Assignment 3

Negin Baghbanzadeh

Explain the main difference between RDDs, DataFrames, and DataSets.

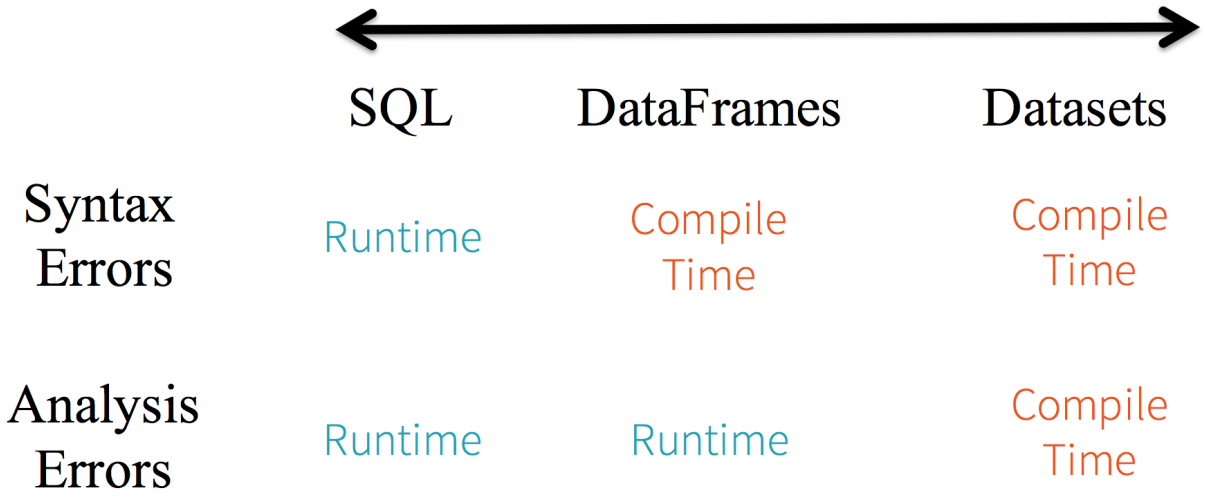
At the core, an RDD is an immutable distributed collection of data elements, partitioned across nodes in a cluster that can be operated in parallel with a low-level API that offers transformations and actions. Data in RDD is unstructured so if our data doesn't have a specific structure, using RDD will be a good idea. RDDs can be used when low-level transformation and actions and control of the dataset are needed. Using RDDs offers low-level transformation and actions and control of the dataset.

Like an RDD, a DataFrame is an immutable distributed collection of data. Unlike an RDD, data is organized into named columns, like a table in a relational database. Designed to make large data sets processing even easier, DataFrame allows developers to impose a structure onto a distributed collection of data, allowing higher-level abstraction; it provides a domain-specific language API to manipulate your distributed data; and makes Spark accessible to a wider audience, beyond specialized data engineers. DataFrames are untyped.

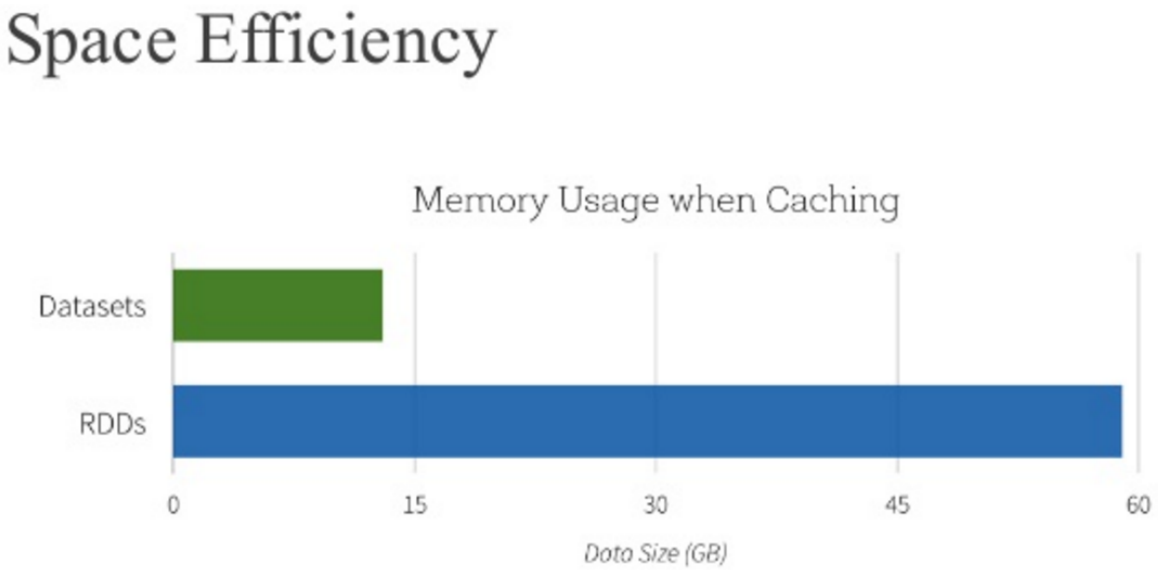
Datasets are like DataFrames but strongly typed (This means for example that the contents of a column of the table it represents are of a single type, like a relational table. DataSets may be less convenient for exploratory data analysis but are more robust for business applications).

Starting in Spark 2.0, Dataset takes on two distinct APIs characteristics: a *strongly-typed* API and an *untyped* API.

when using Datasets, analysis error can be detected at compile-time, hence saving developer time and costs. But when using DataFrames, an analysis error is detected at run time.



Space efficiency and performance gains are another benefit of using DataFrames and Dataset APIs over RDDs.



When should we use DataSets and/or DataFrames:

1. If we want rich semantics, high-level abstractions, and domain-specific APIs.
2. If we're processing demands high-level expressions, filters, maps, aggregation, averages, sum, SQL queries, columnar access, and use of lambda functions on semi-structured data.
3. If we want a higher degree of type safety at compile-time, we should use DataSets.

When Should we use RDDs:

1. If we have low-level transformation and actions and control on the dataset.
2. If our data is unstructured.

Link to Question 2:

https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/2901214958767 16/3412613163418 85/4913019624089264/latest.html