knowledge17

May 7–11, 2017 • Orange County Convention Center • Orlando, FL

Lab Guide

Performance Analytics 302:

Advanced Use Cases and Performance Analytics Administration Topics

Pieter Goris & Paul Rice

Default Login / Password: Knowledge17

admin / Knowledge17

itil / Knowledge17

employee / Knowledge17

Lab Sponsored by Microsoft Azure



servicenow

This
Page
Intentionally
Left
Blank

Lab Goal

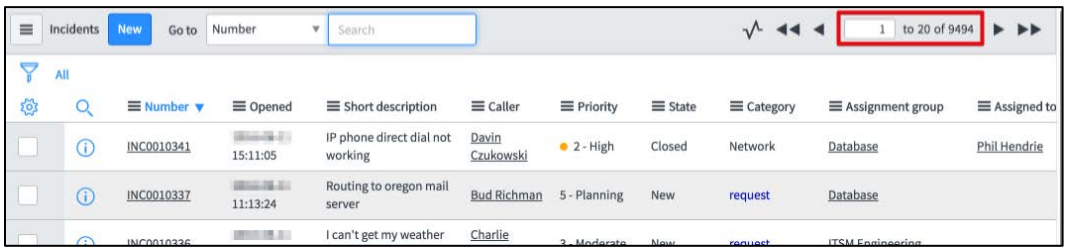
Lab 1
Collect Scores

Run jobs to collect scores for automated indicators. The scores are stored in separate tables within the ServiceNow database. After the scores are collected, the scorecards and dashboards are populated for both the automated indicators and the formula indicators.

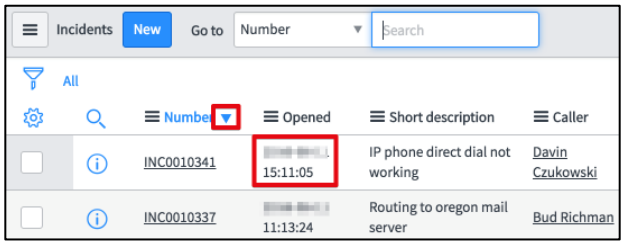
Run a historic scores collection to collect scores for multiple collection periods. This populates the trends for the indicator scores. When running historic scores collections, be aware that for some indicators and/or indicator-breakdown combinations, accurate historic numbers cannot be collected.

Verify the Incidents in the Instance

1. Type **Incident** in the **Type filter text** box on the left-hand navigation pane.
2. Navigate to **Incident > All** and verify that there is a total of **9494** incidents in the instance:



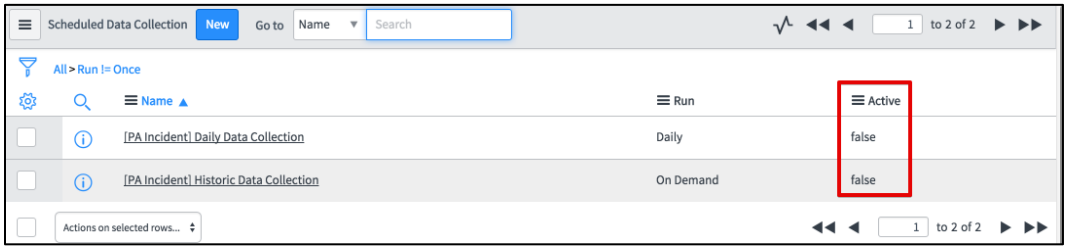
3. Also check that the highest numbered incident has an Opened date of yesterday.



Configure Existing Job for Historic Scores Collection

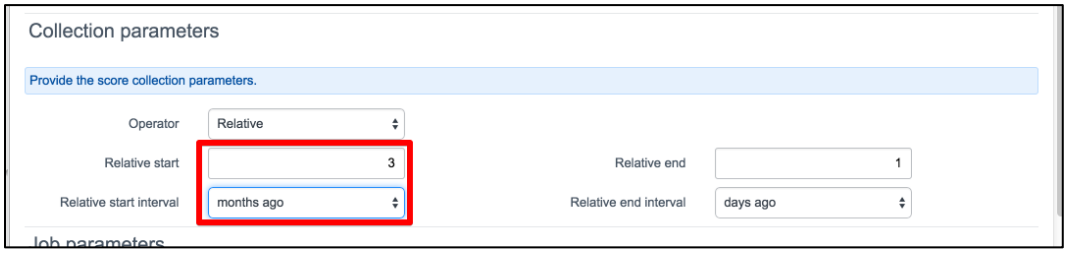
1. Type **Jobs** in the **Type filter text** box on the left-hand navigation pane.

2. Navigate to **Performance Analytics > Data Collector > Jobs**. Verify that there are two jobs and for both **Active** is set to **false**:

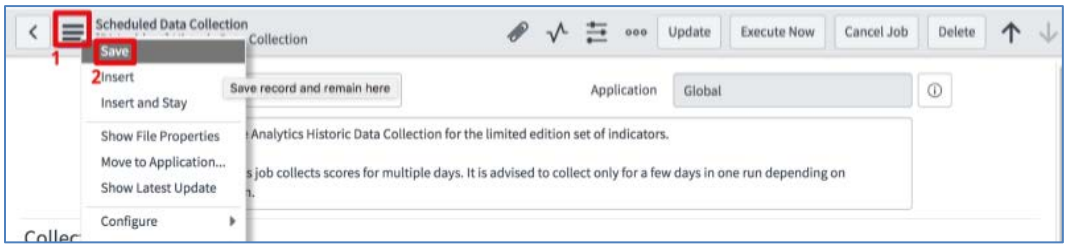


3. Open the **[PA Incident] Historic Data Collection**.

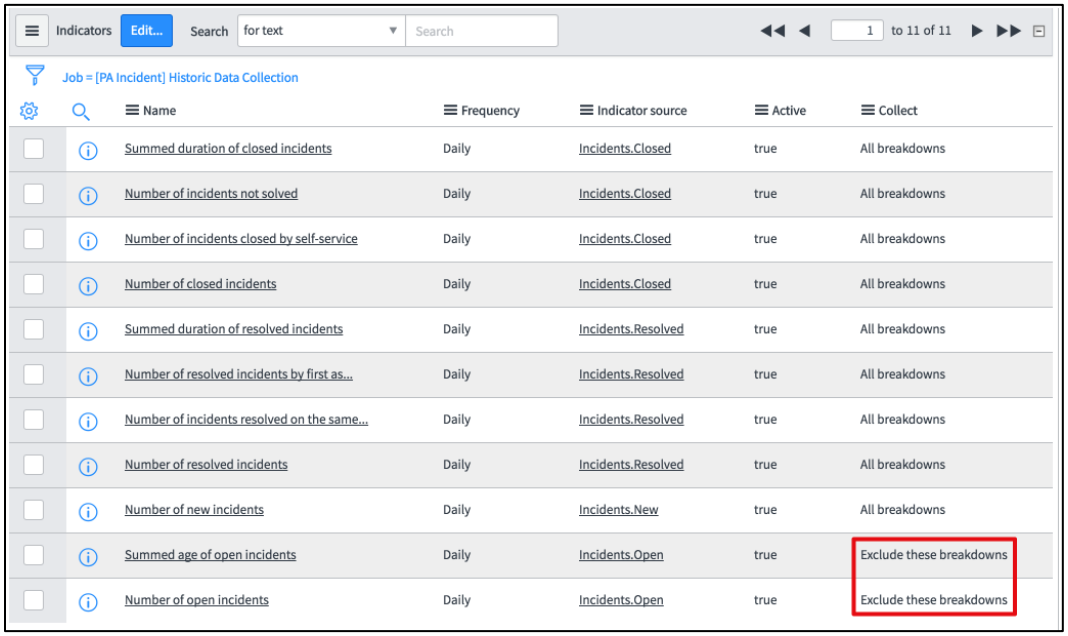
4. Update in the **Collection parameters** section:
Relative start: **3**
Relative start interval: **months ago**:



5. Save the updated job record: **click** on the 'hamburger' icon (1) in the title bar or open the context menu by right-clicking anywhere in the title bar of the record and select **Save** (2) from the drop-down menu:



6. Scroll to the Job indicators section and notice that for the indicators **Summed age of open incidents** and **Number of open incidents** the job will not collect scores for some breakdowns:



Name	Frequency	Indicator source	Active	Collect
Summed duration of closed incidents	Daily	Incidents.Closed	true	All breakdowns
Number of incidents not solved	Daily	Incidents.Closed	true	All breakdowns
Number of incidents closed by self-service	Daily	Incidents.Closed	true	All breakdowns
Number of closed incidents	Daily	Incidents.Closed	true	All breakdowns
Summed duration of resolved incidents	Daily	Incidents.Resolved	true	All breakdowns
Number of resolved incidents by first as...	Daily	Incidents.Resolved	true	All breakdowns
Number of incidents resolved on the same...	Daily	Incidents.Resolved	true	All breakdowns
Number of resolved incidents	Daily	Incidents.Resolved	true	All breakdowns
Number of new incidents	Daily	Incidents.New	true	All breakdowns
Summed age of open incidents	Daily	Incidents.Open	true	Exclude these breakdowns
Number of open incidents	Daily	Incidents.Open	true	Exclude these breakdowns

Note about indicators in a historic scores collection job

On the Job indicators section, there are only eleven (automated) indicators related to this collection job. The six automated indicators from the Complimentary Version that are not in this list are:

- Number of open incidents not updated in last 5 days
- Number of open incidents not updated in last 30 days
- Number of open overdue incidents
- Number of reassigned open incidents
- Summed age of last update of open incidents
- Summed re-assignment of open incidents

The first two indicators have a query condition like '**Updated relative on or before** x **days ago**'. Therefor a historic scores collection for these two indicators would not collect accurate numbers as the collection is run against the 'sys_updated_on' column of the incident table and not the audit log. So the job retrieves the current 'sys_updated_on' date value for an incident, which is likely not the same date value as the incident had two weeks ago.

The other indicators also have conditions to which the same logic applies; therefor they are not fit for historic scores collection.

*In general, for an indicator (source) on **open records** which holds **conditions** based on fields for which the values are likely to change during the lifecycle of the open record, an accurate historic scores collection cannot be run.*

Also in the Complimentary Version set there are two indicators that are combined with two breakdowns that do not collect accurate numbers in a historic scores collection. These indicator-breakdown combinations are:

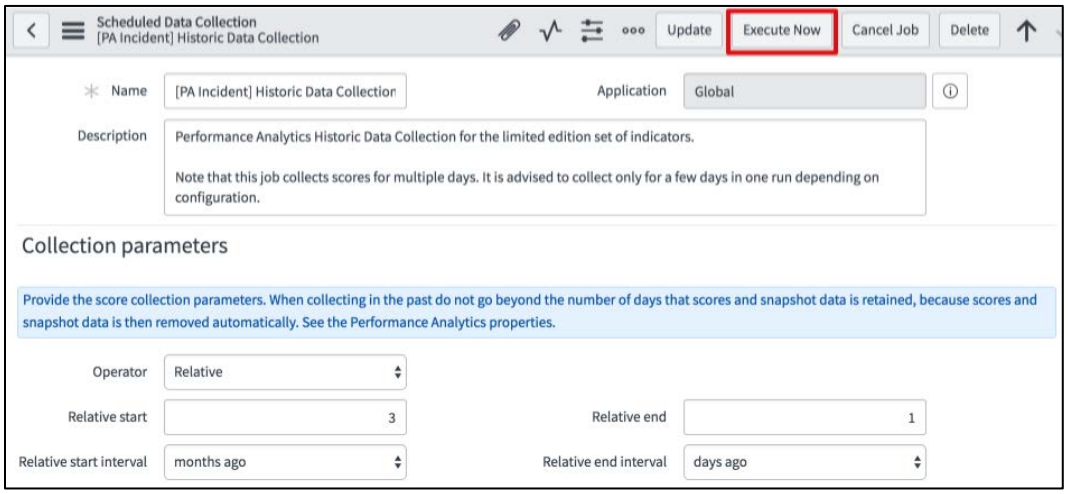
- **Number of open incidents**, combined with:
  - **Assignment Group**
  - **State**
- **Summed age of open incidents**, combined with:
  - **Assignment Group**
  - **State**

The reason for not collecting accurate numbers is the same as above: a collection retrieves the current Assignment Group and State values for an incident, which are likely not the same values the incident had two weeks ago. In the historic scores collection job these breakdowns should not be included for the two indicators.

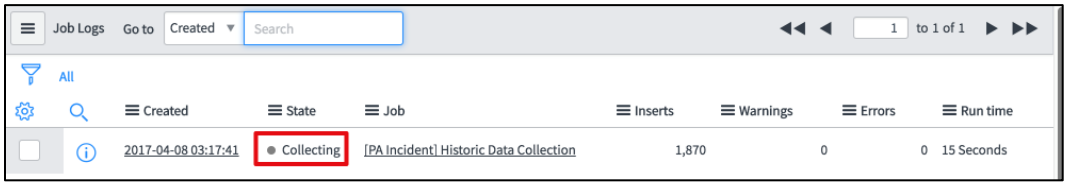
*In general, for an indicator (source) on **open records** which has **breakdowns** based on fields for which the values are likely to change during the lifecycle of the open record, an accurate historic scores collection cannot be run.*

Run Historic Scores Collection

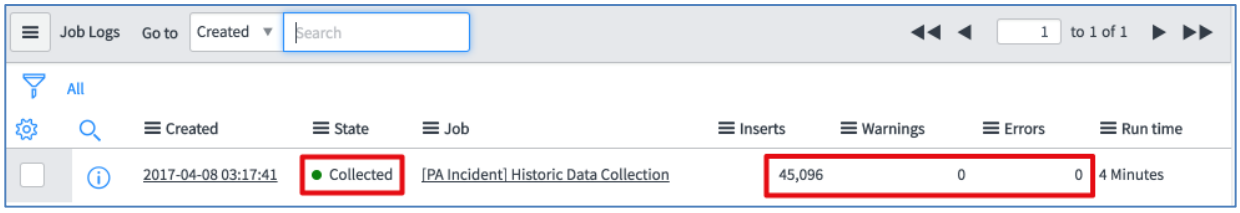
1. While still in the [**PA Incident] Historic Data Collection** record in edit mode, click the **Execute Now** button:



2. Navigate to **Performance Analytics > Job logs**. When the collection is still running, the view shows one log with **State** on **Collecting**:



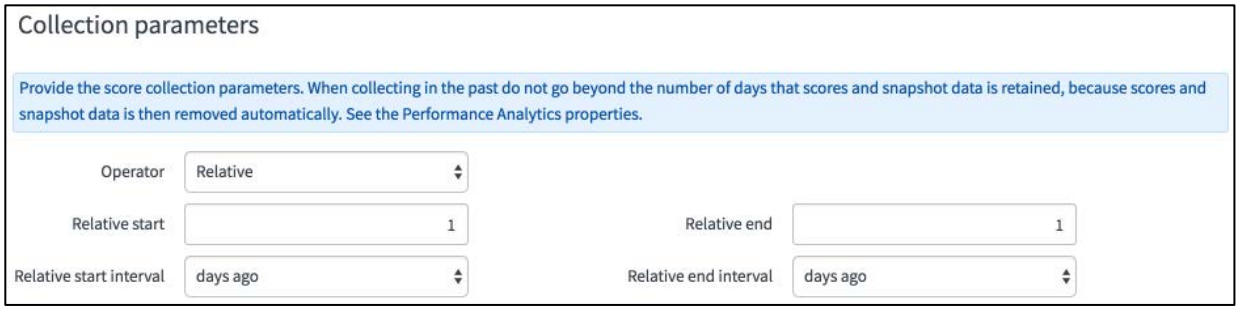
When the collection has finished the collection log shows:
State is **Collected**,
Inserts is **45,476**,
Warnings is **0**,
Errors is **0**:



The number of inserts might slightly differ as it depends on the actual number of days in the last 3 months, but the number of warnings and errors should be 0.

Run a 1-Day Scores Collection for Yesterday

1. Navigate back to **Performance Analytics > Jobs** and open the **[PA Incident] Daily Data Collection**. There is no need to change any of the settings. So the **Collection parameters** should look like:



   The **Job indicators** section should list the 16 OOB automated indicators for all breakdowns for incident management that come with the Complimentary version.

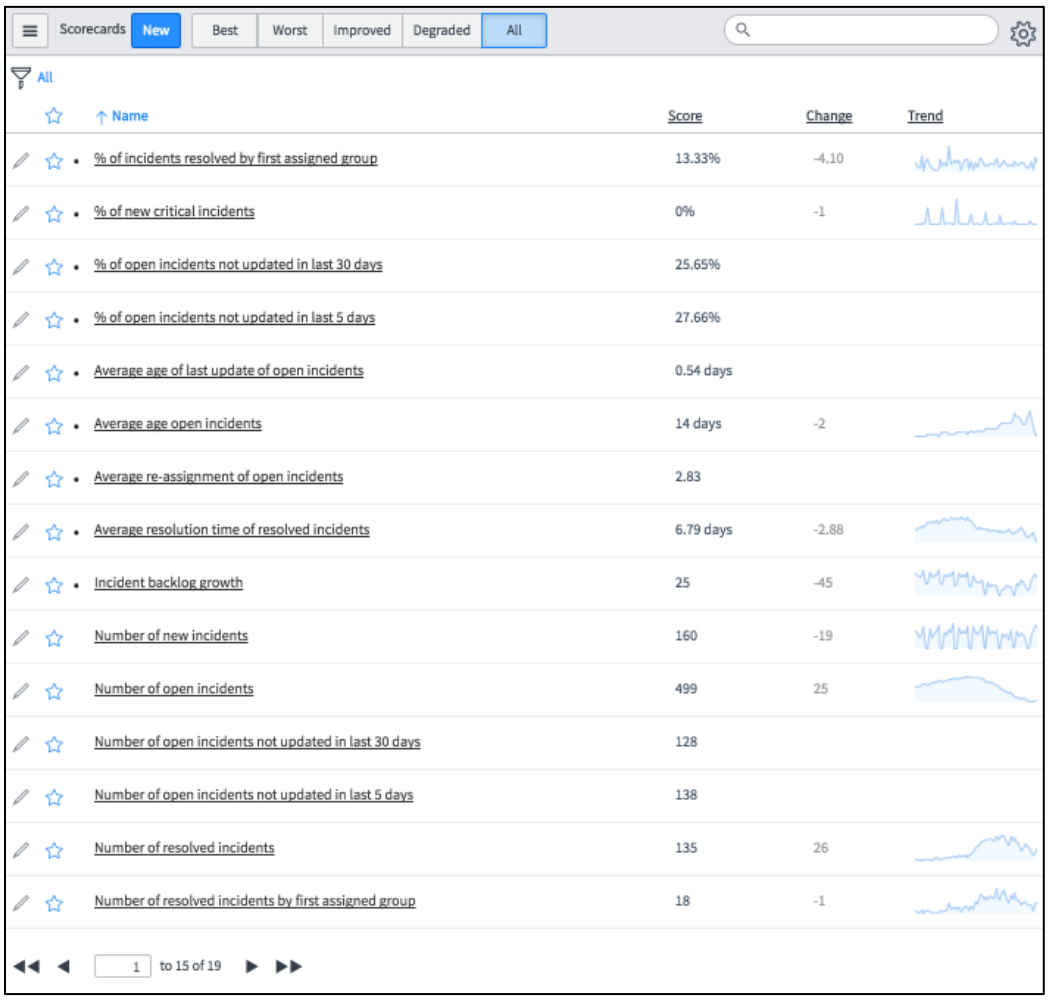
2. Click the **Execute Now** button.

3. When the collection has finished the collection log shows:
   State is **Collected**,
   Inserts is **1952**,
   Warnings is **0**,
   Errors is **0**:

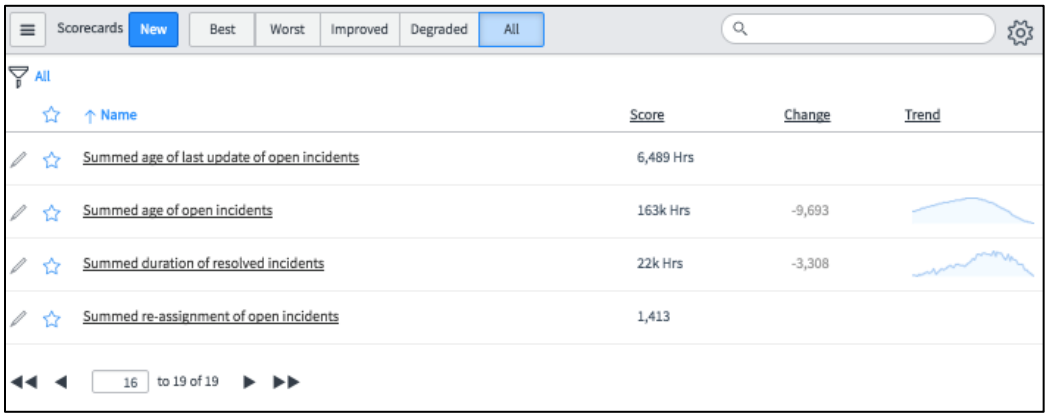
   Once more, the number of inserts might slightly differ as it depends on the actual number of days in the last 3 months, but the number of warnings and errors should be 0.

Lab Success Verification

1. Navigate to **Performance Analytics > Scorecards**. Your screen should look like this**:**



2. And:



Notice that also for the formula indicators, the top 9 in the list above, scores have been calculated.

Lab Goal

Lab 2 Breakdown Elements Security

Breakdown elements security lists can be defined on breakdown sources. With a security list, access to specific elements can be set to one or more roles. A security list is a combination of one or more elements and one or more roles.

The security list **type** can be one of the following:

- Blacklist: The elements on the list are not accessible to the roles on the list, and implicitly access is granted to the unlisted roles
- Whitelist: The elements on the list are only accessible to the roles on the list, and implicitly access is denied to the unlisted roles.

The **roles** on the security list must be manually picked.

The **elements** on the security list can be:

- Manually selected
- Defined by using a static filter such as **Active is true**
- Defined by using a dynamic filter such as **Manager is (dynamic) me**

This last option is interesting when used in combination with a breakdown dashboard.
In this lab, a breakdown elements security list is created on the breakdown source Groups.

Create a Whitelist Breakdown Elements Security Lists for Groups

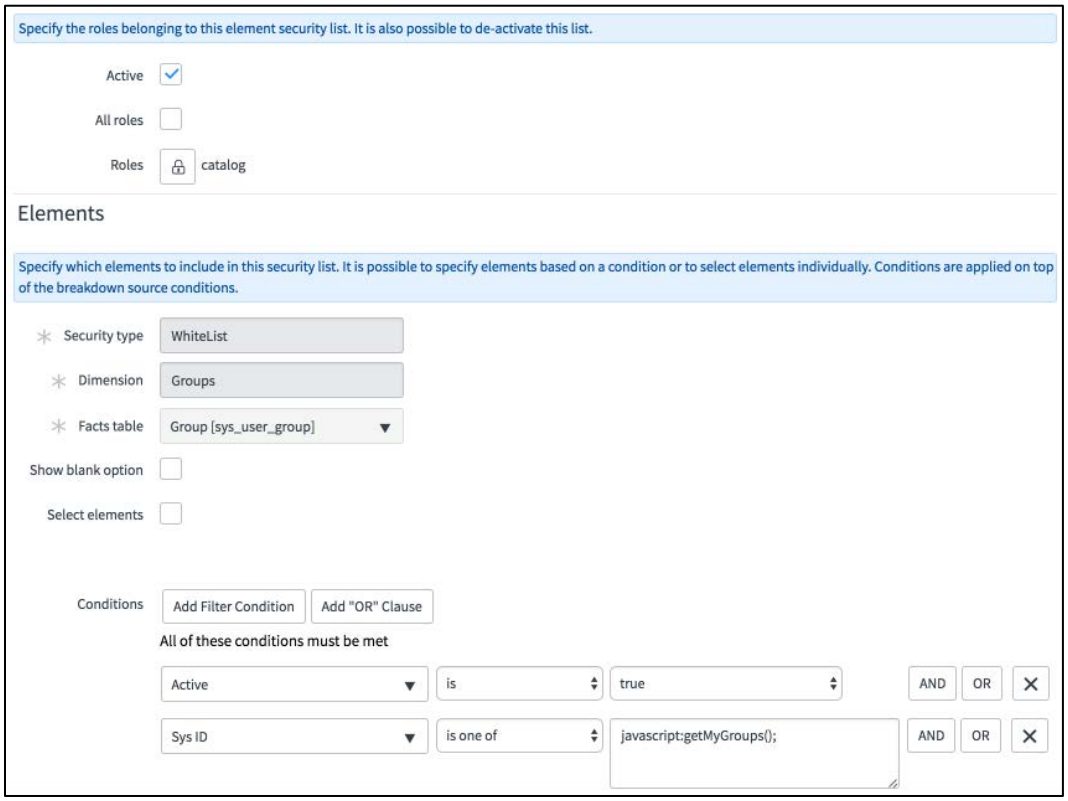
1. Open the existing breakdown source **Groups** (**Performance Analytics > Sources >Breakdown Sources**).

2. On the Security tab, set the Security type to **Whitelist** and **Save** the form.

3. Scroll to the Elements Security List section and click **New**.

4. Create an Elements Security List by setting the following parameters:

   Name: **I am Member of**
   Access control section:
   Active box: **selected**
   Role: **catalog**
   Elements section:
   deselect the **Show blank option** box (users cannot see the overall indicator score)
   leave the **Select elements** box deselected (use the condition builder to create a filter for the elements on the list)
   Conditions:
   Active is true AND
   Sys ID is one of javascript:getMyGroups(); :*



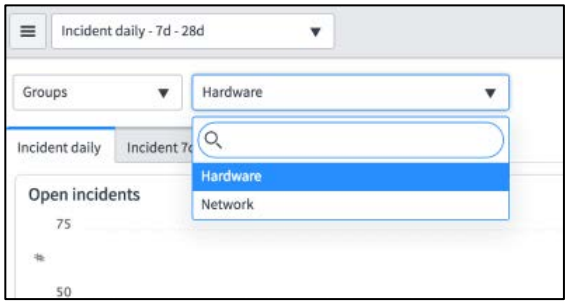
* When using the Sys ID attribute in combination with the 'is' or 'is one of' operator, any script or script includes that returns Sys ID's, will work.

Lab Success Verification

1. Navigate to **User Administration > Users** and select the user David Dan. Assign the roles **pa_viewer** and the **catalog**. While on the user record, notice that David Dan is member of 2 groups:
   Hardware
   Network

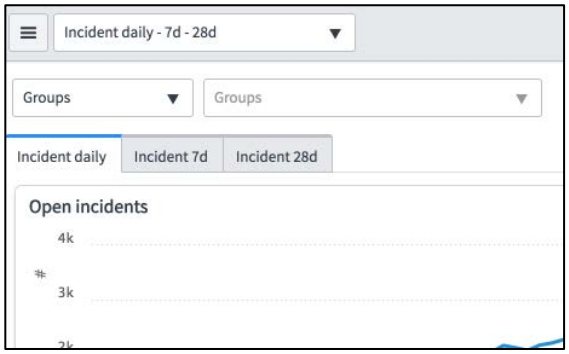
2. Next, select the user Beth Anglin and assign the role **pa_viewer**. While on the user record, notice that Beth Anglin already has the role **catalog,** and she is member of 6 groups:
   Catalog Request Approvers for Sales
   Database
   Hardware
   Network CAB Managers
   Service Desk
   Software

3. Impersonate David Dan and go to the dashboard **Incident daily - 7d – 28d**. Notice that David has access only to the two groups in which he is a member.



4. Next, impersonate Beth Anglin and go to the dashboard **Incident daily - 7d – 28d**. Notice that Beth has access to the six groups in which she is a member.

5. Log on as **admin**. On the security element list, select the **Show blank** option box. Re-impersonate David Dan or Beth Anglin and notice how they can now view the **overall** indicator score:



Lab Goal

Lab 3
Index
Indicators

The performances of processes, services, groups, etcetera are usually tracked and monitored using more than one indicator. When viewing and analyzing performance of these processes, business services, or workgroups, the overall picture can be confusing and ambiguous. For example:

- Although the scores for 3 indicators improved somewhat, the scores for 2 of them are still below target and 1 is above target.
- The score for 1 indicator remained more or less the same and is still below target.
- The score for 1 indicator did significantly deteriorate, but is fortunately just above target.

So, what is the overall picture? Is the overall performance of the process/service/group still at or above the desired level? Did the overall performance improve?

An index indicator can answer these questions. An index indicator is a formula indicator. With an index indicator, the scores of multiple indicators are aggregated into one score. It is a (weighted) average of ***x*** indicators. If the (weighted) sum of these ***x*** indicators is improving, the calculated score of the index formula goes up. Like any other indicator, the index indicator shows if the score is good or not and if the score has improved or not.

The idea is to calculate an indexed score value (as an offset of 100) for each indicator. With indexed scores, you are mathematically allowed to calculate an overall average.

The basic formula to calculate the indexed score for an indicator that has a Maximize direction is:

100 + (((actual score – target) / target) * 100)

For indicators that have a Minimize direction the formula is:

100 – (((actual score – target) / target) * 100)

If the score of an underlying indicator is improving (up or down), the index indicator score is increasing.

If there is not a target value set for an indicator, use a norm value instead. Indicators that have a target or norm value equal to 0 cannot be used in the index indicator. In the formula for the indexed score, there would be a division by 0. That is not allowed.

For an index indicator, a target should be set at 100. That is the calculated, overall, indexed score if all underlying indicators have an actual score equal to their target or norm value. The direction of the index indicator is always Maximize.

In this lab, an index indicator is created for the Incident Management process: one indicator that shows what the overall performance on the incident management process is and how it is developing. This process index indicator is based on 3 underlying indicators that already exist:

Indicator	Direction	Target or norm value
Average resolution time of resolved incidents	Minimize	17.5
% of incidents resolved by first assigned group	Maximize	70
% of new critical incidents	Minimize	5

First, for the three underlying indicators, an index formula indicator is created. Next, these index formula indicators are used to create the overall index indicator for the Incident Management process.

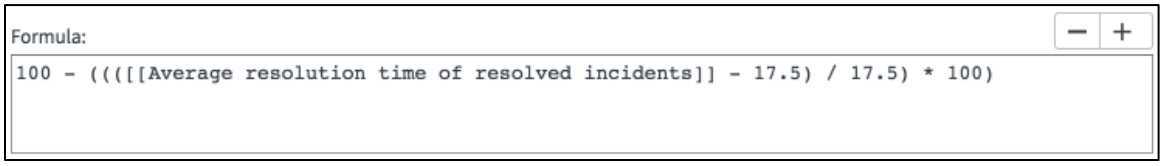
Create 3 Index Formula Indicators for the Underlying Indicators

1. Navigate to **Indicators > Formula Indicators** and click the **New** button.

2. Update the following on the new formula indicator form:

   Name: **Index for Average resolution time of resolved incidents**
   Direction: Maximize section
   Unit: **Index** (use the look up icon to create a new unit, a **Format** is not necessary for this unit)
   Precision: **2**

   On the Formula section add the formula:

   On the Access control section, select the **Publish on scorecard**.
   On the Other section, select the **Apply time series to result**.
   On the Time series exclusion related list (so save the new formula record first), add all SUM time series to the exclusion list.

3. Repeat this (using **Insert and Stay with Relations** from the context menu) for:

   the underlying indicator **% of incidents resolved by first assigned group** using as formula:

```
Formula:
100 + (((([% of incidents resolved by first assigned group]] - 70) / 70) * 100)
```

   the underlying indicator **% of new critical incidents** using as formula:

```
Formula:
100 - (((([% of new critical incidents]] - 5) / 5) * 100)
```

Create the Index Indicator for the Incident Management Process

1. Navigate to **Indicators > Formula Indicators** and click **New**.

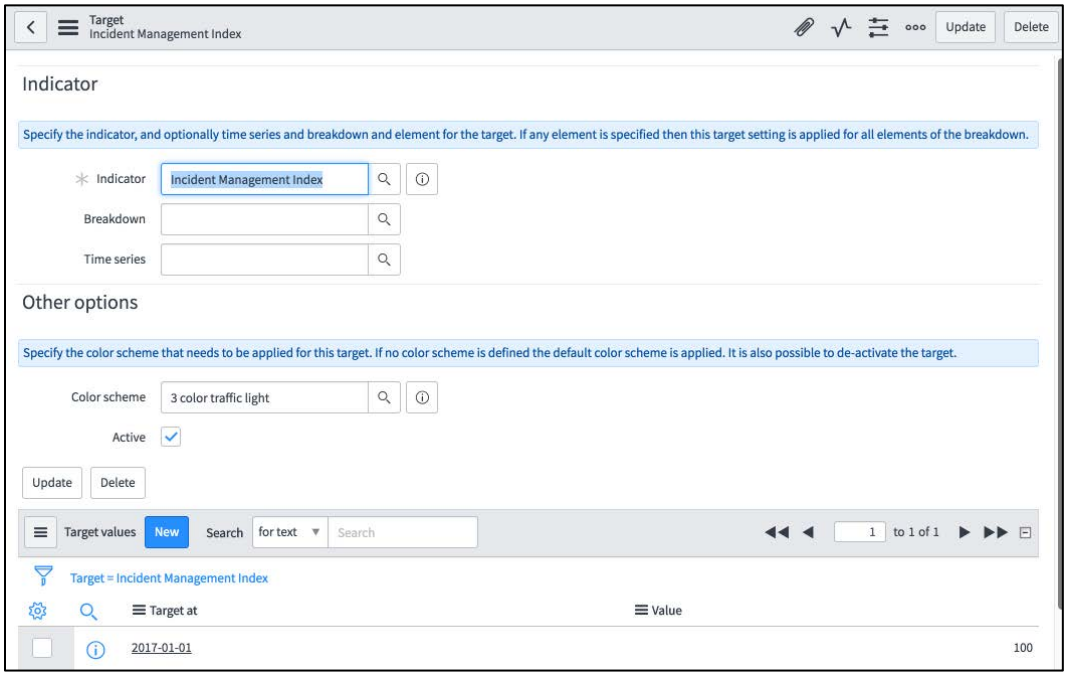
   Update the new formula indicator form with the same settings as for the underlying indicators, except for:

   Name: set it to **Incident Management Index**, (and set **Precision** to **0**, not a must, but it looks better on the detailed scorecard and widgets)
   And create a formula like:

```
Formula:
(1/3 * [[Index for Average resolution time of resolved incidents]]) +
(1/3 * [[Index for % of incidents resolved by first assigned group]]) +
(1/3 * [[Index for % of new critical incidents]])
```

   In this formula, the 3 underlying indicators for the incident management process all have the same weight, 1/3. You can adjust these weights; just make sure that the weights add up to 1.

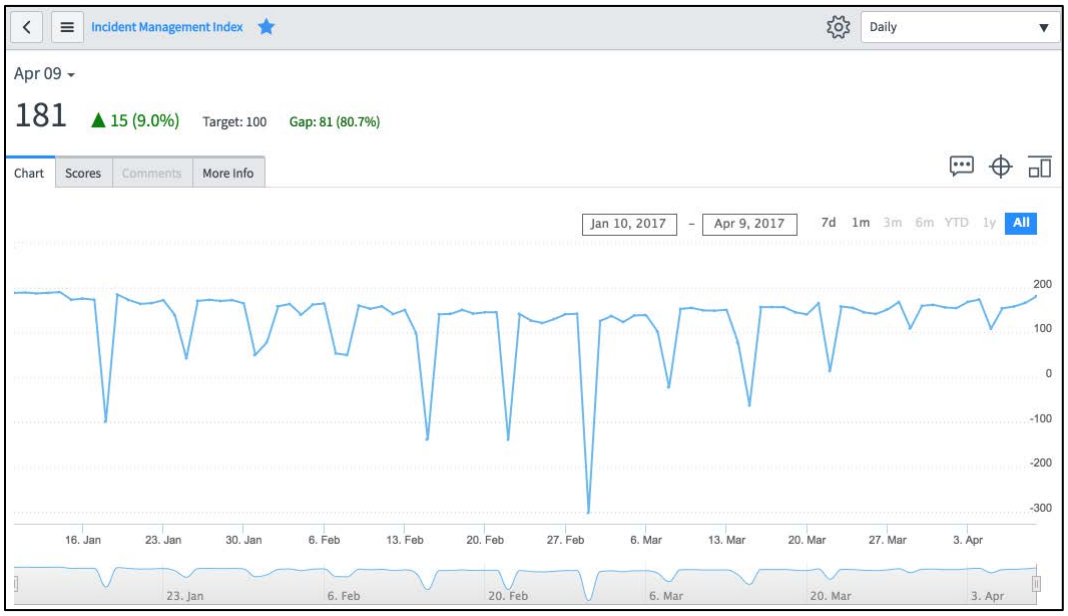
2. Navigate to **Indicators > Targets** and click **New**.
   Update the form:
   Indicator: **Incident Management Index**
   Other options section:
   Color scheme: **3 color traffic light**
   Active: **selected**
   Save the new target from the context menu and add a new target value.
   Update the form
   Target at: **2017-01-01**
   Value: **100**



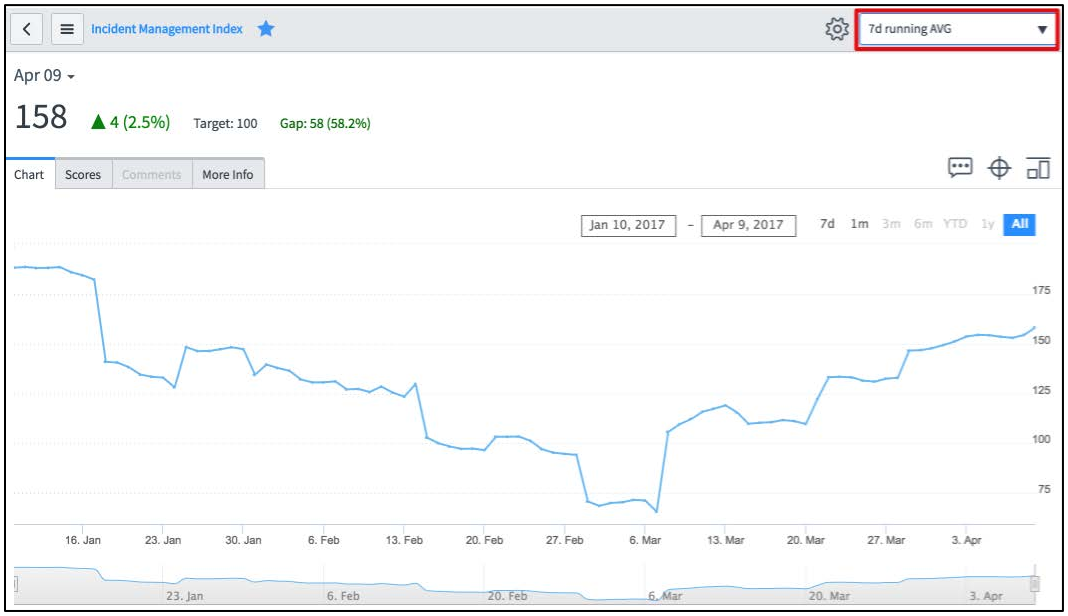
3. Create another target for this indicator with the same settings except for **Time series**. Set it to **7d running AVG**.

Lab Success Verification

1. Go to the detailed scorecard for the **Incident Management Index** indicator. It should look like the example:

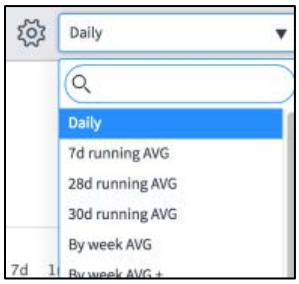


2. In general, when comparing Performance Analytics with reporting, the patterns/trends are more important than a single score. Even more so with index indicators as the score value itself is not meaningful. An index score above 100 indicates that the performance is good, above desired level. To get a better impression of the performance over time, select the **7d running AVG** from the time series drop-down list in the top-right corner of the detailed scorecard:



Notice that the trend appears to change when changing the time series. On the **Daily** view, a somewhat stable trend shows with a few major negative day scores. In the **7d running AVG** view, there is a downward trend for 8 weeks followed by a 5-week upward trend; the score more than doubles in the last 5 weeks. So the trend appears to change, but it is primarily due to a different y-axis scale.

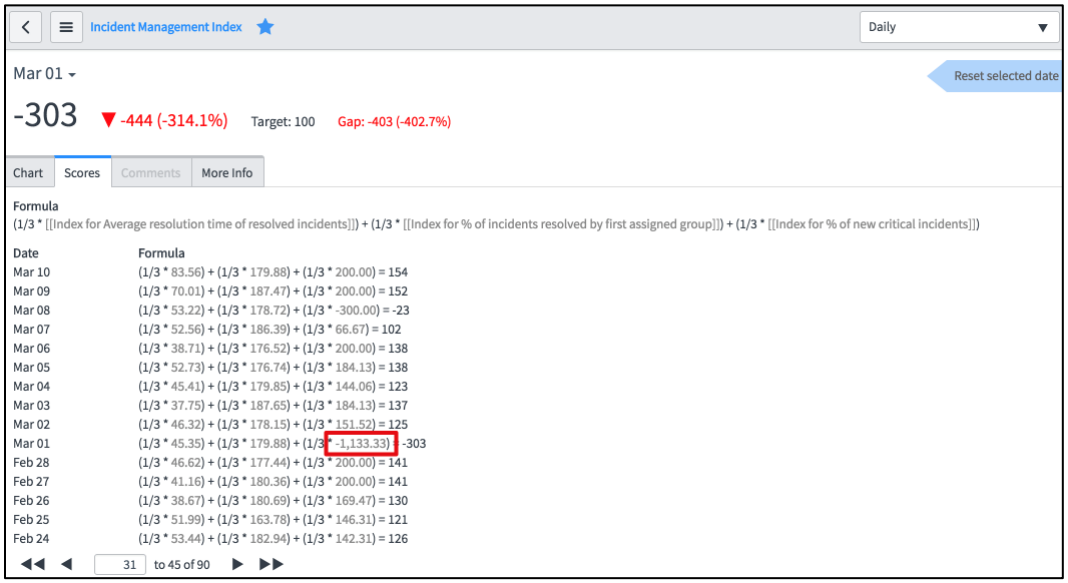
Also notice that on the time series, the SUM time series are not available:



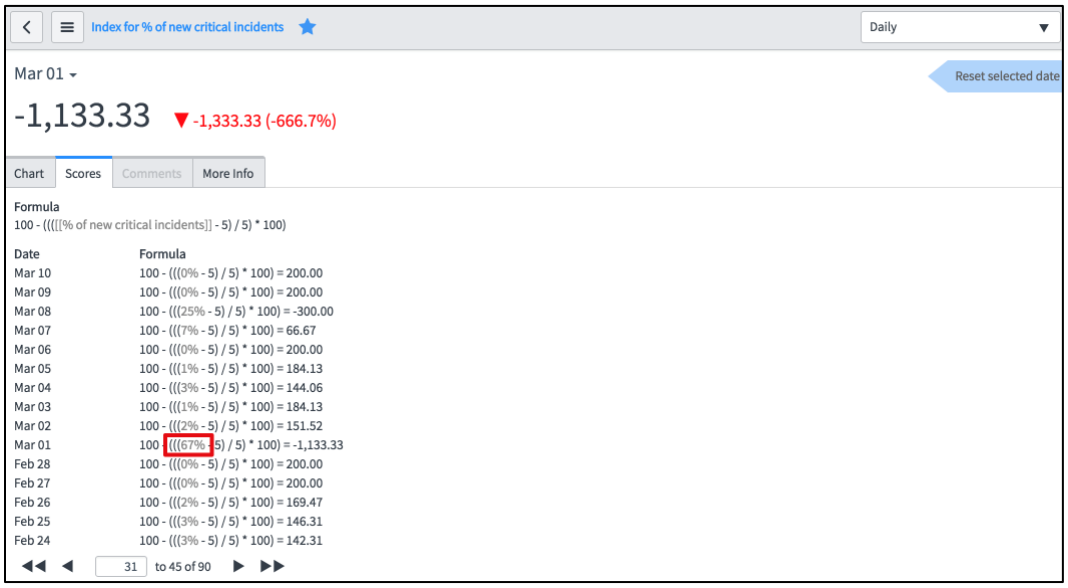
3. Next, to further analyze the dips in the performance:
   switch back to the **Daily** view (1),
   select the lowest score on the chart (2),
   notice the date (3) (that will not be March 1 in the lab session!),
   open the **Scores** tab (4):



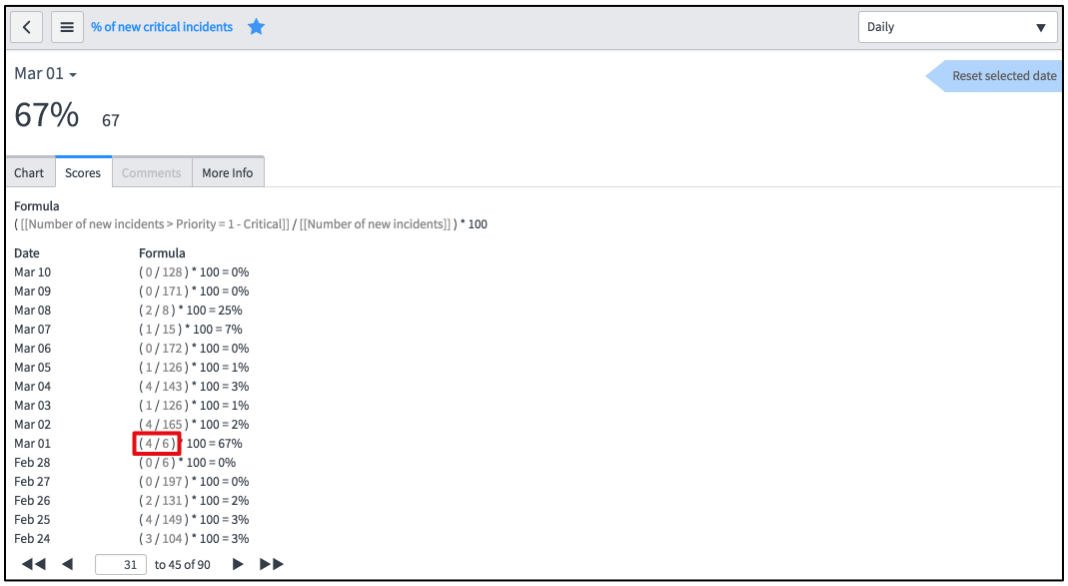
4. On the scores tab, notice the very negative score for the **Index for % of new critical incidents** for the selected date:



5. Click on the score. This opens the detailed scorecard for **Index for % of new critical incidents**. Open the **Scores** tab for this indicator and notice the exceptionally high score for the **% of new critical incidents** for the selected date:



6. Click on the score again. Now the detailed scorecard for **% of new critical incidents** opens. Go to the **Scores** tab and notice that there is a 67% score value because there are only 6 new incidents registered for the selected date, of which 4 are of **Priority 1 – Critical**.



This is a most likely a weekend day (like the day before) because only a few incidents are registered.

The weekend days have a distorting effect on the performance trend of the indicator. This effect also aggregates to the indexed score of the indicator and the score of the index indicator **Incident Management Index**. That is why the **7d running AVG** gives a better view of the performance trend: as it takes out this weekend effect. Consider setting the **7d running AVG** as the **Default time series** on the **Other** tab of the indicator configuration form.

Note about index indicators

The index indicator can also be created without creating the indexed formula indicators for the underlying indicators first. The configuration of the index indicator would be the same, except for the formula. That formula should look like the example:

Formula:

```
(1/3 * (100 - (((([Average resolution time of resolved incidents]] - 17.5) / 17.5) * 100))) +
(1/3 * (100 - (((([% of incidents resolved by first assigned group]] - 70) / 70) * 100))) +
(1/3 * (100 - (((([% of new critical incidents]] - 5) / 5) * 100)))
```

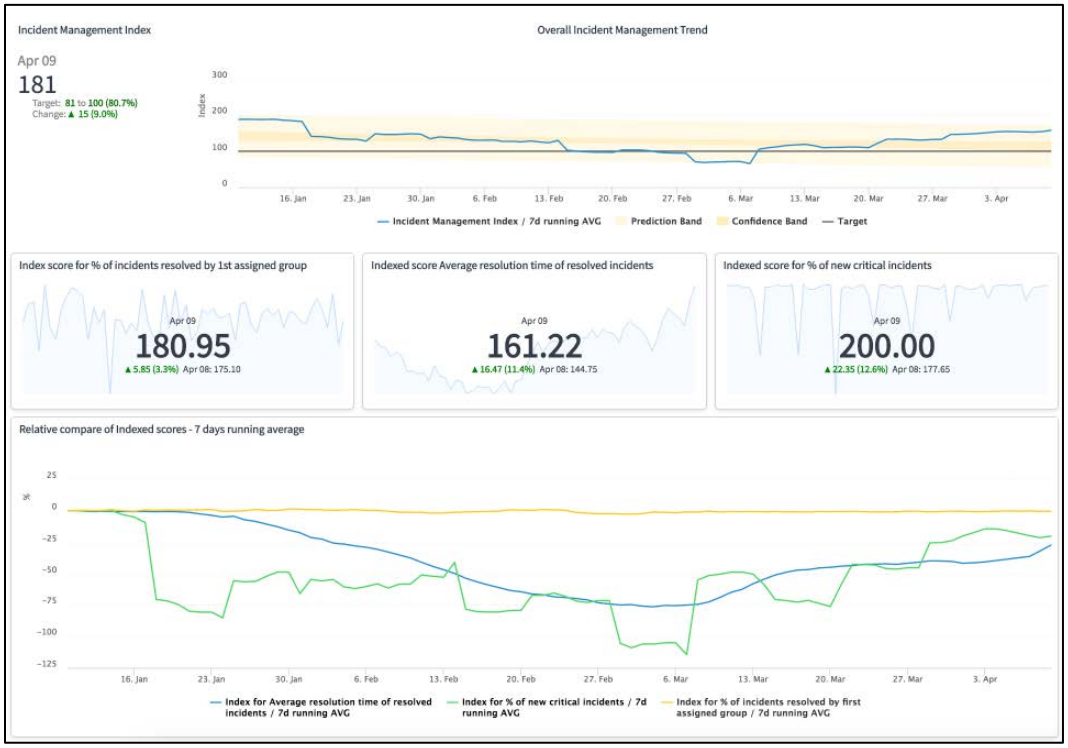
On the detailed scorecard, the **Scores** tab for the index indicator looks like the example:



So the analysis is a bit more difficult, especially if there are more than 3 underlying indicators.

Of course, without having separate indexed formula indicators for the underlying indicators, the indexed scores for these indicators cannot be visualized separately.

When having the separate indexed indicators for the underlying indicators, a dashboard like the example could be created:



Notice how the 2 lower rows of widgets are visualizations of the indexed scores of the underlying indicators.

Labs Introduction

Lab 4
Indicators on Metrics

Metrics allow the storing of events related to operational records that reflect how those records are worked. For example, metrics can track different assignment groups assigned to an incident while it was open instead of only the currently assigned group (that is stored in the incident table). Often the information that is stored using metrics, is about durations. With metrics, the time that a record (an incident) has been in a particular state, assigned to a group, or assigned to a person is measured and stored in a separate table. Metrics are not limited to durations only. Any change of value of a field can be captured by a metric.

With metrics, there are 2 important elements. A **Metric Definition** defines what should be measured—for example Incident State Duration. Each definition is stored as a record in the metric_definition table. A **Metric Instance** is the occurrence of what is defined in a Metric Definition. For example, when the state of an incident changes, a record is created in the metric_instance table for the Incident State Duration definition. In the metric instance record the state and duration are captured.

Next to the tables mentioned above, there are out of box database views available. For each process for which metrics have been defined (change request, problem, incident, request, etc.) a database view joins the metric_definition, the metric_instance and the process table. This database view makes it easier to report on the metrics.

If the metrics feature is not available, activate the Metric Definition (com.glide.metrics) plugin which has been available since 2008. For more about metrics and how to create new metric definitions check the product documentation (Metric Definition Support).

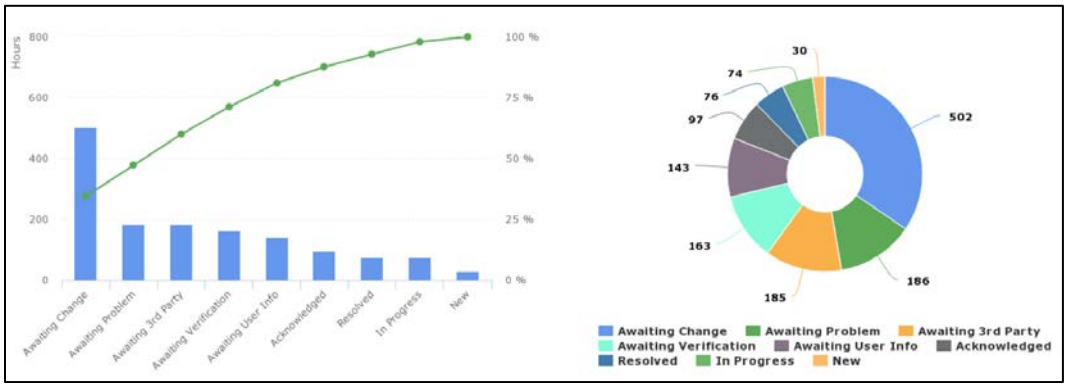
Be aware that next to metrics, the tables SLAs and SLA tasks are also available for (duration) reporting.

Performance Analytics Indicators on Metrics

The metrics feature can also be leveraged using Performance Analytics. Examples of indicators to be used on the metrics table are:

- Average duration of resolved incidents broken down by incident state
- Average duration of assignments of resolved incidents broken down by assignment group
- Number of new, open, closed assignments and backlog growth of assignments broken down by assignment group

Visualizations of the first indicator could look like the following example:



From the Pareto chart visualization above:

- ~33% of resolution time is spent on 'Awaiting change
- ~80% of resolution time is spent is on a waiting state

So here the breakdown scores add great value in identifying where the bottlenecks are in a process.

When indicators are needed in the context of a process record (like incident), use the database view to build the indicator. With the database view, attributes of the process record can be used as conditions: return all incident state duration records *of incidents resolved yesterday*.

When the context of a process record is not needed, the indicators can be built directly on the metrics table (metric_instance). For example, the average duration of assignments closed today. Or the number of new, open, closed and backlog growth of assignments. Combining the assignment indicators of the different process records (task types) into one visualization would give an overall view of the workload of assignment groups. Remember that questions, such as if assignments are responded to in time, are captured in the SLA tables and database views.

Metrics can keep track of durations on attributes for which the values change during the lifecycle of the process record. Most notably this works for the attributes state and assignment (groups). This information is not (easily) available in other tables.

Indicators based on the metrics table have a condition such as **metric definition is …**. The result of that is that only one breakdown that is based on a metric definition, can be applied. A breakdown matrix of assignments (durations) by assignment groups by incident state is technically not possible.

For indicators based on the metric database views, the fixed/non-changing attributes from, for example, incidents such as priority, category, contact type, closure code, could be combined with the assignment groups or incident state in a breakdown matrix. But, although technically possible, the second-level breakdown data does not provide useful insights or actionable information that will improve the process. So, in general do not use second-level breakdowns with metric indicators.

Lab Goal

Lab 4.1 Indicators on Metrics—Average Duration of Assignments

To gain better insight and easily identify bottlenecks in the incident process, the **Average duration of assignments broken down by assignment groups** is needed.

In this lab, the indicator is about the assignments related to incidents resolved on yesterday. That means the indicator source is on the metric database view. This includes assignments with an end date further in the past than yesterday!

In general, an incident record is resolved by changing the state to **resolved**, and then a business rule sets the resolved date. When the state changes on the incident record, the related metric instance record for the *incident state duration* is updated with an end date and a duration (and a new metric instance record is created for the state 'resolved'). But the related, open metric instance record for the *assignment group* remains open! Thus, for the indicator score, the duration for the last assignment group should only be calculated up to the resolved date of the incident.

Create an Indicator Source on the Metric Database View

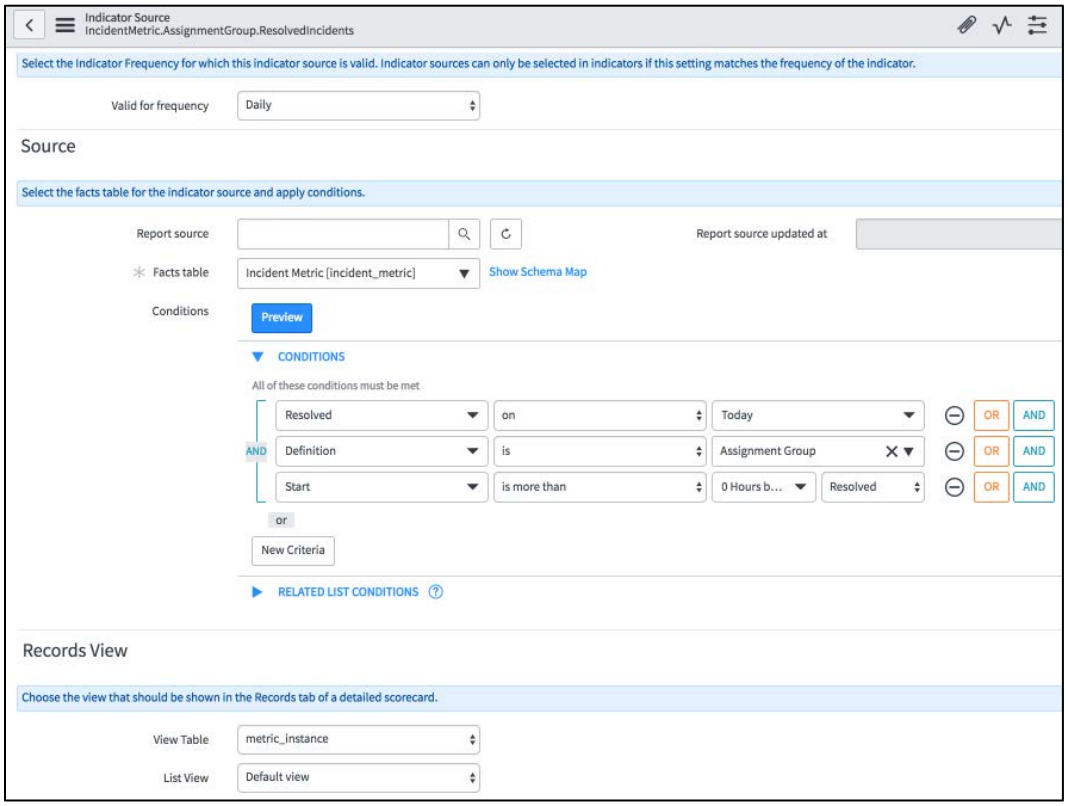
1. Create an indicator source (**Performance Analytics > Sources > Indicator Sources**) and update the following on the indicator source form:

   Name: **IncidentMetric.AssignmentGroup.ResolvedIncidents**
   Valid for frequency: **Daily**
   Source section:
   Facts table**: Incident Metric [incident_metric]**
   Conditions:

Resolved	**on**	**Today**,	
Definition	**is**	**Assignment Group**	
Start	**is more than**	**0 Hours before**	**Resolved**

   Records View section:

   View Table: **metric_instance**, (when records are collected, it is from this table)
   List View: **Default view**



2. **Submit** the new indicator source record.

Create a Script for the Duration Calculation

1. Create a new script (**Performance Analytics > Automation > Scripts**) and update the following on the new script form:

   Name: **IncidentMetric.Duration.Hours**
   Source section:
   Facts table: **Incident Metric [incident_metric]**
   Fields: **Start**, **End** and **Resolved**
   copy the following code to the **Script** field:

```
var diff=function(x,y){return y.dateNumericValue() - x.dateNumericValue();};
var hours=function(x,y){return diff(x,y)/(60*60*1000);};

var endDate = current.mi_end;
if (!current.mi_end && !current.inc_resolved_at)
        endDate = score_end;
else if (!current.mi_end && current.inc_resolved_at)
        endDate = current.inc_resolved_at;
else if (current.mi_end > current.inc_resolved_at)
        endDate = current.inc_resolved_at;
hours(current.mi_start, endDate);
```

   The script calculates the duration in hours between two dates. In this script the begin date, the x date, is always the **Start** of the metric_instance record. Used as the stop date for the duration calculation , the y date is
   - **End** date (from metric_instance), or
   - if **End** date and **Resolved** date (from incident) do not exist, the **Collection** date (score_end) is used *, or
   - if **End** date does not exist but **Resolved** date does, the **Resolved** date is used, or
   - if **End** date > **Resolved** date, the **Resolved** date is used.

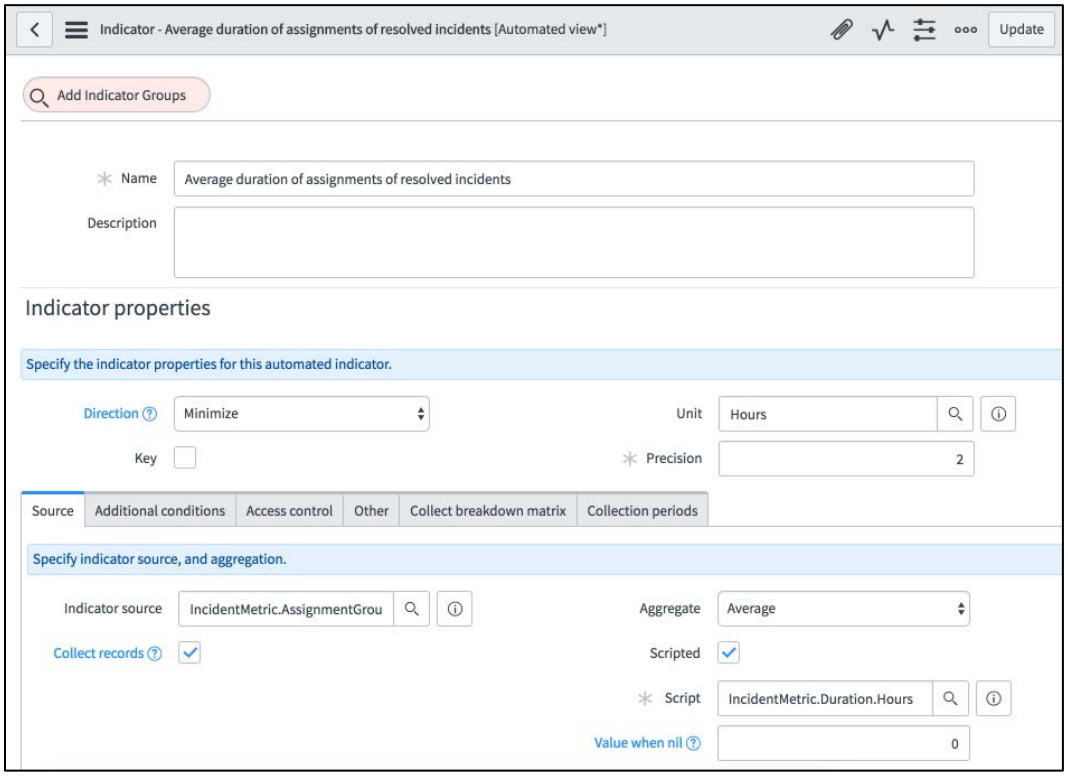
   * This makes the script generic so it can also be used for open process records.

2. **Submit** the new script record.

Create a New Indicator

1. Create a new indicator (**Performance Analytics > Indicators > Automated Indicators**) and update the following on the new indicator form:

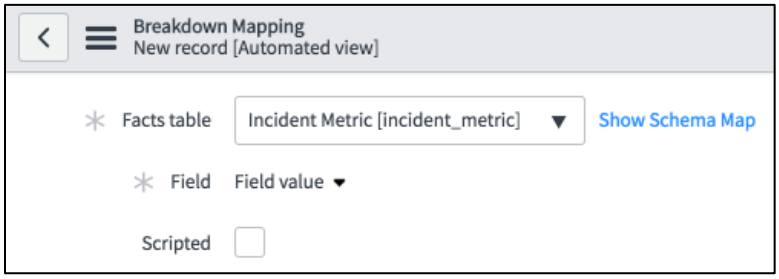
   Name: **Average duration of assignments of resolved incidents**
   Direction**: Minimize**
   Unit**: Hours**
   Precision**: 2**
   Source tab:
   Indicator source: **IncidentMetric.AssignmentGroup.ResolvedIncidents**
   Collect records check box: **selected**
   Aggregate: **Average**
   Scripted check box: **selected**
   Script: **IncidentMetric.Duration.Hours**
   Value when nil: **0**



   Access control tab: select **Publish on scorecards**
   Other tab: deselect **Show delta**. The delta record sets never have a **common** section since the indicator source holds a condition **[incidents] Resolved on Today**.

2. **Submit** the new record.

3. Create a new breakdown mapping to the **incident_metric** view for the existing **Assignment group** breakdown. The mapping is on the **Field value** column.



   By default, the **Field value** column is not on the form or in list views as it only holds Sys IDs.

4. Relate the **Assignment group** breakdown to the new automated indicator **Average duration of assignments of resolved incidents**.

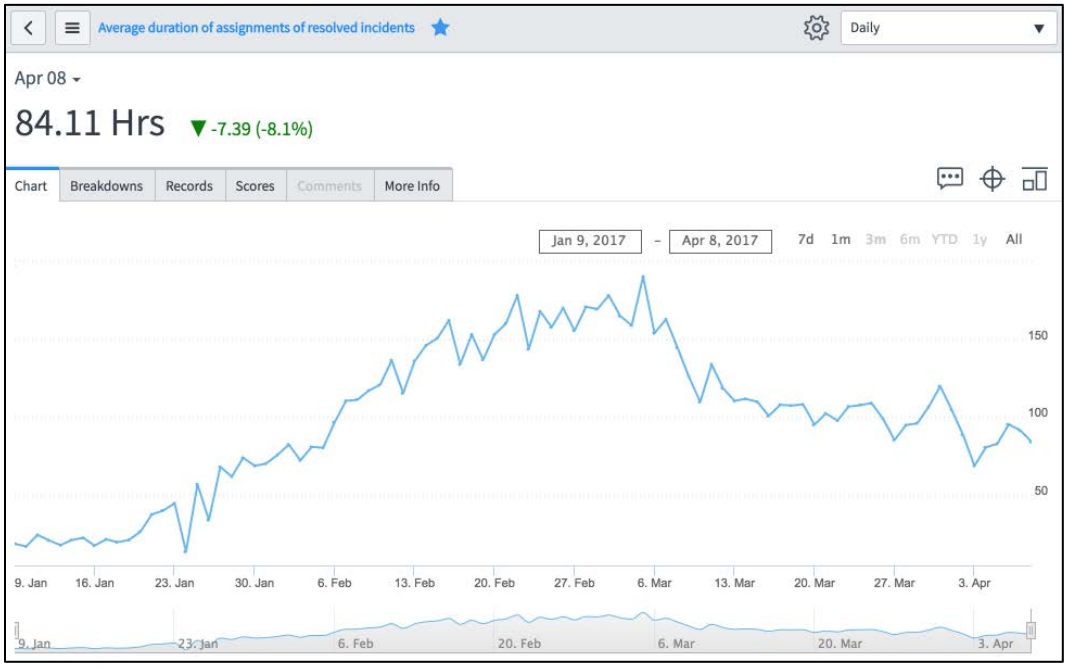
Lab Success Verification

1. Run a historic scores collection for the **Average duration of assignments of resolved incidents** indicator for the last 3 months. When the collection finishes, the collection log looks like:

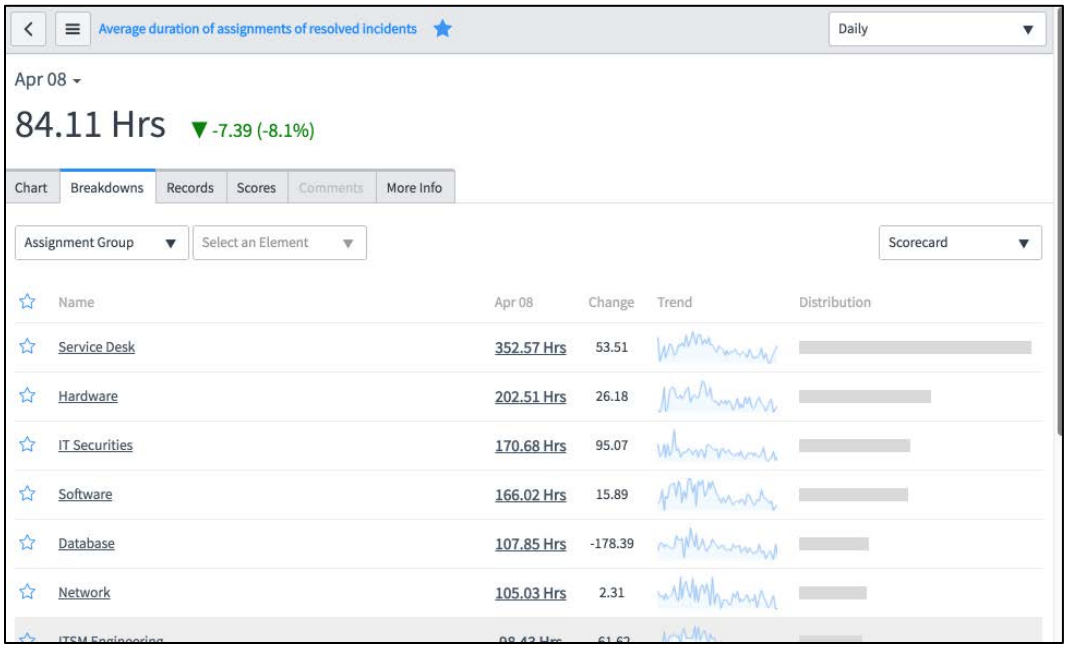
   Inserts: **2,454**
   Warnings: **0**
   Errors: **0**

   (The number of inserts may differ slightly as it depends on the actual number of days in the last 3 months; but the number of warnings and errors should be 0.)

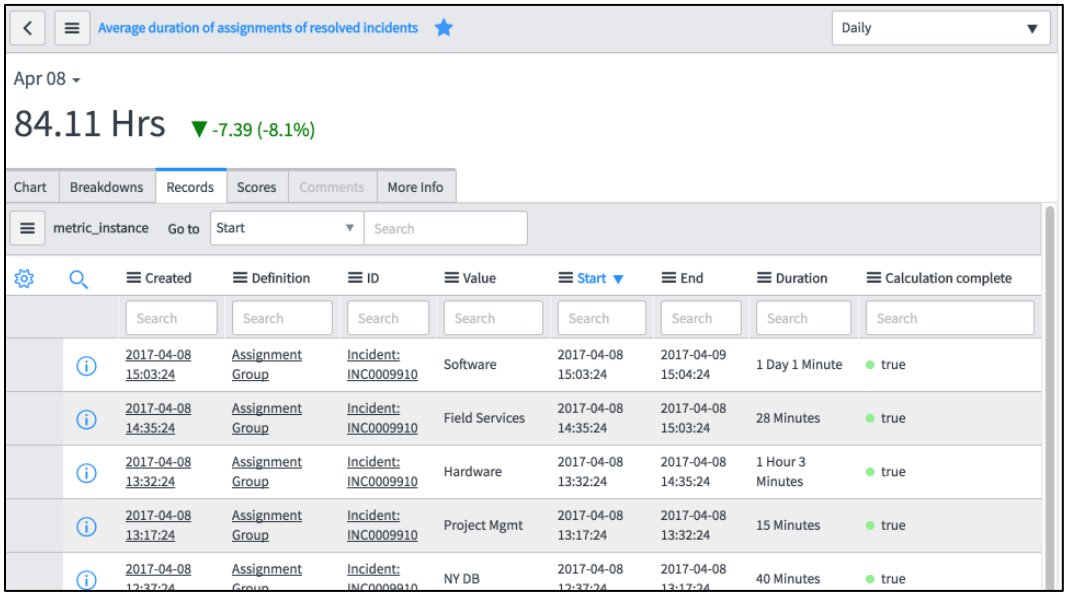
2. Check the detailed scorecard of the **Average duration of assignments of resolved incidents** indicator.



3. Notice that scores have collected also for the (one) breakdown **Assignment Group**.



Also, records have been collected:



These are the metric_instances records, set to be collected in the indicator source.

Lab Goal

Lab 4.2 Indicators on Metrics—Average Unassigned Duration

To identify if there is an opportunity to shorten the throughput time of incidents, an indicator is needed that tracks how long incidents are not assigned (i.e., do not have an assigned to person). The process as conducted in this ServiceNow instance, is that an incident always has an **assignment group**, but not an **assigned to person**. When assigned to an assignment group, a dispatcher within each group assigns the incident to a person. If an incident does not have an assigned person, nobody works on it.

To be able to collect scores about the **Average Unassigned Duration**, first a metric definition is needed that creates a metric instance when an incident is assigned to a group. The metric instance should be updated with an end date when the incident is:

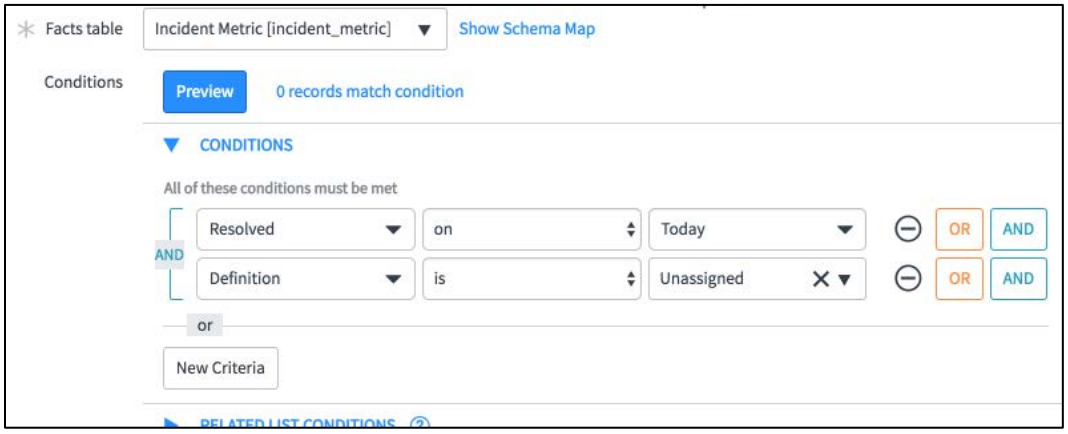
- Dispatched within the group (the incident gets an **assigned to**), or
- Re-assigned to another group because it was assigned to the wrong one, or
- Resolved.

In this lab, the metric definition (**Unassigned duration incident assignment**), and instances based on this definition, are already available. Thus, the indicator **Average Unassigned Duration** must be created. The indicator scores should also be broken down by assignment group. For now, the analysis is focused on resolved incidents only.

Create an Indicator Source on the Metric Database View

1. The indicator source that is needed, is like the one created in the previous lab. Open the existing indicator source **IncidentMetric.AssignmentGroup.ResolvedIncidents** (**Performance Analytics > Sources > Indicator Sources**) and update the following on the indicator source form:

   Name: **IncidentMetric.Unassigned.ResolvedIncidents**
   Source section:
   Conditions:
   Definition is Unassigned
   Remove the condition **Start is more than o Hours before Resolved**.



2. From the context menu select **Insert** to save the updates as a new record.

Create a Script for the Duration Calculation

In this use case, with the way the metric is set up, the end date for the open metric instance is also set when the incident is resolved. If an incident is on a resolved or closed state and is reassigned to another group, no metric instance is generated. The script needed for this indicator is different from the one in the previous lab.

1. Create a new script (**Performance Analytics > Automation > Scripts**) and update the following on the new script form:

   Name: **IncidentMetric.UnassignedDuration.Hours**
   Source section:
   set the Facts table to **Incident Metric [incident_metric]**
   set the Fields to **Start** and **End**
   Copy the following code to the **Script** field:

```
var diff=function(x,y){return y.dateNumericValue() - x.dateNumericValue();};
var hours=function(x,y){return diff(x,y)/(60*60*1000);};

var endDate = current.mi_end;
if (!current.mi_end)
        endDate = score_end;
hours(current.mi  start, endDate);
```

   If the End date for a metric instance does not exist because the incident is still open, the **Collection** date (score_end) is used. This makes the script fit to be used for open process records as well.

2. **Submit** the new script record.

Create a New Automated Indicator

1. Create a new indicator (**Performance Analytics > Indicators > Automated Indicators**) and update the following on the new indicator form:

   - the **Name** to **Average unassigned duration of resolved incidents**,
   - the **Direction** to **Minimize**,
   - the **Unit** to **Hours**,
   - the **Precision** to **2**,

   On the **Source** tab:
   - set the **Indicator source** to **IncidentMetric.Unassigned.ResolvedIncidents**,
   - check the **Collect records** checkbox,
   - set the **Aggregate** to **Average**,
   - check the **Scripted** checkbox,
   - set the **Script** to **IncidentMetric.UnassignedDuration.Hours**,
   - the **Value** when nil to **0**:

   On the **Access control** tab,
   - check the Publish on scorecards checkbox.

2. From the context menu **Save** the new record.

3. Scroll down and relate the **Assignment group** breakdown to the new indicator.

Create a New Formula Indicator

Combining the automated indicator created above with the one in the previous lab gives the percentage of resolution time that incidents are not assigned to a person. When broken down to assignment group, that insight is shown at a more granular level as well.

1. Create a new indicator (**Performance Analytics > Indicators > Formula Indicators**) and update the following on the new indicator form:

   Name: **% of resolution time incidents are not assigned**
   Direction: **Minimize**
   Unit: **%**
   Precision: **2**
   Key check box: **Selected**
   Formula box:
   [[Average unassigned duration of resolved incidents / 7d running AVG]] /
   [[Average duration of assignments of resolved incidents / 7d running AVG]] * 100
   Access control tab:
   Publish on scorecards check box: **Selected**

2. From the context menu **Save** the new record.

3. Scroll down and relate the **Assignment group** breakdown to the new indicator.

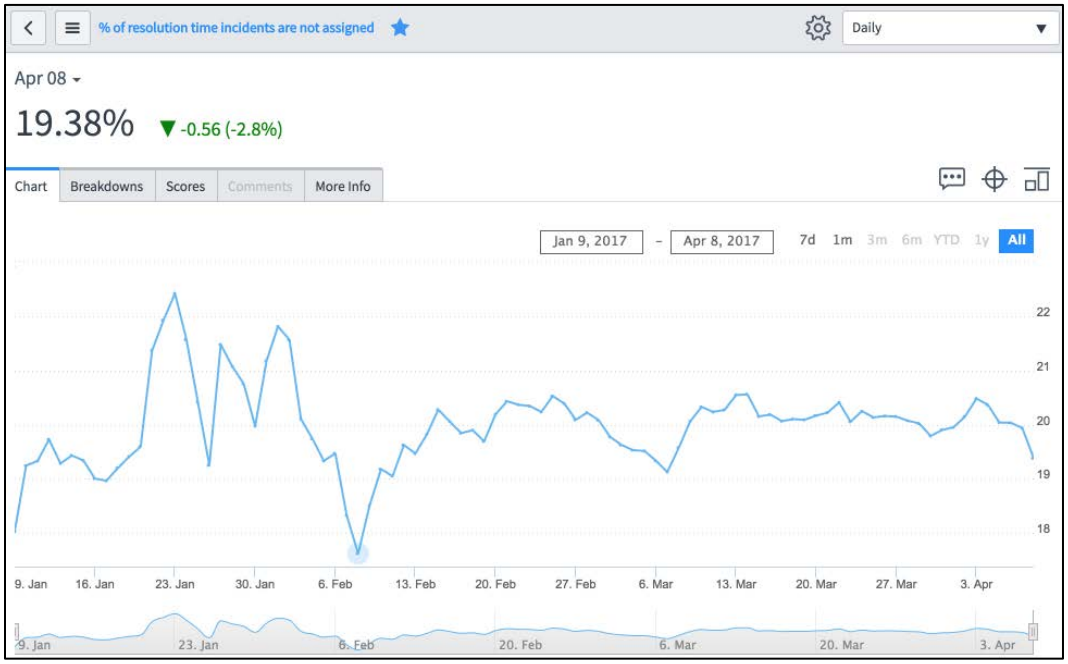
Lab Success Verification

1. Run a historic scores collection for the **Average unassigned duration of resolved incidents** indicator for the last 3 months. When the collection finishes, the collection log look: like:

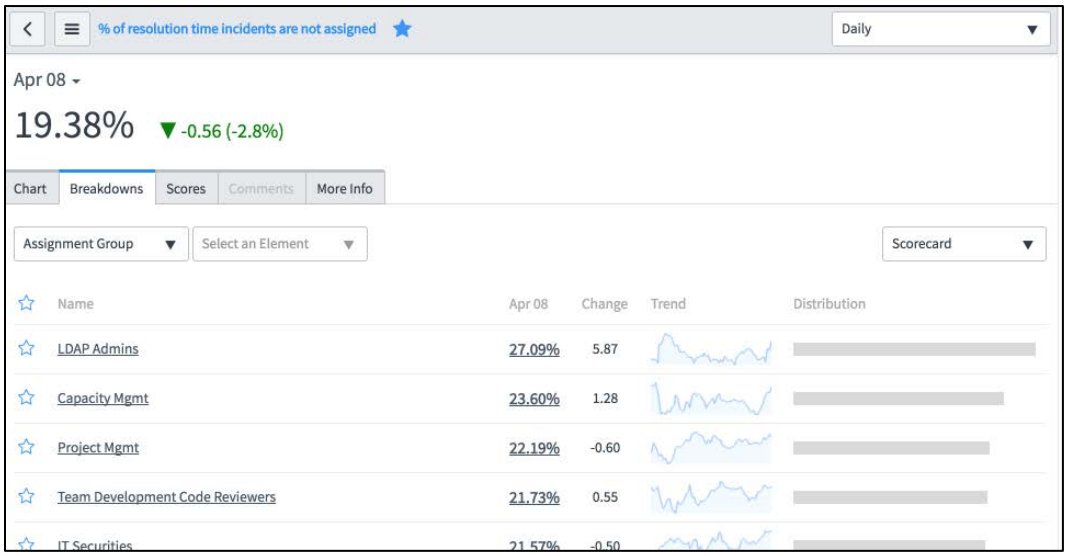
   Inserts: **2,454**
   Warnings: **0**
   Errors: **0**
   (The number of inserts might slightly differ as it depends on the actual number of days in the last 3 months, but the number of warnings and errors should be 0.)

   So the exact same number of inserts as for the **Average duration of assignments of resolved incidents** in the previous lab!

2. With the scores collection, not only does the detailed scorecard become available for the **Average unassigned duration of resolved incidents** indicator, but also for the **% of resolution time incidents are not assigned**:



Notice that scores will also be calculated for the (one) breakdown **Assignment Group**:



Lab Goal

Lab 4.3 Indicators on Metrics—Incident Assignments Workload

To see where the workload is and how it is developing over time, **New**, **Closed**, and **Open incident assignments** broken down by assignment groups are needed.

In this lab, the indicators are about the assignments only; thus the indicator sources are on the metric_instances table.

As always when developing new indicators, breakdowns, and scripts, check what comes out of box that could serve as examples on how to create new configuration records.

In this lab, the assignments are restricted to incident assignments only. This can easily be extended with assignments of other task types as well. That can be done in two ways: as separate indicators for each task type, or when the indicator source includes all task types (i.e., all relevant metric definitions), as a breakdown on task type.

Create Three Indicator Sources on the Metric Instances Table

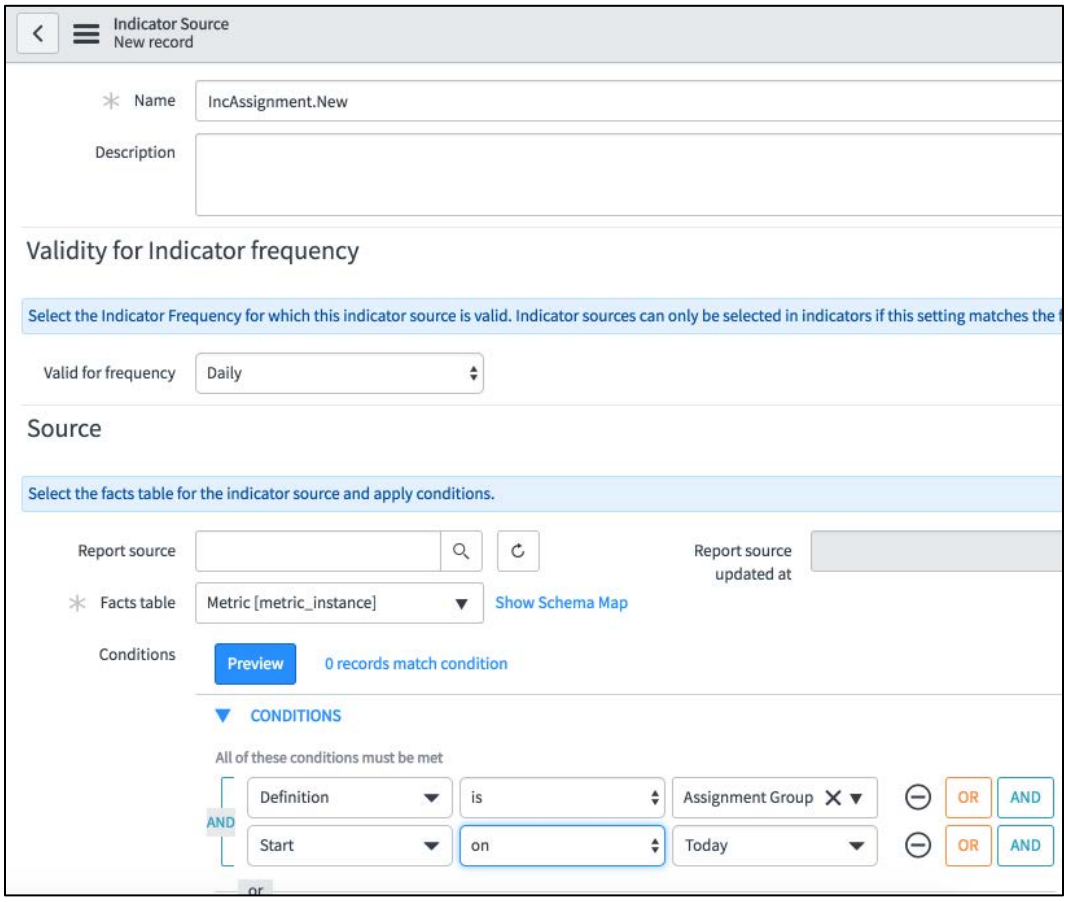
1. Create 3 new indicator sources **IncAssignment.New**, **IncAssignment.Open** and **IncAssignment.Closed**.

   The new indicator sources are similar to the existing sources **Incident.New**, **Incident.Open** and **Incident.Closed**. Use these as examples.

   For the three new indicator sources:

   Table: **metric_instance**
   Conditions:
   Definition: **Assignment Group**
   Use **Start** and **End** as dates

2. For **IncAssignment.New**, the indicator source looks like the example.

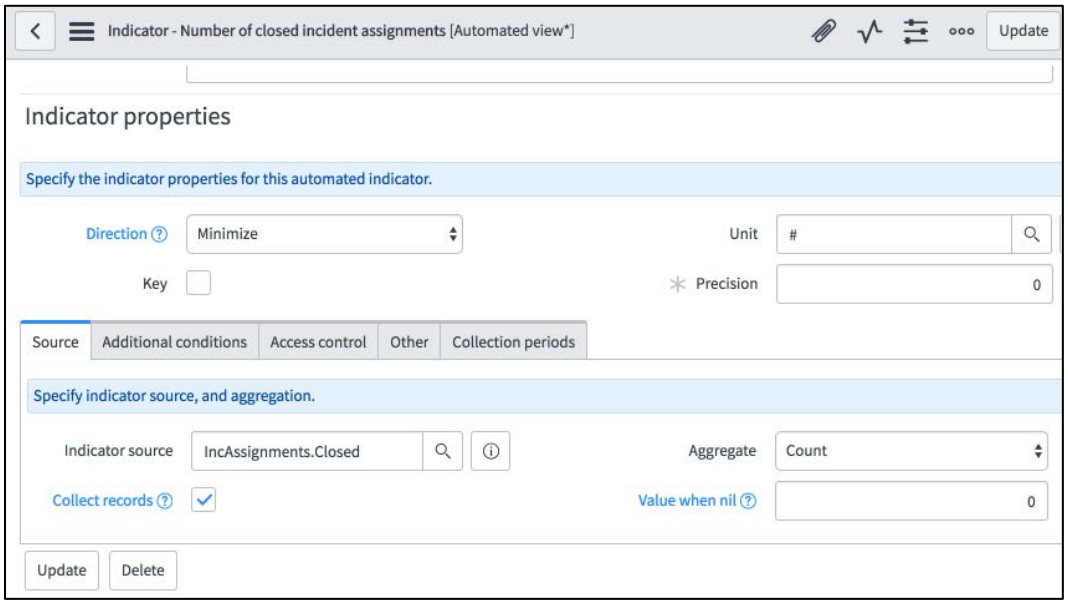


Create Three Indicators

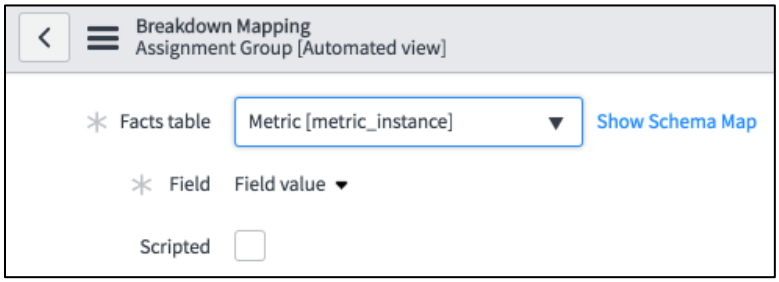
1. Create 3 new automated indicators: **Number of new incident assignments**, **Number of open incident assignments** and **Number of closed incident assignments**. Each new indicator uses the **Count** aggregate.

   Again, use the existing similar indicators **Number of new incidents**, **Number of open incidents**, and **Number of resolved incidents** as examples.

2. For **Number of closed assignments**, the indicator looks like the example.



3. Create a new breakdown mapping to the **metric_instance** table for the existing **Assignment group** breakdown. The mapping is on the **Field value** column:



By default, the **Field value** column is not on the form or in list views as it only holds Sys IDs.

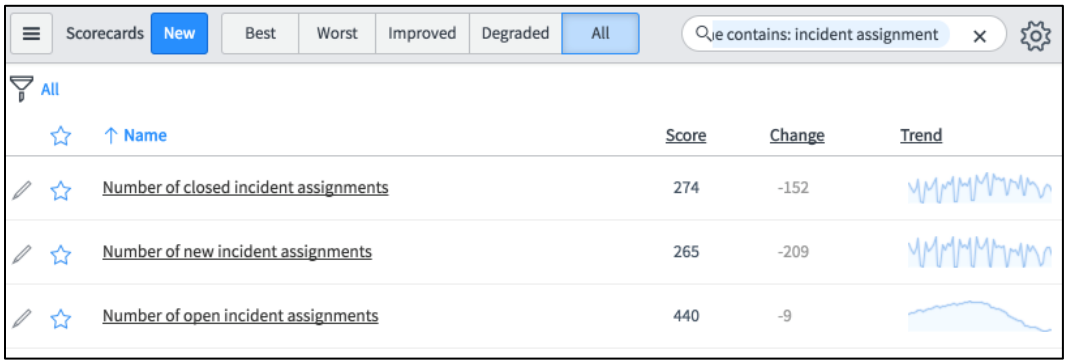
4. Relate the **Assignment group** breakdown to the three new automated indicators.

Lab Success Verification

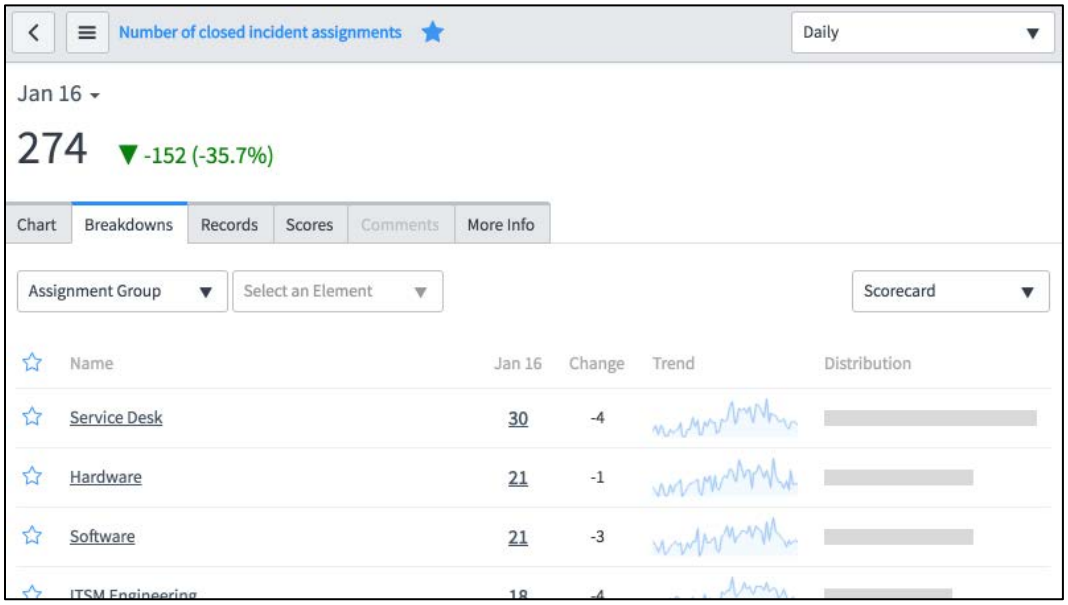
1. Run a historic scores collection for the three new incident assignments indicators for the last 3 months. When the collection finishes, the collection log looks like:

   Inserts: **5,517**
   Warnings: **0**
   Errors: **0**

2. Check the scorecards of the three indicators.



3. Notice that scores have collected also for the **Assignment Group** breakdowns.



With the scores collected and available, different type of visualizations can be created and added to a dashboard. That includes specific **breakdown** widgets and **follow element** widgets to be used on a breakdown dashboard.