Summary & Highlights

In this module, you learned that:

* RDDs are Spark's primary data abstraction partitioned across the nodes of the cluster​. Transformations leave existing RDDs intact and create new RDDs based on the transformation function​. With a variety of available options, apply functions to transformations perform operations. Next, actions return computed values to the driver program. Transformations undergo lazy evaluation, meaning they are only evaluated when the driver function calls an action.
* A dataset is a distributed collection of data that provides the combined benefits of both RDDs and SparkSQL​. Consisting of strongly typed JVM objects​, datasets make use of DataFrame typesafe capabilities and extend object-oriented API capabilities. Datasets work with both Scala and Java APIs​.  DataFrames are not typesafe. You can use APIs in Java, Scala, Python. Dataset​s are Spark's latest data abstraction.
* The primary goal of Spark SQL Optimization is to improve the run-time performance of a SQL query, by reducing the query’s time and memory consumption, saving organizations time and money. ​Catalyst is the Spark SQL built-in rule-based *query* optimizer.​ Catalyst performs analysis, logical optimization, physical planning, and code generation.​ Tungsten is the Spark built-in cost-based optimizer for CPU and memory usage that enables cache-friendly computation of algorithms and data structures.
* Basic DataFrame operations are reading, analysis, transformation, loading, and writing. ​You can use a Pandas DataFrame in Python to load a dataset and apply the print schema, select function, or show function for data analysis. ​For transform tasks, keep only relevant data and apply functions such as filters, joins, column operations, grouping and aggregations, and other functions.
* Spark SQL consists of Spark modules for structured data processing that can run SQL queries on Spark DataFrames and are usable in Java, Scala, Python and R. Spark SQL supports both temporary views and global temporary views​. Use a DataFrame function or an SQL Query + Table View for data aggregation. Spark SQL supports Parquet files, JSON datasets and Hive tables​.