IDERA® | SQL Server Whitepaper

DETECT BASELINE ANOMALIES

BY SCOTT STONE

INTRODUCTION

In the world of database performance, normal is a highly prized goal. It means nothing extraordinary is happening that requires intervention or diagnosis. It implies no fire-fighting or frantic finger-pointing exercises across functions in the IT department. So, in this case, healthy is the goal. Typical performance means normal service levels and minimal drama from dissatisfied users and customers. However, to be measured and monitored, normal must first be defined and measured. That is sometimes more easily said than done.

If you have attended any best practice presentations or courses for database administrators over the years, the presented undoubtedly told that the first step in measuring and monitoring performance is to establish baselines. The trick is that different people throw that term around without definition while meaning very different things. So I want to discuss some of those meanings and then explain what it means in the context of monitoring tools. Moreover, even in that isolated context of monitoring tools, you need to understand how different vendors use the same term.

WHAT ARE BASELINES

So, what do consultants mean when they advise you to take a baseline as the first step to monitoring and improving performance? At its most basic definition, a baseline is any starting measurement you chose to evaluate future changes against the same. It does not have to be representative. You do not need to take it over time. However, if you want it to be meaningful, then you want to do more than making a single measurement and call it the baseline. You want to take it at an appropriate time when you want to compare future measurements. You also want to make multiple measures to make sure you do not have an outlier as the baseline.

The best way to accomplish this task is to take many measurements and evaluate those measurements statistically to determine a normal operating range for the system. Of course, you need to make sure you take these measurements when the overall performance is acceptable to you. You do not want the normal to be abnormal from the perspective of the users.

DETECT ANOMALIES

Every performance monitoring product is ultimately seeking to provide anomaly detection. However, it can mean an anomaly from different things. The most basic monitoring allows you to detect a deviation from some best-practice default settings for performance metrics. The monitoring identifies the performance metrics as leading indicators of overall system performance. Moreover, the key is to detect problems early before they start affecting user performance or impact service level agreements.

The downside to this sort of essential anomaly detection is balancing between the danger of missing problems (that is, false negatives) and the overhead of managing too many false positive alerts. False positives mean the alerts are overly cautious and generating warnings when no real underlying problem exists to solve.

Experienced database administrators who have used monitoring systems over many years develop their own best practice settings for alerts based on:

- Personal tolerance for risk of missing a real problem
- Knowledge of the behavior of different sorts of applications and particular instance performance

- The criticality of the specific application or system to the business itself.
- Tuning alert thresholds in this manner can be more of an art than a science.

Moreover, even experienced database administrators may not have complete familiarity with every instance under their care to know the right balance to strike immediately. They need to watch performance over time and manipulate thresholds gradually. They usually establish a template of limits for a particular set of instances based on different combinations of the three criteria listed.

When manipulating pure thresholds is insufficient, the database administrator may resort to some filtering or alert suppression such as filtering out temporary spikes.

It is also possible to filter out alerts on particular databases or files or disks that may be unimportant to the database administrator for one reason or another.

However, ideally, we would like to auto-calibrate these threshold tuning adjustments as much as possible to minimize the need for extreme experience with every application and instance behavior and the trial-and-error process for setting and adjusting these threshold template settings.

BASELINE APPROACHES

Many monitoring products claim to automate the baseline measurement function. However, not all of them are referencing the same thing with the term. They all either plot a reference line or a range of values and leave it to you to assign a meaning to that range. I hope to correct that to some degree with this whitepaper.

The title of this whitepaper purposely references the desire to measure and monitor normal behavior in a database instance. However, just because you see a metric plotted against a range of values does not necessarily mean the range represents what is typical for the database instance.

WHAT IS NORMAL VERSUS WHAT IS EXTREME

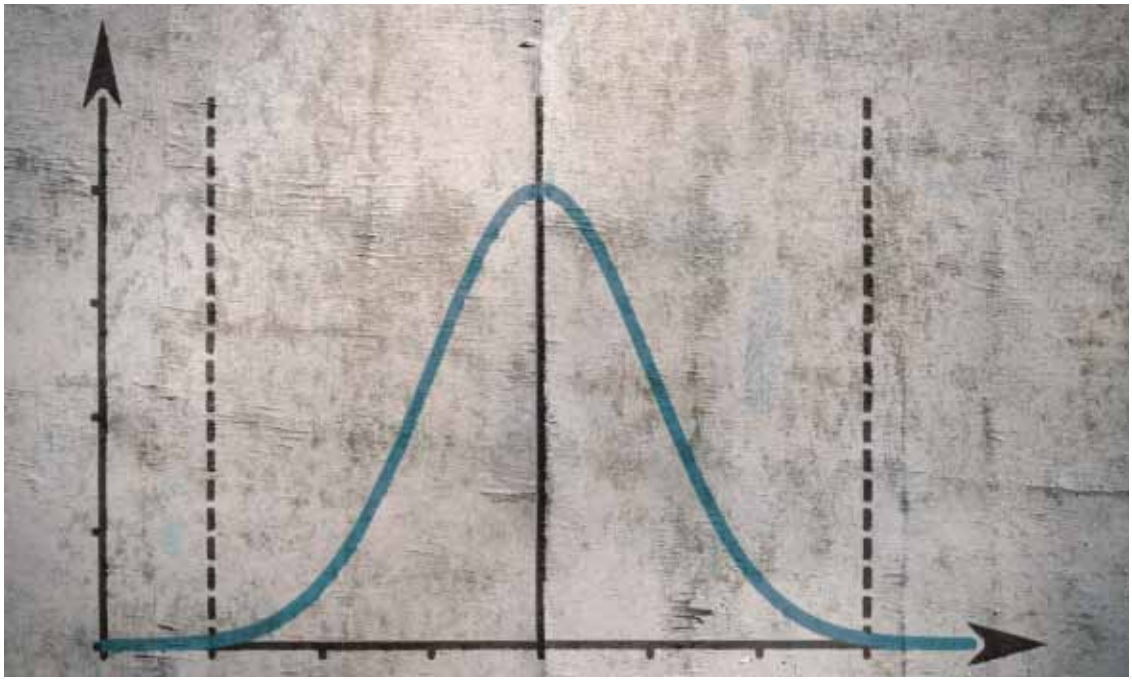
The easiest way to determine a range of values for comparison to the current value is to use minimum and maximum values. If you save minimums and maximums over some time, then you can use it to construct a range for a particular time of day or day of the week. It looks cool to plot this changing range over some time and compare it to the current readings.

However, what looks good on a graph can be counterproductive if used for managing a system or used as an alerting threshold. In effect, you are allowing the points of comparison to be determined exclusively by outliers. The monitoring treats a metric with a minimal variance but occasional wild spikes the same as an extremely noisy metric.

You do not need much of a background in statistics to recognize that using historical outlier values as an approximation of normal behavior for a metric is misguided at best and misleading at worst.

DOES NORMAL MEAN GAUSSIAN

So, we need a better representation of the average variance of any given metric than just keeping a history of maximum or minimum values. The Gaussian or normal distribution is often used in natural and social sciences to represent random variables whose distributions are not known. The reason is that, given the right restrictions, experimentation usually proves out this assumption.



Anyone who has attended school is familiar with the old bell curve regardless if you know it by its other names. (I was always amazed that a histogram of the grades of any sufficiently large class, at any level, seemed to conform to this familiar shape. The central limit theorem works!) As a refresher, the fundamental values here are the mean and the standard deviation. The standard deviation determines the shape of the curve. The mean determines the position of the curve. Using only the upper threshold of the distribution (we usually only care about performance metrics if they are abnormally bad, not unusually good), we can estimate that 84% of the time the parameter should stay under the mean plus one standard deviation and 97.5% it stay under two standard deviations.

So, does the distribution apply to the behavior of database or system performance metrics over time? The answer is that it does not need to apply directly to the probability distribution to be useful. It only needs to apply to the error or noise component of the metric when predicting future behavior. If there is a long term trend or cyclical nature to the value, then after removing those factors, you are left with a normally distributed variable.

In a time-series metric, as we use for performance monitoring, we can correct for trends by using a moving window of time for our calculation population. Moreover, we can fix for cyclicality by using different calculations for times when we observe different performance because of changing the load on the database or system.

USE CALCULATED BASELINES

For specific metrics, you can use these assumptions about the mean and standard deviation to calculate more useful benchmarks. These help you calibrate the alert thresholds to the behavior of the particular environment, or you can use them directly for the alerts.

WHAT DO DEFAULT BASELINES MEAN?

The default baseline period is whatever you determine are the times that you define as the relevant periods for normal operation. Or rather, it is a way to exclude the periods of collections that are less relevant for your mean and standard deviation calculations. That would be during times of maintenance or relative inactivity for example. It is a way to determine a default baseline when you have no more specific baseline period. It would usually be comprised of a typical work week for business systems and working hours for each day.

The user can add as many named baseline calculations to the default performance baseline calculation as desired. The default calculation period can also overlap with any other baseline period. The default calculation period can also overlap with any other baseline period. The default baseline is used uniquely in a few ways:

- It is used to calculate alert recommendations. When the normal variation of the performance metrics varies significantly from the selected thresholds, then you receive an advisory to adjust those thresholds. The suggested limits refer to the default mean and standard deviation values.

- It is always in effect whenever the user does not schedule another specific baseline.

You can use a dynamic baseline on the past seven days of data. Alternatively, you can specifya particular period if you want more review control before changing alert thresholds or wish to preserve a specific high point of activity for threshold settings.

ALERT RECOMMENDATIONS

One way to use baselines effectively is to periodically, but manually, adjust the alert thresholds after reviewing recommended settings. The recommended changes always refer to some set percentage of the default baseline, which is the mean plus one standard deviation. Only the alerts that normally exceed current thresholds generate alert recommendations. Users can accept all recommendations or choose certain ones to change. Recommendations are set to default at 20% and 30% above the calculated baseline. However, those levels can be configured as well if the user is more concerned about false positive alerts or false negative alerts.

DIFFERENT BASELINES FOR DIFFERENT ACTIVITY LEVELS

The default baseline and alert recommendations are an excellent way to tune static thresholds based on measured performance of particular databases or systems. However, what if you want to be alerted differently for abnormal performance during specific periods of the day or week?

VISUALIZING BASELINES

The first step in setting up baseline periods is to decide what days and times to use for those periods. You may already know what you want because you know when batch jobs run or when

peak online activity is in effect or conversely when the database or system should be relatively idle. If so, you can directly set those times as described later.

However, if you need a visual guide to spot any abrupt and recurring changes in activity level, visualize baselines in charts to overlay the week-over-week measurements of any selected metric. You can also use this tool as a general comparison for the week-over-week performance of any given parameter. The critical thing to look for on such a chart is recurring clusters of activity at different times of the day or week.

SET UP MULTIPLE BASELINES

Now that you know which times of the day and days of the week you want to use for the names baselines, it is relatively simple to set them up. The process for defining a named baseline is:

1. Decide whether to use a specific date range or a dynamic seven-day baseline. You can make different choices for different benchmarks.

2. Choose the time range and days of the week to select the data to be used in the calculation.

3. Decide when this baseline be in effect. Usually, this is identical to the calculation days and time range. However, you may choose to limit the calculation to specific peak periods and yet schedule it for a broader scale.

For example, separate baseline calculations for batch processing and lunch hours based on dynamic seven-day calculations and a set date range for a close week baseline for end of the quarter activity that might be different from the dynamic calculation.

SET THRESHOLDS DIRECTLY FROM BASELINES

If you do not want to keep getting alert recommendations and reacting to them, directly set alert thresholds based on baseline calculations. Automatically change the threshold as the data within a particular baseline changes. However, you will also want to schedule changes for which dynamic baseline is in effect.

GLOBAL BASELINE SETTINGS

While these settings should help users calibrate alerts to the typical performance of a given system and help shorten the process of tuning alert thresholds, it could be tedious to set it up separately for each instance managed.

We have encountered some vendors calling out a feature of global baselines to mean calculating baselines on one instance and then using those exact static thresholds on multiple other instances. We would not call that functionality a global baseline, although users can already do the same thing using alert templates. Using alert recommendations to adjust a given alert template, the user can choose to apply that template with identical settings to as many instances or groups of instances by tag as they prefer.

Instead, it makes more sense, and it better fits the definition of global baseline to set up the baseline calculation periods and definitions once but apply them to other instances and recalculate separate benchmarks for each instance based on its own collected history. Then, apply the baseline configuration from one instance to other instances. Finally, select which instances or servers you wish to set with an identical baseline configuration. You can edit the baseline configuration of any instance directly after the initial setup is applied. This way, you can set up model instances for baseline configurations and then use the same settings to many other instances.

FUTURE CONSIDERATIONS

Look for baseline enhancements in the future. For example, considering the following baseline enhancements:

1. Dynamic baselines automatically updated on an hourly basis
2. Dedicated baseline visualization screen with each metric plotted against its baseline range, and
3. Separate baseline alerts to permit simultaneous usage of static alerts for best practice failsafe settings in addition to detecting deviations from normal behavior

ABOUT THE AUTHOR

Scott manages IDERA's database performance management products. He has over twenty years of experience in product management and product marketing in the software and technology industry from small start-ups to Fortune 500 companies. For the past fifteen years, Scott focused on database performance and security products at various companies. Earlier in his career, he was a software engineer in the space and defense industry. Scott holds an MBA from Rice University as well as bachelor's and master's degrees in electrical engineering from the Georgia Institute of Technology.