a. What benefits does Spark offer when processing large-scale datasets like the NYC Taxi data compared to traditional pandas-based processing?

Spark lets you spread the work across many CPU cores (or even many machines) instead of cramming everything into one laptop. It keeps most data in memory and runs tasks in parallel, so heavy jobs—like crunching months of NYC taxi trips—finish way faster and don't blow up your RAM the way a single-threaded pandas' script might.

b. Why is it important to perform data cleaning (e.g., removing invalid trips) before aggregation and analysis?

Data cleaning matters because inaccurate entries—like invalid trips with zero distance or negative fares—can mess up your results. Removing these ensures your aggregations reflect real, meaningful data, not errors or noise.

c. How do Spark's lazy evaluation and distributed computation contribute to performance efficiency?

With lazy evaluation, Spark waits to run code until you call an action (like show () or collect ()), so it can look at the whole plan and optimize it in one shot. Then it breaks the job into smaller tasks and spreads them out across all available cores, so everything runs in parallel—faster results, less sitting around waiting.