

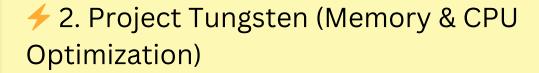
 PySpark provides an API that allows users to work with Apache Spark using Python. It enables handling big data efficiently with a distributed computing framework.

from pyspark.sql import SparkSession

spark =
SparkSession.builder.appName("PySpark
Example").getOrCreate()

Creating a DataFrame
data = [("Alice", 25), ("Bob", 30),
("Charlie", 35)]
df = spark.createDataFrame(data,
["Name", "Age"])

Using SQL-like operations df.filter(df.Age > 28).show()



- Project Tungsten is Spark's effort to improve execution speed by optimizing memory, CPU efficiency, and code generation.
- Why Tungsten Matters?
- ✓ Up to 10x faster than older Spark versions
- ✓ Reduces garbage collection overhead
- ✓ Minimizes serialization & deserialization cost

spark.conf.set("spark.sql.execution.arr
ow.pyspark.enabled", "true")
df = spark.range(1,
1000000).toDF("num")
df.selectExpr("num * 2 as double_num").show()

3. Catalyst Optimizer (Query Optimization Engine)

 Catalyst Optimizer is Spark's query optimization engine that improves execution plans for better performance.



1. Analysis:-

Parses query, resolves column names, and verifies data types.

- 2. Logical Optimization:-Rewrites query for efficiency (e.g., pushing filters down).
- 3. Physical Planning:-Generates an optimized execution plan.
- 4. Code Generation (Tungsten Integration):-

Generates JVM bytecode for efficient execution.