**Monitoring Amazon RDS**

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon RDS and your AWS solutions. You should collect monitoring data from all of the parts of your AWS solution so that you can more easily debug a multi-point failure if one occurs. Before you start monitoring Amazon RDS, we recommend that you create a monitoring plan that includes answers to the following questions:

* What are your monitoring goals?
* What resources will you monitor?
* How often will you monitor these resources?
* What monitoring tools will you use?
* Who will perform the monitoring tasks?
* Who should be notified when something goes wrong?

The next step is to establish a baseline for normal Amazon RDS performance in your environment, by measuring performance at various times and under different load conditions. As you monitor Amazon RDS, you should consider storing historical monitoring data. This stored data will give you a baseline to compare against with current performance data, identify normal performance patterns and performance anomalies, and devise methods to address issues.

For example, with Amazon RDS, you can monitor network throughput, I/O for read, write, and/or metadata operations, client connections, and burst credit balances for your DB instances. When performance falls outside your established baseline, you might need change the instance class of your DB instance or the number of DB instances and Read Replicas that are available for clients in order to optimize your database availability for your workload.

In general, acceptable values for performance metrics depend on what your baseline looks like and what your application is doing. Investigate consistent or trending variances from your baseline. Advice about specific types of metrics follows:

* **High CPU or RAM consumption** – High values for CPU or RAM consumption might be appropriate, provided that they are in keeping with your goals for your application (like throughput or concurrency) and are expected.
* **Disk space consumption** – Investigate disk space consumption if space used is consistently at or above 85 percent of the total disk space. See if it is possible to delete data from the instance or archive data to a different system to free up space.
* **Network traffic** – For network traffic, talk with your system administrator to understand what expected throughput is for your domain network and Internet connection. Investigate network traffic if throughput is consistently lower than expected.
* **Database connections** – Consider constraining database connections if you see high numbers of user connections in conjunction with decreases in instance performance and response time. The best number of user connections for your DB instance will vary based on your instance class and the complexity of the operations being performed. You can determine the number of database connections by associating your DB instance with a parameter group where the User Connectionsparameter is set to a value other than 0 (unlimited). You can either use an existing parameter group or create a new one. For more information, see [Working with DB Parameter Groups](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_WorkingWithParamGroups.html).
* **IOPS metrics** – The expected values for IOPS metrics depend on disk specification and server configuration, so use your baseline to know what is typical. Investigate if values are consistently different than your baseline. For best IOPS performance, make sure your typical working set will fit into memory to minimize read and write operations.

**Monitoring Tools**

AWS provides various tools that you can use to monitor Amazon RDS. You can configure some of these tools to do the monitoring for you, while some of the tools require manual intervention. We recommend that you automate monitoring tasks as much as possible.

**Automated Monitoring Tools**

You can use the following automated monitoring tools to watch Amazon RDS and report when something is wrong:

* **Amazon RDS Events** – Subscribe to Amazon RDS events to be notified when changes occur with a DB instance, DB cluster, DB snapshot, DB cluster snapshot, DB parameter group, or DB security group. For more information, see [Using Amazon RDS Event Notification](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_Events.html).
* **Database log files** – View, download, or watch database log files using the Amazon RDS console or Amazon RDS API actions. You can also query some database log files that are loaded into database tables. For more information, see [Amazon RDS Database Log Files](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_LogAccess.html).
* **Amazon RDS Enhanced Monitoring** — Look at metrics in real time for the operating system that your DB instance or DB cluster runs on. For more information, see [Enhanced Monitoring](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_Monitoring.OS.html).

In addition, Amazon RDS integrates with Amazon CloudWatch for additional monitoring capabilities:

* **Amazon CloudWatch Metrics** – Amazon RDS automatically sends metrics to CloudWatch every minute for each active database instance and cluster. You are not charged additionally for Amazon RDS metrics in CloudWatch. For more information, see [Viewing DB Instance Metrics](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Monitoring.html#USER_Monitoring).
* **Amazon CloudWatch Alarms** – You can watch a single Amazon RDS metric over a specific time period, and perform one or more actions based on the value of the metric relative to a threshold you set. For more information, see [Monitoring with Amazon CloudWatch](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Monitoring.html#monitoring-cloudwatch)
* **Amazon CloudWatch Logs** – MariaDB, MySQL, and Aurora MySQL enable you to monitor, store, and access your database log files in CloudWatch Logs. For more information, see [Amazon CloudWatch Logs User Guide](http://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/)

**Manual Monitoring Tools**

Another important part of monitoring Amazon RDS involves manually monitoring those items that the CloudWatch alarms don't cover. The Amazon RDS, CloudWatch, AWS Trusted Advisor and other AWS console dashboards provide an at-a-glance view of the state of your AWS environment. We recommend that you also check the log files on your DB instance.

* From the Amazon RDS console, you can monitor the following items for your resources:
  + The number of connections to a DB instance
  + The amount of read and write operations to a DB instance
  + The amount of storage that a DB instance is currently utilizing
  + The amount of memory and CPU being utilized for a DB instance
  + The amount of network traffic to and from a DB instance
* From the AWS Trusted Advisor dashboard, you can review the following cost optimization, security, fault tolerance, and performance improvement checks:
  + Amazon RDS Idle DB Instances
  + Amazon RDS Security Group Access Risk
  + Amazon RDS Backups
  + Amazon RDS Multi-AZ
  + Amazon Aurora DB Instance Accessibility

For more information on these checks, see [Trusted Advisor Best Practices (Checks)](https://aws.amazon.com/premiumsupport/trustedadvisor/best-practices/).

* CloudWatch home page shows:
  + Current alarms and status
  + Graphs of alarms and resources
  + Service health status

In addition, you can use CloudWatch to do the following:

* + Create [customized dashboards](http://docs.aws.amazon.com/AmazonCloudWatch/latest/DeveloperGuide/CloudWatch_Dashboards.html) to monitor the services you care about
  + Graph metric data to troubleshoot issues and discover trends
  + Search and browse all your AWS resource metrics
  + Create and edit alarms to be notified of problems

**Monitoring with Amazon CloudWatch**

You can monitor DB instances using Amazon CloudWatch, which collects and processes raw data from Amazon RDS into readable, near real-time metrics. These statistics are recorded for a period of two weeks, so that you can access historical information and gain a better perspective on how your web application or service is performing. By default, Amazon RDS metric data is automatically sent to CloudWatch in 1-minute periods. For more information about CloudWatch, see [What Are Amazon CloudWatch, Amazon CloudWatch Events, and Amazon CloudWatch Logs?](http://docs.aws.amazon.com/AmazonCloudWatch/latest/DeveloperGuide/WhatIsCloudWatch.html) in the *Amazon CloudWatch User Guide*.

**Amazon RDS Metrics and Dimensions**

When you use Amazon RDS resources, Amazon RDS sends metrics and dimensions to Amazon CloudWatch every minute. You can use the following procedures to view the metrics for Amazon RDS.

**To view metrics using the Amazon CloudWatch console**

Metrics are grouped first by the service namespace, and then by the various dimension combinations within each namespace.

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. If necessary, change the region. From the navigation bar, select the region where your AWS resources reside. For more information, see [Regions and Endpoints](http://docs.aws.amazon.com/general/latest/gr/rande.html).
3. In the navigation pane, choose **Metrics**. Choose the **RDS** metric namespace.


                        Choose metric namespace
                    

1. Select a metric dimension, for example, **By Database Class**.
2. To sort the metrics, use the column heading. To graph a metric, select the check box next to the metric. To filter by resource, choose the resource ID and then choose **Add to search**. To filter by metric, choose the metric name and then choose **Add to search**.


                          Filter metrics
                      

**To view metrics using the AWS CLI**

* At a command prompt, use the following command:

**aws cloudwatch list-metrics --namespace AWS/RDS**

***Amazon RDS Metrics***

The AWS/RDS namespace includes the following metrics.

|  |  |
| --- | --- |
| **Metric** | **Description** |
| BinLogDiskUsage | The amount of disk space occupied by binary logs on the master. Applies to MySQL read replicas.  Units: Bytes |
| BurstBalance | The percent of General Purpose SSD (gp2) burst-bucket I/O credits available.  Units: Percent |
| CPUUtilization | The percentage of CPU utilization.  Units: Percent |
| CPUCreditUsage | [T2 instances] The number of CPU credits spent by the instance for CPU utilization. One CPU credit equals one vCPU running at 100% utilization for one minute or an equivalent combination of vCPUs, utilization, and time (for example, one vCPU running at 50% utilization for two minutes or two vCPUs running at 25% utilization for two minutes).  CPU credit metrics are available at a five-minute frequency only. If you specify a period greater than five minutes, use the Sum statistic instead of the Average statistic.  Units: Credits (vCPU-minutes) |
| CPUCreditBalance | [T2 instances] The number of earned CPU credits that an instance has accrued since it was launched or started. For T2 Standard, theCPUCreditBalance also includes the number of launch credits that have been accrued.  Credits are accrued in the credit balance after they are earned, and removed from the credit balance when they are spent. The credit balance has a maximum limit, determined by the instance size. Once the limit is reached, any new credits that are earned are discarded. For T2 Standard, launch credits do not count towards the limit.  The credits in the CPUCreditBalance are available for the instance to spend to burst beyond its baseline CPU utilization.  When an instance is running, credits in the CPUCreditBalance do not expire. When the instance stops, the CPUCreditBalance does not persist, and all accrued credits are lost.  CPU credit metrics are available at a five-minute frequency only.  Units: Credits (vCPU-minutes) |
| **DatabaseConnections** | **The number of database connections in use.**  **Units: Count** |
| **DiskQueueDepth** | **The number of outstanding IOs (read/write requests) waiting to access the disk.**  **Units: Count** |
| FreeableMemory | The amount of available random access memory.  Units: Bytes |
| **FreeStorageSpace** | **The amount of available storage space.**  **Units: Bytes** |
| MaximumUsedTransactionIDs | The maximum transaction ID that has been used. Applies to PostgreSQL.  Units: Count |
| NetworkReceiveThroughput | The incoming (Receive) network traffic on the DB instance, including both customer database traffic and Amazon RDS traffic used for monitoring and replication.  Units: Bytes/second |
| NetworkTransmitThroughput | The outgoing (Transmit) network traffic on the DB instance, including both customer database traffic and Amazon RDS traffic used for monitoring and replication.  Units: Bytes/second |
| OldestReplicationSlotLag | The lagging size of the replica lagging the most in terms of WAL data received. Applies to PostgreSQL.  Units: Megabytes |
| **ReadIOPS** | **The average number of disk read I/O operations per second.**  **Units: Count/Second** |
| **ReadLatency** | **The average amount of time taken per disk I/O operation.**  **Units: Seconds** |
| ReadThroughput | The average number of bytes read from disk per second.  Units: Bytes/Second |
| **ReplicaLag** | **The amount of time a Read Replica DB instance lags behind the source DB instance. Applies to MySQL, MariaDB, and PostgreSQL Read Replicas.**  **Units: Seconds** |
| ReplicationSlotDiskUsage | The disk space used by replication slot files. Applies to PostgreSQL.  Units: Megabytes |
| SwapUsage | The amount of swap space used on the DB instance.  Units: Bytes |
| TransactionLogsDiskUsage | The disk space used by transaction logs. Applies to PostgreSQL.  Units: Megabytes |
| TransactionLogsGeneration | The size of transaction logs generated per second. Applies to PostgreSQL.  Units: Megabytes/second |
| **WriteIOPS** | **The average number of disk write I/O operations per second.**  **Units: Count/Second** |
| **WriteLatency** | **The average amount of time taken per disk I/O operation.**  **Units: Seconds** |
| WriteThroughput | The average number of bytes written to disk per second.  Units: Bytes/Second |

***Amazon RDS Dimensions***

Amazon RDS metrics data can be filtered by using any of the dimensions in the following table:

|  |  |
| --- | --- |
| **Dimension** | **Description** |
| DBInstanceIdentifier | This dimension filters the data you request for a specific DB instance. |
| DBClusterIdentifier | This dimension filters the data you request for a specific Amazon Aurora DB cluster. |
| DBClusterIdentifier, Role | This dimension filters the data you request for a specific Amazon Aurora DB cluster, aggregating the metric by instance role (WRITER/READER). For example, you can aggregate metrics for all READER instances that belong to a cluster. |
| DatabaseClass | This dimension filters the data you request for all instances in a database class. For example, you can aggregate metrics for all instances that belong to the database class db.m1.small |
| EngineName | This dimension filters the data you request for the identified engine name only. For example, you can aggregate metrics for all instances that have the engine name mysql. |

***Creating CloudWatch Alarms to Monitor Amazon RDS***

You can create a CloudWatch alarm that sends an Amazon SNS message when the alarm changes state. An alarm watches a single metric over a time period you specify, and performs one or more actions based on the value of the metric relative to a given threshold over a number of time periods. The action is a notification sent to an Amazon SNS topic or Auto Scaling policy.

Alarms invoke actions for sustained state changes only. CloudWatch alarms will not invoke actions simply because they are in a particular state, the state must have changed and been maintained for a specified number of periods. The following procedures outlines how to create alarms for Amazon RDS.

**To set alarms using the CloudWatch console**

1. Sign in to the AWS Management Console and open the CloudWatch console at<https://console.aws.amazon.com/cloudwatch/>.
2. Choose **Alarms** and then choose **Create Alarm**. This launches the **Create Alarm Wizard**.
3. Choose **RDS Metrics** and scroll through the Amazon RDS metrics to locate the metric you want to place an alarm on. To display just the Amazon RDS metrics in this dialog box, search for the identifier of your resource. Select the metric to create an alarm on and then choose **Next**.
4. Fill in the **Name**, **Description**, **Whenever** values for the metric.
5. If you want CloudWatch to send you an email when the alarm state is reached, in the **Whenever this alarm:** field, choose **State is ALARM**. In the **Send notification to:** field, choose an existing SNS topic. If you select **Create topic**, you can set the name and email addresses for a new email subscription list. This list is saved and appears in the field for future alarms.

**Note**

If you use **Create topic** to create a new Amazon SNS topic, the email addresses must be verified before they receive notifications. Emails are only sent when the alarm enters an alarm state. If this alarm state change happens before the email addresses are verified, they do not receive a notification.

1. At this point, the **Alarm Preview** area gives you a chance to preview the alarm you’re about to create. Choose **Create Alarm**.

**To set an alarm using the AWS CLI**

* Call [put-metric-alarm](http://docs.aws.amazon.com/cli/latest/reference/put-metric-alarm.html). For more information, see [*AWS CLI Command Reference*](http://docs.aws.amazon.com/cli/latest/reference/).

**To set an alarm using the CloudWatch API**

* Call [PutMetricAlarm](http://docs.aws.amazon.com/AmazonCloudWatch/latest/APIReference/API_PutMetricAlarm.html). For more information, see [*Amazon CloudWatch API Reference*](http://docs.aws.amazon.com/AmazonCloudWatch/latest/APIReference/)

**Publishing Database Engine Logs to Amazon CloudWatch Logs**

You can configure your Amazon RDS database engine to publish log data to a log group in Amazon CloudWatch Logs. With CloudWatch Logs, you can perform real-time analysis of the log data, and use CloudWatch to create alarms and view metrics. You can use CloudWatch Logs to store your log records in highly durable storage, which you can manage with the CloudWatch Logs Agent. For example, you can determine when to rotate log records from a host to the log service, so you can access the raw logs when you need to.

You can export logs for Amazon RDS MariaDB (versions 10.0 and 10.1), Amazon RDS MySQL (versions 5.6 and 5.7), and Aurora MySQL.

**Note**

You must have a Service Linked Role before you enable log data publishing. For more information about Service Linked Roles, see the following: [Using Service-Linked Roles for Amazon RDS](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/UsingWithRDS.IAM.ServiceLinkedRoles.html).

For specific requirements for these engines, see the following:

* [Publishing MariaDB Logs to CloudWatch Logs](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_LogAccess.Concepts.MariaDB.html#USER_LogAccess.MariaDB.PublishtoCloudWatchLogs)
* [Publishing MySQL Logs to CloudWatch Logs](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_LogAccess.Concepts.MySQL.html#USER_LogAccess.MySQLDB.PublishtoCloudWatchLogs)
* [Publishing Amazon Aurora MySQL Logs to Amazon CloudWatch Logs](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/AuroraMySQL.Integrating.CloudWatch.html)

**Configuring CloudWatch Log Integration**

To publish your database log files to CloudWatch Logs, choose which logs to publish. Make this choice in the **Advanced Settings** section when you create a new DB instance. You can also modify an existing DB instance to begin publishing.


                    Add CloudWatch Logs
                

After you have enabled publishing, Amazon RDS continuously streams all of the DB instance log records to a log group. For example, you have a log group /aws/rds/instance/log type for each type of log that you publish. This log group is in the same AWS Region as the database instance that generates the log.

After you have published log records, you can use CloudWatch Logs to search and filter the records. For more information about searching and filtering logs, see [Searching and Filtering Log Data](http://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/MonitoringLogData.html).

**Viewing DB Instance Metrics**

Amazon RDS provides metrics so that you can monitor the health of your DB instances and DB clusters. You can monitor both DB instance metrics and operating system (OS) metrics.

This section provides details on how you can view metrics for your DB instance using the RDS console and CloudWatch. For information on monitoring metrics for the operating system of your DB instance in real time using CloudWatch Logs, see [Enhanced Monitoring](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_Monitoring.OS.html).

**Viewing Metrics by Using the Console**

**To view DB and OS metrics for a DB instance**

1. Sign in to the AWS Management Console and open the Amazon RDS console at<https://console.aws.amazon.com/rds/>.
2. In the navigation pane, choose **Instances**.
3. Select the check box to the left of the DB you need information about. For **Show Monitoring**, choose the option for how you want to view your metrics from these:
   * **CloudWatch** – Shows a summary of DB instance metrics available from Amazon CloudWatch. Each metric includes a graph showing the metric monitored over a specific time span.
   * **Enhanced monitoring** – Shows a summary of OS metrics available for a DB instance with Enhanced Monitoring enabled. Each metric includes a graph showing the metric monitored over a specific time span.
   * **OS Process list** – Shows details for each process running in the selected instance.


                            RDS metrics viewing options
                        

**Tip**

You can select the time range of the metrics represented by the graphs with the time range drop-down list.

You can choose any graph to bring up a more detailed view. You can also apply metric-specific filters to the data.