Flight Data Analysis

Milestone:2

1. Find the 3 airlines with the highest and lowest probability, respectively, for being on schedule

(1) MapperOne

Since each airline and flight number combination is unique, we can combine them as keys here. MapperOne calculates the sum of ArrDelay and DepDelay for each flight (each row in the table).
If this time is longer than the starting time (15 minutes), it can be considered that the flight did not take off as planned. This is followed by total content (total 1) and total content (count 1). In addition, all airline flights will be postponed and all flights will increase by one. Otherwise, when the sum of ArrDelay and DepDelay is not more than the threshold value, it means that the flight is on time, in this case we need to add 1 to the flight time from the written points (count 1). This will not change the aircraft's flight delay; it will only increase the total flight time by 1.

(2) ReducerOne

To calculate the on_schedule result, you need all flights of each airline and all flights of that airline that are not on_schedule. Then, each carrier's total flights are included, as well as the number of non-scheduled flights. So here, divide the two numbers, the number of unscheduled flights and the total number of flights, to find the delay of each plane. In this reducer the delay will be stored in a link. Then, by analyzing these ongoing links, we can buy the plane with the best performance at that moment by analyzing the links below, we voluntarily buy the plane in the least amount of time. Keep two lists of size 3 to determine the highest percentage of on-time flights and the lowest percentage of on-time flights.

2. Find the 3 airports with the longest and shortest average taxi time per flight (both in and out), respectively

(1) Mapper Two

Mapper Two use airport code as key and corresponding taxi time as value. It produces the Origin airport and departure taxi time (Origin, taxiTime) and taxi out time (Dest, taxiTime).

(2) Reducer Two

ReducerTwo calculates the total taxi time (arrival time and departure time) for each airport by summing the TaxIn and TaxiOut values. ReducerTwo also calculates all flights for each airport. The total taxi time is divided by the number of flights and the average taxi time can be calculated from these two values. This value will then be saved in a file attachment. Then, by checking the decision to go up through this list, we can reach the airport as soon as possible by taxi ride. We can reach the airport as soon as possible by checking the name link. Keep two lists of size 3 to keep the airport with the highest average taxi season and the airport taxi season with the lowest.

3. Find the most common reason for flight cancellations

(1) MapperThree

MappercThree count this cancellation reason by context write the cancellation code and 1 when a flight is cancelled.

(2) ReducerThree

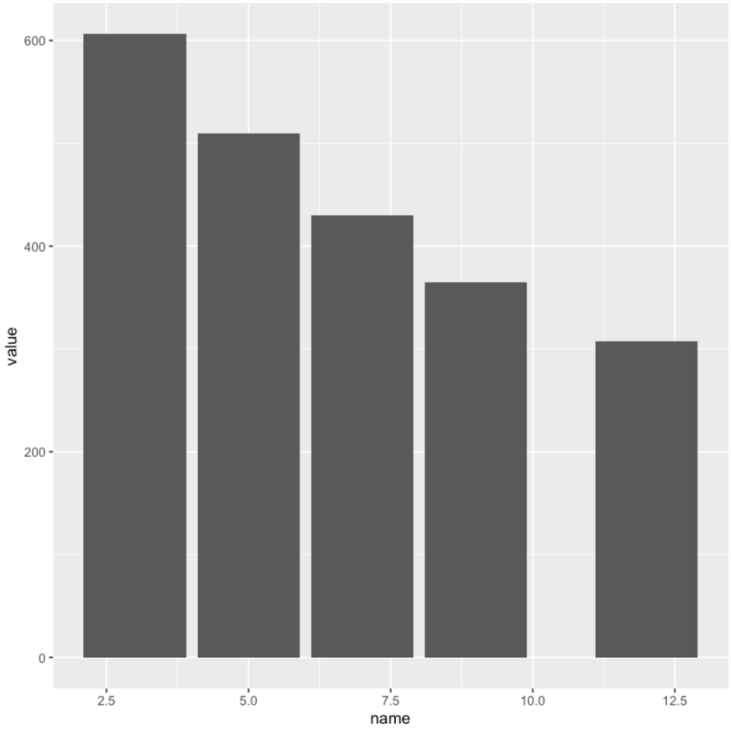
This Reducer is used to find the top cause of flight cancellation. All we need to do is manage the cancellation code with the most flight cancellations. The number of flights canceled for the same reason (same cancellation code) is counted here. When the number increases, it should be compared with the number of flights most frequently cancelled. If the calculated value is greater than the current maximum value, the cancel code is overridden and the calculated value is used as the new maximum value. This way you can find the most common reasons for flight cancellations.

4. Performance measurement

A performance measurement plot that compares the workflow execution time in response to an increasing number of VMs used for processing the entire data set (22 years)

Increase the cluster size from 3 to 12 nodes

Cluster Size	**3**	**5**	**7**	**9**	**12**
Execution Time	10 min 6 s	8 min 30 s	7 min 10 s	6 min 5 s	5 min 7 s



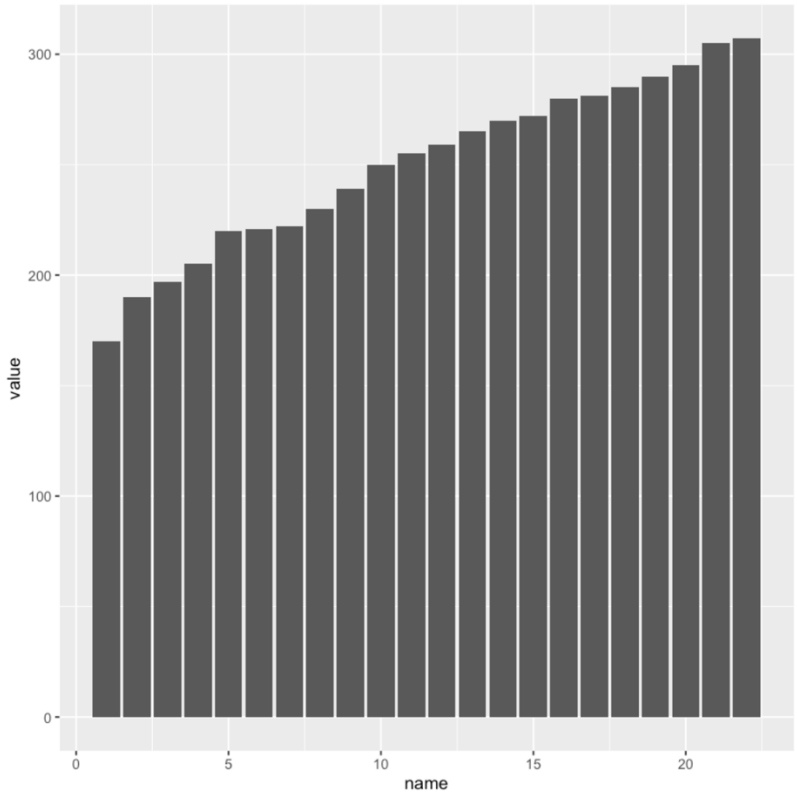
- As the number of VMs increases from 3 to 12, there is a clear downward trend in the execution time.

- The most significant drop-in time occurs when moving from 3 VMs (10 min 6 s) to 5 VMs (8 min 30 s), suggesting that increasing the cluster size significantly boosts processing efficiency initially.
- The diminishing returns as VMs increase suggest a point of saturation were adding more VMs yields less benefit in terms of reducing execution time.
- A performance measurement plot that compares the workflow execution time in response to an increasing data size (from 1 year to 22 years)
- Changing dataset from 1 year to 22 years with 12 nodes

A performance measurement plot that compares the workflow execution time in response to an increasing data size (from 1 year to 22 years)

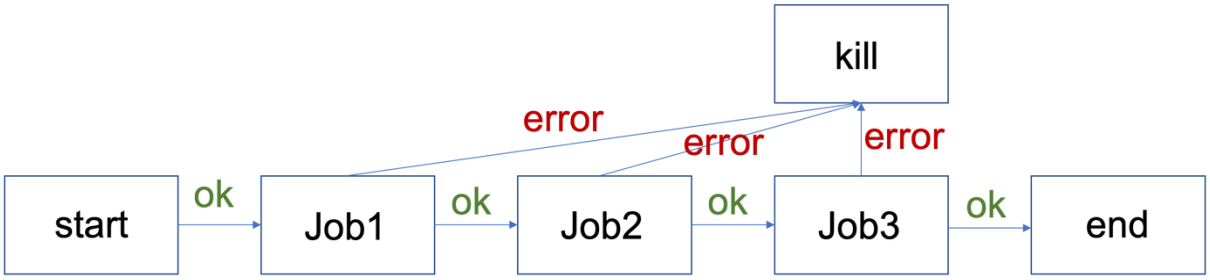
Changing dataset from 1 year to 22 years with 12 nodes

Years	Time
1	2 min 50 s
2	3 min 10 s
3	3 min 17 s
4	3 min 25 s
5	3 min 40 s
6	3 min 41 s
7	3 min 42 s
8	3 min 50 s
9	3 min 59 s
10	4 min 10 s
11	4 min 15 s
12	4 min 19 s
13	4 min 25 s
14	4 min 30 s
15	4 min 32 s
16	4 min 40 s
17	4 min 41 s
18	4 min 45 s
19	4 min 50 s
20	4 min 55 s
21	5 min 5 s
22	5 min 7 s



- Execution time increases almost linearly as the data size increases from 1 year to 22 years.
- The rate of increase in time is not uniform; slight variations occur, but overall, the trend shows a steady increment.
- The linear increase in execution time with data size indicates that the system scales proportionally, which is a desirable property in data processing systems.
- The slight fluctuations in the rate of increase might be influenced by factors such as data complexity or variations in data content for different years.
- Overall, the system appears to handle scaling in data size effectively, maintaining a manageable increase in execution time, which suggests good performance stability.

5. Workflow structure graph:


kill
error
error
error
start
ok
Job1
ok
Job2
ok
Job3
ok
end