Database Monitors

MySQL (Advanced Metrics)

The MySQL (Advanced Metrics) monitor checks the performance of MySQL databases and instances that are running on a system against the thresholds that you define. If MySQL is not responding, the database can process queries but the results will demonstrate behavior that alerts you to a problem.

The MySQL (Advanced Metrics) monitor can:

- determine whether or not a MySQL instance is running on your system
- check whether or not MySQL is listening on a specific port
- check performance values to determine the efficiency of a MySQL instance

Configuring MySQL (Advanced Metrics) Monitors

To configure MySQL (Advanced Metrics) monitors, do the following:

1. Complete the monitor information fields.
   To learn about monitor information fields, see Monitor Identification.
2. Complete the following settings by entering the appropriate Warning and Critical thresholds.
   If the thresholds that you set are exceeded, then *up.time* generates an alert. For more information, see Configuring Warning and Critical Thresholds.
   - MySQL Port
     The number of the port on which the MySQL instance is listening. The default is *3306* .
   - Username
     The user name that is required to log into the MySQL instance.
   - Password
     The password that is required to log into the MySQL instance.
   - Uptime
     The number of seconds that MySQL has been running.
   - Questions
     The number of queries that have been sent to the database.
   - Slow Queries
     The number of queries that take longer than *long_query_time* to complete.
     When started with the *--log-slow-queries[=file_name]* option, MySQL writes a log file containing all SQL statements that took more than the *long_query_time* to execute. The time taken to acquire the initial table locks is not counted as execution time.
     If the *file_name* value is not specified, the information is written to a file with the name of the host machine along with the suffix *-slow.log* . If a filename is given, but not as an absolute path name, the file is written to the default MySQL data directory.
     You can use the *--log-queries-not-using-indexes* option to log queries that do not use indexes to the slow query log.
     Queries handled by the query cache are not added to the slow query log, nor are queries that would not benefit from the presence of an index because a database table has no rows or just one row.
   - Open Tables
     The number of database tables that are opened independently by each concurrent thread.
     Multiple clients can simultaneously issue queries for a given table. Each table is opened independently by each concurrent thread to ensure that multiple client threads do not have different states on the same table.
     For each concurrent thread, the table must be opened twice if two threads access the same table or if a thread accesses the table twice in the same query. Each concurrent open requires an entry in the table cache. The first time any table is opened, it takes file descriptors for the data file and the index file. Each additional use of the table takes only a descriptor for the data file. The index file descriptor is shared among all threads.
     The cache of open tables should be at the level specified by *table_cache* entries. The default value is *64* . MySQL may temporarily open more tables to execute queries.
     Unused tables are closed and removed from the table cache when any of the following occurs:
     - the cache is full and a thread tries to open a table that is not in the cache
     - the cache contains more than *table_cache* entries and a thread is no longer using a table
     - a table flushing operation occurs. This happens when someone issues a *FLUSH TABLES* statement, or executes either the *mysqladmin flush-tables* or *mysqladmin refresh* commands

     When the table cache fills up, the server locates a cache entry to release tables that are not currently in use, in least-recently used order. If a new table needs to be opened, but the cache is full and no tables can be released, the cache is temporarily extended as necessary. When the cache is in a temporarily extended state and a table goes from a used to an unused state, the table is closed and released from the cache.
   - QPSA
     The average number of queries, per second, that must be exceeded before *up.time* generates an alert.
   - Bytes Received
     The number of bytes received by the server.
   - Bytes Sent
     The number of bytes sent by the server to all clients.
   - Delayed Insert Threads
     Select a comparison method for the Warning and Critical Thresholds. Then, enter the number of delayed insert threads that must be exceeded before *up.time* sends an alert.
     The *DELAYED* option for the *INSERT* statement is a MySQL extension to standard SQL that you can use with clients that cannot wait for the *INSERT* statement to complete.
     When a client uses the *INSERT DELAYED* statement, the row is immediately queued to be inserted when the table is not in use by any other thread. *INSERT DELAYED* also bundles inserts from multiple clients and writes them in one block.
     The *DELAYED* option has the following constraints:
     - it only works with *MEMORY* tables
     - " *INSERT DELAYED* " can only be used for " *INSERT* " statements that specify value lists, as the server ignores " *DELAYED* " for " *INSERT DELAYED ... SELECT* " statements
     - the server ignores " *DELAYED* " for " *INSERT DELAYED ... ON DUPLICATE UPDATE* " statements

        - you cannot use " *LAST_INSERT_ID()* " to get the " *AUTO_INCREMENT* " value the statement might generate because the statement returns immediately before the rows are inserted
        - " *DELAYED* " rows are not visible to " *SELECT* " statements until they actually have been inserted
    - Delayed Errors
      The number of delayed insert threads that had an error.
    - Max Used Connections
      The maximum number of connections that have been in simultaneous use since the server was started.
    - Open Files
      The number of open files that must be exceeded before *up.time* generates an alert.
    - Open Streams
      The number of open data streams that must be exceeded before *up.time* generates an alert.
    - Table Locks Immediate
      The number of times that a table lock is acquired immediately. For more information on table locks, see the Knowledge Base article"SQL Server Locks."
    - Table Locks Waited
      The number of table locks waited that must be exceeded before *up.time* generates an alert. For more information on table locks, see the Knowledge Base article"SQL Server Locks."
    - Threads Cached
      The number of threads in the thread cache that must be exceeded before *up.time* generates an alert.
    - Threads Connected
      The maximum number of clients that can be connected to the database at any one time.
    - Threads Running
      The number of threads that are running, which can be used to determine whether or not the database is becoming overloaded.
      If the database is overloaded, the monitor will report an increased number of running queries. However, you can have values that exceed this limit for very short times.
    - QCache Queries in Cache
      The number of queries in the query cache (QCache) that must be exceeded before *up.time* generates an alert.
    - QCache Inserts
      The number of queries added to the query cache.
      You should compare the value of the *qcache_hits* to the total number of select queries to determine the current hit rate. You can increase or decrease *query_cache_size* to find the value which provides optimal performance.
    - QCache Hits
      The number of hits to the query cache (*qcache_hits*) to determine the number of query results taken directly from the cache instead of executing them. When this number is exceeded, *up.time* generates an alert.
      This metric shows the number of query results taken directly from the query cache instead of executing them. You should compare the value of QCache Hits to the total number of your *SELECT* queries to determine the current hit rate. Then, you can increase or decrease the *query_cache_size* to find the value which provides optimal performance.
    - QCache Lowmem Prunes
      The number of *QCache_lowmem_prunes* that can be deleted from the cache because of low memory.
      This variable counts the number of queries that have been removed from the cache to free up memory for caching new queries. The query cache removes the least-recently used queries from the cache.
    - QCache Not Cached
      The maximum number of queries that are not cached.
    - QCache Free Memory
      The amount of free memory for the query cache.
    - QCache Free Blocks
      The number of free memory blocks in query cache.
    - QCache Table Blocks
      The amount of query cache memory fragmentation.
    - Response Time
      Enter the Warning and Critical Response Time thresholds for the overall time required to perform a service check. For more information, see Configuring Warning and Critical Thresholds
3. Click the *Save for Graphing* checkbox to save the data for a metric to the DataStore, which can be used to generate a report or graph.
4. Complete the following settings:
    - Timing Settings (see Adding Monitor Timing Settings Information for more information)
    - Alert Settings (see Monitor Alert Settings for more information)
    - Monitoring Period settings (see Monitor Timing Settings for more information)
    - Alert Profile settings (see Alert Profiles for more information)
    - Action Profile settings (see Action Profiles for more information)
5. Click *Finish*.

MySQL (Basic Checks)

The MySQL (Basic Checks) monitor does the following:

- determines whether or not a host that is running a MySQL database is available
- determines whether or not you can log into a MySQL database
- evaluates a response based on a script that is executed against a database or database instance

Configuring MySQL (Basic Checks) Monitors

To configure MySQL (Basic Checks) monitors, do the following

1. In the MySQL (Basic Checks) monitor template, complete the monitor information fields.
   To learn about monitor information fields, see Monitor Identification.
2. Complete the following fields:

> *Note - If you enter a value in the SID field, up.time can capture the port value from the SID.*

- Port Check (Optional)
  Select this option to open a socket connection that determines whether or not the database is listening on the defined port.
- Username
  The user name that is required to login to the MySQL database.
- Password
  The password that is required to login to the MySQL database.
- Database
  The name of the MySQL database instance.
- Script
  Type or copy the script that you want *up.time* to match against the database. Use this option if your script is short or will not regularly change. This option is required if you do not have access to the file system on the Monitoring Station.
- Script File
  As an alternative to directly entering a script, enter the full path on the Monitoring Station to the script that this monitor will run against the database.
- Match
  Enter a string that you want to match against the return value from the script.
- Response Time
  Enter the Warning and Critical Response Time thresholds. For more information, see Configuring Warning and Critical Thresholds.

3. Click the *Save for Graphing* checkbox to save the data for a metric to the DataStore, which can be used to generate a report or graph.
4. Complete the following settings:
   - Timing Settings (see Adding Monitor Timing Settings Information for more information)
   - Alert Settings (see Monitor Alert Settings for more information)
   - Monitoring Period settings (see Monitor Timing Settings for more information)
   - Alert Profile settings (see Alert Profiles for more information)
   - Action Profile settings (see Action Profiles for more information)
5. Click *Finish*.

Oracle (Advanced Metrics)

The Oracle (Advanced Metrics) monitor captures a number of performance tuning metrics for your Oracle database. Some Oracle metrics are for tuning devices for long-term performance gains, rather than avoiding outages. This applies to following probes: Buffer Cache, Data Dictionary Cache, Disk Sort Ratio, Library Cache, and Redo Log. You should schedule the monitor to gather data less frequently - perhaps every hour or every two days.

Configuring Oracle (Advanced Metrics) Monitors

To configure Oracle (Advanced Metrics) monitors, do the following:

1. In the Oracle (Advanced Metrics) monitor template, complete the monitor information fields.
   To learn how to configure monitor information fields, see Monitor Identification.
2. Complete the following fields:
   - Username
     The user name that is required to login to the database.
   - Password
     The password that is required to login to the database.
   - SID
     The Oracle System Identifier (SID) that identifies this Oracle instance. The SID defaults to the database name. The SID is included in the *CONNECT DATA* paths of the connect descriptors in the *tnsnames.ora* file, and in the definition of the TNS listener in the *listener.ora* file.

     > *Note - If you do not complete the Username and Password fields up.time will attempt to connect to the database. If connection fails, the database returns a SQL exception error.*

   - Buffer Cache Hits Ratio
     Enter the Warning and Critical thresholds for buffer cache hits that are completed without accessing disk I/O. To gather as much application data as possible, you should enter a high buffer cache hits ratio.
     An Oracle database maintains its own buffer cache inside the system global area for each instance. A properly-sized buffer cache can yield a cache hit ratio over 90%. If a buffer cache is too small, the cache hit ratio will be small and the database uses more physical disk I/O. If a buffer cache is too large, then parts of the buffer cache will waste memory resources.
   - Data Dictionary Cache Hits Ratio
     Enter the Warning and Critical thresholds for data dictionary cache hits that are completed without accessing disk I/O.
     The data dictionary cache tables provide information about all of the objects stored in your dictionary - for example tablespaces, files, users, rollback segments, constraints, synonyms. A hit ratio approaching 100% is ideal.
   - Library Cache Hits Ratio
     Enter the Warning and Critical thresholds for the rate at which library cache pin misses occur.
     A pin miss occurs when an session executes a statement that has already been parsed, but which is no longer in the shared pool.
   - Redo Log Space Request Ratio
     Enter the Warning and Critical thresholds for the number of redo log space requests per minute that have been made since the server was started.
   - Disk Sort Rate
     Enter the Warning and Critical thresholds for the rate of Oracle sorts that are too large to be completed in memory and which are sorted using a temporary segment.
   - Active Sessions
     Enter the Warning and Critical thresholds for the number of active sessions based on the value of *V$PARAMETER.PROCESSES* in the file *init.ora* .

- Oracle Blocking Sessions
  Enter the Warning and Critical thresholds for the number of sessions that are preventing other sessions from committing changes to the Oracle database.
- Oracle Idle Sessions
  Enter the Warning and Critical thresholds for the number of Oracle sessions that are idle, as determined by the Time Idle value that you specify. Only the sessions that have been idle for the duration (measured by the Time Idle value), in seconds, are considered idle.
- Response Time
  Enter the Warning and Critical Response Time thresholds for the length of time a service check needs to complete. For more information, see Configuring Warning and Critical Thresholds.

3. Click the *Save for Graphing* checkbox to save the data for a metric to the DataStore, which can be used to generate a report or graph.
4. Complete the following settings:
   - Timing Settings (see Adding Monitor Timing Settings Information for more information)
   - Alert Settings (see Monitor Alert Settings for more information)
   - Monitoring Period settings (see Monitor Timing Settings for more information)
   - Alert Profile settings (see Alert Profiles for more information)
   - Action Profile settings (see Action Profiles for more information)
5. Click *Finish*.

Oracle (Basic Checks)

The Oracle (Basic Checks) monitor does the following:

- determines whether or not a host running an Oracle database is available
- determines whether or not an Oracle service is running on a system
- determines whether or not you can log into an Oracle database
- evaluates a response based on a script that you have executed against a database or database instance

> *Note - Use the Oracle Tablespace Check monitor to check Oracle tablespaces.*

Configuring Oracle (Basic Checks) Monitors

To configure Oracle (Basic Checks) monitors, do the following

1. In the Oracle (Basic Checks) monitor template, complete the monitor information fields.
   To learn about monitor information fields, see Monitor Identification.
2. Complete the following fields:
   - Port
     The number of the port on which the Oracle service is listening.
     Note - If you enter a value in the SID field, up.time can capture the port value from the SID of the Oracle instance.
   - Port Check (Optional)
     Select this option to open a socket connection that determines whether or not the database is listening on the defined port.
   - Username
     The user name that is required to login to the Oracle database.
   - Password
     The password that is required to login to the Oracle database.
   - SID
     The Oracle System Identifier (SID) that identifies the Oracle instance. The SID defaults to the database name.
     If you enter a value in this field, *up.time* can capture the number of the port on which Oracle is listening.
   - Script File
     Click the *Script File* check box and then enter the full path on the Monitoring Station to the script that this monitor will run against the database.
     Note - If you configured your database to allow logins with a user name and password and you specify the script file but no login information, the script will fail and an error message appears in the Global Scan panel. The script will run if you have configured your database to allow logins without a user name and password.
   - Script
     Select this option and then type or copy the script that you want *up.time* to against the database into this text box. Use this option if you do not have access to the file system on the Monitoring Station or if your script is short or will not regularly change.
   - Match
     Enter a string that you want to match against the return value from the script.
   - Response Time
     Enter the Warning and Critical Response Time thresholds. For more information, see Configuring Warning and Critical Thresholds.
3. Click the *Save for Graphing* checkbox to save the data for a metric to the DataStore, which can be used to generate a report or graph.
4. Complete the following settings:
   - Timing Settings (see Adding Monitor Timing Settings Information for more information)
   - Alert Settings (see Monitor Alert Settings for more information)
   - Monitoring Period settings (see Monitor Timing Settings for more information)
   - Alert Profile settings (see Alert Profiles for more information)
   - Action Profile settings (see Action Profiles for more information)
5. Click *Finish*.

Oracle Tablespace Check

The Oracle Tablespace Check monitors the size (as a percentage) of individual tablespaces within Oracle database instances. The Oracle Tablespace Check alerts you when a tablespace in your instance exceeds the defined thresholds.

Each database is logically divided into one or more tablespaces. One or more data files are explicitly created for each tablespace to physically store the data in a tablespace. The combined size of the data files in a tablespace is the total storage capacity of the tablespace. For example:

ID#	TABLESPACE_NAME	Total Bytes	Bytes Free	% Free
6	INDX	56,623,104	56,614,912	99.99
8	OEM_REPOSITORY	31,465,472	3,473,408	11.04
3	RBS	104,857,600	75,489,280	71.99
9	SUPPORT	52,428,800	52,420,608	99.98
1	SYSTEM	56,623,104	2,850,816	5.03
4	TEMP	71,303,168	71,294,976	99.99
2	TOOLS	8,388,608	8,380,416	99.90
7	UNANET	314,572,800	300,539,904	95.54
5	USERS	20,971,520	20,963,328	99.96

In the above table, the *SYSTEM* tablespace is over 95% full. If you set the Warning threshold to 90%, and the Critical threshold to 95%, the Oracle Tablespace Check returns a status of Critical.

Use the Oracle (Basic Checks) monitor to determine the availability of Oracle databases, the performance of services, and the matched response of scripts. For more information, see Sybase.

Configuring Oracle Tablespace Check Monitors

To configure Oracle Tablespace Check monitors, do the following:

1. In the Oracle Tablespace Check monitor template, complete the monitor information fields.
   To learn how to configure monitor information fields, see Monitor Identification.
2. Complete the following fields:
   - Port
     The number of the port on which the Oracle service is listening. The default is *1521* .
   - Username
     The user name that is required to login to the Oracle database.
   - Password
     The password that is required to login to the Oracle database.
   - SID
     The Oracle System Identifier (SID) that identifies the Oracle instance. The SID defaults to the database name.
     The SID is a unique name for an Oracle instance to switch between Oracle databases. The SID is included in the *CONNECT DATA* paths of the connect descriptors in the *tnsnames.ora* file. As well, the SID is in the definition of the TNS listener in the *listener.ora* file.
     Note - If you do not complete the Username , Password , SID fields up.time will attempt to connect to the database. If connection fails, the database returns a SQL exception error.
   - Full Warning Threshold (Mandatory)
     Enter a value that will change the status of the Oracle Tablespace Check from OK to Warning.
     The warning threshold should be a percentage of the maximum file size, against which the monitor will check data files and log files.
   - Full Critical Threshold (Mandatory)
     Enter a value that will change the status of the Oracle Tablespace Check from OK to Warning.
     The critical threshold should be a percentage of the maximum file size, against which the monitor will check data files and log files.
   - Response Time
     Enter the Warning and Critical Response Time thresholds for the length of time that a service check takes to complete. For more information, see Configuring Warning and Critical Thresholds.
3. Click the *Save for Graphing* checkbox to save the data for a metric to the DataStore, which can be used to generate a report or graph.
4. Complete the following settings:
   - Timing Settings (see Adding Monitor Timing Settings Information for more information)
   - Alert Settings (see Monitor Alert Settings for more information)
   - Monitoring Period settings (see Monitor Timing Settings for more information)
   - Alert Profile settings (see Alert Profiles for more information)
   - Action Profile settings (see Action Profiles for more information)
5. Click *Finish.*

SQL Server (Basic Checks)

The SQL Server (Basic Checks) monitor compares the performance of SQL Server databases and instances running on a system to the thresholds that you define. The SQL Server (Basic Checks) monitor does the following:

- determines whether or not SQL Server is running on your system
- checks whether or not SQL Server is listening on a specific port
- determines whether or not SQL Server can process queries
- checks for values in base and computed tables
  You can use regular expressions to identify a wide range of responses and to detect problems after they occur. You can also run scripts through *up.time* to alert you when a database component that is
  being monitored is not performing as required.

> *Note - To configure this monitor, you should have a strong knowledge of regular expressions, Transact-SQL, and SQL Server.*

Configuring SQL Server (Basic Checks) Monitors

To configure SQL Server monitors, do the following:

1. In the SQL Server (Basic Checks) monitor template, complete the monitor information fields.
   To learn about monitor information fields, see Monitor Identification.
2. Complete the following fields:
   - SQL Server Port
     The number of the port on which SQL Server is listening.
     SQL Server uses Static Port Allocation or Dynamic Port Allocation ports. For more information, see the Knowledge Base article"SQL Server Ports."
   - Port Check (Optional)
     Select this option to open a socket connection that determines whether or not the database is listening on the defined port.
     You should perform a port check because SQL Server can communicate statically on a defined or default port, or communicate dynamically on a port assigned by the operating system.
   - Username
     The user name that is required to log into the SQL Server database.
   - Password
     The password that is required to log into the SQL Server database.
   - Instance
     The name of the SQL server instance to which you want to connect.
     You can install multiple versions of Microsoft SQL Server on one computer. When installing a new version of SQL Server 2000, or maintaining an existing installation, you can specify it as:
     - A default instance of SQL Server
       This instance is identified by the network name of the computer on which it is running. SQL Server version 6.5 or SQL Server version 7.0 servers can operate as default instances. However, a computer can have only one version functioning as the default instance at one time.
     - A named instance of SQL Server
       This instance is identified by the network name of the computer plus an instance name, in the format <*computername*>\<*instancename*> .
       Most applications must use SQL Server 2000 client components to connect to a named instance. However, you can use the SQL Server version 7.0 Client Network Utility to configure a server alias name that the SQL Server version 7.0 client components can use to connect to a named instance.
       A computer can concurrently run any number of named instances of SQL Server. An instance name cannot exceed 16 characters.
   - Database
     The name of the SQL Server database that you want to monitor.
     up.time views each database along the path /<*system*>/<*instance*>/<*database*> .
     Each instance of SQL Server has four system databases - *master* , *model* , *tempdb* , and *msdb* - and one or more user databases.
     Depending on their permissions, users can access some or all of the databases in an instance.
     A connection to an instance is associated with a particular database on the server, called the *current database* . You can switch from one database to another using the Transact-SQL *USE database_name* statement.
     up.time gathers information from all of the databases in all instances on a system and aggregates this information in the metrics it returns to you. Unless you must identify a particular database on your system - for example, you have applied a name to the default instance - you should leave the *Database* field blank.
   - Script File
     Click the *Script File* check box and then enter the full path on the Monitoring Station to the script that this monitor will run against the database.
     Note - If you configured your database to allow logins with a user name and password and you specify the script file but no login information, the script will fail and an error message appears in the Global Scan panel. The script will run if you have configured your database to allow logins without a user name and password.
   - Script
     Click the *Script* checkbox and then type or copy the script that you want *up.time* to against the database into this text box. Use this option if you do not have access to the file system on the Monitoring Station or if your script is short or will not regularly change.
   - Match
     The value to match the script results against, which can be either a string or a regular expression. For more information, see Comparison Methods. For example, you can enter the following in the *Match* text box:
     ^[OK]+
     Where:
     - ^ means start the match at the beginning of the line.
     - *[OK]* is the pattern to match.
     - + is the pattern to match anywhere on the line.

       The value that your script returns can be a string that you can match to. If you match to the value you checked for, the status of the service monitor is OK. Otherwise, the status of the service monitor is Critical.
   - Response Time
     Enter the Warning and Critical Response Time thresholds. For more information, see Configuring Warning and Critical Thresholds.
3. Click the *Save for Graphing* checkbox to save the data for a metric to the DataStore, which can be used to generate a report or graph.
4. Complete the following settings:
   - Timing Settings (see Adding Monitor Timing Settings Information for more information)
   - Alert Settings (see Monitor Alert Settings for more information)
   - Monitoring Period settings (see Monitor Timing Settings for more information)
   - Alert Profile settings (see Alert Profiles for more information)
   - Action Profile settings (see Action Profiles for more information)
5. Click *Finish*.

SQL Server (Advanced Metrics)

SQL Server (Advanced Metrics) monitor collects information on the availability and performance of individual SQL Server databases.

You only need to configure one SQL Server (Advanced Metrics) monitor for each system. You can, however, create multiple SQL Server (Advanced Metrics) monitors for a system if you need to separately capture different SQL Server performance metrics. See the section for more information.

For example, consider a host configured to have the following:

- an *up.time* agent installed
- two database instances
- four databases

The SQL Server (Advanced Metrics) monitor can capture performance information from all four databases. It can also aggregate the information to present a single performance value for each metric.

Using Multiple SQL Server (Advanced Metrics) Monitors

You can create several SQL Server (Advanced Metrics) monitors for a system if you must separately capture different SQL Server performance metrics. For example, the SQL Server (Advanced Metrics) monitor provides metrics for SQL Server locks including lock requests, waits, and averages. For information about locks, see the Knowledge Base article“SQL Server Locks.”

Lock requests do not always provide meaningful information. When you compare the length of waits with the number of lock requests, the length of the lock waits should be much lower than requests. If the lengths of waits and requests are about the same, then there is a performance problem. When the average lock wait time is high, there is a problem with SQL Server.

Configuring SQL Server (Advanced Metrics) Monitors

To configure SQL Server (Advanced Metrics) monitors, do the following:

1. In the SQL Server (Advanced Metrics) monitor template, complete the monitor information fields.
   To learn how to configure monitor information fields, see Monitor Identification.
2. In the *Instance* field, type the name of the SQL server instance to which you want to connect.
   Note - If you have configured your agent to use SSL but do not select Use SSL , up.time will not receive performance information.
3. Complete the following options by clicking the checkbox beside each option, then specifying a warning and critical threshold.
   If the thresholds that you set are exceeded, then *up.time* generates an alert. For more information, see Configuring Warning and Critical Thresholds.
   - Lock Wait / Sec.
     The amount of time, in seconds, to wait for a database lock. For more information about locks, see the Knowledge Base article“SQL Server Locks.”
   - Lock Requests / Sec.
     The number of new database locks and lock conversions that are requested from the lock manager every second. For more information about locks, see the Knowledge Base article“SQL Server Locks.”
   - Average Lock Wait Time
     The average time, in milliseconds, that you must wait for database locks to clear before *up.time* sends an alert.
   - User Connections
     The number of user connections that are allowed before *up.time* sends an alert.
     For example, a single host is running two databases. There are five users logged on to the first database and three users logged on to the second database. The total number of user connections is eight.
   - Transactions / Sec.
     In the Warning and Critical threshold fields, enter the number of transactions started for the databases across the host per second.
   - Data File(s) Size / KB
     The cumulative size of all the files in all of the databases on the host system.
     This metric is returned from the SQL Server Database object. The Database object provides such information about the database as the amount of free log space available or the number of active transactions in the database. There can be multiple instances of this object.
   - Total Latch Wait Time (ms)
     The total time, in milliseconds, that it takes to complete the latch requests that were waiting over the last second.
   - Latch Waits / Sec.
     The number of latch requests that were not immediately granted, and which waited before being granted.
   - Average Latch Wait Time (ms)
     The average time, in milliseconds, that latch requests had to wait before being granted.
   - Maximum Workspace Memory (KB)
     The maximum amount of memory, in kilobytes, that the server has available to execute such processes as sort, bulk copy, hash, and index creation.
     This metric is returned by the SQL Server Memory Manager object, which monitors overall server memory usage. By monitoring overall server memory usage, you can determine whether or not:
     - Bottlenecks exist due to a lack of available physical memory for storing frequently accessed data in cache. If so, SQL Server must retrieve the data from the disk.
     - You can improve query performance by adding more memory or by making more memory available to the data cache or to SQL Server internal structures.
   - Connection Memory (KB)
     The total amount of dynamic memory, in kilobytes, that the server is using to maintain connections.
   - SQL Cache Memory (KB)
     The amount of memory, in kilobytes, that the server is using for the dynamic SQL cache.
   - Total Server Memory (KB)
     The total amount of committed memory from the buffer pool, in kilobytes, that the server is using.
   - Response Time
     Enter the Warning and Critical Response Time thresholds. If the amount of time taken to perform a check exceeds the defined thresholds, it could indicate a problem that requires investigation.

4. To save the data from the thresholds for graphing or reporting, click the *Save for Graphing* checkbox beside each of the metrics that you selected in step 3.
5. Complete the following settings:
    - Timing Settings (see Adding Monitor Timing Settings Information for more information)
    - Alert Settings (see Monitor Alert Settings for more information)
    - Monitoring Period settings (see Monitor Timing Settings for more information)
    - Alert Profile settings (see Alert Profiles for more information)
    - Action Profile settings (see Action Profiles for more information)
6. Click *Finish*.

SQL Server Tablespace Check

The SQL Server Tablespace Check monitor evaluates the size of data files within SQL Server databases. *up.time* gathers information from all the databases across all instances on a system and aggregates this information in the metrics that it returns.

This monitor also reports whether or not any of the data files in a filegroup or any log file in any database in the instance exceeds warning and critical thresholds. If warning or critical thresholds are exceeded, *up.time* generates an alert.

Structure of a SQL Server Database

Each SQL Server database consists of at least two files:

- a primary data file, with the extension *.mdf*
- a log file, with the extension *.ldf*

There are also secondary data files, with the extension *.ndf* . A database can have only one primary data file, zero or more secondary data files, and one or more log files. Each database file can only be used by one database.

In a database, data files store persistent data. For ease of management, you can group one or more data files into logical tablespaces. The SQL Server equivalent of an Oracle tablespace is the *filegroup* . SQL Server filegroups come under and are associated with the individual databases. The SQL Server data hierarchy is:

Instance / Database / FileGroup / Data file

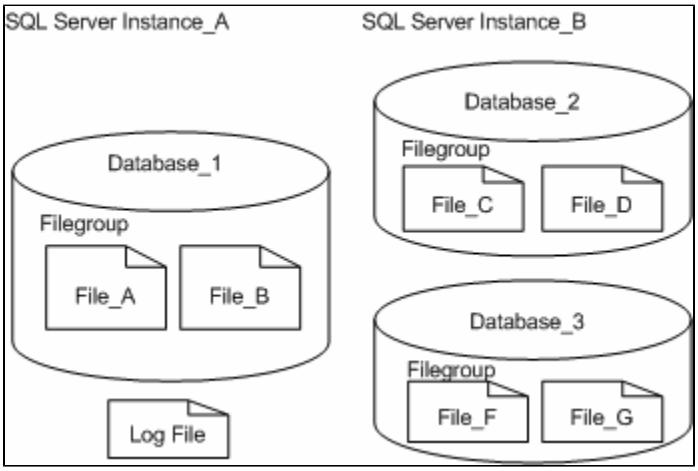
Each data file can be a member of only one filegroup, but the log files are managed separately from one another. There are three types of filegroups:

- primary
- user-defined
- default

When you configure your SQL Server databases, you can the maximum size of data files to prevent disk drives from running out of space. If you do not specify the size of data files, the database assumes that the size is unlimited.

> *Note - Unlimited datafiles cannot initiate up.time alerts. Since up.time measures datafile and log file size as a percentage of their maximum size, if a datafile size has an infinite maximum size, the percent of maximum datafile size must be near zero. You should always set the maximum size of each datafile.*

The following diagram illustrates six data files in three file groups in three databases across two instances of a system.



If you set SQL Server Instance_B with a Critical threshold of 90% and a Warning threshold of 70%, the SQL Server Tablespace Check monitor watches the size of all data files in that instance. The monitor sends an alert if any of the files reaches or exceeds the defined thresholds.

Configuring SQL Server Tablespace Check Monitors

To configure SQL Server Tablespace Check monitors, do the following:

1. In the SQL Server Tablespace Check monitor template, complete the monitor information fields.
   To learn how to configure monitor information fields, see Monitor Identification.
2. Complete the following fields:
   - SQL Server Port
     The number of the port on which the SQL Server is listening.
     SQL Server can use static or dynamic ports. For information about SQL ports and how to determine and configure port allocation, see the Knowledge Base article “Configuring SQL Server Ports.”
   - Username
     The user name that is required to login to the SQL Server database.
     When a user connects through a Windows user account, SQL Server re-validates the account name and password by contacting a Windows domain controller to determine the network user name. SQL Server then verifies the credentials of the users, and then permits or denies login access.
   - Password
     The password that is required to login to the SQL Server database.
     When a user connects with a specified login name and password from a non-trusted connection, SQL Server determines if a SQL Server login account has been set up and if the specified password matches the one previously recorded. If SQL Server does not find a login account, authentication fails and the user receives an error message.
     SQL Server authentication is provided for backward compatibility because applications written for SQL Server version 7.0 or earlier may require the use of SQL Server logins and passwords.
     *Note - If you do not complete the Username and Password fields up.time will attempt to connect to the database. If the connection attempt fails, the database returns a SQL exception error.*SQL Server can use one of the following authentication modes:
     - Windows Authentication Mode
       Enables users to connect to a SQL Server instance through a Windows user account.
     - Mixed Mode
       Enables users who to connect to a SQL Server instance through a Windows account to use either Windows authentication or SQL Server authentication.
   - Instance
     The SQL Server instance name. This is usually the default instance.
     You can install multiple instances of SQL Server on one computer. An instance can be:
     - The default instance
       This instance is identified by the network name of the computer on which it is running. Applications using client software from earlier versions of SQL Server can connect to a default instance. SQL Server version 6.5 or 7.0 servers can operate as default instances. A computer can have only one version functioning as the default instance at a time.
     - A named instance of SQL Server
       This instance is identified by the network name of the computer plus an instance name, in the format <*computername*>\<*instancename*> .
       Most applications must use SQL Server client components to connect to a named instance. However, you can use the SQL Server version 7.0 Client Network Utility to configure a server alias name that the version 7.0 client components can use to connect to a named instance of SQL Server. A computer can concurrently run any number of named instances of SQL Server. A named instance can run at the same time as an existing installation of SQL Server version 6.5 or SQL Server version 7.0.
       The instance name cannot exceed 16 characters.

     A new instance name must begin with a letter, an ampersand (*&*), or an underscore (_), and can contain numbers, letters, or other characters. Do not use SQL Server sysnames and reserved names as instance names. For example, *default* is a reserved name and should not be used as an instance name.
     You can have multiple instances of SQL Server installed on one computer. Each instance operates independently from the other instances, and applications can connect to any of the instances.
   - Full Warning Threshold
     Enter a percentage of the maximum file size you want to set as your warning threshold.
   - Critical Warning Threshold
     Enter a percentage of the maximum file size you want to set as your critical threshold.
   - Response Time
     Enter the Warning and Critical Response Time thresholds for the length of time a service check takes. For more information, see Configuring Warning and Critical Thresholds
3. Click the *Save for Graphing* checkbox to save the data for a metric to the DataStore, which can be used to generate a report or graph.
4. Complete the following settings:
   - Timing Settings (see Adding Monitor Timing Settings Information for more information)
   - Alert Settings (see Monitor Alert Settings for more information)
   - Monitoring Period settings (see Monitor Timing Settings for more information)
   - Alert Profile settings (see Alert Profiles for more information)
   - Action Profile settings (see Action Profiles for more information)
5. Click *Finish*.

Sybase

The Sybase monitor does the following:

- determines if the database is responding on the standard port
- sends Sybase/Transact-SQL scripts to the database for processing
  The Transact-SQL scripts can be very basic SQL statements, such as:
  sphelp_db sampledb1; exit (select 1);
  The scripts can also be more complex statements that involve functions and other data processing.

Configuring Sybase Monitors

To configure Sybase monitors, do the following:

1. In the Sybase monitor template, complete the monitor information fields.
2. To learn how to configure monitor information fields, see Monitor Identification.
3. Complete the following fields:

- Port
  The number of the port number on which the database is listening. The default is *5000* .
- Port Check (Optional)
  Select this option to open a socket connection that determines whether or not the database is listening on the defined port.
- Username
  The user name that is required to login to the database.
- Password
  The password that is required to login to the database.
- Database
  The name of the Sybase database to which you want to connect.
- Script
  Click the *Script* checkbox and then type or copy the script that you want *up.time* to against the database into this text box. Use this option if you do not have access to the file system on the Monitoring Station or if your script is short or will not regularly change.
- Script File
  Click the *Script File* check box and then enter the full path on the Monitoring Station to the script that this monitor will run against the database.
  Note - If you configured your database to allow logins without a user name and password and you specify the script file but no login information, the script will fail and an error message appears in the Global Scan panel. The script will run if you have configured your database to allow logins without a user name and password.
- Match (Regular Expression)
  Enter a regular expression that you want to match against the string returned from the database. If the string matches, the status is OK. Otherwise, the status is Critical.
- Response Time
  Enter the Warning and Critical Response Time thresholds. For more information, see Configuring Warning and Critical Thresholds.

4. Click the *Save for Graphing* checkbox to save the data for a metric to the DataStore, which can be used to generate a report or graph.
5. Complete the following settings:
   - Timing Settings (see Adding Monitor Timing Settings Information for more information)
   - Alert Settings (see Monitor Alert Settings for more information)
   - Monitoring Period settings (see Monitor Timing Settings for more information)
   - Alert Profile settings (see Alert Profiles for more information)
   - Action Profile settings (see Action Profiles for more information)
6. Click *Finish*.