

SuiteCloud Processors

SuiteCloud Processors is the current system used to process scheduled scripts and map/reduce scripts. Before SuiteCloud Processors was introduced, scheduled scripts and map/reduce scripts were exclusively processed by scheduling queues. All scheduled script and map/reduce script jobs submitted to the same queue were processed on a FIFO (first in, first out) basis, based on the queue submission time stamp. This system had several limitations.

The scheduling queues did not provide automated load balancing or a way to prioritize specific jobs. Users with access to multiple queues (accounts with SuiteCloud Plus) were forced to manually determine the optimal configuration of jobs to queues. For the jobs that needed to be processed in a certain order, this method was useful. But it created unintended dependencies among many of the jobs submitted. If there was a delay in processing one job, a bottleneck would form. The result would be several jobs waiting in one queue when other queues were under utilized or not utilized at all.

SuiteCloud Processors resolves many of the limitations with scheduling queues. A scheduler now automatically determines the order in which jobs start to process. The scheduler uses algorithms that are based on user-defined priority levels, submission time, and user-defined preferences. The result is increased throughput, reduced wait times, and the elimination of most bottlenecks. In addition, SuiteCloud Processors requires less user intervention and enables map/reduce scripts and scheduled scripts to start sooner.

Note: Some features of SuiteCloud Processors are available only to accounts that have one or more SuiteCloud Plus licenses. For more information about SuiteCloud Plus, see the help topic [SuiteCloud Plus Settings](#).

For additional information about SuiteCloud Processors, see:

- [SuiteCloud Processors – Terminology](#)
- [SuiteCloud Processors – Basic Architecture](#)
- [SuiteCloud Processors – Supported Task Types](#)
- [SuiteCloud Processors – Processor Allotment Per Account](#)
- [SuiteCloud Processors – Priority Levels](#)
- [SuiteCloud Processors – Priority Elevation and Processor Reservation \(Advanced Settings\)](#)
- [Monitoring SuiteCloud Processors Performance](#)

SuiteCloud Processors – Terminology

Term	Definition
Job	A job is a piece of work submitted to SuiteCloud Processors for processing. Each job is executed by a single processor.
Priority	A priority is a property of a job. The priority of submitted jobs determines the order in which the scheduler sends the jobs to the processor pool. Priorities are set on the deployment record or from the SuiteCloud Processors – Priority Settings Page .
Processor	A processor is a virtual unit of processing power that executes a job. It is not distinguished as an individual physical entity, but as a single processing thread.
Processor Pool	The processor pool represents the number of processors available to a specific account. For accounts without SuiteCloud Plus, the processor pool contains one processor. For accounts with

Term	Definition
	SuiteCloud Plus, the processor pool contains multiple processors. For additional information, see the help topic SuiteCloud Plus Settings .
Queue	With SuiteCloud Processors, a queue is no longer a separate processing mechanism. On scheduled script deployments, the Queue field remains to accommodate deployments that rely on the FIFO (first in, first out) order imposed by an individual queue. However, all jobs that use queues are actually processed by the same processor pool that handles the jobs that do not use queues. All jobs compete with each other using the same common processing algorithm.
Scheduler	The scheduler determines the order in which jobs are sent to the processor pool. The scheduler uses algorithms that are based on user-defined priority levels, submission time, and user-defined preferences.
Task	A task is a script instance that is submitted for processing. Each task is handled by one or more jobs.

SuiteCloud Processors – Basic Architecture

SuiteCloud Processors currently supports the processing of scheduled scripts and map/reduce scripts. Each submitted scheduled script instance (task) is handled by one job. Each submitted map/reduce script task is handled by multiple jobs: one each for the getInput, shuffle, and summarize stages; and a minimum of one each for the map and reduce stages.

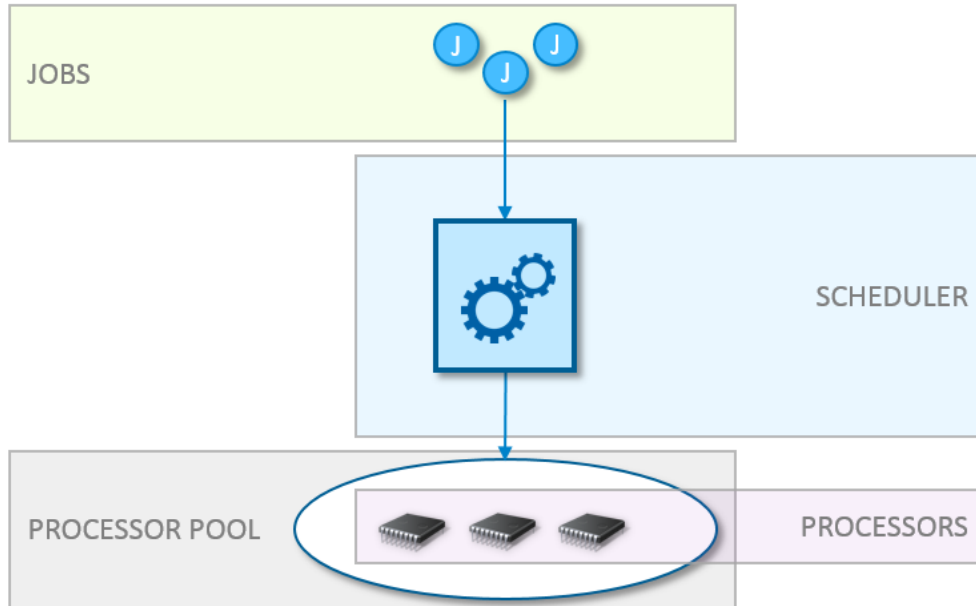
Scheduled script tasks and map/reduce script tasks are submitted for processing in one of three ways:

- By setting a one-time or recurring submission schedule from the script deployment record UI
- By selecting **Save and Execute** from the script deployment record UI to submit an on demand instance of the script
- By using a SuiteScript API to submit an on demand instance of the script

i Note: For additional information on submitting a script with the deployment record, see the help topic [SuiteScript 2.0 Scheduled Script Type](#) (SuiteScript 2.0), [SuiteScript 2.0 Map/Reduce Script Type](#) (SuiteScript 2.0), or [Scheduled Scripts](#) (SuiteScript 1.0). For additional information on submitting a script with a SuiteScript API, see the help topic [task.ScheduledScriptTask](#) (SuiteScript 2.0), [task.MapReduceScriptTask](#) (SuiteScript 2.0), or [nlapiScheduleScript\(scriptId, deployId, params\)](#) (SuiteScript 1.0).

The scheduler sends the resulting jobs to the processor pool in a particular order. This order is determined by the [SuiteCloud Processors – Priority Levels](#) and the order of submission. Jobs with a higher priority are sent before jobs with a lower priority. Jobs with the same priority go to the processor pool in the order of submission.

SuiteCloud Processors includes advanced settings that can also impact the order in which jobs are sent to the processor pool. For additional information, see [SuiteCloud Processors – Priority Elevation and Processor Reservation \(Advanced Settings\)](#).



SuiteCloud Processors – Supported Task Types

SuiteCloud Processors currently supports processing for two task types:

- SuiteScript 2.0 Map/Reduce Script Type (SuiteScript 2.0 only)
- SuiteScript 2.0 Scheduled Script Type (SuiteScript 1.0 and SuiteScript 2.0)

SuiteCloud Processors Impact on Map/Reduce Deployments

All map/reduce script deployments include a **Priority** field. For additional information, see [SuiteCloud Processors – Priority Levels](#).

For accounts with SuiteCloud Plus, the **Queues** multi-select field is replaced with the **Concurrency Limit** field. Instead of designating a specific queue, you set the maximum number of processors available to the deployment. This value equates to the number of jobs submitted for the map and reduce stages.

For map/reduce deployments created prior to the introduction of SuiteCloud Processors, the **Concurrency Limit** value is set by default. The default value is equivalent to the number of options selected on the **Queues** field. If **Queues** was previously set to 1, 3, 7, and 9, then **Concurrency Limit** is set to 4. If **Select All** was previously enabled, then **Concurrency Limit** is set to empty (the number of jobs initially created for the map and reduce stages is equivalent to the total number of processors in the processor pool).

For additional information, see the help topic [SuiteScript 2.0 Map/Reduce Script Type](#).

SuiteCloud Processors Impact on Scheduled Script Deployments

All scheduled script deployments include a **Priority** field. For additional information, see [SuiteCloud Processors – Priority Levels](#).

For all scheduled script deployments created **after** the introduction of SuiteCloud Processors, the **Queue** field is removed. These deployments cannot use queues.

For all scheduled script deployments created **prior** to the introduction of SuiteCloud Processors, the **Queue** field remains by default. This applies to accounts with and without SuiteCloud Plus. You have control over whether to stop using queues. The deployment record includes a **Remove Queue** option. After you select this option, the deployment no longer uses a queue **and cannot revert back to using a queue**.

The **Queue** field remains to accommodate deployments that rely on the FIFO order of processing imposed by an individual queue. However, all jobs that use queues are actually processed by the same processor pool that handles the jobs that do not use queues. All jobs compete with each other using the same common processing algorithm.

Note: For deployments that continue to use queues, all jobs assigned to the same queue should have the same priority. In most cases, you can keep the default (standard) priority of these jobs. However, in some cases, you may want to change these jobs to a higher or lower priority. One scenario is if you want to ensure that a specific queue always has a processor available. In that case, designate the jobs assigned to that queue as high priority. Alternatively, if you have a group of lower priority jobs, you can designate them as low priority and assign them to the same queue. That will ensure that only one is processed at a time.

Important: If your existing scheduled script deployments rely on implicit dependencies imposed by queues, you should update and test these scripts before you remove queues. Your scripts may be impacted if they rely on the sequence of FIFO (first in, first out).

One possible solution is to programmatically submit scripts in a certain order. To do this, use `task.ScheduledScriptTask` (SuiteScript 2.0) or `nlapiScheduleScript(scriptId, deployId, params)` (SuiteScript 1.0) within the first script to submit the second script. This will ensure that the jobs are submitted to the processor pool in the correct order.

SuiteCloud Processors – Processor Allotment Per Account

Important: SuiteCloud Processors is used to execute (process) all scheduled script and map/reduce script instances. However, this topic does not apply to scheduled script deployments that continue to use queues. See the help topic [SuiteCloud Plus Settings](#) to determine the number of queues available to deployments that continue to use queues.

- Processor Allotment for Map/Reduce and Scheduled Scripts
- Accounts Without a SuiteCloud Plus License

Processor Allotment for Map/Reduce and Scheduled Scripts

The following table illustrates SuiteCloud Plus support:

SuiteCloud Plus Licenses	Available Map/Reduce and Scheduled Script Processors
0	2 (see Accounts Without a SuiteCloud Plus License)
1	5

SuiteCloud Plus Licenses	Available Map/Reduce and Scheduled Script Processors
2	10
3	15
4	20
5	25
6	30
7	35
8	40
9	45
10 or more	50

Accounts Without a SuiteCloud Plus License

Accounts without a SuiteCloud Plus license now have access to two processors. The extra processor doubles the processing bandwidth for scheduled scripts and map/ reduce scripts.

If any of your scripts depend on implicit dependencies imposed by having one processor, you may be required to update your existing scripts for this feature. When an account has access to only one processor, all jobs are processed one at a time. If all jobs have the same priority, the order of processing is always first in, first out. Some scripts may depend on this behavior. With the addition of an extra processor, these scripts may no longer process in the correct order.

For example, if you have a script that pre-processes data for a second script, the first script must complete execution before the second script begins. With one processor, you can submit the scripts in the appropriate order and know that the order of processing is as expected. With two processors, there is a possibility that the second script submitted starts execution before the first script completes.

To handle the above scenario, perform the following steps:

1. Audit your scheduled scripts and map/reduce scripts. Look for scripts that depend on a specific order of execution.
2. Use the following SuiteScript APIs within the first script to programmatically submit the dependent script. This action ensures that the scripts are always executed in the correct order.

Script Type	SuiteScript 1.0 API	SuiteScript 2.0 API	Notes
Scheduled Script	<code>nlapiScheduleScript(scriptId, deployId, params)</code>	<code>task.ScheduledScriptTask</code>	Place the code at the end of the script.
Map/Reduce Script	Not Applicable	<code>task.MapReduceScriptTask</code>	Place the code at the end of the summarize stage.

SuiteCloud Processors – Priority Levels

With SuiteCloud Processors, script processing factors in submission time and priority level. This means that jobs with the highest priority are sent to the processor pool first. Jobs of lower priority are sent to the processor pool after all higher priority jobs are sent. The scheduler sends jobs with the same priority to the processor pool in the order of submission. For example, if jobs 1 and 4 are high

priority, jobs 2 and 5 are standard priority, and job 3 is low priority, the scheduler sends the jobs to the processor pool in the following order:

- Job 1
- Job 4
- Job 2
- Job 5
- Job 3

By default, all jobs have a standard priority. When you change the default priority, you change when the scheduler sends the job to the processor pool. You can do this on the deployment record or on the [SuiteCloud Processors – Priority Settings Page](#). For examples that demonstrate how changing the priority can change the order of processing, see [SuiteCloud Processors – Priority Scheduling Examples](#).

The available priority options are:

- **High:** Use to mark critical jobs that require more immediate processing. The scheduler sends these jobs to the processor pool first.
- **Standard:** This is the default setting. It is considered to be a medium priority level. The scheduler sends these jobs to the processor pool if there are no high priority jobs waiting.
- **Low:** Use to mark jobs that can tolerate a longer wait time. The scheduler sends these jobs to the processor pool if there are no high or standard priority jobs waiting.

SuiteCloud Processors – Priority Settings Page

The Priority Settings page lists each scheduled script deployment and map/reduce script deployment created for your account. Each line item corresponds to one deployment record. To access the Priority Settings page, go to Customization > Scripting > Priority Settings.

The primary purpose of this page is to manage the [SuiteCloud Processors – Priority Levels](#) for multiple deployments at once. You can also use this page to remove queues for scheduled script deployments.

Buttons

Button	Description
Submit	Saves all changes made to the page
Mark All Queues for Removal	Places a check in the Remove Queue column for all scheduled script deployments that are still using queues. Queues are not shown as removed until you submit the page. If none of your scheduled scripts use queues, this button no longer displays.
Unmark All Queues for Removal	Removes all check marks currently in the Remove Queues column. If none of your scheduled scripts use queues, this button no longer displays.
Reset Priorities	Resets all values in the Priority column to Standard (the default setting)

Filters

Filter	Description
Type	Filters for all map/reduce script deployments or all scheduled script deployments. Columns not applicable to the selected script type are hidden.
Status	Filters for the values listed in the Status column

Filter	Description
API Version	Filters for script deployments by SuiteScript 1.0 or SuiteScript 2.0
Script	Filters for all deployments of a specified script record
Page Size	Determines the number of line items shown on each result page
Show Undeployed	If enabled, the results include deployments where the Deployed option is disabled

Columns

Column	Description
Internal ID	The internal ID for the script deployment record, as seen on the Script Deployments list page (for example, 345)
Edit View	Links to the edit and view mode of the deployment record
ID	The ID of the script deployment record (for example, customdeploy_testscript1)
Script	Corresponds to the Name field value on the script record associated with the deployment record
API Version	Indicates whether the script is SuiteScript 1.0 or SuiteScript 2.0
Status	Indicates how and when a script can be submitted for processing. This value is set with the Status field on the deployment record. Possible values are: <ul style="list-style-type: none"> ■ Testing: Indicates that you can test the script in the SuiteScript Debugger. The script deployment can be submitted for processing by the script owner only. ■ Scheduled: Indicates that you can schedule a single or recurring instance of the script on the deployment record. You cannot submit an on demand instance of the script when it has this status. ■ Not Scheduled: Indicates that you can submit an on demand instance of the script with the Save and Execute button or a SuiteScript API. You can submit an on demand instance of the script only if there is no other unfinished instance of the same script.
Remove Queue	Only applicable to scheduled script deployments. Select this box to stop using queues on an existing scheduled script deployment. You can remove queues for multiple deployments at once with this column. Queues are not shown as removed until you submit the page.
Type	Indicates whether the deployment is for a scheduled script or map/reduce script
Queue	Only applicable to scheduled script deployments. Shows a value if the deployment is still using queues. After you remove queues, this value is empty.
Concurrency Limit	Only applicable to map/reduce script deployments on accounts that use SuiteCloud Plus. The maximum number of processors available to a map/reduce script deployment. You set this value on the map/reduce deployment record.
Priority	The priority setting for the deployment. For additional information, see SuiteCloud Processors – Priority Levels

SuiteCloud Processors – Priority Scheduling Examples

The two examples in this section, [Example 1 - Default Priority Scheduling](#) and [Example 2 – Varying Priority Scheduling](#), each show three diagrams. The two sets of diagrams compare the same 18 jobs as they are handled by:

- Pre-2017.2 Scheduling Queues
- 2017.2 SuiteCloud Processors: All queues are removed

- 2017.2 SuiteCloud Processors: Some queues are removed; jobs 1 - 3, 15, and 18 still use queues

This table shows the duration of each job submitted. As demonstrated in the examples, the processing environment has no impact on the amount of time required to complete each job after processing starts.

Note: Each map/reduce stage is handled by a minimum of one job. Therefore, the duration listed for map/reduce jobs is per stage. For example, the reduce stage jobs 11 and 12 have a combined duration of 5. This means that the sum of the duration of jobs 11 and 12 is 5. When one of the jobs is canceled or not created, the combined duration becomes the duration of the remaining job from the pair.

Job	Duration
1	5
2	2
3	2
4	6
5 + 6	2
7 + 8	1
9 + 10	5
11 + 12	5
13 + 14	2
15	3
16	3
17	3
18	2

Example 1 - Default Priority Scheduling

This example assumes default priority settings (standard priority) and default preferences. It also assumes that each job executes in full without yielding.

The map/reduce deployment in the first diagram has a **Queues** value of 2 and 3. The map/reduce deployment in the second and third diagrams has a **Concurrency Limit** value of 2.

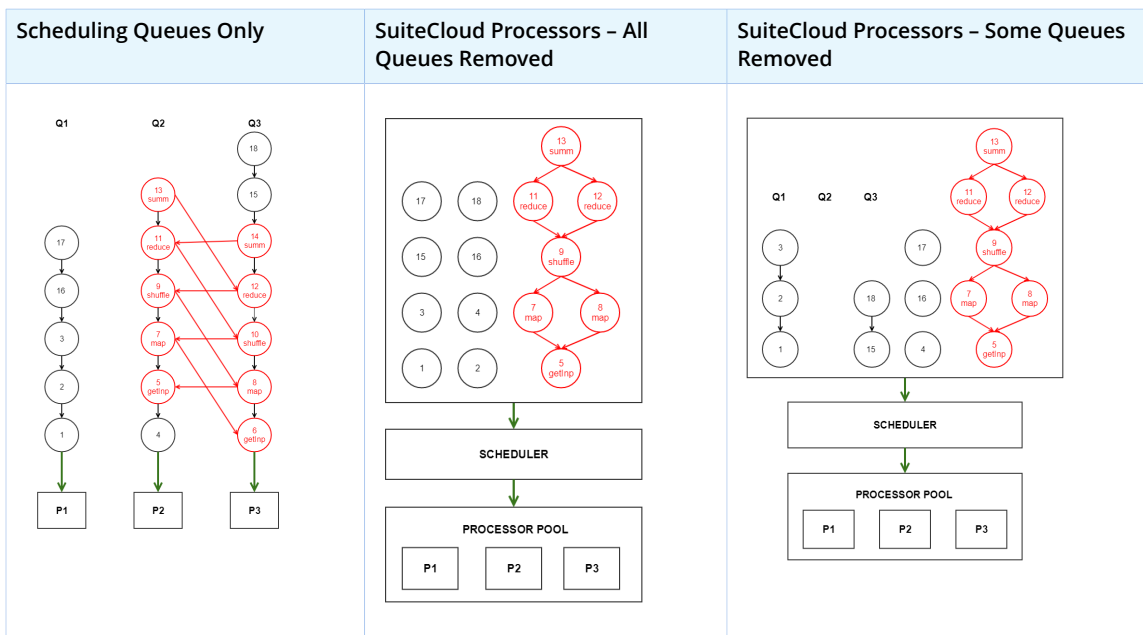
Note: With the introduction of SuiteCloud Processors, the **Queues** field is replaced with the **Concurrency Limit** field on map/reduce deployments.

Diagram Key

- The black circles are scheduled script jobs. Each scheduled script is processed by a single job.
- The red circles are map/reduce jobs. In this example, all of the map/reduce jobs belong to a single map/reduce task. The map/reduce jobs include a label that indicates the map/reduce stage being processed.
- The jobs in columns labeled Q1, Q2, and Q3 are for deployments that still use queues. The jobs in columns without a label are for deployments that no longer use queues.
- P1, P2, and P3 represent the processors that are available in the processor pool.

Note: Although processors are labeled for the purpose of this example, they are not technically individual entities. Also, a real account would never have only three processors available to it. The minimum number of processors available to an account with SuiteCloud Plus is listed at [SuiteCloud Plus Settings](#).

- The numbers within each circle represent the time stamp on the job submission, with 1 being the first job submitted. The example also uses this number as a unique identifier for each job.
- Arrows represent dependencies. For example, in the Scheduling Queues diagram, job 2 cannot start until job 1 is complete.
 - Black arrows represent queue dependencies. Jobs within a queue must be processed in the order they were submitted.
 - Red arrows represent dependencies imposed by the map/reduce algorithm.
 - Green arrows indicate access to processors.



Time Slot	Scheduling Queues Only	SuiteCloud Processors – All Queues Removed	SuiteCloud Processors – Some Queues Removed
101	Jobs 1, 4, and 6 start.	Jobs 1, 2, and 3 start.	Jobs 1, 4, and 5 start.
102	Job 6 cancels job 5. The getInput stage cannot be processed by multiple jobs, so the first getInput job to start (job 6) automatically cancels all others in the queues.		
103	Job 6 completes. The next job in Q3, job 8, starts.	Jobs 2 and 3 complete. Jobs 4 and 5 start.	Job 5 completes. There is no job 6 with SuiteCloud Processors. In the Scheduling Queues example, jobs 5 and 6 are both getInputstage jobs. The getInput stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple getInput


Time Slot	Scheduling Queues Only	SuiteCloud Processors – All Queues Removed	SuiteCloud Processors – Some Queues Removed
			stage jobs are no longer needed. Job 7 starts.
104	Job 8 completes. The map stage can be processed by multiple jobs. But since the other map stage job, job 7, is unable to start (the job utilizing its processor, job 4, is not yet complete), job 8 completes the entire map stage. Job 7 is no longer needed, so it is canceled. The next job in Q3, job 10, starts.		Job 7 completes. Since job 7 completes the entire map stage, job 8 is no longer needed. Job 8 is canceled. Job 9 starts.
105	Job 10 cancels job 9. The shuffle stage cannot be processed by multiple jobs, so the first shuffle job to start (job 10) automatically cancels all others in the queues.	Job 5 completes. There is no job 6 with SuiteCloud Processors. In the Scheduling Queues example, jobs 5 and 6 are both getInputstage jobs. The getInput stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple getInput stage jobs are no longer needed. Job 7 starts.	
106	Job 1 completes. The next job in Q1, job 2, starts.	Job 7 completes. Since job 7 completes the entire map stage, job 8 is no longer needed. Job 8 is canceled. Job 1 completes. Jobs 9 and 15 start. Jobs 11 - 13 are dependent on job 9. There is no job 10 or job 14 with SuiteCloud Processors. In the Scheduling Queues example, jobs 9 and 10 are both shuffle stage jobs and jobs 13 and 14 are both summarize stage jobs. The shuffle and summarize stages cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple shuffle and summarize stage jobs are no longer needed.	Job 1 completes. The next job in Q1, job 2, starts.
107	Job 4 completes. Q2 is blocked. The next job in Q2, job 11, cannot start. It is dependent on job 10, and job 10 is not complete.		Job 4 completes. The first job in Q3, Job 15, starts.
108	Job 2 completes. The next job in Q1, job 3, starts.		Job 2 completes. The next job in Q1, job 3, starts.
109	Job 10 completes. Job 11 can now start and Q2 is no longer blocked. The next job in Q3, job 12, starts. These are both reduce stage jobs. The reduce stage can be processed by multiple jobs.	Jobs 4 and 15 complete. Jobs 16 and 17 start.	Job 9 completes. There is no job 10 with SuiteCloud Processors. In the Scheduling Queues example, jobs 9 and 10 are both shuffle stage jobs. The shuffle stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using

Time Slot	Scheduling Queues Only	SuiteCloud Processors – All Queues Removed	SuiteCloud Processors – Some Queues Removed
			queues, multiple shuffle stage jobs are no longer needed. Job 11 starts.
110	Job 3 completes. The next job in Q1, job 16, starts.		Jobs 3 and 15 complete. There are no additional jobs to process in Q1. Jobs 12 and 16 start.
111	Job 11 completes. Q2 is blocked. The next job in Q2, job 13, cannot start. It is dependent on job 12, and job 12 is not complete.	Job 9 completes. Job 11 starts.	
112	Job 12 completes. Job 13 can now start and Q2 is no longer blocked. The next job in Q3, job 14, can also start. But these are both summarize stage jobs, and the summarize stage cannot be processed by multiple jobs. Job 13 starts first, so job 14 is canceled. Since job 14 is canceled, the next job in Q3, job 15, starts	Jobs 16 and 17 complete. Jobs 12 and 18 start.	Jobs 11 and 12 complete. Jobs 13 and 17 start.
113	Job 16 completes. The next job in Q1, job 17, starts.		Job 16 completes. The next job in Q3, Job 18, starts.
114	Job 13 completes. There are no additional jobs to process in Q2.	Jobs 11, 12, and 18 complete. Job 13 starts	Job 13 completes. There is no job 14 with SuiteCloud Processors. In the Scheduling Queues example, jobs 13 and 14 are both summarize stage jobs. The summarize stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple summarize stage jobs are no longer needed.
115	Job 15 completes. The next job in Q3, job 18, starts.		Jobs 17 and 18 complete. There are no additional jobs to process.
116	Job 17 completes. There are no additional jobs to process in Q1.	Job 13 completes. There are no additional jobs to process.	
117	Job 18 completes. There are no additional jobs to process in Q3.		

Example 2 – Varying Priority Scheduling

This example assumes default preferences. It also assumes that each job executes in full without yielding.

The map/reduce deployment in the first diagram has a **Queues** value of 2 and 3. The map/reduce deployment in the second and third diagrams has a **Concurrency Limit** value of 2.


 **Note:** With the introduction of SuiteCloud Processors, the **Queues** field is replaced with the **Concurrency Limit** field on map/reduce deployments.

The second and third diagrams vary from [Example 1 - Default Priority Scheduling](#) as follows:

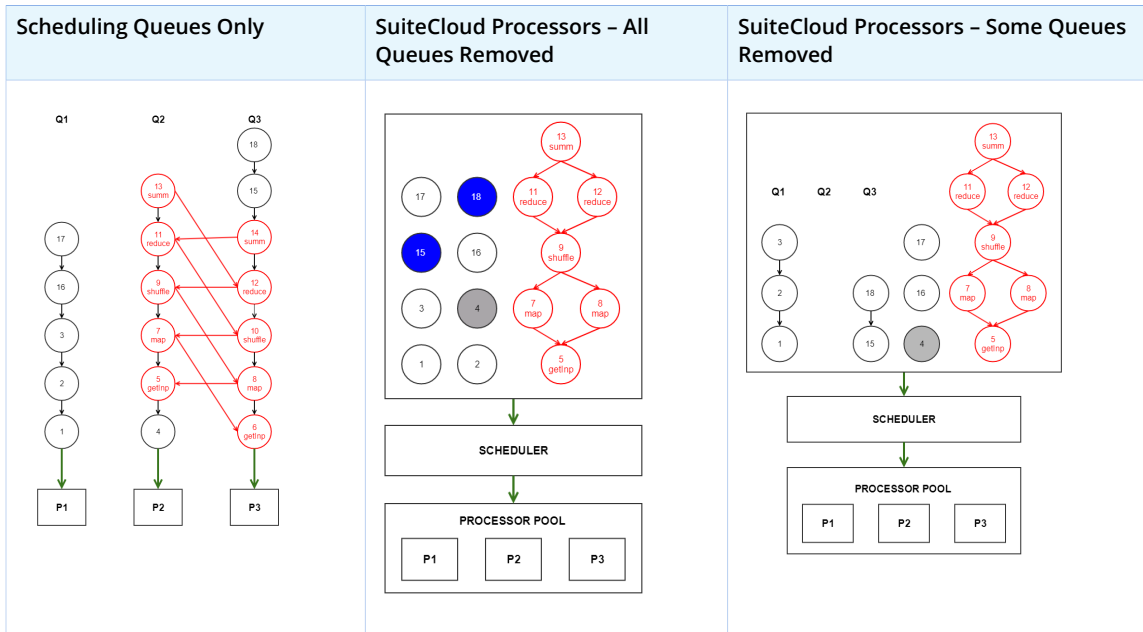
- In the second diagram, jobs 15 and 18 are high priority. Job 4 is low priority. The remaining jobs are standard priority.
- In the third diagram, job 4 is low priority. Jobs 15 and 18 in the third diagram are standard priority.

Diagram Key

- The black circles are scheduled script jobs. Each scheduled script is processed by a single job.
- The red circles are map/reduce jobs. In this example, all of the map/reduce jobs belong to a single map/reduce task. The map/reduce jobs include a label that indicates the map/reduce stage being processed.
- The circles with blue fill are high priority. The circles with gray fill are low priority. The circles with white fill are standard priority.
- The jobs in columns labeled Q1, Q2, and Q3 are for deployments that still use queues. The jobs in columns without a label are for deployments that no longer use queues.
- P1, P2, and P3 represent the processors that are available in the processor pool.

 **Note:** The processors are labeled for the purpose of this example. Although technically, processors are not distinguished as individual entities. Also, a real account would never have only three processors available to it. The minimum number of processors available to an account with SuiteCloud Plus is listed at [SuiteCloud Plus Settings](#).

- The numbers within each circle represent the time stamp on the job submission, with 1 being the first job submitted. The example also uses this number as a unique identifier for each job.
- Arrows represent dependencies. For example, in the Scheduling Queues diagram, job 2 cannot start until job 1 is complete.
 - Black arrows represent queue dependencies. Jobs within a queue must be processed in the order they were submitted.
 - Red arrows represent dependencies imposed by the map/reduce algorithm.
 - Green arrows indicate access to processors.



Time Slot	Scheduling Queues Only	SuiteCloud Processors – All Queues Removed	SuiteCloud Processors – Some Queues Removed
101	Jobs 1, 4, and 6 start.	Jobs 15, 18, and 1 start. Jobs 15 and 18 are designated as high priority so they are assigned to processors first (in the order of submission). Job 1 is the first standard priority job submitted so it is the next job assigned to a processor.	Jobs 1, 5, and 15 start. Job 4 is designated low priority, so it is processed after all other available standard priority jobs. All other standard priority jobs submitted before job 15 have dependencies that are blocking them.
102	Job 6 cancels job 5. The getInput stage cannot be processed by multiple jobs, so the first getInput job to start (job 6) automatically cancels all others in the queues.		
103	Job 6 completes. The next job in Q3, job 8, starts.	Job 18 completes. Job 2 starts.	Job 5 completes. There is no job 6 with SuiteCloud Processors. In the Scheduling Queues example, jobs 5 and 6 are both getInput stage jobs. The getInput stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple getInput stage jobs are no longer needed. Job 7 starts.
104	Job 8 completes. The map stage can be processed by multiple jobs. But since the other map stage job, job 7, is unable to start (the job utilizing its processor, job 4, is not yet complete), job 8 completes the entire map stage. Job 7 is no longer needed, so it is canceled. The next job in Q3, job 10, starts.	Job 15 completes. Job 3 starts.	Jobs 7 and 15 complete. Since job 7 completes the entire map stage, job 8 is no longer needed. Job 8 is canceled. Jobs 9 and 16 start.

Time Slot	Scheduling Queues Only	SuiteCloud Processors – All Queues Removed	SuiteCloud Processors – Some Queues Removed
105	Job 10 cancels job 9. The shuffle stage cannot be processed by multiple jobs, so the first shuffle job to start (job 10) automatically cancels all others in the queues.	Job 2 completes. Job 5 starts. Job 4 is designated low priority, so it is processed after all other available standard priority jobs.	
106	Job 1 completes. The next job in Q1, job 2, starts.	Jobs 1 and 3 complete. Jobs 16 and 17 start.	Job 1 completes. The next job in Q1, job 2, starts.
107	Job 4 completes. Q2 is blocked. The next job in Q2, job 11, cannot start. It is dependent on job 10, and job 10 is not complete.	Job 5 completes. There is no job 6 with SuiteCloud Processors. In the Scheduling Queues example, jobs 5 and 6 are both getInputstage jobs. The getInput stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple getInput stage jobs are no longer needed. Job 7 starts.	Job 16 completes. Job 17 starts.
108	Job 2 completes. The next job in Q1, job 3, starts.	Job 7 completes. Since job 7 completes the entire map stage, job 8 is no longer needed. Job 8 is canceled. Job 9 starts.	Job 2 completes. The next job in Q1, job 3, starts.
109	Job 10 completes. Job 11 can now start and Q2 is no longer blocked. The next job in Q3, job 12, starts. These are both reduce stage jobs. The reduce stage can be processed by multiple jobs.	Jobs 16 and 17 complete. Job 4 starts. Job 4 is designated as low priority, but all the remaining standard priority jobs are dependent on job 9.	Job 9 completes. There is no job 10 with SuiteCloud Processors. In the Scheduling Queues example, jobs 9 and 10 are both shuffle stage jobs. The shuffle stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple shuffle stage jobs are no longer needed. Job 11 starts.
110	Job 3 completes. The next job in Q1, job 16, starts.		Jobs 3 and 17 complete. There are no additional jobs to process in Q1. Job 12 starts. The next job in Q3, job 18, starts.
111	Job 11 completes. Q2 is blocked. The next job in Q2, job 13, cannot start. It is dependent on job 12, and job 12 is not complete.		
112	Job 12 completes. Job 13 can now start and Q2 is no longer blocked. The next job in Q3, job 14, can also start. But these are both summarize stage jobs, and the summarize stage cannot be processed by multiple jobs. Job 13 starts first, so job 14 is canceled. Since job 14 is canceled, the next job in Q3, job 15, starts		Jobs 11, 12 and 18 complete. There are no additional jobs to process in Q3. Jobs 13 and 4 start.
113	Job 16 completes. The next job in Q1, job 17, starts.	Job 9 completes. There is no job 10 with SuiteCloud Processors. In	

Time Slot	Scheduling Queues Only	SuiteCloud Processors – All Queues Removed	SuiteCloud Processors – Some Queues Removed
		the Scheduling Queues example, jobs 9 and 10 are both shuffle stage jobs. The shuffle stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple shuffle jobs are no longer needed. Jobs 11 and 12 start.	
114	Job 13 completes. There are no additional jobs to process in Q2.		Job 13 completes. There is no job 14 with SuiteCloud Processors. In the Scheduling Queues example, jobs 13 and 14 are both summarize stage jobs. The summarize stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple summarize stage jobs are no longer needed.
115	Job 15 completes. The next job in Q3, job 18, starts.	Jobs 4 and 12 complete.	
116	Job 17 completes. There are no additional jobs to process in Q1.	Job 11 completes. Job 13 starts.	
117	Job 18 completes. There are no additional jobs to process in Q3.		
118		Job 13 completes. There is no job 14 with SuiteCloud Processors. In the Scheduling Queues example, jobs 13 and 14 are both summarize stage jobs. The summarize stage cannot be processed by multiple jobs. Since this map/reduce deployment is not using queues, multiple summarize stage jobs are no longer needed. There are no additional jobs to process.	Job 4 completes. There are no additional jobs to process.

SuiteCloud Processors – Priority Elevation and Processor Reservation (Advanced Settings)

Priority Elevation

Priority elevation enables you to automatically increase the priority of each low or standard priority job after a specific time interval. The time interval starts when the job is submitted. You do not need to use these settings in most cases. Because of this, priority elevation is disabled by default. However, you may need to use them if lower priority jobs experience excessive wait times.

Note: Priority elevation only impacts lower priority jobs with a wait time greater than the time interval indicated. If a lower priority job is sent to the processor pool before the time interval is over, that job is processed with its original priority.

To access the priority elevation settings, go to Setup > Preferences > SuiteCloud Processors.

The screenshot shows the 'SuiteCloud Processors Preferences' dialog box. At the top, there are three buttons: 'Save' (highlighted in blue), 'Cancel', and 'Basic'. Below these buttons is a section titled 'Priority Elevation' with a light blue header. Under this header, a descriptive text states: 'Priority Elevation prevents high priority jobs from using all processing resources.' There are four radio button options:

- NO PRIORITY ELEVATION**: Jobs are processed according to their original priority.
- MODERATE PRIORITY ELEVATION**: Job priority is raised after a moderate wait time.
- INTENSIVE PRIORITY ELEVATION**: Job priority is raised after a short wait time.
- CUSTOM PRIORITY ELEVATION**: Elevate priority of job after a 'TIME INTERVAL'. Below this option is a dropdown menu labeled 'TIME INTERVAL' with a downward arrow.


There are three predetermined settings listed:

- **No Priority Elevation:** The default setting
- **Moderate Priority Elevation:**
 - If a low priority job is still waiting after four hours, it is elevated to standard priority
 - If a standard priority job is still waiting after four hours, it is elevated to high priority

With this option, if priority elevation is applicable, the system elevates low priority jobs to high priority jobs after eight hours. Specifically, the jobs in this scenario are elevated to standard priority after four hours and then elevated to high priority after another four hours.
- **Intensive Priority Elevation:**
 - If a low priority job is still waiting after one hour, it is elevated to standard priority
 - If a standard priority job is still waiting after one hour, it is elevated to high priority


With this option, if priority elevation is applicable, the system elevates low priority jobs to high priority jobs after two hours. Specifically, the jobs in this scenario are elevated to standard priority after one hour and then elevated to high priority after another hour.
- **Custom Priority Elevation:** This option enables you to specify a custom time interval for priority elevation from the **Time Interval** dropdown list.

The **Time Interval** field specifies the time interval set for priority elevation. When **Custom Priority Elevation** is selected, this field is enabled for editing. Otherwise, the field displays a value that corresponds with the option selected, but it cannot be edited.

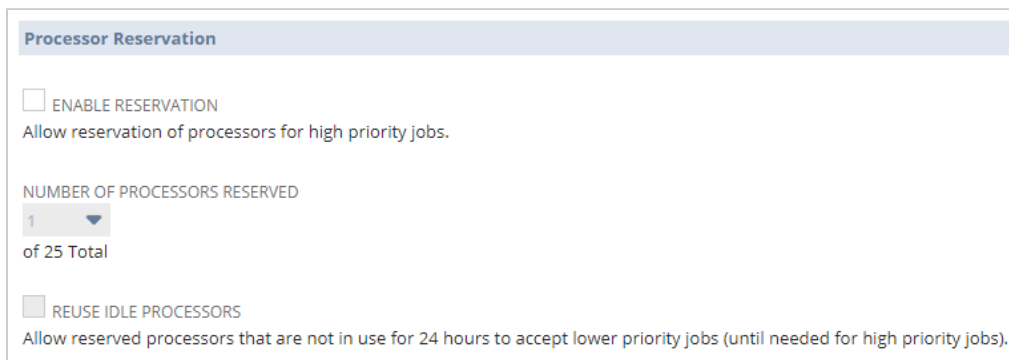
 **Note:** Click **Advanced** at the top of the page to access **Custom Priority Elevation and Time Interval**.

Processor Reservation


Processor reservation enables you to reserve a specified number of processors for high priority jobs. You do not need to use these settings in most cases. Because of this, processor reservation is disabled by default.

 **Important:** Processor reservation is available for SuiteCloud Plus accounts only.

To access the priority elevation settings, go to Setup > Preferences > SuiteCloud Processors. Click **Advanced** at the top of the page.




When you select **Enable Reservation**, you can reserve all but one of your available processors from the **Number of Processors Reserved** dropdown list. If a high priority job is submitted, it is sent to the processor pool if there is at least one processor available. If a standard or low priority job is submitted, it is sent to the processor pool only if there are more processors available than the number reserved. For example, you have 10 processors reserved out of 25 total processors. A standard or low priority job is sent to the processor pool if there are at least 11 processors available. If there are 10 processors or fewer available, the lower priority job must wait.

 **Important:** Changes to the Number of Processors Reserved apply to all jobs that have not yet started. This can have an immediate impact on map/reduce scripts since each stage is processed by a minimum of one job. If there is a high priority map/reduce script instance (task) executing and this setting is changed, the new value is applied to all jobs for this task that have not yet started. This includes jobs that may be created from yielding.

Processor reservation decreases the number of processors available for standard and low priority jobs. Therefore, it can reduce the throughput of these jobs. The **Reuse Idle Processors** setting temporarily releases reserved processors that have not been used in the past 24 hours. This increases the number of processors for lower priority jobs.

When **Reuse Idle Processors** is enabled, it initiates an hourly recurring audit. The system uses the data collected to determine whether to release reserved processors. After reserved processors are released, the audit data is also used to determine whether the system needs to increase reserved processors.

 **Important:** If the system decreases or increases the number of reserved processors, additional decreases are not made for 24 hours. However, additional increases (up to the selected limit) can still be made after each hourly audit. This process continues as long as **Reuse Idle Processors** is enabled.

The system analyzes the following data points during this process:

- **a** = total number of processors available
- **b** = the value you set for **Number of Processors Reserved** (the maximum number of reserved processors; this number does not change)
- **c** = maximum number of jobs concurrently processed in the last 24 hours
- **d** = maximum number of high priority jobs that concurrently waited more than a minute in the last hour
- **e** = current number of reserved processors (this number can change if **Reuse Idle Processors** is enabled)

If	Then
c is less than a This means that some reserved processors were not used.	The number of processors available to lower priority jobs is increased by a - c (up to the value of a).
e is less than b AND d is more than 0	The number of reserved processors is increased by d (up to the value of b) The system cannot increase the number of reserved processors over the limit set for Number of Processors Reserved . In other words, e cannot be greater than b .