**Lower-Level API vs Higher-Level API**

Lower-Level API

The lowest abstraction layer that gives you direct control over data and computation logic.

In Spark:  
 The RDD (Resilient Distributed Dataset) API is the lower-level API.

Features

* Fine-grained control over operations (map, filter, reduce, etc).
* No built-in optimizations (you manage logic yourself).
* More flexible → can handle unstructured data or complex transformations.
* Requires more code to accomplish tasks.
* Error-prone — you need to manage performance, serialization, partitioning.

When to use

* When you need custom processing logic that DataFrames/Datasets can’t express.
* When working with raw, unstructured, or complex nested data.
* When you want full control over partitioning, caching, or data flow.

Higher-Level API

A more abstract, user-friendly layer that simplifies data processing.

In Spark:  
The DataFrame, Dataset, and Spark SQL APIs are higher-level APIs.

Features

* Declarative → you specify *what* to do, not *how* to do it.
* Backed by optimizers (e.g. Catalyst) → Spark rewrites queries for performance.
* Easier to write, read, and maintain code.
* Integrates with SQL; BI-friendly.
* Better memory and CPU efficiency (Tungsten engine, encoders).

When to use

* When working with structured/semi-structured data.
* When you want Spark to optimize execution plans.
* When doing SQL-like operations (filter, select, groupBy, join).