Ray Autoscaler 2.0 Changes

Global `min_workers` field is removed

The top level `min_workers` field is removed, as it is ambiguous with multiple node types.

`min_workers` should now be set as a property per node type.

Pre 2.0	**2.0**

`cluster_name: default`
 	
`min_workers: 0`
	`cluster_name: default`
	
	`available_node_types:`
	`  worker_type:`
	`  node_config:`
	`  ...`
	`  min_workers: 1`

`initial_workers` field is removed

Due to downscaling of idle nodes, to guarantee a minimum set of workers exist when a job starts, the `min_workers` field should be used instead.

Pre 2.0	**2.0**

`cluster_name: default`
 	
`initial_workers: 1`
	`cluster_name: default`
	
	`available_node_types:`
	`  worker_type:`
	`  node_config:`
	`  ...`
	`  min_workers: 1`

`autoscaling_mode` field is removed

The autoscaling_mode field is removed in favor of the more granular `upscaling_speed` field which can provide the same behavior.

Pre 2.0	2.0

`cluster_name: default`
 	
`autoscaling_mode: <default | aggressive>`
	`cluster_name: default`
	`# The values 1 and 9999 correspond to default and aggressive upscaling respectively. It can also be set to other positive numbers.`
	`upscaling_speed: <1 | 9999>`

`target_utilization_fraction` field and `default_worker_type` field are removed

The target_utilization_fraction and the “utilization based autoscaling” feature has been removed in favor of “resource demand based autoscaling”. To continue to reserve idle resources in your cluster, use the `ray.autoscaler.sdk.request_resources()` function programmatically, or set `min_workers`. Since there is no `target_utilization_fraction`, there is no need for a `default_worker_type` to launch for it.

Pre 2.0	2.0

`cluster_name: default`
 	
`target_utilization_fraction: 0.9`
	`import ray`
	
	`ray.init(address="auto")`
	
	`total = int(`
	`  ray.cluster_resources(`
	`  ).get("CPU", 0))`
	`avail = ray.available_resources(`
	`  ).get("CPU", 0)`
	`used_resources = total - avail`
	`ray.autoscaler.sdk.request_resources`
	`(num_cpus = used_resources / 0.9)`

`worker_nodes` field is removed

Worker nodes must be specified using the `available_node_types field`. Note that `resources` must be specified for multiple node types. Note for some cloud providers (e.g. AWS), the resources for a node type can be inferred.

Pre 2.0	2.0

cluster_name: default
 	
worker_nodes:
  <config>
	cluster_name: default
	
	available_node_types:
	  worker_type:
	  node_config:
	  <config>
	  resources: {"CPU": 8}

`head_node` field is removed

The head node should be specified via the `available_node_types` field and `head_node_type` field. To ensure a node type is only used for the head node, set max_workers: 0

Pre 2.0	2.0

cluster_name: default
 	
head_node:
  <config>
	cluster_name: default
	head_node_type: head_type
	
	available_node_types:
	 head_type:
	  node_config:
	  <config>
	  resources: {}
	  max_workers: 0