



# InformixHQ Tutorial

IIUG WORLD 2019

SEPTEMBER 2019



# InformixHQ Tutorial

IIUG 2019

## Contents

Introduction .....	3
Part 1: Understanding the Tutorial Environment.....	3
Part 2: Exploring a Database Server in InformixHQ .....	5
Server Dashboard .....	5
Server Configuration.....	7
Server Logs .....	7
Server Performance .....	8
Replication .....	9
Schema Manager.....	11
Server Administration.....	12
Storage.....	13
SQL Tracing.....	17
System Reports.....	18
System Resources.....	18
Part 3: Configuring Monitoring .....	19
How Monitoring Works .....	19
The Benefits of Monitoring.....	20
Setting up a Monitoring Profile .....	21
Viewing Monitored Data.....	24
Part 4: Configuring Alerting .....	27
How Alerting Works.....	27
Setting up an Alerting Profile .....	28
Viewing Alerting Incidents .....	31
Enabling Alerting Notifications.....	33
Part 5: Customizing InformixHQ.....	34
Creating Custom Dashboards.....	34
Creating a Single Server Dashboard.....	35

Creating a Multi-Server Dashboard .....	37
Creating Custom SQL Sensors .....	38
Adding Custom Sensors .....	38
Using Custom Sensors.....	41
Part 6: Users and Permissions.....	43
Adding Users and Configuring Permissions .....	43
Conclusion.....	45
Resources.....	45

## Introduction

InformixHQ is a modern web console for visualizing, monitoring, and managing your Informix server instances. It is purpose built for ease-of-use, scaling out, and optimizing DevOps needs. It provides critical performance management capabilities, monitoring how key performance metrics are changing over time and tracking how efficiently Informix is running your workload even when you've stepped away from your screen. Its monitoring system feeds directly into a customizable alerting system so you can be immediately alerted via email, Twilio, or PagerDuty whenever an issue occurs on one of your Informix database server instances. InformixHQ is designed to be scalable to efficiently manage and monitor as many Informix database server instances as you need. Moreover, it's a tool that can be shared by the DBAs, the app developers, the ops engineers, and management and accessed from any desktop, laptop, or mobile device. InformixHQ is the centralized hub for graphical monitoring, alerting, and administration of your Informix database servers.

This tutorial will offer a deep-dive into the world of InformixHQ. From exploring your database servers in InformixHQ to monitoring and alerting, and all the way to customizing and managing the tool, this tutorial will walk you through the key aspects you will need to know to start using this tool in your organization and to start capturing the full benefit that InformixHQ can bring to monitoring and administrating your database servers.

## Part 1: Understanding the Tutorial Environment

This tutorial is based on the VM image named **Ubuntu18-Docker-HOL** that you can find and run within **Oracle VM VirtualBox** on the tutorial laptops.

The **Ubuntu18-Docker-HOL** image contains a **Docker Compose** project that will be used to setup and run a set of Docker containers that will host the Informix database servers and the InformixHQ server used in this tutorial.

The Docker Compose project used in this tutorial environment is available in the following Informix GitHub repository if you would like to clone this later in your own environment:

<https://github.com/informix/compose-hq-demo>

#### To start the Ubuntu18-Docker-HOL image:

1. Open Oracle VM VirtualBox
2. Select Ubuntu18-Docker-HOL from the list of imported images
3. Click the green start arrow to power-on the image
4. Once the VM image is booted up, you can log on to the image using the credentials

```
User: informix
Password: informix
```

#### To start the Docker Compose project that runs the Informix instances and InformixHQ:

1. Open a terminal window
2. Type the following commands:

```
> sudo bash
> cd lab-darint/compose-hq-demo
> docker-compose up
```

3. It will take a couple minutes to start up all of the Docker containers, initialize all of the Informix instances, and start the InformixHQ server and agents.
4. Once the InformixHQ server is running (you should see a console log message saying the HQServer is started), you can access from the web browser *within the VM image* at the address <http://localhost>. Login to InformixHQ using the following credentials:

```
User: admin
Password: Passw0rd
```

The HQ Demo Docker Compose project contains the following Docker images:

Container name	Contents
server1	Informix instance "server1" + HQ monitoring agent
Server2	Informix instance "server2" + HQ monitoring agent
hqserver	InformixHQ server + an additional Informix instance "hqserver" that serves as the repository database for "server1" and "server2"

Throughout this tutorial, I'll refer to server names *server1*, *server2*, and *hqserver* as these are the names of the Informix database servers that come as part of the Docker Compose HQ Demo (<https://github.com/informix/compose-hq-demo>) and are used in this tutorial environment. If you have your own InformixHQ setup, feel free to follow along this tutorial using your own database servers.

## Part 2: Exploring a Database Server in InformixHQ

Now that you have your Informix Docker containers up and running for both the Informix database servers and InformixHQ, let's explore the InformixHQ UI.

To get the full benefit of this tutorial, make sure you are simultaneously clicking through the described pages of InformixHQ in your web browser as you make your way through this guide! Hands-on exploration of the tool itself is the best way to experience InformixHQ.

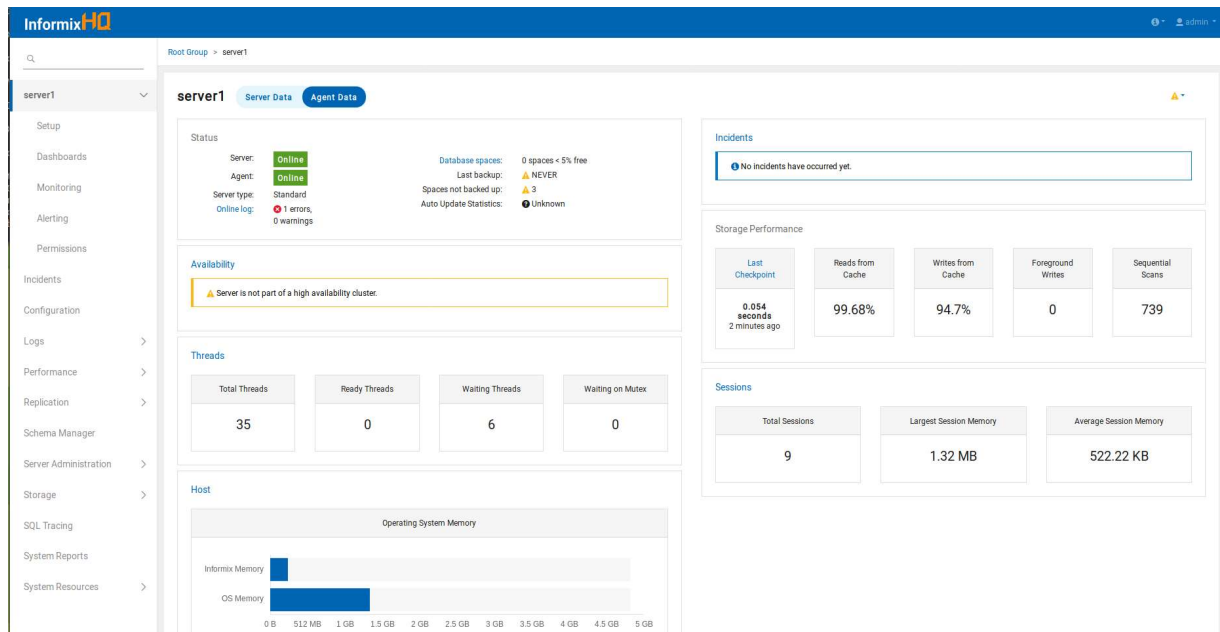
In this section, we will explore all that InformixHQ has to offer when it comes to looking at a particular database server.

### Server Dashboard

We'll start on the server dashboard, which is the first place you land when clicking on a server in InformixHQ.

Let's navigate to the server dashboard for the server named *server1*. Start at the Root Group's dashboard page, which is the first page you get to when logging into InformixHQ. It can also be accessed later on by clicking the InformixHQ logo in the page header. The Root Group is the parent group for all servers and groups in InformixHQ. From this page, you can get a high level view of your servers and groups, including whether any servers have any unread alerts (more on that later). But you can also use this page to navigate to any server defined in InformixHQ.

The server named *server1* is defined directly in the Root Group, so just click on that server to navigate to the *server1*.



Take a moment to take in all of the data presented on the server dashboard. This page is intended to give you a high-level picture of your database server and its current performance.

On the server dashboard, you will find:

- **Status:**
  - Server status: online/offline
  - Agent status: online/offline
  - Server type: Standard, Primary, Secondary (HDR), SDS, RSS
  - Number of recent errors and warnings in the online log

**Activity:** If your database server has an error or warning, click the “Online log” link to see the online log file and find out what issues occurred recently on your database

- Number of spaces that are less than 5% free
  - Time of the last backup
  - Number of spaces that have not been backed up
  - Auto Update Statistics status
- **Incidents:**
  - Any alerting incidents that have occurred on this server
- **High Availability:**
  - The Docker Dev Edition is not configured for HA, but if it was, you would see the list of servers in the HA cluster, the status, type, and lagtime for each server
- **Storage Performance Metrics:**
  - Checkpoints
  - Cache Hit Rate for Reads

- Cache Hit Rate for Writes
- Foreground Writes
- Sequential Scans
- Threads Statistics:
  - Ready, Waiting, and Total Threads
- Host Operating System:
  - Memory
  - CPU
- Session Statistics:
  - Number of connected sessions
  - Max and average session memory
- Server Info:
  - Number of VPs (total and cpu)
  - Version
  - Up time
  - Hostname
  - OS
  - Is IWA configured?

## Server Configuration

We are going to skip over the monitoring, dashboards, alerting, and incidents pages for now (we'll come back to them later in this tutorial!). Let's continue our exploration tour of your database server by going to the **Configuration** link from the side-bar menu.

The **Configuration** page shows you your database server's onconfig settings. All onconfig parameters are shown in the table, including a description of what that parameter does and its currently configured value. For each onconfig parameter that is dynamically editable (i.e. can be changed without restarting the engine), the blue edit (pencil) will be enabled. Clicking on that edit icon will allow you to inline edit the value of the onconfig parameter and save it on the engine.

**Activity:** Take a moment to click through and explore this page. Maybe discover a new onconfig parameter that you never knew existed. Also try your hand at editing an onconfig parameter. Might I suggest enabling `AUTO_CKPTS` or setting `DEF_TABLE_LOCKMODE` to "row".

## Server Logs

InformixHQ provides three pages showing various logs for your databases: the **Online Log** (the primary log file of your database server), the **ON-Bar Activity Log** (the log file for ON-Bar backups, so most likely empty right now for your Docker Compose *server1* database server), and the **SQL Admin API Log**.

Click through each of these log pages in turn. The **SQL Admin API Log** should show the record of the command(s) for editing the onconfig parameter(s) you did on the **Configuration** page. Most of the administration functionality in InformixHQ is done using the **SQL Admin API**, so this page provides a good reference if you are looking for commands used to run the admin actions you are performing in the InformixHQ UI.

## Server Performance

Next, we'll turn our attention to the four pages under the **Performance** section of menu. First up is the **Checkpoints** page. This page allows you to view graphically, as well as in a list format, the most recent checkpoints run on your database server: the checkpoint times, transaction waits, and so forth. From this page, you can also edit any of the onconfig parameter that impact checkpoint frequency (AUTO\_CKPTS, RTO\_SERVER\_RESTART, and CKPTINTVL). Or you can take action and trigger an immediate normal or sync checkpoint.

Use the **Sessions** page to identify and drill down on current session activity on your database server. The **Sessions** page presents you a list of the sessions current connected to your database server. Use the magnifying glass icon to drill down on any session to find the program, current SQL statement, memory usage, open tables, locks, threads, network activity, environment, and profile statistics for that session.

**Activity:** Create a dbaccess session in the terminal that you can drill down on in the InformixHQ UI.

To create a dbaccess session:

1. Open a terminal window on the VM
2. Run the following commands:  

```
> dbash server1  
> dbaccess sysmaster -
```

Then run some SQL queries within dbaccess.

With your dbaccess session still open, refresh the **Sessions** page in InformixHQ, find your session in the list and then drill down to see session details.

Moving on to the **Threads** page, find the list of database threads including thread state and CPU time.

Lastly under **Performance**, you will find the **Virtual Processors** page. Find information on all of your Informix VPs. Drill down on any VP class to get a list of each individual VP. Use the **Add** or **Drop** action buttons to dynamically increase or decrease your database server's VPs.



## Replication

Next up on the InformixHQ menu are the **Replication** pages. We are going to skip these for the purposes of this tutorial since the Docker servers set up in your environment are stand-alone servers. They are not configured for High Availability or Enterprise Replication.

But before moving on, take a look at the following screenshots which demonstrate what you'll find on these pages in InformixHQ.

The screenshot displays the InformixHQ interface for the 'High Availability' section. The left sidebar shows the navigation menu with 'High Availability' selected. The main content area is titled 'High Availability' and includes tabs for 'Cluster Topology', 'Cluster Metrics', 'SMX Info', and 'Configuration'. The 'Cluster Topology' tab is active, showing a diagram of six servers: utm\_serv1 (PRIMARY), utm\_serv4 (HDR), utm\_serv2 (SDS), utm\_serv3 (SDS), utm\_serv5 (RSS), and utm\_serv6 (RSS). To the right of the diagram is a 'Cluster Status Information' box showing 'Active Connection Managers: 0' and 'Failover Arbitration: SDS,HDR,RSS'. Below the diagram is a table with the following data:

Server	Type	Replication Status	Connection Status	Updatable	Workload	Lagtime (seconds)	Approx Log Backlog
utm_serv1	PRIMARY	Active	Connected	✓	13.74 %	0.00000	-
utm_serv4	HDR	Active	Connected	✓	0.03 %	0.06082	0
utm_serv2	SDS	Active	Connected	✓	0.69 %	0.41687	1
utm_serv3	SDS	Active	Connected	✓	1.34 %	0.41678	1
utm_serv5	RSS	Active	Connected	✓	6.31 %	0.00018	0
utm_serv6	RSS	Active	Connected	✓	6.52 %	0.00022	0



For High Availability clusters, you can:

- Visualize your cluster topology
- Find server status, workload, lagtime, and log backlog statistics for each server in the cluster
- Find details on SMX (network) connections
- View and edit HA onconfig parameters
- Monitoring the performance of your HA cluster and visualize this data graphically over time:
  - HA connection status
  - HA CPU workload %
  - HA transaction latency
  - HA lagtime for each server in the cluster
  - HA logical log rate (log records processed per second)
  - HA log backlog

InformixHQ

admin

dev5\_serv1 > Replication > Enterprise Replication

Enterprise Replication

er\_node\_0

Capture

Status: Running

Pages until block: 19046

Send queue

Spooled transactions: 0

Disk

Used: 921 B / 48.83 KB (1.8%)

Network

State: Running

Connected nodes: 1 / 1

Receive queue

Pending transactions: 0

Apply

State: Running

Average latency (seconds): 0

Fail rate (transactions/s): 0

ATS file count: 0

RIS file count: 0

Q Search...

Name	State	Type	Members	Version
er_node_0	Active	Root	server_0	14.10.FC1
er_node_1	Active	Root	server_1	14.10.FC1
er_node_2	Active	Root	server_2	14.10.FC1
er_node_3	Active	Root	server_3	14.10.FC1
er_node_4	Disabled	Root	server_4	14.10.FC1

For servers participating in Enterprise Replication, InformixHQ allows you to:

- Visualize the topology of your ER domain and see high level info about each node
  - Member servers (stand-alone vs. cluster)
  - Node type (root, non-root, leaf)
  - Server version
- Detailed statistics for each ER node
  - Capture, apply, and network status
  - Number of connection nodes
  - Pages until block
  - Spooled and pending transactions
  - Disk usage
  - Average latency
  - Fail rate
  - ATS/RIS file count

## Schema Manager

Next up in on server exploration tour is the **Schema Manager** page. This page allows you to explore your database schema and to run ad-hoc SQL statements.

Select the *sysadmin* database from the drop-down to find the list of table objects in that database. Select a table from the list to view the columns, indexes, and constraints for that table, for example the *command\_history* table.

To run ad-hoc SQL, click on the **SQL** tab. You can run SQL queries and page through the results. You can also run non-queries: INSERTS, UPDATES, DELETES, CREATE or any other SQL statements are supported here.

Note that you also have the option to change the user name used to connect to your selected database by clicking on the **Change User** button. InformixHQ will automatically use the server's defined admin user (or if the admin user is not defined, then the monitoring user) when querying information from your selected database. But if needed, you can click **Change User** and enter a new set of credentials to be used when exploring your user databases and running ad-hoc queries.

**Activity:** Take a moment to switch over the *server2*'s Schema Manager page. The *server2* instance has the *stores\_demo* database. Explore that database's objects and schema and run a few ad-hoc queries.

The screenshot shows the InformixHQ interface for *server2*. The left sidebar contains a navigation menu with options like Setup, Dashboards, Monitoring, Alerting, Permissions, Incidents, Configuration, Logs, Performance, Replication, Schema Manager (selected), Server Administration, Storage, SQL Tracing, System Reports, and System Resources. The main area is titled 'Schema Manager' and shows the 'stores\_demo' database selected. A search bar is present. Below the search bar, a list of tables is shown, with 'catalog' selected. The 'Info' tab for the 'catalog' table is active, displaying metadata: Owner: informix, Last modified: September 23, 2019, First extent: 16 KB, Next extent: 16 KB, Lock level: Row, Page size: 2 KB, Statistics level: Automatic, and Statistics last updated: Sep 23, 2019, 11:11:34 PM. Below this, the 'Columns' table is displayed, showing columns like catalog\_num, stock\_num, manu\_code, cat\_descr, cat\_picture, and cat\_advert with their data types and constraints. The 'Indexes' table is also shown, listing indexes 109\_21 and 109\_22 with their types, columns, levels, leaves, unique keys, clusters, enabled status, filtering, and error status.

Name	Data Type	Constraints	Visible
catalog_num	SERIAL	u109_21: PRIMARY KEY n109_23: NOT NULL	✓
stock_num	SMALLINT	aa: REFERENCES stock(stock_num) n109_24: NOT NULL	✓
manu_code	CHAR(3)	aa: REFERENCES stock(manu_code) n109_25: NOT NULL	✓
cat_descr	TEXT	-	✓
cat_picture	BYTE	-	✓
cat_advert	VARCHAR(255, 65)	-	✓

Name	Type	Columns	Levels	Leaves	Unique keys	Clusters	Enabled	Filtering	With Errors
109_21	Unique	catalog_num	1	1	74	9	✓	✗	✗
109_22	Non-unique	stock_num, manu_code	1	1	41	9	✓	✗	✗

## Server Administration

Now we'll turn to the **Server Administration** section of the menu for *server1*. This is where you will find the **Auto Update Statistics**, **Privileges**, and **Task Scheduler** pages in InformixHQ.

The **Auto Update Statistics** page allows you to view and manage your automatic update statistics policies. Updated optimizer statistics are critical to ensuring optimal query performance. Having the database server automatically run update statistics allows you a hands-free way of ensuring that your queries run efficiently even as your data changes over time.

On the **Overview** tab, view high level AUS statistics. On both a whole server and on a per database level, see how many of your large vs. small tables need statistics updated and how many are missing statistics. Use the **Configuration** tab to view and edit the AUS policies controlling when a table's statistics need updating and also manage the schedule for when the AUS evaluation and refresh tasks get run. Use the **Alerts** and **Commands** tabs to view AUS alerts and the pending and completed update statistics commands respectively.

The **Privileges** page allows you to view and manage the varying level of privileges on your database server. Whether database-level privileges, table-level privileges, SQL Admin API privileges, or internal (mapped) users, you can use Informix to view, add, edit, or revoke privileges on the database server.

The **Task Scheduler** page allows you to manage and customize the sysadmin tasks. The Informix database is equipped with a task scheduler feature, housed in the sysadmin database. You can modify the schedule for when built-in system tasks are run. But you can also create and schedule your own custom tasks as well! InformixHQ makes it easy for you to:

- Create your own tasks
- Edit the schedule of tasks
- Edit task parameters
- Run a task immediately
- View execution statistics for your tasks

## Storage

Storage is a critical aspect of your database server, so InformixHQ comes equipped with five different that pages that drill-down and provide you the ability to easily visualize, analyze, and administer the various storage characteristics of your database.

Our first stop on the **Storage** tour of InformixHQ takes us to the **Spaces** page. This page presents the list of spaces (and chunks) on your database server. View your list of database spaces, including space type, size, and percent used. Use the buttons to create a new space, to modify a space's expansion characteristics, to expand a space, or to drop a space.

server1 > Storage > Spaces

Spaces

Search name or type...

Number	Name	Status	Type	% Used	Size	Page Size	Expandable	Create Size	Extend Size	Last Backup	
1	rootdbs	<span style="color: green;">●</span>	dbspace	<div><div></div></div>	341.8 MB	2 KB	<span style="color: green;">✓</span>	10%	9.77 MB	NONE	<span>✎</span> <span>⛶</span> <span>🗑️</span>
3	twsdbspace	<span style="color: green;">●</span>	sbpace	<div><div></div></div>	20 MB	2 KB	<span style="color: green;">✓</span>	10%	0%	NONE	<span>✎</span> <span>⛶</span> <span>🗑️</span>
2	twsdb	<span style="color: green;">●</span>	dbspace	<div><div></div></div>	200 MB	8 KB	<span style="color: green;">✓</span>	10%	9.77 MB	NONE	<span>✎</span> <span>⛶</span> <span>🗑️</span>

**Activity:** Walk through the process of creating a new space on your database server. Choose “create the space from the storage pool” option to avoid having to pre-create the file on your Docker image.

If you’d prefer to view your spaces by chunks, you can change the “View As” drop-down to **Chunks**. There you will find similar info and actions, just by chunks instead of spaces.

Navigate to the **Pool** link on the menu to view information about your database server’s storage pool. Using a storage pool allows the database server to automatically expand existing spaces when they become full. The *server1* instance has been configured with a storage pool, so you can use this page to view and manage its storage pool.

server1 > Storage > Pool

Storage Pool

Storage Pool Information

Space remaining in fixed entries: 0 B  
Number of extendable entries: 1

Automatic Expansion Configuration

Automatic Expansion: ✓  
Free space threshold: Off  
Wait Time: 30 seconds

Storage Pool Entries

Filter Add Entry

Path	Offset	Space Remaining	Status	Priority	Chunk Size	Last Accessed	Actions
/opt/ibm/data/spaces	0 B	Extendable	<span style="color: green;">●</span> Active	★ High	64 MB		Action

Next on the **Storage** menu is the **Tables and Indexes** page. Use this page to analyze the storage characteristics of the tables and indexes in each of your databases. Select a database from the drop down, then optionally choose if you want to further filter by a particular dbspace. Based on your selection, you will be shown the list of tables and indexes in that database (and dbspace if one is selected). Here you can find the number of rows, number of extents, used size, and whether the object is compressed or not. You will also be shown graphics of the relative space and page usage of each object.

The screenshot displays the 'Tables & Indexes' section of the InformixHQ interface. The left sidebar contains navigation links for various database management tasks. The main content area shows a table of database objects. The 'fragtab1' table is selected, and its partition details are shown below the main table.

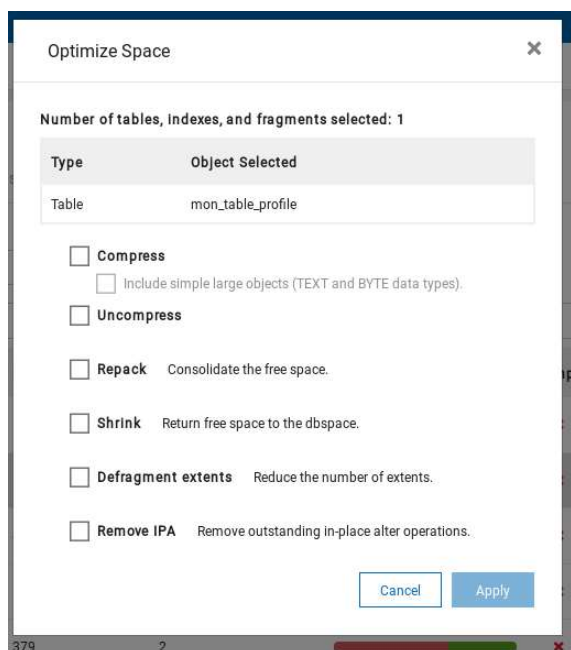
Name	Type	Rows	Extents	Space Usage	Compressed	Used Size	Page Usage																												
fragtab1	Table	10840	3		X	408 KB																													
<table border="1"> <thead> <tr> <th>Partition Number</th> <th>Partition Name</th> <th>Rows</th> <th>Extents</th> <th>Compressed</th> <th>Used Pages</th> <th>Space Usage</th> </tr> </thead> <tbody> <tr> <td>0x00400002</td> <td>datadbs1</td> <td>3613</td> <td>1</td> <td>X</td> <td>17</td> <td></td> </tr> <tr> <td>0x00500002</td> <td>datadbs2</td> <td>3613</td> <td>1</td> <td>X</td> <td>17</td> <td></td> </tr> <tr> <td>0x00600002</td> <td>datadbs3</td> <td>3614</td> <td>1</td> <td>X</td> <td>17</td> <td></td> </tr> </tbody> </table>								Partition Number	Partition Name	Rows	Extents	Compressed	Used Pages	Space Usage	0x00400002	datadbs1	3613	1	X	17		0x00500002	datadbs2	3613	1	X	17		0x00600002	datadbs3	3614	1	X	17	
Partition Number	Partition Name	Rows	Extents	Compressed	Used Pages	Space Usage																													
0x00400002	datadbs1	3613	1	X	17																														
0x00500002	datadbs2	3613	1	X	17																														
0x00600002	datadbs3	3614	1	X	17																														
tab1	Table	10840	1		X	416 KB																													
tab2	Table	6600	1		X	254 KB																													
tab3	Table	6000	1		X	232 KB																													
fragtab1_idx	Index	0	3		X	144 KB																													
tab1_idx	Index	0	1		X	68 KB																													

The space usage graphic shows you how much used (red + pink) vs. free (green) space is allocated to the object. It also shows if the table or index is a good candidate for compression as the pink color represents the estimated space savings compression.

The page usage graphic gives an indication of the portion of pages for each table or index that are full vs. mostly used vs. only partially used vs. unused.

These graphics can help you decide which, if any, storage optimization actions your database server can benefit from. Clicking the zipper icon opens the storage optimization pop-up where you can choose to **compress** or **uncompress**, **repack** (consolidate the free space), **shrink** (return free space to the dbspace), **defragment** (reduce the number of extents), or remove in-place alters.

Besides the zipper icon, there is a refresh icon for each table and index. This refresh will refresh the space usage and page usage calculations. That data calculated by the mon\_page\_usage scheduler task and saved in sysadmin. If your table has recently changed significantly, you may want to click the refresh button to have the database recalculate those values so you can see an accurate reflection of that table or index's storage characteristics.



The **Tables and Indexes** page has a **Server Optimization Policies** tab to aid you in managing your automatic storage optimization policies. The Informix database server has a built-in scheduler task that you can enable and configure to automatically run storage optimization actions when your tables exceed certain configurable thresholds. Use these policies to have the database server automatically take action to keep space usage optimization without the need for DBA intervention. For example, you can configure the database server to automatically compress tables and fragments that exceed a certain number of rows, to automatically repack tables and fragments that exceed a certain percentage of discontinuous space, or to defragment the tables and fragments that exceed a certain number of extents.

The **Tables and Indexes Task Status** tab allows you to view the task status of actions taken on the **Tables and Indexes** page. Since these tasks are run in the background and can take some time on very large tables and indexes, the **Task Status** also you to view the current status of the actions you take and to view the result when the actions complete.

The **Backups** page allows you to see your current backup status and to manage and configure automatic ON-Bar and ontape backups of your database server. If you configure automatic backups, you can also use this page to run an ad-hoc backup at any time just from a single click in the UI.

Last in the **Storage** section is the **Recovery Logs** page. Use this page to view and manage your physical log and logical logs. View the current state of your recovery logs or perform administrative actions to move your physical log, add logical logs, or switch your logical log.

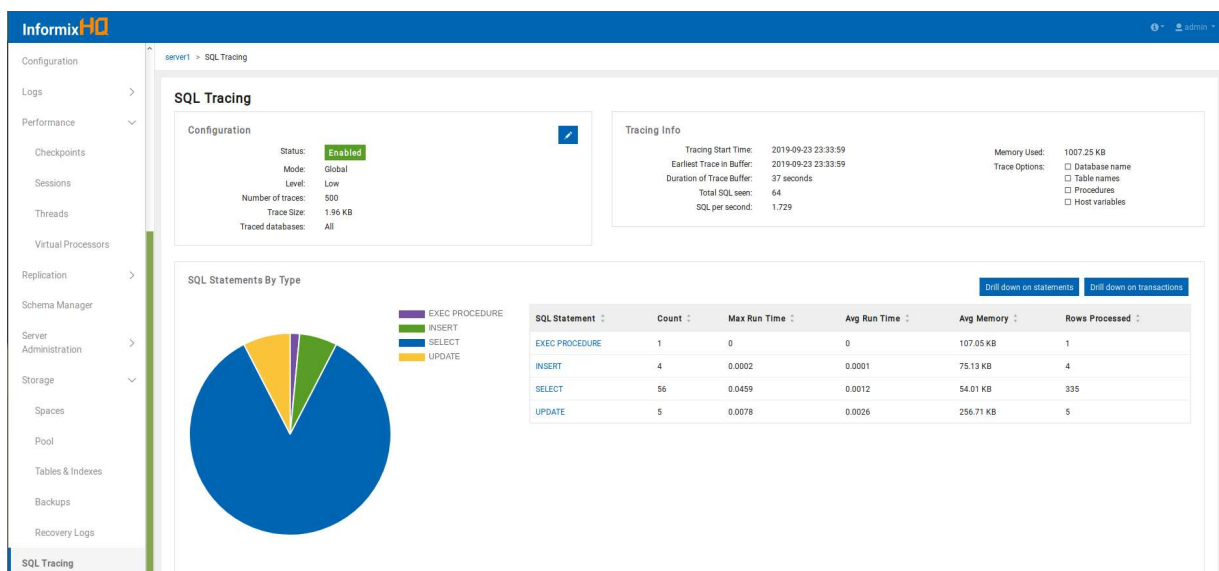


## SQL Tracing

On the **SQL Tracing** page, you can enable and configure tracing of the SQL statements run on the database server. You can trace all statements, or restrict tracing to certain users and/or certain databases.

**Activity:** On the **SQL Tracing** page, enable SQL tracing on your database.

Whenever SQL tracing is enabled on your database, use this page to view traced statements. This page allows various drill downs – on statement type, on transactions, on the SQL statement itself – until you get to an individual execution of a SQL statement. A SQL statement profile allows you to view detailed statistics on that statement execution – including the query plan, the estimated cost, the time the statement executed, the number of rows returned, read, write, and I/O statistics, and memory and disk sorts, among other things.



**Statement Profile**

Statement Info

Session ID:	62
User ID:	200
Statement Type:	SELECT
PQG:	0
Statement Completion Time:	2019-09-23 23:34:20
Response Time:	0.0042061
Database:	<None>

SQL

```
SELECT il_nptotal AS nptotal, 1 AS is_fragment, T AS type, il_npusd AS npusd, il_nrows AS nrow, il_nextms AS nextms, decode (
  symmaster.bitval(il_flags/0x00000000), 1, 'yes', 'no') as compressed, decode ( symmaster.bitval(p_flags/0x00000001), 1, 'yes', 'no') as
  auto_compressed, decode ( symmaster.bitval(il_flags/0x00000000) + symmaster.bitval(p_flags/0x00000001), 0, 'yes', 'no') as
  uncompressed, t2.partition, dspace, tabname, owner from symmaster.systabinfo, symmaster.systabinfo p, select head(decode(p.partition,
  0, partn, partnum)), tabname as tabname, systables.tabid, trim(owner) as owner, tabtype, pagesize, trim(partition) as dspace
  from systables, outer systabinfo where systables.tabid = systabinfo.tabid and fragtype = 'T') as t2 (partnum, tabname, tabid,
  owner, tabtype, pagesize, dspace) where t2.partition = t2.partition and t2.partition = p.partition UNION SELECT il_nptotal AS nptotal,
  1 AS is_fragment, 1 AS type, 1, npusd AS npusd, il_nrows AS nrow, il_nextms AS nextms, decode (
  symmaster.bitval(il_flags/0x00000000), 1, 'yes', 'no') as compressed, decode (
```

Statistics

Reads		Writes		I/O		Executions	
Page	0	Page	0	IO Waits	0	Total	1
Buffer	309	Buffer	14	Wait Time	0	Total Time	0.0042
Cache	100 %	Cache	100 %	Avg Wait	0	Avg	0.0042
				Rows Per Second	25914.48491	Max	0.0042

Optimizer		SQL Info		Locks		Misc	
Estimated Cost	284	SQL Error	0	Requests	3	Disk Sorts	0
Estimated Rows	47	ISAM Error	0	Waits	0	Memory Sorts	1
Actual Rows	109	Isolation Level	1 (Dirty read, read only)	Wait Time	0	Log Space	0 B
Num Iterators	17	SQL Memory	315.06 KB			# of Tables	0

Query Tree

1. Sort	
Cost	1
Estimated rows	1
Actual rows	109
Elapsed Time	0.0167

## System Reports

The **System Reports** page provides a list of detailed reports on various aspects of your database server, from locks and buffer pools, to tables and session activity.

**Activity:** Open a few of the System Reports that are of most interest to you. Do they provide you information that is helpful to understanding the performance of your database server?

## System Resources

If you click on the **System Resources** link in the menu, it will open a dashboard relating to the system resources on the database server's host machine. Most of this dashboard is based on graphs of monitored data collected by the agent, so we'll return to this page later on in the tutorial after we have configured monitoring. But for now, just take note that here you can find information about your host machine – OS name, release, version, hostname, processors, etc.

Last on our tour of a particular database server, we land on the **Memory** page. This page allows you to see database vs. OS memory. It also allows to you manage and configure your database server's Low

Memory Manager configuration which is incredibly useful if you are running your server in a resource restricted environment.

Now that we've toured the entirety of what InformixHQ offers for exploring, visualizing, managing, and administering an instance of Informix, hopefully you have a better understanding of the what InformixHQ offers from a single server perspective. But you may have noted that everything we have seen so far is just a reflection of your database server's status and performance right now, a point in time measure of how things look at the second that you load each UI page. While certainly useful, this is not all that InformixHQ has to offer. One of InformixHQ major values adds is the ability to monitor your database server 24x7, to track its performance over time. So that is what we will cover in the next section. So let's pivot and take a look at monitoring in InformixHQ.

## Part 3: Configuring Monitoring

Now that you've fully explored your *server1* database server, it's time to dive into the monitoring aspects of InformixHQ. InformixHQ was built with monitoring front and center, so enabling monitoring is a critical part of getting the full benefit of what InformixHQ has to offer.

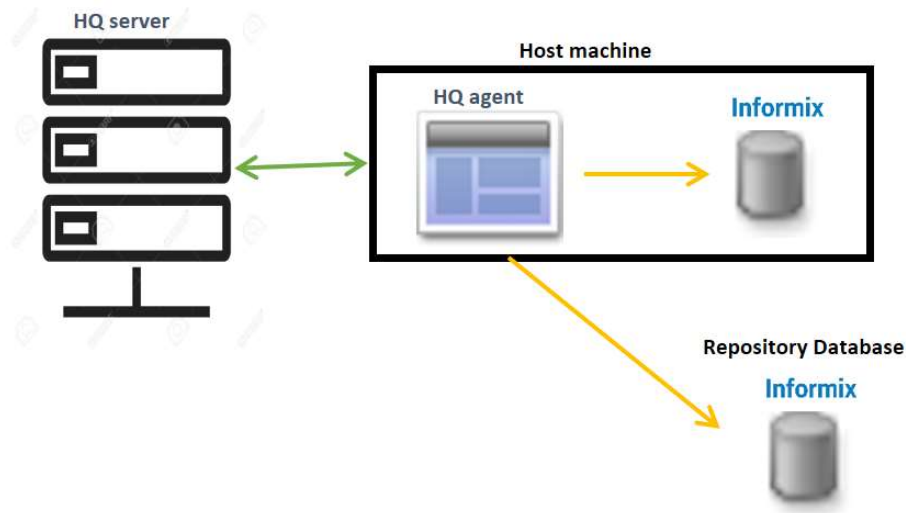
### How Monitoring Works

Up until now, everything in this tutorial has dealt with the **InformixHQ server**. The InformixHQ server hosts the web UI and the REST API it depends on. It has knowledge about all database servers and groups in InformixHQ and creates JDBC connections directly to the database instances to gather live data for the various UI pages. The InformixHQ server also manages the monitoring and alerting profiles of all servers and groups, evaluates the alerting conditions as new monitoring data comes, and dispatches notifications when alerting incidents occur.

The actual monitoring of the database servers, however, is not handled by the InformixHQ server. That is the job of the **InformixHQ agent**. The InformixHQ agent is a separate lightweight java process that is designed to run alongside your Informix database server, monitoring the server's status and performance through JDBC queries as well as monitoring the host operating system through native OS commands.

The agent's sole job is to gather monitoring data and store it in the repository database of your choice. The repository database can be a database in the same instance that is being monitored or it can be a database in a remote centralized Informix repository holding monitoring data for many servers.

## Monitoring in InformixHQ



In the case of the VM image / Docker Compose project used in this tutorial, the Informix database server named *hqserver* is defined to server as the repository database for the other two Informix servers: *server1* and *server2*. The repository data is stored in the *hqmon* database within the *hqserver* instance.

As the user of InformixHQ, you have full control to define what is monitored on your database servers. You can define which sensor metrics the agent should gather and how often. You can also configure how long monitored data for each sensor is stored in the repository database. If you configure a “Data Delete” period after which monitored data should be deleted (the default value is 30 days), the InformixHQ server will periodically check the age of monitored data in the repository database and delete any expired data points.

### The Benefits of Monitoring

The benefits of enabling monitoring in InformixHQ can be summarized as follows:

1. Have the agent monitor your Informix database server 24x7 even when you’ve stepped away from the screen.

2. Have the ability to see the history of performance metrics in the InformixHQ UI, allowing you to identify trends that are happening slowly over time.
3. Have the ability to configure alerts so that InformixHQ notifies you when something is wrong on your database server. Alerts in InformixHQ are solely based on monitoring data collected by the agent. If you don't have the agent monitoring your database server, you cannot configure alerts.

## Setting up a Monitoring Profile

In the section, we will walk through the process of configuring a monitoring profile for your database server.

In your tutorial setup, the agent has already been deployed on *server1* and *server2* with the *hqserver* configured as the repository database for both. Since the agent is already running, you just need to define what data the agent should monitor.

What data the agent should monitor, and how often, is defined in a server or group's monitoring profile. To configure a server or group's monitoring profile, click on the **Monitoring** link from the menu.

For the purposes of this tutorial, let's start with configuring the monitoring profile for the Root Group. Navigate to the Root Group's page, which you can get to by clicking on the InformixHQ logo in the header, and then click on the **Monitoring** menu item.

Since monitoring has never been setup in this environment, you will see a message indicating that no sensors are defined. Click the **Add Sensors** button to add a sensor to the Root Group's monitoring profile.

From the **Add Sensor** pop-up you can select all of the sensors you want to add to the monitoring profile. For the purposes of this tutorial, I suggest adding all sensors except for the High Availability ones (HA log backlog, connection status, lagtime, logical log rate, transaction latency, and workload sensor) as those sensors are only applicable to cluster and all servers in this tutorial environment are stand-alone servers.

A **sensor** defines a set of metrics to monitor.

A **monitoring profile** is the set of sensors defined for a server or group, along with some metadata for each sensor (e.g. run interval and data delete interval).

Add Sensors

Search...

<input type="checkbox"/>	Name ^	Description
<input checked="" type="checkbox"/>	AF files	Monitors for the presence of AF (assert fail) files in the Informix database server's DUMPDIR.
<input checked="" type="checkbox"/>	Backups per dbspace	Monitors the time of the last backups run for each dbspace on the Informix database server
<input checked="" type="checkbox"/>	Buffer and Disk I/O	Monitors buffer and disk I/O activity on the Informix database server
<input checked="" type="checkbox"/>	Checkpoint	Monitors checkpoint statistics for the Informix database server
<input checked="" type="checkbox"/>	Chunk Writes	Monitors chunk writes by the Informix database server
<input checked="" type="checkbox"/>	DBSpace Usage	Monitors the space usage for dbspaces on the Informix database server
<input checked="" type="checkbox"/>	Foreground Writes	Monitors foreground writes by the Informix database server
<input type="checkbox"/>	High Availability Approximate Log Backlog	Monitors the approximate log backlog in seconds for each secondary server in a high availability cluster. This sensor should be run on the primary server of a cluster.
<input type="checkbox"/>	High Availability Connection Status	Monitors the connection status for each secondary server in a high availability cluster. This sensor can be run on any server of a cluster.
<input type="checkbox"/>	High Availability Lagtime	Monitors the lagtime in seconds for each server in a high availability cluster. This sensor should be run on the primary server of a cluster.

Previous
1
2
3
Next

Rows per page: 10

Add Sensors
Cancel

After clicking the **Add Sensors** button, you can use the **Edit** (pencil) icon to edit the **Run Interval** (how often that sensor's metrics are gathered by the agent) and the **Data Delete Interval** (how long the monitored data for that sensor is kept in the repository database). After adding all of the sensors and setting your desired run and data delete intervals, click **Save Changes** to save the monitoring profile.

<input type="checkbox"/>	Name *	Run Interval	Data Retention Interval	
<input type="checkbox"/>	AF files	1 hour	30 days	
<input type="checkbox"/>	Backups per dbspace	1 day	30 days	
<input type="checkbox"/>	Buffer and Disk I/O	15 seconds	30 days	
<input type="checkbox"/>	Checkpoint	1 minute	30 days	
<input type="checkbox"/>	Chunk Writes	15 seconds	30 days	
<input type="checkbox"/>	DBSpace Usage	5 minutes	30 days	
<input type="checkbox"/>	Foreground Writes	15 seconds	30 days	
<input type="checkbox"/>	Informix Server Status	15 seconds	30 days	
<input type="checkbox"/>	LRU Writes	15 seconds	30 days	
<input type="checkbox"/>	Memory	1 minute	30 days	
<input type="checkbox"/>	Memory Semantics	1 minute	30 days	

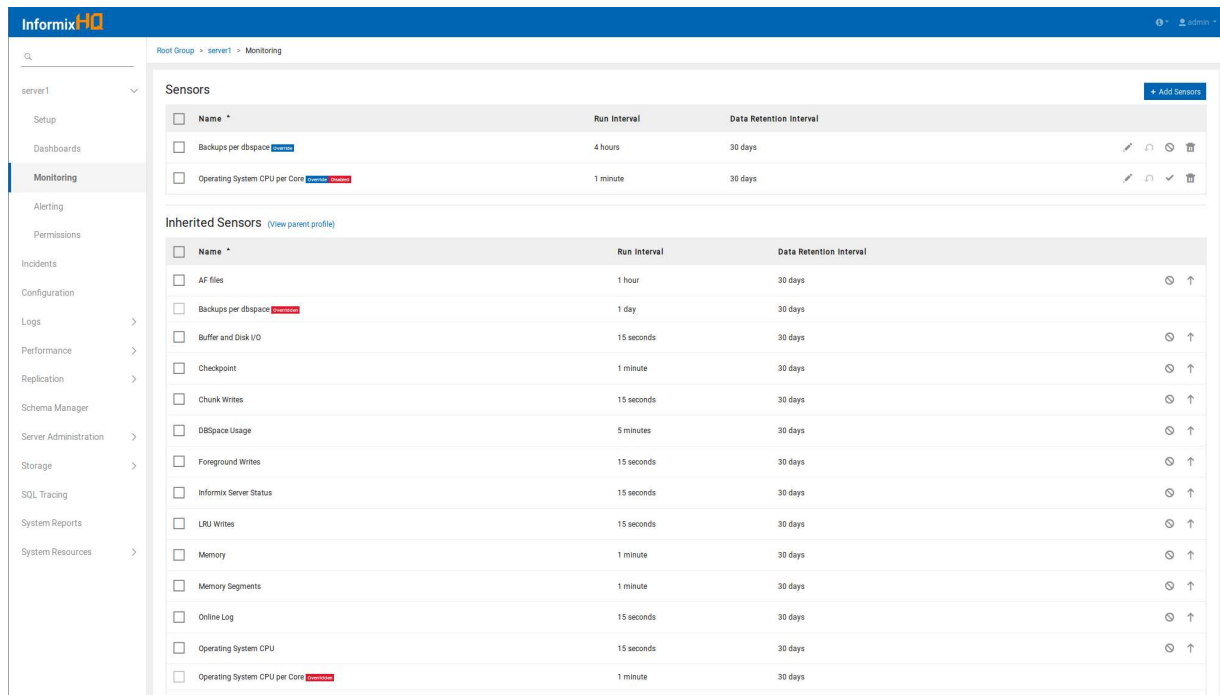
By adding sensors to the monitoring profile of the Root Group, all servers and sub-groups of that group will inherit all of these sensors. That includes the *server1* and *server2* instances that are defined in the Root Group, as well as the *hqserver* instance that is part of the HQ Server sub-group that is defined in this tutorial environment.

The servers and groups in InformixHQ uses a hierarchical model for monitoring and alerting. When you define a group’s monitoring and alerting profile (more on that in Part 4), those sensors and alerts will automatically be inherited by any of the child servers and child groups. This hierarchical group model makes it easy to manage the monitoring and alerting profiles of many Informix server instances.

You can see this hierarchical model in practice by navigating to *server1*’s monitoring profile page. Go back to the Root Group dashboard and click on *server1* and then click on the **Monitoring** link in the menu. This will take you to the monitoring profile for *server1*. Here you will see in the **Inherited Sensors** section all of the sensors that you defined for the Root Group and are thus inherited by *server1*.

If for some reason, you wanted to disable certain sensors on *server1*, to use a different run or data retention interval on *server1*, or to even add additional sensors, you can do it on this page.

For example, in the screenshot below, I’ve overridden and disabled the “Operating System CPU per Core” sensor and I’ve overridden and modified the “Backups per Dbspace” run interval.



If you have agents connected, as we do in this tutorial environment for *server1* and *server2*, every time you save a change to a monitoring profile, the InformixHQ server sends a message, through its websocket connection, to each of the agents affected. This means that as soon as you save a monitoring profile change in the InformixHQ UI, the relevant agents are immediately notified and will start monitoring any new sensors that you have added. In our tutorial setup, by saving our monitoring profile in the Root Group, the *server1* and *server2* agents were immediately notified and thereby started monitoring all of the sensors selected at the specified run intervals.

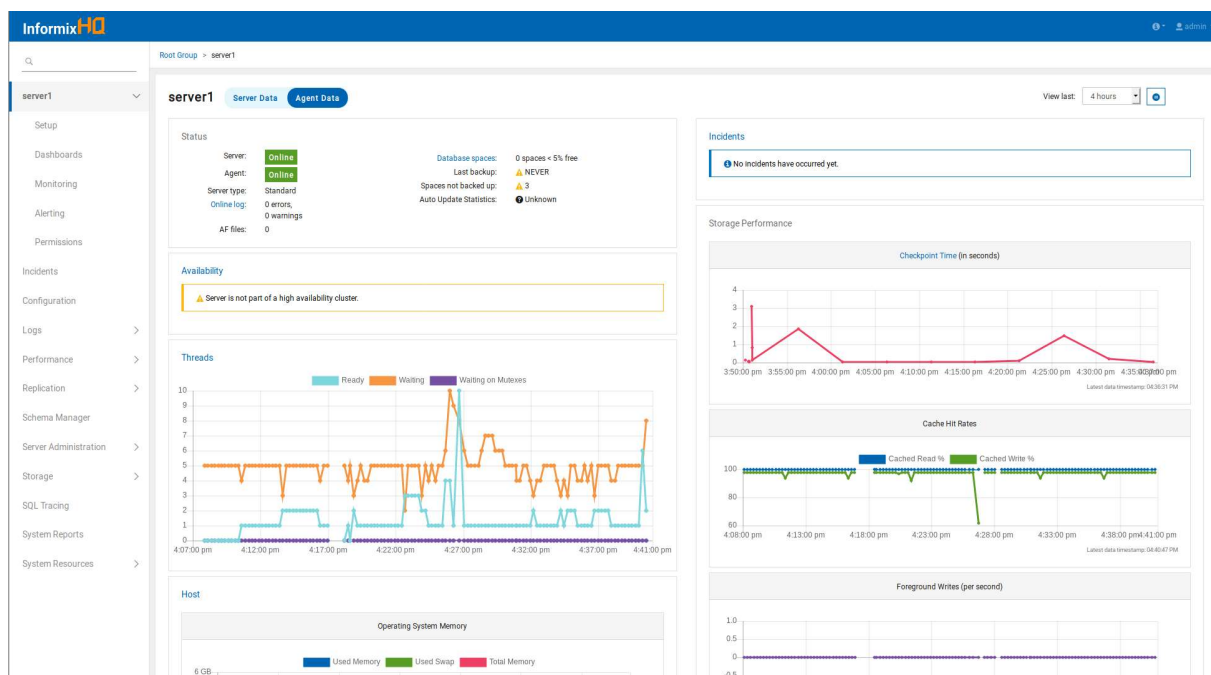
## Viewing Monitored Data

Since we now have monitoring set up in our tutorial environment, let's explore how this data is reflected in the UI. Let's do another mini-tour of the *server1* instance in the InformixHQ UI to see how various pages now show monitoring data as well.

Let's start at the *server1* dashboard, which you can get to by clicking on the *server1* card from the Root Group dashboard.

Note how the server dashboard now has a set of line charts showing the various dashboard metrics (threads, checkpoint times, cache hit rate, OS memory, etc.) over time instead of the single point in time metrics we saw in Part 1.



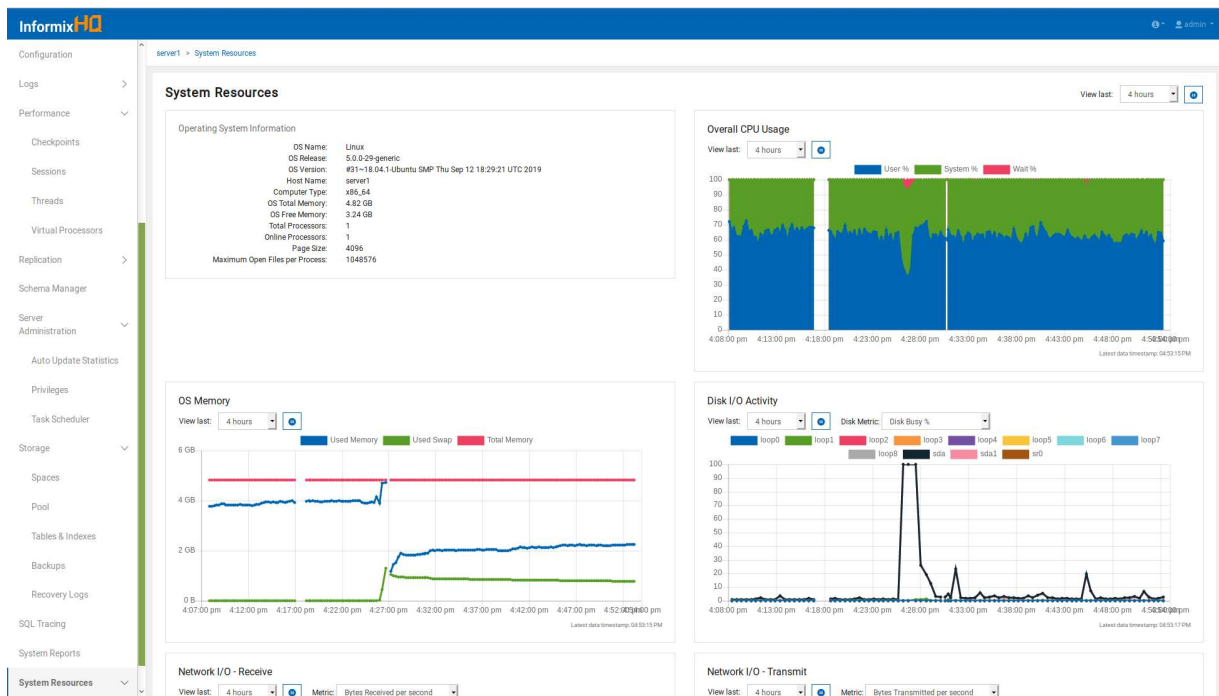
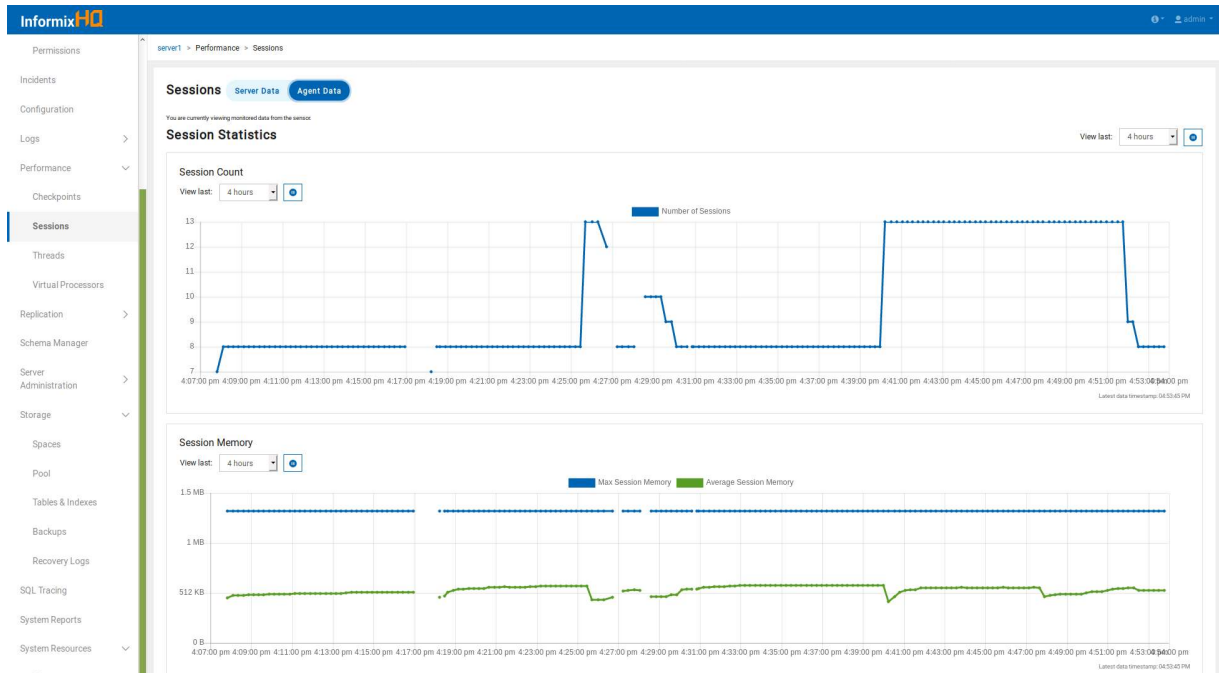


In fact many InformixHQ UI pages become even more useful as you start using the agent because it will automatically show you the history of various performance metrics and allows you to identify any trends or concerning outliers!

Since we just enabled monitoring, you won't see much history, but as you continue to go through this tutorial you'll see the history slowly start to build and you can imagine how useful this becomes once you have the agent collecting data for a week, a month, or even longer.

InformixHQ dashboards will show the last 4 hours of data by default, but you can use the "view last" drop-down at the top of any dashboard or sensor metric chart to view data for longer intervals.

Besides the server dashboard, check out the **Performance > Sessions** page (the **Agent Data** tab) and the **System Resources** dashboard page to see other examples of how sensor data is integrated into various server UI pages.



**Activity:** Do you want to explore the repository database's schema and the data coming in from the sensors?

Navigate from the Root Group's dashboard to the HQ Server Group's dashboard to the *hqserver* instance. This is our repository database. Navigate to *hqserver*'s **Schema Manager** page to explore the schema and query the actual data coming back from the agent.

Hint: any table name prefixed with "s\_1\_" holds sensor data for *server1* and any table name prefixed with "s\_2\_" holds sensor data for *server2*.

## Part 4: Configuring Alerting

Monitoring flows right into alerting in InformixHQ so that's what we'll cover in this section.

It is important to note that all InformixHQ alerts are based on monitoring data collected by the agent. If you do not have the agent connected to your database server, you cannot get InformixHQ alerts on that database server.

### How Alerting Works

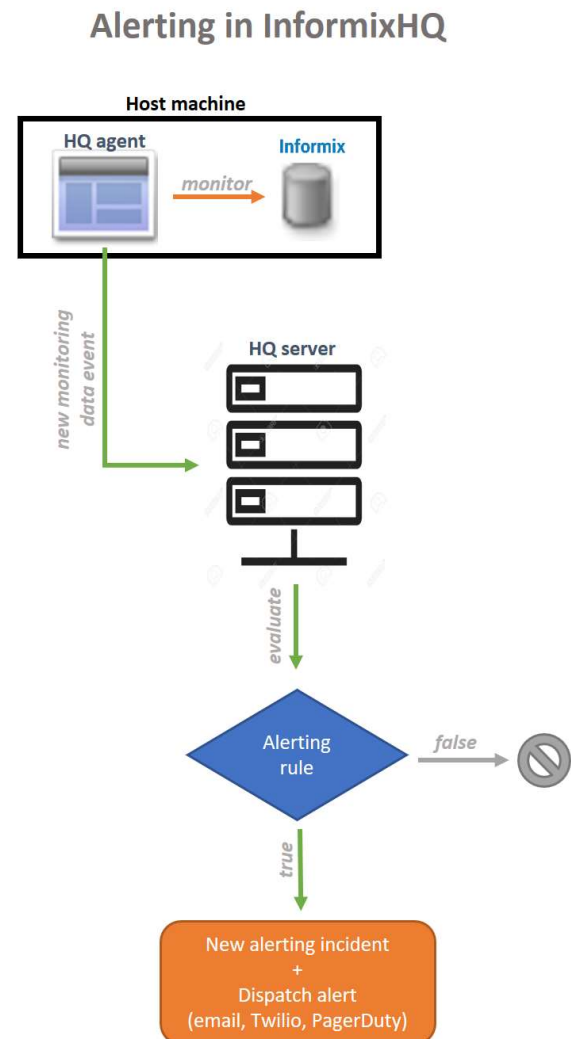
For each server or group in InformixHQ, you can define an alerting profile. Similar to a monitoring profile, the alerting profile defines the set of alerting conditions for that server or group. Alerts are inherited, meaning alerts that are defined in parent groups are automatically inherited by child servers or child groups unless they are explicitly disabled.

Alerts in InformixHQ are defined based on monitoring data. An alerting condition can be defined for any sensor metric that is gathered by the agent.

The InformixHQ server is responsible for managing the alerting profile and for evaluating alerting conditions as new monitoring data comes in.

The general flow for how alerts are evaluated is as follows:

1. When an alerting condition is defined, the InformixHQ server subscribes to that sensor's data from the agent.
2. The agent periodically gathers monitoring data for that sensor from the Informix database server, as defined by the monitoring profile.
3. For sensors metrics that the InformixHQ server is subscribed to, the agent sends a web socket message to the InformixHQ server every time a new monitoring data point is gathered.
4. The InformixHQ server evaluates the new monitoring data point against the defined alerting condition rule.
5. If the alerting condition evaluates to true for the new data point, the InformixHQ server creates an alerting incident.
6. For any InformixHQ users that are subscribed to alerting notifications, InformixHQ server will dispatch an alerting incident message to their configured alerting service (email, Twilio, PagerDuty).



## Setting up an Alerting Profile

Now let's set up an alerting profile in InformixHQ to see this in practice.

To configure a server or group's alerting profile, click on the **Monitoring** link from the menu for that server or group.

For the purposes of this tutorial, since we defined our monitoring profile for the Root Group, we'll also define our alerting profile on the Root Group. The servers *server1* and *server2* will automatically inherit the alerting conditions we define in the Root Group's alerting profile. Navigate to the Root Group's page, which you can get to by clicking on the InformixHQ logo in the header, and then click on the **Alerting** menu item.

Similar to a monitoring profile, an **alerting profile** is the set of alerting conditions defined for a server or group.

An **alerting incident** is generated each time monitoring data is gathered that violates an alerting condition.

When you get to the **Alerting** page, you will see a message that no alerts are defined. Click on the **Add Alert** button to add your first alerting condition.

There are three types of data that you can alert on (the **Alert me when** field in the UI). Most alerts you define will be based on the latest value of a sensor metric. But we have two additional special cases: alerting on the current status of your Informix database server (online, offline, etc.) or on the agent status.

Start by defining an alerting condition for your database server status. Provide an alert name, select to be alerted on the "Current status of Informix" and then select one or more server status to be alerted on. For example, select to be alerted when the database server goes offline. (A note about the "Unknown" state: the database server status will be set to "Unknown" whenever the agent gets disconnected from the InformixHQ server. When the agent goes offline, InformixHQ is no longer monitoring your database server, so InformixHQ changes its status to "Unknown").

The screenshot shows the InformixHQ web interface. The top navigation bar is blue with the InformixHQ logo and a user profile icon labeled 'admin'. A left sidebar contains links for Dashboards, Monitoring, Alerting (which is highlighted), Permissions, and Incidents. The main content area is titled 'Root Group > Alerting' and 'Alerts'. It features a '+ Add Alert' button in the top right. Below this is a 'Create a new alert' section with a 'Name' input field containing 'server status alert'. The 'Alert me when' section has a dropdown menu set to 'Current status of Informix' and a sub-dropdown set to 'is'. Below these are two rows of buttons representing server states: Offline (highlighted in green), Online, Initialization, Quiescent, Recovery, Backup, Shutdown, Abort, Single User, and Unknown. At the bottom of the form are '+ Add' and 'x Cancel' buttons.

If you are relying on InformixHQ to monitor your database server, it is also wise to define an alerting condition on the agent status so that you are notified any time the agent goes offline. If you select "is







one of” as your alerting condition and select both “Offline” and “Online”, you will get an alerting incident whenever the agent comes online or goes offline.

The screenshot shows the InformixHQ interface with the 'Alerting' section selected in the left sidebar. The main content area is titled 'Alerts' and contains a form to 'Create a new alert'. The form has a 'Name' field with the value 'agent status'. The 'Alert me when' section has a dropdown menu set to 'Agent status' and a sub-dropdown set to 'Is one of'. Below this, there are two green buttons labeled 'Offline' and 'Online'. At the bottom of the form are '+ Add' and 'Cancel' buttons.

Now let's define an alert based on a sensor metric. Choose the sensor name from the drop-down, and then since sensors can collect one or more metrics, choose the particular sensor metric that you want to be alerted on. Then define your condition (is less than, is greater than, etc.) and provide your threshold. In the screenshot below, I've defined an alerting condition so that I'm alerted when dbspace usage percent free values below 5% of any of my dbspaces.

The screenshot shows the InformixHQ interface with the 'Alerts' section selected in the left sidebar. The main content area is titled 'Alerts' and contains a form to 'Create a new alert'. The form has a 'Name' field with the value 'dbspace usage alert'. The 'Alert me when' section has a dropdown menu set to 'Latest value of a metric', a sub-dropdown set to 'DBSpace Usage', and another sub-dropdown set to 'Percent free'. Below this, there is a dropdown menu set to 'is less than' and a text input field with the value '5' and a '%' symbol. At the bottom of the form are '+ Add' and 'Cancel' buttons.

Below the form is a table listing existing alerts:

Name	Run...	
server status alert <span>new</span>	on change	  
agent status <span>new</span>	on change	  

At the bottom of the page are 'Save Changes' and 'Discard Changes' buttons.

## Viewing Alerting Incidents

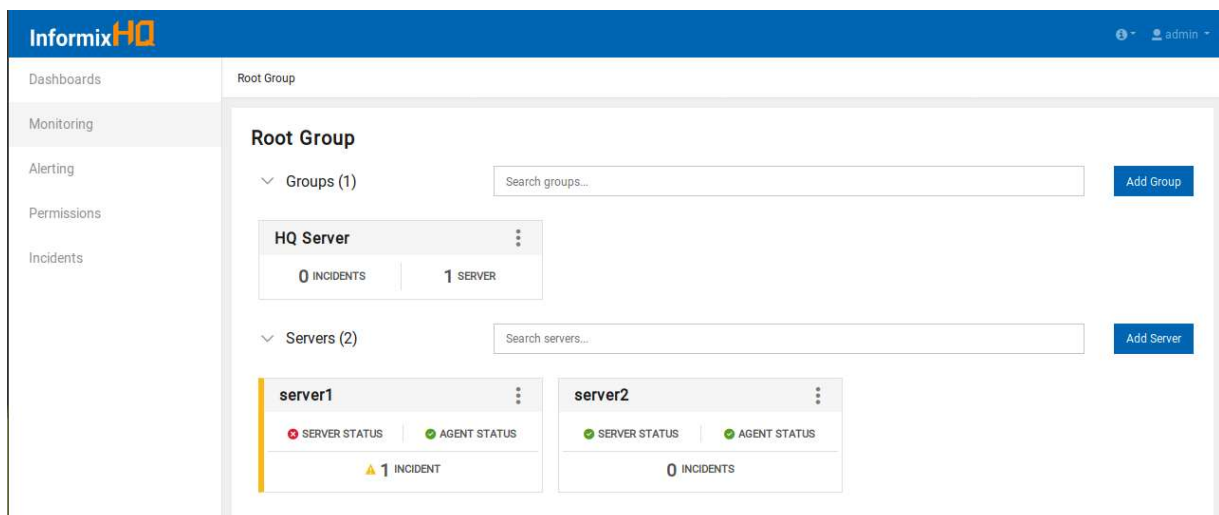
When viewing any server or group in the UI, there will be an **Incidents** page on the menu. This page will show you any incidents that have occurred on the database server. Unacknowledged incidents will also show up on the group dashboard or server dashboard respectively.

To see an alerting incident pop-up, we need to trigger an alerting condition. Follow the following steps to bounce the *server1* database server instance. This will allow you to validate that InformixHQ quickly and accurately detects if your database server goes down.

1. Ensure that you have clicked “Save Changes” to save your alerting profile.
2. Click the InformixHQ logo in the header to navigate to the Root Group dashboard. You can use this page to watch for alerting incidents to pop-up as soon as InformixHQ detects the server goes offline.
3. Now open a terminal window in the VM and run the following commands to enter into the *server1* docker container and bounce the Informix instance:

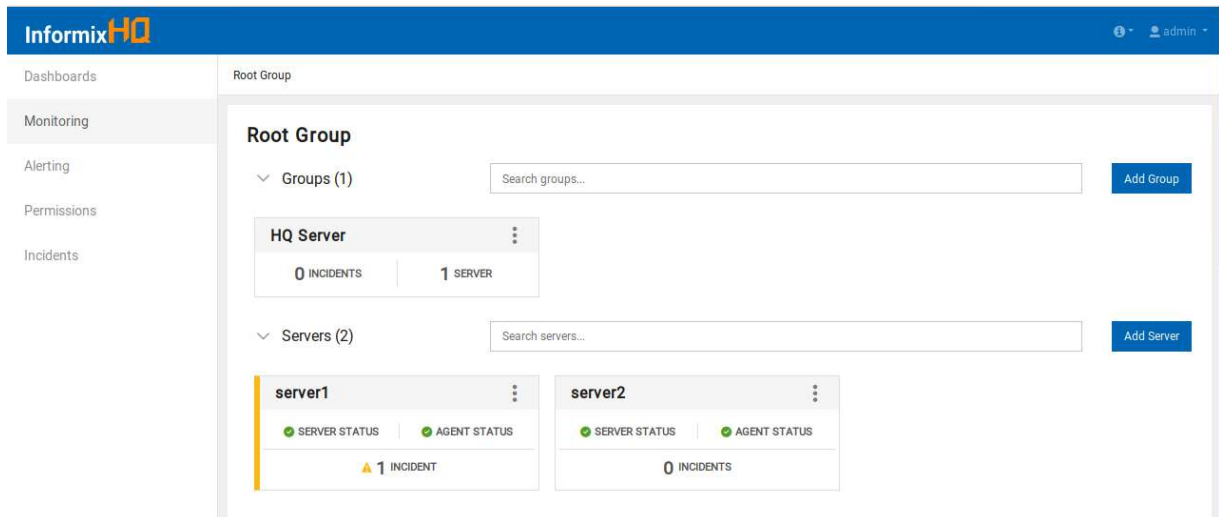
```
> sudo bash
> docker exec -it server1 bash
> onmode -ky
> oninit -vy
```

If you are watching the Root Group dashboard, as you do this, you should see the *server1* status go offline (red ‘x’ icon) when the database server goes offline and you should see 1 new incident pop-up.

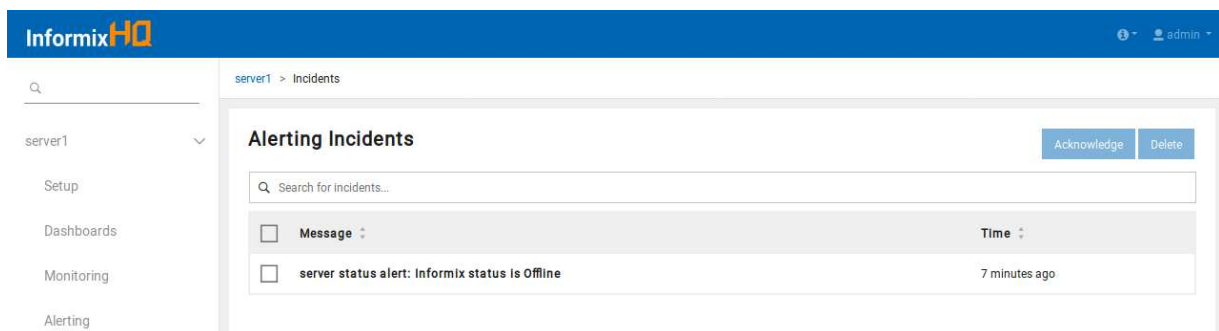


After running `oninit -vy`, the server status should go back to green, but the 1 incident will remain.

Note: the number of incidents you will see may vary depending on how you defined your server status alert. If you defined to be alerted when the server status is offline (as shown in the screenshot in this tutorial), you will get one alerting incident. If you configured that alerting condition to be on more than one status, you may see additional alerting incidents as the database server comes back online.



You can view the actual incident by clicking on the **Incidents** link on the menu which will show you all alerting incidents for the Root Group, or you can navigate to *server1* and see the incidents listed on the server dashboard or use the server side-bar menu to navigate to the **Incidents** page specific to that database server.



From the **Incidents** page, you can acknowledge or delete alerting incidents. Acknowledging incidents makes it so they no longer show up on the group dashboard, but they will continue to be available on the **Incidents** page. Deleting incidents will permanently delete them from InformixHQ.



## Enabling Alerting Notifications

To get the full benefit of alerts, you will likely want to enable alerting notifications so that you do not have to keep checking the InformixHQ UI for alerts. Instead you can have InformixHQ notify you when an alerting incident occurs.

InformixHQ supports four alerting notification mechanisms:

1. Email
2. Twilio
3. PagerDuty
4. Custom script

It is beyond the scope of this tutorial to set up and demo these alerting notification integrations, but we'll briefly touch on how you would go about setting up each one.

*To receive **email** alerting notifications:*

1. The "admin" user of InformixHQ needs to go to the **System Settings > Alerting Configuration** page and enable email notifications.
2. Enter the host, port, user, and password for your SMTP server, change the state to "Enabled", and click save.
3. The "admin" user can optionally send a test email by entering an email address and clicking the **Test** button.
4. After email notifications are enabled at the system level, each user in InformixHQ who wants to receive email alerts will have to login and enable it for their user. Each user should
  - a. Go to **My Settings > Alerting Configuration**
  - b. Click to enable alerting for their user
  - c. Select email alerts and provide their email address

*To receive **Twilio** alerting notifications:*

1. The "admin" user of InformixHQ needs to go to the **System Settings > Alerting Configuration** page and enable Twilio notifications.
2. Enter the account SID, auth token, and from phone number for your Twilio account, change the state to "Enabled", and click save.
3. The "admin" user can optionally send a test notification by entering a phone number and clicking the **Test** button.
4. After Twilio notifications are enabled at the system level, each user in InformixHQ who wants to receive Twilio alerts must login and enable it for their user. Each user should
  - a. Go to **My Settings > Alerting Configuration**
  - b. Click to enable alerting for their user
  - c. Select Twilio alerts and provide their phone number

*To receive **PagerDuty** alerting notifications:*

1. The "admin" user of InformixHQ needs to go to the **System Settings > Alerting Configuration** page and enable PagerDuty notifications.

2. Enter the PagerDuty service key, change the state to “Enabled”, and click save.
3. The “admin” user can optionally send a test notification by clicking the **Test** button.

PagerDuty notifications are enabled globally in InformixHQ. Users do not have to enable PagerDuty notifications individually. Once the “admin” user enables PagerDuty, a PagerDuty notification will be sent whenever an alerting incident occurs on any server or group in InformixHQ.

*To enable **Custom Script** alerting notifications:*

Custom script notifications in InformixHQ provide an extensible way to integration InformixHQ alerts with any alerting mechanism used by your organization. The InformixHQ server will set environment variables with information about the alerting incident and then run your custom script (or any OS command) on the host machine on which the InformixHQ server resides. Your custom program or script can process the InformixHQ alerting incident in any way required by your organization.

1. The “admin” user of InformixHQ needs to go to the **System Settings > Alerting Configuration** page and enable “Run Script” notifications.
2. Enter the script location (or it can actually be any OS command), change the state to “Enabled”, and click save.
3. The “admin” user can optionally send a test notification through your custom script by clicking the **Test** button.

Custom script notifications are enabled globally in InformixHQ. Users do not have to enable custom script notifications individually. Once the “admin” user enables custom script notification, that script or OS command will be executed for any alerting incident that occurs in InformixHQ.

## Part 5: Customizing InformixHQ

This section of the tutorial covers two ways of customizing InformixHQ:

- Customizing the UI by creating custom dashboards
- Customizing what is monitored by the agent by creating custom sensors

### Creating Custom Dashboards

Custom dashboards allow you to define UI pages that show you the Informix monitoring data that is most important to you. You can create single or multi-server dashboards. On each dashboard, you can define a set of line graph panels that show monitored data from the repository database. You can drag, drop, resize, select colors, and edit graph label to customize the look and feel of your dashboard.

Once a dashboard is created, you can dynamically change the server or set of servers shown on your dashboard depending on the server(s) that you have interest in.

We'll now walk through the process of creating a custom dashboard in the UI. Dashboards are always assigned to the current group, but you can access them from either a group or server's sidebar menu.

Clicking on **Dashboards** on the menu takes you to the page where you can define or open your custom dashboards. Go to the Root Group's **Dashboards** page now and click **New Dashboard**.

When you enter the **Dashboards** page from the perspective of a group, the first thing you'll have to do is to select a server or set of servers for the dashboard. This server or set of servers will be used to preview the dashboard as you build it.

### Creating a Single Server Dashboard

Let's start by building a single server dashboard, so select *server1* in the **Select Servers** pop-up. Then click the **Add Panel** or plus icon to add your first dashboard panel.

Currently there is only one type of dashboard panel: a sensor graph. You select one or more sensor metrics to graph on your dashboard panel.

Let's add an operating system CPU panel to our new dashboard:

To add dashboard a panel:

1. Provide a panel title
2. On the **Data** tab, add the data sources (sensor metrics) for you graph
3. On the **Series** tab, optionally edit the color, label, and axis for each data series
4. On the **Charts** tab, optionally configure the units, min/max, and labels for your left and right axis

1. Edit the panel name: OS CPU
2. Click **Add New Data Source** and select the Operating System CPU sensor. For sensors that have more than one metric, you can choose to graph all metrics in a single graph or select a particular metric. For this example, let's just graph all OS CPU metrics in a single graph.
3. Click over to the **Series** tab. On this tab, you can edit the series labels, series colors, or axis.
4. Click over to the **Charts** tab and review the axes configuration. InformixHQ will attempt to auto-configure your axes based on the sensor metrics you chose, so this is case, our Operating System CPU metric axis is properly configured to use the left axis for all metrics with a percentage unit and min/max of 0 and 100.
5. When you are done editing the panel, click the "<" button to return to the dashboard. Note that panels are auto-saved to your dashboard so you are not having to click save for every change you make to a dashboard.
6. Once you are back on your dashboard, you can resize the panel as desired.

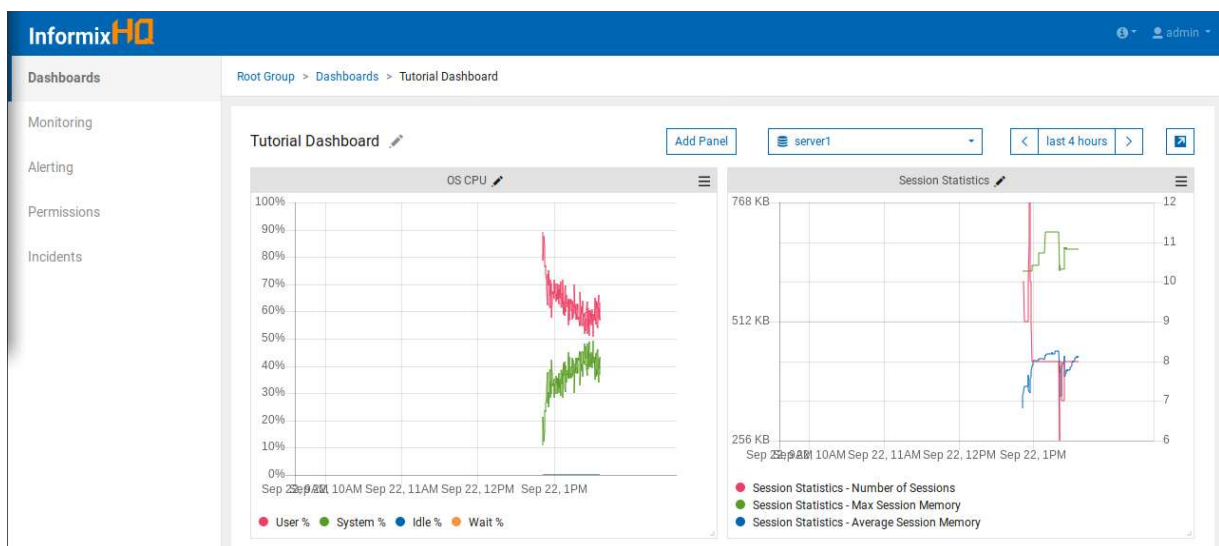
Let's add a second dashboard panel for database session statistics:

1. Click **Add Panel** and then select the location of the new panel by clicking on the plus icon in your desired location for the new panel.
2. Edit the panel name: Session Statistics
3. Click to add a **New Data Source** and select the Session Statistics sensor. Again, we'll graph "All Metrics" from that session which will graph the number of sessions, max session memory, and average session memory.

4. Click over to the **Series** tab and make any desired changes to series labels and colors. Note here that InformixHQ has automatically configured the two memory metrics to use the left axis and the number of sessions metric to use the right axis.
5. Click over to the **Charts** tab and review the axes configuration, which again is auto-configured for you, just with two axes this time.
6. When you are done editing the panel, click the “<” button to return to the dashboard.
7. Once you have more than one panel, you can now drag and drop in addition to resize in order to arrange your dashboard panels as you like.

Besides panel, there are a couple of other fields to configure for your dashboard:

1. Add a dashboard title, for example “Tutorial Dashboard”
2. Optionally set a default server for this dashboard. If you don’t set a default dashboard, when you open this dashboard from the group perspective, you’ll be first prompted to select a server. If you open it from the menu of a server, it will automatically open with that server selected regardless of the default server.



If you click back to the **Dashboards** page, you’ll now see your “Tutorial Dashboard” on the list. You can open the dashboard at any time by clicking on it from this list, selecting a server if you haven’t defined a default. When viewing a dashboard, the default time slice will be four hours. You can use the time controls at the top to change the time slice or move backwards in time to see older data. You can also use your mouse to select any portion of the graph to zoom in. The dashboard will keep all graphs in sync, so when you zoom in on one dashboard panel, it will zoom into the same time slice on all dashboard panels.

## Creating a Multi-Server Dashboard

Now let's walk through the process of creating a multi-server dashboard. It's very similar to the single server dashboard scenario except that you now have an additional server drop-down when defining your panel data sources. For each panel, you can decide if you want to graph data from a particular server or from multiple servers in the same graph.

Navigate again to the Root Group's **Dashboards** page and click the **New Dashboard** button. Select both *server1* and *server2* this time.

Now let's add our first panel. This panel will graph *server1* and *server2* monitoring data on the same graph.

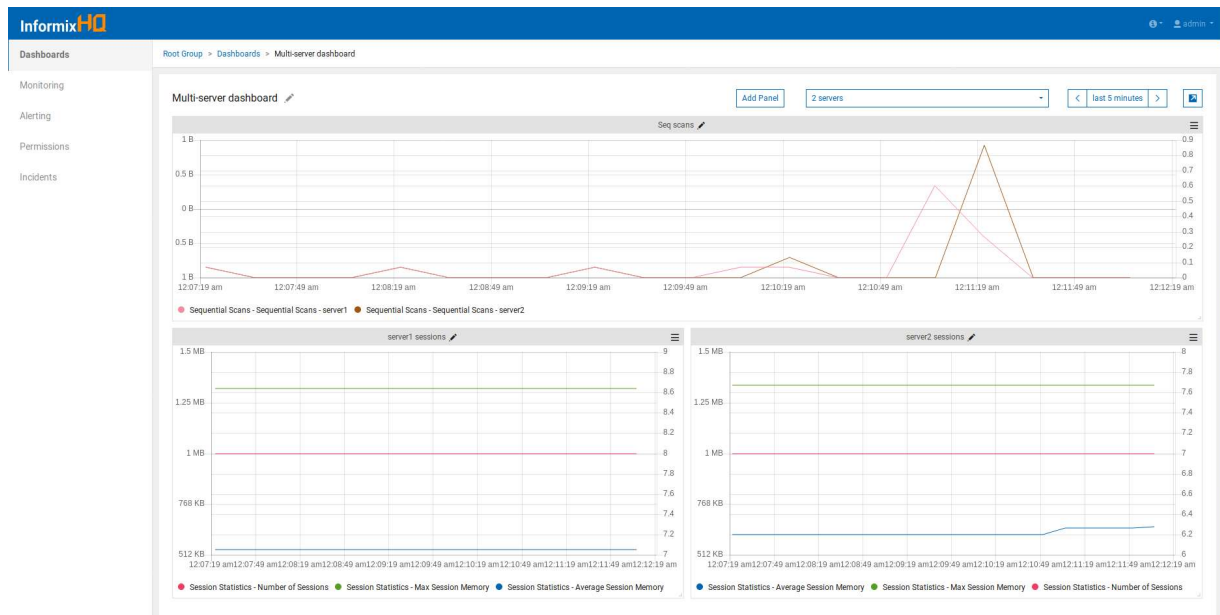
1. Click **Add Panel**
2. In the **Data** tab, notice that there is now both a Servers and a Data Sources section. Keep the server's as "All Servers" and click to add a **New Data Source** and let's select the Sequential Scans sensor this time.
3. Give the panel a name and then click "<" to return the dashboard.

Besides graph multiple servers on the same graph, you can also tie particular panels to a single server and then create two side-by-side graphs, each with a different server. If you are choosing a sensor with multiple metrics that you want in a single graph, sometimes side-by-side graphs may be a better route to comparing two different servers than overlying the two servers on the same graph.

For our side by side graphs, define the following panel.

1. Click **Add Panel**
2. In the **Data** tab, in the Servers drop-down, select *server1*.
3. Then add a **New Data Source**. Let's use the Session Statistics, all metrics, again.
4. Give the panel a name and then click "<" to return the dashboard.

Now define the same panel again for *server2* and then drag and drop so they are side by side.



**Activity:** Now add your own panels to these custom dashboards. Or create entirely new ones. What sensor metrics would you want graphed on your own custom dashboard?

## Creating Custom SQL Sensors

In addition to customizing the InformixHQ UI with our dashboards feature, you can now customize the sensors that are monitored by the agent as well. This allows you to customize InformixHQ's monitoring to gather exactly the data that you care about when it comes to monitoring your database server's performance.

Any sysmaster SQL query can now be turned into a sensor. And InformixHQ offers an easy to use UI for defining these sensors, including a preview of sensor data. Let's walk through that process right now.

## Adding Custom Sensors

You define custom SQL sensors on the **System Settings > Sensor Management** page which means that only system administrative users in InformixHQ can define these custom sensors. Since you are logged in as the "admin" user in our tutorial though, you can navigate to the **System Settings** page (located on the drop-down menu when you click on your user icon in the top right corner of the header) and then click on the **Sensor Management** page. From here you can review or delete any custom SQL sensors, or add new ones.

Let's walkthrough defining a SQL sensor:

1. Click **Create Sensor**
2. On the **SQL** tab
  - a. Select a server. This server will be used to validate your SQL query and to preview the sensor data.
  - b. Enter your SQL query. For example, the following query returns the average IO and lock wait times for database sessions:  
  

```
SELECT avg(iowaittime) as average_io_waittime, avg(lkwaittime) as average_lock_waittime from syscsblst a, sysrstcb b WHERE a.address = b.scb
```
  - c. Click **Run** to execute the query and preview the sensor data
  - d. For this query, we'll skip the optional transpose and primary key options.
3. On the **Metrics** tab, define each metric returned by the query:
  - a. Name – display name for the metric in the UI
  - b. Optional Unit (bytes, percentage, or none) – will be used for formatting when displaying the data graphically in the UI. Just keep "none" for our example query.
  - c. Optional default value. This is useful if a query can return null for a metric. Any null metrics will be replaced by the default value before being save in the repository if a default value is provided.
  - d. Optional delta. Leave this unchecked for this sensor. We'll come back to this in the next example.
4. On the **Sensor** tab
  - a. Add a unique sensor id
  - b. Provide a sensor name – this will be the display name for the sensor in the UI
  - c. Provide a sensor description
  - d. Optionally edit the default run interval and data retention interval
5. On the **Preview** tab, you can review your sensor definition
6. Then click **Save Sensor** to save your custom sensor.

**InformixHQ** admin

- Alerting Configuration
- Data Cleanup Configuration
- Sensor Management**
- Server Configuration
- User Management

### Create Sensor

[Save Sensor](#)

SQL Metrics Sensor Preview

Selected server: Idev5\_serv1 [Select...](#)

```
SELECT avg(iowaittime) as average_io_waittime, avg(lkwaittime) as average_lock_waittime from syscsblist a, sysrstcb b WHERE a.address = b.scb
```

[Run](#)

**Data configuration**

Transpose ?  
 Disabled

Primary Key ?  
 None

[Next >](#)

**Data preview**

average_io_waittime	average_lock_waittime
0.3752807896641775	0

**InformixHQ** admin

- Alerting Configuration
- Data Cleanup Configuration
- Sensor Management**
- Server Configuration
- User Management

### Create Sensor

[Save Sensor](#)

SQL Metrics Sensor **Preview**

**Sensor JSON Preview**

```
{
  id: "session_wait_time"
  name: "Session Wait Time"
  description: "Monitors session wait time"
  meta: {
    default: {
      type: "sql"
      sql: "SELECT avg(iowaittime) as average_io_waittime, avg(lkwaittime) as average_lock_waittime from syscsblist a, sysrstcb b WHERE a.address = b.scb"
      sleepBetweenExecution: 60
      dataRetentionInterval: 30
    }
    defaults: {
      average_io_waittime: "Avg IO Wait Time"
      average_lock_waittime: "Avg Lock Wait Time"
    }
  }
  metrics: {
    average_io_waittime: {
      name: "average_io_waittime"
    }
    average_lock_waittime: {
      name: "average_lock_waittime"
    }
  }
}
```

Now you've created a custom SQL sensor to monitor session wait times on your Informix instances.

Before moving putting this sensor to use, let's create two more custom SQL sensors that demonstrate delta sensor metrics.

Define two more custom SQL sensors for the following queries:

1. Returns the number of commits performance on the database



```
SELECT value as commits from syssmhdr where name = 'pf_iscommits'
```

2. Returns the number of rollbacks performance on the database

```
SELECT value as rollbacks from syssmhdr where name = 'pf_isrollbacks'
```

Use the same process as described above to create custom sensors to monitor commit and rollback activity on your database server. The only additional thing for these sensor is that you want to make sure to select the **delta** option on the **Metrics** tab. When you specify a metric to be a delta metric, you are indicating to the agent that you want it to store in the repository the delta between the previous and current values instead of the raw values. Furthermore, when computing delta metrics, the agent will divide by the number of seconds between the two readings to compute the **delta per second** for the metrics. This means that you are getting a sensor that will monitor and store the “commits per second” and the “rollbacks per second” that are being performed on the database server.

### Using Custom Sensors

Once defined custom SQL sensors look and behave like any of the built-in sensors.

Go to the **Monitoring** page for your Root Group and click the **Add Sensors** button. Notice that your three custom SQL sensors – Session Wait Times, Commits, and Rollbacks – now show up in this list with the names and description you provided when creating the sensors. Add these three sensors to your monitoring profile and save to have the *server1* and *server2* agent start monitoring those metrics.

**Add Sensors**

Search...

<input type="checkbox"/>	Name ^	Description
<input type="checkbox"/>	Memory Segments	Monitors the memory segments used by the Informix database server
<input type="checkbox"/>	Online Log	Monitors the Informix database server's log file.
<input type="checkbox"/>	Operating System CPU	Monitors CPU usage on the operating system
<input type="checkbox"/>	Operating System CPU per Core	Monitors CPU usage per core on the operating system
<input type="checkbox"/>	Operating System Disk I/O	Monitors Disk I/O activity on the operating system
<input type="checkbox"/>	Operating System Disk Utilization	Monitors operating system disk usage for the storage devices used by the Informix database server.
<input type="checkbox"/>	Operating System Memory	Monitors memory usage on the operating system
<input type="checkbox"/>	Operating System Network I/O	Monitors network i/o activity on the operating system
<input type="checkbox"/>	Session Wait Time	Monitors session wait time

Previous 1 2 Next Rows per page: 10

Add Sensors Cancel

Now go the **Alerting** page and add some alerts for these new custom sensors.

Finally go back to your custom dashboards and add some new panels to graph session wait times and to graph commit and rollback activity on *server1* and *server2*.

As you can see, custom SQL sensors behave exactly the same as built-in sensors from a monitoring, alerting, and dashboarding perspective. This means that you can create your very own sensor for monitoring your database server and have these custom sensors be fully integrated into everything else you are doing in InformixHQ.

**Activity:** Now that you've added a few custom sensors based on suggestions provided in this tutorial and you've started monitoring those sensors and added them to your dashboard, take a few minutes to create your own SQL sensors. What you would like to monitor on your database server? Create your own custom SQL sensor, add it to your monitoring profile, and then add it to a dashboard!

## Part 6: Users and Permissions

The last topic to cover in this tutorial is users and permissions. InformixHQ supports the creation of users so that many people in your organization can login and use InformixHQ and each user can be granted the appropriate level of permissions in the tool.

Users of InformixHQ are unique to the tool. They are not related to the database servers you use to connect to your Informix database server or to the operating system users on the host machine it's running on.

There is one special permission in InformixHQ – **system administrator** – that can be granted to one or more users. System administrator users of InformixHQ:

- Have full privileges (read + SQL + admin) on all database servers and groups defined in InformixHQ
- Can add users to InformixHQ and modify existing user privileges
- Can configure InformixHQ settings including which user alerting notification services are available and when the data repository cleanup job is run.

The “admin” user that is created when InformixHQ is initialized for the very first time is a system administrator user for InformixHQ. Other users can subsequently be created that also have system administrative access.

The system administrator will have to define the permissions for all other users of InformixHQ. Non-system administrator users can be granted **Read**, **SQL**, and **Admin** permissions on the various database servers and groups that are managed through InformixHQ. **Read** permissions allows the user to see a particular server in the InformixHQ UI. **SQL** permissions allows them to run ad-hoc SQL queries against that database server on the **Schema Manager** page. **Admin** permissions allows the user to administer the database server in InformixHQ, for example, edit a scheduler task or create a new dbspace.

**Read**, **SQL**, and **Admin** permissions are mutually exclusive. Think of them like the file permissions on a Linux/UNIX operating system. If you want a user to be able to see and administer a server in InformixHQ, you must that user **Read + Admin** permissions.

Permissions in InformixHQ are hierarchical. If you grant a user permission on a group, that user will inherit the same permission on all child servers and sub-groups of that group.

### Adding Users and Configuring Permissions

To add users to InformixHQ in our tutorial, make sure you are logged in as the “admin” user and then go to the **System Settings > User Management** page (which is accessed through the menu shown when you click on the user icon in the header).

Use the **User Management** page to add new users, edit existing user permissions, remove users, unlock/lock users, or reset user passwords.

When you add a new user, you will provide a user name, password, and choose their level of permissions.

**InformixHQ** admin

Alerting Configuration  
Data Cleanup Configuration  
Sensor Management  
Server Configuration  
**User Management**

### New User

**User Info**

Username:

New Password:

Confirm New Password:

☐ System Administrator

**Save**

**Select Permissions** Clear All

Groups & Servers	Read	SQL	Administer
Root Group	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
HQ Server	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
server1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
server2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Create the following user in InformixHQ:

1. user1 – Read access on *server1* and *server2*, no access on the *HQ Server* group
2. user2 – Read access on the *Root Group*, read+sql+admin access on *server2*
3. user3 – Read+admin access on *server1*, no access on any other servers or groups

Besides managing user permissions from the **User Management** page, you can also see and manage the permissions on each server or group in InformixHQ on the corresponding **Permissions** page on the side bar menu of that object. For example, navigate to the **Permissions** page for *server1* and *server2* to see how you can view and manage the permissions of all users for each particular server from a single page.

**InformixHQ** admin

Root Group > server1 > Permissions

server1

Setup  
Dashboards  
Monitoring  
Alerting  
**Permissions**  
Incidents  
Configuration  
Logs  
Performance

### User Permissions

**Search...**

User Name	Permissions
user3	Read, Administer
user1	Read

**Inherited User Permissions** [\(View parent permissions\)](#)

**Search...**

User Name	Permissions
user2	Read

**Activity:** Now that we have granted differing permissions to *user1*, *user2*, and *user3* log out of InformixHQ as the “admin” user and log in as each of these users in turn to see the different levels of access. Note that servers that you don’t have read access to don’t even show up on the group dashboard. Servers that you don’t have admin access to have all of the admin action buttons disabled. Servers that don’t have SQL access to do not have the SQL tab available on the **Schema Manager** page.

## Conclusion

Congratulations for making it to the end of this tutorial! Hopefully you’ve gotten a good feel for InformixHQ and learned how it can benefit you in your organization.

We are committed to continue investing in InformixHQ and making it even better with each future release. If you have feedback on InformixHQ after completing this tutorial, we’d love to hear it! We really want to build a tool that serves your needs and provides a complete solution for monitoring and managing your Informix database server instances.

## Resources

- Find InformixHQ documentation in the 14.10 Knowledge Center:  
[https://www.ibm.com/support/knowledgecenter/en/SSGU8G\\_14.1.0/com.ibm.ifxhq.doc/informixhq.htm](https://www.ibm.com/support/knowledgecenter/en/SSGU8G_14.1.0/com.ibm.ifxhq.doc/informixhq.htm)
- Find InformixHQ videos on the Informix Channel on YouTube:  
[https://www.youtube.com/channel/UCtp2l98gWjv\\_6QmCa92tQYQ/featured](https://www.youtube.com/channel/UCtp2l98gWjv_6QmCa92tQYQ/featured)
- Open feature requests (RFE) for InformixHQ in the IBM AHA tool:  
<https://ibmanalytics.ideas.aha.io/?project=INFX>