Goal: Join 3 tables, customers, orders, and zipcodes CSV using explicitly hinting Spark to use shuffle hash and sort merge join, and compare execution times and DAGs

Shuffle Hash Join

1. The result of joining the 3 tables using shuffle hash first shuffles all tables before creating a hash of the smaller table, in our case, the customer table.
2. The maximum time for the shuffle step was between 7 and 15 seconds for both the customer and orders tables on a 2 GB per executor configuration with 2 workers.

Sort Merge Join

1. The result of joining three tables using a sort merge join explicitly was a shuffle step and a sort step on a 2 GB per-core executor with two workers.
2. The max execution time for the shuffle and sort step was more than 8 min with 313 shuffle partitions.
3. The final zipcode dataset was joined using broadcast join, which the Spark optimizer auto-selected

In summary, even though sort-merge join is more resilient and fault-tolerant, as there is no risk of OOM, as nothing is stored in-memory, it is a resource-intensive join strategy and should be avoided.