
DIGGIBYTE
CREATING VALUE WITH DATA

DIGGIBYTE
CREATING VALUE WITH DATA

RDD and Dataframe Transformations

Presented by,
Amit Kundu

DIGGIBYTE
CREATING VALUE WITH DATA

Agenda

- Introduction
- Difference between RDDs and DataFrames
- Transformations on RDDs
- Transformations on DataFrames
- Transformations Specific to RDDs
- Transformations Specific to DataFrames

DIGGIBYTE
CREATING VALUE WITH DATA

Introduction

- Apache Spark provides two main data abstractions: Resilient Distributed Datasets (RDDs) and DataFrames
- Transformations are operations applied to RDDs or DataFrames to create new RDDs or DataFrames
- Some transformations are available for both RDDs and DataFrames, while others are specific to each abstraction

DIGGIBYTE
CREATING VALUE WITH DATA

Difference between RDDs and DataFrames

- RDDs represent unstructured distributed collections of objects
- DataFrames represent structured, tabular data with a defined schema
- RDD transformations operate on individual elements within partitions, making them suitable for fine-grained control and processing any type of data, including unstructured and complex types. DataFrame transformations, on the other hand, work at a higher level, operating on columns and rows of structured data, providing optimizations for tabular data processing and a more SQL-like interface.
- The difference in data representation and semantics makes certain transformations only applicable to one abstraction

DIGGIBYTE
CREATING VALUE WITH DATA

Transformations on RDDs

- map: Applies a function to each element of the RDD
- filter: Selects elements from the RDD that pass a condition
- flatMap: Applies a function to each element and flattens the resulting sequences
- union: Combines two RDDs into a new RDD
- intersection: Returns a new RDD containing only the elements present in both RDDs
- distinct: Returns a new RDD with duplicate elements removed
- sample: Returns a random sample of elements from the RDD
- repartition: Reshuffles the data across a different number of partitions

DIGGIBYTE
CREATING VALUE WITH DATA

Transformations on DataFrames

- map (on DataFrame RDD): Applies a function to each row of the DataFrame
- filter: Selects rows from the DataFrame that pass a condition
- flatMap (on DataFrame RDD): Applies a function to each row and flattens the resulting sequences
- union: Combines two DataFrames into a new DataFrame
- intersect: Returns a new DataFrame containing only the rows present in both DataFrames
- distinct: Returns a new DataFrame with duplicate rows removed
- sample: Returns a random sample of rows from the DataFrames
- repartition: Reshuffles the data across a different number of partitions

DIGGIBYTE
CREATING VALUE WITH DATA

Transformations Specific to RDDs

- mapPartitions: Applies a function to each partition of the RDD
- mapPartitionsWithIndex: Applies a function to each partition of the RDD with the partition index
- coalesce: Decreases the number of partitions in the RDD

These transformations are specific to RDDs because they operate on the partitions of the RDD, which is a lower-level concept not applicable to DataFrames.

DIGGIBYTE
CREATING VALUE WITH DATA

Transformations Specific to DataFrames

- join: Joins two DataFrames based on a specified condition
- sort: Sorts the DataFrame based on specified columns
- alias: Renames a column in the DataFrame
- withColumn: Adds a new column or updates an existing column in the DataFrame
- drop: Drops specified columns from the DataFrame

These transformations are specific to DataFrames because they operate on the structured, tabular data and schema of the DataFrame, which is not applicable to the unstructured RDD abstraction.

DIGGIBYTE
CREATING VALUE WITH DATA

Summary

- RDDs and DataFrames offer a range of transformations for data processing
- Some transformations are common to both, while others are specific to each abstraction
- The choice between RDDs and DataFrames depends on the requirements of the data processing task and the level of control needed
- In general, DataFrames are preferred for structured data processing, while RDDs can be used for more low-level or custom data processing tasks

DIGGIBYTE

CREATING VALUE WITH DATA

THANK YOU !!