Python & Spark ⟶ Pyspark

1. GCP Dataproc
2. Terminal of dataproc
3. Google Colab
4. Databricks

Python map reduce higher order fn

Reading & Partioning Data in Spark

Data is read and divided into partition

What is partitioning in Spark?

> Logical chunks of data

Processing

large dataset is divided into smaller chunks
called partition.

1. when already divided into blocks → hdfs, S3, lake gen 2

2. read data & create partition based on the configure.

execution of one partition of data = one task
= 1 cpu core required for execution

≈ Mapper task

(executor) performs a task & it can have more than 1 core as well

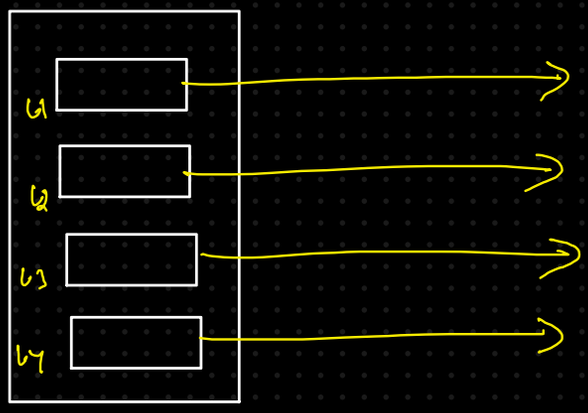
8 blocks (1gb) → 8 partition → 8 tasks

- inc. → i will get more ||ism
- dec → less parallelism

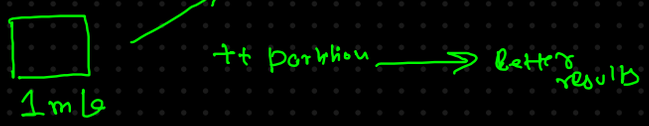
Spark parallelize

Subset of data

we can create an rdd using parallelize & then work on it



< 128 mb
↓
2 partition ⇒ 2 cores



→ Read the data → Parallelize & partition, count key value

→ Transformations → map, reduce, filter, reduceByKey

Reduce By Key

transformation

Count key value

action

Notebook → functions in pyspark

reducebyKey 5 keys → 5 values

map 100 rows → 100

filter 100 rows → 0 — 100

This is still the most diff. way of writing code in spark
⇒ more easy way

. Save as text file (hdfs)

Transformation in Spark : Narrow vs wide

transformations are operation that create a new rdd from existing one

Narrow

data is not shuffled across partition

faster → no shuffle

map, filter, flatmap

dependent on one parent partition

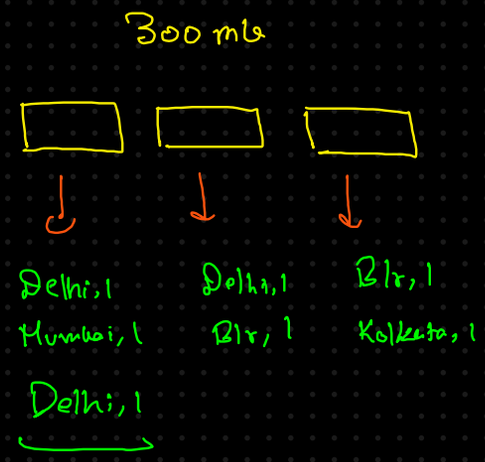
Wide

Data is shuffled across our partitions

Slower – as shuffling ⇒ I/O network

groupbyKey
reduceByK
SortbyKey

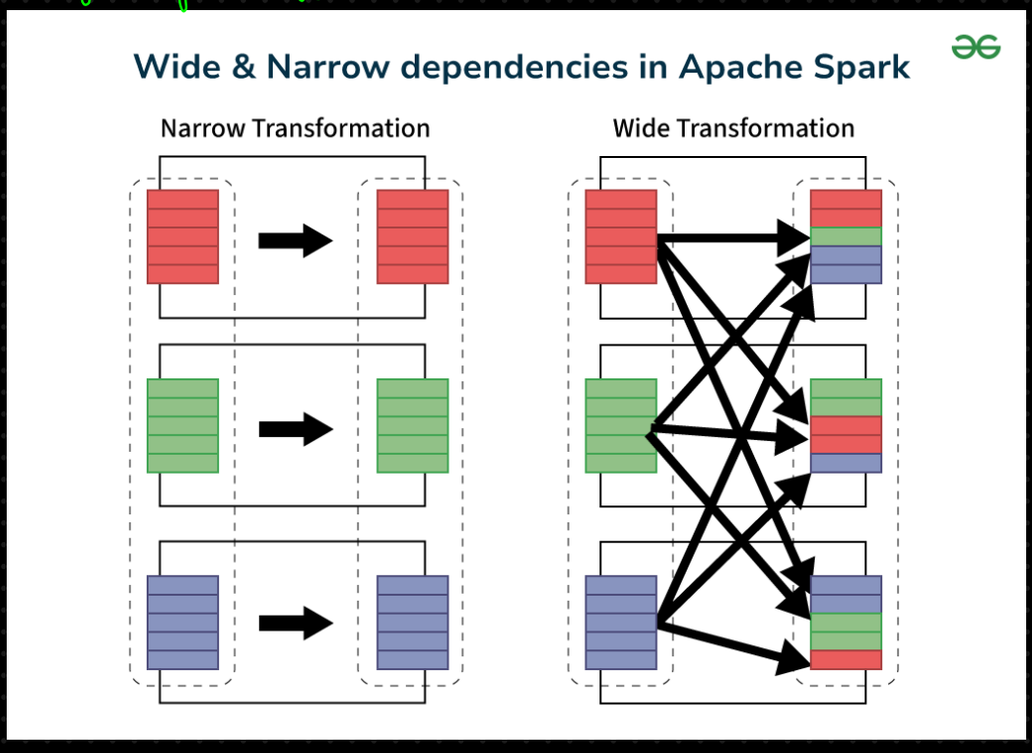
dependent on multiple partition



either

→ try to have very less wide transformation

→ narrow down the data so that data getting shuffled is less.



Job ⟶ Stage ⟶ tasks

||

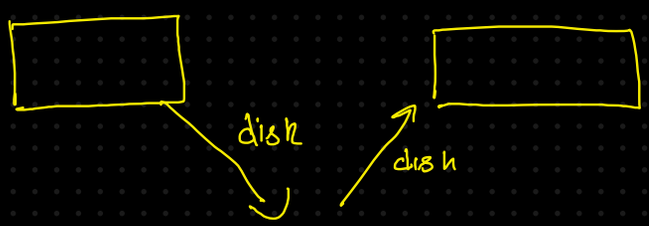
||

action ⟶ wide transformation ⟶ partitions

+

1

When we do a wide transformation ⇒ a stage is created



→ homework

→ just run what I did & understand spark UI

→ be comfortable in pyspark

from next class, lots of code without major break

Reduce By Key & Group by Key