Arquivo queues.conf

```
[general]
;
; Global settings for call queues
;
; Persistent Members
;    Store each dynamic member in each queue in the astdb so that
;    when asterisk is restarted, each member will be automatically
;    read into their recorded queues. Default is 'yes'.
;

persistentmembers = yes

;
; Keep Stats
;    Keep queue statistics during a reload. Default is 'no'
;

keepstats = no

;
; AutoFill Behavior
;    The old/current behavior of the queue has a serial type behavior
;    in that the queue will make all waiting callers wait in the queue
;    even if there is more than one available member ready to take
;    calls until the head caller is connected with the member they
;    were trying to get to. The next waiting caller in line then
;    becomes the head caller, and they are then connected with the
;    next available member and all available members and waiting callers
;    waits while this happens. The new behavior, enabled by setting
;    autofill=yes makes sure that when the waiting callers are connecting
;    with available members in a parallel fashion until there are
;    no more available members or no more waiting callers. This is
;    probably more along the lines of how a queue should work and
;    in most cases, you will want to enable this behavior. If you
;    do not specify or comment out this option, it will default to no
;    to keep backward compatibility with the old behavior.
;
autofill = yes
;
; Monitor Type
;    By setting monitor-type = MixMonitor, when specifying monitor-format
;    to enable recording of queue member conversations, app_queue will
;    now use the new MixMonitor application instead of Monitor so
;    the concept of "joining/mixing" the in/out files now goes away
;    when this is enabled. You can set the default type for all queues
;    here, and then also change monitor-type for individual queues within
;    queue by using the same configuration parameter within a queue
;    configuration block. If you do not specify or comment out this option,
;    it will default to the old 'Monitor' behavior to keep backward
;    compatibility.
;
monitor-type = MixMonitor
;
; UpdateCDR behavior.
;    This option is implemented to mimic chan_agents behavior of populating
;    CDR dstchannel field of a call with an agent name, which you can set
;    at the login time with AddQueueMember membername parameter.
;
; updatecdr = no

;
; Note that a timeout to fail out of a queue may be passed as part of
; an application call from extensions.conf:
; Queue(queuename,[options],[optionalurl],[announceoverride],[timeout])
; example: Queue(dave,t,,,45)

; shared_lastcall will make the lastcall and calls received be the same in
; members logged in more than one queue.
; This is useful to make the queue respect the wrapuptime of another queue
; for a shared member
```

```
;
shared_lastcall=no
;
;[markq]
;
; A sample call queue
;
; Musicclass sets which music applies for this particular call queue.
; The only class which can override this one is if the MOH class is set
; directly on the channel using Set(CHANNEL(musicclass)=whatever) in the
; dialplan.
;
;musicclass = default
;
; An announcement may be specified which is played for the member as
; soon as they answer a call, typically to indicate to them which queue
; this call should be answered as, so that agents or members who are
; listening to more than one queue can differentiated how they should
; engage the customer
;
;announce = queue-markq
;
; A strategy may be specified.  Valid strategies include:
;
; ringall - ring all available channels until one answers (default)
; leastrecent - ring interface which was least recently called by this queue
; fewestcalls - ring the one with fewest completed calls from this queue
; random - ring random interface
; rrmemory - round robin with memory, remember where we left off last ring pass
; linear - rings interfaces in the order specified in this configuration file.
;          If you use dynamic members, the members will be rung in the order in
;          which they were added
; wrandom - rings random interface, but uses the member's penalty as a weight
;           when calculating their metric. So a member with penalty 0 will have
;           a metric somewhere between 0 and 1000, and a member with penalty 1 will
;                       have a metric between 0 and 2000, and a member with penalty 2 will have
;           a metric between 0 and 3000. Please note, if using this strategy, the member
;           penalty is not the same as when using other queue strategies. It is ONLY used
;           as a weight for calculating metric.
;
;strategy = ringall
;
; Second settings for service level (default 0)
; Used for service level statistics (calls answered within service level time
; frame)
;servicelevel = 60
;
; A context may be specified, in which if the user types a SINGLE
; digit extension while they are in the queue, they will be taken out
; of the queue and sent to that extension in this context.
;
;context = qoutcon
;
;----------------------QUEUE TIMING OPTIONS-------------------------------------
; A Queue has two different "timeout" values associated with it. One is the
; timeout parameter configured in queues.conf. This timeout specifies the
; amount of time to try ringing a member's phone before considering the
; member to be unavailable. The other timeout value is the second argument
; to the Queue() application. This timeout represents the absolute amount
; of time to allow a caller to stay in the queue before the caller is
; removed from the queue. In certain situations, these two timeout values
; may clash. For instance, if the timeout in queues.conf is set to 5 seconds,
; the retry value in queues.conf is set to 4, and the second argument to Queue()
; is 10, then the following may occur:
;
; A caller places a call to a queue.
; The queue selects a member and attempts to ring that member.
; The member's phone is rung for 5 seconds and he does not answer.
; The retry time of 4 seconds occurs.
; The queue selects a second member to call.
;
; How long does that second member's phone ring? Does it ring for 5 seconds
; since the timeout set in app_queue is 5 seconds? Does it ring for 1 second since
```

```
; the caller has been in the queue for 9 seconds and is supposed to be removed after
; being in the queue for 10 seconds? This is configurable with the timeoutpriority
; option. By setting the timeoutpriority to "conf" then you are saying that you would
; rather use the time specified in the configuration file even if it means having the
; caller stay in the queue longer than the time specified in the application argument.
; For the scenario described above, timeoutpriority=conf would result in the second
; member's phone ringing for 5 seconds. By specifying "app" as the value for
; timeoutpriority, you are saying that the timeout specified as the argument to the
; Queue application is more important. In the scenario above, timeoutpriority=app
; would result in the second member's phone ringing for 1 second.
;
; There are a few exceptions to the priority rules. For instance, if timeoutpriority=appp
; and the configuration file timeout is set to 0, but the application argument timeout is
; non-zero, then the timeoutpriority is ignored and the application argument is used as
; the timeout. Furthermore, if no application argument timeout is specified, then the
; timeoutpriority option is ignored and the configuration file timeout is always used
; when calling queue members.
;
; In timeoutpriority=conf mode however timeout specified in config file will take higher
; priority than timeout in application arguments, so if config file has timeout 0, each
; queue member will be called indefineately and application timeout will be checked only
; after this call attempt. This is useful for having queue members with custom timeouts
; specified within Dial application of Local channel, and allows handling NO ANSWER which
; would otherwise be interrupted by queue destroying child channel on timeout.
;
; The default value for timeoutpriority is "app" since this was how previous versions of
; Asterisk behaved.
;
;timeout = 15
;retry = 5
;timeoutpriority = app|conf
;
;----------------------END QUEUE TIMING OPTIONS---------------------------------
; Weight of queue - when compared to other queues, higher weights get
; first shot at available channels when the same channel is included in
; more than one queue.
;
;weight=0
;
; After a successful call, how long to wait before sending a potentially
; free member another call (default is 0, or no delay)
;
;wrapuptime=15
;
; Autofill will follow queue strategy but push multiple calls through
; at same time until there are no more waiting callers or no more
; available members. The per-queue setting of autofill allows you
; to override the default setting on an individual queue level.
;
;autofill=yes
;
; Autopause will pause a queue member if they fail to answer a call
;
;autopause=yes
;
; Maximum number of people waiting in the queue (0 for unlimited)
;
;maxlen = 0
;
; If set to yes, just prior to the caller being bridged with a queue member
; the following variables will be set
; MEMBERINTERFACE is the interface name (eg. Agent/1234)
; MEMBERNAME is the member name (eg. Joe Soap)
; MEMBERCALLS is the number of calls that interface has taken,
; MEMBERLASTCALL is the last time the member took a call.
; MEMBERPENALTY is the penalty of the member
; MEMBERDYNAMIC indicates if a member is dynamic or not
; MEMBERREALTIME indicates if a member is realtime or not
;
;setinterfacevar=no
;
; If set to yes, just prior to the caller being bridged with a queue member
; the following variables will be set:
```

```
; QEHOLDTIME callers hold time
; QEORIGINALPOS original position of the caller in the queue
;
;setqueueentryvar=no
;
; If set to yes, the following variables will be set
; just prior to the caller being bridged with a queue member
; and just prior to the caller leaving the queue
; QUEUENAME name of the queue
; QUEUEMAX maxmimum number of calls allowed
; QUEUESTRATEGY the strategy of the queue;
; QUEUECALLS number of calls currently in the queue
; QUEUEHOLDTIME current average hold time
; QUEUECOMPLETED number of completed calls for the queue
; QUEUEABANDONED number of abandoned calls
; QUEUESRVLEVEL queue service level
; QUEUESRVLEVELPERF current service level performance
;
;setqueuevar=no
;
; if set, run this macro when connected to the queue member
; you can override this macro by setting the macro option on
; the queue application
;
; membermacro=somemacro

; How often to announce queue position and/or estimated
; holdtime to caller (0=off)
; Note that this value is ignored if the caller's queue
; position has changed (see min-announce-frequency)
;
;announce-frequency = 60
;
; The absolute minimum time between the start of each
; queue position and/or estimated holdtime announcement
; This is useful for avoiding constant announcements
; when the caller's queue position is changing frequently
; (see announce-frequency)
;
;min-announce-frequency = 15
;
; How often to make any periodic announcement (see periodic-announce)
;
;periodic-announce-frequency=60
;
; Should the periodic announcements be played in a random order? Default is no.
;
;random-periodic-announce=no
;
; Should we include estimated hold time in position announcements?
; Either yes, no, or only once.
; Hold time will be announced as the estimated time.
;
;announce-holdtime = yes|no|once
;
; Queue position announce?
; Valid values are "yes," "no," "limit," or "more." If set to "no," then the caller's position will
; never be announced. If "yes," then the caller's position in the queue will be announced
; to the caller. If set to "more," then if the number of callers is more than the number
; specified by the announce-position-limit option, then the caller will hear that there
; are more than that many callers waiting (i.e. if a caller number 6 is in a queue with the
; announce-position-limit set to 5, then that caller will hear that there are more than 5
; callers waiting). If set to "limit," then only callers within the limit specified by announce-position-limit
; will have their position announced.
;
;announce-position = yes
;
; If you have specified "limit" or "more" for the announce-position option, then the following
; value is what is used to determine what announcement to play to waiting callers. If you have
; set the announce-position option to anything else, then this will have no bearing on queue operation
;
;announce-position-limit = 5
;
```

```
; What's the rounding time for the seconds?
; If this is non-zero, then we announce the seconds as well as the minutes
; rounded to this value.
; Valid values are 0, 5, 10, 15, 20, and 30.
;
; announce-round-seconds = 10
;
; Use these sound files in making position/holdtime announcements.  The
; defaults are as listed below -- change only if you need to.
;
; Keep in mind that you may also prevent a sound from being played if you
; explicitly set a sound to be an empty string. For example, if you want to
; prevent the queue from playing queue-thankyou, you may set the sound using
; the following line:
;
; queue-thankyou=
;
                        ;       ("You are now first in line.")
;queue-youarenext = queue-youarenext
                        ;       ("There are")
;queue-thereare = queue-thereare
                        ;       ("calls waiting.")
;queue-callswaiting = queue-callswaiting
                        ;       ("The current est. holdtime is")
;queue-holdtime = queue-holdtime
                        ;       ("minutes.")
;queue-minutes = queue-minutes
                        ;       ("seconds.")
;queue-seconds = queue-seconds
                        ;       ("Thank you for your patience.")
;queue-thankyou = queue-thankyou
                        ;       ("Hold time")
;queue-reporthold = queue-reporthold
                        ;       ("All reps busy / wait for next")
;periodic-announce = queue-periodic-announce
;
; A set of periodic announcements can be defined by separating
; periodic announcements to reproduce by commas. For example:
;periodic-announce = queue-periodic-announce,your-call-is-important,please-wait
;
; The announcements will be played in the order in which they are defined. After
; playing the last announcement, the announcements begin again from the beginning.
;
; Calls may be recorded using Asterisk's monitor/MixMonitor resource
; This can be enabled from within the Queue application, starting recording
; when the call is actually picked up; thus, only successful calls are
; recorded, and you are not recording while people are listening to MOH.
; To enable monitoring, simply specify "monitor-format";  it will be disabled
; otherwise.
;
; You can specify the monitor filename with by calling
;    Set(MONITOR_FILENAME=foo)
; Otherwise it will use MONITOR_FILENAME=${UNIQUEID}
;
; Pick any one valid extension for monitor format recording. If you leave
; monitor-format commented out, it will not record calls.
;
; monitor-format = gsm|wav|wav49
;
; Monitor Type
;    By setting monitor-type = MixMonitor, when specifying monitor-format
;    to enable recording of queue member conversations, app_queue will
;    now use the new MixMonitor application instead of Monitor so
;    the concept of "joining/mixing" the in/out files now goes away
;    when this is enabled. If you do not specify or comment out this option,
;    it will default to the old 'Monitor' behavior to keep backward
;    compatibility.
;
; monitor-type = MixMonitor
;
; ----------------------- TYPE MIXMONITOR OPTIONS -----------------------------
;
;
```

```
; You can specify the options supplied to MixMonitor by calling
;   Set(MONITOR_OPTIONS=av(<x>)V(<x>)W(<x>))
; The 'b' option for MixMonitor (only save audio to the file while bridged) is
; implied.
;
; You can specify a post recording command to be executed after the end of
; recording by calling
;   Set(MONITOR_EXEC=mv /var/spool/asterisk/monitor/^{MONITOR_FILENAME} /tmp/^{MONITOR_FILENAME})
;
; The command specified within the contents of MONITOR_EXEC will be executed when
; the recording is over. Any strings matching ^{X} will be unescaped to ${X} and
; all variables will be evaluated just prior to recording being started.
;
; The contents of MONITOR_FILENAME will also be unescaped from ^{X} to ${X} and
; all variables will be evaluated just prior to recording being started.
;
;
; This setting controls whether callers can join a queue with no members. There
; are three choices:
;
; yes    - callers can join a queue with no members or only unavailable members
; no     - callers cannot join a queue with no members
; strict - callers cannot join a queue with no members or only unavailable
;          members
; loose  - same as strict, but paused queue members do not count as unavailable
;
; joinempty = yes
;
;
; If you wish to remove callers from the queue when new callers cannot join,
; set this setting to one of the same choices for 'joinempty'
;
; leavewhenempty = yes
;
;
; If this is set to yes, the following manager events will be generated:
; AgentCalled, AgentDump, AgentConnect, AgentComplete; setting this to
; vars also sends all channel variables with the event.
; (may generate some extra manager events, but probably ones you want)
;
; eventwhencalled = yes|no|vars
;
; If this is set to yes, the following manager events will be generated:
; QueueMemberStatus
; (may generate a WHOLE LOT of extra manager events)
;
; eventmemberstatus = no
;
; If you wish to report the caller's hold time to the member before they are
; connected to the caller, set this to yes.
;
; reportholdtime = no
;
; If you want the queue to avoid sending calls to members whose devices are
; known to be 'in use' (via the channel driver supporting that device state)
; uncomment this option. (Note: only the SIP channel driver currently is able
; to report 'in use'.)
;
; ringinuse = no
;
; If you wish to have a delay before the member is connected to the caller (or
; before the member hears any announcement messages), set this to the number of
; seconds to delay.
;
; memberdelay = 0
;
; If timeoutrestart is set to yes, then the timeout for an agent to answer is
; reset if a BUSY or CONGESTION is received.  This can be useful if agents
; are able to cancel a call with reject or similar.
;
; timeoutrestart = no
;
; If you wish to implement a rule defined in queuerules.conf (see
```

```
; configs/queuerules.conf.sample from the asterisk source directory for
; more information about penalty rules) by default, you may specify this
; by setting defaultrule to the rule's name
;
; defaultrule = myrule
;
; Each member of this call queue is listed on a separate line in
; the form technology/dialstring.  "member" means a normal member of a
; queue.  An optional penalty may be specified after a comma, such that
; entries with higher penalties are considered last.  An optional member
; name may also be specified after a second comma, which is used in log
; messages as a "friendly name".  Multiple interfaces may share a single
; member name. An optional state interface may be specified after a third
; comma. This interface will be the one for which app_queue receives device
; state notifications, even though the first interface specified is the one
; that is actually called.
;
; It is important to ensure that channel drivers used for members are loaded
; before app_queue.so itself or they may be marked invalid until reload. This
; can be accomplished by explicitly listing them in modules.conf before
; app_queue.so.  Additionally, if you use Local channels as queue members, you
; must also preload pbx_config.so and chan_local.so (or pbx_ael.so, pbx_lua.so,
; or pbx_realtime.so, depending on how your dialplan is configured).
;
;member => DAHDI/1
;member => DAHDI/2,10
;member => DAHDI/3,10,Bob Johnson
;member => Agent/1001
;member => Agent/1002
;member => Local/1000@default,0,John Smith,SIP/1000

;
; Note that using agent groups is probably not what you want.  Strategies do
; not propagate down to the Agent system so if you want round robin, least
; recent, etc, you should list all the agents in this file individually and not
; use agent groups.
;
;member => Agent/@1             ; Any agent in group 1
;member => Agent/:1,1           ; Any agent in group 1, wait for first
                                ; available, but consider with penalty

; Cola de administradores

[administradores]
music=default
strategy=ringall
timeout=15
retry=5
wrapuptime=0
maxlen = 0
announce-frequency = 0
announce-holdtime = yes
member => SIP/7000
member => SIP/5006
```