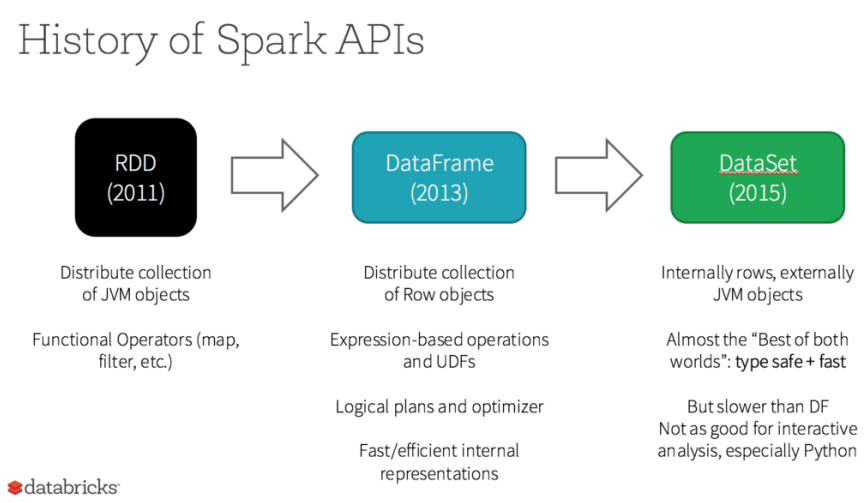
(/)

Resilient Distributed Dataset (RDD)

Back to glossary (/glossary/)



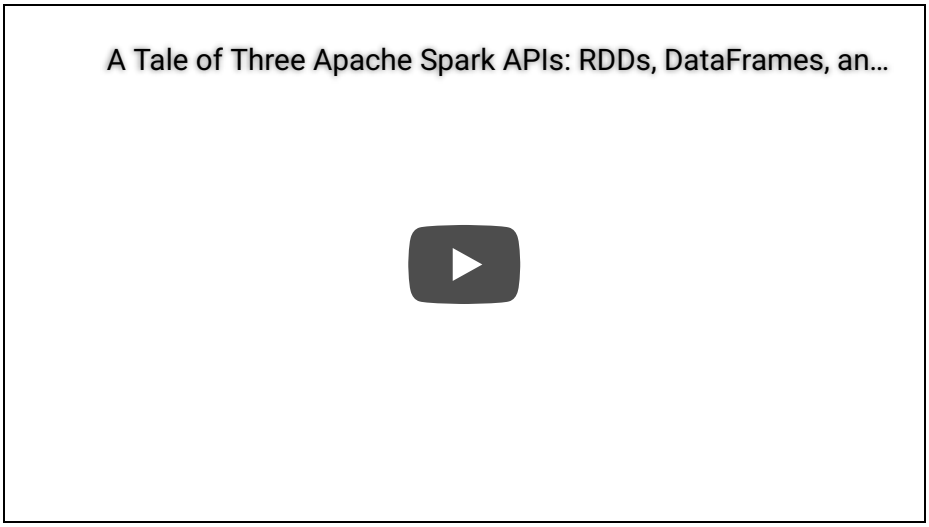
RDD was the primary user-facing API in Spark since its inception. At the core, an RDD is an immutable distributed collection of elements of your data, partitioned across nodes in your cluster that can be operated in parallel with a low-level API that offers transformations and actions.

5 Reasons on When to use RDDs

1. You want low-level transformation and actions and control on your dataset;
2. Your data is unstructured, such as media streams or streams of text;
3. You want to manipulate your data with functional programming constructs than domain specific expressions;
4. You don't care about imposing a schema, such as columnar format while processing or accessing data attributes by name or column; and
5. You can forgo some optimization and performance benefits available with DataFrames and Datasets for structured and semi-structured data.

What happens to RDDs in Apache Spark 2.0?

Are RDDs being relegated as second class citizens? Are they being deprecated? The answer is a resounding NO! What's more is you can seamlessly move between DataFrame or Dataset and RDDs at will—by simple API method calls—and DataFrames and Datasets are built on top of RDDs.



Additional Resources

- **A Tale of Three Apache Spark APIs: RDDs vs DataFrames and Datasets (https://databricks.com/blog/2016/07/14/a-tale-of-three-apache-spark-apis-rdds-dataframes-and-datasets.html)**
- **Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing Research Paper (https://databricks.com/research/resilient-distributed-datasets-a-fault-tolerant-abstraction-for-in-memory-cluster-computing-nsdi-2012)**

Back to glossary (/glossary/)