# Table of Contents

**Introduction** .................................................................................................................................................. 12
  Introduction to DPA ........................................................................................................................................ 13
  DPA architecture ............................................................................................................................................... 14
    Two key functions of the DPA server ................................................................................................................. 15
    Monitored database instances ........................................................................................................................... 15
    Monitored virtualization environment ............................................................................................................. 15

**DPA licensing** ................................................................................................................................................. 16
  License types .................................................................................................................................................... 16
    Individual licenses .......................................................................................................................................... 16
      All individual licenses are floating ................................................................................................................ 17
    Clustered environments ................................................................................................................................. 17
  Subscription licensing ...................................................................................................................................... 17
  Learn more ...................................................................................................................................................... 18

Registration and licensing options for clustered environments ........................................................................... 18
  SQL Server AGs ............................................................................................................................................... 18
    Register each SQL Server instance in the cluster ............................................................................................ 18
    Register the AG listener .................................................................................................................................. 18
  Oracle RACs .................................................................................................................................................... 19
  Learn more ...................................................................................................................................................... 19

Requirements for monitoring a database instance running in a VM cluster ......................................................... 20
  Create a user on the vCenter Server or the ESX/ESXi host ............................................................................... 20
  Assign user permissions to inventory objects .................................................................................................. 20

Purchase and view licenses .................................................................................................................................. 20
  Purchase licenses ............................................................................................................................................. 21
  View purchased licenses ................................................................................................................................ 21
  Next steps ....................................................................................................................................................... 21

Activate DPA licenses ........................................................................................................................................ 21
  Activate licenses online ................................................................................................................................... 21
Activate licenses offline ................................................................. 22
Next steps ..................................................................................... 23
Allocate or deallocate DPA licenses .................................................. 23
Automatic license allocation ............................................................. 23
Manually allocate licenses to database instances ............................... 24
View current license allocation ....................................................... 24
Allocate licenses to database instances .......................................... 24
Allocate VM licenses to VM database instances ............................... 24
Deallocate licenses ......................................................................... 24
Learn more ..................................................................................... 25
Troubleshoot over-allocated licenses ............................................... 25
Deactivate your DPA licenses ........................................................... 25
Deactivate online ........................................................................... 26
Deactivate offline ........................................................................... 26

Register a database instance for monitoring .................................... 27
Register multiple database instances ................................................. 27
Register a single database instance .................................................. 27
Self-Managed .................................................................................... 27
Amazon RDS ..................................................................................... 27
Azure .................................................................................................. 27
Database instances DPA can monitor ............................................... 27
Self-managed databases ................................................................... 28
Amazon RDS databases ................................................................... 29
Key differences for Oracle databases on Amazon RDS ...................... 29
Key differences for SQL Server databases on Amazon RDS ............... 29
About repointing database instances ............................................... 30
Azure databases ............................................................................... 30
Key differences between self-managed SQL Server and Azure SQL database instances .......................... 31
About repointing database instances ............................................... 32
Register multiple database instances ................................................. 32
Show or hide groups on the DPA homepage .......................................................... 67

**Monitor database instances with DPA** ................................................................. 68

Update a monitored database instance .................................................................. 69
Stop monitoring a database instance for a period of time ...................................... 69
Troubleshooting tips ................................................................................................. 70
Logs .......................................................................................................................... 70
Access log data through the DPA log viewer ......................................................... 70
Open log files in a text editor .................................................................................... 70
Access to a database instance .................................................................................. 70
Issues after the Oracle PDB that stores the repository is moved ......................... 71
DPA returns a connection error ................................................................................. 71
DPA returns an invalid login error ............................................................................. 71

**Investigate performance issues with DPA** ......................................................... 72

Access DPA query or table tuning advisors ............................................................. 72
View all advisors for a database instance ............................................................... 72
Open an advisor ......................................................................................................... 73

Use DPA’s query performance analysis to find the root cause of performance issues .......... 73
Open the Query Details page .................................................................................... 73
Select a time period .................................................................................................. 74
See what type of waits are affecting performance .................................................. 74
Review query and table tuning advisors .................................................................. 75
Correlate query wait times with other events .......................................................... 76
Display other statistics charts ................................................................................. 77
Display other metrics charts ..................................................................................... 77

Investigate inefficient queries running against a table ........................................... 78
What are table tuning advisors? ............................................................................... 78
Open a table tuning advisor ...................................................................................... 79
Quick start ................................................................................................................ 79

Examine the list of inefficient queries ...................................................................... 80
Tips for using this information ................................................................................ 81
Examine query details ........................................................................................................... 81
  Tips for using this information ......................................................................................... 82
  Warnings ............................................................................................................................ 82
Examine table statistics ....................................................................................................... 83
  Tips for using this information ......................................................................................... 83
Examine index details ......................................................................................................... 83
  Tips for using this information ......................................................................................... 83
  Correcting common index problems ................................................................................. 84
  Indexing trade-offs ........................................................................................................... 85
Identify blocking sessions and deadlocks ........................................................................ 85
Identify blockers causing the longest waits ....................................................................... 85
Find the last activity of an idle blocker ............................................................................. 86
Investigate deadlocks on SQL Server instances ............................................................... 88
Find and investigate unusually long wait times (anomalies) ........................................... 90
  Get notified when wait time is higher than expected ..................................................... 91
  View information about wait time anomalies ............................................................... 91
    Wait time meter ............................................................................................................ 91
    Anomaly Detection chart (30-day period) .................................................................. 91
    Anomaly Detection chart (one-day period) ................................................................. 93
Investigate higher-than-expected wait times ................................................................... 93
  Determine when anomalies occurred ............................................................................ 93
  Display historic wait times and performance analysis for these queries .................... 94
  Use DPA reports to review wait or query details ........................................................... 95
Investigate lower-than-expected wait times ................................................................... 97
About anomaly detection in DPA ..................................................................................... 97
  How does DPA's anomaly detection work? ................................................................... 97
  How DPA determines the status of an incomplete hour .............................................. 98
  SQL statements excluded from the Trends chart ......................................................... 99
  Does anomaly detection work well for all database instances? .................................. 99
Large gaps in the learning data ......................................................................................... 99
# Anomaly Detection in SolarWinds Database Performance Analyzer (DPA)

- Specify the learning date after the load on a database instance changes ........................................... 100
- Disable anomaly detection for a database instance .............................................................................. 100
- Add an annotation to document a change to the database ................................................................. 100
- Name SQL statements ......................................................................................................................... 102

## Resource metrics in DPA ................................................................. 104

- View resource metrics in DPA ........................................................................................................... 104
  - View all available metrics on the Resources tab .............................................................................. 104
  - Correlate wait time with resource metrics on the Trends tab ...................................................... 104
  - View resource metrics related to the performance of a query .................................................. 105
- About resource metric baselines ....................................................................................................... 105
  - Show or hide baselines ................................................................................................................. 105
  - How are baselines calculated? ..................................................................................................... 106
  - Change the days included in baseline calculations ...................................................................... 106
- View or change resource metric thresholds ...................................................................................... 106
  - View the current thresholds ........................................................................................................ 106
  - Change the thresholds ................................................................................................................ 107

## DPA user accounts ........................................................................ 108

- DPA roles and privileges ..................................................................................................................... 108
  - Administrator role ....................................................................................................................... 108
  - Read Only on All Instances role .............................................................................................. 108
  - Custom Privileges role ............................................................................................................... 109
- Create a DPA user account and assign privileges ........................................................................... 109
- User authentication options ............................................................................................................. 111
  - AD user authentication .............................................................................................................. 111
  - LDAP user authentication ......................................................................................................... 111
  - Single sign-on ............................................................................................................................ 111
  - Common Access Cards ............................................................................................................. 111
- Configure Active Directory or LDAP .............................................................................................. 111
- Connection information .................................................................................................................. 112
Domain name .................................................................................................................. 112
Port .................................................................................................................................. 112
User and Password .......................................................................................................... 112
Did the connection test fail? ............................................................................................ 112
Base search location ........................................................................................................ 113
Base DN ............................................................................................................................ 113
Advanced settings ............................................................................................................. 113
Summary .......................................................................................................................... 114
Configure authentication and permissions for groups of users ........................................ 114
Log in to DPA .................................................................................................................. 114

**DPA alerts** .................................................................................................................. 116
View the status of DPA alerts ........................................................................................... 116
Configure a DPA alert ....................................................................................................... 116
  Configure a Wait Time alert ......................................................................................... 117
  Configure a Resources alert ....................................................................................... 119
  Configure an Administrative alert ............................................................................... 121
  Configure a Custom alert ........................................................................................... 122
Send SNMP traps from DPA alerts .................................................................................. 123
  The DPA MIB file ....................................................................................................... 124
  Create an SNMP contact ............................................................................................ 124
Stop DPA alerts for a period of time ................................................................................ 125
Alert on increases in SQL wait times .............................................................................. 125
  Determine the average execution time of your queries ............................................... 125
  Create an alert based on the wait threshold .................................................................. 126
Create a custom alert to display the name of the active node in a SQL Server cluster .... 127
  Task 1: Create an alert to get the physical machine name ........................................ 127
  Task 2: Create an alert to append the machine name to the cluster name ............... 129
Create an alert group ....................................................................................................... 133
Create and manage DPA contacts and contact groups .................................................. 133
  Create a contact .......................................................................................................... 134
Create a contact group ................................................................. 134
Edit a contact or contact group .................................................. 135
Delete a contact or contact group ................................................ 135
Notification policy for DPA alerts ................................................ 135
Notify when level not visited since normal .................................. 135
Notify when level changes and is not normal .............................. 136
Notify when level is not normal .................................................. 136

DPA reports ............................................................................. 138
About DPA reports .................................................................... 138
 Differences between report data and detailed chart data displayed in the DPA interface ............. 138
 Report types ............................................................................. 138
Access and run DPA reports ....................................................... 139
Create a DPA report .................................................................. 140
Schedule a report for email delivery ......................................... 141
Create a report group ............................................................... 143

Link together separate DPA servers ....................................... 145
Set up a Central Server and add remote DPA servers ............... 145
 Set up a Central Server ............................................................. 145
 Add remote DPA servers ......................................................... 145
Configure authentication for a DPA Central Server .................. 146
 Log in with a SolarWinds DPA user ......................................... 146
 Log in with an Active Directory or LDAP user ......................... 146
View information from remote servers on the Central Server .... 146
Central Server advanced configuration ..................................... 147
 General Central Server settings ................................................. 147
 Thread pool settings ................................................................. 148
 Client factory cache ............................................................... 149

Use the DPA REST API ............................................................ 150
Manage tokens used for authentication to the DPA API ............ 150
Create a refresh token ............................................................. 150
About storing refresh tokens ........................................................................................................... 151
Delete a refresh token .................................................................................................................... 152
Learn about and experiment with the DPA API .............................................................................. 152
Access the DPA API documentation and get authorization to make API calls ....................... 152
View the DPA API documentation .................................................................................................. 154
Make an API call from the Swagger interface .................................................................................. 156
Examples of using Python scripts to make DPA API calls ............................................................ 158
Prerequisites ..................................................................................................................................... 158
If your DPA server does not use HTTPS or your certificates are self-signed ............................ 158
Get an access token ........................................................................................................................ 158
Database Monitor examples ............................................................................................................. 160
  Get information about one monitored database instance ......................................................... 160
  Start and stop monitoring a database instance given its database ID ........................................ 161
  Get information about all monitored database instances ......................................................... 162
  Stop and start monitoring for all database instances ............................................................... 163
  Update the user password for a monitored database instance ................................................ 164
License Allocation examples ........................................................................................................... 164
  Get information about currently installed licenses ................................................................. 164
  Get license information for a single database instance ........................................................... 165
  Update license information for a database instance ............................................................... 166
Annotation examples ........................................................................................................................ 167
  Get a list of annotations for the last 30 days ............................................................................ 167
  Create a new annotation ............................................................................................................. 168
  Delete an annotation .................................................................................................................. 168
Database Registration examples ..................................................................................................... 169
  Register and unregister a SQL Server database instance for monitoring ...................... 169
  Register and unregister an Oracle database instance for monitoring ............................... 171
  Register and unregister a MySQL database instance for monitoring ............................... 172
Full working script .......................................................................................................................... 174
Examples of using PowerShell scripts to make DPA API calls .................................................... 185
Prerequisite  .......................................................................................................................... 185
If your DPA server does not use HTTPS or your certificates are self-signed .................. 185
Get an access token ............................................................................................................. 186
Database Monitor examples ............................................................................................... 187
  Get information about one monitored database instance ............................................. 187
  Start and stop monitoring a database instance given its database ID ....................... 188
  Get information about all monitored database instances .............................................. 189
  Stop and start monitoring for all database instances ..................................................... 190
  Update the user password for a monitored database instance .................................... 191
License Allocation examples ............................................................................................... 191
  Get information about currently installed licenses ....................................................... 191
  Get license information for a single database instance ................................................. 192
  Update license information for a database instance ...................................................... 192
Annotation examples .......................................................................................................... 193
  Get a list of annotations for the last 30 days ................................................................. 193
  Create a new annotation ................................................................................................. 194
  Delete an annotation ....................................................................................................... 195
Database Registration examples .......................................................................................... 195
  Register and unregister a SQL Server database instance for monitoring ................ 196
  Register and unregister an Oracle database instance for monitoring ...................... 197
  Register and unregister a MySQL database instance for monitoring ....................... 199
Full working script ............................................................................................................. 200

DPA administrative tasks ................................................................................................. 213
  Stop and start DPA ........................................................................................................... 213
  Stop and start DPA on a Windows server ...................................................................... 213
  Stop and start DPA on a Linux server ........................................................................... 213
  Set advanced DPA configuration options ................................................................... 213
  Enable SNMP Monitoring in SCOM ............................................................................. 213
Introduction

See the following topics to get an overview of DPA features and learn about DPA architecture:

- Introduction to DPA
- DPA architecture
Introduction to DPA

You can use SolarWinds Database Performance Analyzer (DPA) to monitor, diagnose, and resolve performance problems for Oracle, SQL Server, MySQL, DB2, Sybase, and Azure SQL databases.

SolarWinds DPA has agentless architecture and uses wait-based analytics for extended database monitoring. SolarWinds DPA uses less than one percent of resources on production systems.

Get a walk-through of DPA functionality from the DPA Getting Started Guide.

See the following topics to start using DPA:

<table>
<thead>
<tr>
<th>Understand and apply DPA licenses</th>
<th>License types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registration and licensing options for clustered environments</td>
<td></td>
</tr>
<tr>
<td>Purchase and view licenses</td>
<td></td>
</tr>
<tr>
<td>Activate DPA licenses</td>
<td></td>
</tr>
<tr>
<td>Allocate or deallocate DPA licenses</td>
<td></td>
</tr>
<tr>
<td>Troubleshoot over-allocated licenses</td>
<td></td>
</tr>
<tr>
<td>Deactivate your DPA licenses</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Register the database instances you want to monitor</th>
<th>Database instances DPA can monitor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register multiple database instances</td>
<td></td>
</tr>
<tr>
<td>Register an Oracle database</td>
<td></td>
</tr>
<tr>
<td>Register a SQL Server database</td>
<td></td>
</tr>
<tr>
<td>Register a Sybase database</td>
<td></td>
</tr>
<tr>
<td>Register a DB2 database</td>
<td></td>
</tr>
<tr>
<td>Register a MySQL database</td>
<td></td>
</tr>
<tr>
<td>Register an Amazon RDS for Oracle database</td>
<td></td>
</tr>
<tr>
<td>Register an Amazon RDS for SQL Server database</td>
<td></td>
</tr>
<tr>
<td>Register an Amazon RDS for MySQL or Aurora database</td>
<td></td>
</tr>
<tr>
<td>Register an Azure SQL database</td>
<td></td>
</tr>
<tr>
<td>Unregister a monitored database instance</td>
<td></td>
</tr>
<tr>
<td>Investigate performance issues</td>
<td>• Access DPA query or table tuning advisors</td>
</tr>
<tr>
<td></td>
<td>• Use DPA's query performance analysis to find the root cause of performance issues</td>
</tr>
<tr>
<td></td>
<td>• Investigate inefficient queries running against a table</td>
</tr>
<tr>
<td></td>
<td>• Identify blocking sessions and deadlocks</td>
</tr>
<tr>
<td></td>
<td>• Find and investigate unusually long wait times (anomalies)</td>
</tr>
<tr>
<td></td>
<td>• About anomaly detection in DPA</td>
</tr>
<tr>
<td></td>
<td>• Add an annotation to document a change to the database</td>
</tr>
<tr>
<td></td>
<td>• Name SQL statements</td>
</tr>
</tbody>
</table>

| Manage monitored database instances                               | • Update a monitored database instance                          |
|                                                                 | • Stop monitoring a database instance for a period of time      |
|                                                                 | • Troubleshooting tips                                         |

| Configure DPA alerts and reports                                 | • Configure a DPA alert                                       |
|                                                                 | • Send SNMP traps from DPA alerts                             |
|                                                                 | • Create an alert group                                       |
|                                                                 | • Create and manage DPA contacts and contact groups           |
|                                                                 | • Create a report group                                       |

| Link DPA servers together                                       | • Set up a Central Server and add remote DPA servers          |
|                                                                 | • Configure authentication for a DPA Central Server           |
|                                                                 | • View information from remote servers on the Central Server  |
|                                                                 | • Central Server advanced configuration                       |

| Automate tasks with the DPA REST API                            | • Manage tokens used for authentication to the DPA API        |
|                                                                 | • Learn about and experiment with the DPA API                |
|                                                                 | • Examples of using Python scripts to make DPA API calls      |
|                                                                 | • Examples of using PowerShell scripts to make DPA API calls  |

**DPA architecture**

SolarWinds Database Performance Analyzer consists of:

- A SolarWinds DPA server
- A SolarWinds DPA repository database
- One or more database instances you want to monitor

The SolarWinds DPA server collects performance data from a set of database instances you choose to monitor. SolarWinds DPA stores this data in the repository database.
For optimal performance, the DPA server, repository database, and the monitored database instances must reside on the same high-speed LAN. If your environment contains database instances that are on separate LANs, SolarWinds recommends creating a repository database on each LAN. For cloud monitoring, SolarWinds recommends deployment to the same region.

The SolarWinds DPA server provides a web interface that displays performance data in a web browser from any computer with access to the SolarWinds DPA server.

SolarWinds recommends installing one SolarWinds DPA instance on a computer. If you must install multiple instances on the same computer, submit a support ticket.

Two key functions of the DPA server

- Collect data from the monitored database instances and store the data in the repository database.
- Provide a web interface that displays performance data from any computer with access to the SolarWinds DPA server. From this interface, you can configure monitoring, alerting, and email reports.

Monitored database instances

SolarWinds DPA remotely connects to each database instance using Java Database Connectivity (JDBC). SolarWinds DPA causes less than 1% overhead on the instance. No software is installed on the monitored server.

Monitored virtualization environment

In a virtual environment, SolarWinds DPA can remotely connect to each VMware vCenter Server, ESX, or ESXi host. SolarWinds DPA causes less than 1% overhead on the monitored systems. No software is installed in the vCenter Server, ESX or ESXi host, or virtual machines.
DPA licensing

See the following topics to learn more about DPA licensing:

- Learn about DPA license types and subscription licensing for DPA servers deployed in the Amazon Web Services (AWS) Marketplace.
- Learn about registration and licensing options for clustered environments.
- If you are monitoring a database instance that runs in a VM cluster, see the requirements to create a user.
- Purchase licenses and view your purchased licenses in the Customer Portal.
- Activate DPA licenses to make them available to a DPA server.
- Allocate licenses to the DPA database instances you want to monitor, or deallocate a license to make it available to another instance.
- If licenses are over-allocated, troubleshoot and resolve the issue.
- Deactivate DPA licenses to make them available to a different DPA server.

License types

SolarWinds DPA provides the following licensing options:

- Subscription licensing is used for DPA servers deployed in the Amazon Web Services (AWS) Marketplace.
- For other DPA servers, an individual license is required for each monitored database instance.

Individual licenses

SolarWinds sells individual licenses by category according to the database edition they are authorized to monitor. You must buy an individual license for each database instance you monitor. In addition, you can buy virtual machine licenses to monitor the virtual infrastructure hosting a database instance.

To purchase licenses, contact the SolarWinds sales team.

<table>
<thead>
<tr>
<th>LICENSE TYPE</th>
<th>CAN MONITOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category 1 licenses</td>
<td>Category 1 licenses can monitor all database types. They are required for:</td>
</tr>
<tr>
<td></td>
<td>- Oracle: all editions except Standard and Express</td>
</tr>
<tr>
<td></td>
<td>- Sybase: all editions except Express</td>
</tr>
<tr>
<td></td>
<td>- DB2: all editions except Express</td>
</tr>
</tbody>
</table>
### License Type

<table>
<thead>
<tr>
<th>Category 2 licenses</th>
<th>Can Monitor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle: Standard and Express editions (including Amazon RDS)</td>
<td></td>
</tr>
<tr>
<td>SQL Server: all editions (including Amazon RDS)</td>
<td></td>
</tr>
<tr>
<td>MySQL: all editions (including Amazon RDS and Aurora)</td>
<td></td>
</tr>
<tr>
<td>Sybase: Express edition</td>
<td></td>
</tr>
<tr>
<td>DB2: Express edition</td>
<td></td>
</tr>
</tbody>
</table>

If you run out of Category 2 licenses, you can use Category 1 licenses instead.

<table>
<thead>
<tr>
<th>Azure SQL Database license</th>
<th>Can Monitor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Azure SQL Database: all editions, including databases in elastic pools</td>
<td></td>
</tr>
</tbody>
</table>

If you run out of Azure SQL Database licenses, you can use Category 1 or Category 2 licenses instead.

<table>
<thead>
<tr>
<th>VM Option licenses</th>
<th>Can Monitor</th>
</tr>
</thead>
<tbody>
<tr>
<td>If a database instance runs on a virtual machine (VM), you can apply an optional VM license in addition to the Category 1 or Category 2 license. When you apply a VM license, DPA collects performance metrics from the VM and the physical host on which the database instance runs. This information is displayed in the Virtualization view.</td>
<td></td>
</tr>
</tbody>
</table>

### All individual licenses are floating

You can register more instances than you have licenses for. On the license allocation page, assign the licenses to the instances you want to monitor.

DPA does not collect data from registered database instances that are not licensed. However, you can view previously collected data on those database instances.

### Clustered environments

For information about registering SQL Server AGs and Oracle RACs, see [Registration and licensing options for clustered environments](#).

If you are monitoring a database instance that runs in a virtual machine (VM) cluster, a [user with at least read-only permissions is required](#) on the hosts and VMs that will be monitored.

### Subscription licensing

When you deploy the DPA server from the AWS Marketplace, DPA uses subscription licensing. As you register databases and monitor them, DPA charges your Amazon Subscription through the AWS Metering Service. DPA charges are based on the number of database instances you monitor each hour. See the AWS Marketplace for details and pricing.

With subscription licensing, you can monitor any supported database type (like the Category 1 license described in the following section). However, you cannot access the VM-related information that is available with a VM Option license.
If you want to use individual DPA licenses in the Amazon cloud, you can deploy an EC2 instance, install DPA, and apply your licenses. You cannot use both individual DPA licenses and a subscription on a single DPA server.

Learn more
For more information about purchasing and allocating licenses, see:

- Purchase and view licenses
- Activate individual DPA licenses
- Allocate or deallocate individual DPA licenses
- Deactivate your licenses
- Troubleshoot over-allocated licenses

Registration and licensing options for clustered environments
To get the maximum value from DPA, SolarWinds recommends the following options for registering and licensing SQL Server Availability Groups (AGs) and Oracle Real Application Clusters (RACs).

Every environment is different, so talk with your SolarWinds representative for other options.

SQL Server AGs
You can register SQL Server availability groups (AGs) using either of the following options:

- Register each SQL Server instance in the cluster
- Register the AG listener

Register each SQL Server instance in the cluster
If there are multiple AGs in the cluster, this option is recommended because it ensures that DPA does not monitor the same instance more than once. DPA monitors all activity on each instance, including primary and secondary AG activity.

With this option, DPA does not follow AGs when they fail over. Monitoring all instances in the cluster ensures that you see all activity when AG failovers occur.

Register the AG listener
Use this option if you want to monitor activity on the instance that contains the primary replica of an AG. When the AG fails over, DPA follows the listener and begins monitoring the SQL Server instance that now acts as the AG's primary replica.
SolarWinds recommends registering only one listener per cluster unless you can ensure that no instance in the cluster will act as the primary replica for multiple AGs. If you register multiple listeners and the same instance acts as the primary replica for more than one of the AGs, DPA monitors that instance multiple times. Duplicate monitoring is not recommended.

SQL Server logins are **not** automatically replicated. To enable DPA to continue monitoring after a failover, you must [manually create the DPA login on all instances](https://www.solarwinds.com/support/article/31152) in the cluster that can act as the primary replica for the AG.

## Oracle RACs

For Oracle RAC (Real Application Clusters), register every instance in the cluster. Do not register the virtual IP that distributes load across the RAC instances.

For Oracle RAC with Data Guard, register both environments but only monitor the primary one. If a failover occurs, simply reassign the licenses to the instances in the secondary RAC environment.

When you register a RAC instance, there may be listener configuration changes needed if you are not listening on the physical IP address. SolarWinds recommends:

- If you are registering pluggable databases (PDBs) on a RAC instance, consider registering with the physical IP address of the host.

  > To monitor an Oracle multitenant container database (CDB), register each PDB contained in the CDB. You cannot register the CDB directly.

  > When you register two or more Oracle PDBs in the same CDB, DPA automatically creates a group for the CDB. For more information, see [About monitoring Oracle multitenent databases](https://www.solarwinds.com/support/article/31152) (CDBs).

- If you are registering a non-PDB RAC instance, consider registering with the SID.

- If you are using the Service Name, consider using the physical IP address of the host. Do not use the virtual IP address (VIP) or the Oracle Single Client Access Name (SCAN) IP address.

## Learn more

For more information about licensing, see the following topics:

- [License types](https://www.solarwinds.com/support/article/31152)
- [Purchase and view licenses](https://www.solarwinds.com/support/article/31152)
- [Activate individual DPA licenses](https://www.solarwinds.com/support/article/31152)
- [Allocate or deallocate individual DPA licenses](https://www.solarwinds.com/support/article/31152)
Requirements for monitoring a database instance running in a VM cluster

If you are monitoring a database instance that runs in a virtual machine (VM) cluster, a user with at least read-only permissions is required on the hosts and VMs that will be monitored. The monitored hosts and VMs include all of the following:

- The VMs that monitored database instances are running on.
- All hosts that those VMs could potentially run on (for example, all hosts in a DRS cluster).
- Other VMs on those hosts.

SolarWinds recommends giving the user read-only permissions on the entire vCenter Server or ESX/ESXi host so that DPA can dynamically monitor any entity as system changes take place.

Create a user on the vCenter Server or the ESX/ESXi host

Before you can assign user permissions, you must create the user:

- vCenter Server user: Authorized users for vCenter Server are those included in the Windows domain list referenced by vCenter Server or local Windows users on the vCenter Server system. To edit the user list or change user passwords, use the tools you use to manage your Windows domain or Active Directory.
- ESX/ESXi host user: Log in to an ESX/ESXi host as root using the vSphere Client. Then use the Users and Groups tab to add users, remove users, change passwords, set group membership, and assign the required permissions.

Assign user permissions to inventory objects

Use the vSphere Client to assign user permissions to inventory objects, such as the vCenter server, data center, host, or folder. Requirements and best practices:

- You must have modify permission on an object to be able to assign permissions to that object.
- SolarWinds recommends selecting the entire vCenter Server or ESX/ESXi host and assigning permissions to it.
- Make sure that the Propagate to Child Objects option is selected. This ensures that when new objects are inserted into the inventory hierarchy, they inherit the required permissions.

Purchase and view licenses

For DPA servers that use individual licenses, SolarWinds DPA has a 14-day evaluation license. During the evaluation period, you can monitor and view data for an unrestricted number of database instances. After the evaluation period, to continue monitoring you must purchase the correct quantity and type of licenses for your database instances.
Purchase licenses

Contact our sales team to purchase licenses directly from SolarWinds.

Only buy licenses for active database instances. Standby database instances used for disaster recovery or high availability do not need licenses.

- Online quote tool
- sales@solarwinds.com
- 866.530.8100

View purchased licenses

After you purchase individual licenses, you can view your SolarWinds DPA licenses in the SolarWinds Customer Portal.

1. Access the Customer Portal.
2. Click Licenses > Manage Licenses.

The licenses for your SolarWinds DPA product are listed by license type.

Next steps

After you have purchased licenses, you must activate them on a DPA server, register database instances for monitoring, and then allocate licenses to the registered instances.

Activate DPA licenses

After the DPA trial period ends, DPA monitors only licensed instances. If your DPA server uses individual licenses, you must activate a license for each database instance that you want to monitor. Make sure that you have the correct license types for the database instances you want to monitor.

- For information about licensing options for SQL Server Availability Groups and Oracle RAC environments, see Registration and licensing options for clustered environments.

Activate licenses online

If the DPA server is connected to the Internet, you can activate licenses online.
2. Choose Licenses > Manage Licenses.
3. Locate the license, and expand it.
4. Copy the activation key.
5. On the SolarWinds DPA homepage, click License Management. Then click License Manager.
6. Click Enter Activation Key.
7. Select Online Activation, and click Next.
8. On the Online Activation page, paste the activation key into the correct field.
9. In the Amount to Activate section, select All Available or Specify Amount.

Unactivated licenses can be activated later. You can reuse an activation key on a different SolarWinds DPA server and activate remaining licenses there.

10. Enter the remaining information, and click Activate.

Activate licenses offline

Offline activation requires a transfer of files between the SolarWinds DPA server and a computer connected to the Internet. You can use email, shared storage, or a USB flash drive.

1. In SolarWinds DPA, click License Management > License Manager.
2. Click Enter Activation Key.
3. Select Offline Activation, and click Next.
4. On the Offline Activation page, copy the text string next to the license type you want to activate, and save it to a text file. This is your unique machine ID. Include the brackets. For example:

   [7R12-X2QN-U8XM-WXTD-23H7-0TD7-59QH-6ERF-5BRN-2M17-328G-0DT2-MNMS-005C-000Z-04Q2-0000]

5. Transfer this text file to a computer with Internet access.
7. Locate the license, and expand it.
8. Click Activate license manually.
9. Paste the text string into the Unique Machine ID field, and enter the other required information.
10. Click Generate License File to download the license file.
11. Transfer the license file to the SolarWinds DPA server.
12. On the Offline Activation page, click Choose File and browse to the license file you just transferred.
13. Click Activate.
Next steps
When you activate a license, DPA automatically allocates the license to a registered database instance if you have enough licenses to monitor all registered instances in that license category. If you do not have enough licenses to monitor all registered instances, you must manually allocate licenses to the instances you want to monitor.

Allocate or deallocate DPA licenses
If your DPA server uses individual licenses, a license must be allocated to each registered database instance that you want to monitor. DPA starts monitoring new instances immediately after licenses are allocated.

Allocate VM licenses if there are enough VM licenses to cover all database instances that:
- Are linked to a VM
- Have been allocated a Category 1 or 2 license

If you have not activated enough licenses to cover all instances that require that license type, DPA does not allocate any of the licenses. You must manually allocate licenses to the database instances you want to monitor.

Example 1:
1. You register 10 Oracle Enterprise Edition database instances, which require Category 1 licenses.
2. You activate 15 Category 1 licenses.
   Result: DPA automatically allocates 10 of the licenses to the Oracle Enterprise Edition database instances.
   Result: DPA automatically allocates the remaining 5 Category 1 licenses.

Example 2:
1. You register 15 MySQL database instances, which require Category 2 (or greater) licenses.
2. You activate 10 Category 2 licenses. No Category 1 licenses are available.
Result: DPA does not allocate any of the licenses. You can either activate at least 5 additional licenses, or manually allocate licenses to the instances you want to monitor.

Instances without a license allocated to them remain registered with DPA, and you can view performance data that was collected in the past. You can deallocate a license from one registered instance and allocate it to another if necessary.

Manually allocate licenses to database instances

Use License Allocation to configure how your licenses are allocated to database instances.

View current license allocation

1. On the DPA homepage, click License Management.
2. See the current license allocations in the summary boxes near the top of the License Allocation page.

Allocate licenses to database instances

1. On the License Allocation page, find the database instance you want in the list of registered database instances.
2. Select the Cat 1, Cat 2, or Azure check box next to the instance.
3. Click Save.
   The license count is updated after you allocate a license.

Allocate VM licenses to VM database instances

If a database instance runs on a virtual machine (VM), you can allocate a VM license to it in addition to a Category 1 or 2 license. When you allocate a VM license, DPA collects performance metrics from the VMware system (vCenter Server or ESX/ESXi Host) on which the database instance runs.

1. On the License Allocation page, locate a VM-hosted database instance that has a Category 1 or 2 license allocated to it.
2. Select the VM check box next to the instance.
3. Click Save.

   If you are monitoring a database instance that runs in a virtual machine (VM) cluster, a user with at least read-only permissions is required on the hosts and VMs that will be monitored.

Deallocate licenses

You can deallocate the license from one database instance to make it available to another database instance.

1. On the DPA homepage, click License Management.
2. Clear the Cat 1, Cat 2, or Azure check box to deallocate licenses.
If you clear a Category 1 or 2 license from an instance that also has a VM license, SolarWinds DPA automatically clears the VM license as well.

Learn more

For more information about licensing, see the following topics:

- Purchase and view licenses
- Activate DPA licenses
- Troubleshoot over-allocated licenses

Troubleshoot over-allocated licenses

The DPA homepage displays a red banner if SolarWinds DPA is monitoring more registered database instances than you have licenses to monitor. This can happen in two situations:

- A license expires when you have unexpired licenses of the same type on the server.
- You deactivate a license and have other licenses of the same type on the server.

If SolarWinds DPA licenses are over-allocated, you cannot view or analyze your database instances until you deallocate the extra licenses. SolarWinds DPA continues monitoring the databases, so you will not lose data while you bring the allocated licenses to an allowable level.

If necessary, you can purchase and activate additional licenses.

To correct an issue of over-allocated licenses, deallocate database instances until you reach the proper number of licenses. If Category 2 licenses are over-allocated, assign available Category 1 licenses to cover the shortage. If Azure SQL Database licenses are over-allocated, assign Category 1 or 2 licenses to cover the shortage.

1. On the DPA homepage, click License Management.
2. Locate the over-allocated license type on the allocations chart. Over-allocated license types are shown in red.
3. Clear Cat 1, Cat 2, Azure, or VM check boxes until the chart is no longer red.
4. Click Save.
   
   You should now see your database instances in your views.

Deactivate your DPA licenses

You can deactivate individual licenses on a SolarWinds DPA server to make the licenses available elsewhere.

If your DPA server has direct access to the internet, you can deactivate licenses online.

Evaluation licenses and temporary keys cannot be deactivated.
Deactivate online

1. On the DPA homepage, click License Management.
2. On the License Allocation page, click License Manager.
3. In the Licenses section, locate the License Key you want to deactivate.
4. Click Deactivate.

Deactivate offline

To deactivate a license offline in SolarWinds DPA 10.0 or earlier, contact SolarWinds customer support.

To deactivate a license offline in SolarWinds DPA 10.1 and later:

1. From the SolarWinds DPA homepage, click License Management > License Manager.
2. Click Deactivate next to the license.
3. Click Yes to confirm the offline deactivation, and download the deactivation receipt file.
4. Log in to the SolarWinds Customer Portal, and go to the License Management page.
5. Select the DPA instance, and click Deactivate license manually.
6. On the Manage License Deactivation page, browse for the deactivation receipt file, and click Upload.
Register a database instance for monitoring

For more information about the database instances you can monitor and the requirements for the privileged user, see Database instances DPA can monitor.

Register multiple database instances

If you are monitoring a large number of database instances, use the DPA mass registration feature to quickly register multiple databases.

Register a single database instance

To register a database instance for monitoring, select the type you want to register.

Self-Managed

- Oracle
- Microsoft SQL Server
- SAP Sybase ASE
- DB2 UDB
- MySQL

Amazon RDS

- Amazon RDS for Oracle
- Amazon RDS for SQL Server
- Amazon RDS for MySQL or Aurora

Azure

- Azure SQL DB

Database instances DPA can monitor

SolarWinds DPA can monitor database instances you manage on both physical and virtual servers or Amazon RDS instances hosted in the Amazon Elastic Compute Cloud (EC2). The server hosting DPA must be able to connect to the monitored server.
Self-managed databases

When you register a self-managed database instance, you must provide the credentials of a **privileged user**. The required privileges are listed below. During registration, the privileged user either creates the monitoring user, or grants privileges as needed to an existing user designated as the monitoring user. DPA does not store the credentials of the privileged user.

<table>
<thead>
<tr>
<th>DATABASE</th>
<th>PRIVILEGES REQUIRED FOR THE PRIVILEGED USER</th>
<th>SUPPORTED VERSIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle</td>
<td>SYS user</td>
<td>• 18.3.x (single tenant and multitenant*)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 12.2.x (single and multitenant)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 12.1.x (single and multitenant)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 11.2.x</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>SYSADMIN role</td>
<td>• 2017 (Windows and Linux)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2016 SP2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2014 SP2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2012 SP4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2008 R2 SP3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2008 SP4</td>
</tr>
<tr>
<td>SAP Sybase ASE</td>
<td>SA_ROLE</td>
<td>• 16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 15.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 15.5</td>
</tr>
<tr>
<td>IBM DB2 LUW</td>
<td>SYSADM</td>
<td>• 11.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 10.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 10.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 9.7</td>
</tr>
<tr>
<td>MySQL</td>
<td>The privileged user requires the CREATE USER permission and must be able to grant the following permissions:</td>
<td>• 5.7.9</td>
</tr>
<tr>
<td></td>
<td>PROCESS on <em>.</em></td>
<td>• 5.6.10</td>
</tr>
<tr>
<td></td>
<td>SELECT &amp; UPDATE on performance_schema.*</td>
<td>• Percona 5.6 and 5.7</td>
</tr>
<tr>
<td></td>
<td>To enable the retrieval of query execution plans, this privileged user must be able to grant the following permissions:</td>
<td>• Maria 10.0, 10.1, and 10.2</td>
</tr>
<tr>
<td></td>
<td>SELECT, INSERT, UPDATE, DELETE on <em>.</em></td>
<td><img src="https://example.com/dpa_cannot_monitor_mysql_8_0_database_instances.png" alt="DPA cannot monitor MySQL 8.0 database instances." /></td>
</tr>
</tbody>
</table>
To monitor an Oracle multitenant container database (CDB), register each pluggable database (PDB) contained in the CDB. Register each PDB just as you would register an Oracle single tenant database. For more information, see Registration and licensing options for clustered environments.

Amazon RDS databases

DPA can monitor Amazon RDS Oracle, Microsoft SQL Server, and MySQL instances.

<table>
<thead>
<tr>
<th>Amazon RDS</th>
<th>Supported Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle</td>
<td>12.2.x</td>
</tr>
<tr>
<td></td>
<td>12.1.x</td>
</tr>
<tr>
<td></td>
<td>11.2.x</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>2017</td>
</tr>
<tr>
<td></td>
<td>2016</td>
</tr>
<tr>
<td></td>
<td>2014 SP1</td>
</tr>
<tr>
<td></td>
<td>2012 SP2</td>
</tr>
<tr>
<td></td>
<td>2008 R2 SP3</td>
</tr>
<tr>
<td>MySQL</td>
<td>5.7.9+</td>
</tr>
<tr>
<td></td>
<td>5.6.10+</td>
</tr>
<tr>
<td></td>
<td>Aurora 5.6.10a+</td>
</tr>
</tbody>
</table>

Key differences for Oracle databases on Amazon RDS

Because of Amazon RDS access restrictions, some features that are available on Oracle self-managed database instances are not available for Amazon RDS instances.

<table>
<thead>
<tr>
<th>Category</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unavailable alerts</td>
<td>Oracle Alert Log Error uses V$DIAG_ALERT_EXT instead of X$DBGALERTEX.</td>
</tr>
<tr>
<td>Explain plans</td>
<td>Explain plans cannot be generated with a SYS account. You must specify a different account to generate the live plan.</td>
</tr>
<tr>
<td>Workarounds</td>
<td><strong>To kill a real time session, use</strong> RDSADMIN.RDSADMIN_UTIL.KILL.</td>
</tr>
<tr>
<td></td>
<td><strong>Trace session permissions granted through</strong> START_TRACE_IN_SESSION and STOP_TRACE_IN_SESSION.</td>
</tr>
</tbody>
</table>

Key differences for SQL Server databases on Amazon RDS

Because of Amazon RDS access restrictions, some features that are available on SQL Server self-managed database instances are not available for Amazon RDS instances.
### Category: Details

| Unavailable alerts | SQL Server Windows Service Not Running  
|                    | SQL Server Long Running Jobs  
|                    | SQL Server Log Has Many Virtual Logs  
|                    | SQL Server Job Failure  
|                    | SQL Server Error Log Alert  

**Unavailable metrics**

|          | CPU Queue Length  
|          | CPU Utilization  
|          | Disk Queue Length  
|          | Memory Paging Rate  
|          | Memory Utilization  
|          | Physical I/O Rate  
|          | Physical Read Rate  
|          | Physical Write Rate  

**Explain plans**

The DPA monitoring user does not have a sysadmin role and may have limited access to objects. You can specify a different user to generate the live plan before you generate the plan.

**Workaround for not having a SYSADMIN role**

DPA user is a member of PROCESSADMIN role

**Deadlock polling**

The monitoring user and database administrator (DBA) do not have permission to create a custom Extended Events Session. Only the default `system_health` Extended Events Session can be used for deadlock polling.

### About repointing database instances

You cannot transfer a registered Oracle or SQL Server database instance between Amazon RDS and a self-managed database and retain DPA historical data. An Oracle or SQL Server database instance transferred between Amazon RDS and a self-managed instance must be registered in DPA as a separate instance.

MySQL database instances can be repointed. After you transfer a MySQL database instance between Amazon RDS and self-managed, you can repoint DPA to the new instance and continue monitoring where you left off. To repoint, use the Update Connection Info wizard in DPA to update the connection details of the registered database instance to point to the new location.

### Azure databases

<table>
<thead>
<tr>
<th>Database</th>
<th>Required Privileges</th>
<th>Supported Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Azure SQL</td>
<td>db_owner role</td>
<td>V12</td>
</tr>
</tbody>
</table>

---

**Page 30**
### Key differences between self-managed SQL Server and Azure SQL database instances

<table>
<thead>
<tr>
<th><strong>Category</strong></th>
<th><strong>Details</strong></th>
</tr>
</thead>
</table>
| Unavailable Alerts                         | - Transaction Log Freespace  
- Windows Service Not Running – SQL Server  
- SQL Server Abnormal Mirroring Status  
- SQL Server Error Log Alerts  
- SQL Server Job Failure  
- SQL Server Log has Many Virtual Logs  
- SQL Server Long Running Jobs |
| Unavailable CPU Metrics                    | - Signal Waits  
- O/S CPU Utilization |
| Unavailable Memory Metrics                 | - Page Life Expectancy  
- O/S Memory Utilization  
- Plan Cache Size  
- Buffer Cache Size  
- Plan Cache Hit Ratio  
- Buffer Cache Hit Ratio  
- Log Bytes Flushed  
- Log Flushes  
- SQL Compilation  
- SQL Re-Compilations |
| Unavailable Disk Metrics                   | - Total I/O Wait Time  
- Total Read I/O Wait Time  
- Total Write I/O Wait Time  
- O/S Disk Queue Length  
- Page Reads  
- Page Writes  
- SQL Disk Read Latency  
- SQL Disk Write Latency |
| Unavailable Sessions Metrics               | - Transaction Rate  
- Batch Requests |
| Unavailable License Compliance Metrics     | - Core Count |
| Additional DTU metrics                     | - DTU Utilization  
- DTU Consumption  
- DTU Limit |
### CATEGORY

<table>
<thead>
<tr>
<th>Additional Memory metrics</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Memory Usage Utilization</td>
</tr>
<tr>
<td></td>
<td>• XTP Storage Utilization</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Additional Disk metrics</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Data I/O Utilization</td>
</tr>
<tr>
<td></td>
<td>• Log Write Utilization</td>
</tr>
<tr>
<td></td>
<td>• Database Storage Consumption</td>
</tr>
<tr>
<td></td>
<td>• Database Size</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Additional Sessions metrics</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Max Worker Utilization</td>
</tr>
<tr>
<td></td>
<td>• Max Session Utilization</td>
</tr>
</tbody>
</table>

### About repointing database instances

Repointing database instances is not possible between Azure SQL and SQL Server.

### Register multiple database instances

If you are monitoring a large number of database instances, use the DPA mass registration feature to quickly register multiple databases.

You can also use a wizard to register a single database instance.

Complete the following steps to download a predefined template and enter the required information for all database instances.

1. In the DPA menu bar, click Options.
2. Under Monitor Setup > Database Instances, click Mass registration.
3. In the Choose a Database Type drop-down, select one of the following:
   - If all database instances are the same type (for example, all Oracle or all SQL Server), select the database type.
   - To register different types of database instances, select All database types.

4. Specify whether you want to edit and save the template on the DPA server or on your local computer.

Based on your selections, instructions are displayed in the right pane.
5. Under How to, click the template link to download the registration file template for the selected database type.
   Or, for Azure SQL database instances, click the link to instructions for auto-generating the registration file.

6. Edit the file to add information about each database instance;
   - Click the required information link in the How to section for information about what to enter in each column.
   - Do not edit the header row.

7. Save the file in .csv format.

   ![Tip]
   If you selected From DPA server, the file **must** be saved in the following location:
   
   `<DPA_home>/iwc/tomcat/ignite_config/registration`
   
   Depending on the database type, it must have one of the following file names:
   - massreg_mixed.csv
   - massreg_oracle.csv
   - massreg_sql_server.csv
   - massreg_azure_sql_database.csv
   - massreg_mysql.csv

8. If you selected From local computer, click the Choose File button and select the file you saved.

9. Click Load Registration File.

If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must **activate a license** to monitor the database instance.

### Register an Oracle database

   ![Tip]
   If you are monitoring an Oracle Real Application Cluster (RAC), see Registration and licensing options for clustered environments.

To register an Oracle database for DPA to monitor:

1. On the DPA homepage, click Register DB Instance for Monitoring.
2. Under Self-Managed, click Oracle.
3. Click Next.
4. Complete the wizard panels as described in the following table.
**Panel** | **Instructions**
--- | ---
Enter Monitored Database Instance Connection Information | SolarWinds DPA monitors all databases within the instance. If more than one instance exists on the server, you must register each instance separately in DPA.

Oracle database instances have three connection options:
- Direct Connect
- Transparent Network Substrate (TNS) Connect Descriptor
- Lightweight Directory Access Protocol (LDAP) or TNS Name

**Direct Connect**

Enter the Service Name or System Identifier (SID), host name or IP address, and port. The default port is 1521.

**TNS Connect Descriptor**

The Connect Descriptor value contains everything after `NAME=` in the `tnsnames.ora` file. The beginning `(DESCRIPTION=` is necessary. For example:

```
(DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP) (HOST = demo.confio.com) (PORT = 1521))) (CONNECT_DATA = (SERVICE_NAME = demo)))
```

**LDAP or TNS Name**

To use this option, Oracle Name Resolution must be configured. For instructions, see [Connect to Oracle using name resolution](#).

After you configure Oracle Name Resolution, you can use the LDAP/TNS Name when registering additional monitored database instances.

SolarWinds DPA uses the Oracle network configuration `.ora` files to connect to the database instance.

If connection information other than the name has changed, update the `.ora` files. SolarWinds DPA detects changes to these files automatically. You do not have to update the connection information through this wizard.

If the name has changed, update the `.ora` files. Then select the check box next to LDAP/TNS Name, and update the value.
<table>
<thead>
<tr>
<th>PANEL</th>
<th>INSTRUCTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter Monitored Database Instance Connection Information (continued)</td>
<td>RAC instances</td>
</tr>
<tr>
<td></td>
<td>If this is an Oracle Real Application Clusters (RAC) instance, there may be listener configuration changes needed if you are not listening on the physical IP address. SolarWinds recommends:</td>
</tr>
<tr>
<td></td>
<td>- If you are registering pluggable databases (PDBs) on a RAC instance, consider registering with the physical IP address of the host.</td>
</tr>
<tr>
<td></td>
<td>- If you are registering a non-PDB RAC instance, consider registering with the SID.</td>
</tr>
<tr>
<td></td>
<td>- If you are using the Service Name, consider using the physical IP address of the host. Do not use the virtual IP address (VIP) or the Oracle Single Client Access Name (SCAN) IP address.</td>
</tr>
<tr>
<td>DBA user</td>
<td>Enter DBA credentials for SolarWinds DPA to register the database instance.</td>
</tr>
<tr>
<td>Enter the Monitoring User</td>
<td>Create or specify the account that DPA will use to gather information. To ensure that the account has the required permissions, SolarWinds recommends creating a new account.</td>
</tr>
<tr>
<td></td>
<td>To create a new account:</td>
</tr>
<tr>
<td></td>
<td>1. Next to Create Monitoring User, click Yes.</td>
</tr>
<tr>
<td></td>
<td>2. Enter the user name and password.</td>
</tr>
<tr>
<td></td>
<td>3. Select a Tablespace and Temp Tablespace on the monitored database. This is primarily used for gathering Explain Plan data for monitored queries.</td>
</tr>
<tr>
<td></td>
<td>To specify an existing account:</td>
</tr>
<tr>
<td></td>
<td>1. Next to Create Monitoring User, click No.</td>
</tr>
<tr>
<td></td>
<td>2. Enter the user name and password.</td>
</tr>
<tr>
<td></td>
<td>If you create the user manually, DPA uses the default Tablespaces for that user.</td>
</tr>
<tr>
<td></td>
<td>If you are registering multiple Oracle Real Application Clusters (RAC) nodes, you may receive an error that the user already exists. You can create a different monitoring user or clear the Create a New Monitoring User check box and continue.</td>
</tr>
</tbody>
</table>
### Oracle Monitoring Information

If the monitored instance contains the Oracle E-Business Suite, SolarWinds DPA can collect additional information about the suite.

SolarWinds DPA can capture Oracle E-Business data to identify the screens, modules, and users generating the database requests. This gives you increased visibility into the causes of performance problems in the Oracle E-Business Suite, Oracle Enterprise Resource Planning (ERP), and Oracle Applications environments.

The SYS password is requested only if remote login as SYS is enabled on the monitored Oracle instance. This option is not available for Amazon RDS instances.

If you do not have remote SYS access to the computer, click the link to open the Manual Steps for Monitored Database Instance Registration.

You can run a script to run on the monitored instance to install a utility package for SolarWinds DPA that grants Execute permissions for that package to the monitoring user.

To register the monitored database instance manually:

1. Click Register the monitored database instance manually.
2. Click Select All, copy the script, and paste it into a text file.
3. As an Oracle Administrator, log in as SYS to the database instance to be monitored.
5. Execute the script.

### Oracle Repository Tablespace

If your repository database is not Oracle, the wizard skips this step.

Choose the tablespace in the repository database to store DPA performance data for this monitored instance.

By default, the performance data is stored in the default tablespace of the repository user. However, data for monitored instances can be stored in separate tablespaces.

### Select the Alert Groups

If you have no Alert Groups set up, or if this new database instance does not match the database type of the Alert Group, the wizard skips this step.

Alert Groups simplify alert configuration and help make alerting more consistent across the monitored database instances.

Select the Alert Groups you want the new database instance to join.

### Summary

Review the information and click Register Database Instance.

### Database Instance Registration Complete

Click Finish to return to the DPA homepage.
If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must activate a license to monitor the database instance.

**Register a SQL Server database**

If you are monitoring a SQL Server Availability Group (AG), see Registration and licensing options for clustered environments.

To register a SQL Server database for DPA to monitor:

1. On the DPA homepage, click Register DB Instance for Monitoring.
2. Under Self-Managed, click Microsoft SQL Server.
3. Click Next.
4. Complete the wizard panels as described in the following table.
**Enter Monitored Database Instance Connection Information**

1. Enter connection information for the SQL Server instance:
   - If the SQL Server Browser service is running, enter the server name or IP address and the instance name in this format: `Server\Instance`.
   - If the SQL Server instance contains one or more Availability Groups, click Note for Availability Groups for instructions on how to register primary and secondary replicas.
   - Otherwise, enter the server name or IP address and the port number.

   - SolarWinds DPA monitors all databases within the instance. If more than one instance exists on the server, you must register each instance separately in DPA.

2. Select the type of authentication you want to use. If Mixed Mode was selected during the SQL Server installation, you can choose either option.

3. Enter a SYSADMIN login that DPA can use to register the instance.

   - DPA does not use or store these credentials after you complete the wizard.

   - For Windows authentication, enter `<DOMAIN>\<username>` in the SYSADMIN User field.
   - For SQL Server authentication, enter the credentials that you enter on the Connect to Server dialog in SQL Server Management Studio (with Database Engine as the Server type).

SSL is requested by default. If the server does not support SSL, a plain connection is used.

Are you receiving errors? See [DPA for SQL Server installation troubleshooting](#).
Enter the Monitoring User

Create or specify the account that DPA will use to gather information. To ensure that the account has the required permissions, SolarWinds recommends creating a new account.

To create a new account:

1. Click Yes.
2. Select SQL Server as the authentication method. (DPA cannot create a new Windows account.)
3. Enter a user name and password for the new account, or accept the default values.

To specify an existing account:

1. Click No.
2. Select either authentication method.
3. Enter the user name and password of an existing account.
   
   For Windows authentication, enter `<DOMAIN>\<username>` in the Monitoring User field.

   You can also authenticate using a Windows Computer Account.

   For SQL Server authentication, only the user name is required. Do not specify a domain.

Oracle Repository Tablespace

If your repository database is not Oracle, the wizard skips this step.

Choose the tablespace in the repository database to store DPA performance data for this monitored instance.

By default, the performance data is stored in the default tablespace of the repository user. However, data for monitored instances can be stored in separate tablespaces.

Select the Alert Groups

If you have no Alert Groups set up, or if this new database instance does not match the database type of the Alert Group, the wizard skips this step.

Alert Groups simplify alert configuration and help make alerting more consistent across the monitored database instances.

Select the Alert Groups you want the new database instance to join.

Summary

Review the information and click Register Database Instance.

Database Instance Registration Complete

Click Finish to return to the DPA homepage.

If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must [activate a license](#) to monitor the database instance.
Register a Sybase database

To register a Sybase database for DPA to monitor:

1. On the DPA homepage, click Register DB Instance for Monitoring.
2. Under Self-Managed, click SAP Sybase ASE.
3. Click Next.
4. Complete the wizard panels as described in the following table.

<table>
<thead>
<tr>
<th>PANEL</th>
<th>INSTRUCTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter Monitored Database Instance Connection Information</td>
<td>Enter the server name or IP address and port of the Sybase server.</td>
</tr>
<tr>
<td></td>
<td>SolarWinds DPA monitors all databases within the instance. If more than one instance exists on the server, you must register each instance separately in DPA.</td>
</tr>
<tr>
<td></td>
<td>Enter SA_ROLE credentials for SolarWinds DPA to register the database instance.</td>
</tr>
<tr>
<td></td>
<td>The Sybase Monitor Server does not need to be configured for SolarWinds DPA to monitor the database.</td>
</tr>
<tr>
<td>Enter the Monitoring User</td>
<td>Create or specify the account that DPA will use to gather information. To ensure that the account has the required permissions, SolarWinds recommends creating a new account.</td>
</tr>
<tr>
<td></td>
<td>To create a new account:</td>
</tr>
<tr>
<td></td>
<td>1. Next to Create Monitoring User, click Yes.</td>
</tr>
<tr>
<td></td>
<td>2. Select the Authentication method, and enter the user name and password.</td>
</tr>
<tr>
<td></td>
<td>To specify an existing account:</td>
</tr>
<tr>
<td></td>
<td>1. Next to Create Monitoring User, click No.</td>
</tr>
<tr>
<td></td>
<td>2. Enter the user name and password.</td>
</tr>
<tr>
<td></td>
<td>SolarWinds DPA requires the monitoring user to have SA_ROLE and MON_ROLE privileges for data collection.</td>
</tr>
<tr>
<td></td>
<td>SolarWinds DPA ignores data on the monitored database instance from the specified monitoring user. Make sure the monitoring user will not cause load on the monitored instance.</td>
</tr>
</tbody>
</table>
If your repository database is not Oracle, the wizard skips this step.

Choose the tablespace in the repository database to store DPA performance data for this monitored instance.

By default, the performance data is stored in the default tablespace of the repository user. However, data for monitored instances can be stored in separate tablespaces.

If you have no Alert Groups set up, or if this new database instance does not match the database type of the Alert Group, the wizard skips this step.

Alert Groups simplify alert configuration and help make alerting more consistent across the monitored database instances.

Select the Alert Groups you want the new database instance to join.

Review the information and click Register Database Instance.

Click Finish to return to the DPA homepage.

If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must activate a license to monitor the database instance.

### Register a DB2 database

To register a DB2 database for DPA to monitor:

1. On the DPA homepage, click Register DB Instance for Monitoring.
2. Under Self-Managed, click DB2 UDB.
3. Click Next.
4. Complete the wizard panels as described in the following table.

<table>
<thead>
<tr>
<th>PANEL</th>
<th>INSTRUCTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Repository Tablespace</td>
<td>If your repository database is not Oracle, the wizard skips this step.</td>
</tr>
<tr>
<td></td>
<td>Choose the tablespace in the repository database to store DPA performance</td>
</tr>
<tr>
<td></td>
<td>data for this monitored instance.</td>
</tr>
<tr>
<td></td>
<td>By default, the performance data is stored in the default tablespace of the</td>
</tr>
<tr>
<td></td>
<td>repository user. However, data for monitored instances can be stored in</td>
</tr>
<tr>
<td></td>
<td>separate tablespaces.</td>
</tr>
<tr>
<td>Select the Alert Groups</td>
<td>If you have no Alert Groups set up, or if this new database instance does</td>
</tr>
<tr>
<td></td>
<td>not match the database type of the Alert Group, the wizard skips this step.</td>
</tr>
<tr>
<td></td>
<td>Alert Groups simplify alert configuration and help make alerting more</td>
</tr>
<tr>
<td></td>
<td>consistent across the monitored database instances.</td>
</tr>
<tr>
<td></td>
<td>Select the Alert Groups you want the new database instance to join.</td>
</tr>
<tr>
<td>Summary</td>
<td>Review the information and click Register Database Instance.</td>
</tr>
<tr>
<td>Database Instance Registration Complete</td>
<td>Click Finish to return to the DPA homepage.</td>
</tr>
</tbody>
</table>

SolarWinds DPA requires the DB2 instance-wide parameter `{DFT_MON_STMT}` to be turned on to collect monitoring data. Follow the on-screen instructions to check and set the parameter.

If `{DFT_MON_STMT}` is set to OFF, you can use SolarWinds DPA to register the database instance. Later, you can set it to ON and restart the database instance during an approved maintenance window. In the meantime, the database shows a status of Idle.
## Enter Monitored Database Instance Connection Information

Enter the host name or IP address and port of the DB2 server.

Enter the DB2 database for SolarWinds DPA to monitor in the Database field. SolarWinds DPA collects information from all DB2 instances in a cluster configuration for the specified database.

If the connection information changes for the DB2 server, all databases on that instance must be updated separately through the Update Database Instance Connection Wizard.

SolarWinds DPA can monitor all databases in the specified instance, or an individual database.

- SolarWinds DPA 9.0 and later monitors all DB2 databases in the specified instance.
- Do you want to monitor a single database in an earlier version of SolarWinds DPA? See [Switch to DB2 instance-wide monitoring](#).
- To monitor a single database, each database must be registered separately through this wizard, even if multiple databases are contained on a single DB2 server instance.
- For instance-wide monitoring, one database must be registered for the SolarWinds DPA connection.

Enter SYSADM credentials for SolarWinds DPA to monitor the database instance.

Do you want more information on the DB2 permissions needed by SolarWinds DPA for the monitoring user? See [Required DB2 permissions needed by DPA for monitoring](#).

## Enter the Monitoring User

Create or specify the account that DPA will use to gather information. To ensure that the account has the required permissions, SolarWinds recommends creating a new account.

To create a new account:

1. Next to Create Monitoring User, click Yes.
2. Select the Authentication method, and enter the user name and password.

To specify an existing account:

1. Next to Create Monitoring User, click No.
2. Enter the user name and password.

SolarWinds DPA requires the monitoring user to have SA_ROLE and MON_ROLE privileges for data collection.

SolarWinds DPA ignores data on the monitored database instance from the specified monitoring user. Make sure the monitoring user will not cause load on the monitored instance.
**Panel** | **Instructions**
---|---
Oracle Repository Tablespace | If your repository database is not Oracle, the wizard skips this step.
| Choose the tablespace in the repository database to store DPA performance data for this monitored instance.
| By default, the performance data is stored in the default tablespace of the repository user. However, data for monitored instances can be stored in separate tablespaces.

Select the Alert Groups | If you have no Alert Groups set up, or if this new database instance does not match the database type of the Alert Group, the wizard skips this step.
| Alert Groups simplify alert configuration and help make alerting more consistent across the monitored database instances.
| Select the Alert Groups you want the new database instance to join.

Summary | Review the information and click Register Database Instance.

Database Instance Registration Complete | Click Finish to return to the DPA homepage.

If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must [activate a license](#) to monitor the database instance.

## Register a MySQL database

To register a MySQL database for DPA to monitor:

1. On the DPA homepage, click Register DB Instance for Monitoring.
2. Under Self-Managed, click MySQL.
3. Click Next.
4. Complete the wizard panels as described in the following table.
Enter the host name or IP address and port of the server.

SolarWinds DPA monitors all databases within the instance. If more than one instance exists on the server, you must register each instance separately in DPA.

SolarWinds DPA ignores data generated by the monitoring user on the monitored database instance. For this reason, do not specify a user that causes load on the monitored instance. SolarWinds recommends creating a separate account for the monitoring user.

To create a new account:

1. Click Provide a privileged user.
2. Enter the credentials of an existing user with privileges to create the monitoring user and to grant the required permissions.
   - The credentials for the privileged user are not used or stored after the registration.
   - The privileged user requires the CREATE USER permission and must be able to grant the following permissions:
     - `PROCESS` on `*.*`
     - `SELECT` & `UPDATE` on `performance_schema.*`
   - To enable the retrieval of query execution plans, this privileged user must be able to grant the following permissions:
     - `SELECT, INSERT, UPDATE, DELETE` on `*.*`
3. Enter credentials for the monitoring user. You can create a new user or use an existing one.

To specify an existing account:

1. Click Provide the monitoring user.
2. Enter credentials. DPA encrypts the password.

Alternatively, you can use the script that DPA provides to create a monitoring user.

1. Click Monitoring User Creation Script, and follow the on-screen instructions.
2. Copy the edited script to the MySQL console, and run it.
3. Provide this user as your monitoring user.

SolarWinds DPA ignores data on the monitored database instance from the specified monitoring user. Make sure the monitoring user will not cause load on the monitored instance.
<table>
<thead>
<tr>
<th>PANEL</th>
<th>INSTRUCTIONS</th>
</tr>
</thead>
</table>
| Oracle Repository Tablespace | If your repository database is not Oracle, the wizard skips this step.  
Choose the tablespace in the repository database to store DPA performance data for this monitored instance.  
By default, the performance data is stored in the default tablespace of the repository user. However, data for monitored instances can be stored in separate tablespaces. |
| Select the Alert Groups | If you have no Alert Groups set up, or if this new database instance does not match the database type of the Alert Group, the wizard skips this step.  
Alert Groups simplify alert configuration and help make alerting more consistent across the monitored database instances.  
Select the Alert Groups you want the new database instance to join. |
**MySQL Configuration for Monitoring**

Select a Typical or Custom configuration. SolarWinds recommends the Typical configuration.

- The DPA Recommended option is used for Performance Schema setup.
- `EXPLAIN` can be run on `SELECT` statements.

Select Custom to change the Performance Schema setup and to allow `EXPLAIN` to be run on different statements.

### Performance Schema setup

Specify what data the Performance Schema collects and maintains. This table shows which consumers and instruments each option enables.

> The MySQL Performance Schema must be enabled. If you select Leave As Is, verify that Global Instrumentation and Thread Instrumentation are enabled in the existing Performance Schema configuration.

<table>
<thead>
<tr>
<th>Option</th>
<th>Server Default</th>
<th>DPA Recommended</th>
<th>Detailed</th>
<th>Leave As Is</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consumer Global Instrumentation</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Consumer Thread Instrumentation</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Consumer Statement Digest</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Consumer Statement (Current)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Consumer Wait (Current)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Instrument Wait (Lock/*)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Instrument Wait (I/O table)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Instrument Wait (I/O/socket)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Instrument Wait (Synch/*)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
</tbody>
</table>
MySQL Configuration for Monitoring (continued)

✔ = Enabled.
NC = No change. DPA does not change the existing Performance Schema configuration.

* Values that are outside of the MYSQL_PERFORMANCE_SCHEMA configuration scope of DPA are not changed. For example, an instrument named stage exists in the MySQL Performance Schema. If you enable or disable that instrument, DPA will not change it.

Allow EXPLAIN to be run on

This section is displayed if you specified a privileged user to create the DPA monitoring user.

Select what type of statements you want DPA to collect execution plans for. The monitoring user can run EXPLAIN on the selected statement types.

Summary

Review the information and click Register Database Instance.

| Database Instance Registration Complete | Click Finish to return to the DPA homepage. |

If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must activate a license to monitor the database instance.

Register an Amazon RDS for Oracle database

To register an Amazon RDS for Oracle database for DPA to monitor:

1. On the DPA homepage, click Register DB Instance for Monitoring.
2. Under Amazon RDS, click Amazon RDS for Oracle.
3. Click Next.
4. Complete the wizard panels as described in the following table.

If registration fails because your DPA server cannot connect to the instance's server, see DPA database registration failure when attempting to register a database on an external network.
Enter Monitored Database Instance Connection Information

Amazon Relational Database Service (RDS) for Oracle database instances have three connection options:

- Direct Connect
- Transparent Network Substrate (TNS) Connect Descriptor
- Lightweight Directory Access Protocol (LDAP) or TNS Name

Direct Connect

Enter the Service Name or System Identifier (SID), host name or IP address, and port. The default port is 1521.

TNS Connect Descriptor

The Connect Descriptor value contains everything after NAME= in the tnsnames.ora file. The beginning (DESCRIPTION= is necessary. For example:

```
(DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = demo.confio.com)(PORT = 1521))(CONNECT_DATA =(SERVICE_NAME = demo)))
```

LDAP or TNS Name

To use this option, Oracle Name Resolution must be configured. For instructions, see [Connect to Oracle using name resolution](#).

After you configure Oracle Name Resolution, you can use the LDAP/TNS Name when registering additional monitored database instances.

RAC instances

If this is an Oracle Real Application Clusters (RAC) instance, there may be listener configuration changes needed if you are not listening on the physical IP address. SolarWinds recommends:

- If you are registering pluggable databases (PDBs) on a RAC instance, consider registering with the physical IP address of the host.
- If you are registering a non-PDB RAC instance, consider registering with the SID.
- If you are using the Service Name, consider using the physical IP address of the host. Do not use the virtual IP address (VIP) or the Oracle Single Client Access Name (SCAN) IP address.

DBA user

Enter DBA credentials for SolarWinds DPA to register the database instance.
<table>
<thead>
<tr>
<th><strong>Panel</strong></th>
<th><strong>Instructions</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter the Monitoring User</td>
<td>Create or specify the account that DPA will use to gather information. To ensure that the account has the required permissions, SolarWinds recommends creating a new account.</td>
</tr>
<tr>
<td></td>
<td>To create a new account:</td>
</tr>
<tr>
<td></td>
<td>1. Next to Create Monitoring User, click Yes.</td>
</tr>
<tr>
<td></td>
<td>2. Enter the user name and password.</td>
</tr>
<tr>
<td></td>
<td>3. Select a Tablespace and Temp Tablespace on the monitored database. This is primarily used for gathering Explain Plan data for monitored queries.</td>
</tr>
<tr>
<td></td>
<td>To specify an existing account:</td>
</tr>
<tr>
<td></td>
<td>1. Next to Create Monitoring User, click No.</td>
</tr>
<tr>
<td></td>
<td>2. Enter the user name and password.</td>
</tr>
<tr>
<td></td>
<td>If you create the user manually, DPA uses the default Tablespaces for that user.</td>
</tr>
<tr>
<td>Oracle Monitoring Information</td>
<td>If the monitored instance contains the Oracle E-Business Suite, SolarWinds DPA can collect additional information about the suite.</td>
</tr>
<tr>
<td></td>
<td>SolarWinds DPA can capture Oracle E-Business data to identify the screens, modules, and users generating the database requests. This gives you increased visibility into the causes of performance problems in the Oracle E-Business Suite, Oracle Enterprise Resource Planning (ERP), and Oracle Applications environments.</td>
</tr>
<tr>
<td></td>
<td>The SYS password option is not available for Amazon RDS instances.</td>
</tr>
<tr>
<td>Oracle Repository Tablespace</td>
<td>If your repository database is not Oracle, the wizard skips this step.</td>
</tr>
<tr>
<td></td>
<td>Choose the tablespace in the repository database to store DPA performance data for this monitored instance.</td>
</tr>
<tr>
<td></td>
<td>By default, the performance data is stored in the default tablespace of the repository user. However, data for monitored instances can be stored in separate tablespaces.</td>
</tr>
<tr>
<td>Select the Alert Groups</td>
<td>If you have no Alert Groups set up, or if this new database instance does not match the database type of the Alert Group, the wizard skips this step.</td>
</tr>
<tr>
<td></td>
<td>Alert Groups simplify alert configuration and help make alerting more consistent across the monitored database instances.</td>
</tr>
<tr>
<td></td>
<td>Select the Alert Groups you want the new database instance to join.</td>
</tr>
<tr>
<td>Summary</td>
<td>Review the information and click Register Database Instance.</td>
</tr>
<tr>
<td>Database Instance Registration Complete</td>
<td>Click Finish to return to the DPA homepage.</td>
</tr>
</tbody>
</table>
If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must [activate a license](#) to monitor the database instance.

## Register an Amazon RDS for SQL Server database

To register an Amazon RDS for SQL Server database for DPA to monitor:

1. On the DPA homepage, click Register DB Instance for Monitoring.
2. Under Amazon RDS, click Amazon RDS for SQL Server.
3. Click Next.
4. Complete the wizard panels as described in the following table.

If registration fails because your DPA server cannot connect to the instance's server, see [DPA database registration failure when attempting to register a database on an external network](#).

<table>
<thead>
<tr>
<th>PANEL</th>
<th>INSTRUCTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter Monitored Database Instance Connection Information</td>
<td>Enter the server name or IP address and port. SolarWinds DPA monitors all databases within the instance. If more than one instance exists on the server, you must register each instance separately in DPA. SSL is requested by default. If the server does not support SSL, a plain connection is used. Enter an Amazon RDS Master User for SolarWinds DPA to register the database instance. If you do not want to enter the Master User that created the database instance, use the SQL statement below to create a new Master User. Replace <code>dpa</code> with the new user name.</td>
</tr>
</tbody>
</table>

```sql
CREATE LOGIN [dpa] WITH PASSWORD=N'Password1';
GRANT ALTER ANY LOGIN TO dpa;
GRANT VIEW SERVER STATE TO dpa WITH GRANT OPTION;
GRANT VIEW ANY DEFINITION TO dpa WITH GRANT OPTION;
GRANT VIEW ANY DATABASE TO dpa WITH GRANT OPTION;
ALTER SERVER ROLE [processadmin] ADD MEMBER [dpa];
```
<table>
<thead>
<tr>
<th>PANEL</th>
<th>INSTRUCTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter the Monitoring User</td>
<td>Create or specify the account that DPA will use to gather information. To ensure that the account has the required permissions, SolarWinds recommends creating a new account.</td>
</tr>
<tr>
<td></td>
<td>To create a new account:</td>
</tr>
<tr>
<td></td>
<td>1. Click Yes.</td>
</tr>
<tr>
<td></td>
<td>2. Select SQL Server as the authentication method. (DPA cannot create a new Windows account.)</td>
</tr>
<tr>
<td></td>
<td>3. Enter a user name and password for the new account, or accept the default values.</td>
</tr>
<tr>
<td></td>
<td>To specify an existing account:</td>
</tr>
<tr>
<td></td>
<td>1. Click No.</td>
</tr>
<tr>
<td></td>
<td>2. Select either authentication method.</td>
</tr>
<tr>
<td></td>
<td>3. Enter the user name and password of an existing account.</td>
</tr>
<tr>
<td></td>
<td>For Windows authentication, enter <code>&lt;DOMAIN&gt;\&lt;username&gt;</code> in the Monitoring User field.</td>
</tr>
<tr>
<td></td>
<td>You can also authenticate using a Windows Computer Account.</td>
</tr>
<tr>
<td></td>
<td>For SQL Server authentication, only the user name is required. Do not specify a domain.</td>
</tr>
<tr>
<td>Oracle Repository Tablespace</td>
<td>If your repository database is not Oracle, the wizard skips this step.</td>
</tr>
<tr>
<td></td>
<td>Choose the tablespace in the repository database to store DPA performance data for this monitored instance.</td>
</tr>
<tr>
<td></td>
<td>By default, the performance data is stored in the default tablespace of the repository user. However, data for monitored instances can be stored in separate tablespaces.</td>
</tr>
<tr>
<td>Select the Alert Groups</td>
<td>If you have no Alert Groups set up, or if this new database instance does not match the database type of the Alert Group, the wizard skips this step.</td>
</tr>
<tr>
<td></td>
<td>Alert Groups simplify alert configuration and help make alerting more consistent across the monitored database instances.</td>
</tr>
<tr>
<td></td>
<td>Select the Alert Groups you want the new database instance to join.</td>
</tr>
<tr>
<td>Summary</td>
<td>Review the information and click Register Database Instance.</td>
</tr>
<tr>
<td>Database Instance Registration Complete</td>
<td>Click Finish to return to the DPA homepage.</td>
</tr>
</tbody>
</table>
If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must [activate a license](#) to monitor the database instance.

## Register an Amazon RDS for MySQL or Aurora database

To register an Amazon RDS for MySQL or Aurora database for DPA to monitor:

1. On the DPA homepage, click Register DB Instance for Monitoring.
2. Under Amazon RDS, click Amazon RDS for MySQL.
3. Click Next.
4. Complete the remaining wizard panels as described in the following table.

If registration fails because your DPA server cannot connect to the instance's server, see [DPA database registration failure when attempting to register a database on an external network](#).
Enter the host name or IP address and port of the server.

SolarWinds DPA monitors all databases within the instance. If more than one instance exists on the server, you must register each instance separately in DPA.

SolarWinds DPA ignores data generated by the monitoring user on the monitored database instance. For this reason, do not specify a user that causes load on the monitored instance. SolarWinds recommends creating a separate account for the monitoring user.

To create a new account:

1. Click Provide a privileged user.
2. Enter the credentials of an existing user with privileges to create the monitoring user and to grant the required permissions.
   The credentials for the privileged user are not used or stored after the registration.
   The privileged user requires the CREATE USER permission and must be able to grant the following permissions:
   ```sql
   PROCESS on *.*
   SELECT & UPDATE on performance_schema.*
   ```
   To enable the retrieval of query execution plans, this privileged user must be able to grant the following permissions:
   ```sql
   SELECT, INSERT, UPDATE, DELETE on *.*
   ```
3. Enter credentials for the monitoring user. You can create a new user or use an existing one.

To specify an existing account:

1. Click Provide the monitoring user.
2. Enter credentials. DPA encrypts the password.

Alternatively, you can use the script that DPA provides to create a monitoring user.

1. Click Monitoring User Creation Script, and follow the on-screen instructions.
2. Copy the edited script to the MySQL console, and run it.
3. Provide this user as your monitoring user.

SolarWinds DPA ignores data on the monitored database instance from the specified monitoring user. Make sure the monitoring user will not cause load on the monitored instance.
<table>
<thead>
<tr>
<th>PANEL</th>
<th>INSTRUCTIONS</th>
</tr>
</thead>
</table>
| Oracle Repository Tablespace | If your repository database is not Oracle, the wizard skips this step.  
Choose the tablespace in the repository database to store DPA performance data for this monitored instance.  
By default, the performance data is stored in the default tablespace of the repository user. However, data for monitored instances can be stored in separate tablespaces. |
| Select the Alert Groups      | If you have no Alert Groups set up, or if this new database instance does not match the database type of the Alert Group, the wizard skips this step.  
Alert Groups simplify alert configuration and help make alerting more consistent across the monitored database instances.  
Select the Alert Groups you want the new database instance to join. |
MySQL Configuration for Monitoring

Select a Typical or Custom configuration. SolarWinds recommends the Typical configuration.

The DPA Recommended option is used for Performance Schema setup. Select Custom to change the Performance Schema setup.

Performance Schema setup

Specify what data the Performance Schema collects and maintains. This table shows which consumers and instruments each option enables.

The MySQL Performance Schema must be enabled. If you select Leave As Is, verify that Global Instrumentation and Thread Instrumentation are enabled in the existing Performance Schema configuration.

<table>
<thead>
<tr>
<th>Option</th>
<th>Server Default</th>
<th>DPA Recommended</th>
<th>Detailed</th>
<th>Leave As Is</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consumer Global Instrumentation</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Consumer Thread Instrumentation</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Consumer Statement Digest</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Consumer Statement (Current)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Consumer Wait (Current)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Instrument Wait (Lock/*)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Instrument Wait (I/O table) (I/O/file)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Instrument Wait (I/O/socket)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
<tr>
<td>Instrument Wait (Synch/*)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>NC</td>
</tr>
</tbody>
</table>
MySQL Configuration for Monitoring (continued)

✔ = Enabled.
NC = No change. DPA does not change the existing Performance Schema configuration.

* Values that are outside of the MYSQL_PERFORMANCE_SCHEMA configuration scope of DPA are not changed. For example, an instrument named stage exists in the MySQL Performance Schema. If you enable or disable that instrument, DPA will not change it.

Summary Review the information and click Register Database Instance.

Database Instance Registration Complete

Click Finish to return to the DPA homepage.

If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must activate a license to monitor the database instance.

Register an Azure SQL database

To register multiple Azure SQL databases using the Mass Registration feature, follow the instructions in this KB article.

To register a single Azure SQL database for DPA to monitor:

1. On the DPA homepage, click Register DB Instance for Monitoring.
2. Under Azure, click Azure SQL DB.
3. Click Next.
4. Complete the wizard panels as described in the following table.
**Enter Monitored Database Instance Connection Information**

Enter the server name, port, and database name. You cannot use an IP address in the Server Name field.

Choose a method for creating or configuring the monitoring user.

To create a new account:

1. Click Let DPA create a new contained user or configure an existing contained user for me.
2. Enter the credentials of an existing user with privileges to create the monitoring user and to grant the required permissions.
   - The privileged user must be a member of the db_owner role.
   - The credentials for the privileged user are not used or stored after the registration.

To specify an existing account:

1. Click I'll create the database user.
2. Enter credentials. DPA encrypts the password.

Alternatively, you can use the script that DPA provides to create a monitoring user.

1. Click Monitoring User Creation Script, and follow the on-screen instructions.
2. Copy and run the edited script on your Azure SQL database.
3. Provide this user as your monitoring user.
<table>
<thead>
<tr>
<th>PANEL</th>
<th>INSTRUCTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter the Monitoring User</td>
<td>DPA gathers information through this user from the monitored database. You can create a monitoring user through DPA or use an existing user, such as for read-only replica databases.</td>
</tr>
<tr>
<td></td>
<td>To register a read-only geo-replica, you must create a monitoring account through the primary server first.</td>
</tr>
<tr>
<td></td>
<td>SolarWinds recommends creating a new account because DPA requires special permissions that existing users may not have.</td>
</tr>
<tr>
<td></td>
<td>To create a new account:</td>
</tr>
<tr>
<td></td>
<td>1. Click Let DPA create a new contained user.</td>
</tr>
<tr>
<td></td>
<td>2. Enter credentials.</td>
</tr>
<tr>
<td></td>
<td>To specify an existing account:</td>
</tr>
<tr>
<td></td>
<td>1. Run the following SQL statement on the Azure SQL database:</td>
</tr>
<tr>
<td></td>
<td>CREATE USER [USERNAME] WITH PASSWORD=N'PASSWORD'; ALTER ROLE db_owner ADD member &lt;USERNAME&gt;;</td>
</tr>
<tr>
<td></td>
<td>2. Click Let DPA configure an existing contained user.</td>
</tr>
<tr>
<td></td>
<td>3. Enter credentials.</td>
</tr>
<tr>
<td></td>
<td>Oracle Repository Tablespace</td>
</tr>
<tr>
<td></td>
<td>If your repository database is not Oracle, the wizard skips this step.</td>
</tr>
<tr>
<td></td>
<td>Choose the tablespace in the repository database to store DPA performance data for this monitored instance.</td>
</tr>
<tr>
<td></td>
<td>By default, the performance data is stored in the default tablespace of the repository user. However, data for monitored instances can be stored in separate tablespaces.</td>
</tr>
<tr>
<td></td>
<td>Select the Alert Groups</td>
</tr>
<tr>
<td></td>
<td>If you have no Alert Groups set up, or if this new database instance does not match the database type of the Alert Group, the wizard skips this step.</td>
</tr>
<tr>
<td></td>
<td>Alert Groups simplify alert configuration and help make alerting more consistent across the monitored database instances.</td>
</tr>
<tr>
<td></td>
<td>Select the Alert Groups you want the new database instance to join.</td>
</tr>
<tr>
<td></td>
<td>Summary</td>
</tr>
<tr>
<td></td>
<td>Review the information and click Register Database Instance.</td>
</tr>
<tr>
<td></td>
<td>Database Instance Registration Complete</td>
</tr>
<tr>
<td></td>
<td>Click Finish to return to the DPA homepage.</td>
</tr>
</tbody>
</table>
Enable deadlocks for read-only geo-replicas

To enable the deadlock feature for read-only geo-replica Azure SQL databases, you must create and enable an Extended Event Session (EES).

If you registered the primary server first, an EES is already created and synced. Skip to step 2.

Otherwise, connect to the primary server first to create an EES.

1. Run the following SQL statement:

   ```sql
   CREATE EVENT SESSION [dpa_deadlock_capture] ON DATABASE
   ADD EVENT sqlserver.xml_deadlock_report
   ADD TARGET package0.ring_buffer(SET max_events_limit=(1000),
       max_memory=(256))
   WITH (MAX_MEMORY = 256KB,
       EVENT_RETENTION_MODE = ALLOW_SINGLE_EVENT_LOSS,
       MAX_DISPATCH_LATENCY = 30 SECONDS,
       MAX_EVENT_SIZE = 0KB,
       MEMORY_PARTITION_MODE = NONE,
       TRACK_CAUSALITY = OFF,
       STARTUP_STATE = ON);
   -- ALTER EVENT SESSION [dpa_deadlock_capture] ON DATABASE STATE = START;
   ```

2. Connect to the read-only replica database.

3. Click Extended Events > Sessions.

4. Enable the dpa_deadlock_capture session.

If you register a database instance within the 14-day trial period, DPA begins monitoring the instance immediately. After the trial period, you must activate a license to monitor the database instance.
Unregister a monitored database instance

If you want to remove one of your monitored database instances from DPA, you must unregister it.

⚠️ If you unregister a monitored database instance, DPA stops monitoring the instance and removes all historical performance data from the repository.

1. In the DPA menu bar, click Options.
2. Under Monitor Setup > Database Instances, click Unregister DB Instance.
3. Select a database instance, and click Next.
4. Select the database instance objects to be removed, and click Next.
   Depending on the database type, you can remove one or both of the following objects:
   - **Monitoring User**: You can remove the monitoring user if no other applications, including other installations of DPA, are using this user.
   - **DPA Database Objects**: This refers to tables that are created in the schema of the monitoring user. If you remove the monitoring user, these objects are removed since they are owned by the monitoring user. You can remove these objects if no other installations of DPA are monitoring this instance.

⚠️ You cannot remove objects on certain database types, such as read-only replicas.

5. Confirm the unregistration information, and click Unregister Database Instance. This may take several minutes.
6. Click Finish to complete the unregistration.
Database instance groups

See the following topics for information about creating and monitoring groups in SolarWinds DPA:

- **About monitoring SQL Server Availability Groups with DPA** describes the information that DPA provides about SQL Server AGs.
- **About monitoring Oracle multitenent databases (CDBs)** describes the information that DPA provides about Oracle CDBs.
- **Manually group database instances** describes how to create and modify custom groups.

About monitoring SQL Server Availability Groups with DPA

DPA provides status information, annotations, and alerts for your SQL Server Availability Groups (AGs).

For information about options for registering SQL Server AGs, see [Registration and licensing options for clustered environments](#).

DPA does not support monitoring distributed AGs. DPA can monitor the SQL Server instances that participate in a distributed AG, but the AG monitoring features are not enabled for distributed AGs.

Automatic naming

When you [register an AG listener](#), DPA automatically names the instance using the following format:

<PrimaryReplicaName> via <ListenerName>

When a failover occurs, the name is automatically updated to reflect the new primary replica.

If you manually change the display name of an AG that is registered via the listener, by default DPA overwrites the name each time the monitor starts. To change the default behavior and manually specify the name, [change the advanced option](#) AG_INSTANCE_NAME_UPDATE_ENABLED.

AG information in DPA

On the DPA homepage, the AG Status Summary box in the Status Summary area shows the number of database instances with partially healthy or not healthy AGs. As with other status boxes, click the filter next to a status to display only instances associated with the selected status.
The AG status icon identifies database instances that include AGs. The color of the dot provides status information (described in the following section). Possible statuses are:

- Green for healthy
- Yellow for partially healthy
- Red for not healthy
- Gray for unknown

To view detailed information:

1. From the DPA homepage, click an AG status icon to open the Availability Group Summary view. This view shows information about each AG in the database instance. DPA shows status information for primary replicas in the instance. For secondary replicas, the status of the primary replica is displayed if DPA is also monitoring the primary replica.
2. Click any link to view detailed information about the databases and replicas in the AG.

How DPA determines the AG status

If the instance is monitored directly and acting as a primary replica

If you are monitoring the instance directly (not through a listener), DPA looks at the status of all AGs that the instance acts as the primary replica for, and displays the worst status.

**Example:** An instance is acting as the primary replica for four availability groups. Their statuses are:

- AG1: Healthy
- AG2: Healthy
- AG3: Partially Healthy
- AG4: Not Healthy

DPA shows the status as **Not Healthy**.

<table>
<thead>
<tr>
<th>AG1</th>
<th>AG2</th>
<th>AG3</th>
<th>AG4</th>
<th>DPA status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Healthy</td>
<td>Healthy</td>
<td>Partially Healthy</td>
<td>Not Healthy</td>
<td>Not Healthy</td>
</tr>
</tbody>
</table>

If the instance is monitored directly and acting as a secondary replica

If an instance is acting as a secondary replica for any AGs, that AG's status is Unknown. If the instance also acts as a primary replica for one or more AGs, the Unknown status is ignored.

**Example:** An instance acts as the primary replica for three availability groups. Their statuses are:

- AG1: Healthy
- AG2: Healthy
- AG3: Partially Healthy

The instance also acts as a secondary replica for one AG. Its status is Unknown.
DPA ignores the Unknown status, and shows the status as **Partially Healthy.**

<table>
<thead>
<tr>
<th>AG1</th>
<th>AG2</th>
<th>AG3</th>
<th>AG4</th>
<th>DPA status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Healthy</td>
<td>Healthy</td>
<td>Partially Healthy</td>
<td>Unknown</td>
<td>Partially Healthy</td>
</tr>
</tbody>
</table>

If DPA shows the AG status as Unknown, that typically indicates that the instance is acting as a secondary replica for all AGs.

**If the instance is monitored via the listener**

If you are monitoring the instance via the listener, by default DPA displays the aggregate status as described above. However, you can change the advanced option AG_STATUS_ROLLUP_USE_PRIMARY to determine the status using only the AG associated with the listener.

**AG alerts**

DPA provides the following AG alerts: SQL Server Availability Group Failover and SQL Server Availability Group Status Change.

**SQL Server Availability Group Failover**

This alert is triggered when an AG failover occurs. DPA sends alerts based on how you registered instances:

- If you registered database instances directly (not through a listener), when a failover occurs DPA sends an alert for each instance involved in the failover that it is monitoring. For example, if an AG fails over from Instance1 to Instance2 and DPA is monitoring both instances, you receive two alerts. If DPA is monitoring only one of the instances, you receive only one alert.
- If you registered the AG through a listener and the AG associated with the listener fails over, DPA sends one alert, because the listener moves with the AG from Instance1 to Instance2.

If multiple AG failovers occur in a short period of time, DPA aggregates them into one alert per instance.

**SQL Server Availability Group Status Change**

This alert is triggered when an AG status changes to Partially Healthy or Not Healthy. DPA evaluates AG statuses every 10 minutes by default. You are alerted if the status changes from Healthy to Partially Healthy or Not Healthy between the evaluations. If the status changes from Healthy to another status and then back to Healthy during the same evaluation period, you are not alerted.

**Example:** In this example, DPA is monitoring an instance that acts as a primary replica for three AGs. The following table shows how the alerts would behave for each of DPA's alert policies. The intervals are arranged from most noisy to least noisy.
<table>
<thead>
<tr>
<th>INTERVAL</th>
<th>AG STATUS (CHANGE IN RED)</th>
<th>POLICY: NOTIFY WHEN LEVEL NOT VISITED SINCE NORMAL</th>
<th>POLICY: NOTIFY WHEN LEVEL CHANGES AND IS NOT NORMAL</th>
<th>POLICY: NOTIFY WHEN LEVEL IS NOT NORMAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AG1 Healthy</td>
<td>No alert</td>
<td>No alert</td>
<td>No alert</td>
</tr>
<tr>
<td></td>
<td>AG2 Healthy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AG3 Healthy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>AG1 Healthy</td>
<td>No alert</td>
<td>No alert</td>
<td>No alert</td>
</tr>
<tr>
<td></td>
<td>AG2 Healthy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AG3 Healthy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>AG1 Healthy</td>
<td>AG2 Not Healthy</td>
<td>AG2 Not Healthy</td>
<td>AG2 Not Healthy</td>
</tr>
<tr>
<td></td>
<td>AG2 Not Healthy</td>
<td>AG3 Not Healthy</td>
<td>AG3 Not Healthy</td>
<td>AG3 Not Healthy</td>
</tr>
<tr>
<td></td>
<td>AG3 Not Healthy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>AG1 Healthy</td>
<td>No alert</td>
<td>AG2 Partially Healthy</td>
<td>AG2 Partially Healthy</td>
</tr>
<tr>
<td></td>
<td>AG2 Partially Healthy</td>
<td>AG3 Partially Healthy</td>
<td>AG3 Partially Healthy</td>
<td>AG3 Partially Healthy</td>
</tr>
<tr>
<td></td>
<td>AG3 Partially Healthy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>AG1 Healthy</td>
<td>AG2 Partially Healthy</td>
<td>AG2 Partially Healthy</td>
<td>AG2 Partially Healthy</td>
</tr>
<tr>
<td></td>
<td>AG2 Partially Healthy</td>
<td>AG3 Healthy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>AG1 Healthy</td>
<td>AG3 Partially Healthy</td>
<td>AG3 Partially Healthy</td>
<td>AG2 Partially Healthy</td>
</tr>
<tr>
<td></td>
<td>AG2 Partially Healthy</td>
<td>AG3 Partially Healthy</td>
<td>AG3 Partially Not Healthy</td>
<td>AG3 Partially Not Healthy</td>
</tr>
<tr>
<td>7</td>
<td>AG1 Partially Healthy</td>
<td>AG1 Partially Healthy</td>
<td>AG1 Partially Healthy</td>
<td>AG1 Partially Healthy</td>
</tr>
<tr>
<td></td>
<td>AG2 Healthy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AG3 Healthy</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Automatic annotations when AG failovers occur

Annotations are automatically added to wait time charts when an AG failover occurs. The annotations allow you to compare changes in performance before and after a failover.

The number of annotations depends on how you registered instances:

- If you registered database instances directly (not through a listener), when a failover occurs DPA adds an annotation for each instance involved in the failover that it is monitoring. For example, if an AG fails over from Instance1 to Instance2 and DPA is monitoring both instances, DPA adds two annotations. If DPA is monitoring only one of the instances, DPA adds only one annotation.
- If you registered the AG through a listener and the AG associated with the listener fails over, DPA adds one annotation.

If you do not want to add an annotation when an AG failover occurs, change the value of the [advanced system option](#) _AG_EVENT_ANNOTATIONS_ ENABLED._

### About monitoring Oracle multitenent databases (CDBs)

#### Registration and automatic grouping

To monitor an Oracle multitenant container database (CDB), register the pluggable databases (PDBs) contained in the CDB. Register each PDB just as you would register an Oracle single tenant database. For more information, see Registration and licensing options for clustered environments.

When you register two or more Oracle PDBs in the same CDB, DPA automatically creates a group for the CDB. This group is used for all registered PDBs from the CDB. If a DBA moves a PDB to a new CDB, SolarWinds DPA processes and groups the instance.

#### View the PDB load

On the DPA homepage, click the CDB to view summary data across the PDB instances. Use this view to determine which PDB has the most wait time, and what types of waits the PDBs are experiencing.
Automatic annotations

Annotations are automatically added to wait time charts when a PDB is added, removed, or moved from one CDB to another. The annotations allow you to compare performance before and after the change.

Turn off automatic grouping of Oracle CDBs

If you do not want DPA to automatically group the PDBs within a CDB, you can turn automatic grouping off.

1. Click Options.
2. Under Administration > Configuration, click Advanced Options.
3. Click the ORACLE_CDB_AUTO_GROUP system option.
4. Select False from the New Value list, and click Update.

After you set this option to false, grouping of registered database instances does not change. Only newly registered or updated database instances are affected, and are not grouped.

Manually group database instances

SolarWinds DPA automatically groups Oracle Real Application Clusters (RAC) instances and Oracle multitenant container databases (CDB) containing pluggable databases (PDB). You can manually group other database instances so that they are displayed together on the DPA homepage. For example, you can create groups based on type or location.

A database instance can be included in only one group.

Create a custom group

1. On the DPA homepage above the list of database instances, click Group Settings.
   The Manage Instance Groups dialog box lists the existing groups.
2. Click Add.
   The Add Instance Group dialog box lists the database instances that are not members of an existing group.
3. Enter a name, select the database instances to include, and then click Add.

![Add Instance Group]

Show or hide groups on the DPA homepage

Toggle the Show Groups button above the list of database instances to show or hide groups.

- When you show groups (the default), the DPA homepage lists ungrouped database instances first, followed by groups in alphabetical order. You can expand or collapse each group.

```
  Boulder Data Center (3)

  Chicago Data Center (2)

  ON       DPASY16SP2:5000
  ON       JOHNLENNON via MUSICDB-LIST
```

- When you hide groups, the DPA homepage lists database instances alphabetically.
Monitor database instances with DPA

The DPA homepage displays a list of the monitored database instances with status and wait time information. Click Action to start or stop monitoring.

Monitoring is always active after it is started. It is not necessary to restart the SolarWinds DPA monitor if the repository instance or the monitored database instance was unavailable for a period of time. Monitoring resumes when both are available again.

If there is a period of time when monitoring should not occur, you can stop monitoring a database instance.

If you are having problems connecting to or monitoring a database instance, see Troubleshooting tips.
Update a monitored database instance

If connection or user information changes with one of your monitored database instances, you must update that information in DPA.

1. In the DPA menu bar, click Options.
2. Under Monitor Setup > Database Instances, click Update Connection.
3. Select the database instance, and click Next.
4. Select the check box next to the property, update the value, and click Next.

   For database-specific connection information, see the following:
   - Oracle
   - SQL Server
   - Azure SQL
   - Sybase
   - DB2
   - MySQL
   - Amazon RDS for Oracle
   - Amazon RDS for SQL Server
   - Amazon RDS for MySQL

5. Confirm the connection information, and click Update Connection.
6. Click Finish, or Update Another Database Instance to continue updating.

Stop monitoring a database instance for a period of time

A blackout is a period of time when SolarWinds DPA stops monitoring a certain database instance.

1. In the DPA menu bar, click Options.
2. Under Monitor Setup > Database Instances, click Monitor Blackout Periods.
3. Select a database instance from the list on top.
4. Set a day and time to stop and start monitoring, and click Add New Blackout Period.
Troubleshooting tips

Logs

DPA logs information about each monitored database instance. Use this information to help you determine why a database instance is not being monitored, or if data are missing.

Access log data through the DPA log viewer

Use the DPA Log Viewer to view log information for a specific database instance, or for all database instances and the DPA repository.

1. Open the Log Viewer:
   - To display log messages for a specific database instance:
     From the DPA homepage, click Action > Log next to the database instance.
   - To display messages for all monitored database instances and the DPA repository:
     In the DPA menu bar, click Options. Then, under Support > Utilities, click Log Viewer.

2. Use any of the following options to locate information:
   - Use filters to help you find specific information. To change the filters, click Advanced and select the filter criteria. For example, you can filter by date range, a text string, or message level.
   - For any message above Info, click Details to view additional information from the log.
   - Click Log Files for Support to create a compressed file you can send to SolarWinds Support.

Open log files in a text editor

Log files are stored in the installDir/iwc/tomcat/logs/ directory.

Access to a database instance

If DPA cannot access the server that hosts a database instance you want to monitor:

- Make sure a firewall is not running on the server.
- Make sure another process is not using the default SolarWinds DPA ports.

If the ports are being used by another process, you can change the default ports of 8123, 8124, and 8127. To specify different ports for DPA to use:

1. Open the following file in a text editor:
   
   installDir/iwc/tomcat/conf/server.xml
2. Update the following lines with new port numbers:

```xml
<Server port="8127" shutdown="SHUTDOWN">
  <Connector port="8123"/>
  <Connector port="8124"/>
</Server>
```

3. Save the file and restart DPA.

Issues after the Oracle PDB that stores the repository is moved

If the DPA repository is created on an Oracle pluggable database (PDB), you might experience the following issues after the PDB is moved to a different container database (CDB).

DPA returns a connection error

The PBD moved to a CDB on a different server, and the connection string is incorrect. Update the connection string in the `repo.properties` file in the following location:

```bash
<DPA_Install_Dir>\iwc\tomcat\ignite_config\iwc\repo.properties
```

DPA returns an invalid login error

Verify that the DPA monitoring user exists in the CDB. Common users (prefaced with C##) exist in only one CDB.
Investigate performance issues with DPA

DPA uses an approach called wait-based analysis to help you focus on issues that provide the greatest performance improvements. Query advisors and table tuning advisors identify performance issues and help you find the root cause:

- Access DPA query or table tuning advisors
- Use DPA's query performance analysis to find the root cause of performance issues
- Investigate inefficient queries running against a table
- Identify blocking sessions and deadlocks
- Find and investigate unusually long wait times (anomalies)
- About anomaly detection in DPA

See a walk-through of using DPA to investigate an application issue in the DPA Getting Started Guide.

Access DPA query or table tuning advisors

DPA provides two types of advisors:

- **Query advisors** provide information to help you improve the performance of a specific query, including what type of waits were responsible for significant wait time, whether the statement was blocked by other sessions, and whether execution plans include potentially expensive steps such as full table scans.

- **Table tuning advisors** are generated when a significant number of inefficient queries run against a table. These advisors provide aggregated information about the table, the inefficient queries that ran against it, and any existing indexes.

The Tuning column on the DPA homepage displays a warning or critical icon when advisors with a warning or critical status are available for a database instance. A green check mark in this column indicates that there are no advisors or that all advisors are informational.

<table>
<thead>
<tr>
<th>Database Instance</th>
<th>Wait</th>
<th>Tuning</th>
<th>CPU</th>
<th>Mem</th>
<th>Disk</th>
<th>Sess</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVANTIA@BOULDER</td>
<td>Action▼</td>
<td>![yellow]</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>AVANTIO</td>
<td>Action▼</td>
<td>![red]</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

View all advisors for a database instance

To view all advisors for a database instance, do either of the following to open the Tuning Advisors page:

- From the DPA homepage, click the icon in the Tuning column.
- If you have drilled in to view information about a database instance, click the Tuning tab in the top-
right corner of the instance details page.

| TRENDS | TUNING | CURRENT | RESOURCES |

A red or yellow bar on the Tuning tab indicates that critical or warning advisors are available.

The Tuning Advisors page displays the latest query and table tuning advisors. Use the drop-down menu at the top of the page to display advisors generated for a previous date.

Query advisors are calculated every hour. Table tuning advisors are calculated once a day, at the end of the day. The most recent table tuning advisors are for the previous day.

Open an advisor

For detailed information to help you resolve performance issues:

- Click a query advisor to open the Query Detail page, which displays detailed information about the query, along with the most relevant statistics and metrics charts.
- Click a table tuning advisor to open the Table Tuning Advisor page, which displays aggregated information about the table and the inefficient queries that ran against it.

Use DPA's query performance analysis to find the root cause of performance issues

To help you investigate the root cause of a query's performance problems, DPA intelligently assembles the most relevant data about the query and displays it on the Query Details page. Use the Query Details page to:

- View waits, statistics, and metrics from any time period
- See what type of waits are affecting performance
- Review query and table tuning advisors
- Examine statistics and metrics charts to correlate query wait times with other events

See an example of using the Query Details page to investigate an increase in wait time.

Open the Query Details page

Click the SQL hash or name in any chart legend to open the Query Details page.
Select a time period

All data on the Query Details page reflects the selected time period, which is displayed at the top of the page.

When you open the Query Details page, it defaults to the time period selected for the previous chart. For example, if you open the Query Details page while viewing the Top SQL Statements for one day, the Query Details page shows data for that day.

To select a different time period, you can:

- Click a bar to drill in to that time period.
- Click the date range at the top of the page to open the date picker. Then select a predefined time period or enter specific dates.

See what type of waits are affecting performance

The Top Waits chart at the top of the page shows the query's execution time for the selected time period. The bars are color-coded by the type of wait. Knowing what type of waits are causing the performance issue can help you determine how to fix the issue.

On this chart, you can:

- Click the 🔍 next to an entry in the legend to display detailed information about that type of wait, including possible resolutions.
Hover over an entry in the legend to dim other waits in the chart and better visualize the impact of this type of wait.

Review query and table tuning advisors

The Query Advisors section shows the latest advice for the selected time period. Query advisors provide information such as:

- What type of wait activities the SQL statement spend significant time on.
- Whether the statement was blocked by other sessions.
- Whether the statement took longer than normal to execute.
- If multiple execution plans were used, or if plans include potentially expensive steps such as full table scans.

If any table tuning advisors included information about this query, you can click through for aggregated information about the table and all inefficient queries that ran on it.
Correlate query wait times with other events

To help you find the root cause of performance issues, the Query Details page includes the most relevant statistics, blocking, plan, and metrics charts. Sections with data to display are automatically expanded. Other sections are collapsed by default. For example, if there is no blocking data, the Blocking section is collapsed.

When you scroll down to view these charts, the Top Waits chart at the top of the page remains visible so you can correlate query wait times with other events during the same time period.

DPA uses the predominant type of wait and other information to automatically select the most relevant charts. For example, if the predominant type for an Oracle database instance is Memory/CPU, DPA includes charts such as OS/CPU Utilization, CPU Utilization by DB, and Buffer Cache Hit Ratio.

The predominant type of wait is the type responsible for the majority of the time that a query spent waiting during the specified period.

To be considered predominant, the type must be responsible for more than a certain percent of the total wait time for that period. By default, this threshold is 20%. You can change the threshold by changing the advanced option PREDOMINANT_WAIT_THRESHOLD.

You can manually select other statistics or metrics charts to include.
Display other statistics charts

1. If the Statistics section is collapsed, expand it.
2. On the right side of the Statistics section, click Add Statistic.
3. Select the statistics you want to include, and deselect any you want to remove.

![Add Statistic](Image)

4. Click outside the drop-down to close it.

Display other metrics charts

1. If the Instance Resource Metrics section is collapsed, expand it.
2. On the right side of the section, click Add Metrics.
   The Add Metrics dialog box opens.
3. Filter or sort the list to locate the metrics you want to add:
   - Select one or more categories to filter by those categories.

![Categories](Image)

   - Enter a string in the Search box to show only metric names containing that string. (Wildcards are not supported.)

```
read
```

- Sort by name or by category.
4. Select one or more metrics. Use the selection drop-down menu to quickly select multiple metrics:
   - All: Selects all metrics on the current page.
   - All Pages: Select all metrics on all pages.

5. Click Save Changes to display the selected metrics.

Investigate inefficient queries running against a table

Inefficient queries—that is, queries that perform a large number of reads but return a relatively small number of rows—can significantly add to database performance issues. These queries do a large amount of work for little return. This type of inefficiency results in higher I/O, longer wait times, greater amounts of blocking, and increased resource contention.

Possible solutions include tuning the query, adding an index, or adding columns to an existing index. DPA's table tuning advisors help you make informed decisions about the best course of action.

See the following sections for tips on using the information in each table tuning advisor:

- What are table tuning advisors?
- Open a table tuning advisor
- Quick start
- Examine the list of inefficient queries
- Examine query details
- Examine table statistics
- Examine index details

What are table tuning advisors?

At the end of each day, DPA runs an analysis to identify tables that had inefficient queries run against them during that day. For each of these tables, the Table Tuning Advisor page displays aggregated information about the table, the inefficient queries that ran against it, and any existing indexes. This information helps you optimize query performance while taking indexing trade-offs into account.

Table tuning advisors are available for Oracle, SQL Server (2008 and above), and Azure SQL databases. Table tuning advisors are calculated at the end of each day. Therefore, the most recent table tuning advisors are for the previous day.
Open a table tuning advisor

The **Tuning tab** lists all table tuning advisors for the selected database instance. Click a table tuning advisor to open it.

Quick start

Each table tuning advisor provides detailed information, as described in the following sections. Use the following suggestions to get started:

1. **Update statistics.**
   
   In the Existing Indexes section, look at the age of the index statistics. If the statistics are stale, especially if table churn is high, the optimizer does not have the best information to make good plan choices. Updating statistics is often a good first step before you do any further analysis.

2. **Evaluate indexes.**
   
   Click on several of the top inefficient queries and do the following:
• Review the SQL text to learn more about the WHERE clauses and JOIN conditions that can affect query performance.

• (SQL Server and Azure only) If plans with SQL Server’s index recommendations are provided, consider adding them or extending existing indexes to satisfy them.

• If plans with inefficient table or index access steps are provided:
  ○ Review each plan section and the predicates for each step. The columns in the predicates are candidates for indexes.
  ○ Check for warnings (shown as links below the step if they are detected) and consider their recommendations.
  ○ Consider indexing the candidate columns found across the SQL statements examined:
    ○ Is there an index that might benefit several queries?
    ○ Is there an existing index that could be extended to benefit one or more queries?

3. **Resolve fragmentation.**

   Review the table’s row count, churn, and index fragmentation. For larger tables, consider the following:
   
   • If fragmentation is high, defragmenting the indexes might help resolve performance problems when plan steps are using scan operations.
   • If churn is also high, consider defragmenting the index more frequently.

Examine the list of inefficient queries

The top-left pane lists the inefficient queries that ran against the table on the selected day. DPA assigns a relative efficiency score to each query and uses this score sort the list.

Select a query from this list to display detailed information about it.
Tips for using this information

- Focus your tuning efforts on the queries at the top of the list, which are driving the most inefficient workload against this table.
- A large number of queries in the list could indicate a more widespread performance issue. Perhaps one good index could improve the performance of several similar queries.

Examine query details

The upper-right pane displays information about the selected query that can help you determine the source of read inefficiencies against this table.

![Query Details](image)

1. The performance statistics at the top of the pane show the extent of the query's inefficiency:
   - Reads per Exec is the number of read I/O operations per execution, which indicates how much work the query is doing.
   - Rows per Exec is the number of rows the query returns.
   - The Reads per Row ratio is the number of reads the query needed to do in order to arrive at each row in the query's result set. Statements with the highest Reads per Row ratios could potentially benefit most from tuning.

   For more information about the query, click the SQL name or hash value to view DPA's query performance analysis, which shows when the query ran, the execution statistics, and the most relevant metrics charts.

2. DPA lists each execution plan that it finds. You can click the link to examine the full plan, but DPA lists the steps most likely to need attention below.
(SQL Server and Azure only) Index recommendations made by the SQL Server optimizer, if any, are listed. The Projected Impact is the cost reduction that the optimizer estimates the recommended index will have. Click Show index DDL to see the CREATE INDEX statement for the recommendation.

DPA analyzes the plan and lists steps with the most inefficient access paths.

These steps read data to be processed by subsequent "consumer" plan steps. While consumer steps (for example, sorts) can have a high plan cost, they are usually affected by a preceding step that read too much data.

Information about each step includes:

- The step number and the type of operation being performed in the step (for example, INDEX SCAN).
- The index this step uses, if it uses an index.
- Any predicates. These are snippets of the SQL that the plan step is acting on. They are typically portions of JOIN or WHERE clauses in which a table's column is being compared to another column or value.
- Any warnings that apply to the step.
- The number of rows the optimizer estimates this step will read. Critical and warning icons identify steps that read a high percentage of the table or index rows, and therefore have a greater need for tuning or an index.

Tips for using this information

- Before you add an index, weigh the projected impact or potential performance improvement against indexing trade-offs. Also consider the indexing needs of other queries.
- Click any step to get detailed information about the operation and recommendations for potentially reducing the amount of I/O.

If predicates are listed, they often indicate which columns need to be indexed, or where the optimizer is not using an existing index. For example, if the query calls a function on the column, the plan will not use an index.

If warnings are listed, click the warning for a detailed description of the condition that DPA has identified as a potential reason for concern.

Warnings

DPA provides the following warnings:

- A **predicate warning** occurs when a column needs to be converted to a different data type before it can be used. For example, if a query has a JOIN clause that equates a numeric column to a varchar column, one of the columns will be implicitly converted to the other's type. The optimizer typically does not use an index on an implicitly converted column. This is often why the optimizer doesn't use
an existing index that the query's author expected it to use.

- A **lookup warning** typically indicates that the database is doing an index lookup to identify the target rows, then doing an extra table access to get data not found in the indexed columns. To get better performance, consider adding a covering index, or extending an existing index to include columns needed to avoid the table lookup. However, remember that adding a large number of columns can increase the **index size and maintenance overhead**.

- A **spool warning** indicates that the step's result set is being stored for reuse later in the query's execution. While spool operations are often beneficial, the intermediate data storage can cause disk overhead and contention.

- A **parallel warning** indicates that DPA has detected a parallelism step later in this query's execution, implying that this step's intermediate result set is likely large enough to exceed parallel processing cost thresholds. Look for ways to rewrite the query to reduce the size of intermediate result sets earlier in the query. For example, look for a sub-select that could produce fewer rows or the nested loop join order if more than two tables are involved.

### Examine table statistics

At the top of the Current table information section, DPA provides table statistics, such as the size of the table and the amount of churn. A table's churn is the daily number of insert and delete operation expressed as a percentage of the total number of table rows.

### Tips for using this information

Consider the Size, Rows, and Churn values when making indexing decisions:

- **Size and Rows**: For large tables, indexing is often critical to good query performance, although an index on a large table uses large amounts of disk space. For small tables, full table scans sometimes offer better performance than the use of indexes.

- **Churn**: Each insert and delete statement, as well as some update statements, incur a performance hit due to index maintenance. Generally, the higher the churn, the more caution you should take when adding an index. Before you add a new index, weigh the query execution time saved against the time spent on index maintenance.

### Examine index details

DPA displays information about all existing indexes on the table, including the structure, the amount of fragmentation, how long ago the statistics were generated, and when the index was last used.

### Tips for using this information

Before you make any indexing decisions, first review the existing indexes. Consider the following questions.

ℹ Take **indexing trade-offs** into account when you are considering adding or extending an index.
Are the statistics stale? If the statistics are old and data churn is high, statistics should be updated frequently to provide the optimizer with the information it needs to make better plan decisions. If statistics are old and churn is high, consider updating the statistics before adding or modifying indexes.

Is there an existing index that an inefficient query should be using? Look for ways to adjust the query so that it uses the index.

If an inefficient query is using an existing index, are there inefficient table or index access steps on columns that aren't included in the index? Consider adding those columns to the existing index.

If an inefficient query is using an existing index, are there inefficient table or index access steps that indicate a lookup warning? Consider adding those columns to the existing index to make it a covering index for the query.

Is there no existing index that would improve an inefficient query's performance? Consider adding a new index.

Are indexes fragmented? Fragmentation occurs as a result of numerous insert and delete statements. Fragmentation causes index data to become out of order on the disk, with gaps between index data. This is not a major concern for small tables, but for large tables this can cause slow performance when the index is read using a scan operation.

Consider defragmenting your indexes on a regular basis for large tables, especially if data churn is high and many scans are occurring.

Correcting common index problems

After you determine what indexes are needed to improve query performance, look for additional benefits by identifying poor index usage, such as:

**Unused indexes**: Can indexes be removed without negatively affecting query performance? To help you find unused indexes, DPA lists how long ago each index was used. However, before you remove an index:

- Be aware that sometimes the Last Used value can show only the date since the monitored database instance was last started.
- Consider whether queries that run infrequently (for example, monthly or quarterly) might use the index.

**Too many indexes**: A large number of indexes on a table might be necessary for important queries to run quickly. However, you should also consider the performance overhead of index maintenance on other DML statements. Look for opportunities to:

- Combine similar indexes.
- Remove unused or rarely used indexes.
- Remove indexes that were added for queries that are not performance sensitive.

**Overlapping indexes**: Two indexes overlap if they both have the same leading edge columns in the exact same order, but one index has at least one additional column at the end. In this case, the
larger index (with more columns) is all that you need, and you can remove the smaller, redundant index. Alternatively, you might choose to remove the larger one if the additional columns are not being used, or if the additional columns offer little benefit compared to the cost of index maintenance.

- **Questionable index structure:** The following might indicate a poorly-constructed index:
  
  - Many columns: Indexes with many columns require more storage, and increase the cost of index maintenance. Perhaps the index was defined this way to make it a "covering index" for some queries. If not, consider removing trailing edge columns.
  
  - Wide columns: Some DBAs question the benefit of adding wide columns (for example, long varchar) to an index, because of the high amounts of storage needed for the index and the maintenance overhead. With this in mind, if your queries do a lot of searching on any column, consider indexing it.

**Indexing trade-offs**

While indexes can provide performance benefits for some queries, consider the following trade-offs when making indexing decisions:

- **Index maintenance:** When a table row is inserted or deleted, the corresponding entry in each index must also be inserted or deleted. If an indexed column is updated, the associated entries in the index must also be updated. These operations on indexes increase the time an insert, delete, or update statement takes to run. The cost of index maintenance increases as the amount of data churn increases.

- **Disk space:** Indexes consume disk space. The larger the table and the more columns in the index, the more disk space it needs.

**Identify blocking sessions and deadlocks**

DPA provides information to help you determine if blocking sessions and deadlocks are affecting performance, and to investigate the root cause of these issues. See the following sections:

- Identify blockers causing the longest waits
- Find the last activity of an idle blocker
- Investigate deadlocks on SQL Server instances

**Identify blockers causing the longest waits**

Are blocking sessions causing performance problems in your environment? Use the Blocking tab to identify the root blockers, find out which SQL statements are being blocked, and determine which blocking sessions are responsible for the longest overall waits. DPA shows the aggregated wait time for each blocker, which helps you focus your tuning efforts on blockers with the largest impact.
To view information about blocking sessions, click a database instance name on the DPA homepage to display the Trends charts. If necessary, click a bar on the chart to drill down to the time period you're interested in. Then click the Blocking tab below any Trends chart to view correlated information about blockers during that time period. The size of each segment in a bar provides a visual indicator of the waits that session caused.

Find the last activity of an idle blocker

Idle blockers can be difficult to diagnose because they are currently not performing any activity in the database. To help you find and fix the root problem, use DPA to determine what that session was doing before it became idle.

From the DPA homepage, click the name of a database instance. Click a bar on the Top SQL Statements chart to drill into a day, and then click a bar to drill into an hour. DPA displays information about the type of waits experienced during that hour.
Click the Blockers tab above the chart to see a list of the blockers for that time period. You can expand a blocker to see information about the waits it caused. Each idle blocker row has a Find Last Activity link on the right.

To find out what a blocking session was doing before it went idle, click Find Last Activity. The Find Last Activity dialog tells you when the last activity occurred.
Click a button to view the activity. The Timeslice tab shows a bar representing the last SQL statement executed by the idle blocker. You can drill in to investigate further. If you clicked Annotate and View Activity, the SQL statement is automatically annotated to make it easy to find in the future.

Investigate deadlocks on SQL Server instances

Deadlocks occur when two sessions have a lock on different resources, and each session needs the resource of the other to complete its task. One session (the victim) eventually releases its lock and does not complete its task. The transaction time that the victim spent in contention is a good measure of the impact that the deadlock had on performance.

For monitored SQL Server database instances, DPA provides detailed information about deadlocks, including the Victim Impact (how long the deadlocked transactions ran). Click the Deadlocks tab below any Trends chart to see the latest deadlocks for that time period.
Click the link in the Time column to open the Deadlock Details page, which includes the following sections:

- The Deadlock Summary section shows high-level information, including the Total Victim Impact.
- The Victims section shows details about the queries that were rolled back.
- The Survivor section shows details about the query that was completed.
- The Deadlocked Resources section shows the type of lock and the lock mode. Click the links for expert advice.

<table>
<thead>
<tr>
<th>Time</th>
<th># Sessions</th>
<th>Victim Impact</th>
<th>Objects</th>
<th>Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tue, Sep 6, 2016 6:56:15 PM</td>
<td>3</td>
<td>00:00:20.636</td>
<td>s4load.dbo product</td>
<td>Accounting, Load Test, Order Entry</td>
</tr>
<tr>
<td>Tue, Sep 6, 2016 6:56:10 PM</td>
<td>2</td>
<td>00:00:10.660</td>
<td>s4load.dbo product</td>
<td>Accounting, Load Test, Order Entry</td>
</tr>
<tr>
<td>Tue, Sep 6, 2016 6:52:59 PM</td>
<td>2</td>
<td>00:00:10.070</td>
<td>s4load.dbo address, s4load.dbo product</td>
<td>Load Test, Order Entry</td>
</tr>
<tr>
<td>Tue, Sep 6, 2016 6:52:54 PM</td>
<td></td>
<td></td>
<td>s4load.dbo product</td>
<td>Accounting, Load Test, Order Entry</td>
</tr>
<tr>
<td>Tue, Sep 6, 2016 6:49:41 PM</td>
<td>2</td>
<td>00:00:12.910</td>
<td>s4load.dbo address, s4load.dbo product</td>
<td>Accounting, Load Test</td>
</tr>
</tbody>
</table>

Victim Impact shows how long the deadlocked transactions ran.
Find and investigate unusually long wait times (anomalies)

DPA’s anomaly detection algorithm identifies unexpected increases in wait time. DPA collects historical data and uses it to "learn" what normal is. DPA's proprietary algorithm makes predictions based on this data. When wait times for a time period are higher than expected, DPA reports an anomaly.
Get notified when wait time is higher than expected

Configure the Database Instance Wait Time Anomaly alert to be notified if the wait time increases significantly for a database instance. This alert is triggered if the wait time for an instance was abnormally high during the most recently completed hour.

View information about wait time anomalies

The wait time meter on the DPA home page indicates recently detected anomalies. Drill in to a database instance to view more detailed information on the Anomaly Detection charts.

Wait time meter

On the DPA homepage, the color of the wait time meter indicates whether wait times are higher than expected for a database instance. Yellow indicates a warning status, and red indicates a critical status. (For information about these thresholds, see Anomaly thresholds.)

The wait time meter reflects recent database activity. The status is calculated every 10 minutes. The wait time meter indicates the status during the six most recent 10-minute intervals (a rolling one-hour time period).

Anomaly Detection chart (30-day period)

If DPA detects wait time anomalies for a database instance, click the database instance on the DPA homepage to drill in for more information. The Top SQL Statements chart and the Anomaly Detection chart show information from the past 30 days. These charts work together to help you understand the waits occurring in this database instance:

- The Top SQL Statements chart identifies the SQL statements with the highest wait times. In many cases, these are candidates for tuning. But in other cases, further tuning is not possible or the wait times are not a problem. The large bars are normal, and you are more interested in unexpected increases in wait time.

  - An anomaly is detected when the combined wait time for all SQL statements is higher than expected. The Top SQL Statements chart shows only the SQL statements with the highest waits, which might not be responsible for the anomaly.

- The Anomaly Detection chart identifies days when wait times were significantly higher than expected (wait time anomalies occurred).
Each bar on the Anomaly Detection chart shows a roll-up of the amount of wait time that the database instance experienced during that day.

- Red segments indicate that wait times for one or more hours were much higher than expected (critical).
- Yellow segments indicate that wait times for one or more hours were higher than expected (warning).
- Green segments above the baseline (0) indicate that wait times for one or more hours were within the normal range, but slightly higher than expected.
- Green segments below the baseline indicate that wait times for one or more hours were lower than expected.

DPA classifies all lower-than-expected wait times as normal, and does not alert on them.

Show only warning and critical segments

To focus only on segments that indicate wait time anomalies, you can deselect Show Normal Bars to hide the green bars.
Drill in further

Click a bar that represents a day when anomalies occurred to display the Anomaly detection chart for that day.

Anomaly Detection chart (one-day period)

The Anomaly Detection chart for a one-day period shows the differences between the predicted wait times and actual wait times for each hour. The bar for the current hour shows the differences during the six most recent 10-minute intervals (a rolling one-hour time period).

The baseline (0) represents the predicted value for the hour.

Investigate higher-than-expected wait times

After you determine when anomalies are occurring, you can use either query performance analysis or DPA reports to help you determine which SQL statements are responsible for the anomalies.

Determine when anomalies occurred

Use the Anomaly Detection charts to determine when anomalies occurred, and to see which SQL statements were running during that time period.

1. From the DPA homepage, click the database instance that is experiencing anomalies to display the 30-day Anomaly Detection chart.

2. Click a bar that represents a day when wait times were much higher than expected.

   The one-day Anomaly Detection chart shows the hours when anomalies occurred. In this example, the 2 PM hour had the highest unexpected wait times.
3. Open the Anomaly Detection chart for a one-day period, and find the hours with large red segments. These are the hours when wait times were much higher than expected.

4. Click the bar that represents the hour, and view information about the SQL statements with high wait times that ran during that hour.

Display historic wait times and performance analysis for these queries

To determine which SQL statement is causing the anomaly, you can use the Query Detail page to view the historic wait times. It’s usually a good idea to start with the bars at the top of the list. Also remember that more than one SQL statement might be causing the anomaly.
1. Click a bar that represents a SQL statement.
   The Query Details page displays wait times for that SQL statement during the selected one-hour time period.

2. Click the time period at the top of the page and change the time range. For example, select Last 30 days or Last 90 days.
   In this example, the wait times for February 2 are clearly an anomaly.

3. You can also scroll down to review DPA's query performance analysis for this SQL statement.
   In this example, we can see that the SQL statement was being blocked by other queries when the anomaly occurred.

Use DPA reports to review wait or query details

After you determine when the anomalies are occurring, you can create a report to review the wait times for that hour during the last 30 days to look for unusually high wait times.
1. Click Reports.

2. Select the database instance that is experiencing anomalies.

3. Select Top Waits as the Report Type.

   ![Database Instance: SQL2K17, Report Type: Top Waits]

4. Click Report Options.

5. Under Waits to Display, select the Top 50 Waits.

   ![Top Waits Ranked by Cumulative Wait Time]

6. Under Dates to Display, select Last N Days as the Date Range, and leave 30 as the number of days.

7. Change the Hour Range to the time period when anomalies are occurring.

   ![Date Range: Last N Days, Hour Range: 2:00pm to 3:00pm, Days of Week: Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Dates: January 7, 2019 - February 5, 2019]

8. Click Display Report and review the wait times.

   In this example, the anomaly stands out.
Investigate lower-than-expected wait times

If wait times are much lower than expected, consider investigating to determine whether any SQL statements that normally run during that time period are missing.

1. Open the Anomaly Detection chart for a one-day period, and find the hour with the largest green segment below the baseline. Note the date and hour.

2. Click Reports.

3. Select the database instance, and select Top SQLs as the Report Type.

4. Click Report Options.

5. Under SQL Statements to Display, select the Top 50 SQL Statements.

6. Under Dates to Display, select Last N Days as the Date Range, and leave 30 as the number of days.

7. Change the Hour Range to the time period when wait time was much lower than expected.

8. Click Display Report and review the SQLs that ran each day to help determine if anything is missing.

About anomaly detection in DPA

DPA uses an anomaly detection algorithm to determine if the wait times for a database instance are significantly higher than usual. In some cases, high wait times are normal and expected. With anomaly detection, DPA can alert you to unexpected increases in wait times, and help you investigate these anomalies.

How does DPA’s anomaly detection work?

A machine learning algorithm uses wait time data that DPA collects to predict future wait times. DPA uses these predictions to detect wait times that are significantly higher than expected.
Step 1: Data collection
DPA gathers the data that the algorithm will use to learn what normal is and to predict future wait times. Up to 90 days of historical hourly data is used for learning.

Anomaly detection requires a minimum of three days of learning data. DPA does not show any information about anomalies until it has collected at least three days of data. Predictions improve as more data is collected.

Step 2: Data analysis and predictions
Based on the learning data, the algorithm calculates:

- The amount of wait time that the database instance is likely to experience during each 1-hour period for the next 30 days.
- The standard deviation for the entire data set (which is used to calculate thresholds).

When enough data is available, predictions include daily and weekly seasonality (patterns of predictable fluctuations):

- Daily seasonality accounts for differences during each hour. For example, normal wait times at 2 AM are probably different than normal wait times at 2 PM.
- Weekly seasonality accounts for differences during each day of the week. For example, normal wait times at 2 PM on Saturday are probably different than normal wait times at 2 PM on Wednesday. (Weekly seasonality requires at least 30 days of learning data.)

Step 3: Anomaly detection
For each hour, DPA compares the actual amount of wait time during that hour to the predicted value. If the actual amount of wait time is above the warning or critical threshold, DPA:

- Changes the color of the wait time meter on the DPA homepage.
- Displays yellow or red segments on the bars in Anomaly Detection charts.
- Triggers the Database Instance Wait Time Anomaly alert, if it has been configured.

How DPA determines the status of an incomplete hour
To determine if the wait time meter and hourly Anomaly Detection chart should show a warning or critical status for an incomplete hour, DPA uses the last 6 completed 10-minute intervals (a rolling one-hour interval). The status is updated every 10 minutes. For example, to determine the status of the 2:00 hour:

- From 2:00 to 2:09, DPA uses data from 1:00 to 1:59.
- From 2:10 to 2:19, DPA uses data from 1:10 to 2:09.
- From 2:20 to 2:29, DPA uses data from 1:20 to 2:19 (and so on).

For each 10-minute interval of the current hour, DPA uses a rolling one-hour interval to determine the status shown on the wait time meter. For example, 2:10 to 2:19 uses data from 1:10 to 2:09.
SQL statements excluded from the Trends chart

The anomaly detection algorithm uses the total wait time for the database instance, including wait time from any SQL statements that you have excluded from the Trends charts. In most cases, a statement is excluded from the Trends charts because it always has high wait times and the large bar dominates the charts. If the statement runs on a regular schedule with the expected amount of wait time, no anomaly would be detected during that time period, because high wait times are normal during that period. An anomaly would be detected only if wait times during that period were significantly higher than normal, in which case you might want to investigate the change.

Does anomaly detection work well for all database instances?

DPA's anomaly detection algorithm, like most algorithms associated with workloads, works best when:

- The monitored database instances have a consistent workload executing against them.
- Daily and weekly seasonality is consistent. For example, database wait times are similar each Monday at 10 AM.
- DPA monitoring is always on (not shut down for hours or days at a time).

The algorithm might not work well when:

- The workload for a database instance is sporadic (for example, QA or reporting instances with inconsistent wait times).
- Daily and weekly seasonality is not consistent. For example, the workload on Monday at 10 AM varies from one week to the next, with no predictable pattern.
- DPA is not monitoring the instance consistently, and so it cannot get a good understanding of what normal is.

If anomaly detection does not work well for any of your monitored instances, SolarWinds recommends disabling anomaly detection for those instances.

Large gaps in the learning data

If monitoring stops for more than 30 days, the anomaly detection algorithm does not make predictions based on the stale learning data collected before the 30-day gap. DPA collects new learning data and, after three days, begins to make predictions based on the current data.

Anomaly thresholds

Anomalies are classified as warning and critical. The threshold for each classification is based on the standard deviation of the wait times for the associated time period.

![Standard deviation is a measure of how dispersed the values in a data set typically are.](image)

The default values for the thresholds are listed below. You can edit the associated configuration option to change the default values.
<table>
<thead>
<tr>
<th>CLASSIFICATION</th>
<th>DEFAULT THRESHOLD</th>
<th>CONFIGURATION OPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Warning</td>
<td>The predicted wait time for the hour + 2 standard deviations</td>
<td>ANOMALY_DETECTION_THRESHOLD_WARNING</td>
</tr>
<tr>
<td>Critical</td>
<td>The predicted wait time for the hour + 3 standard deviations</td>
<td>ANOMALY_DETECTION_THRESHOLD_CRITICAL</td>
</tr>
</tbody>
</table>

Specify the learning date after the load on a database instance changes

If the load on a database instance changes significantly (for example, because of changes in the network environment), the previously collected learning data is no longer accurate. To prevent this data from being used for anomaly detection, set the advanced configuration option ANOMALY_DETECTION_FORCE_LEARNING_DATE to the date when the load change occurred. Wait time data collected before this date will not be used to predict future wait times.

Disable anomaly detection for a database instance

By default, anomaly detection is enabled for all database instances. To disable anomaly detection for a database instance that with an inconsistent workload or sporadic monitoring, set the advanced configuration option ANOMALY_DETECTION_ENABLED to False for that instance.

Add an annotation to document a change to the database

Check out this video (1:19) on using annotations.

When you make a change that could affect performance (such as adding an index, tuning a query, or adding resources), you can add an annotation in DPA to show when that change was made. The annotations are displayed on all trend and timeslice charts. By comparing performance data before and after the change, you can see what effect the change had.

1. From the DPA homepage, click the name of the database instance affected by the change.
2. Click Annotate in the upper-right corner of the trend chart.
3. Name the annotation, specify when it was added, and provide details about what change was made and why.

If your DPA server is in a different time zone, enter the DPA server time.
4. Click Save.

The annotation is displayed as a flag on the chart. Point to the flag to see a summary, or click it to see details.
Name SQL statements

On the right side of each trend chart, a legend identifies each SQL statement. By default, SQL statements are identified by their hash values.

When you are investigating a specific SQL statement, you can give it a name to make it easier to identify. The name appears in reports and chart legends.

1. In the chart legend, click the hash value that represents the SQL statement.
   The Query Details page displays information about the SQL statement.
2. In the top-right corner, click SQL Properties.
3. In the SQL Properties dialog, enter the name.
4. Click Save.

5. In the upper-left corner of the Query Detail page, click Back to return to the previous page.

The legend displays the name instead of the hash value.

The colors representing the SQL statements on the chart might change after you name the SQL statement.
Resource metrics in DPA

Use the topics in this section to:

- View resource metrics in DPA
- View and learn about resource metric baselines
- View or change resource metric thresholds

View resource metrics in DPA

Resource metrics provide information about how resources (such as CPU, disk, and memory) are being used at specific points in time. These metrics show what was happening in the rest of your environment during database slow-downs, and can provide context to help you identify the root cause of performance problems.

DPA displays resource metrics in the following locations.

View all available metrics on the Resources tab

The Resources tab displays all available resource metrics for the selected database instance.

1. On the DPA homepage, click a database instance to view detailed information.
2. In the upper-right corner, click the Resources tab.

   The Resources tab displays all available resource metrics for the selected database instance. You can:
   - Click a time period in the upper-right corner to change the time range.
   - Show or hide baselines.
   - Click the Information link to display information about a metric.
   - Click Settings to view or change the thresholds for that metric.

Correlate wait time with resource metrics on the Trends tab

When you are viewing wait time charts on the Trends tab, you can scroll down to determine if unexpectedly long wait times correlate with resource contention.

1. On the DPA homepage, click a database instance to view detailed information.
2. In the upper-right corner, click the Trends tab.
3. Scroll down and click the Resources tab below the wait time charts.

   The Resources tab displays a subset of the available resource metrics for the selected database instance. You can:
• Click Add Resource Chart to include additional charts.
• Click ☀ to display information about a metric.
• Click ⚙ to view or change the thresholds for that metric.
• Use other icons to the right of a chart to remove it, change its location, or replace it with a different chart.

View resource metrics related to the performance of a query

To help you find the root cause of long wait times for a query, the Query Details page includes the most relevant statistics, blocking, plan, and metrics charts. When you scroll down to view these charts, the Top Waits chart at the top of the page remains visible so you can correlate query wait times with other events during the same time period.

About resource metric baselines

When you are viewing resource metrics on the Resources tab, you can display baselines to compare values from a specific period to historical norms. Baselines provide context for the current values. Metric values that are far above or below the baseline could indicate areas in need of tuning or reconfiguration.

Monitoring must be active for at least one day before baselines can be calculated, and baselines become more representative as more monitoring days pass.

Baselines are not available for metrics collected for the VM Option.

Show or hide baselines

Baselines are available when the selected time period is one week or less.

Click the Show baseline or Hide baseline button near the top of the Resources tab to show or hide baselines. When you show baselines:

• A dark line represents the average value.
• If the time period is less than one week, a shaded area represents values between the 10th and 90th percentiles.
How are baselines calculated?

Baselines are calculated for each one-hour period. By default, baselines are calculated using data only from weekdays (Monday through Friday). Each baseline is calculated using data from the corresponding hour for all weekdays, so the value for a specific hour is the same across all days. (For example, the value for 1 - 2 PM is the same Monday through Friday.)

Baselines are calculated using historical data from before the earliest time shown on the chart. For example, if a chart covers one week and starts on May 10th, all baselines are calculated using data from May 9th and earlier. For this reason, one-week charts show repeating patterns for each day.

Change the days included in baseline calculations

To change the days included in baseline calculations, update the advanced configuration option METRICS_BASELINE_TYPICAL_HOUR_CALCULATION. This option can be set globally or for a specific monitored database instance.

Choose one of the following values:

<table>
<thead>
<tr>
<th>VALUE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weekday Only (M-F)</td>
<td>Baselines are computed for each one-hour period using data from the corresponding hour on weekdays (Monday through Friday). This is the default.</td>
</tr>
<tr>
<td>All Days of the Week</td>
<td>Baselines are computed for each one-hour period using data from the corresponding hour on all days.</td>
</tr>
<tr>
<td>Same Day of Week</td>
<td>Baselines are computed for each one-hour period using data from the corresponding hour on the corresponding day. (For example, the value for 1 - 2 PM on Monday uses data from the corresponding hour on Mondays, and is therefore different than the value for 1 - 2 PM on Friday.) Be aware that this option increases the number of baselines per metric from 24 to 168.</td>
</tr>
</tbody>
</table>

View or change resource metric thresholds

Resource metric charts in DPA indicate when the metric has exceeded a Warning or Critical threshold. You can change the default thresholds to meet the needs of your environment. The custom thresholds can apply to a specific database instance or all monitored instances.

View the current thresholds

1. On the DPA homepage, click the database instance whose resource metric thresholds you want to view.

   ![Tip] If you are going to change the default thresholds for all database instances, you can click any instance.

2. Click the Resources tab.
3. Click the tab that displays the metric whose thresholds you want to view or change.
4. Locate the metric chart and click Settings below the chart.

The Resource Settings page displays the current thresholds.

Change the thresholds

1. Select Custom.
2. Enter the new Critical and Warning threshold values.
3. Do one of the following:
   - To use the new values only for this database instance, click OK.
   - To use the new values for all database instances, click Save As Default. Then click Yes and OK at the confirmation prompts.

The new default threshold values are used for all database instances unless custom thresholds have been specified for an instance. Any database instance with Custom selected (instead of Use Defaults) will continue to use those custom thresholds.
DPA user accounts

Use the following topics to create user accounts, assign privileges, and specify how users will log in to DPA:

- DPA roles and privileges
- Create a user account
- User authentication options
- Configure Active Directory or LDAP

DPA roles and privileges

When you add user accounts in DPA, you assign each user a role. The role determines the user's privileges.

Administrator role

Administrators have access to all DPA functionality, including all setup, administration, and support options.

DPA requires at least one Administrator account, which is created during installation.

Only administrators can perform certain actions, such as:

- Register and unregister database instances and VMs
- Allocate licenses
- Run advanced support utilities
- Edit system-wide Advanced Options
- Start and stop all monitors
- Create, edit, and delete report schedules
- Create, edit, and delete alert groups
- Configure the mail server
- Administer users, contacts, and contact groups

Read Only on All Instances role

Users with this role can perform the following actions for all database instances:

- View performance data and metrics
- Run reports and view existing report groups
- View existing alerts
- View logs
Custom Privileges role

The Custom Privileges role specifies which privileges a user has, and which database instances these privileges apply to. Use this role to:

- Prevent users from seeing data about certain database instances
- Give users privileges to manage monitoring options, alerts, and reports without granting them full administrative privileges

When you assign this role to a user, you can grant any of the following privileges. Privileges can apply to all database instances or only selected instances.

<table>
<thead>
<tr>
<th>PRIVILEGE</th>
<th>ACTIONS ALLOWED AGAINST SELECTED DATABASE INSTANCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>View Data</td>
<td>• View performance data and metrics</td>
</tr>
<tr>
<td></td>
<td>• Run reports and view existing report groups</td>
</tr>
<tr>
<td></td>
<td>• View logs</td>
</tr>
<tr>
<td>Manage Reports</td>
<td>Create, edit, and delete report groups</td>
</tr>
<tr>
<td>View Alerts</td>
<td>View existing alerts</td>
</tr>
<tr>
<td>Manage Alerts</td>
<td>• Create, edit, and delete alerts</td>
</tr>
<tr>
<td></td>
<td>• View existing alert groups</td>
</tr>
<tr>
<td>Manage Monitoring</td>
<td>• Create, edit, and delete blackout periods for monitoring</td>
</tr>
<tr>
<td></td>
<td>• Manage I/O configuration</td>
</tr>
<tr>
<td></td>
<td>• Update Advanced Options for a specific database instance</td>
</tr>
<tr>
<td></td>
<td>• Add annotations</td>
</tr>
<tr>
<td></td>
<td>• Exclude SQL statements from trend charts</td>
</tr>
<tr>
<td></td>
<td>• Start and stop individual monitors</td>
</tr>
</tbody>
</table>

Users with Manage Monitoring permissions cannot see the charts at the top of the DPA homepage.

Create a DPA user account and assign privileges

You must add a user account for each person who needs to log in to DPA. Each user is assigned a role, which determines the user’s permissions.

Optionally, you can integrate DPA with your company's Active Directory (AD) or LDAP service. If you do this:

- Users can log in to DPA with their domain accounts.
- You can add AD or LDAP groups to DPA and assign privileges to each group.

Before you add users, determine who needs access to DPA and which privileges each user needs. For more information about the available options, see DPA roles and privileges.
1. In the DPA menu bar, click Options.
2. Under Administration > Users & Contacts, click User Administration.
3. On the User Administration page, click Create User.
4. Enter a unique user name and a password.
5. Specify the user's privileges:
   - To assign privileges associated with predefined roles, select Administrator or Read Only on All Instances.
   - To assign custom privileges:
     a. Select Custom Privileges Specified Below.
     b. To grant access to data from all database instances, select privileges in the top row. To limit access, select privileges for each database instance.

![Image](image1.png)

   The View Data privilege is automatically selected when you select any higher privilege.

In the example below, the user can access data from only one database instance. This user can make changes to monitoring options, run reports, and view existing alerts for the selected instance.

<table>
<thead>
<tr>
<th>Database Instance</th>
<th>Type</th>
<th>View Data</th>
<th>Manage Reports</th>
<th>Alerts</th>
<th>Manage Monitoring</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change All →</td>
<td></td>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPA-SUSE-MYSQL56:3306</td>
<td>MySQL</td>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPA-WIN-MYSQL57:3306</td>
<td>MySQL</td>
<td></td>
<td></td>
<td>View</td>
<td></td>
</tr>
</tbody>
</table>

6. Click Save.

If you configured DPA to point to your Active Directory or LDAP server, you will see an option to either create a user or a group. The group corresponds to a group in Active Directory or LDAP.
User authentication options

SolarWinds DPA supports Active Directory (AD) and Lightweight Directory Access Protocol (LDAP) authentication. Using your existing authentication infrastructure eliminates the need to duplicate your user accounts in DPA. After you configure AD or LDAP authentication, users can log in with their domain account or a custom user account created by DPA.

AD user authentication

DPA integrates with Windows Active Directory (AD). DPA uses the security group information from AD to assign permissions to groups. To configure DPA user authentication and permissions using AD, see Configure Active Directory or LDAP.

If your repository database is Azure SQL and you are monitoring one or more Azure SQL databases, you can use Azure AD authentication in DPA. To configure DPA user authentication and permissions using Azure AD, see Use Azure AD authentication in DPA.

LDAP user authentication

DPA integrates with most LDAP implementations to assign permissions to groups. To configure DPA user authentication and permissions using LDAP, see Configure Active Directory or LDAP.

Single sign-on

Using single sign-on (SSO), your AD users can log in to DPA without re-entering the domain credentials they used to log in to their operating system. Before you configure DPA for SSO, configure DPA for AD authentication.

Common Access Cards

You can use a Common Access Card (CAC) to log in to Windows and DPA. Before using a CAC, configure DPA for AD, and then for SSO as described in the sections above.

Configure Active Directory or LDAP

To use AD or LDAP user authentication in DPA:

1. Gather the following information from your domain administrator:
   - Directory service type: AD or LDAP
   - Domain name
   - Port number: Used to connect to the directory service
   - User: The domain user DPA uses to query the directory for users and groups
   - Password: The password of the domain user, preferably one that does not expire

2. In DPA, click Options. Then, under Administration > Users & Contacts, click Configure AD/LDAP.
3. Select the type of directory service you have: Active Directory or LDAP.
4. Click Next.

Connection information

Domain name
Enter the domain name.

SolarWinds recommends using a domain name, not the name of a specific domain controller.

Do you have multiple domains?
If your domain users authenticate from a different domain other than the domain specified here, you must connect to the global catalog ports 3268 or 3269. The domain users must belong to a universal group, and that universal group must be added under Options > Administration > Users & Contacts > User Administration.

Port
Select the port number.
If you use a unique port, select Other non-standard port. Enter the port number, and select SSL if required.

User and Password
DPA uses this user to search the directory service for users and groups.

Active Directory user name
For the AD user name, use one of the following formats:
- Distinguished Name (DN): cn=BobSmith,cn=Users,dc=domain,dc=local
- User Principal Name (UPN): bsmith@domain.local

LDAP user name
For the LDAP user name, use the following format:
- Distinguished Name (DN): cn=BobSmith,cn=Users,dc=domain,dc=local

Did the connection test fail?
If you use an SSL port and the verification fails, DPA must import its certificate. Click Yes on the confirmation window to try again.
Base search location

Base DN

Use the default

SolarWinds recommends selecting the default, so DPA uses the detected base DN from the previous step.

Example of default base DN: \texttt{dc=east,dc=acme,dc=com}

Use a custom value

You may use a value other than the default base DN. For example: You use a global catalog that supports multiple domains, and you want to broaden the scope of the search.

Example for multiple domains: \texttt{dc=acme,dc=com}

Advanced settings

If this is your first time using this wizard, do not use the advanced settings.

Only use advanced settings if you completed this wizard and you experience slow domain user logins or group searches.

Are domain user logins slow?

Set the User Search Base value if domain user logins take a long time.

If your company has one domain, specify the location in the directory tree that contains all of the domain users that will use SolarWinds DPA.

If you do not know what to put here, ask the domain administrator of your company the following questions:

"What folder, or organization unit (OU), in the directory tree of the domain contains all of the users? I must specify a search base for users. What is the distinguished name of the folder?"

Example: \texttt{cn=users OR ou=users}

Are domain group searches slow?

Set the Group Search Base value if domain group searches in User Administration take a long time.

Specify the location in the directory tree that contains all of the groups to which SolarWinds DPA users belong.

If your company has multiple domains, you can enter the group search bases individually. After you add groups to SolarWinds DPA using the group search base from one domain, update this wizard to specify a group search base in another domain.

If you do not know what to put here, ask your the domain administrator of your company the following:

"What folder, or organization unit (OU), in the directory tree of the domain contains all of the groups? I must specify a search base for groups. What is the distinguished name of the folder?"
Example: cn=groups OR ou=groups

Summary

Confirm the information for configuring DPA with your directory service, and click Finish.

You must restart the DPA server for the settings to take effect.

Configure authentication and permissions for groups of users

After you have set up DPA to use Active Directory or LDAP, do the following:

1. In AD or LDAP, determine which groups contain the users that you want to grant access to DPA. You may need to create a group if a suitable group does not exist.
2. In the DPA menu bar, click Options.
4. Click Add Active Directory Group or Add LDAP Group.
5. Click Search for a Group.
6. Find and select the group you want and click Save.
7. Assign privileges to the group, just as you would for a user. This assigns those permissions to the domain users who are members of the group.

DPA does not support single sign-on (SSO) for individual accounts. It only supports AD or LDAP groups.

8. Click Save.

All domain users in the selected group can log in to DPA using their domain credentials. The users have the privileges you set up for the group in DPA.

You can add multiple AD or LDAP groups in DPA. If a domain user is a member of more than one group, DPA grants them the combined privileges from all of their groups.

Log in to DPA

When you enter the domain user name and password in the DPA login screen, DPA searches your directory service for a matching user name, and then authenticates using the password. If the domain user belongs to one of the groups that you configured as a DPA custom user, the login succeeds.

Name formats for AD login

DPA supports three types of login name formats for Active Directory:

- SAM account name: username
- User principal name: username@domain.local
- NT/AD: domain\username
User name for LDAP

The user name used by DPA is the LDAP user object uid attribute.
DPA alerts

Use DPA alerts to become aware of issues and address them proactively before they affect end users. To work with DPA alerts, see the following topics:

- View the status of DPA alerts
- Configure a DPA alert
- Send SNMP traps from DPA alerts
- Stop DPA alerts for a period of time
- Alert on increases in SQL wait times
- Create a custom alert to display the name of the active node in a SQL Server cluster
- Create an alert group
- Create and manage DPA contacts and contact groups
- Notification policy for DPA alerts

View the status of DPA alerts

From the DPA menu, click Alerts to view information about all saved alerts, including the alert status (for example, High, Normal, or Inactive) and the current value. From this page, you can:

- Choose a database instance from the drop-down menu in the upper-right corner to filter the list of alerts.
- Click the History button to display the status, value, and execution time for this alert. A maximum of 5,000 records can be displayed. You can filter by Status, Date Range, or Database Instance.
- Click the name of the alert to open it for editing. For more information about the available fields, see Configure a DPA alert.
- Click a column heading to sort the list of alerts.

Configure a DPA alert

SolarWinds DPA alerts give you proactive control of your databases, notifying you of issues before they become problems for end users. Set thresholds on key wait time statistics, resource metrics, or standard administration indicators. The result is improved customer service, fewer help desk tickets, and increased compliance with database service-level agreements.

DPA includes the following types of alerts:

- **Wait Time** alerts are based on the amount of time users or applications waited on the database. They are triggered when wait time exceeds a user-defined threshold, or when wait time is much higher than normal (an anomaly).
- **Resources** alerts are triggered when a resource metric, such as CPU utilization or memory usage, exceeds its threshold.
• **Administrative** alerts are used to monitor the health of the database system. For example, you can configure an alert that is triggered when the database instance is not accessible or when any database parameter changes.

• **Custom** alerts are user-specified queries that are run against the monitored database or the DPA repository. The query returns a value (or set of values) that might trigger an alert depending on user-defined threshold settings.

**Configure a Wait Time alert**

1. Click Alerts, and then click the Manager Alerts tab.
2. Select Wait Time as the Alert Category.
3. Select the type of alert.
   
   ![Image](https://via.placeholder.com/150)
   
   To find out more about an alert type, select it to display a description on the right.
4. Click Create Alert.
5. In the Alert Information section, enter a unique name and select the execution interval.
   
   The execution interval specifies how often the alert runs and also the amount of data that DPA examines. For example, if the execution interval is 10 minutes, DPA executes the alert every 10 minutes and queries the last 10 minutes of data to determine whether to trigger the alert.

   SolarWinds recommends an execution interval of **at least 10 minutes**. This allows time for valid samples and prevents unnecessary alerts from a single slow execution.
   
   ![Image](https://via.placeholder.com/150)
   
   Alerts are active by default. Clear the Active check box to disable an alert but not delete it.
6. Enter the text to be sent with the email or page notification. You can include an explanation of the alert and the suggested resolution.

   ![Image](https://via.placeholder.com/150)
   
   7. Select the database instances that the alert applies to.
8. If Alert Parameters are required for the type of alert you are configuring (for example, the SQL hash or wait type), click Search, enter a search string, and select the parameter.

9. For all Wait Time alerts except Database Instance Wait Time Anomaly, specify the thresholds for each alert level you want to enable.

   Leave the max value for the highest level blank to alert on anything above the minimum value for that level. If you configure multiple levels, the Max value for lower levels must equal the Min value for the next higher level.

<table>
<thead>
<tr>
<th>Level</th>
<th>Min (seconds)</th>
<th>Max (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HIGH</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>MEDIUM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INFO</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Alert thresholds for anomalies have default values, which can be changed through advanced configuration options.

10. Select the person or group who gets notified when an alert level is triggered or when the alert is broken. (The alert status is set to Broken if an error occurs during execution.)

   - If you have not added the person or group as a contact in DPA, click Add Contact or Add Contact Group and create the contact or group.
   - Select an SNMP contact to send SNMP traps when this alert is triggered.

11. Verify or change the notification policy.

12. Click Test Alert and verify that the recipients receive the email.

13. Click Save.
Configure a Resources alert

A Resources alert can monitor a single resource metric (such as Buffer Cache Hit Ratio) or all metrics in a resource category (such as Memory).

1. If necessary, view or change the Warning and Critical thresholds for the resource metrics that you want to alert on.

   For Resources alerts, the thresholds are specified in the resource settings instead of in the alert configuration.

2. Click Alerts, and then click the Manager Alerts tab.

3. Select Resources as the Alert Category.

4. Select the type of alert.

   To find out more about an alert type, select it to display a description on the right.

5. Click Create Alert.

6. In the Alert Information section, enter a unique name and select the execution interval.

   The execution interval specifies how often the alert runs and also the amount of data that DPA examines. For example, if the execution interval is 10 minutes, DPA executes the alert every 10 minutes and queries the last 10 minutes of data to determine whether to trigger the alert.

   SolarWinds recommends an execution interval of at least 10 minutes. This allows time for valid samples and prevents unnecessary alerts.

   Alerts are active by default. Clear the Active check box to disable an alert but not delete it.

7. Enter the text to be sent with the email or page notification. You can include an explanation of the alert and the suggested resolution.

8. Select the database instances that the alert applies to.

9. Specify the following Alert Parameters:

   a. For alerts against all metrics in a category, select the category.
      For alerts against a single resource metric, you can select a category to filter the Resource list.

   b. For alerts against a single resource metric, select the resource.

   c. Specify the calculation that is used to determine the alert level for an execution interval.
      To determine the alert level, DPA looks at the values collected during an execution interval and applies the specified calculation. For example for a single resource alert, if the metric value is collected once each minute and the execution interval is 10 minutes, DPA looks at the 10 values collected for an interval and applies one of the following calculations.
### Calculation Description

<table>
<thead>
<tr>
<th>Calculation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>% meeting metric alarm criteria</td>
<td>DPA assigns an alert level when a certain percentage of the values collected during an interval meet or exceed the threshold for that level. Use the Percentage drop-down to specify the percentage. For example, if the Percentage is 75% and 8 of 10 metric values are at or above the critical threshold, DPA assigns an alert level of Critical for the interval.</td>
</tr>
<tr>
<td>Average</td>
<td>DPA uses the average of the values collected during an interval to assign the alert level for that interval.</td>
</tr>
<tr>
<td>Median</td>
<td>DPA uses the median value collected during an interval to assign the alert level for that interval.</td>
</tr>
<tr>
<td>Maximum</td>
<td>DPA uses the maximum value collected during an interval to assign the alert level for that interval.</td>
</tr>
<tr>
<td>Minimum</td>
<td>DPA uses the minimum value collected during an interval to assign the alert level for that interval.</td>
</tr>
</tbody>
</table>

10. Select the person or group who gets notified when an alert level is triggered or when the alert is broken. (The alert status is set to Broken if an error occurs during execution.)

- If you have not added the person or group as a contact in DPA, click Add Contact or Add Contact Group and create the contact or group.
- Select an SNMP contact to send SNMP traps when this alert is triggered.
11. Verify or change the notification policy.
12. Click Test Alert and verify that the recipients receive the email.
13. Click Save.

Configure an Administrative alert

1. Click Alerts, and then click the Manager Alerts tab.
2. Select Administrative as the Alert Category.
3. Select the type of alert.
   
   - To find out more about an alert type, select it to display a description on the right.

4. Click Create Alert.
5. In the Alert Information section, enter a unique name, select the execution interval, and enter the email notification text.
   
   SolarWinds recommends an execution interval of at least 10 minutes.
6. Select the database instances that the alert applies to.
7. Enter the Alert Parameters, if any.
8. Specify the thresholds for each alert level you want to enable. (Some Administrative alerts have only one level.)

   Leave the max value for the highest level blank to alert on anything above the minimum value for that level. If you configure multiple levels, the Max value for lower levels must equal the Min value for the next higher level.

<table>
<thead>
<tr>
<th>Level</th>
<th>Min (occurrences)</th>
<th>Max (occurrences)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HIGH</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>MEDIUM</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>LOW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INFO</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

9. Select the person or group who gets notified when an alert level is triggered or when the alert is broken. (The alert status is set to Broken if an error occurs during execution.)
   
   - If you have not added the person or group as a contact in DPA, click Add Contact or Add Contact Group and create the contact or group.
   - Select an SNMP contact to send SNMP traps when this alert is triggered.
10. Click Test Alert and verify that the recipients receive the email.
11. Click Save.

Configure a Custom alert

Use Custom alerts to execute SQL statements or stored procedures against the monitored database or DPA repository to check for conditions not covered by other DPA alerts. Any parameter that can be returned to DPA using a SQL statement or stored procedure can be used as the basis for a custom alert.

The SQL statements or stored procedures in a custom alert must return one of the following types of values:

<table>
<thead>
<tr>
<th>VALUE RETURNED</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single numeric value</td>
<td>The SQL statement or stored procedure returns a single numeric value. The alert is triggered if the value exceeds the defined High, Medium, Low, and Info thresholds.</td>
</tr>
</tbody>
</table>
| One or more name/numeric value pairs (SQL statements only) | The SQL statement returns one or more rows of data. Each row contains a string in the first column and a numeric value in the second column. For example, the query could return database names and the amount of free space for each one:
  | DB1 | 120 |
  | DB2 | 840 |
  | DB2 | 35  |
| Single Boolean value                    | The SQL statement or stored procedure returns a string value of TRUE or FALSE. The alert is triggered if TRUE is returned. |
| Single alert status                     | The SQL statement or stored procedure returns a string value that specifies the alert status. Valid values are NORMAL, INFO, LOW, MEDIUM, and HIGH. |

1. Click Alerts, and then click the Manager Alerts tab.
2. Select Custom as the Alert Category.
3. Select the type of alert.
To find out more about an alert type, select it to display a description on the right.

4. Click Create Alert.

5. In the Alert Information section, enter a unique name, select the execution interval, and enter the email notification text.
   SolarWinds recommends an execution interval of at least 10 minutes.

6. To run the SQL statement or stored procedure against a monitored instance (instead of the DPA repository), select the database instances.

7. Under Alert Parameters, enter the SQL statement or stored procedure to execute.

8. In the Execute Against drop-down, indicate if the SQL statement or stored procedure should be executed against the selected monitored database instances or against the DPA repository database.

9. If the alert returns a numeric value, specify the thresholds for each alert level you want to enable. Leave the max value for the highest level blank to alert on anything above the minimum value for that level. If you configure multiple levels, the Max value for lower levels must equal the Min value for the next higher level.

10. Select the person or group who gets notified when an alert level is triggered or when the alert is broken. (The alert status is set to Broken if an error occurs during execution.)

   - If you have not added the person or group as a contact in DPA, click Add Contact or Add Contact Group and create the contact or group.
   - Select an SNMP contact to send SNMP traps when this alert is triggered.

11. Click Test Alert and verify that the recipients receive the email.

12. Click Save.

**Send SNMP traps from DPA alerts**

You can configure DPA alerts to send SNMPv2c traps to an SNMP-enabled Network Management System (NMS) when an alert level is reached. The trap contains the name of the monitored database instance, alert name, alert level, and response instructions.

To configure an alert to send an SNMP trap, complete the following tasks:

1. Import the DPA MIB file into your NMS.

2. Create one or more SNMP contacts. The SNMP contact defines the response instructions included in the trap, and so you must create different contacts for alerts with different response instructions.

3. In the alert definition, add the SNMP contact as recipient for the alert level that you want to send a trap.
The DPA MIB file

DPA contains a Management Information Base (MIB) file that defines the trap and the associated data sent with each trap. The MIB file defines the following:

- Private Enterprise Number
- One Trap Definition (NOTIFICATION-TYPE)
- Four string objects bound to each trap: database name, alert name, alert level, and response instructions

Before configuring DPA to send SNMP traps, provide the MIB file to the person responsible for importing MIB files into the NMS. The MIB file is in the following location:

<DPA_install_dir>/iwc/CONFIO-MIB.mib

Create an SNMP contact

The NMS that receives the trap is represented as an SNMP contact in DPA.

1. In the DPA menu bar, click Options.
2. Under Administration > Users & Contacts, click Contact Management.
3. Click Create SNMP Contact.
4. Enter a name and description to identify the associated alert(s).

   - By default, contacts are Active. You can select Inactive to disable the contact. When a contact is disabled, alerts associated with this contact do not send traps to the NMS.

5. Identify the NMS to send the trap to:
   - a. In the Trap Receiver Host field, enter the hostname or IP address of the server where the NMS is running.
   - b. In the Trap Receiver Port field, enter the port number where the NMS host is receiving traps. The default is 162.
   - c. In the Community String field, enter the community string used by the NMS for traps.
6. Enter the response instructions to be included in the trap.
7. To test the configuration, click Send Test SNMP Trap, and then verify that the NMS received the trap.
8. (Optional) To add the contact to a group, select the group and click Add.

   For example, when an alert reaches a certain level, you might want to send an email to the on-call personnel and send a trap to the NMS. You can add the SNMP contact to the On Call group.
Stop DPA alerts for a period of time

To stop alerting for a period of time, create an alert blackout. For example, you can create an alert blackout to suppress alerts during a maintenance window.

1. Click Alerts, and then click the Alert Blackouts tab.
2. Specify whether you want to create a blackout for an alert, a database instance, or an alert group.
3. Select the alert, database instance, or alert group, and then click Add Blackout.

4. Specify the beginning and end of the blackout period. Times are based on a 24-hour clock.
5. If the blackout period is for an alert, select All Databases or specify the database instances that this alert should not run on.
   If the blackout period is for a database instance, select All Alerts or specify the alerts that should not run.

6. Click Create Blackout.

The blackout is effective each week on the specified day and time until it is deleted.

Alert on increases in SQL wait times

Use the steps below to determine the average execution response time of your SQL queries, and then create an alert to warn you of significant deviations.

Determine the average execution time of your queries

1. On the DPA homepage, click a Database Instance.
2. Select the first query from the View Historical Charts for SQL list.
3. Scroll down to the Average Wait Time per Execution chart.
This chart shows the average execution time for the SQL statement on each day. The black bar denotes the average wait for this query across the days in the chart.

4. Point the cursor to each bar to see details for the day.
5. Point the cursor to the black bar to see the baseline of average wait time for this query for all executions for the month.

SolarWinds recommends alerting if the average wait time is twice as long as this baseline.

For example, if the baseline of average wait time for the month is 1.5 seconds, set the alert to 3 seconds.

Identify the top three queries and estimate the typical wait time. Choose a threshold, approximately twice as long, that you will use to create alerts.

Create an alert based on the wait threshold

1. Click Alerts > Manage Alerts.
2. Verify Wait Time is selected as the Alert Category.
3. Select Average Wait Time for a Single SQL from the Alert Type list.
4. Click Create Alert.
5. Name the alert and enter notification text.
6. Set the Execution Interval to 10 minutes or more. This allows time for valid samples and prevents unnecessary alerts from a single slow execution.
7. Select an Available Database Instance, and click Add.
8. Under Alert Parameters, search for one of the three SQL queries you identified earlier.

   SolarWinds DPA always returns the hash value from a search, even when you select a named query.

9. Under Alert Levels & Notifications, choose an alert level and enter a minimum time in seconds. This value should be twice as long as the baseline of average wait time for this query.

   The average wait time is specific to the execution interval. If the interval is 10 minutes, SolarWinds DPA looks at the average wait time for the SQL statement chosen for that 10-minute period and compares these values to your thresholds.

10. Select a Notification Group or Contact.
11. Click Test Alert, and verify you received the email.
12. Click Save.
13. Repeat steps 3 - 13 for the other two SQL queries you identified.
Create a custom alert to display the name of the active node in a SQL Server cluster

Custom alerts in DPA are user-defined SQL statements or stored procedures that run against a monitored database or the DPA repository. Each SQL statement or procedure returns a number (or set of numbers) that can trigger an alert depending on user-defined thresholds. Custom alerts can be used to alert against a wide variety of conditions.

This example shows how to configure two alerts that work together to cause DPA to display the physical machine name of the active node in a SQL Server failover cluster. By default, DPA shows only the cluster name.

- The first alert gets the name of the active node.
- The second alert appends this name to the name of the cluster that DPA displays. If the active node changes, it updates the name and also notifies recipients that a failover has occurred.

Task 1: Create an alert to get the physical machine name

This alert runs against a monitored database instance and retrieves the name of the physical machine that the instance is currently running on.

1. On the DPA menu, click Alerts.
2. Click the Manage Alerts tab.
3. Choose Custom as the category and Custom SQL Alert - Multiple Numeric Return as the type.

4. Click Create Alert.
5. Enter a unique name and select the execution interval.

Set the execution interval based on the average frequency of failovers in your clustered environment. The execution frequency affects the accuracy of the machine names that DPA displays. In this example, the alert interval is 10 minutes. But if failovers occur infrequently, you might want to choose a longer interval.
6. In the Notification Text box, provide a description of the alert.

This alert does not send notifications, but the description provides information for other users.

This alert gets the physical machine name for the active node in a SQL server cluster. It is used with the Change Physical Machine Name to display and update the active node name on the DPA homepage. The execution interval for this alert must be smaller than the interval for the Change Physical Machine Name alert.

7. Under Database Instances, select the SQL Server cluster.

8. Enter the following SQL in the SQL Statement box:

```sql
select coalesce(SERVERPROPERTY('ComputerNamePhysicalNetBIOS'),
SERVERPROPERTY('MachineName')) HOST, 0
```

9. Verify that Monitored Database is selected from the Execute Against drop-down menu.

10. Enter Machine Name 0 in the Units box.
11. Under Configure Alert Levels and Recipients:
   a. Enter 1 as the Min value for the High level.
      The query returns a numeric value of 0. Entering 1 as the threshold ensures that the status is always Normal. Because this alert only retrieves the machine name, it should not be triggered.
   b. Do not select a contact because no one needs to receive notifications.

12. Click Save.

Task 2: Create an alert to append the machine name to the cluster name

This alert runs against the DPA Repository database. It appends 'Node: @nodeName' to the database instance name that DPA displays. Each time it runs, it determines whether the node name has changed. If so, it updates the display name and sends a notification so that you can investigate why the failover occurred.

This alert can also determine why the name has changed with different error levels. The error levels are: 0=no change, 1=node change, 2=initial update.

1. From the Manage Alerts tab, choose Custom as the category and Custom SQL Alert - Multiple Numeric Return as the type.
2. Click Create Alert.

3. Enter a unique name and select the execution interval.

This execution interval must be **larger** than the interval for the Get Physical Machine Name alert. The execution intervals for both alerts affect the accuracy of the name that DPA displays. In this example, the execution intervals are 10 minutes and 12 minutes, but you should determine what intervals are appropriate for your environment.

4. Enter the email notification text.

   "The active node for the DPASQL2K12 cluster has changed. Investigate the reason for the failover."

5. Under Database Instances, select the SQL Server cluster.

   You must select the SQL Server cluster for both alerts. If you do not select it for this alert, the instance name is not updated in DPA. If you do not select it for the Get Physical Machine Name alert, that alert does not run and has a status of Broken.
6. Enter the following SQL in the SQL Statement box:

```sql
declare
@mach_name varchar(100),
@current_name varchar(100),
@update_flag smallint

begin
select @mach_name=c.name
from con_alert_db a, con_alert b, con_alert_db_results c
where b.id = a.alertid
and b.alertname = 'Get Physical Machine Name'
and a.alertid = c.ALERTID
and a.DBID = c.DBID
and a.DBID = #DBID#

if @mach_name is not null
    select @update_flag = CHARINDEX('  Node',NAME)
    from COND
    where ID = #DBID#

if @update_flag != 0
begin
select @current_name = substring(NAME, CHARINDEX('  Node',NAME) +7,
100)
from cond
where ID = #DBID#

if ltrim(rtrim(@current_name)) != ltrim(rtrim(@mach_name))
begin
    update COND
    set NAME = substring(NAME, 1,CHARINDEX('  Node',NAME)-1 )
    where ID = #DBID#

    update COND
    set NAME = NAME + '  Node: ' + @mach_name
    where ID = #DBID#

    select @mach_name,1
    end
else
begin
    update COND
    set NAME = NAME + '  Node: ' + @mach_name
    end
else
begin
    update COND
    set NAME = NAME + '  Node: ' + @mach_name
end
end
```
where ID = #DBID#

    select @mach_name,2
    end;
    select @mach_name, 0
    end;

7. Select Repository from the Execute Against drop-down menu.

8. Enter :Node|Error Level in the Units box.

9. Under Configure Alert Levels and Recipients:
   a. Enter 1 as the Min value for the High level.
      When the name of the active node changes, this alert is triggered at the High level.
   b. Select a contact to receive the email when this alert is triggered.

   i If you have not added the person or group as a contact in DPA, click Add Contact or Add Contact Group. See Create contacts and contact groups.

10. Click Save.
Create an alert group

An alert group defines a set of alerts to be run against a set of database instances. Alert groups simplify alert configuration and help make alerting more consistent across the monitored database instances. When you add alerts to an alert group, you do not have to select database instances within each alert definition. Instead, you select the database instances just once for the entire group. If the list of instances changes, you can update it in only one place.

1. On the DPA menu, click Alerts.
2. Click the Alert Groups tab.
3. Click Create Alert Group.
4. Enter a unique name and a brief description.

5. Select the alerts to include in this group.

6. Select the database instances on which to execute these alerts.

7. Click Save.

Create and manage DPA contacts and contact groups

Before you create DPA alerts or schedule reports, define the contacts and contact groups who can receive the alerts and reports. As your organization changes, you can edit or delete contacts or contact groups.
If you want to send DPA alerts as SNMP traps to your Network Management System (NMS), create an SNMP contact.

Create a contact

Contacts are people who can receive email notifications when an alert is triggered, or who can receive scheduled reports through email. When you define an alert or schedule a report, you can select the recipient from the list of available contacts.

1. In the DPA menu bar, click Options.
2. Under Administration > Users & Contacts, click Contact Management.
3. In the Email Contacts section, click Create Contact.
4. Enter the contact's name and email address. Optionally, add the contact to an existing group.

5. Click Save.

Create a contact group

Contact groups are used to send emails to multiple people when an alert is triggered or when a scheduled report runs. DPA provides several default contact groups, but you can create other groups.

1. In the DPA menu bar, click Options.
2. Under Administration > Users & Contacts, click Contact Management.
3. In the Groups section, click Create Group.
4. Enter a group name and description. Optionally, add existing contacts to the group.
Edit a contact or contact group

1. In the DPA menu bar, click Options.
2. Under Administration > Users & Contacts, click Contact Management.
3. Click the name of the contact or contact group to open the Update Contact or Update Group dialog box.
4. Make the necessary changes and click Save.

Delete a contact or contact group

1. In the DPA menu bar, click Options.
2. Under Administration > Users & Contacts, click Contact Management.
3. In the right column of the contact or group table, click Delete.
4. On the confirmation dialog, click Yes.

Notification policy for DPA alerts

When you create an alert in DPA, you can accept the default notification policy or apply a different policy to that alert. The notification policy determines when the alert is triggered. The following sections describe each policy.

Notify when level not visited since normal

A notification is sent if the alert status is not Normal and the alert has not been in this status since the last time the status was Normal. If the alert returns the same status for multiple polling periods without returning to Normal, you are notified only once for each status. For example:
<table>
<thead>
<tr>
<th><strong>EXECUTION INTERVAL</strong></th>
<th><strong>ALERT LEVEL</strong></th>
<th><strong>NOTIFICATION SENT?</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Medium</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>High</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>High</td>
<td>No (this alert level was returned previously)</td>
</tr>
<tr>
<td>5</td>
<td>Medium</td>
<td>No (this alert level was returned previously)</td>
</tr>
<tr>
<td>6</td>
<td>Low</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>Low</td>
<td>Yes</td>
</tr>
</tbody>
</table>

This is the default policy, but a DPA administrator can change the default policy in Advanced Options.

**Notify when level changes and is not normal**

A notification is sent if the alert status is not Normal and has changed since the previous execution interval. You are notified each time the level changes, but only once for each change. For example:

<table>
<thead>
<tr>
<th><strong>EXECUTION INTERVAL</strong></th>
<th><strong>ALERT LEVEL</strong></th>
<th><strong>NOTIFICATION SENT?</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Medium</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>High</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>High</td>
<td>No (the alert status has not changed)</td>
</tr>
<tr>
<td>5</td>
<td>Medium</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Low</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>Low</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Notify when level is not normal**

A notification is sent if the alert status is not Normal, regardless of the alert's previous status. For example:
<table>
<thead>
<tr>
<th>EXECUTION INTERVAL</th>
<th>ALERT LEVEL</th>
<th>NOTIFICATION SENT?</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Medium</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>High</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>High</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Medium</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Low</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Normal</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>Low</td>
<td>Yes</td>
</tr>
</tbody>
</table>
DPA reports

To work with DPA reports, see the following topics:

- About DPA reports
- Access and run DPA reports
- Create a DPA report
- Schedule a report for email delivery
- Create a report group

About DPA reports

Use reports to communicate the long-term performance of your databases and supply evidence to support your work. Reports can capture the results of performance tuning and highlight database trends. You can send reports to managers, team members, and customers.

Differences between report data and detailed chart data displayed in the DPA interface

The data shown in reports differs from the detailed chart data shown in the DPA interface.

<table>
<thead>
<tr>
<th>REPORT DATA</th>
<th>DETAILED CHART DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage period and granularity</td>
<td>Reports can show data captured over longer intervals and display long-term trends. To generate reports, DPA summarizes repository data to make long-term information available in a manageable size. The previous 90 days of data are summarized by hour. After 90 days, data are summarized by day. This information is available for five years.</td>
</tr>
</tbody>
</table>

Data collection period | Reports can be generated after a one-hour data collection period. SolarWinds recommends allowing a 24-hour data collection period before you create a report. | Charts display data after a 10-minute data collection period. |

Report types

SolarWinds DPA has many standard reports that include the most commonly used wait time statistics. You can customize the reports to meet your needs. Available reports include:
Top SQLs

This report shows the top five SQL statements ranked by total wait time across a custom interval. You can change the report to display the top 10 or select specific SQL statements from the SQL Statements tab on the Report Properties dialog. The report will show a descriptive name for the SQL if it has been defined.

Top Waits for single SQL

This report identifies the top wait events (up to 50), ranked by wait time, for a specific SQL statement.

Top SQLs for single Wait

This report identifies the top SQL statements for a single database wait.

Typical Day of Wait

This report displays a bar graph showing the average hourly wait time for a database. It displays the distribution of average wait time versus time of day.

Typical Day of Wait for single SQL

This report displays a stacked graph showing average hourly wait time for the top waits contributing to the total wait time for an individual SQL statement. The top of the bar is the sum of all other waits. This report can identify the peak loads during your business hours based on long-term observation of the system.

Top Waits

This report charts waits for an entire database. SolarWinds DPA displays the waits with the highest accumulated wait time.

Top Programs

This report displays the total accumulated wait time for programs.

Top Files

This report displays the busiest files ranked by total I/O wait time. SolarWinds DPA calculates the total wait time for all I/O operations on each file for selected intervals.

Access and run DPA reports

From the Reports tab, you can view existing reports and create new reports.

1. From the DPA menu, click Reports.

   The Reports section lists the reports that have been created on this DPA server.

2. In the Reports section, you can:
   - Choose a database instance from the drop-down menu in the upper-right corner to filter the list of reports.
   - Click Show to run and open a report.
• Click a column heading to sort the list of reports.
• In the right column, click Delete to delete a report.

Create a DPA report

Use DPA reports to identify database trends and track the results of your performance tuning.

1. In the DPA menu bar, click Reports.
2. Select the Database Instance and the Report Type.

   Database Instance: DPAORA11R2_DPA0RA11R2
   Report Type: Top Waits for single SQL

3. Depending on the report type, specify the waits, SQL statements, or other elements to display in the report.

   Waits to Display
   □ Top Waits Ranked by Cumulative Wait Time
     Top 10 Waits
   □ User-Defined Waits

4. Under Dates to Display, specify the dates that the report should include.

   The Data Range at the bottom of this section shows the time period for which data is available.

5. Under General, complete the following fields.

7. Choose one of the following options:

<table>
<thead>
<tr>
<th>CLICK</th>
<th>IF YOU WANT TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save</td>
<td>Save the report with the name you entered previously.</td>
</tr>
<tr>
<td>Save As</td>
<td>Save the report with a different name.</td>
</tr>
<tr>
<td>Edit</td>
<td>Return to the Report - Advanced Options page and make changes.</td>
</tr>
<tr>
<td>Email Report</td>
<td>Send the report to one or more users.</td>
</tr>
</tbody>
</table>

You can view the report from the Reports tab at any time, or schedule the report to run automatically and be emailed to a group of recipients.

**Schedule a report for email delivery**

Report schedules automatically email reports (or report groups) at regular intervals. You can send reports to managers, team members, and customers. Use report schedules to communicate database trends to people who do not have direct access to DPA, and to and highlight performance improvements across your organization.

- Only DPA administrators can create report schedules.
- The report recipients must be added as DPA contacts.

1. Log in to DPA using an account with administrator privileges.
2. In the DPA menu bar, click Reports.
3. Click the Report Schedules tab.
4. Click Create Schedule.
5. Name the schedule and enter an email subject and body text.
6. Specify when you want the report delivered, and click Add. You can specify multiple delivery times.

7. Under Available Reports, select the reports or report groups, and click Add.

8. If you want to review the email that will be sent when the schedule runs, click Send Test Email and enter an email address.

9. Under Available Contacts, select the recipients of the report, and click Add.

   If you have not added the recipients as contacts in DPA, click Add Contact or Add Contact Group.
10. Click Create Schedule.
    
    Your schedule is added to the list of report schedules.

Does your network or firewall require an internal SMTP server? If so, see [SMTP mail server for outgoing email](#).

**Create a report group**

Use report groups to display data from related reports on the same page. With report groups, you can quickly run or schedule multiple reports.

1. On the DPA menu bar, click Reports.
2. Click the Report Groups tab.
3. Click Create Report Group.
4. Give the group a name and (optionally) a description.

![Create Report Group](image)

5. Select the reports to include in this group and click Add.
6. Click OK.

This group is added to the list of report groups.
Link together separate DPA servers

Use Central Server mode to link separate SolarWinds DPA servers together. This is useful if you want to monitor more than 250 database instances, or if your monitored databases are distributed geographically. See the following topics:

- Set up a Central Server and add remote DPA servers
- Configure authentication for a DPA Central Server
- View information from remote servers on the Central Server
- Central Server advanced configuration

Set up a Central Server and add remote DPA servers

Use Central Server mode to link separate SolarWinds DPA servers together. This is useful if:

- You want to monitor more than 250 database instances. You can divide monitoring tasks between different DPA servers.
- Your monitored databases are distributed geographically. You can install separate DPA servers in each location.

The Central Server collects information from your remote servers and consolidates the data into a single interface. The Central Server has low overhead and no additional information is added to its repository database.

Set up a Central Server

1. Install SolarWinds DPA on a server. This will be your Central Server.
2. Log in to that instance as an administrator.
3. In the DPA menu bar, click Options.
4. Under Administration > Display, click Manage Central.
   Your SolarWinds DPA server should be listed as the Central DPA Server in the list of Registered Servers.

Add remote DPA servers

The user credentials for the Central and remote DPA servers must match. See Configure authentication for a DPA Central Server for more information.

1. In the DPA menu bar, click Options.
2. Under Administration > Display, click Manage Central.
3. Click Add Server.
4. Enter information about the remote SolarWinds DPA server.
5. Click Test connection, and click Save.

A successful test indicates that SolarWinds DPA can communicate with the remote server through the provider host and port. It does not indicate that DPA can authenticate users.

If the test fails, check the host name in the Server Name field. Does it contain an underscore (_) character? An underscore is not valid for host names. If you cannot rename the host, enter the IP address.

6. Repeat steps 1 - 4 for the remaining remote SolarWinds DPA servers.

The details of your remote SolarWinds DPA servers are not stored in the repository, but in a file on your Central Server, located here:

```plaintext
<DPA_Home>/iwc/tomcat/ignite_config/iwc/central/RemoteRepositories.json
```

This is a plain-text JavaScript Object Notation (JSON) file. No sensitive data is stored in this file.

### Configure authentication for a DPA Central Server

You can authenticate to the [Central Server](#) and the remote servers using one account. The account must be added to each server as a SolarWinds DPA user, or through an Active Directory (AD) or LDAP group.

**Log in with a SolarWinds DPA user**

You must create the user on the Central Server and each remote server. See [Create a DPA user account and assign privileges](#) for more information.

The password must match on all servers.

Read-only permissions are sufficient to view data from the remote repositories.

**Log in with an Active Directory or LDAP user**

You must first set up AD or LDAP on the Central Server and each remote server. See [User authentication options](#) for more information.

Next, create the AD or LDAP group of the user on the Central Server and each remote server. See [Create and manage DPA contacts and contact groups](#) for more information.

Read-only permissions are sufficient to view data from the remote repositories.

**View information from remote servers on the Central Server**

The default homepage of a [Central Server](#) is the SolarWinds DPA homepage. Navigate to the Central Server page to see database information from all registered remote servers.

1. Log in to your DPA Central Server as an administrator.
2. In the menu, click Central.
Central Server advanced configuration

You may need to change the Central Server configuration to make it run more efficiently in your environment.

To change the default behavior, you can edit the system.properties file in the /iwc/tomcat/ignite_config/idc directory of your Central Server and add the desired setting.

General Central Server settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.confio.iwc.central.enabled</td>
<td>true (default) false</td>
<td>Enables or disables the use of Central Server mode.</td>
</tr>
<tr>
<td>com.confio.iwc.token.login.supported</td>
<td>true (default) false</td>
<td>Enables or disables the use of encrypted login tokens when jumping from the Central Server to a remote instance. If true, a web service call authenticates the user and creates a temporary token to identify the incoming user and bypass the login process. If false, the user is always prompted to log in to the remote instance.</td>
</tr>
<tr>
<td>com.confio.iwc.show.all.errors</td>
<td>true false (default)</td>
<td>Determines which users see failures in the Unavailable SolarWinds DPA Servers section. If true, all users see failures for all instances. If false, only administrators see failures. Set this option to false if you do not want all users to know about other SolarWinds DPA instances in the organization.</td>
</tr>
<tr>
<td>com.confio.iwc.automatic.update</td>
<td>true (default) false</td>
<td>Enables or disables a process that performs simple checks on the file when SolarWinds DPA starts. For example, flagging any local instances as the Central Server.</td>
</tr>
</tbody>
</table>
**com.confio.iwc.alarm.level**

**Value:** Warning

The minimum message level to include on the Alarm Details tab. Valid values are below. If (empty) is set, details are disabled.

- Critical
- Warning
- Normal
- (empty)

**com.confio.iwc.alarm.count**

**Value:** 200

The number of detail rows to show on the Alarm Details tab.

---

**Thread pool settings**

These settings control the number of threads that are used by the Central Server to make web service calls to other remote servers. The default settings are set for a few concurrent users hitting up to 100 remote instances. If you have more than 100 instances or many concurrent users, SolarWinds recommends adjusting these settings higher.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.confio.iwc.centralServiceTaskExecutor.corePoolSize</td>
<td>20</td>
<td>The core number of threads that Central Server uses to make web service calls to remote servers.</td>
</tr>
<tr>
<td>com.confio.iwc.centralServiceTaskExecutor.maxPoolSize</td>
<td>40</td>
<td>The maximum number of threads that Central Server uses to make web service calls to the remote servers. Central Server adds more threads only when all core threads are in use and the task queue is full.</td>
</tr>
<tr>
<td>com.confio.iwc.centralServiceTaskExecutor.queueCapacity</td>
<td>1000</td>
<td>The maximum number of requests in the queue before Central Server either creates new threads to help with the work or rejects the request. Tasks are rejected if all 40 threads cannot keep up with the requests being made.</td>
</tr>
<tr>
<td>com.confio.iwc.centralServiceTaskExecutor.keepAliveSeconds</td>
<td>120</td>
<td>The number of seconds to keep an idle thread before removing it.</td>
</tr>
</tbody>
</table>

---
Client factory cache

A client factory creates web service clients that talk to remote instances on a per-user basis. One client factory is created per host:port combination (not per user), so the same factory is used to create individual clients for different users. Factory creation is resource-intensive because an initial handshake is done between the client and server, and kept in a cache for reuse.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.confio.iwc.client.factory.cache.size</td>
<td>100</td>
<td>The maximum number of client factories held in the cache.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is 100, which equates to 100 unique remote SolarWinds DPA instances.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Increase this value if you are connecting to more than 100 remote instances.</td>
</tr>
<tr>
<td>com.confio.iwc.client.factory.cache.timeout</td>
<td>1800</td>
<td>The number of seconds a client factory remains in the cache without being used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is 1800 seconds, which is equal to 30 minutes.</td>
</tr>
<tr>
<td>com.confio.iwc.client.factory.connection.timeout</td>
<td>15</td>
<td>The number of seconds a client attempts to establish a connection before it times out.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is 15.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Zero (0) specifies that the client will continue to attempt to open a connection indefinitely.</td>
</tr>
<tr>
<td>com.confio.iwc.client.factory.read.timeout</td>
<td>30</td>
<td>The number of seconds the client waits for a response before it times out.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is 30 seconds.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Zero (0) specifies that the client will wait indefinitely.</td>
</tr>
<tr>
<td>com.confio.iwc.client.factory.enable.chunking</td>
<td>true</td>
<td>Enables or disables HTTP chunking.</td>
</tr>
<tr>
<td></td>
<td>false (default)</td>
<td>False is the safer option.</td>
</tr>
<tr>
<td>com.confio.iwc.client.factory.enable.log</td>
<td>true (default)</td>
<td>Enables logging of inbound and outbound messaging to capture the web service calls. Log levels are still controlled in the log4j.xml file.</td>
</tr>
<tr>
<td></td>
<td>false</td>
<td>Set this value to false to disable logging.</td>
</tr>
</tbody>
</table>
Use the DPA REST API

Use the DPA REST API to securely connect to the DPA server and issue commands. DPA API calls can retrieve information and automate management tasks, such as registering database instances, stopping and starting monitors, adding annotations, and allocating licenses.

See the following topics to learn more about using the DPA API:

- Manage tokens used for authentication to the DPA API
- Learn about and experiment with the DPA API
- Examples of using Python scripts to make DPA API calls
- Examples of using PowerShell scripts to make DPA API calls

Manage tokens used for authentication to the DPA API

Two types of tokens are required to authenticate requests to the DPA API:

- An **access token** is required to make authenticated calls to the DPA API. Access tokens are obtained when needed (for example, at the beginning of a script that makes API calls, or when you use the Swagger interface to experiment with the DPA API).

  By default, access tokens expire after 900 seconds. You can change the default through the advanced option API_ACCESS_TOKEN_EXPIRATION. Access tokens also expire if the DPA server is rebooted.

- A **refresh token** is used to obtain access tokens. Refresh tokens are obtained through the DPA interface by an administrator and stored in a secure location, as described below.

  Refresh tokens typically have long lifespans. When you create a refresh token, you can specify the expiration date or set it to never expire. The default expiration date is after 90 days. You can change the default through the advanced option API_REFRESH_TOKEN_EXPIRATION.

Create a refresh token

1. Log in to DPA as a user with administrative privileges.
2. In the DPA menu bar, click Options.
4. On the API Refresh Token Management page, click Create token.
5. Enter a name and specify when the token expires.

  By default, refresh tokens for the DPA API expire after 90 days. However, you can choose to create refresh tokens that never expire.
6. Click Create. The token string is displayed.

7. Click Copy to clipboard, and then click Close.

If you create a refresh token and fail to copy the string or lose the copied string, the refresh token cannot be used. Delete that token and create a new one.

About storing refresh tokens

Store refresh tokens in a secure location, such as a password-protected file system or an encrypted database. Limit access to users who need the tokens to make API calls.

If you believe that a refresh token has been accessed by an unauthorized user, delete it and create a new one.
Delete a refresh token

You can delete a refresh token at any time. For example, you can delete refresh tokens that have expired. If you delete a refresh token that has not expired, any access tokens obtained using that refresh token are invalidated and can no longer be used.

1. Log in to DPA as a user with administrative privileges.
2. In the DPA menu bar, click Options.
4. On the API Refresh Token Management page, select one or more tokens.
5. Click Delete.

Learn about and experiment with the DPA API

The DPA API is documented in the Swagger interface. Use this interactive interface to explore the available API endpoints and try out API calls.

You can also see examples of Python and PowerShell scripts that call the DPA API.

Access the DPA API documentation and get authorization to make API calls

You can access the URL and review the DPA API documentation without being authorized to make API calls. However, an access token is required to make API calls. Complete the following steps to access the API documentation and authenticate with an access token.

1. Create a new refresh token and copy it to the clipboard, or copy an existing refresh token that your organization has stored in a secure location.
2. In the DPA menu bar, click Options.
   The Swagger interface opens.
4. Use the refresh token to obtain an access token:
   a. From the Select a spec menu on the Swagger interface, make sure that Access Token API is selected.

   ![Select a spec menu](image)

   b. Click Access Token to expand it.

   ![Access Token interface](image)

   c. Click Post to expand that section.
   d. Click the Try it out button.
   e. Paste the refresh token value you copied in step 1 into the refresh_token box.

   ![Refresh token input](image)

   f. Click Execute.
   g. Copy the access token from the Response body.

5. Authenticate with the access token:
   a. From the Select a spec drop-down menu, choose Management API.

   ![Select a spec menu](image)

   b. Click Authorize.
   c. In the Available authorizations dialog, type `bearer` followed by a space, and then paste the access token.
d. Click Authorize.

If the authorization is successful, the following dialog is displayed.

```
Available authorizations

Bearer (apiKey)

Authorized

Name: Authorization
In: header
Value: bearer eyJhbGciOiJIUzI1NIs

Authorize Close
```

e. Click Close.

You can now use the Swagger interface to learn about and execute the available API commands.

**View the DPA API documentation**

The Management API spec provides detailed information about each API endpoint. Endpoints are grouped by function.
1. Click any group to display the endpoints within it.

```
<table>
<thead>
<tr>
<th>Annotatons</th>
<th>Manage Annotations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Monitor</td>
<td>Manage monitored database instances.</td>
</tr>
<tr>
<td><strong>Database Registration</strong></td>
<td>Register or unregister database instances for monitoring.</td>
</tr>
<tr>
<td>POST /databases/register-monitor</td>
<td>Register a new database instance.</td>
</tr>
<tr>
<td>POST /databases/unregister-monitor</td>
<td>Unregister a database instance.</td>
</tr>
</tbody>
</table>
```

2. Click the endpoint to display its parameters and responses.

Complex parameters and responses include an Example Value | Model section. The example value is shown by default.

```
| POST /databases/register-monitor | Register a new database instance. |
| Parameters | Try it out |
| Name | Description |
| registeredDatabase | The information required to register the database instance. See the RegisterDatabase model. |
| (body) | Example Value | Model |

```json
{
  "amazonDb": true,
  "databaseType": "AZURESQLDB",
  "servername": "string",
  "serviceNameOrSID": "string",
  "port": 1433,
  "sysadminNameOrSid": "string",
  "sysadminPassword": "string",
  "monitoringUser": "string",
  "monitoringUserPassword": "string",
  "monitoringUserIsNew": true,
}
```

3. Click Model to display additional information, including the valid values for enumerations.

```
| POST /databases/register-monitor | Register a new database instance. |
| Parameters | Try it out |
| Name | Description |
| registeredDatabase | The information required to register the database instance. See the RegisterDatabase model. |
| (body) | Example Value | Model |

**RegisterDatabase**

- **amazonDb**
- **databaseType**
- **servername**
- **serviceNameOrSID**
- **port**
- **sysadminNameOrSid**
- **sysadminPassword**
- **monitoringUser**
- **monitoringUserPassword**
- **monitoringUserIsNew**

**Valid enum values**

- Azure DB
- DB2
- MySQL
- Oracle
- SQL Server
- SQL Server - The Service Name or IP (or Server/Instance) of the instance being registered
Make an API call from the Swagger interface

The following example shows how to make a call to get the current license allocation for a monitored database instance.

When you make an API call through the Swagger interface, the call affects your DPA server in the same way as it would if it were issued through a command or script.

1. If you do not know the database ID, complete the following steps to get it:
   a. Click Database Monitor to display the endpoints.
   b. Click GET/databases/monitor-information to expand it.
   c. Click Try it out, and then click Execute.
   d. Scroll through the Response body, find the database name, and make a note of the associated ID.

2. Click License Allocation to display the endpoints.

3. Click the GET/databases/{databaseId}/licenses endpoint to expand it.
4. Click Try it out.
   The interface displays a field for the parameter value and an Execute button.

5. Enter the database ID and click Execute.
   The Response body section shows the response from the DPA server, and the Curl section shows the Curl command (including the access token) that could be run to make this API call.

   ![Curl command to make this API call](image)
   ![Response from the DPA server](image)
Examples of using Python scripts to make DPA API calls

The following examples show Python scripts that call the DPA API to retrieve information and perform DPA management functions. The first examples are snippets that demonstrate each API call individually. The last example is a full script that shows how to put the snippets together into a working script.

You can call the DPA API with any programming language that can send HTTP requests. See this topic for PowerShell script examples.

Prerequisites

- Before you can use scripts to make API calls, you must create a refresh token.
- These examples use the Requests HTTP library for Python. This library must be installed for these examples to work. Installation instructions can be found on the Installation of Requests page.

If your DPA server does not use HTTPS or your certificates are self-signed

The examples all use HTTPS, which can cause problems if your DPA server is not configured to use HTTPS or if your certificates are self signed. If this is the case, you can do either of the following:

- Run the examples using HTTP.
- Change the verify_cert value to False in the configuration section to prevent verifying the server's TLS certificate.

```
# ================================================================================
# Configure the variables below for the DPA Host
# ================================================================================
base_url = "https://localhost:8124/iwc/api/
refresh_token = "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJNeVRva2VuIiwiaXN..."
verify_cert = False
# ================================================================================
```

Get an access token

The first step in using the API is to get an access token. An access token is required to make any API calls. This call POSTs the refresh token to DPA, which returns an access token to be used by all other API calls.

- If the call is successful, it prints out the data that was returned from DPA, including the access_token, and then goes on to create HTTP Headers that will contain the access token and other information to be used on subsequent calls.
- If the call is not successful it prints out the error message.

You must set the base_url and the refresh_token variables to match your environment.
# Configure the variables below for the DPA Host
base_url = "https://localhost:8124/iwc/api/
refresh_token = "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJNeVRva2VuIiwiaXN..."
verify_cert = True

# Get Access Token

def get_access_header(prefix_url, rfrsh_token):
    """
    Given a base url and a refresh token retrieve the access token
    and return a header object with it.
    :param prefix_url: the base url
    :param rfrsh_token: refresh token used to get access token
    :return: the request header that contains the access token
    :rtype: dict
    """

    auth_token_url = prefix_url + "security/oauth/token"
    grant_type = "refresh_token"

    payload = {"grant_type": grant_type, "refresh_token": rfrsh_token}
    try:
        # get an access token
        resp = requests.post(auth_token_url, data=payload, verify=verify_cert)
        resp.raise_for_status()
        resp_json = resp.json()

        token_type = resp_json["token_type"]
        access_code = resp_json["access_token"]

        headers = {"authorization": f"{token_type} {access_code}",
                   "content-type": "application/json;charset=UTF-8",
                   "accept": "application/json"
        }

        return headers
    except requests.exceptions.HTTPError as ex:
        print(ex)
        print(ex.response.text)
        # print(json.dumps(json.loads(ex.response.text), indent=2))
return None  # requests is bad return None, can't get access_code

# get the header that contains access token for authentication
header = get_access_header(base_url, refresh_token)
if header is None:
    sys.exit(0)

Database Monitor examples

The following examples show how to use all Database Monitor calls.

Get information about one monitored database instance

# Get information about a single monitored database instance
database_id = 1
monitor_url = f"{base_url}databases/{database_id}/monitor-information"
single_monitor = None
try:
    print(f"\n*** Get Monitor Information for database with id of {database_id} ***")
    response = requests.get(monitor_url, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    single_monitor = response_json["data"]
    print(json.dumps(single_monitor, indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Get Monitor Information for database with id of 1 ***
{
    "dbId": 1,
    "name": "DEV-DPA\SQLEXPRESS",
    "ip": "127.0.0.1",
    "port": "1433",
    "jdbcUrlProperties": "applicationIntent=readOnly",
    "connectionProperties": null,
    "databaseType": "SQL Server",
    "databaseVersion": "12.0.6205.1",
    "databaseEdition": "Enterprise Edition: Core-based Licensing (64-bit)",
    "monitoringUser": "ignite_next",
    "defaultDbLicenseCategory": "DPACAT2",
    "assignedDbLicenseCategory": "DPACAT2",
    "assignedVmLicenseCategory": null,
    "monitorState": "Monitor Stopped"}
Start and stop monitoring a database instance given its database ID

database_id = 1
monitor_url = f"{base_url}databases/{database_id}/monitor-information"
try:
    # Start monitoring a database instance given its database ID.
    print(f"*** Start Monitor for database {database_id} ***")
    body = {"command": "START"}
    response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
    print("Waiting 15 seconds...")
    time.sleep(15)

    # Stop monitoring a database instance given its database ID.
    print(f"*** Stop Monitor for database {database_id} ***")
    body = {"command": "STOP"}
    response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
    print("Waiting 15 seconds...")
    time.sleep(15)

eexcept requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

    # This will print out data like this:
    *** Start Monitor for database 1 ***
Get information about all monitored database instances

database_id = 1
monitor_url = f"{base_url}databases/monitor-information"
try:
    print("*** Get Information for all database instances ***")
    response = requests.get(monitor_url, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    data = response_json['data']
    print(json.dumps(data, indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Get information for all database instances ***
[
  {
    "dbId": 1,
    "name": "DEV-DPA\SQLEXPRESS",
    "ip": "127.0.0.1",
    "port": "1433",
    "jdbcUrlProperties": "applicationIntent=readOnly",
    "connectionProperties": null,
    "databaseType": "SQL Server",
    "databaseVersion": "12.0.6205.1",
    "databaseEdition": "Enterprise Edition: Core-based Licensing (64-bit)",
    "monitoringUser": "ignite_next",
    "defaultDbLicenseCategory": "DPACAT2",
    "assignedDbLicenseCategory": "DPACAT2",
    "assignedVmLicenseCategory": null,
    "monitorState": "Monitor Stopped",
    "oldestMonitoringDate": "2018-12-09T00:00:00.000-07:00",
    "latestMonitoringDate": "2019-01-07T00:00:00.000-07:00",
    "agListenerName": null,
    "agClusterName": null,
  },
Stop and start monitoring for all database instances

# Start monitoring all database instances.
monitor_url = f"{base_url}databases/monitor-status"
try:
    print("*** Starting all Monitors ***")
    body = {"command": "START"}
    response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
    print("Waiting 30 seconds...")
    time.sleep(30)
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# Stop monitoring all database instances.
try:
    print("*** Stopping all Monitors ***")
    body = {"command": "STOP"}
    response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
    print("Waiting 30 seconds...")
    time.sleep(30)
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Starting all Monitors ***
"SUCCESS"
Waiting 30 seconds...

*** Stopping all Monitors ***
"SUCCESS"
Waiting 30 seconds...

Update the user password for a monitored database instance

database_id = 1
monitor_url = f"{base_url}databases/{database_id}/update-password"
try:
    print(f"*** Update the Monitor password for database {database_id} ***")
    body = {"password": "NewPassword!"
    response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Update the Monitor password for database 1 ***
"SUCCESS"

License Allocation examples
The examples below show how to use all License Allocation calls.

Get information about currently installed licenses

license_url = f"{base_url}databases/licenses/installed"
try:
    print("\n*** Getting Installed license information with total amounts available for use and total amounts used ***")
    response = requests.get(license_url, headers=header, verify=verify_cert)
response.raise_for_status()
response_json = response.json()
data = response_json["data"]

print("licenseProduct licenseCategory licensesAvailable licensesConsumed")
print("------------ -------------- -------------- --------------")
for i in range(len(data)):
    print('{:15s}{:<16s}{:>17d}{:>17d}'.format(data[i]["licenseProduct"], data[i]["licenseCategory"], data[i]["licensesAvailable"], data[i]["licensesConsumed"]))

# This will print out data like this:

*** Getting Installed license information with total amounts available
for use and total amounts used ***

<table>
<thead>
<tr>
<th>licenseProduct</th>
<th>licenseCategory</th>
<th>licensesAvailable</th>
<th>licensesConsumed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPACAT1</td>
<td>DPA_DB</td>
<td>100</td>
<td>22</td>
</tr>
<tr>
<td>DPACAT2</td>
<td>DPA_DB</td>
<td>100</td>
<td>16</td>
</tr>
<tr>
<td>DPAAzureSQL</td>
<td>DPA_DB</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DPAVM</td>
<td>DPA_VM</td>
<td>100</td>
<td>12</td>
</tr>
</tbody>
</table>

Get license information for a single database instance

```python
database_id = 1
license_url = f"{base_url}databases/{database_id}/licenses"
try:
    print(f"\n*** Getting current license information for the database instance with
database ID of {database_id} ***")
    response = requests.get(license_url, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)
```

# This will print out data like this:

*** Getting current license information for the database instance with database ID of 1 ***

```json
{
    "serverName": "DEV-DPA",
    "overLicensed": false,
    "performanceLicenseProduct": "DPACAT2",
    "vmLicenseProduct": "DPAVM"
}
```
Update license information for a database instance

database_id = 1
license_url = f"{base_url}databases/{database_id}/licenses"

# Add a DPACAT2 and a DPAVM license
body = {"performanceLicenseProduct": "DPACAT2",
        "vmLicenseProduct": "DPAVM"}
try:
    print(f"\n*** Updating license for database id {database_id} ***")
    response = requests.put(license_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# Remove the DPAVM license
body = {"performanceLicenseProduct": "DPACAT2",
        "vmLicenseProduct": "REMOVE"}
try:
    print(f"\n*** Updating license for database id {database_id} ***")
    response = requests.put(license_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Updating license for database id 1 ***
{
    "serverName": "DEV-BOU-CALLEN",
    "overLicensed": false,
    "performanceLicenseProduct": "DPACAT2",
    "vmLicenseProduct": "DPAVM"
}
*** Updating license for database id 1 ***
{
    "serverName": "DEV-BOU-CALLEN",
    "overLicensed": false,
    "performanceLicenseProduct": "DPACAT2",
    "vmLicenseProduct": null
}
Annotation examples

The examples below show how to use all Annotation calls.

Get a list of annotations for the last 30 days

```python
# Gets a List of annotations for the last 30 days
database_id = 1
annotation_url = f"{base_url}databases/{database_id}/annotations"

# Dates are in ISO 8601 format (2018-12-31T12:00:00.000-07:00)
end_time = datetime.datetime.now()
start_time = end_time + datetime.timedelta(days=-30)
args = {"startTime": start_time.astimezone().isoformat(),
        "endTime": end_time.astimezone().isoformat()}

try:
    print("\n*** Getting Annotations for the last 30 days ***")
    response = requests.get(annotation_url, params=args, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Getting Annotations for the last 30 days ***
[
    
    "id": 112,  
    "title": "Test Title API", 
    "description": "Test Event created by DPA API", 
    "createdBy": "DPA API", 
    "time": "2018-12-11T10:35:07-07:00", 
    "type": "API"
},

    
    "id": 113, 
    "title": "Test Title API", 
    "description": "Test Event created by DPA API", 
    "createdBy": "DPA API", 
    "time": "2018-12-12T15:40-07:00", 
    "type": "API"
},
```

Create a new annotation

database_id = 1
annotation_url = f"{base_url}databases/{database_id}/annotations"

# Dates are in ISO 8601 format, no millis  (2018-12-31T12:00:00-07:00)
create_time = datetime.datetime.now().replace(microsecond=0)
body = {"title": "API Test Title",
        "description": "API Test Description",
        "createdBy": "Test API User",
        "time": create_time.astimezone().isoformat()}

try:
    print("\n*** Creating Annotation ***")
    response = requests.post(annotation_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Creating Annotation ***
{
    "id": 171,
    "title": "API Test Title",
    "description": "API Test Description",
    "createdBy": "Test API User",
    "time": "2019-01-09T11:04:33-07:00",
    "type": "API"
}

Delete an annotation

database_id = 1
annotation_id = 171
annotation_url = f"{base_url}databases/{database_id}/annotations/{annotation_id}"

try:
print(f"\n*** Deleting Annotation with id of {annotation_id} ***")
response = requests.delete(annotation_url, headers=header, verify=verify_cert)
response.raise_for_status()
if response.status_code == 204:
    print(f"Annotation with id of {annotation_id} deleted")
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Deleting Annotation with id of 171 ***
Annotation with id of 171 deleted

Database Registration examples

The examples below show how to use all Database Registration calls.

Register and unregister a SQL Server database instance for monitoring

This example registers a new SQL Server database instance, waits 60 seconds, and then unregisters the database instance.

# -------------------------------
# Register a SQL Server database instance for monitoring.
# -------------------------------
registration_url = f"{base_url}databases/register-monitor"
body = {"databaseType": "SQLSERVER",
        "serverName": "127.0.0.1",
        "port": "1433",
        "sysAdminUser": "sa",
        "sysAdminPassword": "Password",
        "monitoringUser": "dpa_test_m",
        "monitoringUserPassword": "Password",
        "monitoringUserIsNew": True,
        "displayName": "DPA_SQL2K12"}

new_db_id = None
try:
    print("\n*** Register SQL Server database ***")
    response = requests.post(registration_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    responseJson = response.json()
    data = responseJson["data"]
    new_db_id = data["databaseId"]
print(json.dumps(data, indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

print("Waiting 60 seconds...")
time.sleep(60)

# Unregister the SQL Server database instance.
registration_url = f"{base_url}databases/unregister-monitor"
body = {
    "databaseId": new_db_id,
    "removeMonitoringUser": True,
    "removeDatabaseObjects": True,
    "sysAdminUser": "sa",
    "sysAdminPassword": "Password"
}
try:
    print(f\n*** Unregister SQL Server database [{new_db_id}] ***)
    response = requests.post(registration_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    responseJson = response.json()
    print(json.dumps(responseJson["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Register SQL Server database ***
{
    "databaseId": 77,
    "result": "SUCCESS"
}
Waiting 60 seconds...

*** Unregister SQL Server database [77] ***
{
    "databaseId": 77,
    "result": "SUCCESS"}
Register and unregister an Oracle database instance for monitoring

This example registers a new Oracle database instance, waits 60 seconds, and then unregisters the database instance.

```python
# Register an Oracle database instance for monitoring.

registration_url = f"{base_url}databases/register-monitor"
body = {
    "databaseType": "ORACLE",
    "serverName": "127.0.0.1",
    "serviceNameOrSID": "DPA_ORA11R1",
    "port": "1521",
    "sysAdminUser": "system",
    "sysAdminPassword": "Password",
    "monitoringUser": "dpa_test_m",
    "monitoringUserPassword": "Password",
    "monitoringUserIsNew": True,
    "monitoringUserTableSpace": "USERS",
    "monitoringUserTempTableSpace": "TEMP",
    "oracleEBusinessEnabled": False,
    "displayName": "DPA_ORA11R1"
}

new_db_id = None
try:
    print("\n*** Register Oracle database ***")
    response = requests.post(registration_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    responseJson = response.json()
    data = responseJson["data"]
    new_db_id = data["databaseId"]
    print(json.dumps(data, indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

print("Waiting 60 seconds...")
time.sleep(60)

# Unregister the Oracle database instance.

registration_url = f"{base_url}databases/unregister-monitor"
body = {
    "databaseId": new_db_id,
    "removeMonitoringUser": True,
```
try:
    print(f"\n*** Unregister Oracle database [{new_db_id}] ***")
    response = requests.post(registration_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    responseJson = response.json()
    print(json.dumps(responseJson["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Register Oracle database ***
{
    "databaseId": 78,
    "result": "SUCCESS"
}
Waiting 60 seconds...

*** Unregister Oracle database [78] ***
{
    "databaseId": 78,
    "result": "SUCCESS"
}

Register and unregister a MySQL database instance for monitoring

This example registers a new MySQL database instance, waits 60 seconds, and then unregisters the database instance.

registration_url = f"{base_url}databases/register-monitor"
body = {
    "databaseType": "MYSQL",
    "serverName": "127.0.0.1",
    "port": "3306",
    "sysAdminUser": "root",
    "sysAdminPassword": "Password",
    "monitoringUser": "dpa_test_m",
    "monitoringUserPassword": "Password",}
"monitoringUserIsNew": True,
"displayName": "DPA_MYSQL56"

new_db_id = None

try:
    print("\n*** Register MySQL database ***")
    response = requests.post(registration_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    responseJson = response.json()
    data = responseJson["data"]
    new_db_id = data["databaseId"]
    print(json.dumps(data, indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

print("Waiting 60 seconds...")
time.sleep(60)

# Unregister the MySQL database instance.

registration_url = f"{base_url}databases/unregister-monitor"
body = {
    "databaseId": new_db_id,
    "removeMonitoringUser": True,
    "removeDatabaseObjects": True,
    "sysAdminUser": "root",
    "sysAdminPassword": "Password"
}

try:
    print(f"\n*** Unregister MySQL database [{new_db_id}] ***")
    response = requests.post(registration_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    responseJson = response.json()
    print(json.dumps(responseJson["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will print out data like this:
*** Register MySQL database ***
{
    "databaseId": 79,
"result": "SUCCESS"
}
Waiting 60 seconds...

*** Unregister MySQL database [79] ***
{
  "databaseId": 79,
  "result": "SUCCESS"
}

Full working script

The following script combines all of the examples shown above into a script that can be run.

```python
import json
import sys
import time
import datetime
import requests

# ==============================================================
# Configure the variables below for the DPA Host
# ==============================================================
base_url = "http://localhost:8123/iwc/api/
refresh_token = "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJNeVRva2VuIiwiaXN..."
verify_cert = True
# ==============================================================

# ==============================================================
# Get Access Token
# ==============================================================
def get_access_header(prefix_url, rfrsh_token):
    """
    Given a base url and a refresh token retrieve the access token
    and return a header object with it.
    :param prefix_url: the base url
    :param rfrsh_token: refresh token used to get access token
    :return: the request header that contains the access token
    :rtype: dict
    """

    auth_token_url = prefix_url + "security/oauth/token"
    grant_type = "refresh_token"
    payload = {"grant_type": grant_type, "refresh_token": rfrsh_token}
```
try:
    # get an access token
    resp = requests.post(auth_token_url, data=payload, verify=verify_cert)
    resp.raise_for_status()
    resp_json = resp.json()

    token_type = resp_json['token_type']
    access_code = resp_json['access_token']

    headers = {
        "authorization": f"{token_type} {access_code}",
        "content-type": "application/json; charset=UTF-8",
        "accept": "application/json"
    }

    return headers

except requests.exceptions.HTTPError as ex:
    print(ex)
    print(ex.response.text)
    # print(json.dumps(json.loads(ex.response.text), indent=2))
    return None  # requests is bad return None, can't get access_code

# get the header that contains access token for authentication
header = get_access_header(base_url, refresh_token)
if header is None:
    sys.exit(0)

# =========================================================
# Database Monitor Examples
# =========================================================

# Calls for individual monitors...

# Get Monitor Information for a single database instance
database_id = 1
monitor_url = f"{base_url}databases/{database_id}/monitor-information"
single_monitor = None
try:
    print(f"\n*** Get Monitor Information for database with id of {database_id} ***")
    response = requests.get(monitor_url, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    single_monitor = response_json['data']
    print(json.dumps(single_monitor, indent=2))

except requests.exceptions.HTTPError as e:
print(e)
print(e.response.text)

# Start or Stop monitoring a database instance given its database ID.
# If it is already running stop it and then restart it
# If it is not running start it and then stop it
if single_monitor is not None:
    monitor_url = f"{base_url}databases/{database_id}/monitor-status"
    if single_monitor["monitorState"] == "Monitor Running":
        change_command = "STOP"
        revert_command = "START"
    elif single_monitor["monitorState"] == "Monitor Stopped":
        change_command = "START"
        revert_command = "STOP"
    else:
        change_command = None
        revert_command = None

if change_command is not None:
    try:
        print(f"\n*** {change_command} Monitor for database {database_id} ***")
        body = {"command": change_command}
        response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
        response.raise_for_status()
        response_json = response.json()
        print(json.dumps(response_json["data"], indent=2))

        print("Waiting 15 seconds...")
        time.sleep(15)

        print(f"\n*** {revert_command} Monitor for database {database_id} ***")
        body = {"command": revert_command}
        response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
        response.raise_for_status()
        response_json = response.json()
        print(json.dumps(response_json["data"], indent=2))

        print("Waiting 15 seconds...")
        time.sleep(15)

    except requests.exceptions.HTTPError as e:
        print(e)
        print(e.response.text)
# Calls for all monitors...

# Get Monitor Information for all database instances
database_id = 1
running_ids = []
monitor_url = f"{base_url}databases/monitor-information"
try:
    print("\n*** Get Information for a all database instances ***")
    response = requests.get(monitor_url, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    data = response_json["data"]
    print(json.dumps(data, indent=2))

    # Keep a list of running or started monitors to be used later
    for monitor in data:
        state = monitor["monitorState"]
        if state == "Monitor Running" or state == "Monitor Start No License" or 'Start' in state:
            running_ids.append(monitor["dbId"])

    print(f"Running Monitors: {running_ids}"")
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# Start monitoring all database instances.
monitor_url = f"{base_url}databases/monitor-status"
try:
    print("\n*** Starting all Monitors ***")
    body = {"command": "START"}
    response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
    print("Waiting 30 seconds...")
    time.sleep(30)
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)
# Stop monitoring all database instances.
try:
    print("\n*** Stopping all Monitors ***")
    body = {"command": "STOP"}
    response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
    print("Waiting 30 seconds...")
    time.sleep(30)
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# Try to put it back the way we found it by restarting the ones that were running
for db_id in running_ids:
    try:
        print(f"\n*** Starting Monitor for database {db_id} ***")
        monitor_url = f"{base_url}databases/{db_id}/monitor-status"
        body = {"command": "START"}
        response = requests.put(monitor_url, json=body, headers=header, verify=verify_cert)
        response.raise_for_status()
        response_json = response.json()
        print(json.dumps(response_json["data"], indent=2))
    except requests.exceptions.HTTPError as e:
        print(e)
        print(e.response.text)

# Update the monitor database user password (Un-comment to use)
# $monitorURL = $baseURL + "databases/$databaseId/update-password"
# Try {
#     Write-Host "Update the Monitor password for database $databaseId..."
#     $command = @{"password" = "NewPassword!"} | ConvertTo-Json
#     $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
#     $result = $monitorJSON.data
#     Write-Host "Result: $result\r\n"
# }
# Catch {
#     handleError $Error[0]
# }

# Licensing Examples
# Get the currently installed license information
license_url = f"{base_url}databases/licenses/installed"
try:
    print("\n*** Getting Installed license information with total amounts available for use and total amounts used ***")
    response = requests.get(license_url, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    data = response_json["data"]
    print("licenseProduct licenseCategory licensesAvailable licensesConsumed")
    print("-------------------- -------------------- --------------------")
    for i in range(len(data)):
        print('{}:<15s}{:<16s}{:>17d}{:>17d}'.format(data[i]['licenseProduct'], data[i]['licenseCategory'], data[i]['licensesAvailable'], data[i]['licensesConsumed']))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# Get License Information for a single database
license_url = f"{base_url}databases/{database_id}/licenses"
license_info = None
try:
    print(f"\n*** Getting current license information for the database instance with database ID of {database_id} ***")
    response = requests.get(license_url, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    license_info = response_json["data"]
    print(json.dumps(license_info, indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# This will Update License Information for a single database setting the # Performance License and the VM License to what it currently is. # It should succeed but it should make no changes.
if license_info is not None:
    database_id = 1
    license_url = f"{base_url}databases/{database_id}/licenses"
    db_product = license_info["performanceLicenseProduct"]
vm_product = license_info["vmLicenseProduct"]
body = {
    "performanceLicenseProduct": db_product,
    "vmLicenseProduct": vm_product
}

try:
    print(f"\n*** Updating license for database id {database_id} ***")
    response = requests.put(license_url, json=body, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# =========================================================
# Annotation Examples
# ==============================================================

# Gets a List of annotations for the last 30 days
annotation_url = f"{base_url}databases/{database_id}/annotations"

# Dates are in ISO 8601 format ( 2018-12-31T12:00:00.000-07:00 )
end_time = datetime.datetime.now()
start_time = end_time + datetime.timedelta(days=-30)
args = {
    "startTime": start_time.astimezone().isoformat(),
    "endTime": end_time.astimezone().isoformat()
}

try:
    print("\n*** Getting Annotations for the last 30 days ***")
    response = requests.get(annotation_url, params=args, headers=header, verify=verify_cert)
    response.raise_for_status()
    response_json = response.json()
    print(json.dumps(response_json["data"], indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# Create a new annotation
annotation_url = f"{base_url}databases/{database_id}/annotations"
annotation_id = None

# Dates are in ISO 8601 format, no millis ( 2018-12-31T12:00:00-07:00 )
create_time = datetime.datetime.now().replace(microsecond=0)
body = {
    "title": "API Test Title",
    "description": "API Test Description",}
try:
    print("\n*** Creating Annotation ***")
response = requests.post(annotation_url, json=body, headers=header, verify=verify_cert)
response.raise_for_status()
response_json = response.json()
data = response_json["data"]
annotation_id = data["id"]
print(json.dumps(data, indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# Delete an annotation
if annotation_id is not None:
    annotation_url = f"{base_url}databases/{database_id}/annotations/{annotation_id}"
try:
    print(f"\n*** Deleting Annotation with id of {annotation_id} ***")
    response = requests.delete(annotation_url, headers=header, verify=verify_cert)
    response.raise_for_status()
    if response.status_code == 204:
        print(f"Annotation with id of {annotation_id} deleted")
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

# ===============================================
# Registration Examples
# ===============================================

# Register a SQL Server database instance for monitoring.
# -----------------------------------------------
# registration_url = f"{base_url}databases/register-monitor"
# body = {
#   "databaseType": "SQLSERVER",
#   "serverName": "127.0.0.1",
#   "port": "1433",
#   "sysAdminUser": "sa",
#   "sysAdminPassword": "Password",
#   "monitoringUser": "dpa_test_m",
#   "monitoringUserPassword": "Password",
#   "monitoringUserIsNew": True,
#   "displayName": "DPA_SQL2K12"}
# new_db_id = None
# try:
#     print("\n*** Register SQL Server database ***")
#     response = requests.post(registration_url, json=body, headers=header, verify=verify_cert)
#     response.raise_for_status()
#     responseJson = response.json()
#     data = responseJson["data"]
#     new_db_id = data["databaseId"]
#     print(json.dumps(data, indent=2))
# except requests.exceptions.HTTPError as e:
#     print(e)
#     print(e.response.text)
#     print("Waiting 60 seconds...")
#     time.sleep(60)

# Un-register the SQL Server database instance.
# registration_url = f"{base_url}databases/unregister-monitor"
# body = {"databaseId": new_db_id,
#          "removeMonitoringUser": True,
#          "removeDatabaseObjects": True,
#          "sysAdminUser": "sa",
#          "sysAdminPassword": "Password"}
# try:
#     print(f"\n*** Unregister SQL Server database [{new_db_id}] ***")
#     response = requests.post(registration_url, json=body, headers=header, verify=verify_cert)
#     response.raise_for_status()
#     responseJson = response.json()
#     print(json.dumps(responseJson["data"], indent=2))
# except requests.exceptions.HTTPError as e:
#     print(e)
#     print(e.response.text)

# Register an Oracle database instance for monitoring.
# registration_url = f"{base_url}databases/register-monitor"
# body = {"databaseType": "ORACLE",
#          "serverName": "127.0.0.1",
#          "serviceNameOrSID": "DPA ORA11R1",}
new_db_id = None
try:
    print("\n*** Register Oracle database ***")
    response = requests.post(registration_url, json=body, headers=header,
                            verify=verify_cert)
    response.raise_for_status()
    responseJson = response.json()
    data = responseJson["data"]
    new_db_id = data["databaseId"]
    print(json.dumps(data, indent=2))
except requests.exceptions.HTTPError as e:
    print(e)
    print(e.response.text)

print("Waiting 60 seconds...")
time.sleep(60)

-----------------------------
Un-register the Oracle database instance.
-----------------------------
registration_url = f"{base_url}databases/unregister-monitor"
body = {"databaseId": new_db_id,
        "removeMonitoringUser": True,
        "removeDatabaseObjects": True,
        "sysAdminUser": "system",
        "sysAdminPassword": "Password"}
try:
    print(f"\n*** Unregister Oracle database [{new_db_id}] ***")
    response = requests.post(registration_url, json=body, headers=header,
                              verify=verify_cert)
    response.raise_for_status()
    responseJson = response.json()
    print(json.dumps(responseJson["data"], indent=2))
# except requests.exceptions.HTTPError as e:
#     print(e)
#     print(e.response.text)

# Register a MySQL database instance for monitoring.
# registration_url = f"{base_url}databases/register-monitor"
# body = {"databaseType": "MYSQL",
#          "serverName": "127.0.0.1",
#          "port": "3306",
#          "sysAdminUser": "root",
#          "sysAdminPassword": "Password",
#          "monitoringUser": "dpa_test_m",
#          "monitoringUserPassword": "Password",
#          "monitoringUserIsNew": True,
#          "displayName": "DPA_MYSQL56"}
#
# new_db_id = None
# try:
#     print("\n*** Register MySQL database ***")
#     response = requests.post(registration_url, json=body, headers=header,
#                                verify=verify_cert)
#     response.raise_for_status()
#     responseJson = response.json()
#     data = responseJson["data"]
#     new_db_id = data["databaseId"]
#     print(json.dumps(data, indent=2))
# except requests.exceptions.HTTPError as e:
#     print(e)
#     print(e.response.text)
#
# print("Waiting 60 seconds...")
# time.sleep(60)

# Un-register the MySQL database instance.
# registration_url = f"{base_url}databases/unregister-monitor"
# body = {"databaseId": new_db_id,
#          "removeMonitoringUser": True,
#          "removeDatabaseObjects": True,
#          "sysAdminUser": "root",
#          "sysAdminPassword": "Password"}
# try:
#     print(f"\n*** Unregister MySQL database [{new_db_id}] ***")
# response = requests.post(registration_url, json=body, headers=header, verify=verify_cert)
# response.raise_for_status()
# responseJson = response.json()
# print(json.dumps(responseJson["data"], indent=2))
# except requests.exceptions.HTTPError as e:
#    print(e)
#    print(e.response.text)

## Examples of using PowerShell scripts to make DPA API calls

The following examples show PowerShell scripts that call the DPA API to retrieve information and perform DPA management functions. The first examples are snippets that demonstrate each API call individually. The last example is a full script that shows how to put the snippets together into a working script.

> You can call the DPA API with any programming language that can send HTTP requests. See [this topic](#) for Python script examples.

### Prerequisite

Before you can use scripts to make API calls, you must [create a refresh token](#).

### If your DPA server does not use HTTPS or your certificates are self-signed

The examples all use HTTPS, which can cause problems if your DPA server is not configured to use HTTPS or if your certificates are self signed. If this is the case, you can do either of the following:

- Run the examples using HTTP.
- Add the following code below the configuration section.

```csharp
# Adding certificate exception to prevent API errors
#
add-type @
    using System.Net;
    public class TrustAllCertsPolicy : ICertificatePolicy {
        public bool CheckValidationResult(
            ServicePoint srvPoint, X509Certificate certificate,
            WebRequest request, int certificateProblem) {
            return true;
        }
    }
```

---

page 185
Get an access token

The first step in using the API is to get an access token. An access token is required to make any API calls. This call POSTs the refresh token to DPA, which returns an access token to be used by all other API calls.

- If the call is successful, it prints out the data that was returned from DPA, including the access_token, and then goes on to create HTTP Headers that will contain the access token and other information to be used on subsequent calls.
- If the call is not successful it prints out the error message.

You must set the $baseUrl and the $refreshToken variables to match your environment.

```powershell
# Configure the variables below for the DPA Host
(baseUrl = "https://localhost:8124/iwc/api/
$refreshToken = "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJNeVRva2VuIiwiaXN...

# Get an access token
$authTokenURL = $baseUrl + "security/oauth/token"
$body = @{"grant_type" = "refresh_token"

"refresh_token" = "$refreshToken"

Try {
    Write-Host "Getting Access Token..."
    $dpaAuthResponse = Invoke-RestMethod -Uri $authTokenURL -Method POST -Body $body
    $dpaAuthResponse | Format-List
}

Catch {
    $_.Exception.ToString()
    return
}

# If successful we will create our headers to be used for all API calls
$tokenType = $dpaAuthResponse.token_type
$accessToken = $dpaAuthResponse.access_token
$dpHeader = @{}
$dpHeader.Add("Accept", "application/json")
$dpHeader.Add("Content-Type", "application/json;charset=UTF-8")
$dpHeader.Add("Authorization", "$tokenType $accessToken")
```
Getting Access Token...

access_token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyX25hbWUiOiJpZ25pdGUiLCJzY29wZSI6IjI1NjA1MjMxODQ0MzQyOTQ4MzUxODYyMjQyIiwiaXNlclwiOiJhZG1lcyIsInVzZXJuYW1lIjoiZmFsc2UiLCJwYXlsb2FkIjoibmFtZSIsImlhdCI6MTY1NjY5NTE4NCwiZXhwIjoxNjUxMjI0ODQ5fQ.Access_token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyX25hbWUiOiJpZ25pdGUiLCJzY29wZSI6IjI1NjA1MjMxODQ0MzQyOTQ4MzUxODYyMjQyIiwiaXNlclwiOiJhZG1lcyIsInVzZXJuYW1lIjoiZmFsc2UiLCJwYXlsb2FkIjoibmFtZSIsImlhdCI6MTY1NjY5NTE4NCwiZXhwIjoxNjUxMjI0ODQ5fQ.
token_type: bearer
expires_in: 365
id: -1
userType: repo
jti: e0d51295-2010-4ed4-b5ea-982a4e6ae1c5

Database Monitor examples

The following examples show how to use all Database Monitor calls.

Get information about one monitored database instance

```powershell
# Get Monitor Information for a single database
$databaseId = 1
$monitorURL = $baseURL + "databases/$databaseId/monitor-information"
Try {
    $monitorJSON = Invoke-RestMethod -Method Get -Uri $monitorURL -Headers $dpaHeader -TimeoutSec 60
    $monitor = $monitorJSON.data
    $monitor | Format-List
} Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Get Monitor Information for database with id of 1...

(dbId : 1
name : DEV-DPA\SQLEXPRESS
ip : 127.0.0.1
port : 1433
jdbcUrlProperties : applicationIntent=readOnly
connectionProperties :
databaseType : SQL Server
databaseVersion : 12.0.6205.1
databaseEdition : Enterprise Edition: Core-based Licensing (64-bit)
monitoringUser : ignite_next
```
Start and stop monitoring a database instance given its database ID

```powershell
$databaseId = 1
$monitorURL = $baseURL + "databases/$databaseId/monitor-status"

# Start monitoring a database instance given its database ID.
Try {
    Write-Host "Start Monitor for database $databaseId..."
    $command = @{"command" = "START"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
    $result = $monitorJSON.data
    Write-Host "Result: $result"
    Write-Host "Waiting 15 seconds...`r`n"
    Start-Sleep -s 15
}
Catch {
    $_.Exception.ToString()
}

# Stop monitoring a database instance given its database ID.
Try {
    Write-Host "Stop Monitor for database $databaseId..."
    $command = @{"command" = "STOP"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
    $result = $monitorJSON.data
    Write-Host "Result: $result"
    Write-Host "Waiting 15 seconds...`r`n"
```

Start and stop monitoring a database instance given its database ID

```powershell
$databaseId = 1
$monitorURL = $baseURL + "databases/$databaseId/monitor-status"

# Start monitoring a database instance given its database ID.
Try {
    Write-Host "Start Monitor for database $databaseId..."
    $command = @{"command" = "START"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
    $result = $monitorJSON.data
    Write-Host "Result: $result"
    Write-Host "Waiting 15 seconds...`r`n"
    Start-Sleep -s 15
}
Catch {
    $_.Exception.ToString()
}

# Stop monitoring a database instance given its database ID.
Try {
    Write-Host "Stop Monitor for database $databaseId..."
    $command = @{"command" = "STOP"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
    $result = $monitorJSON.data
    Write-Host "Result: $result"
    Write-Host "Waiting 15 seconds...`r`n"
```

Start and stop monitoring a database instance given its database ID
Start-Sleep -s 15
}
Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Start Monitor for database 1...
Result: SUCCESS
Waiting 15 seconds...

Stop Monitor for database 1...
Result: SUCCESS
Waiting 15 seconds...

Get information about all monitored database instances

# Get Monitor Information for all databases
$monitorURL = $baseURL + "databases/monitor-information"
Try {
    Write-Host "Get Monitor Information for all databases..."
    $monitorListJSON = Invoke-RestMethod -Method Get -Uri $monitorURL -Headers $dpaHeader -TimeoutSec 60
    $monitorList = $monitorListJSON.data
    $monitorList | Format-Table -AutoSize
}
Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Get Monitor Information for all databases...

dbId  name              ip      port  jdbcUrlProperties
connectionProperties  databaseType  databaseVersion
----  ------            ---      ----  ---------------------------
----  ----------------  --------  -------  ---------------------------
  1    DEV-DPA\SQLEXPRESS 10.10.10.1 1433  applicationIntent=readOnly
       SQL Server     12.0.6205.1     ...     
  3    DEVORA11_DEVORA11  10.10.10.2  1521
       Oracle         11.2.0.1.0       ...     
 10    DEV-MYSQL:3306    10.10.10.3  3306  dumpQueriesOnException=true
       MySQL          5.7.19          ...     

etc.
Stop and start monitoring for all database instances

```powershell
$monitorURL = $baseURL + "databases/monitor-status"

# Start monitoring all database instances.
Try {
    Write-Host "Starting all Monitors..."
    $command = @{"command" = "START"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
    $result = $monitorJSON.data
    Write-Host "Result: $result"
    Write-Host "Waiting 30 seconds...\r\n"
    Start-Sleep -s 30
}
Catch {
    $_.Exception.ToString()
}

# Stop monitoring all database instances.
Try {
    Write-Host "Stopping all Monitors..."
    $command = @{"command" = "STOP"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
    $result = $monitorJSON.data
    Write-Host "Result: $result"
    Write-Host "Waiting 30 seconds...\r\n"
    Start-Sleep -s 30
}
Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Starting all Monitors...
Result: SUCCESS
Waiting 30 seconds...

Stopping all Monitors...
Result: SUCCESS
Waiting 30 seconds...
```
Update the user password for a monitored database instance

```powershell
$databaseId = 1
$monitorURL = $baseURL + "databases/$databaseId/update-password"
Try {
    Write-Host "Update the Monitor password for database $databaseId..."
    $command = @{"password" = "NewPassword!"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
    $result = $monitorJSON.data
    Write-Host "Result: $result"
} Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Update the Monitor password for database 1...
Result: SUCCESS
```

License Allocation examples

The examples below show how to use all License Allocation calls.

Get information about currently installed licenses

```powershell
$licenseURL = $baseURL + "databases/licenses/installed"
Try {
    Write-Host "Getting Installed license information with total amounts available for use and total amounts used..."
    $licenseListJSON = Invoke-RestMethod -Method Get -Uri $licenseURL -Headers $dpaHeader -TimeoutSec 60
    $licenseList = $licenseListJSON.data
    $licenseList | Format-Table -AutoSize
} Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Getting Installed license information with total amounts available for use and total amounts used...

licenseProduct licenseCategory licensesAvailable licensesConsumed
```
Get license information for a single database instance

```
$databaseId = 1
$licenseURL = $baseURL + "databases/$databaseId/licenses"
Try {
    Write-Host "Getting current license information for the database instance with database ID of $databaseId."
    $licenseListJSON = Invoke-RestMethod -Method Get -Uri $licenseURL -Headers $dpaHeader -TimeoutSec 60
    $licenseInfo = $licenseListJSON.data
    $licenseInfo | Format-Table -AutoSize
} Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Getting current license information for the database instance with database ID of 1.

serverName overLicensed vmLicenseProduct performanceLicenseProduct

---------- ---------- ---------- ---------- ----------
DEV-DPA False DPAVM DPA VM

Update license information for a database instance

```
$databaseId = 1
$licenseURL = $baseURL + "databases/$databaseId/licenses"

# Add a DPACAT2 and a DPAVM license
$licenseAllocation = @{"performanceLicenseProduct" = "DPACAT2";
                        "vmLicenseProduct" = "DPAVM"} | ConvertTo-Json
Try {
    Write-Host "Updating license for database id $databaseId..."
    $licenseResultJSON = Invoke-RestMethod -Method Put -Uri $licenseURL -Body $licenseAllocation -Headers $dpaHeader -TimeoutSec 60
    $licenseResult = $licenseResultJSON.data
    Write-Host "New License Allocation result for the database instance with database ID of $databaseId."
```
$licenseResult | Format-Table -AutoSize

Catch {
    $_.Exception.ToString()
}

# Remove the DPAVM license
$licenseAllocation = @{$"performanceLicenseProduct" = "DPACAT2";
    "vmLicenseProduct" = "REMOVE"} | ConvertTo-Json
Try {
    Write-Host "Updating license for database id $databaseId..."
    $licenseResultJSON = Invoke-RestMethod -Method Put -Uri $licenseURL -Body $licenseAllocation -Headers $dpaHeader -TimeoutSec 60
    $licenseResult = $licenseResultJSON.data
    Write-Host "New License Allocation result for the database instance with database ID of $databaseId."
    $licenseResult | Format-Table -AutoSize
}
Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Updating license for database id 1...
New License Allocation result for the database instance with database ID of 1.

serverName overLicensed vmLicenseProduct performanceLicenseProduct

------------- ------------------ -------------------
DEV-DPA False DPAVM DPACAT2

Updating license for database id 1...
New License Allocation result for the database instance with database ID of 1.

serverName overLicensed vmLicenseProduct performanceLicenseProduct

------------- ------------------ -------------------
DEV-DPA False DPACAT2

Annotation examples
The examples below show how to use all Annotation calls.

Get a list of annotations for the last 30 days

```ps1
$databaseId = 1
$annotationURL = $baseURL + "databases/$databaseId/annotations"
```
# Dates are in ISO 8601 format (2018-12-31T12:00:00.000-07:00)
$endTime = Get-Date
$startTime = $endTime.AddDays(-30)
$startTime = $startTime.ToString("yyyy-MM-ddTHH:mm:ss.fffzzz")
$endTime = $endTime.ToString("yyyy-MM-ddTHH:mm:ss.fffzzz")

$request = [System.UriBuilder]$annotationURL
$request.Query = "startTime=$startTime&endTime=$endTime"
$annotationURL = $request.Uri

Try {
    Write-Host "Getting Annotations for the last 30 days..."
    $annotationListJSON = Invoke-RestMethod -Method Get -Uri $annotationURL -Headers $dpaHeader -TimeoutSec 60
    $annotationList = $annotationListJSON.data
    $annotationList | Format-Table -AutoSize
}
Catch {
    handleError $Error[0]
}

# This will print out data like this:
Getting Annotations for the last 30 days...

<table>
<thead>
<tr>
<th>id</th>
<th>title</th>
<th>description</th>
<th>createdBy</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>98</td>
<td>Test Title API Test Event created by DPA API DPA API 2018-12-04T18:13:04-07:00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>99</td>
<td>Test Title API Test Event created by DPA API DPA API 2018-12-04T18:14:27-07:00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>Test Title API Test Event created by DPA API DPA API 2018-12-04T18:16:46-07:00</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Create a new annotation

$databaseId = 1
$annotationURL = $baseUrl + "databases/$databaseId/annotations"
$createTime = Get-Date
# Dates are in ISO 8601 format, no millis (2018-12-31T12:00:00-07:00)
$createTime = $createTime.ToString("yyyy-MM-ddTHH:mm:ss.fffzzz")
$body = @{"title" = "API Test Title";}
"description" = "API Test Description";
"createdBy" = "Test API User";
"time" = "$createTime"
| ConvertTo-Json

Try {
    Write-Host "Creating Annotation..."
    $dpaResponseJSON = Invoke-RestMethod -Uri $annotationURL -Body $body -Method POST -
    Headers $dpaHeader -TimeoutSec 60
    $dpaResponse = $dpaResponseJSON.data
    $dpaResponse | Format-Table -AutoSize
    $annotationId = $dpaResponse.id
}
Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Creating Annotation...

<table>
<thead>
<tr>
<th>id</th>
<th>title</th>
<th>description</th>
<th>createdBy</th>
<th>time</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>---</td>
<td>---------------</td>
<td>------------------</td>
<td>-----------</td>
<td>-----------------</td>
<td>------</td>
</tr>
<tr>
<td>148</td>
<td>API Test Title</td>
<td>API Test Description</td>
<td>Test API User</td>
<td>2019-01-03T15:20:36-07:00</td>
<td>API</td>
</tr>
</tbody>
</table>

Delete an annotation

$databaseId = 1
$annotationId = 148
$annotationURL = $baseURL + "databases/$databaseId/annotations/$annotationId"
Try {
    Write-Host "Deleting Annotation with id of $annotationId..."
    $dpaResponseJSON = Invoke-RestMethod -Uri $annotationURL -Method DELETE -Headers $dpaHeader -TimeoutSec 60
    Write-Host "Annotation with id of $annotationId deleted"
}
Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Deleting Annotation with id of 148...
Annotation with id of 148 deleted

Database Registration examples

The examples below show how to use all Database Registration calls.
Register and unregister a SQL Server database instance for monitoring

This example registers a new SQL Server database instance, waits 60 seconds, and then unregisters the database instance.

```powershell
# Register a SQL Server database instance for monitoring.
$registrationURL = $baseURL + "databases/register-monitor"
$body = @{
    "databaseType" = "SQLSERVER";
    "serverName" = "127.0.0.1";
    "port" = "1433";
    "sysAdminUser" = "sa";
    "sysAdminPassword" = "Password";
    "monitoringUser" = "dpa_test_m";
    "monitoringUserPassword" = "Password";
    "monitoringUserIsNew" = $true;
    "displayName" = "DPA_SQL2K12";
} | ConvertTo-Json

Try {
    Write-Host "Registering Database..."
    $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -Headers $dpaHeader -TimeoutSec 60
    $dpaResponse = $dpaResponseJSON.data
    $newDbId = $dpaResponse.databaseId
}
Catch {
    $_.Exception.ToString()
}

Write-Host "Waiting 60 seconds...`r`n"
Start-Sleep -s 60

# Un-register the SQL Server database instance.
if ($newDbId) {
    $registrationURL = $baseURL + "databases/unregister-monitor"
    $body = @{
        "databaseId" = $newDbId;
        "removeMonitoringUser" = $true;
        "removeDatabaseObjects" = $true;
        "sysAdminUser" = "sa";
        "sysAdminPassword" = "Password";
    } | ConvertTo-Json

    Try {
        Write-Host "Registering Database..."
        $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -Headers $dpaHeader -TimeoutSec 60
    } Catch {
        $_.Exception.ToString()
    }
}```
# Register an Oracle database instance for monitoring.

This example registers a new Oracle database instance, waits 60 seconds, and then unregisters the database instance.

```
$registrationURL = $baseURL + "databases/register-monitor"
$body = @("databaseType" = "ORACLE",
        "serverName" = "127.0.0.1",
        "serviceNameOrSID" = "DPA_ORA11R1",
        "port" = "1521",
        "sysAdminUser" = "system",
        "sysAdminPassword" = "Password",
        "sysPassword" = "Password",
        "monitoringUser" = "dpa_test_m",
        "monitoringUserPassword" = "Password",
        "monitoringUserIsNew" = $true,
        "monitoringUserTableSpace" = "users",
        "monitoringUserTempTableSpace" = "temp",
        "oracleEBusinessEnabled" = $false"
)
```
Try {
    Write-Host "Registering Database..."
    $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -
    Headers $dpaHeader
    $dpaResponse = $dpaResponseJSON.data
    $dpaResponse | Format-Table -AutoSize
    $newDbId = $dpaResponse.databaseId
}
Catch {
    $_.Exception.ToString()
}

Write-Host "Waiting 60 seconds...`r`n"
Start-Sleep -s 60

# Un-register the Oracle database instance.
if ($newDbId) {
    $registrationURL = $baseURL + "databases/unregister-monitor"
    $body = @{"databaseId" = $newDbId;
    "removeMonitoringUser" = $true;
    "removeDatabaseObjects" = $true;
    "sysAdminUser" = "system";
    "sysAdminPassword" = "Password"} | ConvertTo-Json
    Try {
        Write-Host "Unregistering Database..."
        $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -
        Headers $dpaHeader -TimeoutSec 60
        $dpaResponse = $dpaResponseJSON.data
        $dpaResponse | Format-Table -AutoSize
    }
    Catch {
        $_.Exception.ToString()
    }
}

# This will print out data like this:
Registering Database...
databaseId result
-------------------
71 SUCCESS
Waiting 60 seconds...

Unregistering Database...

databaseId result
-------- ------
71 SUCCESS

Register and unregister a MySQL database instance for monitoring

This example registers a new MySQL database instance, waits 60 seconds, and then unregisters the database instance.

```powershell
# Register a MySQL database instance for monitoring.
$registrationURL = $baseURL + "databases/register-monitor"
$body = @{"databaseType" = "MYSQL",
    "serverName" = "127.0.0.1",
    "port" = "3306",
    "sysAdminUser" = "root",
    "sysAdminPassword" = "Password",
    "monitoringUser" = "dpa_test_m",
    "monitoringUserPassword" = "Password",
    "monitoringUserIsNew" = $true,
    "displayName" = "DPA_MYSQL56"} | ConvertTo-Json

Try {
    Write-Host "Registering Database..."
    $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -Headers $dpaHeader -TimeoutSec 60
    $dpaResponse = $dpaResponseJSON.data
    $dpaResponse | Format-Table -AutoSize
    $newDbId = $dpaResponse.databaseId
}

Catch {
    $_.Exception.ToString()
}

Write-Host "Waiting 60 seconds...\r\n"
Start-Sleep -s 60

# Un-register the MySQL database instance.
if ($newDbId) {
```
$registrationURL = $baseURL + "databases/unregister-monitor"
$body = @{"databaseId" = $newDbId;
   "removeMonitoringUser" = $true;
   "removeDatabaseObjects" = $true;
   "sysAdminUser" = "root";
   "sysAdminPassword" = "Password"} | ConvertTo-Json

Try {
    Write-Host "Registering Database..."
    $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -Headers $dpaHeader -TimeoutSec 60
    $dpaResponse = $dpaResponseJSON.data
    $dpaResponse | Format-Table -AutoSize
}
Catch {
    $_.Exception.ToString()
}

# This will print out data like this:
Registering Database...
databaseId result
----------- ------
72 SUCCESS

Waiting 60 seconds...

Unregistering Database...
databaseId result
----------- ------
72 SUCCESS

**Full working script**

The following script combines all of the examples shown above into a script that can be run.

```
#-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
# Examples:
# - Get an access token
# - Database Monitor Examples
# - Get Monitor Information for a single database
# - Start or Stop monitoring a database instance given its database ID
# - Get Monitor Information for all databases
# - Start monitoring for all database instances
```
# - Stop monitoring for all database instances
# - ERROR: Get Monitor Information for a database that doesn't exist
# - ERROR: Start a database that doesn't exist
# - Licensing Examples
# - Get the currently installed license information
# - Get License Information for a single database
# - Update License Information for a single database
# - Annotation Examples
# - Gets a List of annotations for the last 30 days
# - Create a new annotation
# - Delete an annotation
# - Registration Examples
# - Register a MySQL database instance for monitoring
# - Un-register the MySQL database instance
#-----------------------------------------------------------------------------------

# Configure the variables below for the DPA Host
#-----------------------------------------------------------------------------------
$baseUrl = "https://localhost:8124/iwc/api/
$refreshToken = "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJNeVRva2VuIiwiaXN..." $databaseId = 1
# Nothing to configure below this line
#-----------------------------------------------------------------------------------

# Function to parse the Response Data from DPA and print
# out the error information
#-----------------------------------------------------------------------------------
Function handleError ($thisError) {
    Write-Host "---------------------------------------------" -ForegroundColor Red
    Write-Host "Caught Exception at line:" $_.InvocationInfo.ScriptLineNumber -ForegroundColor Red
    if ($_.Exception.Response) {
        $errResp = $StreamReader.ReadToEnd()
        $StreamReader.Close()
    }
    if ($errResp) {
        # This will format the JSON
        $errResp = $errResp | ConvertFrom-Json | ConvertTo-Json -Depth 100
        Write-Host $thisError.Exception.Message -ForegroundColor Red
Write-Host "Response:
$errResp" -ForegroundColor Red
}
else {
  Write-Host $_.Exception.ToString() -ForegroundColor Red
}
Write-Host "-----------------------------------------------" -ForegroundColor Red

# Adding certificate exception to prevent API errors
# Uncomment this if you are getting trust errors and would
# like to run with self-signed certificates.
#-----------------------------------------------
# add-type @"
# using System.Net;
# public class TrustAllCertsPolicy : ICertificatePolicy {
#   public bool CheckValidationResult(
#     ServicePoint srvPoint, X509Certificate certificate,
#     WebRequest request, int certificateProblem) {
#     return true;
#   }
# }
# "@

# Get an access token
#-----------------------------------------------
$authTokenURL = $baseURL + "security/oauth/token"
$body = @{"grant_type" = "refresh_token"
  "refresh_token" = "$refreshToken"}
Try {
  Write-Host "Getting Access Token..."
  $dpaAuthResponse = Invoke-RestMethod -Uri $authTokenURL -Method POST -Body $body
  $dpaAuthResponse | Format-List
}
Catch {
  handleError $Error[0]
  Write-Host 'Error getting authentication token, cannot continue' -ForegroundColor Red
  return
}

# If successful we will create our headers to be used for all API calls
$tokenType = $dpaAuthResponse.token_type
$accessToken = $dpaAuthResponse.access_token
$dpaHeader = @{}
$dpaHeader.Add("Accept", "application/json")
$dpaHeader.Add("Content-Type", "application/json; charset=UTF-8")
$dpaHeader.Add("Authorization", "$tokenType $accessToken")

# Database Monitor Examples

# Get Monitor Information for a single database
$monitorURL = $baseURL + "databases/$databaseId/monitor-information"
Try {
    Write-Host "Get Monitor Information for database with id of $databaseId..."
    $monitorJSON = Invoke-RestMethod -Method Get -Uri $monitorURL -Headers $dpaHeader -TimeoutSec 60
    $monitor = $monitorJSON.data
    $monitor | Format-Table -AutoSize
} Catch {
    handleError $Error[0]
}

# Start or Stop monitoring a database instance given its database ID.
# If it is already running stop it and then restart it
# If it is not running start it and then stop it
$monitorURL = $baseURL + "databases/$databaseId/monitor-status"
if ($monitor.monitorState -eq "Monitor Running") {
    $changeCommand = "STOP"
    $revertCommand = "START"
}
elseif ($monitor.monitorState -eq "Monitor Stopped") {
    $changeCommand = "START"
    $revertCommand = "STOP"
}
Try {
    Write-Host "$changeCommand Monitor for database $databaseId..."
    $command = @{"command" = $changeCommand} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
    $result = $monitorJSON.data
    Write-Host "Result: $result"
    Write-Host "Waiting 15 seconds...\r\n"
    Start-Sleep -s 15
Write-Host "$revertCommand Monitor for database $databaseId..."
$command = @{"command" = $revertCommand} | ConvertTo-Json
$monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
$result = $monitorJSON.data
Write-Host "Result: $result"
Write-Host "Waiting 15 seconds...``
Start-Sleep -s 15
} Catch {
    handleError $Error[0]
}

# Get Monitor Information for all databases
$monitorURL = $baseURL + "databases/monitor-information"
Try {
    Write-Host "Get Monitor Information for all databases..."
    $monitorListJSON = Invoke-RestMethod -Method Get -Uri $monitorURL -Headers $dpaHeader -TimeoutSec 60
    $monitorList = $monitorListJSON.data
    $monitorList | Format-Table -AutoSize

    # Keep a list of running or started monitors to be used later
    $runningIds = @()
    foreach ($monitor in $monitorList) {
        if ($monitor.monitorState -eq "Monitor Running" -or
            $monitor.monitorState -eq "Monitor Start No License" -or
            $monitor.monitorState -like '*Start*')
        {
            $runningIds += $monitor.dbId
        }
    }
    Write-Host "Running Monitors: $runningIds``
} Catch {
    handleError $Error[0]
}

# Start monitoring all database instances.
$monitorURL = $baseURL + "databases/monitor-status"
Try {
    Write-Host "Starting all Monitors..."
    $command = @{"command" = "START"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
$dpaHeader -TimeoutSec 60
$result = $monitorJSON.data
Write-Host "Result: $result"
Write-Host "Waiting 30 seconds...`r`n"
Start-Sleep -s 30
}
Catch {
    handleError $Error[0]
}

# Stop monitoring all database instances.
Try {
    Write-Host "Stopping all Monitors..."
    $command = @{"command" = "STOP"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
    $result = $monitorJSON.data
    Write-Host "Result: $result"
    Write-Host "Waiting 30 seconds...`r`n"
    Start-Sleep -s 30
}
Catch {
    handleError $Error[0]
}

# Try to put it back the way we found it by restarting the ones that were running
$command = @{"command" = "START"} | ConvertTo-Json
foreach ($dbId in $runningIds) {
    Try {
        $monitorURL = $baseURL + "databases/$dbId/monitor-status"
        Write-Host "Starting Monitor for database $dbId..."
        $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
        $result = $monitorJSON.data
        Write-Host "Result: $result`r`n"
    }
    Catch {
        handleError $Error[0]
    }
}

# Update the monitor database user password (Un-comment to use)
# $monitorURL = $baseURL + "databases/$databaseId/update-password"
# Try {
#    Write-Host "Update the Monitor password for database $databaseId..."
# $command = @{"password" = "NewPassword!"} | ConvertTo-Json
# $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
# $result = $monitorJSON.data
# Write-Host "Result: $result"
#
# Catch {
#     handleError $Error[0]
# }

# Try to cause some errors...

# Get Monitor Information for a database that doesn't exist
$monitorURL = $baseURL + "databases/-1/monitor-information"
Try {
    Write-Host "Get Monitor Information for invalid database..."
    $monitorJSON = Invoke-RestMethod -Method Get -Uri $monitorURL -Headers $dpaHeader -TimeoutSec 60
} Catch {
    handleError $Error[0]
}

# Start a database that doesn't exist
$monitorURL = $baseURL + "databases/-1/monitor-status"
Try {
    Write-Host "START Monitor for invalid database..."
    $command = @{"command" = "START"} | ConvertTo-Json
    $monitorJSON = Invoke-RestMethod -Method Put -Uri $monitorURL -Body $command -Headers $dpaHeader -TimeoutSec 60
} Catch {
    handleError $Error[0]
}

#---------------------------------------------
# Licensing Examples
#---------------------------------------------

# Get the currently installed license information
$licenseURL = $baseURL + "databases/licenses/installed"
Try {
    Write-Host "Getting Installed license information with total amounts available for use and total amounts used..."
$licenseListJSON = Invoke-RestMethod -Method Get -Uri $licenseURL -Headers $dpaHeader -TimeoutSec 60
$licenseList = $licenseListJSON.data
$licenseList | Format-Table -AutoSize
}
Catch {
    handleError $Error[0]
}

# Get License Information for a single database
$licenseURL = $baseURL + "databases/$databaseId/licenses"
Try {
    Write-Host "Getting current license information for the database instance with database ID of $databaseId."
    $licenseListJSON = Invoke-RestMethod -Method Get -Uri $licenseURL -Headers $dpaHeader -TimeoutSec 60
    $licenseInfo = $licenseListJSON.data
    $licenseInfo | Format-Table -AutoSize
}
Catch {
    handleError $Error[0]
}

# This will Update License Information for a single database setting the Performance License and the VM License to what it currently is.
# It should succeed but it should make no changes.
$dbProduct = $licenseInfo.performanceLicenseProduct
$vmProduct = $licenseInfo.vmLicenseProduct
$licenseAllocation = @{"performanceLicenseProduct" = $dbProduct;
                          "vmLicenseProduct" = $vmProduct} | ConvertTo-Json
Try {
    Write-Host "Updating license for database id $databaseId..."
    $licenseResultJSON = Invoke-RestMethod -Method Put -Uri $licenseURL -Body $licenseAllocation -Headers $dpaHeader -TimeoutSec 60
    $licenseResult = $licenseResultJSON.data
    Write-Host "New License Allocation result for the database instance with database ID of $databaseId."
    $licenseResult | Format-Table -AutoSize
}
Catch {
    handleError $Error[0]
}

#-----------------------------------------------
# Annotation Examples
# Gets a List of annotations for the last 30 days
$annotationURL = $baseURL + "databases/$databaseId/annotations"

# Dates are in ISO 8601 format (2018-12-31T12:00:00-07:00)
$endTime = Get-Date
$startTime = $endTime.AddDays(-30)
$startTime = $startTime.ToString("yyyy-MM-ddTHH:mm:ss.fffzzz")
$endTime = $endTime.ToString("yyyy-MM-ddTHH:mm:ss.ffzzz")

$request = [System.UriBuilder]$annotationURL
$request.Query = "startTime=$startTimeZZZendTime=$endTime"
$annotationURL = $request.Uri

Try {
    Write-Host "Getting Annotations for the last 30 days..."
    $annotationListJSON = Invoke-RestMethod -Method Get -Uri $annotationURL -Headers $dpaHeader -TimeoutSec 60
    $annotationList = $annotationListJSON.data
    $annotationList | Format-Table -AutoSize
} Catch {
    handleError $Error[0]
}

# Create a new annotation
# Dates are in ISO 8601 format, no millis (2018-12-31T12:00:00-07:00)
$annotationURL = $baseURL + "databases/$databaseId/annotations"
$createTime = Get-Date
$createTime = $createTime.ToString("yyyy-MM-ddTHH:mm:sszzz")
$body = @{"title" = "API Test Title";
            "description" = "API Test Description";
            "createdBy" = "Test API User";
            "time" = "$createTime"} | ConvertTo-Json

Try {
    Write-Host "Creating Annotation..."
    $dpaResponseJSON = Invoke-RestMethod -Uri $annotationURL -Body $body -Method POST -Headers $dpaHeader -TimeoutSec 60
    $dpaResponse = $dpaResponseJSON.data
    $dpaResponse | Format-Table -AutoSize
    $annotationId = $dpaResponse.id
} Catch {
    handleError $Error[0]
Delete an annotation if ($annotationId) {
    $annotationURL = $baseURL + "databases/$databaseId/annotations/$annotationId"
    Try {
        Write-Host "Deleting Annotation with id of $annotationID..."
        $dpaResponseJSON = Invoke-RestMethod -Uri $annotationURL -Method DELETE -Headers $dpaHeader -TimeoutSec 60
        Write-Host "Annotation with id of $annotationID deleted\r\n"
    }
    Catch {
        handleError $Error[0]
    }
}

# Registration Examples

# Register a SQL Server database instance for monitoring.
$registrationURL = $baseURL + "databases/register-monitor"
$body = @{"databaseType" = "SQLSERVER";
    "serverName" = "127.0.0.1";
    "port" = "1433";
    "sysAdminUser" = "sa";
    "sysAdminPassword" = "Password";
    "monitoringUser" = "dpa_test_m";
    "monitoringUserPassword" = "Password";
    "monitoringUserIsNew" = $true;
    "displayName" = "DPA_SQL2K12"} | ConvertTo-Json
Try {
    Write-Host "Registering Database..."
    $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -Headers $dpaHeader -TimeoutSec 60
    $dpaResponse = $dpaResponseJSON.data
    $dpaResponse | Format-Table -AutoSize
    $newDbId = $dpaResponse.databaseId
}
Catch {
    handleError $Error[0]
}
Write-Host "Waiting 60 seconds...`r`n"
Start-Sleep -s 60

# Un-register the SQL Server database instance.
#-----------------------------------------------
if ($newDbId) {
    $registrationURL = $baseURL + "databases/unregister-monitor"
    $body = @{"databaseId" = $newDbId;
                "removeMonitoringUser" = $true;
                "removeDatabaseObjects" = $true;
                "sysAdminUser" = "sa";
                "sysAdminPassword" = "Password"} | ConvertTo-Json
    Try {
        Write-Host "Unregistering Database..."
        $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST
        $dpaResponse = $dpaResponseJSON.data
        $dpaResponse | Format-Table -AutoSize
    } Catch {
        handleError $Error[0]
    }
}

# Register an Oracle database instance for monitoring.
#------------------------------------------------------------
$registrationURL = $baseURL + "databases/register-monitor"
$body = @{"databaseType" = "ORACLE";
          "serverName" = "127.0.0.1";
          "serviceNameOrSID" = "DPA_ORA11R1";
          "port" = "1521";
          "sysAdminUser" = "system";
          "sysAdminPassword" = "Password";
          "sysPassword" = "Password";
          "monitoringUser" = "dpa_test_m";
          "monitoringUserPassword" = "Password";
          "monitoringUserIsNew" = $true;
          "monitoringUserTableSpace" = "USERS";
          "monitoringUserTempTableSpace" = "TEMP";
          "oracleEBusinessEnabled" = $false;
          "displayName" = "DPA_ORA11R1"} | ConvertTo-Json
Try {
    Write-Host "Registering Database..."
$dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -Headers $dpaHeader
$dpaResponse = $dpaResponseJSON.data
$dpaResponse | Format-Table -AutoSize
$newDbId = $dpaResponse.databaseId

Catch {
    handleError $Error[0]
}

Write-Host "Waiting 60 seconds...`r`n"
Start-Sleep -s 60

#----------------------------------------------
# Un-register the Oracle database instance.
#----------------------------------------------
#if ($newDbId) {
    $registrationURL = $baseURL + "databases/unregister-monitor"
    $body = @{"databaseId" = $newDbId;
             "removeMonitoringUser" = $true;
             "removeDatabaseObjects" = $true;
             "sysAdminUser" = "system";
             "sysAdminPassword" = "Password"} | ConvertTo-Json

    Try {
        Write-Host "Unregistering Database..."
        $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -Headers $dpaHeader -TimeoutSec 60
        $dpaResponse = $dpaResponseJSON.data
        $dpaResponse | Format-Table -AutoSize
    }
    Catch {
        handleError $Error[0]
    }
} #-$

#----------------------------------------------
# Register a MySQL database instance for monitoring.
#----------------------------------------------
$registrationURL = $baseURL + "databases/register-monitor"
$body = @{"databaseType" = "MYSQL";
           "monitoringUser" = "dpa_test_m";
           "port" = "3306";
           "sysAdminPassword" = "Password";
           "sysAdminUser" = "root";
           "serverName" = "127.0.0.1";
           "databaseType" = "MYSQL";"
"monitoringUserPassword" = "Password";
"monitoringUserIsNew" = $true;
"displayName" = "DPA_MYSQL56";} | ConvertTo-Json

Try {
    Write-Host "Registering Database..."
    $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -
    Headers $dpaHeader -TimeoutSec 60
    $dpaResponse = $dpaResponseJSON.data
    $newDbId = $dpaResponse.databaseId
} Catch {
    handleError $Error[0]
}

Write-Host "Waiting 60 seconds...`r`n"
Start-Sleep -s 60

#---------------------------------------------------------------
# Un-register the MySQL database instance.
#---------------------------------------------------------------
if ($newDbId) {
    $registrationURL = $baseURL + "databases/unregister-monitor"
    $body = @{"databaseId" = $newDbId;
               "removeMonitoringUser" = $true;
               "removeDatabaseObjects" = $true;
               "sysAdminUser" = "root";
               "sysAdminPassword" = "Password"} | ConvertTo-Json

    Try {
        Write-Host "Unregistering Database..."
        $dpaResponseJSON = Invoke-RestMethod -Uri $registrationURL -Body $body -Method POST -
        Headers $dpaHeader -TimeoutSec 60
        $dpaResponse = $dpaResponseJSON.data
        $dpaResponse | Format-Table -AutoSize
    } Catch {
        handleError $Error[0]
    }
}

# End of script
DPA administrative tasks

See the following topics for information about administrative tasks in DPA:

- Stop and start DPA
- Set advanced options
- Enable SNMP monitoring in SCOM

Stop and start DPA

Stop and start DPA on a Windows server

Using the Windows Control Panel, stop the Ignite PI Server service. To restart DPA, start the service again.

Stop and start DPA on a Linux server

To stop DPA, run the following command from the DPA directory:

```
shutdown.sh
```

To restart DPA, run the following command from the DPA directory:

```
startup.sh
```

Set advanced DPA configuration options

You can use advanced options to change DPA's default behavior. For example, you can change how long DPA stores data about a database instance, or prevent DPA from annotating wait time charts when a PDB is moved.

1. In the DPA menu bar, click Options.
2. Under Administration > Configuration, click Advanced Options.
   - The System Options tab lists options that apply to all database instances.
3. If you are setting an option that applies to a single database instance, click DB Instance Options and select the database instance.
4. Click the name of an option to open the Edit Option dialog.
5. To change an option value, enter the New Value and click Update.

Enable SNMP Monitoring in SCOM

You can set up SolarWinds DPA to use SNMP to monitor System Center Operations Manager (SCOM).
1. In the DPA menu bar, click Options.
2. Under Administration > Users & Contacts, click Contact Management.
3. Click Create SNMP Contact.
4. Enter the SCOM host IP address and port in the Trap Receiver fields. The default port is 162.
5. Enter the community string that was set up on the SNMP Service on the SCOM host.
   
   **Tip:** This string is case sensitive.

6. On the SolarWinds DPA server, make sure the SNMP service is running and the community string set matches the string you entered in the SNMP Contact window.