Comparaison RDD vs DataFrame

- **RDD** : API bas niveau, flexible, contrôle total sur les transformations et actions, mais moins optimisé.
- **DataFrame** : API haut niveau, optimisations automatiques par Catalyst et Tungsten, plan d'exécution optimisé, meilleure performance sur gros volumes de données.

Clarté et trade-offs

- RDDs permettent un contrôle précis sur les opérations, mais nécessitent plus de code et plus de mémoire.
- DataFrames réduisent le code, optimisent automatiquement les shuffles et projections, et sont mieux adaptés pour l'analyse SQL-like et l'exploitation des optimisations du moteur Spark.

Quand utiliser l'une ou l'autre API

- **RDD** : pour des traitements très personnalisés ou transformations complexes non supportées par les DataFrames.
- **DataFrame** : pour la majorité des analyses, agrégations et opérations SQL, ou lorsque la performance est critique.

Justification des choix pour les optimisations

- **Broadcast** : utilisé sur des petites tables pour éviter les shuffles coûteux.
- **Projection minimale** : sélectionner uniquement les colonnes nécessaires réduit la quantité de données lues et le volume de shuffle, améliorant la performance.
- **Caching** : les DataFrames ou RDD utilisés plusieurs fois peuvent être mis en cache pour éviter de recalculer les transformations.