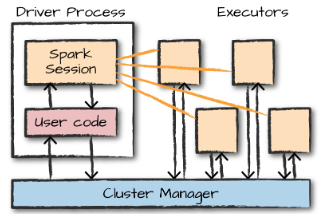
# Apache Spark

Apache Spark is a unified computing engine and a set of libraries for parallel data processing on computer clusters.

# Architecture



The driver process runs your main() function, sits on a node in the cluster, and is responsible for three things:

* maintaining information about the Spark Application
* responding to a user’s program or input
* analyzing, distributing, and scheduling work across the executors (discussed momentarily).

The executors are responsible for executing code assigned to it by the driver, and reporting the state of the computation on that executor back to the driver node.

## The SparkSession

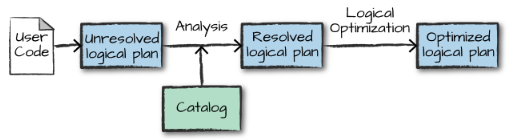
You control your Spark Application through a driver process called the SparkSession. The SparkSession instance is the way Spark executes user-defined manipulations across the cluster.

## Partitions

To allow every executor to perform work in parallel, Spark breaks up the data into chunks called partitions. A partition is a collection of rows that sit on one physical machine in your cluster. If you have one partition, Spark will have a parallelism of only one, even if you have thousands of executors. If you have many partitions but only one executor, Spark will still have a parallelism of only one because there is only one computation resource.

## Logical Planning

In Apache Spark, query processing involves multiple stages, starting with logical planning, where the query is parsed, analyzed, and optimized to produce an optimized plan[3](https://www.chashnikov.dev/post/spark-understanding-physical-plans). The logical plan is an abstract representation of the transformations needed without detailing how they are executed on the driver or worker nodes. The Catalyst Optimizer optimizes the plan by applying its own rules. For instance, it checks which tasks can be computed together in one stage, determines the execution order of queries for better performance in multi-join queries, and optimizes the query by evaluating the filter clause before any project.



## Physical Planning

The logical plan describes *what* needs to be done, using relational operators like Filter and Join, along with respective expressions[3](https://www.chashnikov.dev/post/spark-understanding-physical-plans). Physical planning then determines *how* the logical plan will be executed on the cluster[2](https://blog.knoldus.com/understanding-sparks-logical-and-physical-plan-in-laymans-term/). It specifies the execution sequence of operations like filter, where, and group By clauses. It determines the specific algorithms and strategies for executing the query, such as the order of joins and the type of join, … . The physical plan operates on Resilient Distributed Datasets. Spark evaluates multiple physical plans and selects the most optimal one based on a cost model that estimates execution time and resource usage.

