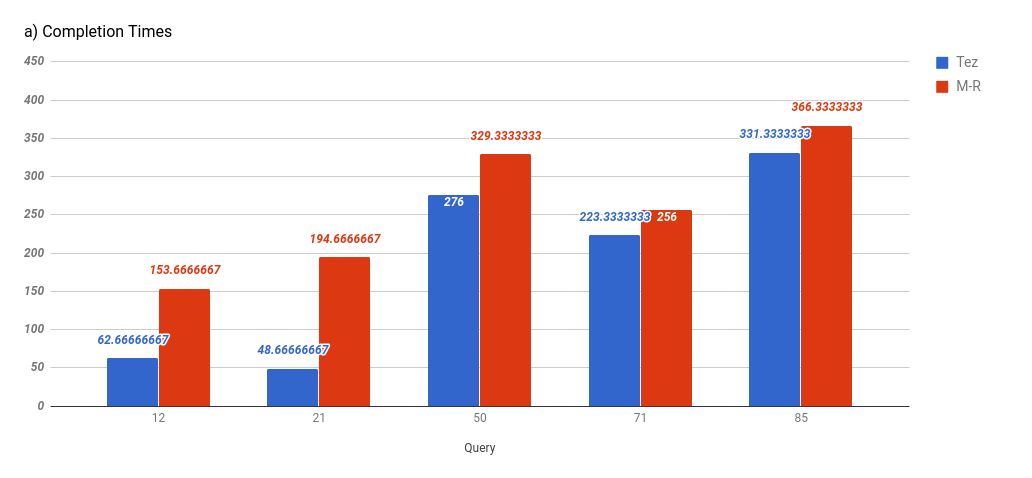
PArt A

1a.

|  |  |  |
| --- | --- | --- |
| Query | Tez | M-R |
| 12 | 62.66666667 | 153.6666667 |
| 21 | 48.66666667 | 194.6666667 |
| 50 | 276 | 329.3333333 |
| 71 | 223.3333333 | 256 |
| 85 | 331.3333333 | 366.3333333 |



Tez is always faster than Map reduce.  
Due to the architecture of Tez the job's steps are computed before execution and the system can cache intermediate job results in memory. But, in MapReduce all intermediate data between MapReduce phases are written to HDFS i.e. disk adding latency.

Part 1b:

Network bandwidth in bytes

|  |  |  |
| --- | --- | --- |
| Tez |  | Avg |
|  | 12 | 6,361,292,950,666,670,000 |
|  | 21 | 1,157,242,779,666,670,000 |
|  | 50 | 24,489,100,656,333,300,000 |
|  | 71 | 35,492,911,377,000,000,000 |
|  | 85 | 18,967,835,682,666,700,000 |
|  |  |  |
| MR | 12 | 8,009,428,532,500,000,000 |
|  | 21 | 6,430,727,694,333,330,000 |
|  | 50 | 30,626,058,147,666,700,000 |
|  | 71 | 35,426,918,164,000,000,000 |
|  | 85 | 14,537,329,973,000,000,000 |

Disk bandwidth:

|  |  |  |
| --- | --- | --- |
| Query | TEZ | MR |
| 12 | 418181120 | 5736886272 |
| 21 | 22323200 | 4653715456 |
| 50 | 8594477056 | 19173908480 |
| 71 | 69925629952 | 10406600704 |
| 85 | 122368 | 7858442240 |

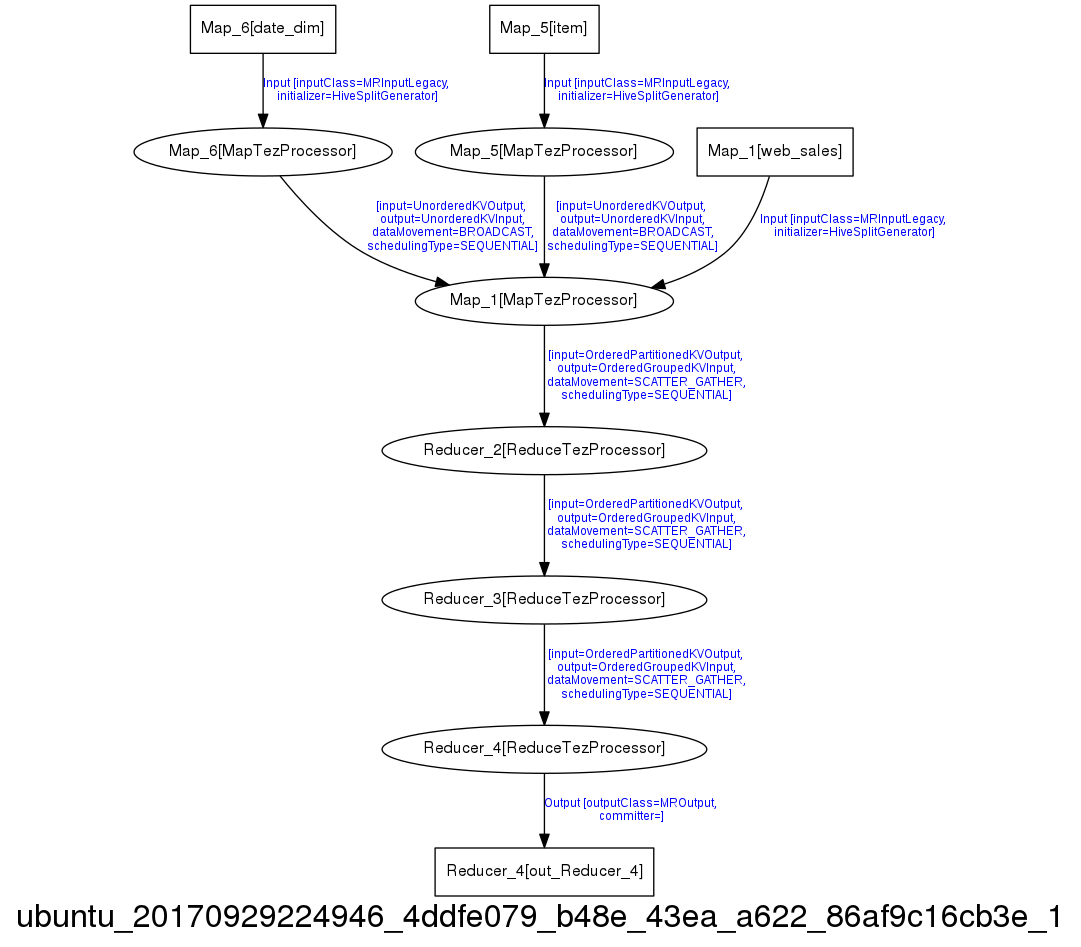
Part 1c:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tasks |  | Aggregate(red) | Read from HDFS(map) | Ratio | Total |
| MR | 12 | 3 | 39 | 0.07692307692 | 42 |
|  | 21 | 2 | 20 | 0.1 | 22 |
|  | 50 | 2 | 98 | 0.02040816327 | 100 |
|  | 71 | 2 | 173 | 0.01156069364 | 175 |
|  | 85 | 8 | 51 | 0.1568627451 | 59 |
|  |  |  |  |  |  |
| Tez | 12 | 7 | 3 | 2.333333333 | 10 |
|  | 21 | 7 | 4 | 1.75 | 11 |
|  | 50 | 9 | 5 | 1.8 | 14 |
|  | 71 | 8 | 14 | 0.5714285714 | 22 |
|  | 85 | 8 | 14 | 0.5714285714 | 22 |

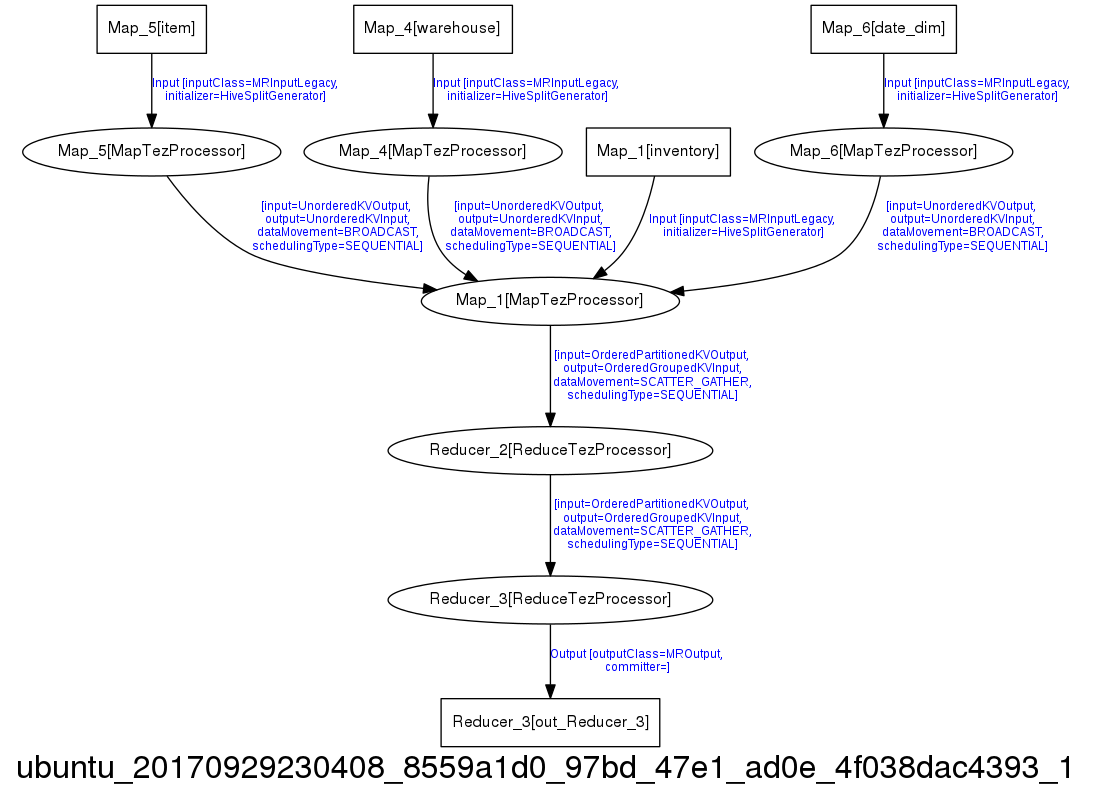
Part 1D:

Tez:

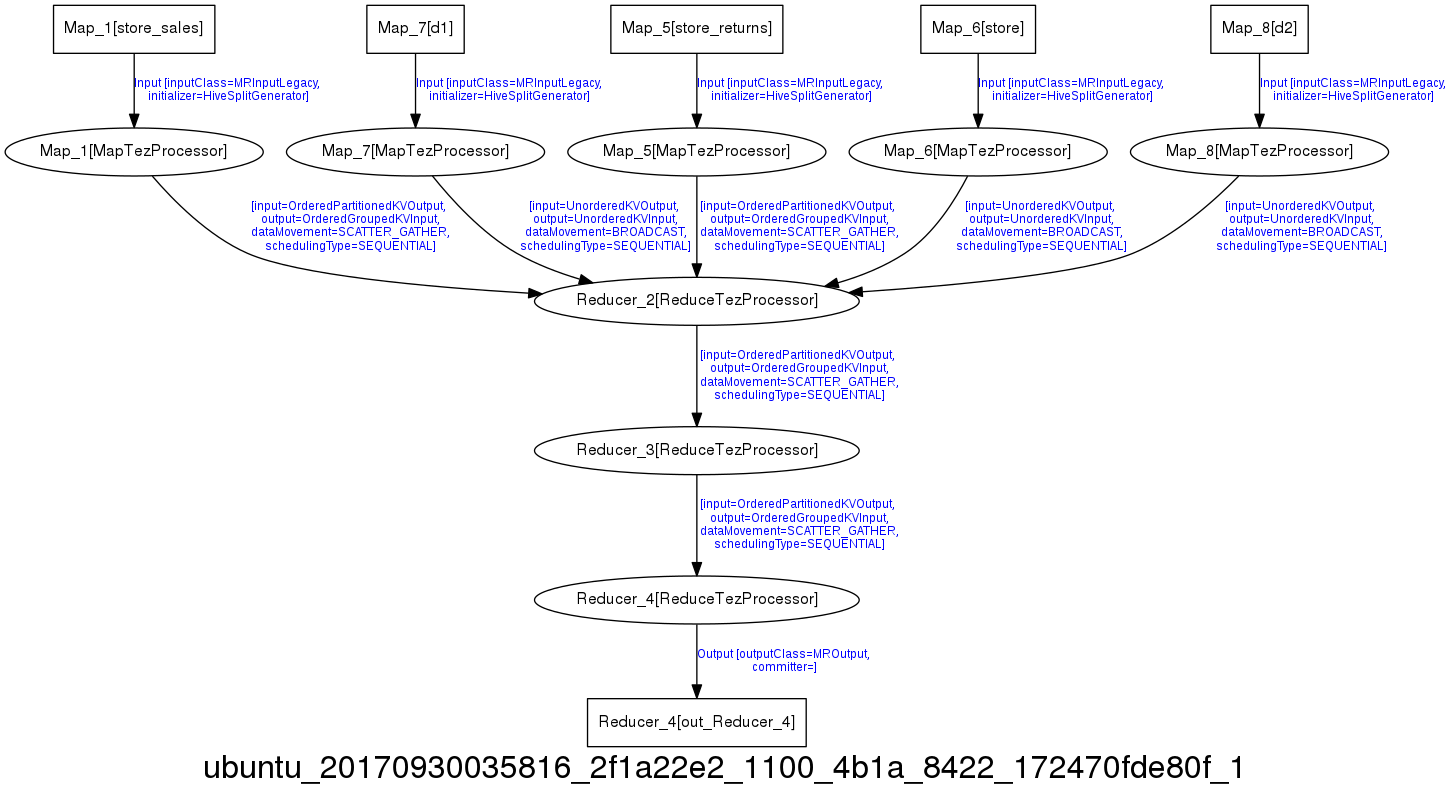
Query 12:



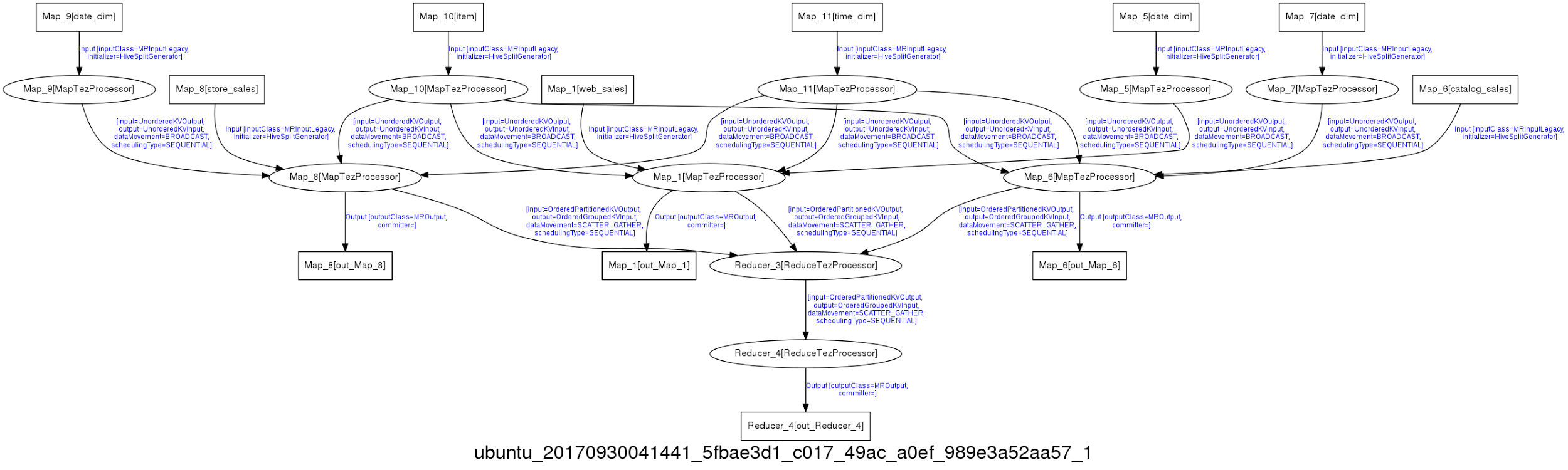
Query 21:



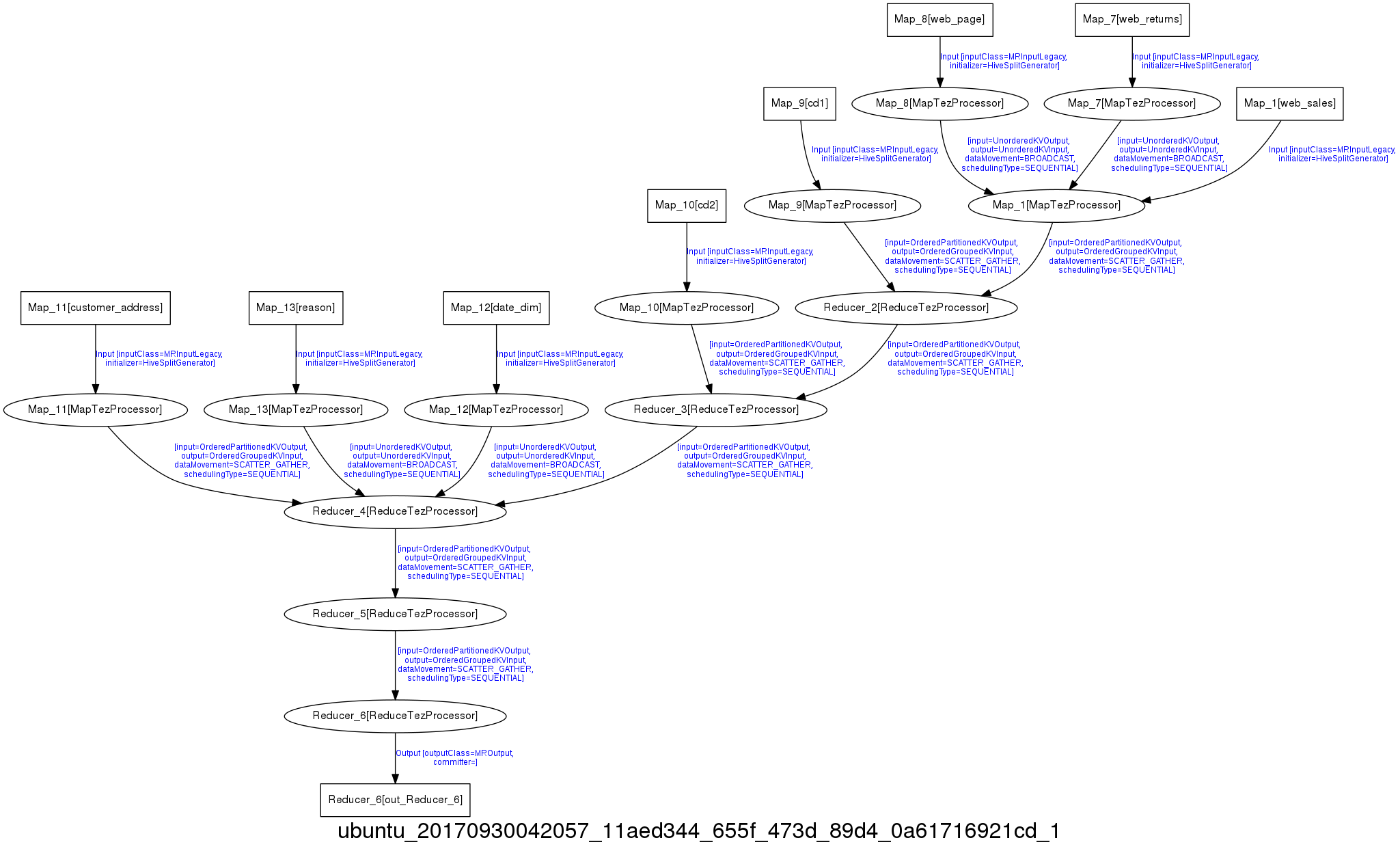
Query 50



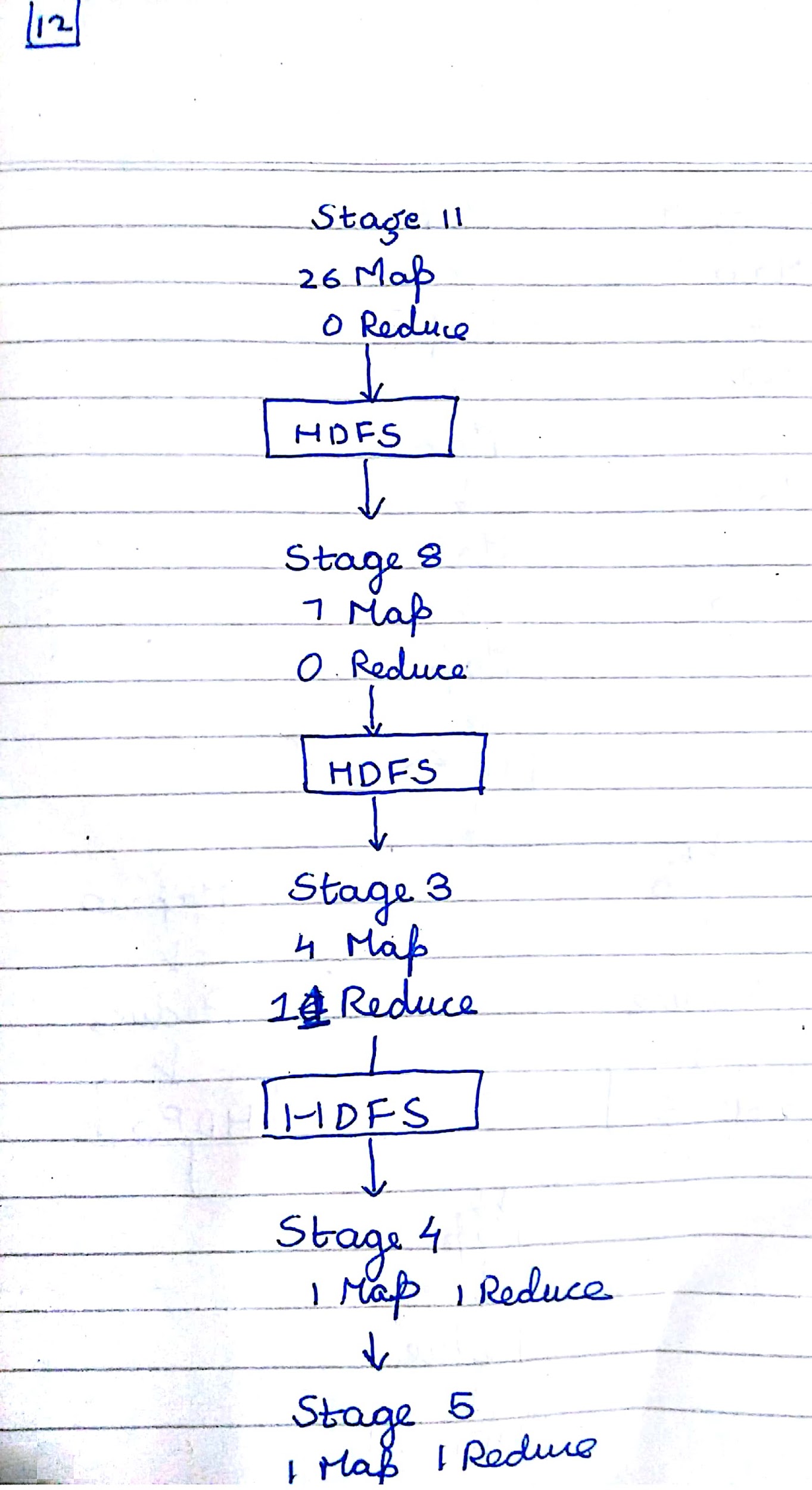
Query 71:



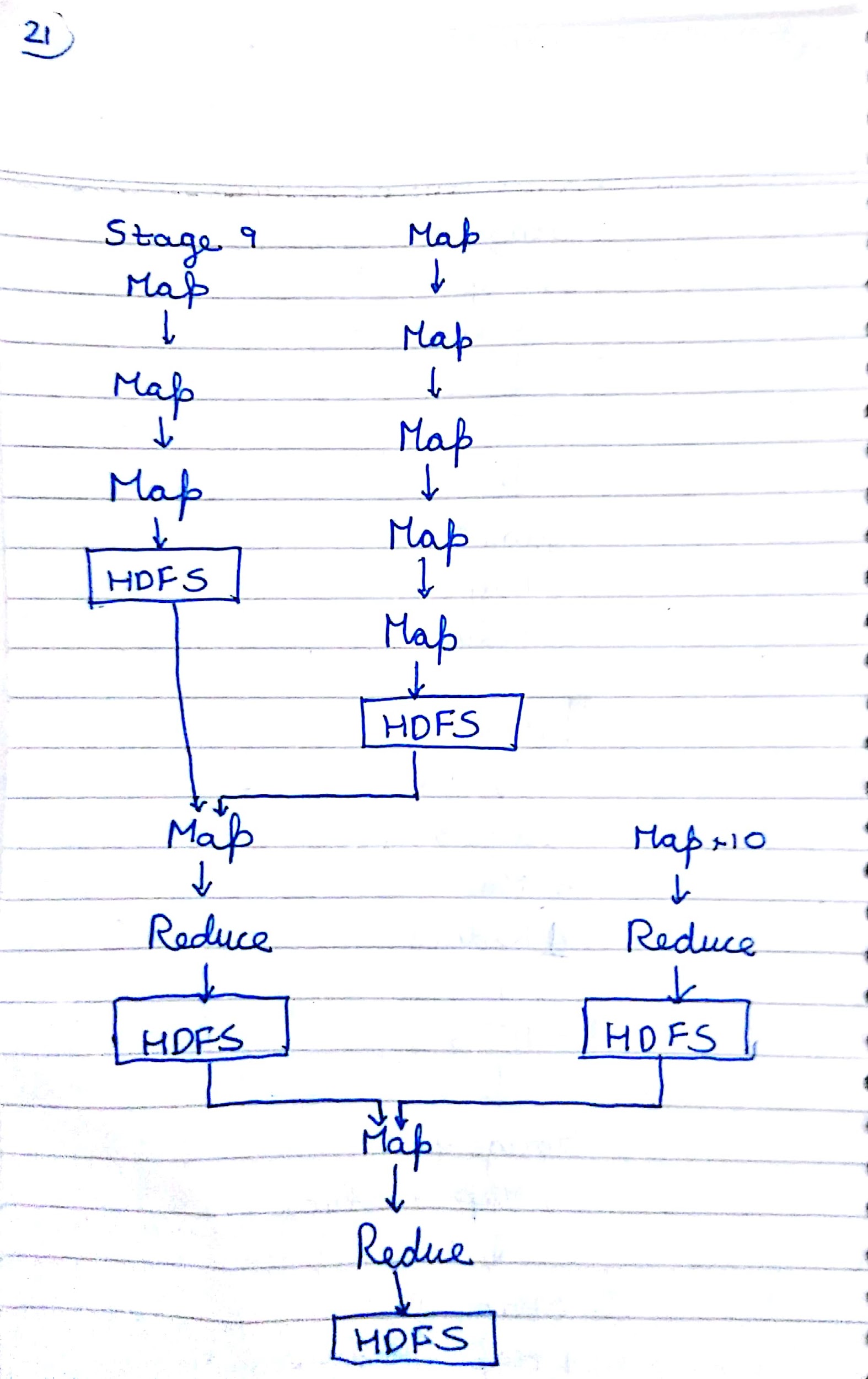
Query 85:



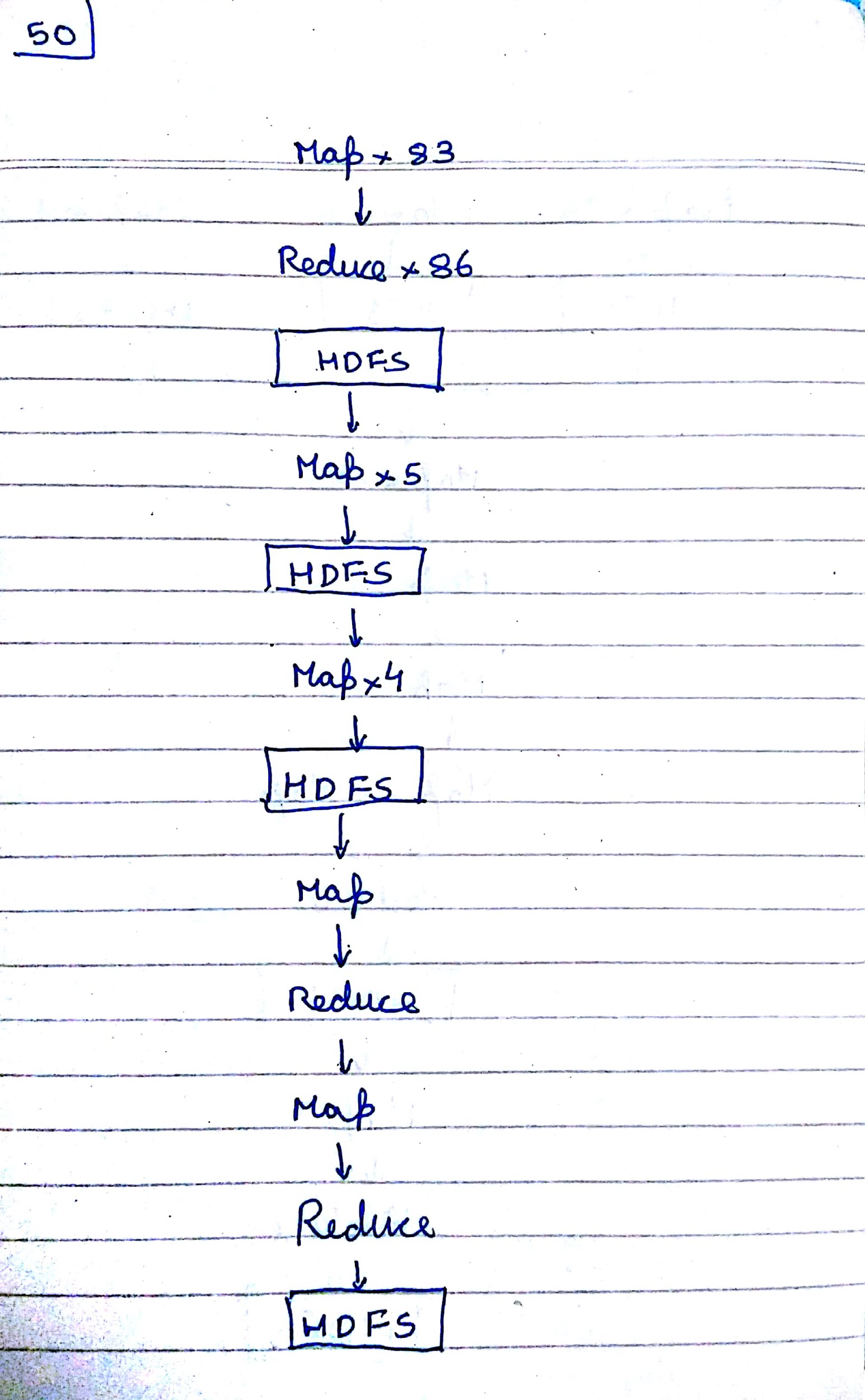
Query 12:



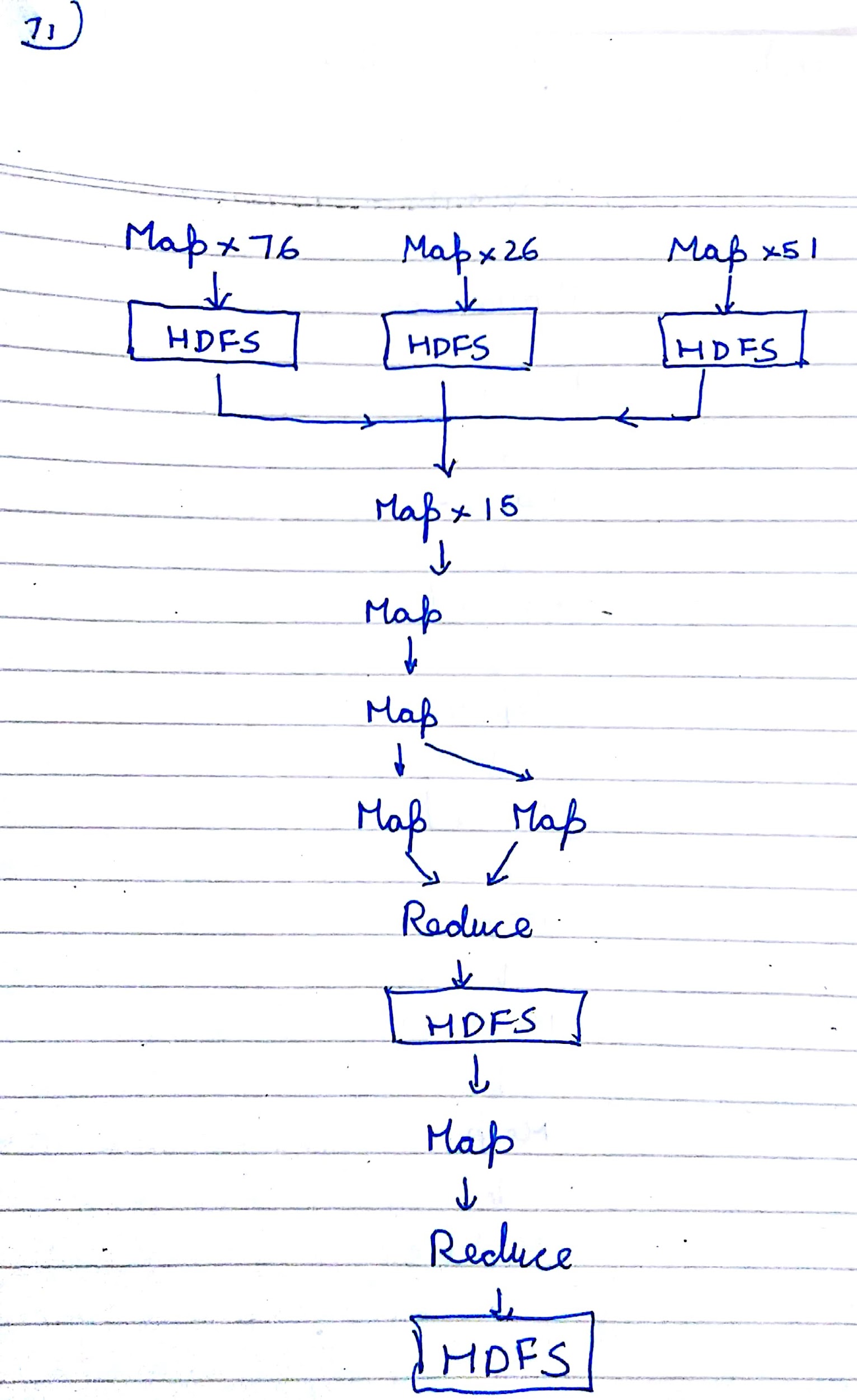
Query 21:



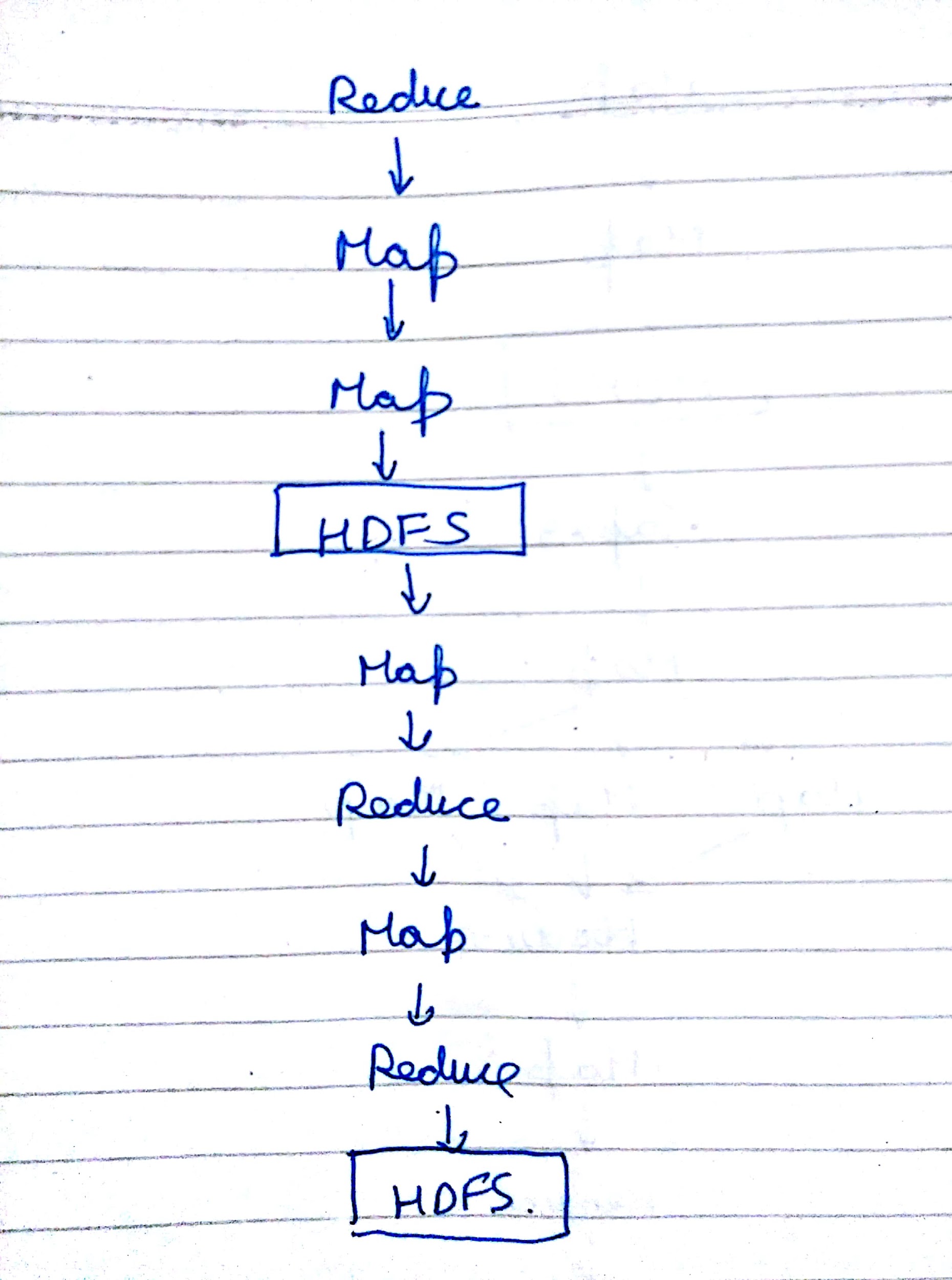
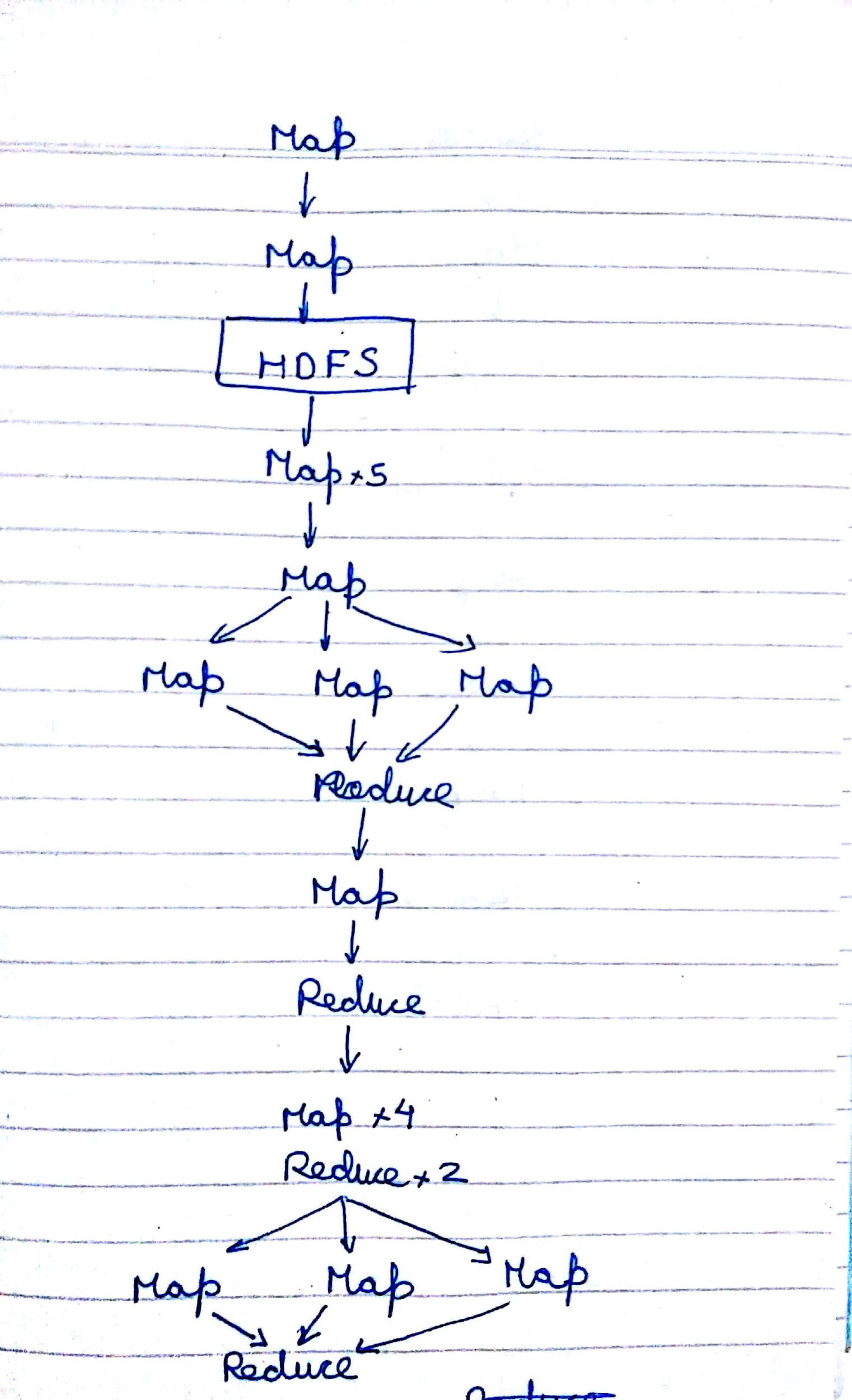
Query 50:



Query 71:



Query 85:



Part 2:

Percent at which data node is killed vs Execution time

|  |  |  |
| --- | --- | --- |
| Framework |  | Execution time |
| Tez | Normal | 220 |
|  | Killed at 25% | 216 |
|  | Killed at 75% | 225 |
|  |  |  |
| MR | Normal | 317 |
|  | Killed at 25% | 320 |
|  | Killed at 75% | 350 |

We observe a small but proportionally insignificant increase in execution times when 1 datanode is killed at various times.

This is due to two reasons:

1. Replication factor of 2 implies there are multiple copies of the data from the failing data node. Thus the framework does not lose any necessary inputs
2. Speculative execution: Hadoop schedules redundant copies of the same task. So if the tasks fail at one node, there are redundant executions of the same whose output can be used.