1. Explain the di1fference between FIFO and Capacity scheduler

|  |  |  |
| --- | --- | --- |
| No | FIFO scheduler | Capacity scheduler |
| 1 | Here the applications were placed in a queue and were made to run in a order of submission (first in, first out). | In Capacity Scheduler the designs were allowed to share a huge cluster as well it provides each organization it holds with a a minimum guarantee for capacity it holds.  The major reason is the availability of resources in Hadoop Map-Reduce cluster were divided between multiple organizations were the funds were collectively clustered based on their computing needs.  There is an additional benefit in organization is that it can access any excess capacity. |
| 2 | When the applications size is large then it will make use of all the resources present in a cluster, so that each application in the queue must wait until its turn comes. | There will not be any waiting or Less Waiting when compared to FIFO.  In case of shared cluster, it is better to use Capacity Scheduler. |
| 3 | FIFO Scheduler were simple to understand and do not need any configuration. | In case of Capacity scheduler it needs some configuration. |
| 4 | Not suitable for Shared cluster. | Highly suitable for Shared cluster. |

1. Explain the difference between FIFO and Fair scheduler

|  |  |  |
| --- | --- | --- |
|  | FIFO | Fair scheduler |
| 1 | Here the applications were placed in a queue and were made to run in a order of submission (first in, first out). | It is a method were the resources can be assigned to jobs so that all jobs might get average or equal share in resources over a course time.  When a single job is made to run, then that job will make use of the entire cluster.  If in case ,When other jobs are submitted, then the tasks slots will get free up and were assigned with new jobs so each job will get same amount of CPU time. |
| 2 | When the applications size is large then it will make use of all the resources present in a cluster, so that each application in the queue must wait until its turn comes. | No waiting compared to FIFO.  On a shared cluster, it is better to use the Capacity Scheduler. |
| 3 | FIFO Scheduler is easy to understand and they do not need any other configuration. | Whereas Capacity scheduler needs some configuration |
| 4 | FIFO Scheduler is not suitable for shared cluster. | Highly suitable for Shared cluster |

**3. Explain the difference between Capacity and Fair scheduler**

|  |  |  |
| --- | --- | --- |
|  | CAPACITY SCHEDULER | FAIR SCHEDULER |
| 1 | In Capacity Scheduler the designs were allowed to share a huge cluster as well it provides each organization it holds with a a minimum guarantee for capacity it holds.  The major reason is the availability of resources in Hadoop Map-Reduce cluster were divided between multiple organizations were the funds were collectively clustered based on their computing needs.  There is an additional benefit in organization is that it can access any excess capacity. | It is a method were the resources can be assigned to jobs so that all jobs might get average or equal share in resources over a course time.  When a single job is made to run, then that job will make use of the entire cluster.  If in case ,When other jobs are submitted, then the tasks slots will get free up and were assigned with new jobs so each job will get same amount of CPU time. |
| 2 | There will be a queue which allows small job to start as soon as it gets submitted. | Separate queue were not needed but resource sharing by tasks in queue takes place. |
| 3 | When the queues were not designed or properly used then there is a chance for queues to get overloaded while some may be underutilized. | But here the overall effect will be high cluster utilization and timely small job completion |
|  |  |  |

**4.What are the limitations of hadoop 1.x and how they were overcome in hadoop 2.x.**

**Limitation Of 1.x:**

1.NameNode- Here the Single Point Of Failure is overcomed by High Availability

2. But Cluster limit is only upto 4000 which could be Overcomed by Federation

**High Availability**

In Hadoop 1.x the disadvantage is mainly due to NameNode because of Single point of Failure (when there is a failure in name node then there will be data loss instead of having a secondary node for updation from name node to secondary name node can done on hourly basis)

Solution of high availability of node :

In Hadoop 2.x there are two name node namely (which does not include any secondary name) and they are:

*1.Active Name Node*

*2.Standby Name Node*

Block reports form data node will send report to both Active NameNode and Standby NameNode.

When there is a failure in active NameNode then Stand By Name node will become active NameNode.

To find the failure of active NameNode we have node called as Journal Node.

Which works on the Basis of polling and it comprises of 3 journal nodes formula in order to calculate the quorom :

(no of journal node + 2)/2(suppose 3 journal node (3+2)/2 =2)) but the expected QUOROM is >2.

When the active name node gets failed then the journal count quorom will be >2 in parallel journal node will automatically gets changed to Stand by name node

**2. HADOOP FEDERATION**

Hadoop 1.x version have only one name node whose memory is 64 gB with which a cluster of 4000 data nodes can be maintained.

With increase in Data generation we are now running towards a scenario where say 10,000 data nodes may be required.

In such case the name node will be in requirement of more amount of memory and the single name node is scale dup with a memory above 64 gb but the cost will be high.

Hadoop 2.x there is a feature called federation in which instead of single name node (i.e active name node, passive name node and secondary name node ) there will be multiple name node (say 3) .

So each name node should maintains metadata for each department separately (so one name node will take care of marketing data alone where as another name node will take care of finance data)

These name nodes can have access over all the data nodes but they will resemble to related information (like marketing name node takes metadata about only marketing and finance name node takes metadata about only finance)