Objet : Fonctionnement des tapes Spark bases sur AbstractStep et AbstractStepMap

Bonjour [Nom du destinataire],

Je vous partage une explication sur la conception des tapes (steps) Spark dans notre pipeline, structures autour des classes abstraites AbstractStep<T> et AbstractStepMap<T>.

AbstractStep<T>

Cette classe est utilise pour dfinir une tape Spark gnrique. Elle permet denchaner dynamiquement des traitements de chargement ou de sauvegarde sur un Dataset<T>. En ltendant, on peut :

- Charger les donnes depuis nimporte quelle source (CSV, Parquet, Hive, etc.)
- Excuter une logique personnalise dans la mthode launch
- Exporter les donnes transformes

Exemples :

- ArivaLoad : chargement dun fichier CSV et mapping vers des objets Ariva
- ArivaSave : export dun Dataset<Ariva> au format Parquet

AbstractStepMap<T>

Cette classe est utilise lorsquon souhaite modifier chaque ligne dun Dataset<T>. Elle fournit la mthode execute(T row) qui sapplique chaque lment.

Exemple :

- ArivaTransform : modification de champs ligne par ligne, comme lajout dune date dingestion.

Orchestration des tapes

Chaque step est annot avec @Step(StepConstants.XXX), ce qui permet au systme de les excuter automatiquement dans lordre dfini par les constantes. On peut donc enchaner plusieurs transformations, chargements et exports sans gestion manuelle de lordre dexcution.

Cela garantit une architecture modulaire, testable et extensible pour le traitement de nos fichiers.

Bien cordialement,

[Votre Prnom Nom]

Subject: Understanding Spark pipeline with AbstractStep and AbstractStepMap

Hi [Recipient's Name],

Heres a detailed explanation of how our Spark steps are designed using the abstract classes AbstractStep<T> and AbstractStepMap<T>.

AbstractStep<T>

This class defines a general-purpose Spark step. It enables dynamic chaining of load or export operations on a Dataset<T>. When extended, it allows you to:

- Load data from any source (CSV, Parquet, Hive, etc.)
- Customize logic inside the launch method
- Export the transformed dataset

Examples:

- ArivaLoad: loads a CSV file and maps it to Ariva objects
- ArivaSave: exports a Dataset<Ariva> to Parquet format

AbstractStepMap<T>

This class is used when we need to process each record of a dataset individually. It provides an execute(T row) method applied to each element.

Example:

- ArivaTransform: modifies individual fields, such as adding an ingestion date to each record.

Step orchestration

Each step is annotated with @Step(StepConstants.XXX), allowing the system to execute them in a predefined order. This way, multiple transformations, loads, and exports can be chained without manually handling execution sequence.

This structure ensures a modular, testable, and extensible Spark pipeline.

Best regards,

[Your First Name Last Name]