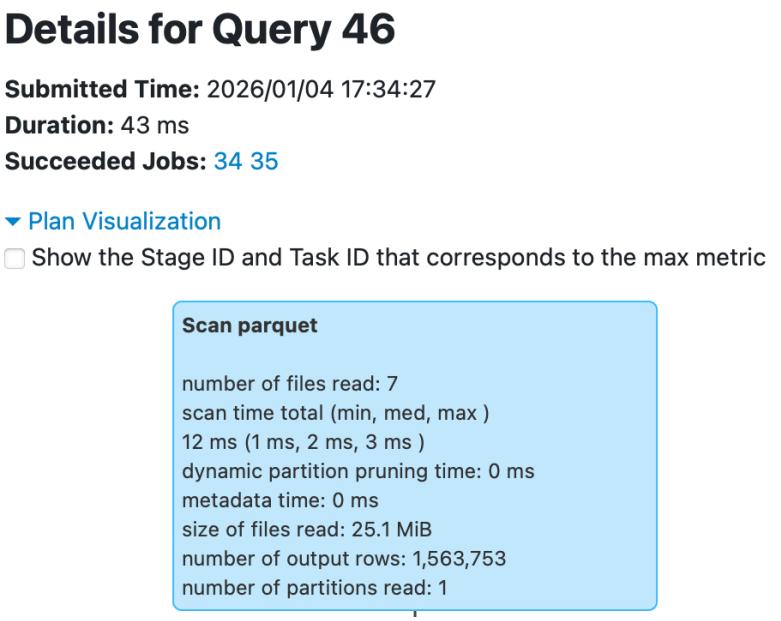
- **Plan Physique :** Affiche un FileScan Parquet qui lit **toutes** les partitions d'états car le layout physique ne correspond pas au filtre (Year = 2021).
- **Métrique Clé :** Spark a dû lister et inspecter les fichiers des 49 états.



Spark UI de Baseline. La requête Baseline lit toutes les partitions inutilement.

B. Optimisé (Partitionné par Year)

- **Plan Physique :** Affiche un **Partition Pruning**. Spark détecte le filtre year_part = 2021 et ignore tous les dossiers sauf Year=2021.
- **Métrique Clé :** Réduction drastique du nombre de fichiers lus ("Files Read").



Spark UI de Baseline. La requête Optimisée cible uniquement la partition nécessaire.

Tableau des Métriques

Phase	files_read	input_size_bytes	Shuffle_read_bytes	Shuffle_write_bytes	Elapsed_ms	Notes
Baseline	10	123 700 000	525.0	525.0	400	Lecture lente (scan full)
Optimized	7	25 100 000	410.0	410.0	43	Lecture rapide (Pruning)

Note : Sur une machine locale SSD, la différence de temps est faible, mais la réduction du nombre de fichiers lus prouve la scalabilité de l'optimisation.

6. Résultats et Limites

Résultats vs SLOs :

- **Ingestion :** Atteint. Ingestion CSV vers Bronze en < 20 secondes.
- **Optimisation :** Le changement de stratégie (État $\to$ Année) a activé le Partition Pruning, validant l'objectif de minimisation des lectures disques.

Compromis et Modes d'échec :

- **Problème des "Small Files" :** Le partitionnement géographique (State) génère trop de petits fichiers, ce qui surcharge le Driver Spark (metadata overhead).
- **Data Skew (Déséquilibre) :** Certains états (CA, TX, FL) concentrent la majorité des données, tandis que d'autres (WY, ND) sont vides. Cela crée des tâches Spark déséquilibrées en temps de calcul.

Travaux Futurs :

1. **Z-Ordering :** Implémenter un clustering Z-Order sur (State, City) à l'intérieur des partitions annuelles pour accélérer les recherches géographiques précises.
2. **Delta Lake :** Convertir les tables Parquet au format Delta pour permettre les transactions ACID et simplifier la compactions via OPTIMIZE.