Monitoring MySQL Server

eG Innovations Product Documentation

www.eginnovations.com

Total Performance Visibility

Table of Contents

Table of Figures

Chapter 1: Introduction

For storage and retrieval of persistent data in an IT infrastructure, application components rely on database servers. A database server is responsible for reliably managing a large amount of data in a multi-user environment so that many users can concurrently access the same data. At the same time, a database server must also prevent unauthorized access and provide efficient solutions for failure recovery.

For ensuring high availability, performance, and security, a database server includes a wealth of data storage, caching, and retrieval functions. To ensure peak performance, a database server needs to be continuously monitored and tuned. In an operational database, specific tables may grow in size, there by choking one or more of the database's tablespaces. Sometimes, there may be a sudden change in workload to the database, resulting in an increase in the number of simultaneously processed transactions. This scenario could result in a performance bottleneck at the database server. Continuous monitoring and optimization of the database server is essential for ensuring that the database server operates at its peak.

The eG Enterprise suite is programmed with a variety of tests that are designed to monitor the critical parameters of MySQL servers. This document describes how the eG Enterprise suite performs monitoring for MySQL servers.

Chapter 2: How to Monitor MySQL Server Using eG Enterprise?

eG Enterprise can monitor the MySQL server in an agent-based or an agentless manner. For monitoring the MySQL server in the agent-based manner, deploy an eG agent on the target host. In case of the agentless approach, a remote agent used to monitor the MySQL server should be deployed on a remote Windows host in the environment.

The broad steps for monitoring the MySQL server using eG Enterprise are as follows:

- Setting up the pre-requisites to enable the eG agent to pull out performance metrics from the target MySQL server. See Section **2.1**
- Managing the MySQL server. For the detailed procedure, refer to Section **2.2**
- Configuring the tests. See Section **2.3**

These steps have been discussed in following sections.

2.1 Pre-requisites for Monitoring the MySQL Server

To run the **MySQL** tests, the eG agent should be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges. If such a user does not pre-exist, then, in the *user* table of the *mysql* database of the target MySQL server, you need to manually create a user account with the aforesaid privileges. To create and configure such a user, follow the procedure discussed below:

1. To create a new user account, you must connect to the MySQL server as the MySQL root user. For that, first login to the MySQL host, and at the command prompt, issue the following command:

   mysql -u root

   If you have assigned a password to the *root* account, you will also need to supply a - password or -p option, as shown below:

   mysql -u root -pegurkha

2. After successfully logging into the MySQL server, issue the following statement to access the *mysql* database, which holds the *user* table:

use mysql

3. Then, at the MySQL prompt, issue the following command to create a user:

   CREATE USER '<username>'@'<IP_address_of_eG_agent>' IDENTIFIED BY '<password>';

   GRANT PROCESS,SELECT ON . TO '<username>'@'<IP_address_of_eG_agent>';

   For instance, to ensure that user *john* (with password *john*) is able to connect to the MySQL server (being monitored) from the eG agent host, *192.168.8.91*, the following command is to be issued:

   CREATE USER 'john'@'192.168.8.91' IDENTIFIED BY 'john';

   GRANT PROCESS,SELECT ON . TO 'john'@'192.168.8.91';

   Note:

   a. The *CREATE* and *GRANT* commands are case-sensitive; therefore, take care while specifying the user name, password, and privileges.

   b. Only the IP address of the eG agent's host can be provided as part of the *CREATE* command's syntax; the host name of the eG agent cannot be provided instead.

4. To ensure that the external agent is able to execute the **MySQL Network** test, make sure that you create a user with the same credentials (i.e., name and password) and privileges as above and map that user to the IP address of the external agent. For instance, in the example above, to enable the external agent at IP address *192.168.8.92* to run the **MySQL Network** test, your command should be:

   CREATE USER 'john'@'192.168.8.92' IDENTIFIED BY 'john';

   GRANT PROCESS,SELECT ON . TO 'john'@'192.168.8.92';

5. Once the above-mentioned commands execute successfully, the user table will be updated with two records for the user account that was newly created - one mapped to the internal/remote agent's IP address and another mapped to the external agent's IP address.

Once the pre-requisites are set, manage the MySQL server using the procedure explained in the Section **2.2**. The MySQL server should be added/managed in the eG administrative interface using its IP address only, and not its host name. Then, proceed to configure the tests. While doing so, remember to configure the **USER** name and **PASSWORD** parameters with the name and password (respectively) that corresponds to the eG agent's IP address in the *user* table.

2.2 Managing the MySQL Server

eG Enterprise is capable of auto-discovering the MySQL servers in the environment. The discovered server can be managed using the eG admin interface. The procedure for achieving this is discussed below:

1. Login to the eG administrative interface as an administrator (admin).
2. Check whether the MySQL server has been auto-discovered. If not, run discovery using the **DISCOVERY** page (Infrastructure -> Components -> Discover) or manually add the server using the **COMPONENTS** page (Infrastructure -> Components -> Add/Modify). The eG Enterprise automatically manages the manually added components. The discovered components however, need to go through a manual management exercise, using the **COMPONENTS - MANAGE/UNMANAGE** page (Infrastructure -> Components -> Manage/unmanage). This process is depicted by Figure 2.1 and Figure 2.2 below.

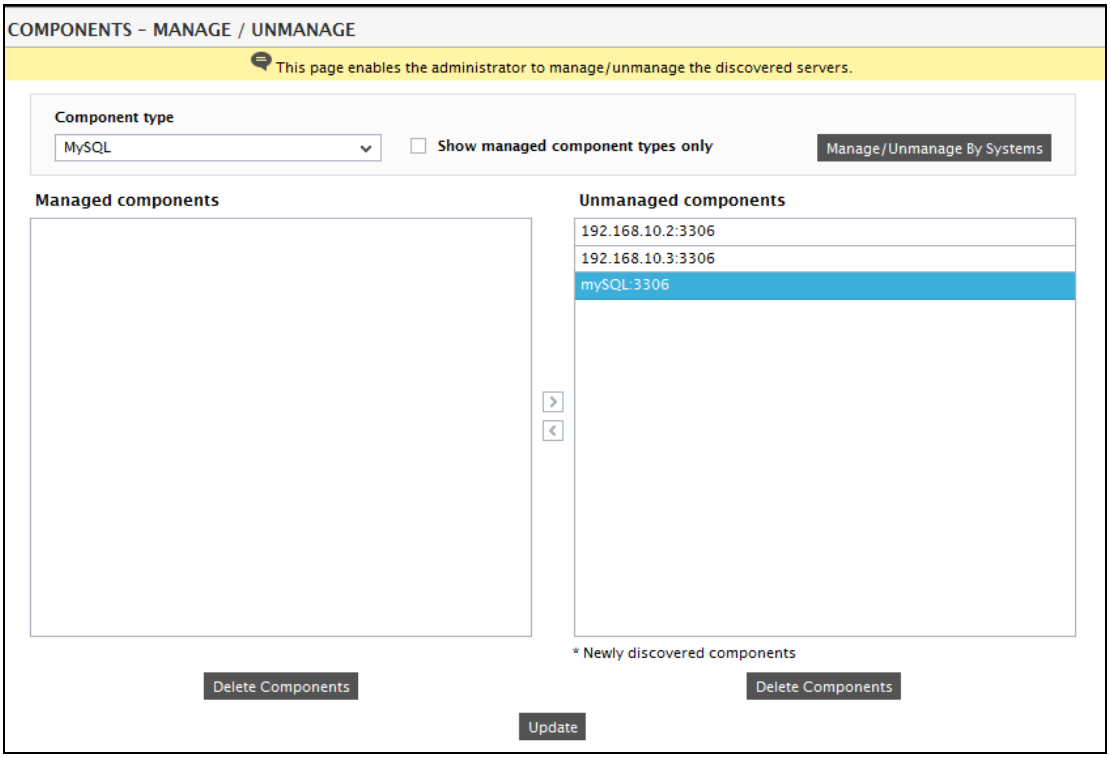


Figure 2.1: Selecting the MySQL server to be managed

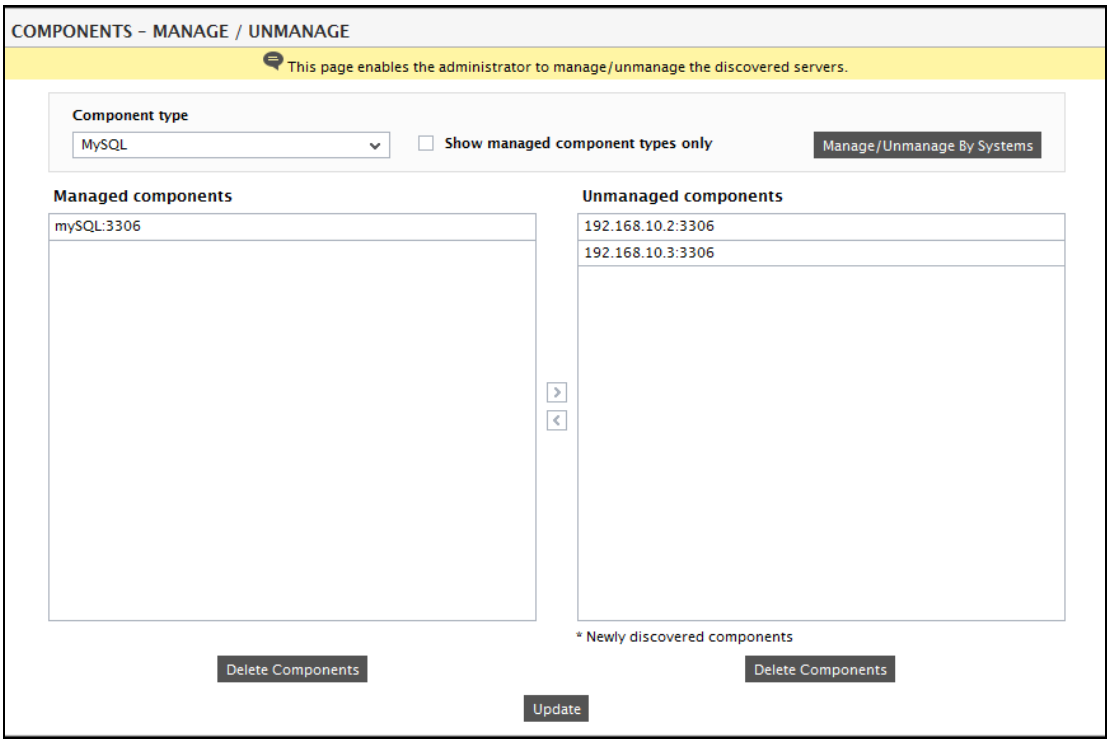


Figure 2.2: Managing the MySQL server

3. Click the **Update** button to manage the discovered MySQL server.

2.3 Configuring the tests

1. When you try to sign out of the user interface, you will be prompted to configure a list of tests for the managed MySQL server (see Figure 2.3).

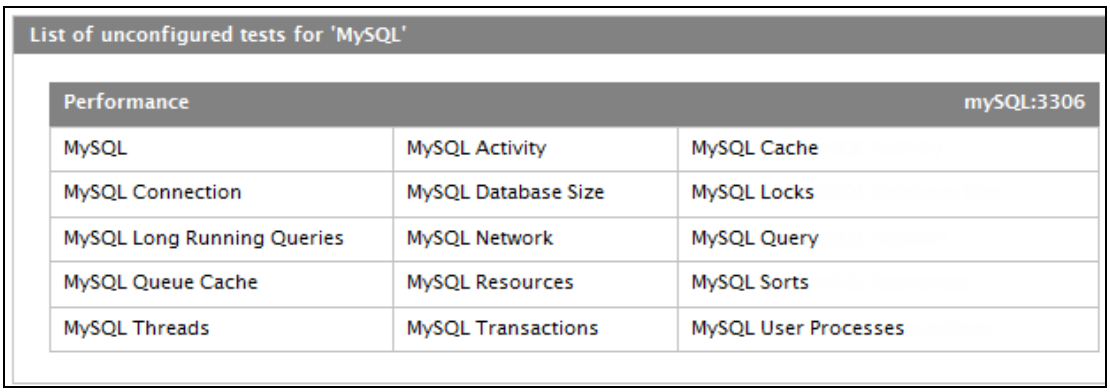


Figure 2.3: List of unconfigured tests for the MySQL server

2. Click on the tests to configure. To know the configuration parameters of the tests and the metrics that the tests report, refer to **Monitoring MySQL Servers** chapter.
3. Finally signout from the eG administrative interface.

Chapter 3: Monitoring MySQL Servers

MySQL is a multithreaded, multi-user SQL database management system, and is one of the most popular databases in the market today. Owing to a rise in its popularity in the past decade, the need to ensure the continuous availability and optimal performance of the MySQL database server has also attained significance.

eG Enterprise provides an exclusive MySQL monitoring model that runs quick health checks on the MySQL database server at configured intervals, and proactively alerts administrators to potential bottlenecks to the performance of the server.

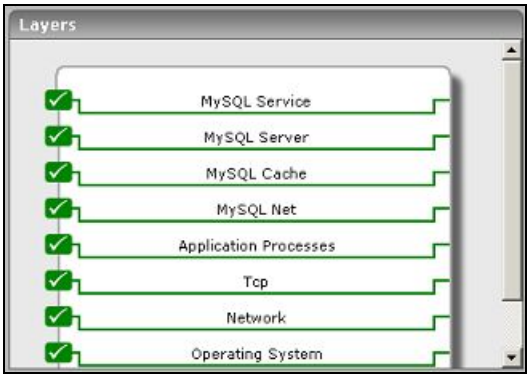


Figure 3.1: The layer model of the MySQL server

Using the model depicted by Figure 3.1, administrators can determine the following:

- Is the database server available? If so, how quickly does it respond to user requests?
- Is the server overloaded?
- Are clients able to connect to the server, or are there too many connection failures?
- Are connections been closed properly? Are there an unusual number of open connections to the server?
- Is the query cache been utilized optimally?
- Has adequate memory been allotted to the cache?
- Is the key buffer cache utilized well?
- Is query execution efficient, or do queries need to be optimized for better performance?
- Are rollbacks kept at a minimum?

- Should the sort_buffer be increased?
- How is the overall locking activity on the server? Are too many requests waiting to acquire locks?

This section will deal with the first four layers of the layer model only, as the rest of the layers have been discussed elaborately in the *Monitoring Unix and Windows Servers* document.

3.1 The MySQL Net Layer

This layer measures the network connectivity of the MySQL server by indicating the availability of the server over a network and its responsiveness to requests. In addition, the layer also tracks the data traffic to and from the server, and the overall health of the client connections to the server.

Figure 3.2: The tests associated with the MySQL Net layer

3.1.1 MySQL Test

This test monitors the availability and responsiveness of the MySQL database server by emulating a client connecting and executing queries on the MySQL server.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server

5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
MySQL server availability:	Indicates the availability of the server to execute a user query.	Percent	The availability is 100% when the server does respond to a request and 0% when it is not. Availability problems may be caused by a misconfiguration / malfunctioning of the database server, or if the server has not been started. Besides the above, this measure will report that the server is unavailable even if a connection to the database instance is unavailable, or if a query to the database fails. In this case, you can check the values of the DB connection availability and Query processor availability measures to know what is exactly causing the database instance to not respond to requests - is it owing to a connection unavailability? or is it due to a query failure?
Connection time to database server:	Indicates the time taken to connect to the database server.	Secs	A dramatic increase in this value may be because the server has reached its connection handling capacity.
Query execution time:	Indicates the time taken to execute a database	Secs	A dramatic increase in this value could indicate a processing

Measurement	Description	Measurement Unit	Interpretation
	query.		bottleneck with the database server.
Total response time:	Indicates the time taken by the database server to respond to a user query. This is the sum of the connection and query execution times.	Secs	A sudden increase in response time is indicative of a bottleneck with the database server.
DB connection availability:	Indicates whether the database connection is available or not.	Percent	If this measure reports the value 100 , it indicates that the database connection is available. The value 0 on the other hand indicates that the database connection is unavailable. A connection to the database may be unavailable if the database is down or if the database is listening on a port other than the one configured for it in the eG manager or owing to a poor network link. If the MySQL server availability measure reports the value 0, then, you can check the value of this measure to determine whether/not it is due to the unavailability of a connection to the server.
Query processor availability:	Indicates whether the database query is executed successfully or not.	Percent	If this measure reports the value 100, it indicates that the query executed successfully. The value 0 on the other hand indicates that the query failed. In the event that the MySQL server availability measure reports the value 0, check the value of this measure to figure out whether the failed query is the

Measurement	Description	Measurement Unit	Interpretation
			reason why that measure reported a server unavailability.
Records fetched:	Indicates the number of records fetched from the database.	Number	The value 0 indicates that no records are fetched from the database

3.1.2 MySQL Network Test

This test monitors the data transmission between the MySQL server and its clients.

Target of the test : A MySQL server

Agent deploying the test : An external agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Data transmit rate:	The rate of data being transmitted by the server in response to client requests during the last	KB/Sec	The data transmission rate reflects the workload on the server.

Measurement	Description	Measurement Unit	Interpretation
	measurement period		
Data received rate:	The rate of data received by the server from clients during the last measurement period	KB/Sec	This measure also characterizes the workload on the server.

3.1.3 MySQL Connection Test

This test gives information about client connections to a MySQL server.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Connection create rate:	The rate at which applications attempted	Conns/Sec	The data transmission rate reflects the workload on the server.

Measurement	Description	Measurement Unit	Interpretation
	to connect to the MySQL server during the last measurement period		
Active connections:	The number of currently open connections to the MySQL server	Number	A high value may be indicative of connections not being closed properly by applications, or a bottleneck in the server
Connection aborts:	The number of connections that were aborted because the client terminated without closing the connection properly during the last measurement period	Number	
Failed connections:	The number of connection attempts to the MySQL server that failed during the last measurement period	Number	A high value can indicate: • a configuration problem with the server/clients; • some malicious attack on the server;
Max connections:	The maximum number of connections that were in use simultaneously in use during the last measurement period	Number	

3.2 The MySQL Cache Layer

The tests associated with this layer monitor the health of the query cache and the buffer cache on the MySQL server.

Figure 3.3: The tests associated with the MySQL Cache layer

3.2.1 MySQL Queue Cache Test

This test monitors the health of the query cache in the MySQL server.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Queries registered in cache:	Indicates the number of queries that are	Number	

Measurement	Description	Measurement Unit	Interpretation
	registered with the cache.		
Queries added to cache:	Indicates the number of queries added to the cache during the last measurement period.	Number	
Cache hits:	Indicates the number of times the cache was accessed during the last measurement period.	Number	
Queries deleted from cache:	Indicates the number of queries that were deleted from the cache during the last measurement period.	Number	A high value could indicate low memory allocation for the query cache. Since reading from the cache is less expensive than reading directly from the database, a higher memory allocation to the cache is advisable.
Non- cached queries:	Indicates the number of queries that were not cached (not cacheable due to their QUERY_ CACHE_ TYPE) during the last measurement period.	Number	
Free memory in cache:	Indicates the amount of free memory in the query cache.	MB	A low value of this measure could cause subsequent queries to be rejected by the cache, owing to low memory availability. It would be good practice to tune the cache memory to handle more load.
Free blocks in cache:	Indicates the number of free memory blocks in	Number	

Measurement	Description	Measurement Unit	Interpretation
	the query cache currently.		
Blocks in cache:	Indicates the total number of blocks in the query cache currently.	Number	
Queries in cache:	This is the sum of cache inserts, cache hits, and non- cached queries during the last measurement period.	Number	

3.2.2 MySQL Cache Test

This test reports statistics pertaining to the key buffer cache in the MySql server.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Key reads:	Indicates the number of physical reads of a key block from the disk during the last measurement period.	Number	A high value of this measure indicates that your *key_buffer_size* variable is probably too small.
Key read requests:	Indicates the number of key read requests received by the cache during the last measurement period.	Number	
Not flushed key blocks:	Indicates the key blocks in the key cache that have been modified, but have not been flushed to the disk during the last measurement period.	Number	

3.3 The MySQL Server Layer

The tests associated with this layer indicate the level of activity on the MySQL server by monitoring the threads, queries, and SQL statements executed on it.

Figure 3.4: The tests associated with the MySQL Server layer

3.3.1 MySQL Threads Test

This test tracks the MySQL server threads and reports various performance statistics pertaining to them.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Threads cached:	Indicates the number of threads in the thread cache currently.	Number	
Threads spawned:	Indicates the number of threads newly created to handle new connections during the last measurement period.	Number	Where possible ensure that the number of total threads is not set to be much larger than the maximum number of simultaneous requests expected for the server.
Threads active:	Indicates the number of	Number	A high value for this measure is

Measurement	Description	Measurement Unit	Interpretation
	threads that are currently active.		indicative of a high load on database server.
Slow launch threads:	Indicates the number of threads that have taken more than slow_launch_ time for creation, during the last measurement period.	Number	A high value for this measure could cause excessive delay in processing new connection requests to the database server.
Delated insert threads:	Indicates the number of delayed insert handler threads in use.	Number	

3.3.2 MySQL Resources Test

This test measures the effect of query execution on the disk and memory resources of the MySQL server.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Opened tables:	Indicates the number of tables that were opened during the last measurement period.	Number	If this value is high, then it indicates that the "table_ cache" value is probably too small.
Created tmp disk tables:	Indicates the number of implicit temporary tables that were created on the disk, while executing statements during the last measurement period.	Number	If the value for this measure is high, consider increasing the 'tmp_ table_size' configuration setting for the server.
Created tmp tables:	Indicates the number of implicit temporary tables that were created in the memory created while executing statements during the last measurement period.	Number	

3.3.3 MySql Query Test

This test reports the performance statistics pertaining to the queries executed on the MySQL server.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening

4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Query rate:	Indicates the rate at which queries are sent to the server.	Queries/Sec	This is an indicator of server workload.
Slow queries:	Indicates the number of queries that have taken more than the 'long_ query_ time' for execution, during the last measurement period.	Number	This value should ideally be 0. If it remains consistently high, the administrator should look to identify and optimize the slow queries.
Handler read first:	Indicates the number of times the first entry was read from an index during the last measurement period.	Number	If the value of this measure is high, it suggests that the server is doing a lot of full index scans.
Handler read key:	Indicates the number of requests that were received in the last measurement period, to read a row based on a key.	Number	If the value of this measure is high, it indicates that your queries and tables are properly indexed.
Handler read next:	Indicates the number of requests received in the last measurement period, to read the next row in the key order.	Number	This will be incremented if you are querying an index column with a range constraint. This will also be incremented if you are doing an index scan.

Measurement	Description	Measurement Unit	Interpretation
Handler read prev:	Indicates the number of requests received in the last measurement period, to read the previous row in the key order.	Number	This is mainly used to optimize ORDER BY... DESC.
Handler read rnd:	Indicates the number of requests received in the last measurement period, to read a row based on a fixed position.	Number	This will be high if you are executing a lot of queries that require sorting of the result. If the value of this measure is high, then you probably have a lot of queries that require MySQL to scan whole tables or you have joins that do not use keys properly.
Handler read rnd next:	Indicates the number of requests received in the last measurement period, to read the next row in the datafile.	Number	This will be high if you are performing a lot of table scans. Generally, this suggests that your tables are not properly indexed or that your queries are not written to use the indexes properly.

3.3.4 MySql Activity Test

This test tracks the writes, inserts, deletes, and flushes happening on a MySQL server database.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening

4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Delayed writes:	Indicates the rate at which rows are written with *INSERT DELAYED*.	Writes/Sec	
Delayed errors:	Indicates the rate at errors occurred in the rows written with *INSERT DELAYED*.	Errors/Sec	
Updates:	Indicates the rate at which requests to update a row in a table were received.	Updates/Sec	
Inserts:	Indicates the rate at which requests to insert a row in a table were received.	Inserts/Sec	
Deletions:	Indicates the rate at which requests to delete a row in a table were received.	Deletions/Sec	
Flushes:	Indicates the rate at which FLUSH commands were executed.	Flushes/Sec	

3.4 The MySQL Service Layer

The tests associated with this layer monitor the locking activity, sorting activity, and transactions executing on the MySQL server.

Figure 3.5: The tests associated with the MySQL Service layer

3.4.1 MySQL Long Running Queries Test

This test tracks the currently executing queries on a MySQL server and determines the number of queries that have been running for a long time. You can also use the detailed diagnosis of this test to drill down to the exact queries that have been running for an unreasonably long time, and thus isolate the resource-intensive queries to the database.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.

6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.
7. **ELAPSED TIME** - In the **ELAPSED TIME** text box, specify the duration (in seconds) for which a query should have executed for it to be regarded as a long running query. The default value is 10.
8. **DETAILED DIAGNOSIS** – To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option.

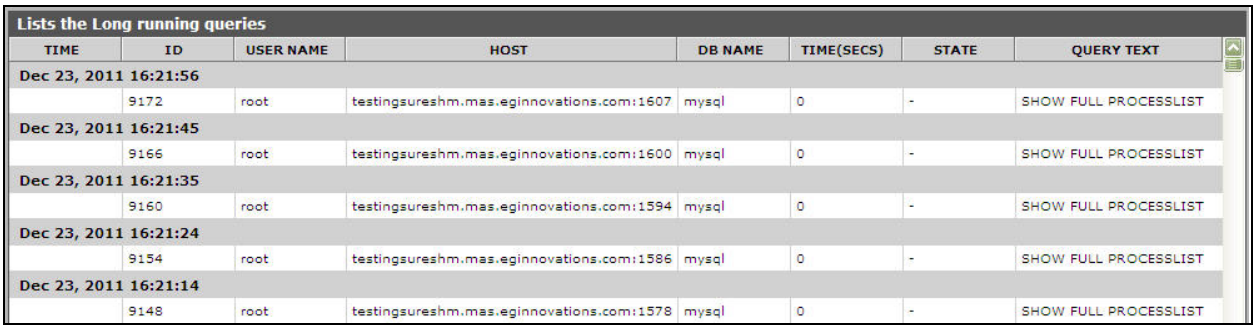
   The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:

   - The eG manager license should allow the detailed diagnosis capability
   - Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Long running queries:	Indicates the number of queries currently executing on the database server that have been running for more time than the configured **ELAPSED TIME**.	Number	The detailed diagnosis for this measure indicates the exact queries and which user is executing the queries. This information can be very useful in identifying queries that may be candidates for optimization.

The detailed diagnosis for the *Long running queries* measure indicates the exact queries and which user is executing the queries. This information can be very useful in identifying queries that may be candidates for optimization.



Lists the Long running queries

TIME	ID	USER NAME	HOST	DB NAME	TIME(SECS)	STATE	QUERY TEXT
Dec 23, 2011 16:21:56							
	9172	root	testingsureshm.mas.eginnovations.com:1607	mysql	0	-	SHOW FULL PROCESSLIST
Dec 23, 2011 16:21:45							
	9166	root	testingsureshm.mas.eginnovations.com:1600	mysql	0	-	SHOW FULL PROCESSLIST
Dec 23, 2011 16:21:35							
	9160	root	testingsureshm.mas.eginnovations.com:1594	mysql	0	-	SHOW FULL PROCESSLIST
Dec 23, 2011 16:21:24							
	9154	root	testingsureshm.mas.eginnovations.com:1586	mysql	0	-	SHOW FULL PROCESSLIST
Dec 23, 2011 16:21:14							
	9148	root	testingsureshm.mas.eginnovations.com:1578	mysql	0	-	SHOW FULL PROCESSLIST

Figure 3.6: The detailed diagnosis of the Long running queries measure

3.4.2 MySQL User Processes Test

This test reports the number and state of the processes of each user who is currently connected to the MySQL server. Using the metrics reported by this test, administrators can promptly isolate idle processes, which are a drain on a server's resources.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for each user currently connected to the MySQL server monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.
7. **IDLE TIME** - Specify the time duration (in seconds) above which the processes that are waiting in the database will be regarded as idle.
8. **DETAILED DIAGNOSIS** – To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be

configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option.

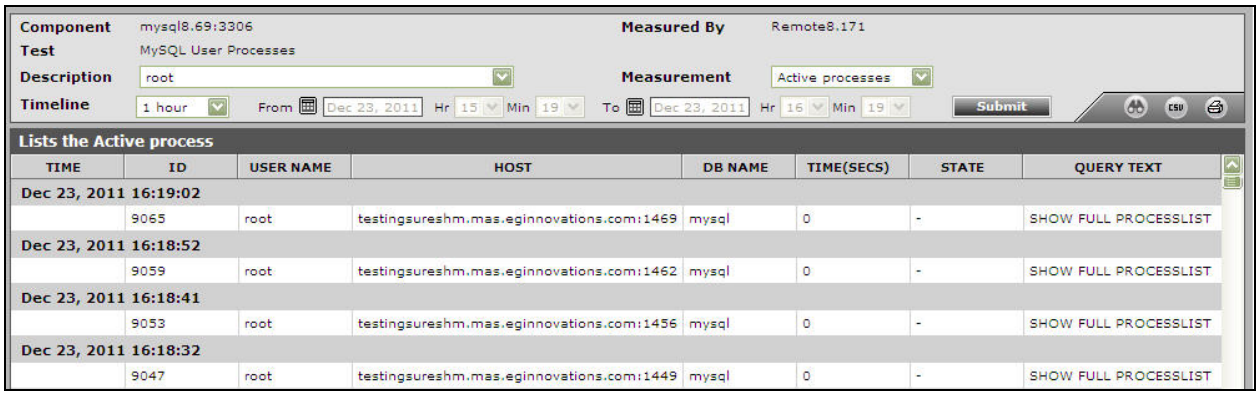
The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:

- The eG manager license should allow the detailed diagnosis capability
- Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Total processes:	Indicates the total number of processes for this user.	Number	The detailed diagnosis of this measure if enabled, lists out all the processes for this user.
Active processes:	Indicates the number of processes of this user that are currently active.	Number	The detailed diagnosis of this measure indicates the exact active processes of this user and the time for which the processes are actually active.
Inactive processes:	Indicates the processes that were inactive for this user in this database.	Number	The detailed diagnosis of this measure if enabled, indicates the inactive processes of this user and the time for which the processes were inactive.
Idle processes:	Indicates the processes that are idle for this user in this database.	Number	The detailed diagnosis of this measure if enabled, indicates the idle processes of this user and the time for which the processes were idle.

The detailed diagnosis of the *Active processes* measure, if enabled, will indicatethe exact active processes of this user and the time for which the processes were actually active.



Component: mysql8.69:3306 | Measured By: Remote8.171
Test: MySQL User Processes
Description: root | Measurement: Active processes
Timeline: 1 hour | From: Dec 23, 2011 Hr 15 Min 19 | To: Dec 23, 2011 Hr 16 Min 19 | Submit

Lists the Active process

TIME	ID	USER NAME	HOST	DB NAME	TIME(SECS)	STATE	QUERY TEXT
Dec 23, 2011 16:19:02							
	9065	root	testingsureshm.mas.eginnovations.com:1469	mysql	0	-	SHOW FULL PROCESSLIST
Dec 23, 2011 16:18:52							
	9059	root	testingsureshm.mas.eginnovations.com:1462	mysql	0	-	SHOW FULL PROCESSLIST
Dec 23, 2011 16:18:41							
	9053	root	testingsureshm.mas.eginnovations.com:1456	mysql	0	-	SHOW FULL PROCESSLIST
Dec 23, 2011 16:18:32							
	9047	root	testingsureshm.mas.eginnovations.com:1449	mysql	0	-	SHOW FULL PROCESSLIST

Figure 3.7: The detailed diagnosis of the Active processes measure

The detailed diagnosis of this measure if enabled, indicates the idle processes of this user and the time for which the processes were idle. Using this information, you can understand how each of the idle connections were made - i.e., using which program - and from where - i.e., from which host.

Lists the Idle process

TIME	ID	USER NAME	HOST	DB NAME	TIME(SECS)	STATE	QUERY TEXT
Dec 23, 2011 15:05:40							
	6415	root	testingsureshm.mas.eginnovations.com:2056	mysql	0	-	-
Dec 23, 2011 14:50:24							
	5865	root	testingsureshm.mas.eginnovations.com:1315	mysql	0	-	-
Dec 23, 2011 12:51:12							
	5334	root	localhost:2974	mysql	241	-	-

Figure 3.8: The detailed diagnosis of the Idle processes meaure

3.4.3 MySQL Database Size Test

This test reports the size of each MySQL database.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for each database on the target MySQL server

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server

5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Database size:	Indicates the current size of this database.	MB	

3.4.4 MySQL Transactions Test

Rollbacks are costly operations on the database. This test monitors the percentage of rollbacks happening for user transactions in a database instance.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
User commits:	Indicates the number of internal COMMIT commands issued during the last measurement period.	Number	
User rollbacks:	Indicates the number of internal ROLLBACK commands issued during the last measurement period.	Number	Ideally, there should be few user rollbacks happening.
Rollbacks:	Indicates the number of internal rollbacks expressed as a percentage of the total transactions with the database.	Percent	Ideally, there should be few user rollbacks happening.

3.4.5 MySQL Sorts Test

The test monitors the sort operation performed on the MySQL database server.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server

5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Range sort rate:	Indicates the rate at which ranges were sorted.	Sorts/Sec	
Row sort rate:	Indicates the rate at which rows were sorted	Sorts/Sec	
Scan sort rate:	Indicates the rate at which sorting was performed by scanning the table	Sorts/Sec	
Merge passed sort rate:	Indicates the rate at which the sort algorithm performed merge passes	Sorts/Sec	If this value is large you should consider increasing the sort_buffer.

3.4.6 MySQL Locks Test

The locking activity of a database server must be monitored carefully because an application holding a specific lock for a long time could cause a number of other transactions relying on the same lock to fail. The **MySQL Locks** test monitors the locking activity on a database server instance.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed

2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1**.
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Lock waits:	Indicates the percentage of lock requests that could not be satisfied immediately and hence, required the caller to wait before being granted the lock.	Percent	A high value of waits can have an adverse impact on application performance. Possible reasons for this behavior could be: • inadequate number of locks available in the database, • unusually high locking behavior of applications accessing the database, • improper database application design, etc.
Table lock waits:	Indicates the number of times in the last measurement period a table lock could not be acquired immediately and a wait was needed.	Number	If the number of waits is high, application performance could suffer. You should first optimize your queries, and then either split your table(s) or use replication.

3.4.7 MySQL Top Tables By Size Test

To make sure that a database does not grow uncontrollably, administrators may want to periodically check the size of the database tables, isolate the tables with the maximum size, and see if the table size can somehow be reduced. The **MySQL Table Size** test helps administrators greatly in this exercise. This test automatically identifies to top 5 tables in a given database in terms of size, and reports the current size of each table.

To enable the test, go to the **ENABLE / DISABLE TESTS** page using the menu sequence : Agents -> Tests -> Enable/Disable, pick *MySQL* as the **Component type**, *Performance* as the **Test type**, choose the test from the **DISABLED TESTS** list, and click on the **<<** button to move the test to the **ENABLED TESTS** list. Finally, click the **Update** button.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for each of the top-5 tables on the database being monitored

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server
5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1** .
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.
7. **TABLE ROWS** - Specify the number of records beyond which this test will consider a table as a large table.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Table size:	Indicates the current size of this table.	MB	Compare the values of this measure across the top-5 tables to know which table is of the maximum size.

3.4.8 MySQL Top Tables By Records Test

In an effort to track the growth of individual databases, administrators may want to time and again check how many rows of data each table in a database contains. In the process, they will be able to identify those tables with more than a permissible number of records. Based on this finding, they can then decide whether deletion of a few or more unnecessary records from these 'large tables' can keep database growth under check. The **MySQL Table Record Count** test helps administrators in this exercise. This test automatically discovers those tables in a given database that contain more than a configured number of records, and reports the number of records fetched from each such table.

To enable the test, go to the **ENABLE / DISABLE TESTS** page using the menu sequence : Agents -> Tests -> Enable/Disable, pick *MySQL* as the **Component type**, *Performance* as the **Test type**, choose the test from the **DISABLED TESTS** list, and click on the **<<** button to move the test to the **ENABLED TESTS** list. Finally, click the **Update** button.

Target of the test : A MySQL server

Agent deploying the test : An internal/remote agent

Outputs of the test : One set of results for each of the tables with a record count that exceeds the **TABLE ROWS** configuration

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** – The IP address of the MySQL server
3. **PORT** – The port on which the server is listening
4. **DB** – the name of a database on the server

5. **USERNAME**and **PASSWORD** - The eG agent has to be configured with the credentials of a user who has server-wide **PROCESS** and **SELECT** privileges on the monitored MySQL server. To know how to create such a user, refer to Section **2.1** .
6. **CONFIRM PASSWORD** – Confirm the **PASSWORD** by retyping it here.
7. **TABLE ROWS** - Specify the number of records beyond which this test will consider a table as a large table.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Total records:	Indicates the number of records fetched from this table.	Number	Compare the values of this measure across the tables to know which table has the maximum number of records.

About eG Innovations

eG Innovations provides intelligent performance management solutions that automate and dramatically accelerate the discovery, diagnosis, and resolution of IT performance issues in on-premises, cloud and hybrid environments. Where traditional monitoring tools often fail to provide insight into the performance drivers of business services and user experience, eG Innovations provides total performance visibility across every layer and every tier of the IT infrastructure that supports the business service chain. From desktops to applications, from servers to network and storage, from virtualization to cloud, eG Innovations helps companies proactively discover, instantly diagnose, and rapidly resolve even the most challenging performance and user experience issues.

eG Innovations is dedicated to helping businesses across the globe transform IT service delivery into a competitive advantage and a center for productivity, growth and profit. Many of the world's largest businesses use eG Enterprise to enhance IT service performance, increase operational efficiency, ensure IT effectiveness and deliver on the ROI promise of transformational IT investments across physical, virtual and cloud environments.

To learn more visit www.eginnovations.com.

Contact Us

For support queries, email support@eginnovations.com.

To contact eG Innovations sales team, email sales@eginnovations.com.