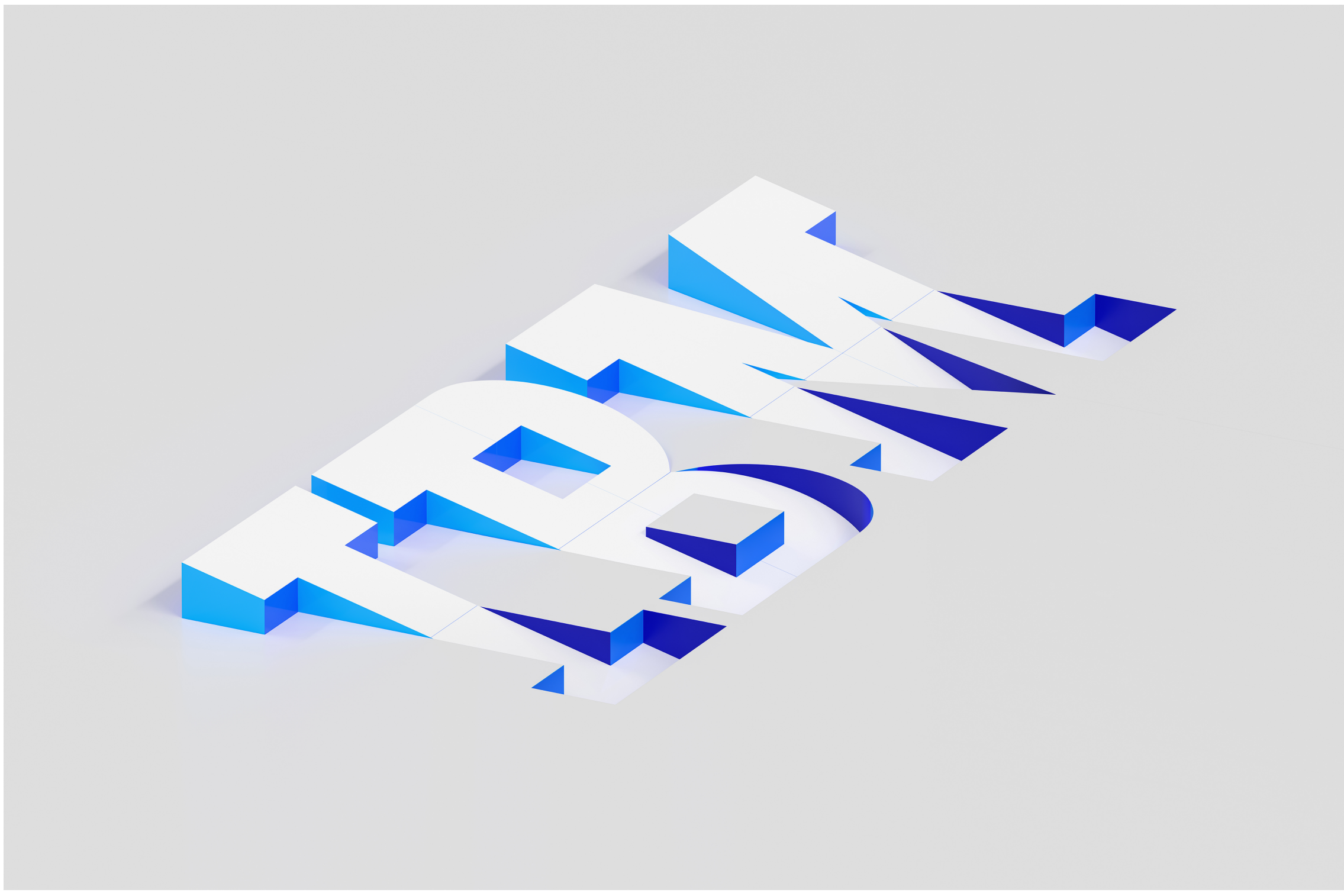
IBM Cloud

Databases For Elasticsearch

Product guide



IBM

Edition notices

This PDF was created on 2026-03-22 as a supplement to *Databases For Elasticsearch* in the IBM Cloud docs. It might not be a complete set of information or the latest version. For the latest information, see the IBM Cloud documentation at https://cloud.ibm.com/docs/databases-for-elasticsearch.

Getting started with Databases for Elasticsearch

This tutorial guides you through the steps to quickly start using an IBM Cloud® Databases for Elasticsearch deployment by provisioning an instance, setting your Admin password and connecting to it.

Follow these steps to complete the tutorial:

Follow these steps to complete the tutorial:

Follow these steps to complete the tutorial:

Follow these steps to complete the tutorial:

Before you begin

- You need an IBM Cloud account.

Step 1: Choose your plan

Databases for Elasticsearch offers two different plans:

- Databases for Elasticsearch **Enterprise** deploys the Basic version of Elasticsearch.
- Databases for Elasticsearch **Platinum** deploys the Platinum version of Elasticsearch.

Both plans provide you with a fully managed and scalable Elasticsearch service, allowing you to focus on your applications and data rather than the underlying infrastructure.

Using APIs

Use the Cloud Databases API to work with your Databases for MongoDB instance. The resource controller API is used to provision an instance.

You will need an API key to perform actions via the API. Follow these steps to create an IBM Cloud API key that enables you to use the API to provision infrastructure into your account. You can create up to 20 API keys.

> **Note:** For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Step 2: Provision through the console

1. Log in to the IBM Cloud console.
2. Click the Databases for Elasticsearch service in the **catalog**.
3. Follow these steps to provision a Databases for Elasticsearch instance.
4. When your instance is provisioned, click the instance name to view more information.

Step 2: Provision through the CLI

You can provision a Databases for Elasticsearch instance by using the CLI. If you don't already have it, you need to install the IBM Cloud CLI.

You can follow these steps to provision an Databases for Elasticsearch instance.

Step 2: Provision through the resource controller API

Follow these steps to provision an Databases for Elasticsearch instance using the Resource Controller API.

Step 2: Provision through Terraform

You need an API key to perform actions via Terraform. Follow these steps to create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

> **Note:** For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Once you have an API Key, follow these steps to provision an Databases for Elasticsearch instance using Terraform.

Step 3: Set the admin password

The admin user

When you provision a Databases for Elasticsearch deployment, an `admin` user is automatically created.

> **Important:** Set the admin password before using it to connect.

Set the admin password through the UI

Set your admin password through the UI by selecting your instance from the IBM Cloud Resource list. Then, select **Settings**. Next, select *Change Database Admin password*.

Set the admin password through the CLI

Use the `cdb user-password` command from the IBM Cloud CLI Cloud Databases plug-in to set the admin password.

For example, to set the admin password for your deployment, use the following command:

```
$ ibmcloud cdb user-password <INSTANCE_NAME_OR_CRN> admin <NEWPASSWORD>
```

Set the admin password through the API

YOu can use the `id` parameter obtained in the response to Step 2 above with the Set specified user's password endpoint to set the admin password.

```
$ curl -X PATCH -H "Authorization: Bearer <TOKEN>" \
    -H 'Content-Type: application/json' \
    -d '{"password":"newrootpasswordsupersecure21"}' \
```

```
"https://api.<REGION>.databases.cloud.ibm.com/v5/ibm/deployments/<DEPLOYMENT_ID>/users/database/admin"
```

⚠ **Important:** The `id` parameter needs to be URL-encoded for the above API call to work.

Setting the admin password through Terraform

The admin password is passed in as one of the database resource parameters in the Terraform script. There is no need for any further action.

Step 4: Connect to your Databases for Elasticsearch instance

Connect to your deployment using Kibana, an open source tool that adds visualization capabilities to your Elasticsearch database. This tutorial runs Kibana in a Docker container by using the Kibana image from the Docker image repository.

Before you begin

- Install Docker so that you can pull the Kibana container image to connect to Databases for Elasticsearch.
- If you prefer to avoid running Kibana locally and install Docker, you can also deploy Kibana using IBM Cloud® Code Engine. For more information, see Deploy Kibana using Code Engine and connect to your Databases for Elasticsearch instance .

To connect, Kibana needs the username, password, URL, and port for your Elasticsearch deployment. It also needs the Elasticsearch TLS certificate to access the database. To get this, copy the certificate information from the *Endpoints* section on the *Overview* page of your created Elasticsearch instance. Then, download the certificate to a local folder. You can use the name that is provided in the download, or your own file name.

⚠ **Important:** Remember where you save the certificate on your file system. If you are running Kibana locally, not in Docker, then the certificate goes in `$KIBANA_HOME/config/<filename>`.

Set up Kibana

Before running the Docker container that includes Kibana, create a configuration file in the same folder with the downloaded Elasticsearch certificate from Step 1. The configuration file will contain some basic Kibana settings as follows.

Create a YAML file called `kibana.yml`. Inside the file, you need the following Kibana configuration settings:

```
$ elasticsearch.ssl.certificateAuthorities: "/usr/share/kibana/config/cacert"
elasticsearch.username: "admin"
elasticsearch.password: "<password>"
elasticsearch.hosts: ["https://<hostname:port>"]
server.name: "kibana"
server.host: "0.0.0.0"
```

The first setting, `elasticsearch.ssl.certificateAuthorities`, is the location where Docker will store the Elasticsearch certificate. It gets placed in this location when you first run Docker. You can change it to a location of your choice, but the example path is the Kibana's configuration directory. Ensure that the certificate name (in our example "cert") within the `kibana.yml` and the certificate name file stored in Step 1 have the same name.

Next, is `elasticsearch.username` and `elasticsearch.password`. Use the deployment's admin username and password. Be sure that you set the admin password before trying to connect. For `elasticsearch.hosts`, enter the deployment's hostname and port, which is separated by a `:`.

Lastly, `server.name` is a machine-readable name for the Kibana instance and `server.hosts` is the host of the backend server where you can connect to Kibana in your web browser.

These settings are just a simplified example to get started. For more infortmation, see Configure Kibana.

If you are running Kibana locally, not in Docker, then the YAML file goes in `$KIBANA_HOME/config/kibana.yml`, where Kibana reads its configuration.

Run the Kibana Container

Now that the `kibana.yml` file is set up, use Docker to attach the YAML file and your certificate file to the Docker container, while pulling the `<kibana_version>` image from the Docker image repository.

⚠ **Important:** Use an image with a version of Kibana that is compatible with the version of Elasticsearch that your deployment is running. Retrieve the Elasticsearch version from the `https_endpoint` API endpoint by using your preferred http client. For more information, see the Elasticsearch compatibility matrix.

Here is an example with curl. If you don't have the certificate that is installed, use the `--insecure` flag to disable peer verification. The `<http_endpoint>` can be found in your instance's *Endpoints* UI:

```
$ curl --cacert <path-to-cert> <https_endpoint>
```

Next, run the Docker command in your terminal to start the Kibana container.

```
$ docker container run -it --name kibana \
-v <path_to_config_folder_created_in_step_1>:/usr/share/kibana/config \
-p 5601:5601 docker.elastic.co/kibana/kibana:<kibana_version>
```

The Docker command has one volume that is attached with the `-v` flag. These are mounted to the Kibana container at the path `/usr/share/kibana/config/`, which is a configuration directory where Kibana looks for configuration files.

- The `-p` specifies which port is exposed from the container, and the port you use to access Kibana.
- The Kibana version should correspond to the version of Elasticsearch you are using.

When you run the command from your terminal, it downloads the Kibana Docker image and runs Kibana. Once Kibana has connected to your Databases for Elasticsearch deployment and is running successfully, you see the output in your terminal.

```
$ log   [01:19:31.839] [info][status][plugin:<kibana_version>] Status changed from uninitialized to green - Ready
log   [01:19:31.925] [info][status][plugin:elasticsearch@<kibana_version>] Status changed from uninitialized to yellow - Waiting for Elasticsearch
log   [01:19:32.120] [info][status][plugin:timelion@<kibana_version>] Status changed from uninitialized to green - Ready
log   [01:19:32.134] [info][status][plugin:console@<kibana_version>] Status changed from uninitialized to green - Ready
log   [01:19:32.147] [info][status][plugin:metrics@<kibana_version>] Status changed from uninitialized to green - Ready
log   [01:19:33.132] [info][status][plugin:elasticsearch@<kibana_version>] Status changed from yellow to green - Ready
log   [01:19:33.378] [info][listening] Server running at http://0.0.0.0:5601
```

Tip: If you don't want to see the output of Kibana in your terminal, use the `-d` flag to detach the container.

Visit `http://0.0.0.0:5601` in your browser to see Kibana. `0.0.0.0` is the `server.host` in `kibana.yml` and `5601` is the port that is exposed from the container. Once you go to the URL, a pop-up window prompts you for your username and password. Use the admin credentials, or any other credentials that you made, to access to your deployment. The credentials don't have to be the same username and password you provided in the `kibana.yml` file.

Next steps

For more information, see the Elasticsearch documentation.

Looking for more tools on managing your databases and data? You can connect to your deployment with the IBM Cloud CLI, the Cloud Databases CLI plug-in, or the Cloud Databases API.

If you plan to use Databases for Elasticsearch for your applications, check out Connecting an external application and Connecting an IBM Cloud application.

To ensure the stability of your applications and your database, check out High-availability and Performance.

Plan overview

Cloud Databases offers two Elasticsearch services: Databases for Elasticsearch Enterprise and Databases for Elasticsearch Platinum. Both plans provide you with a fully managed and scalable Elasticsearch service, allowing you to focus on your applications and data rather than the underlying infrastructure. Databases for Elasticsearch Enterprise Plan deploys the Basic version of Elasticsearch. Databases for Elasticsearch Platinum Plan deploys the Platinum version of Elasticsearch.

> **Note: Databases for Elasticsearch Platinum** is only available on Isolated Compute and Dedicated Cores hosting models. Dedicated cores will be replaced by Isolated Compute as of May, 2025.

Our strategic partnership with Elastic since January 2023 means that we are able to offer more and richer functionality, as well as world-class levels of support.

> **Note:** Databases for Elasticsearch Enterprise Plan does not deploy Elasticsearch Enterprise version.

All Databases for Elasticsearch clusters on any plan can make full use of other Elastic Stack components, such as Kibana, Logstash, and Beats.

Databases for Elasticsearch Enterprise Plan

Databases for Elasticsearch Enterprise Plan has all the key functionalities of the Elasticsearch Service, such as Role Based Access Control (RBAC) and Index Lifecycle Management (ILM), security, alerting, monitoring, reporting, and graph capabilities.

Databases for Elasticsearch Platinum Plan

Databases for Elasticsearch Platinum Plan offers all the features of the Enterprise Plan, plus a richer array of functionality around integrations (connectors and web crawlers), security features (logging and access control), document-level security, and machine learning capabilities (anomaly detection, data frame analysis, and inference and model management), among others.

Access to Platinum features on a fully managed IBM deployment is primarily determined by the accessibility of a feature through the Elasticsearch API and the need for modifications to configuration files. Configuration files update is not currently supported for fully managed IBM deployments.

> **Note:** For questions regarding specific feature support, submit a support ticket.

Databases for Elasticsearch Enterprise and Platinum Feature Comparison

Feature	Enterprise	Platinum
Storage Types - Inverted index (for search)	x	x
Storage Types - Evaluating calculated fields at index time	x	x
Storage Types - Runtime fields	x	x
Storage Types - Lookup runtime field	x	x
Storage Types - Document store (for unstructured)	x	x
Storage Types - Columnar store (for analytics)	x	x
Storage Types - BKD trees (for numeric, dates, & geo)	x	x
Storage Types - Flattened field type	x	x
Storage Types - Histogram field type	x	x
Storage Types - Match only text field type	x	x
Storage Types - Shape field type	x	x

Storage Types - Vector field type	x	x
Storage Types - Version field type	x	x
Storage Types - Wildcard field type	x	x
Data Management - Snapshot/restore APIs	x	x
Data Management - Snapshot as simple archives		x
Data Management - Snapshot lifecycle management	x	x
Data Management - Data rollups	x	x
Data Management - Data streams	x	x
Data Management - Index lifecycle management	x	x
Stack Management - Upgrade Assistant	x	x
Stack Management - License management	x	x
Stack Management - Centralized Logstash pipeline management		x
Scalability & resiliency - Clustering & high availability	x	x
Scalability & resiliency - Cluster rebalancing	x	x
Elastic Stack Security - Encrypted communications	x	x
Elastic Stack Security - Role-based access control	x	x
Elastic Stack Security - File and native authentication	x	x
Elastic Stack Security - Kibana Spaces, Kibana feature controls	x	x
Elastic Stack Security - Kibana sub-feature privileges		x
Elastic Stack Security - API keys management	x	x
Elastic Stack Security - Kibana audit logging		x
Elastic Stack Security - IP filtering		x
Elastic Stack Security - Elasticsearch Token Service		x
Elastic Stack Security - Attribute-based access control		x
Elastic Stack Security - Field- and document-level security		x
Stack Monitoring - Multi-stack monitoring		x
Stack Monitoring - Kibana alerting and actions	x	x
Alerting - Noise reduction capabilities	x	x

Alerting - Geofencing / Tracking containment rule type		x
Alerting - Search threshold rule types for Discover	x	x
Alerting - Case Management	x	x
Alerting - Connectors (Actions) like: email, webhook, Jira, MS Teams, PagerDuty, Slack, IBM Resilient, ServiceNow®, OpsGenie,		x
Full-text search - Vector Search	x	x
Full-text search - Semantic Search		x
Analytics - Geoline aggregation		x
Analytics - Geoshape aggregations		x
Analytics - Geohexgrid aggregations		x
Analytics - Graph exploration		x
Analytics - Text categorization aggregation		x
Data Exploration - Drilldown to URL		x
Machine LEarning - Data Drift		x
Anomaly Detection - Single Metric / Multi Metric		x
Anomaly Detection - Population / Entity Analysis		x
Anomaly Detection - Log message categorization		x
Anomaly Detection - Rare analysis		x
Anomaly Detection - Root cause indication		x
Anomaly Detection - Forecasting on time series		x
Anomaly Detection - Root cause indication		x
Data Frame Analysis - Outlier Detection		x
Data Frame Analysis - Regression		x
Data Frame Analysis - Classification		x
Data Frame Analysis - Feature Importance		x
Model Management - Inference		
Model Management - Third party models support		x
Model Management - Kibana space model separation		x
Model Management - Elastic Learned Sparse Encoder for Semantic Search		x

Data Exploration - Graph analytics	x
Collaboration - PDF and PNG reports	x
Content Management - Custom banners	x
Elastic APM - Service maps	x
Elastic APM - Correlations	x
Elastic Security - Detection alert external actions	x
Elastic Security - Ransomware prevention	x
Elastic Security - Malicious behavior protection	x
Elastic Security - Memory threat protection	x
Elastic Security - Self healing	x
Elastic Security - Host Isolation	x
Elastic Security - Customizable on-endpoint protection notifications	x
Elastic Security - CIS posture findings and dashboards	x
Integrations - Atlassian Jira, Swimlane SOAR, IBM Resilient, ServiceNow ITOM, ITSM, SecOps	x
Elastic Maps - Tracking alerts	x
Elastic Maps - Containment alerts	x
Elastic Maps - Geofencing	x
Elastic Enterprise Search - Ingestion pipeline management	x

Cloud Databases for Elasticsearch Plan Features Comparison

Hosting models

Hosting models overview

To allow for reliable resource allocation, Cloud Databases offers two hosting models; Shared Compute and Isolated Compute. Cloud Databases Shared Compute is a flexible option for your database deployment that preserves performance predictability. Cloud Databases Isolated Compute is our recommendation for production enterprise applications, providing more precise control and enterprise features.

Scaling your Shared Compute or Isolated Compute databases is currently available using the CLI, API, or Terraform only.

Cloud Databases Shared Compute

Shared Compute is a flexible multi-tenant offering for dynamic, fine-tuned, and decoupled capacity selections.

When provisioning Shared Compute through the IBM Cloud console, you have the option to select between the following initial resource allocation presets: **Small** (1 CPU and 8 GB RAM for RabbitMQ, 0.5 CPU and 4 GB RAM for all other databases) or **Custom** (≥ 2 CPU and ≥ 4 GB RAM). Small has a fixed amount of CPU and RAM, but you can change disk. Custom can be completely customized.

With Small allocation preset, you can test out the database with the smallest resource allocation. If you have higher performance requirements, you can easily leverage the flexibility of the Shared model with the Custom allocation preset. With the ability to select the amount of CPU and RAM resources you receive, performance can be scaled to fit your workload.

Cloud Databases Isolated Compute

Isolated Compute is a secure single-tenant offering for complex, highly performant enterprise workloads. By placing your deployment and all associated user-data management agents on an isolated machine, Cloud Databases Isolated Compute provides dedicated computing resources, dedicated storage bandwidth, and hypervisor-level isolation.

When provisioning, choose an initial host size for your instance. Storage is still selected separately, allowing you to determine the size of disk and number of *IOPS* your database receives.

Note: CPU and RAM autoscaling is not supported on Isolated Compute. Disk autoscaling is available. If you provisioned an isolated instance or switched over from a deployment with autoscaling, monitor your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

Isolated Compute sizing

Isolated Compute features 6 size selections:

- 4 CPU x 16 RAM
- 8 CPU x 32 RAM
- 8 CPU x 64 RAM
- 16 CPU x 64 RAM
- 32 CPU x 128 RAM
- 30 CPU x 240 RAM

Cloud Databases Shared Compute

Shared Compute is a flexible multi-tenant offering for dynamic, fine-tuned, and decoupled capacity selections.

Each database instance receives a deterministic CPU allocation. If an instance is provisioned without selecting a CPU amount, Shared Compute automatically allocates a small amount of CPU to your database up to a 2 core max. Automatic CPU is provided at a 1:8 ratio of CPU:RAM; therefore, a user with 4 GB RAM receives 4/8th of a CPU; a user with 8 GB RAM receives 1 CPU; and an user with 20 GB RAM receives 2 CPU due to the 2 CPU limit.

For this fractional, automatic vCPU allocation, input your RAM and disk allocation needs. Selecting `multitenant` means that our system will handle vCPU allocation at a 1/8 ratio to your RAM allocation. We recommend integer allocations when specifying vCPU allocations.

If you have higher performance requirements than 2 CPU, you can easily leverage the flexibility of the Shared model. With the ability to select the amount of CPU and RAM resources you receive, performance can be scaled to fit your workload. Additionally, if you know that your instance will experience variable demand, use RAM autoscaling to set not only the expected load and duration that would initiate resource scaling, but also the resource and cost limit your database will scale to.

Note: Because of each service's individual requirements, Cloud Databases has minimum resource requirements in place for all Shared Compute instances. When all existing multi-tenant instances are transitioned to Shared Compute, these minimum resource requirements will be applied. Current multi-tenant instances will not be charged (that is, they will be *grandfathered*) for any increase to up to these minimum resource requirements actioned by IBM until May 2025. For more information, see Hosting model grandfathering.

Cloud Databases Isolated Compute

Isolated Compute is a secure single-tenant offering for complex, highly-performant enterprise workloads. By placing your deployment and all associated user-data management agents on an isolated machine, Cloud Databases Isolated Compute provides dedicated computing resources, dedicated storage bandwidth, and hypervisor-level isolation.

When provisioning, choose the CPU x RAM size for the machine to set up your database. This machine will be exclusively assigned to running your database instance. Storage is still selected separately, allowing you to determine the size of disk and number of *IOPS* your database receives. Scale your database and change your machine size using your preferred method: the Cloud Databases CLI plug-in, the Cloud Databases API, using Terraform, or through pre-built, open-source and, enterprise-ready Terraform IBM Modules (TIM) that also support the auto-scaling feature.

Isolated Compute sizing

Isolated Compute features 6 size selections:

- 4 CPU x 16 RAM
- 8 CPU x 32 RAM
- 8 CPU x 64 RAM
- 16 CPU x 64 RAM
- 32 CPU x 128 RAM
- 30 CPU x 240 RAM

The `host_flavor` parameter defines your Compute sizing. Input the appropriate value for your desired size. To provision a Shared Compute instance, specify `multitenant`. All other options place you on different Isolated Compute sizes.

Host flavor	host_flavor value
Shared Compute	`multitenant`
4 CPU x 16 RAM	`b3c.4x16.encrypted`
8 CPU x 32 RAM	`b3c.8x32.encrypted`
8 CPU x 64 RAM	`m3c.8x64.encrypted`
16 CPU x 64 RAM	`b3c.16x64.encrypted`
32 CPU x 128 RAM	`b3c.32x128.encrypted`
30 CPU x 240 RAM	`m3c.30x240.encrypted`

Host flavor sizing parameter

Cloud Databases Shared Compute

Shared Compute is a flexible multi-tenant offering for dynamic, fine-tuned, and decoupled capacity selections.

Each database instance receives a deterministic CPU allocation. If an instance is provisioned without selecting a CPU amount, Shared Compute automatically allocates a small amount of CPU to your database up to a 2 core max. Automatic CPU is provided at a 1:8 ratio of CPU:RAM; therefore, a user with 4 GB RAM receives 4/8th of a CPU; a user with 8 GB RAM receives 1 CPU; and an user with 20 GB RAM receives 2 CPU due to the 2 CPU limit.

For this fractional, automatic vCPU allocation, simply input your RAM and disk allocation needs. Selecting `multitenant` means that our system will handle vCPU allocation at a 1/8 ratio to your RAM allocation. We recommend integer allocations when specifying vCPU allocations.

If you have higher performance requirements than 2 CPU, you can easily leverage the flexibility of the Shared model. With the ability to select the amount of CPU and RAM resources you receive, performance can be scaled to fit your workload. Additionally, if you know that your instance will experience variable demand, use RAM autoscaling to set not only the expected load and duration that would initiate resource scaling, but also the resource and cost limit your

database will scale to.

> **Note:** Because of each service's individual requirements, Cloud Databases has minimum resource requirements in place for all Shared Compute instances. When all existing multi-tenant instances are transitioned to Shared Compute, these minimum resource requirements will be applied. Current multi-tenant instances will not be charged (that is, they will be *grandfathered*) for any increase to up to these minimum resource requirements actioned by IBM until May 2025. For more information, see Hosting model grandfathering.

Cloud Databases Isolated Compute

Isolated Compute is a secure single-tenant offering for complex, highly-performant enterprise workloads. By placing your deployment and all associated user-data management agents on an isolated machine, Cloud Databases Isolated Compute provides dedicated computing resources, dedicated storage bandwidth, and hypervisor-level isolation.

When provisioning, choose the CPU x RAM size for the machine to set up your database. This machine will be exclusively assigned to running your database instance. Storage is still selected separately, allowing you to determine the size of disk and number of *IOPS* your database receives. Scale your database and change your machine size using your preferred method: the Cloud Databases CLI plug-in, the Cloud Databases API, using Terraform, or through pre-built, open-source, and enterprise-ready Terraform IBM Modules (TIM) that also support the auto-scaling feature.

Isolated Compute sizing

Isolated Compute features 6 size selections:

- 4 CPU x 16 RAM
- 8 CPU x 32 RAM
- 8 CPU x 64 RAM
- 16 CPU x 64 RAM
- 32 CPU x 128 RAM
- 30 CPU x 240 RAM

The `host_flavor` parameter defines your Compute sizing. Input the appropriate value for your desired size. To provision a Shared Compute instance, specify `multitenant`. All other options place you on different Isolated Compute sizes.

Host flavor	host_flavor value
Shared Compute	`multitenant`
4 CPU x 16 RAM	`b3c.4x16.encrypted`
8 CPU x 32 RAM	`b3c.8x32.encrypted`
8 CPU x 64 RAM	`m3c.8x64.encrypted`
16 CPU x 64 RAM	`b3c.16x64.encrypted`
32 CPU x 128 RAM	`b3c.32x128.encrypted`
30 CPU x 240 RAM	`m3c.30x240.encrypted`

Host flavor sizing parameter

Cloud Databases Shared Compute

Shared Compute is a flexible multi-tenant offering for dynamic, fine-tuned, and decoupled capacity selections.

Each database instance receives a deterministic CPU allocation. If an instance is provisioned without selecting a CPU amount, Shared Compute automatically allocates a small amount of CPU to your database up to a 2 core max. Automatic CPU is provided at a 1:8 ratio of CPU:RAM; therefore, a user with 4 GB RAM receives 4/8th of a CPU; a user with 8 GB RAM receives 1 CPU; and an user with 20 GB RAM receives 2 CPU due to the 2 CPU limit.

For this fractional, automatic vCPU allocation, simply input your RAM and disk allocation needs. Selecting `multitenant` means that our system will handle vCPU allocation at a 1/8 ratio to your RAM allocation. We recommend integer allocations when specifying vCPU allocations.

If you have higher performance requirements than 2 CPU, you can easily leverage the flexibility of the Shared model. With the ability to select the amount of

CPU and RAM resources you receive, performance can be scaled to fit your workload. Additionally, if you know that your instance will experience variable demand, use RAM autoscaling to set not only the expected load and duration that would initiate resource scaling, but also the resource and cost limit your database will scale to.

Note: Because of each service's individual requirements, Cloud Databases has minimum resource requirements in place for all Shared Compute instances. When all existing multi-tenant instances are transitioned to Shared Compute, these minimum resource requirements will be applied. Current multi-tenant instances will not be charged (that is, they will be *grandfathered*) for any increase to up to these minimum resource requirements actioned by IBM until May 2025. For more information, see Hosting model grandfathering.

Cloud Databases Isolated Compute

Isolated Compute is a secure single-tenant offering for complex, highly-performant enterprise workloads. By placing your deployment and all associated user-data management agents on an isolated machine, Cloud Databases Isolated Compute provides dedicated computing resources, dedicated IO and network bandwidth, and hypervisor-level isolation.

When provisioning, choose the CPU x RAM size for the machine to set up your database. This machine will be exclusively assigned to running your database instance. Storage is still selected separately, allowing you to determine the size of disk and number of *IOPS* your database receives. Scale your database and change your machine size using your preferred method: the Cloud Databases CLI plug-in, the Cloud Databases API, using Terraform, or through pre-built, open-source, and enterprise-ready Terraform IBM Modules (TIM) that also support the auto-scaling feature.

Note: CPU and RAM autoscaling is not supported on Cloud Databases Isolated Compute. Disk autoscaling is available. If you provisioned an isolated instance or switched over from a deployment with autoscaling, monitor your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

Isolated Compute sizing

Isolated Compute features 6 size selections:

- 4 CPU x 16 RAM
- 8 CPU x 32 RAM
- 8 CPU x 64 RAM
- 16 CPU x 64 RAM
- 32 CPU x 128 RAM
- 30 CPU x 240 RAM

The `host_flavor` parameter defines your Compute sizing. Input the appropriate value for your desired size. To provision a Shared Compute instance, specify `multitenant`. All other options place you on different Isolated Compute sizes.

Host flavor	host_flavor value
Shared Compute	`multitenant`
4 CPU x 16 RAM	`b3c.4x16.encrypted`
8 CPU x 32 RAM	`b3c.8x32.encrypted`
8 CPU x 64 RAM	`m3c.8x64.encrypted`
16 CPU x 64 RAM	`b3c.16x64.encrypted`
32 CPU x 128 RAM	`b3c.32x128.encrypted`
30 CPU x 240 RAM	`m3c.30x240.encrypted`

Host flavor sizing parameter

Isolated Compute capacity

Isolated Compute fully isolates your database, including database management pods (which touch user data). These management pods take up some overhead in your isolated compute instance, consuming a portion of the machine's compute. The following table shows the estimated remaining compute

for each Isolated Compute size.

Host flavor	CPU remaining	RAM remaining
4 CPU x 16 RAM	2.865	12.193
8 CPU x 32 RAM	6.855	26.952
8 CPU x 64 RAM	6.855	56.519
16 CPU x 64 RAM	14.835	56.519
32 CPU x 128 RAM	30.795	115.738
30 CPU x 240 RAM	28.8	223.596

Isolated Compute capacity

Provisioning

To provision a Cloud Databases service instance, select your **hosting type** from either Shared Compute or Isolated Compute.

To provision a Cloud Databases service instance, add a new `host_flavor` parameter. This parameter allows you to select either Shared Compute (`multitenant`) or Isolated Compute via assigning the parameter value for the requested Isolated instance size. Note that because Isolated Compute sizes implicitly include both CPU and RAM allocations, CPU and RAM sizes should not be provided with an Isolated Compute request.

To provision a Cloud Databases service instance, add a new `host_flavor` parameter. This parameter allows you to select either Shared Compute (`multitenant`) or Isolated Compute via assigning the parameter value for the requested Isolated instance size. Note that because Isolated Compute sizes implicitly include both CPU and RAM allocations, CPU and RAM sizes should not be provided with an Isolated Compute request.

To provision a Cloud Databases service instance, add a new `host_flavor` parameter. This parameter allows you to select either Shared Compute (`multitenant`) or Isolated Compute via assigning the parameter value for the requested Isolated instance size. Note that because Isolated Compute sizes implicitly include both CPU and RAM allocations, CPU and RAM sizes should not be provided with an Isolated Compute request.

For more detailed instructions, see your database specific page.

Scaling and switching between hosting models

For new hosting models, scaling and switching are similar operations. While scaling your database as you normally would, select a different **hosting type** from what your database instance is currently placed on to switch to and between Shared and Isolated Compute.

For new hosting models, scaling and switching are similar operations. While scaling your database as you normally would, switch to and between hosting models by adding a new `host_flavor` parameter set to the hosting model you wish to scale to. Then, moving to the hosting type is as simple as running a scale command with this hosting flavor targeted.

For new hosting models, scaling and switching are similar operations. While scaling your database as you normally would, switch to and between hosting models by adding a new `host_flavor` parameter set to the hosting model you wish to scale to. Then, moving to the hosting type is as simple as running a scale command with this hosting flavor targeted.

For more detailed instructions, commands, and parameters, see your database-specific page.

Note that switching hosting models does not cause downtime, as this is not a backup and restore migration. Instead, the same process is applied as for updates or database instance scaling, where database processes will perform a rolling restart. We recommend ensuring that your application has retry and reconnect login in place to immediately re-establish a connection, as existing connections will be dropped during this time.

Choosing between hosting models

Isolated Compute	Shared Compute

Single-tenanted databases with dedicated storage bandwidth. Database management agents are placed on isolated machine.	Multi-tenanted, logically separated databases sharing bandwidth. Database management pods are also multi-tenanted.
Receive all the available resources in your machine.	Transparent, deterministic CPU allocation. Know exactly what your performance will be and scale up and down as your workload requires.
Some of our database offerings, such as MongoDB Enterprise and Elasticsearch Platinum, will be solely provisioned on Isolated Compute. Future enhancements, such as cross-region replication may be supported solely on Isolated Compute.	Excludes some database offerings, such as MongoDB Enterprise and Elasticsearch Platinum.
Scalability is based on provided machine sizes.	Scalability is fine-grained and linear from a database-specific minimum configuration up to 28 CPU and 112 GB RAM.

Choosing between hosting models

Databases availability by hosting model

The following table shows which model is available for each database.

	Shared Compute	Isolated Compute
PostgreSQL		
MongoDB Standard		
MongoDB Enterprise		
Redis		
Elasticsearch Enterprise		
Elasticsearch Platinum		
MySQL		
RabbitMQ		

Cloud Databases hosting model availability

Hosting models transition timeline and transition placement

Transition timeline from existing hosting models to Isolated and Shared Compute

⚠ **Important:** Multi-tenant users that are automatically transitioned to Shared Compute will be *grandfathered*, meaning that they get RAM and CPU increased to the Shared Compute minimum resource allocations, if required. These increases will not be charged until May 2025.

August 2024 - RAM minimum allocation applied for multi-tenant instances

- Existing multi-tenant instances will begin the transition to Shared Compute; this means that first, RAM minimum allocation on multi-tenant instances will be applied (8 GB RAM for RabbitMQ, 4 GB RAM for all other databases), lifting the RAM of existing instances that fall below these minimums.
- All new provisioning requests will also have to abide to the minimum resource requirements (1 CPU and 8 GB RAM for RabbitMQ, 0.5 CPU and 4 GB RAM for all other databases).
- Existing dedicated core users will not be impacted by minimum resource requirements unless a scale or provision action is invoked on an instance that is currently below these minimums.
- Following this, multi-tenant databases will be gradually transitioned from non-determinstic CPU allocation to the deterministic Shared Compute CPU allocation. Ahead of this transition, monitor your database's CPU usage to determine what allocation is required to maintain your current

performance level.

- Existing multi-tenant users will be grandfathered through to May 2025 for both CPU and minimum RAM resource allocations that are automatically added.

September 2024 - Transition of multi-tenant instances to Shared Compute

- All new multi-tenant provisions will use Shared Compute.
- Existing multi-tenant instances will begin the transition to Shared Compute. First, RAM minimum allocation on multi-tenant instances will be applied (8 GB RAM for RabbitMQ, 4 GB RAM for all other databases), lifting the RAM of existing instances that fall below these minimums.
- All new provisioning requests will also have to abide to the minimum resource requirements (1 CPU and 8 GB RAM for RabbitMQ, 0.5 CPU and 4 GB RAM for all other databases).
- Existing dedicated core users will not be impacted by minimum resource requirements unless a scale or provision action is invoked on an instance that is currently below these minimums.
- Following this, multi-tenant databases will be gradually transitioned from non-determinstic CPU allocation to the deterministic Shared Compute CPU allocation.
- **Action:** Ahead of this transition, monitor your database's CPU usage to determine what allocation is required to maintain your current performance level.
- Existing multi-tenant users will be grandfathered through to May 2025 for both CPU and minimum RAM resource allocations that are automatically added.
- Dedicated cores provisioning remains available.

May 2025 - End of grandfathering for multi-tenant users, transitioning dedicated cores to Isolated Compute

- All existing multi-tenant users will be grandfathered for CPU and minimum RAM allocations until this time.
- At this date, multi-tenant instances will need to comply with the Shared Compute resource allocations.

Between May 18 and 26, 2025, dedicated core users will be transitioned to Isolated Compute. Grandfathering will be removed for Shared Compute instances between June 2 and 9, 2025. All Dedicated Cores instances will be transitioned to the nearest larger Isolated Compute size. Dedicated Core instances can follow the simple switchover steps to transition to Isolated Compute at any time by using the Cloud Databases CLI plug-in, the Cloud Databases API, or through Terraform.

Notifications will be sent ahead of changes, including at the following times:

- Before the transition of multi-tenant to Shared Compute, to notify you of expected changes.
- After all multi-tenant instances are transitioned to Shared Compute resource allocations, to recommend that you review your database performance and adjust resources as necessary.
- Before final shutdown of dedicated cores and the transition to Isolated Compute. Additionally, before the end of grandfathering of Shared Compute instances. You can also find all notifications at IBM Cloud announcements.

Ahead of the May 2025 date, if you have a multi-tenant instance, there are a few exceptions where grandfathering would no longer apply:

- If you have an existing database and change your RAM allocation only, you will be charged corresponding to the RAM changes.
- If you have an existing database and change your CPU allocation, you will be charged for all CPU and RAM allocated to your database.
- If you create a new Shared Compute instance, you will be charged for all CPU and RAM allocated to your database.
- If you transition your multi-tenant instance yourself to Shared Compute, you will be charged for all CPU and RAM allocated to your database.

Shared and Isolated Compute transition placement

To determine how existing hosting models will switch over to Shared and Isolated Compute, review the tables below. In the switchover, the assumption is that the starting points are old style multitenant (CPU unallocated, or 0) and dedicated cores.

If your current resource allocation is N CPU x M RAM (Non-RabbitMQ Databases):	**You will be automatically placed on (Non-RabbitMQ Databases):**
N = 0 CPU, M < 4 GB RAM	0.5 CPU x 4 GB RAM, Shared Compute
N = 0 CPU, 4 GB RAM < M ≤ 16 GB RAM	M/8 CPU x M GB RAM, Shared Compute
N = 0 CPU, M > 16 GB RAM	2 CPU x M GB RAM, Shared Compute

0 CPU < N ≤ 4 CPU, M ≤ 16 GB RAM	4 CPU x 16 GB RAM, Isolated Compute
4 CPU < N ≤ 8 CPU OR 16 GB RAM < M ≤ 32 GB RAM	8 CPU x 32 GB RAM, Isolated Compute
4 CPU < N ≤ 8 CPU OR 32 GB RAM < M ≤ 64 GB RAM	8 CPU x 64 GB RAM, Isolated Compute
8 CPU < N ≤ 16 CPU OR 32 GB RAM < M ≤ 64 GB RAM	16 CPU x 64 GB RAM, Isolated Compute
16 CPU < N ≤ 32 CPU OR 64 GB RAM < M ≤ 128 GB RAM	32 CPU x 128 RAM, Isolated Compute
16 CPU < N ≤ 30 CPU OR 64 GB RAM < M ≤ 240 GB RAM	30 CPU x 240 RAM, Isolated Compute

Automatic transition placement

If your current resource allocation is N CPU x M RAM (RabbitMQ):	**You will be automatically placed on (RabbitMQ):**
N = 0 CPU, M < 8 GB RAM	1 CPU x 8 GB RAM, Shared Compute
N = 0 CPU, 8 GB RAM < M ≤ 16 GB RAM	M/8 CPU x M GB RAM, Shared Compute
N = 0 CPU, M > 16 GB RAM	2 CPU x M GB RAM, Shared Compute
0 CPU < N ≤ 4 CPU, M ≤ 16 GB RAM	4 CPU x 16 GB RAM, Isolated Compute
4 CPU < N ≤ 8 CPU OR 16 GB RAM < M ≤ 32 GB RAM	8 CPU x 32 GB RAM, Isolated Compute
4 CPU < N ≤ 8 CPU OR 32 GB RAM < M ≤ 64 GB RAM	8 CPU x 64 GB RAM, Isolated Compute
8 CPU < N ≤ 16 CPU OR 32 GB RAM < M ≤ 64 GB RAM	16 CPU x 64 GB RAM, Isolated Compute
16 CPU < N ≤ 32 CPU OR 64 GB RAM < M ≤ 128 GB RAM	32 CPU x 128 RAM, Isolated Compute
16 CPU < N ≤ 30 CPU OR 64 GB RAM < M ≤ 240 GB RAM	30 CPU x 240 RAM, Isolated Compute

Automatic transition placement RabbitMQ

Hosting models FAQ

My Shared Compute database has 0 CPU. What does this mean?

Shared Compute automatically allocates CPU for customers that do not specify their CPU amount. On API v5, this automatically allocated setting shows as CPU = 0. We also return an "Allocation as Fraction" result that shows what specific CPU value you have. CPU is allocated at 1/8th of a RAM, starting from each databases' minimum CPU amount up to 2 CPU. This CPU value will continue to scale as the RAM value scales.

Per the Shared Compute transition placement **, my understanding is that because this instance has 8 GB of RAM, it should have 1 CPU allocated per member, but the cloud-databases CLI show 0 CPUs allocated.**

See the previous answer.

I want fractional CPU. How do I do that?

Scale your database to the `multitenant` hosting model, and its CPU to 0. This turns on the Shared Compute automatic CPU allocation, where CPU is 1/8th of the database's RAM allocation, up to 2 CPU.

What does it mean to have fractional CPU?

CPU time is divided into units called periods or time slices. In each time slice you get a duration of CPU run time.

In general, the number of cores you specify decides how much CPU time you get. The default kernel run time is 50ms; therefore, with one core of CPU available, there is 20 time slices of 50ms run time.

If an instance A requests 0.5 cores or 500 millicores, assuming for simplicity that there is only one core of CPU on a node or server, instance A will get 10 time slices of 50ms run time each to complete the task. If it cannot complete the task within 10 time slices, the kernel throttles the instance's process after the 10 runs. The remaining 10 time slices for this CPU (since 1 core can have 20 time slices using default run time duration) is then scheduled for other processes while A waits. Once that is done, instance A gets the CPU for another 10 time slices.

I want to scale my database's resources, but they don't seem to be scaling?

For Shared Compute instances, make sure to set the host flavor to `multitenant` and separately set your CPU and RAM values. For automatically scaling CPU (capped at 2 CPU), set the CPU to 0. To scale Isolated Compute, set the host flavor to your desired size; do not set CPU and RAM values since the Isolated size already includes allocations for both. For more information, see the Scaling documentation.

My database is on a hosting model, even though my Terraform configuration didn't specify it.

Starting August, we begun to switchover customer instances to the new hosting models. This can lead to Terraform scripts that do not reflect the actualstate of the database, since these scripts do not have the new `host_flavor` parameter. Your database will still function as normal, though we recommend adding the `host_flavor` parameter for future ease of use.

What does it mean to switch / migrate between hosting models? Will there be downtime?

Switching between databases is not a large migration, but a standard operation similar to our maintenance work. Therefore, it does not involve downtime, though we recommend your application to have retry and reconnect logic.

If we switch to a hosting model, or switch between hosting models, will the disk size be retained as per the current configuration?

Yes. Your disk will not be impacted due to hosting model changes.

Does autoscaling work?

Autoscaling continues to work for new hosting models. For Shared Compute, RAM and disk autoscaling remains available. For Isolated Compute, disk autoscaling is available.

Note: Due to the size jumps across Isolated Compute, we have currently disallowed RAM/CPU autoscaling on Isolated Compute. If there is interest in this feature, let our team know.

I’m getting a CLI failed status error or a 403 forbidden error, but on the UI and CLI it looks like my scale operation completed?

This is due to CLI token expiry. The CLI expires tokens to quickly to preserve the security of your system, meaning any operations in progress will show a fail state. We recommend refreshing your session, and checking the database for the results of the task.

Hosting models pricing

Elasticsearch, machine learning, and AI

At the heart of artificial intelligence (AI) and computer science lies machine learning (ML). It's where computers learn and adapt by using data and algorithms, just like humans.

In recent decades, technological progress in storage and processing power has paved the way for ML-based innovations. However, the game-changer arrived in 2023 with the introduction of ChatGPT, an AI chatbot that captivated the world. Today, ML's potential is sought after by businesses across the board.

Luckily for most enterprises, much of this can be achieved with existing technologies, such as Elasticsearch. Elasticsearch, a robust search-capable database, is well-suited to seamlessly incorporate essential ML features sought by enterprises embracing new technologies. Elastic have been doing just that, packing their Elasticsearch product with ML capabilities.

Storing Data

Machine learning algorithms try to make sense of unstructured data, like videos or images, by turning these assets into sets of numbers called vector embeddings. Once an asset like an image is transformed into a set of embeddings it can be stored in a database like any other data. Elasticsearch has a specific data type (dense vector type) for these embeddings.

Different ML algorithms (known as models) will analyze and transform data in different ways. ML models specialize in different types of data and tasks. For a full list of Elastic stack supported models, see Compatible third party NLP models. For a comprehensive list of open source ML models, see Hugging Face.

Querying Data

Once inside the database, you can make sense of these assets by using Elasticsearch's vector search. Given a search "term" (for example a vector embedding of the picture of a bird or a car), the search engine finds the vector embeddings in its data set that are mathematically closer, the "known nearest neighbor" or kNN. This produces a list of birds or cars that look similar to the search "term".

How Databases for Elasticsearch can help on your AI journey

Enterprise Plan

If you want to only store and search vector embeddings, the Enterprise Plan of Databases for Elasticsearch (which deploys Elastic's Basic version) may suffice for you. This plan supports the Dense Vector data type as well as the various flavors of vector search that Elastic offers.

Using this plan means that you have to generate the actual embeddings somewhere else and then upload them to the database, as the Enterprise Plan does not support the generation of embeddings.

The Enterprise Plan gives you more flexibility at the cost of higher complexity in your AI data pipeline.

Platinum Plan

For a richer set of functions, the Platinum Plan of Databases for Elasticsearch (which deploys Elastic's Platinum version) may be what you need. With the Platinum Plan you get access to all the features of the Enterprise Plan but also the ability to generate the embeddings themselves, either by using Elastic's own ML model ELSER (ELastic Sparse EncodeR), or by using any of the many open source ML models that Elastic supports.

The Platinum Plan provides a one-stop shop for generating, storing, and searching for vector embeddings.

Machine learning tutorial series

If you need some inspiration to get you started, we've put together a tutorial series on how to use Elastic's ML abilities with third-party models:

- Use Elasticsearch vector search capabilities
- Use ELSER, Elastic's Natural Language Processing model
- Use machine learning models with Elasticsearch to tag content

Plans and Pricing

Visit our plans and pricing pages for more details on features and costs.

The Dashboard overview

Overview

The *Overview* page shows you information about your IBM Cloud® Databases for Elasticsearch deployment. The overview includes essential identifying information.

Deployment details

- **Type:** The type of database that is offered by the service, and the database version that your service uses.
- **CRN (deployment ID):** The ID is a CRN (Cloud Resource Name) that uniquely identifies the database deployment. The CRN is used to refer to the database in the API and can be used with the CLI. The *Overview* pane shows details of your service.

Resources

The resources tile contains information and configuration options on the size and resource usage of your deployment. You can ccale disk, memory, and CPU and configure Autoscaling.

Recent tasks

Every time that you make administrative changes to your service (such as scaling, or taking a manual backup), a task starts up. The *Recent Tasks* panel shows the task name and progress bar for any running tasks, and a list of the most recently completed tasks. Depending on how busy your deployment is, successful tasks are shown for 24 - 48 hours. Unsuccessful tasks are shown for 7 - 8 days. Tasks can also be retrieved from the Cloud Databases API and CLI plug-in. A historical record of tasks from any time period is available through the IBM Cloud® Activity Tracker Event Routing integration.

Observability

Observability: The *Observability* tile provides access to the IBM Cloud® Monitoring, logging, and event tracking integrations available for your deployment.

- IBM Cloud® Activity Tracker Event Routing
- IBM® Cloud Logs
- IBM Cloud® Monitoring

Endpoints

The *Endpoints* pane within the *Overview* pane contains connection strings for your deployment. Each tab contains connection information that is tailored to the type of connection or the protocol that uses it. Basic information includes things like *hostname* and *port*, the TLS service proprietary certificate, TLS/SSL parameters, and the default database of your deployment.

Reference tables for the different connection types are available on the Getting credentials and connection strings page.

Connection strings reflect whether your deployment uses public or private endpoints. You can configure which endpoints are available on your deployment. For more information, see the Service endpoints integration page.

You can manage your Databases for Elasticsearch service through the Cloud Databases API. This panel provides the essential information for using the API. For more information, see the API reference page.

Backups and restore

The *Backups* tab is the UI for managing your deployments backups. All of the available backups are listed with their timestamps. Click a backup to grab its ID or to restore it into a new deployment. More information is on the Managing backups page.

Settings

The *Settings* tab contains the UI for many of the tunable settings for your deployment. You can

- view encryption details. Encryption at rest is enabled for all Databases for Elasticsearch deployments. If you brought your own encryption key from Key Protect, the panel provides a link to your Key Protect instance and the *Encryption Key* field has the name of the key.
- Change the admin password
- Implement or modify an IP allowlist
- Context-based restrictions

Service credentials

You can generate a new set of credentials for cases where you want to manually connect an app or external consumer to an IBM Cloud service. For more information, see Adding and viewing credentials.

View docs

The *View docs* link from the `Actions` drop list opens the main documentation page for Databases for Elasticsearch in a new tab.

Performance

IBM Cloud® Databases for Elasticsearch deployments can be scaled to your usage, configured to autoscale under certain resource conditions, or horizontally scaled with more Elasticsearch nodes. If you are tuning the performance of your deployment, consider a few factors.

Monitoring your deployment

Databases for Elasticsearch deployments offer an integration with the IBM Cloud® Monitoring service for basic monitoring of resource usage on your deployment. Many of the available metrics, like disk usage and IOPS, are presented to help you configure autoscaling on your deployment. Observing trends in your usage and configuring the autoscaling to respond to them can help alleviate performance problems before your databases become unstable due to resource exhaustion.

Elasticsearch sharding

When you add an index to Elasticsearch, it splits the data into shards and spreads those shards across the nodes in the cluster. The sharded configuration allows for Elasticsearch to run concurrent operations on your data across all the nodes. To gain extra concurrency and performance, add nodes to Elasticsearch cluster. When you add nodes, your shards are automatically rebalanced across the cluster to spread resource usage across all the nodes and increasing performance.

Memory management

Elasticsearch memory is divided into to categories, JVM heap size and system memory. It uses heap for internal caching, and the rest of the system memory for the operating system, file system caches, and garbage collection. The more memory that is allocated to the heap, the less is allocated to the rest of the system.

Databases for Elasticsearch deployments have their memory allocation policy set at 50% heap and 50% system memory, with a max heap size of 32 GB. In some cases, it is useful to scale your deployment above 64 GB of RAM even with the heap limit as Elasticsearch does make use of the file system cache and alleviate pressure on disk I/O utilization. You can configure autoscaling to increase memory when disk I/O utilization reaches a certain threshold.

Disk IOPS

The number of Input/Output Operations per second (IOPS) is limited by the type of storage volume. Storage volumes for Databases for Elasticsearch deployments are provisioned on Block storage endurance volumes in the 10 IOPS per GB tier. Hitting IOPS limits can cause your databases to respond slowly or appear unresponsive.

Indexing uses disk, so if your use-case is write-heavy, your indexing speed can be limited by the IOPS available to your deployment. Some bottlenecks can be ameliorated by tuning your indexes for disk usage. In addition, searching can use disk if your working data set does not fit in the file system cache, increasing IOPS load. If your use-case involves searching a large data set, increasing the memory on your deployment can help Elasticsearch rely less on disk.

Another thing to note is the default Lucene file system management policy is `niofs`, which allows concurrent reads on a file, which also can be constrained by disk I/O limits. Information on file system storage types in the Elasticsearch documentation.

If you need more IOPS, you can increase the number IOPS available to your deployment by increasing disk space. If you are aware of trends in your indexing or usage that increases disk I/O, you can configure autoscaling to increase disk based on IOPS.

Pricing

Our strategic partnership with Elastic since January 2023 means that we are able to offer more and richer functionality, as well as world-class levels of support.

All Databases for Elasticsearch plans provision as one highly available Elasticsearch cluster with three data members located in three different zones. Your data is replicated across members. Plans are priced based on the total amount of disk storage, RAM, dedicated cores, and backup storage that is allocated to instances, prorated hourly. Databases for Elasticsearch instances have a minimum of 5 GB of disk and 1 GB of RAM per data member.

IBM Cloud® Databases for Elasticsearch Enterprise Plan

The Databases for Elasticsearch Enterprise Plan provisions the Basic version of Elasticsearch. This plan offers features that were previously offered by Elastic as paid-for add-ons under the X-Pack label, such as Role Based Access Control (RBAC) and Index Lifecycle Management (ILM).

Once provisioned, Databases for Elasticsearch Enterprise Plan clusters make full use of other Elastic Stack components such as Kibana, Logstash, and Beats.

IBM Cloud® Databases for Elasticsearch Platinum Plan

The Databases for Elasticsearch Platinum Plan provisions the Platinum version of Elasticsearch. This plan offers all the features of our Enterprise Plan, plus premium features like the ability to run machine learning workloads, including Elastic's own Elastic Learned Sparse Encoder for semantic search. Our Platinum Plan also includes enhanced security features, like document- and field-level security. See a full comparison of Enterprise and Platinum plans at Databases for Elasticsearch Plans.

Using the Pricing Calculator

Prices for the different plans reflect different levels of available functionality. Templates are provided for ease of use and provide balanced resource allocations appropriate for general-purpose workloads. The **Custom** tab can be used to configure Disk, RAM, and vCPU, as wanted. For pricing estimation, use the **Add to Estimate** button on the Databases for Elasticsearch catalog page. Input your total consumption across three data members into the calculator. For example, 5 GB of disk and 1 GB of RAM across three data members would be priced at 15 GB of disk and 3 GB of RAM.

Dedicated Cores Pricing

You have the option of selecting the CPU allocation for your instance. With dedicated cores, your resource group is given a single-tenant host with a guaranteed minimum reserve of cpu shares. Your instances are then allocated the number of CPUs you specify. The cost of dedicated cores is $30 per core per month, and each member gets the selected number of cores. For example, if you provision an instance with 3 dedicated cores per member, that is a total of 9 cores and is billed at $270 per month.

Dedicated cores are an optional feature. The default `Shared CPU` setting provisions your instance on hosts with shared compute resources and incurs no additional charge.

Backups Pricing

You receive your total disk space purchased, per database, in free backup storage. For example, in a given month, if your Databases for Elasticsearch instance has 20 GB of disk per member and three data members, you receive 60 GB of backup storage free for that month. If your backup storage utilization is greater than 60 GB for the month (in this scenario), you are charged an overage of $0.03/month per gigabyte.

By default, Cloud Databases provides a daily backup that is stored for 30 days. These backups, and any on-demand backups you make, all count toward the above allocation.

In the above example, if your database contains 2 GB of data and you have not taken any on-demand backups, then your total backup size is 2 GB x 30 = 60 GB. Your backup costs are nil.

If your database contains 15 GB of data and you have not taken any on-demand backups, then your total backup size is 15 GB x 30 = 450 GB. In this scenario, your backup costs are (450 GB - 60 GB) * 0.03 = $11.7 per month.

Most instances will not ever go over the allotted credit.

Scaling per Member

Databases for Elasticsearch instances have minimum and maximum allocation for disk and RAM as shown. Scaling instances through the API/CLI provides more granularity and also allows a user to scale a database instance up to 4 TB of disk per member.

Resource	Minimum	Maximum	Scaling Granularity (API/CLI)
Disk	5 GB per member	4 TB per member	1024 MB per member
RAM	1 GB per member	112 GB per member	128 MB per member
CPU (if enabled)	3 CPUs per member	28 CPUs per member	1 CPU per member

Per Member Scaling Limits

Release notes

Use these release notes to learn about the latest updates to IBM Cloud® Databases for Elasticsearch that are grouped by *date* or *build number*.

20 November 2025

Databases for Elasticsearch version policy changes starting version v8.19

To align with the Elasticsearch vendor lifecylce policy, Databases for Elasticsearch (an IBM Cloud® managed service) is updating its version lifecycle policies starting with Elasticsearch v8.19.

- Five weeks advance notification of version deprecation will be provided for a minor and major version.
- Databases for Elasticsearch will support only three versions, the latest, the latest -1 and the latest -2, outside of the few weeks period (minimum 2 weeks) during which you should upgrade from the version being deprecated to the new version.
- Instances that are using a deprecated version will be ‘automatically force upgraded’ to the next major version after the EOL date. However, no SLA will be provided for this upgrade method.

Note: We do not recommend that you wait until the end-of-life date for the following reasons:

- We provide no SLAs for this type of forced upgrade.
- You may experience some data loss.
- Your application may experience downtime.
- Your application may stop working if it has any incompatibilities with the new database version.
- You cannot control the timing of when this upgrade will happen for your deployment.
- There is no rollback process for this forced upgrade.
- For more information on upcoming version EOL, see the Cloud Databases version policy page.

Customer impact: Frequent version upgrades are expected with a shorter 5-weeks timeline to upgrade. Automatic forced upgrade will occur on EOL date.

Customer action needed: You are expected to perform major version upgrade using backup and restore before EOL date to align with the updated policy and to be on a supported version of the service.

11 November 2025

Introducing Databases for Elasticsearch v8.19

Databases for Elasticsearch v8.19 is now GA. This release includes performance improvements, new features, and bug fixes. Version 8.19 also introduces mark token pruning for sparse vectors (GA) and cross-cluster querying in Elasticsearch SQL (GA), enabling more efficient search and enhanced analytics across clusters. For more information, see the Elasticsearch 8.19 release notes and Highlights.

Introducing Databases for Elasticsearch v9.1

Databases for Elasticsearch v9.1 is now GA. This release builds on the 8.x series with new capabilities, optimizations, and fixes, bringing more scalability, and advanced search enhancements. For more information, see the Elasticsearch 9.1 release notes.

9 September 2025

Introducing Databases for Elasticsearch v8.19 (Preview)

Databases for Elasticsearch v8.19 is now available in Preview. This release includes performance improvements, new features, and bug fixes. Version 8.19 also introduces mark token pruning for sparse vectors (GA) and cross-cluster querying in Elasticsearch SQL (GA), enabling more

efficient search and enhanced analytics across clusters. For more information, see the Elasticsearch 8.19 release notes and Highlights.

Introducing Databases for Elasticsearch v9.1 (Preview)

Databases for Elasticsearch v9.1 is now available in Preview. This release builds on the 8.x series with new capabilities, optimizations, and fixes, bringing more scalability, and advanced search enhancements. For more information, see the Elasticsearch 9.1 release notes.

10 March 2025

Databases for Elasticsearch Satellite plan is deprecated

IBM Cloud Satellite® is now deprecated due to changes in market expectations, client fit, and lack of adoption. As of March 10 2025, all documentation relating to Satellite has been removed, as well as the ability to select Databases for Elasticsearch Satellite plan in the Cloud console.

15 November 2024

Cloud Databases logs and events are now available on IBM® Cloud Logs

Cloud Databases has onboarded IBM Cloud Logs, a scalable logging service that persists logs and provides users with capabilities for querying, tailing, and visualizing logs. Customers are expected to use IBM Cloud Logs to review their database logs and events starting **November 15, 2024**. For more information, see Set up logging and monitoring and About IBM Cloud Logs.

16 September 2024

Private endpoints as new default

To ensure best possible security for your databases, private endpoints are now the default in the IBM Cloud® console. CLI and Terraform now require the endpoint type to be provided as part of creating an instance.

1 May 2024

New hosting models

You can choose between two hosting models: Isolated Compute and Shared Compute. Isolated Compute is a secure single-tenant offering for complex, highly performant enterprise workloads. Shared Compute is a flexible multi-tenant offering for dynamic, fine-tuned, and decoupled capacity selections. For more information, see Hosting models.

12 April 2024

Configuring the Index Lifecycle Management capabilities of Databases for Elasticsearch tutorial published

This tutorial is designed to assist you to proactively manage your indices to make efficient use of resources, both in terms of storage and search capabilities. For more information, see Configuring the Index Lifecycle Management capabilities of Databases for Elasticsearch.

06 February 2024

Build an Elasticsearch chatbot tutorial published

This tutorial is designed to assist you in harnessing the full potential of your data by transforming it into an intelligent bot capable of answering queries related to your data. The bot effortlessly stores the knowledge base in an Elasticsearch index and taps into AI to intelligently handle your questions related to the knowledge base. For more information, see Build an Elasticsearch chatbot.

17 January 2024

Elasticsearch machine learning tutorials documentation published

New tutorials created to highlight the ML features of Elasticsearch. For more information, see Use Elasticsearch vector search capabilities, Use ELSER, Elastic's Natural Language Processing model, and Use machine learning models with Elasticsearch to tag content.

03 January 2024

Elasticsearch, machine learning, and AI

Elasticsearch, a robust search-capable database, is well-suited to seamlessly incorporate essential ML features sought by enterprises embracing new technologies. For more information, see Elasticsearch, machine learning, and AI.

22 November 2023

Monitoring Integration documentation updated

Monitoring Integration documentation now lists metrics for all Cloud Databases services. For more information, see Monitoring Integration.

15 November 2023

Databases for Elasticsearch Platinum Plan released

Databases for Elasticsearch Platinum Plan offers all the features of the Enterprise Plan, plus a richer array of functionality around integrations (connectors and web crawlers), security features (logging and access control), and document-level security, among others. For more information, see Databases for Elasticsearch Plans.

26 October 2023

Deploy Kibana using Code Engine and connect to your Databases for Elasticsearch instance

With this tutorial, deploy Kibana using Code Engine and connect to your Databases for Elasticsearch instance. Kibana is a web interface that allows you to visualize the data in Elasticsearch instances. Code Engine is a fully managed, serverless platform that allows you to run workloads without worrying about deploying infrastructure. Elasticsearch is a NoSQL database with powerful search capabilities. For more information, see Deploy Kibana using Code Engine and connect to your Databases for Elasticsearch instance.

12 October 2023

Databases for Elasticsearch v7.17 end of life

End of life announcement: Version 7.17 reaches end of life 26 April 2024. All Databases for Elasticsearch instances on deprecated versions that are still active will be upgraded in-place to the next major version. We recommend that you upgrade following our backup and restore process before

the EOL date of your version.

For more information, see Upgrading to a new Major Version.

02 October 2023

Elasticsearch end of life for versions 7.9 and 7.10

Action is required before 30 November 2023 for your IBM Cloud® Databases for Elasticsearch version 7.9 and 7.10 deployments. On 30 November 2023, these versions will be end of life. Our recommended procedure for this is restoring from a backup. For more information, see Managing Cloud Databases backups.

18 September 2023

Configuring an Enterprise Search 7.17 server with an Databases for Elasticsearch instance

Enterprise Search extends the capabilities of Databases for Elasticsearch to provide a unified search experience across various data sources, including documents, emails, databases, and more. It offers a seamless interface for users to find relevant information across disparate data silos, enhancing productivity and collaboration. By integrating Enterprise Search with your Databases for Elasticsearch instance, you gain a comprehensive search solution that uses the strengths of both platforms to efficiently discover insights from your data. For more information, see Configuring an Enterprise Search 7.17 server with an Databases for Elasticsearch instance.

22 June 2023

Elasticsearch 8.7.0 (preview)

For more information, see IBM Cloud® Databases for Elasticsearch Enterprise Plan.

18 May 2023

Setting up disk alerts for disk utilization tutorial

In this tutorial, you use the IBM Cloud API and the IBM Cloud CLI to set up an alert that emails you whenever the disk utilization of your database exceeds 90%. This specific example creates an alert on a Databases for Elasticsearch deployment, but it is applicable to all the databases in the IBM Cloud Databases catalog. For more information, see Setting up disk alerts for disk utilization.

03 May 2023

IBM Cloud® Databases for Elasticsearch Enterprise Plan

Our strategic partnership with Elastic since January 2023 means that we are able to offer more and richer functionality, as well as world-class levels of support. IBM Cloud® Databases for Elasticsearch Enterprise Plan Version 7.17 is the current supported version of Databases for Elasticsearch and is offered under our Enterprise Plan. This version includes functionality that is not available in our Standard plan. IBM Cloud® Databases for Elasticsearch Standard Plan Versions 7.9 and 7.10 are available on Cloud Databases under the Standard Plan, but will be End of Life (EOL) by 30 November 2023. For more information, see Databases for Elasticsearch Pricing.

19 October 2022

Deploying and Connecting a Cloud Databases Instance Tutorial

This tutorial guides you through the process of deploying a Cloud Databases instance and connecting it to a web front end by creating a webpage that allows visitors to input a word and its definition. These values are then stored in a database running on Cloud Databases. You install the database infrastructure by using Terraform and your web application uses the popular Express framework. The application can then be run locally, or by using Docker. For more information, see Deploying and Connecting a Cloud Databases Instance .

11 October 2022

Protecting IBM Cloud® Databases for Elasticsearch resources with context-based restrictions

Context-based restrictions (CBR) give account owners and administrators the ability to define and enforce access restrictions for IBM Cloud® resources based on the context of access requests. Access to Cloud Databases resources can be controlled with CBR and identity and access management (IAM) policies. For more information, see Protecting Cloud Databases resources with context-based restrictions .

25 March 2022

IBM Cloud® Databases for Elasticsearch Security Goals Updated

Available security and compliance goals updated. See updated goals here.

08 December 2020

IBM Cloud® Databases for Elasticsearch 6 End of Life

On April 29, 2021, all IBM Cloud® Databases for Elasticsearch instances on version 6.x that are still active will be disabled.

09 November 2020

IBM Cloud® Databases for Elasticsearch v7.9 is now available

We are pleased to announce the release of version 7.9 on IBM Cloud® Databases for Elasticsearch.

13 April 2020

IBM Cloud® Databases for Elasticsearch autoscaling

We are excited to announce that autoscaling of your deployments based on disk capacity and disk I/O utilization is now available for IBM Cloud® Databases for Elasticsearch via the UI, API, and CLI.

20 February 2020

IBM Cloud® Databases for Elasticsearch customers are now able to scale their IBM Cloud® Databases for Elasticsearch instances up to 20 members.

This new feature allows Elasticsearch instances to now grow to 2.2 TB RAM, 80 TB of Disk, and 560 vCPUs.

6 August 2019

New Regions Available for IBM Cloud Database Services

IBM Cloud® Databases for Elasticsearch is now available to be deployed in Seoul; South Korea; and Chennai, India.

19 December 2018

General Availability of IBM Cloud® Databases for Elasticsearch

IBM Cloud® Databases for Elasticsearch added to the Cloud Databases family.

Provisioning

Provision a IBM Cloud® Databases for Elasticsearch deployment through the catalog, the Cloud Databases CLI plug-in, the Cloud Databases API, through Terraform, or through pre-built, open-source, and enterprise-ready Terraform IBM Modules (TIM).

Provisioning through the IBM Cloud console

Provision from the console by specifying the following parameters.

Service details

- **Service name** - The name can be any string and is the name that is used on the web and in the CLI to identify the new deployment.
- **Resource group** - If you are organizing your services into resource groups, specify the resource group in this field. Otherwise, you can leave it at default. For more information, see Managing resource groups.
- **Location** - The deployment's public cloud region.

Hosting model

- **Isolated:** Secure single-tenant offering for complex, highly-performant enterprise workloads.
- **Shared:** Flexible multi-tenant offering for dynamic, fine-tuned, and decoupled capacity selections.

> **Note:** Databases for Elasticsearch Platinum features are only available on Isolated Compute.

For more information, see Hosting models and Databases for Elasticsearch plans.

Resource allocation

Fine tune your resource allocation. The available options differ based on your selected hosting model.

- **Isolated:** Use the table to choose the machine size for each member of your deployment, and specify the disk size.
- **Shared:** By default, the smallest possible resource allocation is selected. This is ideal for small applications or testing. For larger allocations, select the *Custom* tile, which allows flexible resource configuration with 2+ cores.

> **Note:** The Shared Compute hosting model supports more fine-grained resource allocations that are not shown in the UI to maintain clarity. For more information, see Hosting models.

> **Note:** Specify the disk size depending on your requirements. It can be increased after provisioning but cannot be decreased to prevent data loss.

Service configuration

- **Database version:** Set only at deployment The deployment version of your database. To ensure optimal performance, run the preferred version. The latest minor version is used automatically. For more information, see Versioning policy.
- **Database edition:** Set only at deployment Select the edition that you want to deploy. Choose from Enterprise or Platinum. Note that Platinum is only available on the Isolated hosting model.
- **Encryption:** Set only at deployment If you use Key Protect, an instance and key can be selected to encrypt the deployment's disk. If you do not use your own key, the deployment automatically creates and manages its own disk encryption key.
- **Endpoints:** Set only at deployment Configure the Service endpoints on your deployment. The default setting is *private*.

> **Note:** A Databases for Elasticsearch deployment cannot have both public and private endpoints simultaneously.

After you select the appropriate settings, click **Create** to start the provisioning process.

Provisioning through the CLI

Create a service instance through the CLI

Before provisioning, follow the instructions provided in the documentation to install the IBM Cloud CLI tool.

1. Log in to IBM Cloud. If you use a federated user ID, it's important that you switch to a one-time passcode (`ibmcloud login --sso`), or use an API key (`ibmcloud --apikey key or @key_file`) to authenticate. For more information about how to log in by using the CLI, see General CLI (ibmcloud) commands under `ibmcloud login` .

   ```
   $ ibmcloud login
   ```

2. Select the hosting model you want your database to be provisioned on. You can change this later.

3. Provision your database with the following command:

   ```
   $ ibmcloud resource service-instance-create <INSTANCE_NAME> <SERVICE_NAME> <SERVICE_PLAN_NAME> <LOCATION> <RESOURCE_GROUP> -p
   '{"members_host_flavor": "<members_host_flavor value>"}' --service-endpoints="<Endpoint>"
   ```

   For example, to provision a Databases for Elasticsearch Shared Compute hosting model instance, use a command like:

   ```
   $ ibmcloud resource service-instance-create test-database databases-for-elasticsearch enterprise us-south -p '{ "backup_id": "<BACKUP_CRN>", "version": "
   <VERSION_STRING>", "members_memory_allocation_mb": "4096" }' --service-endpoints="private"
   ```

   Provision a Databases for Elasticsearch Isolated instance with the same `"members_host_flavor"` -p parameter, setting it to the desired Isolated size. Available hosting sizes and their `members_host_flavor value` parameters are listed in Table 2. For example, `{"members_host_flavor": "b3c.4x16.encrypted"}` . Note that since the host flavor selection includes CPU and RAM sizes (`b3c.4x16.encrypted` is 4 CPU and 16 RAM), this request does not accept both, an Isolated size selection and separate CPU and RAM allocation selections.

   ```
   $ ibmcloud resource service-instance-create test-database databases-for-elasticsearch enterprise us-south -p '{"members_host_flavor": "b3c.4x16.encrypted"}' --service-
   endpoints="private"
   ```

   The fields in the command are described in the table that follows.

Field	Description	Flag
INSTANCE_NAME Required	The instance name can be any string and is the name that is used on the web and in the CLI to identify the new deployment.	
SERVICE_NAME Required	Name or ID of the service. For Databases for Elasticsearch, use `databases-for-elasticsearch`.	
SERVICE_PLAN_NAME Required	`enterprise` or `platinum`. Note that `platinum` requires an `isolated` host flavor.	
LOCATION Required	The location where you want to deploy. To retrieve a list of regions, use the `ibmcloud regions` command.	
RESOURCE_GROUP	The Resource group name. The default value is `default`.	-g
--parameters	JSON file or JSON string of parameters to create service instance	-p
members_host_flavor	To provision an Isolated or Shared Compute instance, use `{"members_host_flavor": "<members_host_flavor value>"}`. For Shared Compute, specify a value of `multitenant`. For Isolated Compute, select desired CPU and RAM configuration. For more information, see the following table or Hosting models.	
--service-endpoints Required	Configure the Service endpoints of your deployment, either `public`, `private` or `public-and-private`.	

Basic command format fields

Note: In the CLI, `service-endpoints` is a flag, not a parameter.

The `members host flavor` parameter

The `members_host_flavor` parameter defines your Compute sizing.

To provision a Shared Compute instance, specify `multitenant`. To provision an Isolated Compute instance, input the appropriate value for your desired CPU and RAM configuration.

Members Host flavor	members_host_flavor value
Shared Compute	`multitenant`
4 CPU x 16 RAM	`b3c.4x16.encrypted`
8 CPU x 32 RAM	`b3c.8x32.encrypted`
8 CPU x 64 RAM	`m3c.8x64.encrypted`
16 CPU x 64 RAM	`b3c.16x64.encrypted`
32 CPU x 128 RAM	`b3c.32x128.encrypted`
30 CPU x 240 RAM	`m3c.30x240.encrypted`

Host flavor sizing parameter

You will see a response like:

$ ```text {: codeblock}
Creating service instance INSTANCE_NAME in resource group default of account USER...
OK
Service instance INSTANCE_NAME was created.
Name: INSTANCE_NAME
ID: crn:v1:bluemix:public:databases-for-elasticsearch:us-south:a/ 40ddc34a846383BGB5b60e:dd13152c-fe15-4bb6-af94-fde0af5303f4::
GUID: dd13152c-fe15-4bb6-af94-fde0af56897
Location: LOCATION
State: provisioning
Type: service_instance
Sub Type: Public
Service Endpoints: private
Allow Cleanup: false
Locked: false
Created at: 2023-06-26T19:42:07Z
Updated at: 2023-06-26T19:42:07Z
Last Operation:
  Status create in progress
  Message Started create instance operation
```
- To check provisioning status, use the following command:

```sh {: pre}
ibmcloud resource service-instance <INSTANCE_NAME>
```
When complete, you will see a response like:

```text {: codeblock}
Retrieving service instance INSTANCE_NAME in resource group default under account USER's Account as USER...
OK

Name:  INSTANCE_NAME
ID:  crn:v1:bluemix:public:databases-for-elasticsearch:us-south:a/40ddc34a953a8c02f109835656860e:dd13152c-fe15-4bb6-af94-fde0af5303f4::
GUID:  dd13152c-fe15-4bb6-af94-fde5654765
Location:  <LOCATION>
Service Name:  databases-for-elasticsearch
Service Plan Name:  standard
Resource Group Name:  default
State:  active
Type:  service_instance
Sub Type:  Public
Locked:  false
Service Endpoints:  private
Created at:  2023-06-26T19:42:07Z
```
```

```
Created by:          USER
Updated at:          2023-06-26T19:53:25Z
Last Operation:
                     Status    create succeeded
                     Message   Provisioning elasticsearch with version 7.17 (100%)
```
- Optional: To delete a service instance, run the following command:

```sh {: pre}
ibmcloud resource service-instance-delete <INSTANCE_NAME_OR_CRN>
```

Note: CPU and RAM autoscaling is not supported on Cloud Databases Isolated Compute. Disk autoscaling is available. If you have provisioned an Isolated instance or switched over from a deployment with autoscaling, keep an eye on your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

The --parameters parameter

The service-instance-create command supports a -p parameter, which allows JSON-formatted parameters to be passed to the provisioning process. For example, you can pass Cloud Resource Names (CRNs) as parameter values, which uniquely identify a resource in the cloud. All parameter names and values are passed as strings.

For example, if a database is being provisioned from a particular backup and the new database deployment needs a total of 12 GB of memory across three members, then the command to provision 4 GBs per member looks like:

```
$ ibmcloud resource service-instance-create databases-for-elasticsearch <INSTANCE_NAME> enterprise us-south \
-p \ '{
  "backup_id": "crn:v1:blue:public:databases-for-elasticsearch:us-south:a/54e8ffe85dcedf470db5b5ee6ac4a8d8:1b8f53db-fc2d-4e24-8470-f82b15c71717:backup:06392e97-df90-46d8-98e8-cb67e9e0a8e6",
  "members_memory_allocation_mb": "4096"
}' --service-endpoints="private"
```

Provisioning through the Resource Controller API

Follow these steps to provision by using the Resource Controller API.

1. Obtain an IAM token from your API token.
2. You need to know the ID of the resource group that you would like to deploy to. This information is available through the IBM Cloud CLI.

   Use a command like:

   ```
   $ ibmcloud resource groups
   ```

3. You need to know the region you want to deploy to.

   To list all of the regions that deployments can be provisioned into from the current region, use the Cloud Databases CLI plug-in.

   The command looks like:

   ```
   $ ibmcloud cdb regions --json
   ```

4. Select the hosting model you want your database to be provisioned on. You can change this later.

   A host flavor represents fixed sizes of guaranteed resource allocations. To see which host flavors are available in your region, call the host flavors capability endpoint like this:

   ```
   $ curl -X POST  https://api.{region}.databases.cloud.ibm.com/v5/ibm/capability/flavors  \
    -H 'Authorization: Bearer <>' \
    -H 'ContentType: application/json' \
    -d '{
      "deployment": {
        "type": "elasticsearch",
        "location": "us-south"
      }
    }'
   ```

This returns:


```
$ {
  "deployment": {
    "type": "elasticsearch",
    "location": "us-south",
    "platform": "classic"
  },
  "capability": {
    "flavors": [
      {
        "id": "b3c.4x16.encrypted",
        "name": "4x16",
        "cpu": {
          "allocation_count": 4
        },
        "memory": {
          "allocation_mb": 16384
        },
        "hosting_size": "xs"
      },
      {
    "id": "b3c.8x32.encrypted",
    "name": "8x32",
    "cpu": {
      "allocation_count": 8
    },
    "memory": {
      "allocation_mb": 32768
    },
    "hosting_size": "s"
  },
  {
    "id": "m3c.8x64.encrypted",
    "name": "8x64",
    "cpu": {
      "allocation_count": 8
    },
    "memory": {
      "allocation_mb": 65536
    },
    "hosting_size": "s+"
  },
  {
    "id": "b3c.16x64.encrypted",
    "name": "16x64",
    "cpu": {
      "allocation_count": 16
    },
    "memory": {
      "allocation_mb": 65536
    },
    "hosting_size": "m"
  },
  {
    "id": "b3c.32x128.encrypted",
    "name": "32x128",
    "cpu": {
      "allocation_count": 32
    },
    "memory": {
      "allocation_mb": 131072
    },
    "hosting_size": "l"
  },
  {
    "id": "m3c.30x240.encrypted",
    "name": "30x240",
    "cpu": {
      "allocation_count": 30
    },
```

```
      "memory": {
        "allocation_mb": 245760
      },
      "hosting_size": "xl"
    },
    {
      "id": "multitenant",
      "name": "multitenant",
      "cpu": {
        "allocation_count": 0
      },
      "memory": {
        "allocation_mb": 0
      },
      "hosting_size": ""
    }
  ]
 }
}
```

As shown, the Isolated Compute host flavors available to a Databases for Elasticsearch instance in the `us-south` region are:

- `b3c.4x16.encrypted`
- `b3c.8x32.encrypted`
- `m3c.8x64.encrypted`
- `b3c.16x64.encrypted`
- `b3c.32x128.encrypted`
- `m3c.30x240.encrypted`

To provision or scale your instance to 4 CPUs and `16384` megabytes or RAM, submit the following command:

```
$ {
  "parameters": {
    "members_host_flavor": "`b3c.4x16.encrypted`"
  }
}
```

To scale your instance up to 8 CPUs and `32768` megabytes of RAM, submit the following command:

```
$ {
  "parameters": {
    "members_host_flavor": "b3c.8x32.encrypted"
  }
}
```

5. Once you have all the information, [provision a new resource instance] with the IBM Cloud Resource Controller.

```
$ curl -X POST \
  https://resource-controller.cloud.ibm.com/v2/resource_instances \
  -H "Authorization: Bearer <TOKEN>" \
  -H 'Content-Type: application/json' \
    -d '{
    "name": "<INSTANCE_NAME>",
    "target": "<targeted-region>",
    "resource_group": "RESOURCE_GROUP_ID",
    "resource_plan_id": "<SERVICE_PLAN_NAME>"
    "parameters": {
        "members_host_flavor": "<members_host_flavor_value>",
        "service_endpoints":" <ENDPOINT>",
        "version":"<VERSION>"
    }
  }'
```

To make a Shared Compute instance, follow the following example. Please note that Premium verisons of Elasticsearch are not available on Shared Compute.

```
$ curl -X POST \
```

```
https://resource-controller.cloud.ibm.com/v2/resource_instances \
-H "Authorization: Bearer <TOKEN>" \
-H 'Content-Type: application/json' \
  -d '{ \
  "name": "my-instance", \
  "target": "us-south", \
  "resource_group": "<RESOURCE_GROUP_ID>", \
  "resource_plan_id": "<SERVICE_PLAN_NAME>", \
  "parameters": {
    "members_host_flavor": "multitenant",
    "service_endpoints": "private",
    "members_memory_allocation_mb": 16384,
    "members_cpu_allocation_count": 4
  } \
}' \
```

Provision a Databases for Elasticsearch Isolated instance with the same `"members_host_flavor"` parameter, setting it to the desired Isolated size. Available hosting sizes and their `members_host_flavor value` parameters are listed in Table 2. For example, `{"members_host_flavor": "b3c.4x16.encrypted"}`. Note that since the host flavor selection includes CPU and RAM sizes (`b3c.4x16.encrypted` is 4 CPU and 16 RAM), this request does not accept both, an Isolated size selection and separate CPU and RAM allocation selections.

```
$ curl -X POST \
  https://resource-controller.cloud.ibm.com/v2/resource_instances \
  -H "Authorization: Bearer <TOKEN>" \
  -H 'Content-Type: application/json' \
    -d '{ \
    "name": "my-instance", \
    "target": "us-south", \
    "resource_group": "5g9f447903254bb58972a2f3f5a4c711", \
    "resource_plan_id": "<SERVICE_PLAN_NAME>", \
    "parameters": {
    "members_host_flavor": "b3c.4x16.encrypted",
    "service_endpoints": "private"
    } \
  }' \
```

The parameters `name`, `target`, `resource_group`, `resource_plan_id`, and `service_endpoints` are all required.

The fields in the command are described in the table that follows.

Field	Description	Flag
NAME Required	The instance name can be any string and is the name that is used on the web and in the CLI to identify the new deployment.	
SERVICE_NAME Required	Name or ID of the service. For Databases for Elasticsearch, use `databases-for-elasticsearch`.	
SERVICE_PLAN_NAME Required	`enterprise` or `platinum`	
TARGET Required	The region where you want to deploy. To retrieve a list of regions, use the `ibmcloud regions` command.	
SERVICE_ENDPOINTS_TYPE	Configure the Service endpoints of your deployment, either `public` or `private`. The default value is `public`.	
RESOURCE_GROUP	The Resource group name. The default value is `default`.	-g
--parameters	JSON file or JSON string of parameters to create service instance	-p
members_host_flavor	To provision an Isolated or Shared Compute instance, use `{"members_host_flavor": "<members_host_flavor value>"}`. For Shared Compute, specify a value of `multitenant`. For Isolated Compute, select desired CPU and RAM configuration. For more information, see the table below, or Hosting models.	
service-endpoints Required	Configure the Service endpoints of your deployment, either `public`, `private` or `public-and-private`.	

Basic command format fields

The `members host flavor` parameter

The `members_host_flavor` parameter defines your Compute sizing. To provision a Shared Compute instance, specify `multitenant`. To provision an Isolated Compute instance, input the appropriate value for your desired CPU and RAM configuration.

Members host flavor	members_host_flavor value
Shared Compute	`multitenant`
4 CPU x 16 RAM	`b3c.4x16.encrypted`
8 CPU x 32 RAM	`b3c.8x32.encrypted`
8 CPU x 64 RAM	`m3c.8x64.encrypted`
16 CPU x 64 RAM	`b3c.16x64.encrypted`
32 CPU x 128 RAM	`b3c.32x128.encrypted`
30 CPU x 240 RAM	`m3c.30x240.encrypted`

Host flavor sizing parameter

Note: CPU and RAM autoscaling is not supported on Cloud Databases Isolated Compute. Disk autoscaling is available. If you have provisioned an Isolated instance or switched over from a deployment with autoscaling, keep an eye on your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

List of additional parameters

In the `--parameters` object you can provide additional information to create your service instance, including:

- `backup_id` - A CRN of a backup resource to restore from. The backup must be created by a database deployment with the same service ID. The backup is loaded after provisioning and the new deployment starts up that uses that data. A backup CRN is in the format `crn:v1:<...>:backup:<uuid>`. If omitted, the database is provisioned empty.
- `version` - The version of the database to be provisioned. If omitted, the database is created with the most recent major and minor version.
- `disk_encryption_key_crn` - The CRN of a KMS key (for example, Hyper Protect Crypto Services or Key Protect), which is then used for disk encryption. A KMS key CRN is in the format `crn:v1:<...>:key:<id>`.
- `backup_encryption_key_crn` - The CRN of a KMS key (for example, Hyper Protect Crypto Services or Key Protect), which is then used for backup encryption. A KMS key CRN is in the format `crn:v1:<...>:key:<id>`.

  Note: To use a key for your backups, you must first enable the service-to-service delegation.

- `members_memory_allocation_mb` - Total amount of memory to be shared between the database members within the database. For example, if the value is "12288", and there are three database members, then the deployment gets 12 GB of RAM total, giving 4 GB of RAM per member. If omitted, the default value is used for the database type is used. This parameter only applies to `multitenant'.
- `members_disk_allocation_mb` - Total amount of disk to be shared between the database members within the database. For example, if the value is "30720", and there are three members, then the deployment gets 30 GB of disk total, giving 10 GB of disk per member. If omitted, the default value for the database type is used. This parameter only applies to `multitenant'.
- `members_cpu_allocation_count` - Enables and allocates the number of specified cores to your deployment. For example, to use two dedicated cores per member, use `"members_cpu_allocation_count":"2"`. If omitted, the default Shared Compute CPU:RAM ratios will be applied. This parameter only applies to `multitenant'.

Provisioning with Terraform

Use Terraform to manage your infrastructure through the `ibm_database` Resource for Terraform supports provisioning Cloud Databases deployments.

Alternatively, you can use Terraform IBM Modules to manage your infrastructure through Terraform IBM Modules for Databases for Elasticsearch.

Select the hosting model you want your database to be provisioned on. You can change this later.

Provisioning Shared Compute with Terraform

Provision a Databases for Elasticsearch Shared hosting model instance with the `"host_flavor"` parameter set to `multitenant`. See the following example:

```
data "ibm_resource_group" "group" {
  name = "<your_group>"
}
resource "ibm_database" "<your_database>" {
  name              = "<your_database_name>"
  plan              = "standard"
  location          = "eu-gb"
  service           = "databases-for-elasticsearch"
  resource_group_id = data.ibm_resource_group.group.id
  service_endpoints = "private"
  tags              = ["tag1", "tag2"]
  adminpassword     = "password12"
  # Pin the database version at provision time.
  # If omitted, the newest available major/minor is selected (per service behavior).
  version = "<VERSION_STRING>"              # e.g. "8.19" for Elasticsearch preferred major
  group {
    group_id = "member"
    host_flavor {
      id = "multitenant"
    },
    cpu {
      allocation_count = 6
    }
    memory {
      allocation_mb = 24576
    }
    disk {
      allocation_mb = 256000
    }
  }
  users {
    name     = "user123"
    password = "password12"
  }
  allowlist {
    address     = "172.168.1.1/32"
    description = "desc"
  }
}
output "ICD Elasticsearch database connection string" {
  value = "http://${ibm_database.test_acc.ibm_database_connection.icd_conn}"
}
```

Provisioning Isolated Compute with Terraform

Provision a Databases for Elasticsearch Isolated instance with the same `"host_flavor"` parameter, setting it to the desired Isolated size. Available hosting sizes and their `host_flavor value` parameters are listed in Table 1. For example, `{"host_flavor": "b3c.4x16.encrypted"}`. Note that since the host flavor selection includes CPU and RAM sizes (`b3c.4x16.encrypted` is 4 CPU and 16 RAM), this request does not accept both, an Isolated size selection and separate CPU and RAM allocation selections.

```
data "ibm_resource_group" "group" {
  name = "<your_group>"
}
resource "ibm_database" "<your_database>" {
  name              = "<your_database_name>"
  plan              = "standard"
  location          = "eu-gb"
  service           = "databases-for-elasticsearch"
  resource_group_id = data.ibm_resource_group.group.id
  service_endpoints = "private"
  tags              = ["tag1", "tag2"]
```

```
  adminpassword     = "password12"
  # Pin the database version at provision time.
  # If omitted, the newest available major/minor is selected (per service behavior).
  version = "<VERSION_STRING>"                # e.g. "8.19" for Elasticsearch preferred major
  group {
    group_id = "member"
    host_flavor {
      id = "b3c.8x32.encrypted"
    }
    disk {
      allocation_mb = 256000
    }
  }
  users {
    name     = "user123"
    password = "password12"
  }
  allowlist {
    address     = "172.168.1.1/32"
    description = "desc"
  }
}
output "ICD Elasticsearch database connection string" {
  value = "http://${ibm_database.test_acc.ibm_database_connection.icd_conn}"
}
```

The `host flavor` parameter

The `host_flavor` parameter defines your Compute sizing.

- **Shared compute** - To provision a Shared Compute instance, specify `multitenant`.
- **Isolated compute** - To provision an Isolated Compute instance, input the appropriate value for your desired CPU and RAM configuration. See the values in the following table.

Host flavor	host_flavor value
Shared compute	`multitenant`
4 CPU x 16 RAM	`b3c.4x16.encrypted`
8 CPU x 32 RAM	`b3c.8x32.encrypted`
8 CPU x 64 RAM	`m3c.8x64.encrypted`
16 CPU x 64 RAM	`b3c.16x64.encrypted`
32 CPU x 128 RAM	`b3c.32x128.encrypted`
30 CPU x 240 RAM	`m3c.30x240.encrypted`

Host flavor sizing parameter

Note: CPU and RAM autoscaling is not supported on Cloud Databases Isolated Compute. Disk autoscaling is available. If you have provisioned an Isolated instance or switched over from a deployment with autoscaling, keep an eye on your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

Working with the Databases for Elasticsearch deployable architecture

IBM Cloud® Databases for Elasticsearch is available as a deployable architecture called Cloud automation for Databases for Elasticsearch in the catalog, so that you can use Elasticsearch as an add-on or building block for the solutions that you build and deploy by using the automation available by managing your resource deployments with IBM Cloud *projects*. Projects are used to manage related resources and deployments across accounts, embracing an Infrastructure as Code (IaC) approach to deployments.

> **Note:** If you are not managing your resource deployments with projects, you can deploy Databases for Elasticsearch directly from the catalog. For more information, see Creating a Databases for Elasticsearch instance.

What is Cloud automation for Databases for Elasticsearch?

Using the Databases for Elasticsearch deployable architecture enables your team to efficiently deploy and manage a fully operational Elasticsearch instance in IBM Cloud®, providing scalable, secure, and optimized search capabilities for your applications.

This deployable architecture is designed to showcase a fully configured Elasticsearch instance, leveraging IBM Cloud's integrated security and scalability features. With support for encrypted data and optional autoscaling, it ensures that your Elasticsearch deployment can meet dynamic workloads while maintaining data security.

This architecture simplifies your organization's search and analytics processes and enhances the reliability of Elasticsearch deployments.

With this architecture, you can:

Centralize search and analytics

: The Elasticsearch instance enables your team to store, search, and analyze large datasets in a centralized location, ensuring quick and reliable access to insights.

Secure data with key management system (KMS) encryption

: Using a pre-configured KMS key, your Elasticsearch data is encrypted, safeguarding it against unauthorized access and ensuring compliance with security standards.

Features and capability

- Deploys and configures an Databases for Elasticsearch instance.
- Integrates with IBM Key Management Service (KMS) for encrypted data storage.
- Supports optional autoscaling rules to optimize resource usage.
- Provides an optional Kibana dashboard for enhanced data visualization and interaction with Elasticsearch data.

Deploying Cloud automation for Databases for Elasticsearch with projects

1. From the IBM Cloud catalog, search for Cloud automation for Databases for Elasticsearch.
2. Add it to an existing project or create a project.
3. Complete the next steps depending on how you plan to use the deployable architecture:
    - Configure it in your project and deploy.
    - You can stack deployable architectures together in a project to create a robust end-to-end solution architecture. You don't need to code Terraform to connect the member deployable architectures within the stack. As you configure input values in a member deployable architecture, you can reference inputs or outputs from another member to link the deployable architectures together. After you deploy the deployable architectures in your stack, you can add the stack to a private catalog to easily share it with others in your organization.

Resources for working with Cloud automation for Databases for Elasticsearch

When using Databases for Elasticsearch after it's deployed, there are some differences regarding responsibilities and managing your deployment through projects. Use the following resources to learn more.

- After you deploy Cloud automation for Databases for Elasticsearch from a project, you can use the Troubleshooting for projects documentation.
- Review the Understanding your responsibilities when you use IBM deployable architectures to learn about the management responsibilities and terms and conditions that you have when you use IBM deployable architectures.

Managing users and roles

Databases for Elasticsearch deployments come with authentication enabled and use Elasticsearch's Built-in users.

Add users in the UI on the *Service credentials* page, with the Cloud Databases CLI plug-in, or the Cloud Databases API.

The admin user

When you provision a Databases for Elasticsearch deployment, an `admin` user is automatically created.

> ⚠ **Important:** Set the admin password before using it to connect.

Setting the Admin password in the UI

Set your Admin password through the UI by selecting your instance from the Resource list. On the dashboard, select the **Settings** tab, then either enter or generate a new admin password and click the *Change password* button.

Setting the Admin password in the CLI

Use the `cdb user-password` command from the IBM Cloud CLI Cloud Databases plug-in to set the admin password.

For example, to set the admin password for a deployment named `example-deployment`, use the following command:

```
$ ibmcloud cdb user-password example-deployment admin <newpassword>
```

Setting the Admin password in the API

The *Foundation endpoint* that is shown on the *Overview* panel in the Deployment details section of your service provides the base URL to access this deployment through the API. Use it with the Set specified user's password endpoint to set the admin password.

```
$ curl -X PATCH `https://api.{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/users/admin` \
-H `Authorization: Bearer <>` \
-H `Content-Type: application/json` \
-d `{"password":"newrootpasswordsupersecure21"}` \
```

The `ibm_superuser` role

For Elasticsearch versions 7.17.7 and later, all users, both new and existing, including the `admin` user, will be assigned the `ibm_superuser` role. The `ibm_superuser` role has the same privileges as `superuser`, except the `ibm_superuser` role cannot access hidden, internal, or restricted indexes. Restricting access to hidden indexes prevents users from inadvertently limiting their own access by closing security indexes.

> ⚠ **Important:** While you have the permissions to change this role back to the original `superuser`, you should not do so. Changing `ibm_superuser` to `superuser` will disrupt your access to your Elasticsearch deployment.

Managing users and roles through the UI

1. Go to the service dashboard for your service.
2. Click **Service credentials** to open the page.
3. Click **New credential**.
4. Choose a descriptive name for your new credential.
5. (Optional) Specify whether the new credentials use a public or private endpoint. Use either `{ "service-endpoints": "public" }` / `{ "service-endpoints": "private" }` in the *Add Inline Configuration Parameters* field to generate connection strings that use the specified endpoint. Use of the endpoint is not enforced. It just controls which hostnames are in the connection strings. Public endpoints are generated by default.
6. Click **Add** to provision the new credentials. A username and password, and an associated Elasticsearch user is auto-generated.

The new credentials appear in the table, and the connection strings are available as JSON in a click-to-copy field under *View credentials*.

Creating a user from the CLI or API doesn't automatically populate that user's connection strings into *Service credentials*. If you want to add them there, you can create a new credential with the existing user information. Enter the username and password in the JSON field under *Add Inline Configuration Parameters*. For example, `{"existing_credentials":{"username":"Robert","password":"supersecure"}}`. Basically, you send in the username and password, and *Service credentials* generates the connection strings with the credentials filled in.

Tip: Generating credentials from an existing user does not check for or create that user.

If you need users that are created from *Service credentials* to have a different role, use the `admin` user to change their role.

Managing users and roles through the CLI

If you need users to have a different role, you can use the `admin` user to change their role.

Users that are created directly from the CLI do not appear in *Service credentials*, but you can add them.

If you manage your service through the Cloud Databases CLI plug-in, create a new user with `cdb user-create`. For example, to create a new user for a deployment named `example-deployment`, use the following command:

```
$ ibmcloud cdb user-create example-deployment <newusername> <newpassword>
```

When the task finishes, retrieve the new user's connection strings with the ibmcloud cdb deployment-connections command, which looks like:

```
$ ibmcloud cdb deployment-connections [--user <userid>] [--password <password>] [--endpoint-type <endpoint type>] [--all] [--only] [--start] [--certroot <path>] [--json]
```

Managing users and roles through the API

If you need users to have a different role, use the admin user to change their role.

Users that are created directly from the API do not appear in *Service credentials*, but you can add them.

The *Foundation endpoint* that is shown on the *Overview* section of your service provides the base URL to access this deployment through the API. To create and manage users, use the base URL with the /users endpoint.

The command looks like:

```
$ curl -X POST 'https://api.{region}.databases.cloud.ibm.com/v4/ibm/deployments/{id}/users' \
-H "Authorization: Bearer $APIKEY" \
-H "Content-Type: application/json" \
-d '{"username":"jane_smith", "password":"newsupersecurepassword"}'
```

To retrieve a user's connection strings, use the base URL with the `/users/{userid}/connections` endpoint.

Elasticsearch-created users and roles

If the built-in users and roles do not suit your environment, create users and roles directly in Elasticsearch. The admin user for your deployment has the power to create any role or set of privileges for use on your deployment.

Upgrading to a new major version

IBM Cloud® Databases for Elasticsearch provides two different upgrade paths:

- In-place upgrade to a new major version
- Restoring from backup (supported for Elasticsearch Enterprise Plan, Elasticsearch Platinum Plan)

In-place major version upgrades

In-place major version upgrade allows you to upgrade your deployment to the next new major version, eliminating the need to restore a backup into a new deployment. This approach maintains the same connection strings, without the need to reconfigure the deployment. However, if the new major version requires application adjustments, these must be addressed.

Options for in-place upgrade

- **With backup**: Creates a backup before performing the upgrade, providing an added layer of safety.
- **Without backup**: Proceeds without creating a backup. If the upgrade fails, restoration must be done from the latest backup into a new deployment.

⚠ **Important:** In-place upgrade without backup is not recommended. It may result in data loss if the upgrade fails at any stage, , as there will be no immediate backup to restore from.

Before you begin

- Ensure that your deployment is in a healthy state.
- Ensure at least 4 GB of free disk space.
- Only upgrade to the next major version, instead of specifying the version of your choice.
- Each major version contains some features that may not be backward-compatible with previous versions. Review the release notes from the database vendor to see any changes that may affect your applications.
- Downgrading a deployment to a previous version is not supported.
- In-place upgrade cannot be cancelled once started.

Upgrading in the UI

1. Create a Databases for Elasticsearch to test the upgrade process. Create the deployment by restoring a backup from your existing deployment with the same version.
2. Point your staging application to the test deployment. Update your staging application to point to the test deployment. Confirm that your test application can connect successfully to the staging deployment and that the application operates as expected. Perform any required performance and operational testing of the staging environment.
3. Upgrade the major version of your test deployment by clicking on the Upgrade major version button on the *Overview* page. Your database will not be accessible during the upgrade window. Note how long the upgrade takes to complete so that you can use the upgrade expiry setting to contain upgrades within your maintenance window.
4. Confirm that your staging application works with the new database version. If your application works, this step confirms that it should be safe to upgrade your production database.
5. Upgrade your production deployment by clicking **Upgrade major version**. Once you confirmed that your application works correctly by using the new version of the database, return to the management console and start the process of upgrading your production deployment. In the *Deployment details* section of the *Overview* page, click **Upgrade major version** and follow the steps.

Once the in-place upgrade starts, it cannot be stopped or rolled back. So, in the unlikely event of an error, your database deployment could become unrecoverable. Therefore, create a backup that you can then use to restore to a new deployment. If you select *In-place major version upgrade with backup* , the backup that is created can be used to restore in a new deployment.

The *expiration for starting upgrade* allows you to configure a 'timeout' period that the upgrade job must start within before it is automatically cancelled. In addition, test the upgrade in staging upfront to ensure that the upgrade completes within your desired time window. If, for example, you want to complete the upgrade within 1 hour, and you tested the upgrade and know that it takes 30 minutes, then your upgrade job must start within 30 minutes of you confirming that you want to upgrade. Therefore, set the expiration to 30 minutes, so that if it doesn't start within that time, it won't overrun your window.

Troubleshooting

Healthchecks

- Before starting an Elasticsearch upgrade, it is critical to verify that the cluster has sufficient resources and is in a healthy state.
- Ensure that the cluster health status is GREEN.
- Confirm that disk usage is below 85% to avoid upgrade failures due to insufficient space.
- Perform a secondary precheck to detect deprecations in the cluster.
- If deprecations are found, the upgrade process will stop and must be retried only after all issues are resolved.

If issues persist, open a support ticket with IBM Cloud support.

Elastic resources

- Elasticsearch health report API
- Elasticsearch migration deprecations API

Restoring from backup

Before a major version of a database reaches its end of life (EOL), upgrade to the next available major version by restoring from a backup into a new database instance.

Prepare to run on, and then migrate to, the latest version before the EOL date. For more information, see Versioning Policy.

Note: Rolling back versions is not supported.

Upgrade to the latest version of Elasticsearch available to Databases for Elasticsearch. Find the latest version from the catalog page, from the Cloud Databases CLI plug-in command `ibmcloud cdb deployables-show`, or from the Cloud Databases API deployables endpoint].

Upgrading is handled by restoring a backup of your data into a new deployment. Restoring from a backup has various advantages:

- The original database stays running and production work can be uninterrupted.
- You can test the new database out of production and act on any application incompatibilities.
- The entire process can be rerun at any point.
- A fresh restoration reduces the likelihood that unneeded artifacts of the older version of the database are carried over to the new database.

Upgrade paths

Current version	Major version upgrade path
Elasticsearch 8.7	Elasticsearch 8.19
Elasticsearch 8.10	Elasticsearch 8.19
Elasticsearch 8.12	Elasticsearch 8.19
Elasticsearch 8.15	Elasticsearch 8.19
Elasticsearch 8.19	Elasticsearch 9.1

Major version upgrade paths

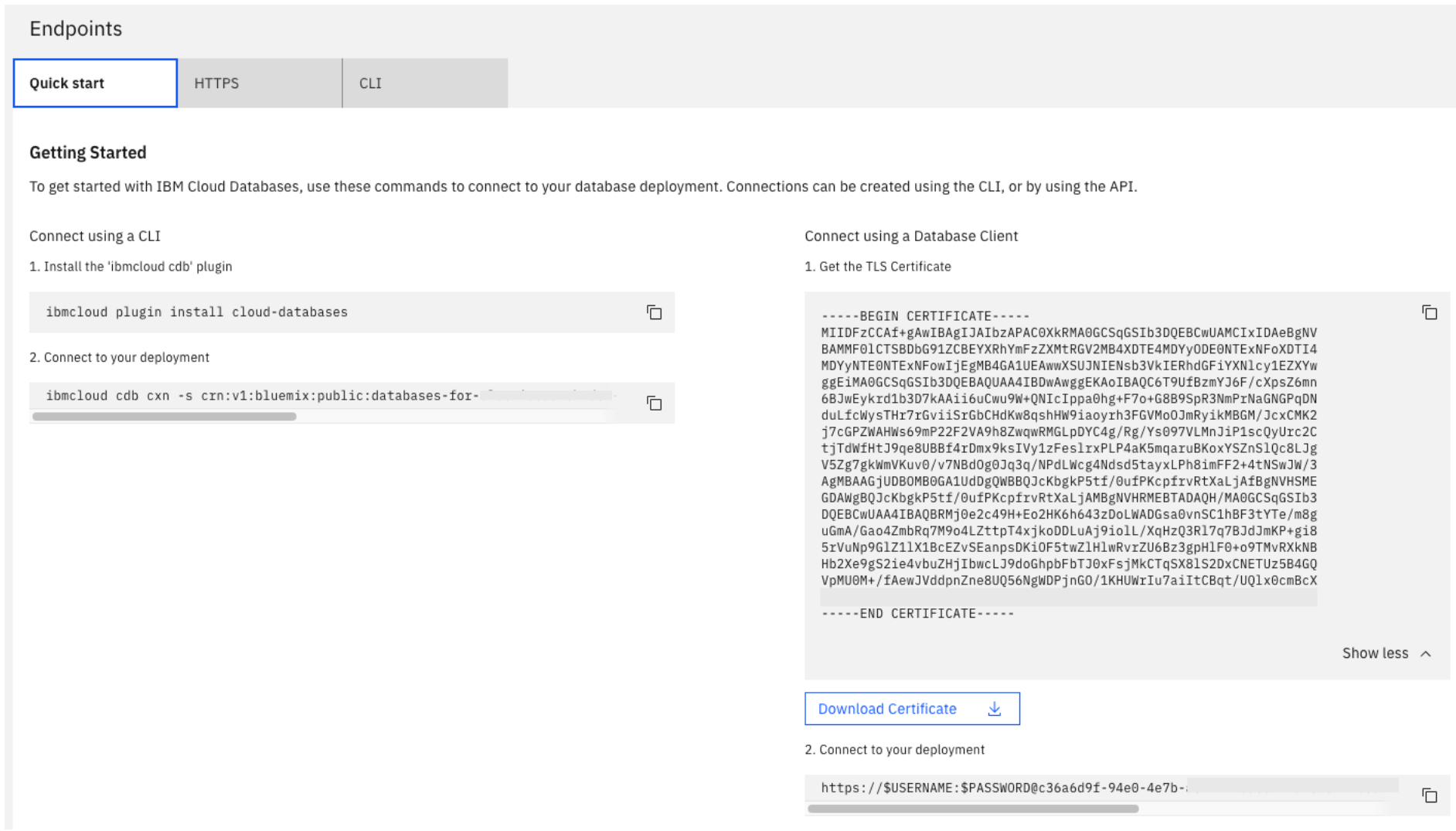
Upgrading in the UI

For new hosting models (isolated compute and shared compute), upgrading to a new major version is available by using the CLI and API.

You can upgrade to a new version by restoring a backup from the *Backups and restore* page of your deployment on the IBM Cloud console. Click **Restore backup** on a backup to open a page in a new tab where you can change some options for the new deployment. One of them is the database version, which is auto-populated with the versions available for you to upgrade to. Select a version and click **Restore backup** to start the provision and restore process.

Getting connection strings

To connect to IBM Cloud® Databases for Elasticsearch, you need some users and connection strings. Connection strings for your deployment are displayed on the *Overview* page, in the *Endpoints* panel. These strings can be used with any set of credentials that you generate.



Endpoints panel

Tip: A Databases for Elasticsearch deployment is provisioned with an admin user, and after you set the admin password, you can use its credentials to connect to your deployment.

Getting connection strings in the CLI

You can also grab connection strings from the CLI.

```
$ ibmcloud cdb deployment-connections example-deployment -u <newusername> [--endpoint-type <endpoint type>]
```

Full connection information is returned by the `ibmcloud cdb deployment-connections` command with the `--all` flag. To retrieve all the connection information for a deployment named "example-deployment", use the following command.

```
$ ibmcloud cdb deployment-connections example-deployment -u <newusername> --all [--endpoint-type <endpoint type>]
```

If you don't specify a user, the `deployment-connections` commands return information for the admin user by default. If you don't specify an endpoint type, the connection string returns the public endpoint by default. If your deployment has only a private endpoint, you must specify `--endpoint-type private` or the commands return an error. The user and endpoint type is not enforced. You can use any user on your deployment with either endpoint (if both exist on your deployment).

Tip: To use the `ibmcloud cdb` CLI commands, you must install the Cloud Databases plug-in.

Getting connection strings with the API

To retrieve user's connection strings from the API, use the `/users/{userid}/connections` endpoint. You must specify in the path which user and which type of endpoint (public or private) should be used in the returned connection strings. The user and endpoint type is not enforced. You can use any user on your deployment with either endpoint (if both exist on your deployment).

```
$ curl -X GET -H "Authorization: Bearer $APIKEY" 'https://api.
{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/users/{user_type}/{user_id}/connections/{endpoint_type}'
```

More users and connection strings

Access to your Databases for Elasticsearch deployment is not limited to the admin user. You can create users by using the *Service Credentials* page, the IBM Cloud CLI, or through the IBM Cloud Databases API.

All users on your deployment can use the connection strings, including connection strings for either public or private endpoints.

Creating users from the UI

1. Navigate to the resource detail page for your service.
2. Click **Service credentials** to open the *Service Credentials* page.
3. Click **New credential**.
4. Choose a descriptive name for your new credential.
5. (Optional) Specify whether the new credentials use a public or private endpoint. Use either `{ "service-endpoints": "public" }` / `{ "service-endpoints": "private" }` in the *Add Inline Configuration Parameters* field to generate connection strings using the specified endpoint. Use of the endpoint is not enforced as it controls which hostnames are in the connection strings. Public endpoints are generated by default.
6. Click **Add** to provision the new credentials. A username and password, and an associated Elasticsearch user is auto-generated.

The new credentials appear in the table, and the connection strings are available as JSON in a click-to-copy field under *View Credentials*.

Creating users from the CLI

If you manage your service through the IBM Cloud CLI and the cloud databases plug-in, you can create a new user with `cdb user-create`. For example, to create a new user for an "example-deployment", use the following command.

```
$ ibmcloud cdb user-create example-deployment <newusername> <newpassword>
```

Once the task has finished, you can retrieve the new user's connection strings with the `ibmcloud cdb deployment-connections` command.

Creating users with the API

The *Foundation Endpoint* that is shown on the *Overview* panel of your service provides the base URL to access this deployment through the API. To create and manage users, use the base URL with the `/users` endpoint.

```
$ curl -X POST 'https://api.{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/users/{user_type}' \
-H "Authorization: Bearer $APIKEY" \
-H "Content-Type: application/json" \
-d '{"user": {"username": "user", "password": "v3ry-1-secUre-pAssword-2"}}'
```

To retrieve a user's connection strings, use the base URL with the `/users/{userid}/connections` endpoint.

Adding users to *service credentials* in the UI

Creating a new user from the CLI doesn't automatically populate that user's connection strings into *Service Credentials*. If you want to add them there, you can create a new credential with the existing user information.

Enter the username and password in the JSON field *Add Inline Configuration Parameters*, or specify a file where the JSON information is stored. For example, putting `{"existing_credentials":{"username":"Robert","password":"supersecure"}}` in the field generates *Service Credentials* with the username "Robert" and password "supersecure" filled into connection strings.

Tip: Generating credentials from an existing user does not check for or create that user.

Connection String Breakdown

The HTTPS Section

The "https" section of a credential created on the *Service credentials* page contains information that is suited to applications that make connections to Elasticsearch.

Field name	Index	Description

Type		Type of connection - for Elasticsearch, it is "uri".
Scheme		Scheme for a URI - for Elasticsearch, it is "https".
Path		Path for a URI.
Authentication	Username	The username that you use to connect.
Authentication	Password	A password for the user - might be shown as `$PASSWORD`.
Authentication	Method	How authentication takes place; "direct" authentication is handled by the driver.
Hosts	0...	A hostname and port to connect to.
Composed	0...	A URI combining Scheme, Authentication, Host, and Path.
Certificate	Name	The allocated name for the service proprietary certificate for database deployment.
Certificate	Base64	A base64 encoded version of the certificate.

Https/URI connection information

- `0...` indicates that there may be one or more of these entries in an array.

The CLI section

The "CLI" section of a credential created on the *Service credentials* page contains information that is suited for connecting with `cURL`.

Field name	Index	Description
Bin		The recommended binary to create a connection; in this case it is `curl`.
Composed		A formatted command to establish a connection to your deployment. The command combines the `Bin` executable, `Environment` variable settings, and uses `Arguments` as command-line parameters.
Environment		A list of key/values you set as environment variables.
Arguments	0...	The information that is passed as arguments to the command shown in the Bin field.
Certificate	Base64	A service proprietary certificate that is used to confirm that an application is connecting to the appropriate server. It is base64 encoded.
Certificate	Name	The allocated name for the service proprietary certificate.
Type		The type of package that uses this connection information; in this case `cli`.

curl connection information

- `0...` indicates that there may be one or more of these entries in an array.

Migrating Elasticsearch

If you are a current user of Elasticsearch, use the Snapshot/restore Elasticsearch API and the S3 repository plug-in to migrate your data into IBM Cloud® Databases for Elasticsearch.

Take a snapshot of your existing Elasticsearch, store the snapshot in an AWS S3 or IBM Cloud® Object Storage bucket and then restore the snapshot to your Databases for Elasticsearch deployment.

To migrate while data is still being written to your existing Elasticsearch, take multiple snapshots and perform multiple incremental restores. After the Databases for Elasticsearch deployment catches up to the state of your Elasticsearch, move your application writes to Databases for Elasticsearch.

Requirements

Ensure you meet the following requirements before migrating:

- Your existing Elasticsearch must have the S3 repository plug-in.
- An Elasticsearch that is version 5.x or 6.x. Migration is possible between major versions, but the versions must have compatible indexes. Indexes that are made in Elasticsearch 2.x are not compatible with 6.x and need reindexing in 5.x before migration.
- A matching Databases for Elasticsearch deployment that has *at least* as many resources allocated to it your existing Elasticsearch. Also, ensure that the same plug-ins are available on Databases for Elasticsearch that you have on the existing Elasticsearch.
- Your our own S3 or IBM Cloud Object Storage repository.

> **Note:** Incremental restores work only if the number of shards of each index on both deployments match. Don't try to reindex and change the number of shards of any indexes once you've taken snapshots.

Example Migration

A migration is explored in detail in Elasticsearch data migration using snapshot and restore.

In this example, data is still being written while the migration occurs, so multiple snapshots and restores are used.

The example's shell script is available in the IBM Cloud GitHub repository. It is provided as a starting point for you to adapt for your use-case.

Migrating resources to a different data center

IBM Cloud®'s investments in data center infrastructure include rolling out newer data centers and multizone regions (MZRs) and closing older data centers that are unsuitable for upgrading.

For a current list of data centers, see Locations for resource deployment.

For information on data center closures, see Data center migrations.

Migrating your resources

To identify your impacted resources, take advantage of special offers, or learn about recommended configurations, use one of the following options to contact the IBM 24x7 Client Success team:

- Live chat
- Phone: (US) 866-597-9687

To avoid any disruption to your service, please complete the following steps to migrate your resources from your current data center to your new location:

- Restore your backup into a new database, in a new region. For more information, see Managing Cloud Databases backups.
- For an IBM Cloud® Databases for PostgreSQL deployment, you can deploy Read Replicas into a new region and turn that Read Replica into a stand-alone database.

Understanding high availability for Cloud Databases

Note: This document covers all the IBM Cloud® Databases, which include Databases for PostgreSQL, Databases for MongoDB, Databases for Redis, Databases for Elasticsearch, IBM Cloud® Databases for MySQL, and Messages for RabbitMQ.

Regions

IBM Cloud® Databases instances are deployed in either a multi-zone region (MZR) (for example, Dallas, Frankfurt, London, Sydney, Tokyo, and Washington), or a single-campus multizone region (for example, Chennai). Each instance is deployed in a highly available configuration; that is, data is replicated by each database onto one or more servers, making the data highly available during normal operations.

- In MZRs, database members are distributed across different data centers, or zones.
- In single-campus multizone regions, database members are distributed across different hosts.

If a single-campus multizone region failure in an MZR or a hardware failure in any region occurs, your data is still accessible as it is replicated onto other fully functioning database servers. Such issues are addressed by IBM Cloud® Specialists in place.

For more information on how your specific database replicates data among each of its members, see your Cloud Databases documentation.

Backups

- In addition to the high-availability configuration, for deployments in IBM Cloud® multi-zone regions, your data is snapshotted and backed up daily by the IBM Cloud® Databases platform and stored in cross-region Cloud Object Storage buckets.
- For most IBM Cloud® single-campus multizone regions, your data is backed up locally in Single-campus multizone region Cloud Object Storage buckets.

If a complete region failure occurs, the database servers in the region might not be accessible, but the backup data remains available. You can initiate a restore from these backups into an available region from the service management console. For more information, see the Cloud Databases backups documentation.

It is your responsibility to create a new service instance in which to restore when the IBM Cloud® Databases platform is restored. You are also responsible for testing the validity and restore time of your backups. For more information, see the Disaster recovery section in the *Shared responsibilities for Cloud Databases* page.

Application-level high availability

Applications that communicate over networks and cloud services are subject to transient connection failures. You want to design your applications to reconnect (not just retry) when errors are caused by a temporary loss in connectivity to your deployment or to IBM Cloud.

Because Cloud Databases is a managed service, regular updates and database maintenance occur as part of normal operations. Such maintenance can occasionally cause short intervals where your database is disabled.

Your applications must be designed to handle temporary interruptions to the database, implement error handling for failed database commands, and implement reconnect logic to recover from a temporary interruption.

Several minutes of database unavailability or connection interruptions are not expected. Open a support ticket with details if you have time periods longer than a minute with no connectivity so we can investigate.

If you have deployments in more than one region, you must provision IBM Cloud® Monitoring and enable platform metrics in each region. For more information, see IBM Cloud Monitoring integration.

SLAs

- See How IBM Cloud ensures high availability and disaster recovery to learn more about the high availability and disaster recovery standards in IBM Cloud.
- All IBM Cloud® Databases general availability (GA) offerings conform to the IBM Cloud® Service Level Agreement (SLA) terms.
- For more information, see the Responsibilities for Cloud Databases page.

Connecting to Databases for Elasticsearch

Connecting an external application

Your applications and drivers use connection strings to make a connection to IBM Cloud® Databases for Elasticsearch. Each deployment has connection strings specifically for drivers and applications. Connection strings are displayed in the *Endpoints* panel of your deployment's *Overview* page, and can also be retrieved from the cloud databases CLI plug-in, and the API.

The connection strings can be used by any of the credentials you create in your deployment. While you can use the admin user for all of your connections and applications, it might be better to create users specifically for your applications to connect with. For more information, see Getting Connection Strings.

Connecting with a language's driver

All the information a driver needs to make a connection to your deployment is in the "https" section of a credential created on the *Service credentials* page. The table contains a breakdown for reference.

Field Name	Index	Description
`Type`		Type of connection - for Elasticsearch, it is "uri".
`Scheme`		Scheme for a URI - for Elasticsearch, it is "https".
`Path`		Path for a uri.
`Authentication`	`Username`	The username that you use to connect.
`Authentication`	`Password`	A password for the user - might be shown as `$PASSWORD`.
`Authentication`	`Method`	How authentication takes place; "direct" authentication is handled by the driver.
`Hosts`	`0...`	A hostname and port to connect to.
`Composed`	`0...`	A URI combining Scheme, Authentication, Host, and Path.
`Certificate`	`Name`	The allocated name for the service proprietary certificate for database deployment.
`Certificate`	Base 64	A base64 encoded version of the certificate.

https/URI connection information

- `0...` indicates that there might be one or more of these entries in an array.

Many Elasticsearch drivers are able to make a connection to your deployment when given the URI-formatted connection string found in the "composed" field of the connection information. See the following example:

```
$ https://admin:$PASSWORD@d5eeee66-5bc4-498a-b73b-1307848f1eac.8f7bfd8f3faa4218aec56e069eb46187.databases.appdomain.cloud:31821
```

The following example uses Java to connect.

```
import io.searchbox.client.JestClientFactory;
import io.searchbox.client.JestResult;
import io.searchbox.client.config.HttpClientConfig;
import io.searchbox.client.JestClient;
import io.searchbox.cluster.Health;
import org.apache.http.conn.ssl.SSLConnectionSocketFactory;
import org.apache.http.ssl.SSLContextBuilder;

import javax.net.ssl.SSLContext;
import java.io.File;
import java.security.*;
import java.security.cert.CertificateException;
```

```
import java.io.IOException;


public class ESConnect {

    public static void main(String[] args) {

        // Add CA cert to truststore using something like:
        //  keytool -import -alias mycert -file /path/to/cert -keystore ./mycert -storetype pkcs12 -storepass mysecret
        // and use the path of the keystore below
        File truststore = new File("/Users/code/java-example/icdcerts");

        try {
            // use your secret and add the variable containing the truststore that you created above with the secret as a CharArray
            SSLContext sslContext = new SSLContextBuilder().loadTrustMaterial(truststore, "mysecret".toCharArray()).build();

            SSLConnectionSocketFactory sslSocketFactory = new SSLConnectionSocketFactory(sslContext);

            // set up a Jest factory
            JestClientFactory factory = new JestClientFactory();

            //configure and build Jest HTTP client with IBM Cloud Databases for Elasticsearch connection strings
            factory.setHttpClientConfig(
                    // add the Elasticsearch host and port
                    new HttpClientConfig.Builder("https://60d1b41b-2478-4767-9fc0-d99b1d00b6d1.bkvfu0nd0m8k95k94ujg.databases.appdomain.cloud:31347")
                            .multiThreaded(true)
                            // Add the credentials username and password
                            .defaultCredentials("admin", "mypassword")
                            .sslSocketFactory(sslSocketFactory)
                            .build());

            // create a JestClient
            JestClient client = factory.getObject();
            // create the call for the cluster health
            Health health = new Health.Builder().build();
            // get the cluster health as a JestResult
            JestResult result = client.execute(health);
            // print out the cluster's health
            System.out.printf("\n\n<------ CLUSTER HEALTH ------>\n%s\n\n", result.getJsonObject());
            // shutdown the connection
            client.close();

        } catch (IOException | KeyStoreException | NoSuchAlgorithmException | KeyManagementException | CertificateException e) {
            e.printStackTrace();
        }

    }

}
```

The following example uses the Python library elasticsearch-py to connect.

```
from elasticsearch import Elasticsearch
from ssl import create_default_context


context = create_default_context(cafile="path/to/cert.pem")

es = Elasticsearch(
    ['60d1b41b-2478-4767-9fc0-d99b1d00b6d1.bkvfu0nd0m8k95k94ujg.databases.appdomain.cloud'],
    http_auth=('admin', 'password'),
    port='31347',
    ssl_context=context
)

health = es.cluster.health()
print(health)
```

Driver TLS and service proprietary certificate support

All connections to Databases for Elasticsearch are TLS 1.2 enabled, so the driver you use to connect needs to be able to support encryption. Your deployment also comes with a service proprietary certificate so the driver can verify the server upon connection.

For more information, see Cloud Databases Certificates FAQ.

Using the service proprietary certificate

1. Copy the certificate information from the *Endpoints* panel or the Base64 field of the service credential connection information.
2. If needed, decode the Base64 string into text.
3. Save the certificate to a file. (You can use the Name that is provided or your own file name).
4. Provide the path to the certificate to the driver or client.

CLI plug-in support for the service proprietary certificate

You can display the decoded certificate for your deployment with the CLI plug-in with the command `ibmcloud cdb deployment-cacert "your-service-name"`. It decodes the base64 into text. Copy and save the command's output to a file and provide the file's path to the driver.

Other drivers

Elasticsearch has a vast array of language drivers. The table covers a few of the most common.

Language	Driver	Documentation
Node	`elasticsearch-js`	Link
Ruby	`elasticsearch-ruby`	Link
Ruby on Rails	elasticsearch-rails	Link
Python	`elasticsearch-py`	Link
Java	`Jest`	Link
Go	`elastic`	Link

Common Elasticsearch drivers

Connecting an IBM Cloud application

Applications running in IBM Cloud can be bound to your IBM Cloud® Databases for Elasticsearch deployment.

Connecting a Kubernetes service application

There are two steps to connecting a Cloud databases deployment to a Kubernetes service application. First, your deployment needs to be bound to your cluster and its connection strings stored in a secret. The second step is configuring your application to use the connection strings.

Tip: The sample app in Connecting a Kubernetes service tutorial provides a sample application that uses Node.js and demonstrates how to bind the sample application to a Cloud Databases deployment.

Before connecting your Kubernetes Service application to a deployment, make sure that the deployment and cluster are both in the same region and resource group.

Binding your deployment

1. **Public or private endpoints**

- **Public endpoints** - If you are using the default public service endpoint to connect to your deployment, you can run the `cluster service bind` command with your cluster name, the resource group and your deployment name.

```
$ ibmcloud ks cluster service bind <your_cluster_name> <resource_group> <your_database_deployment>
```

- **Private endpoints** - If you want to use a private endpoint (if one is enabled on your deployment), then first you need to create a service key for your

database so Kubernetes can use it when binding to the database.

```
$ ibmcloud resource service-key-create <your-private-key> --instance-name <your_database_deployment> --service-endpoint private
```

The private service endpoint is selected with `--service-endpoint private`. After that, you bind the database to the Kubernetes cluster through the private endpoint with the `cluster service bind` command.

```
$ ibmcloud ks cluster service bind <your_cluster_name> <resource_group> <your_database_deployment> --key <your-private-key>
```

2. **Verify** - Verify that the Kubernetes secret was created in your cluster namespace. Running the following command, you get the API key for accessing the instance of your deployment that's provisioned in your account.

```
$ kubectl get secrets --namespace=default
```

For more information, see the Kubernetes service documentation.

Configuring in your Kubernetes app

When you bind your application to Kubernetes Service, it creates an environment variable from the cluster's secrets. Your deployment's connection information lives in `BINDING` as a JSON object. Load and parse the JSON object into your application to retrieve the information your application's driver needs to make a connection to the database.

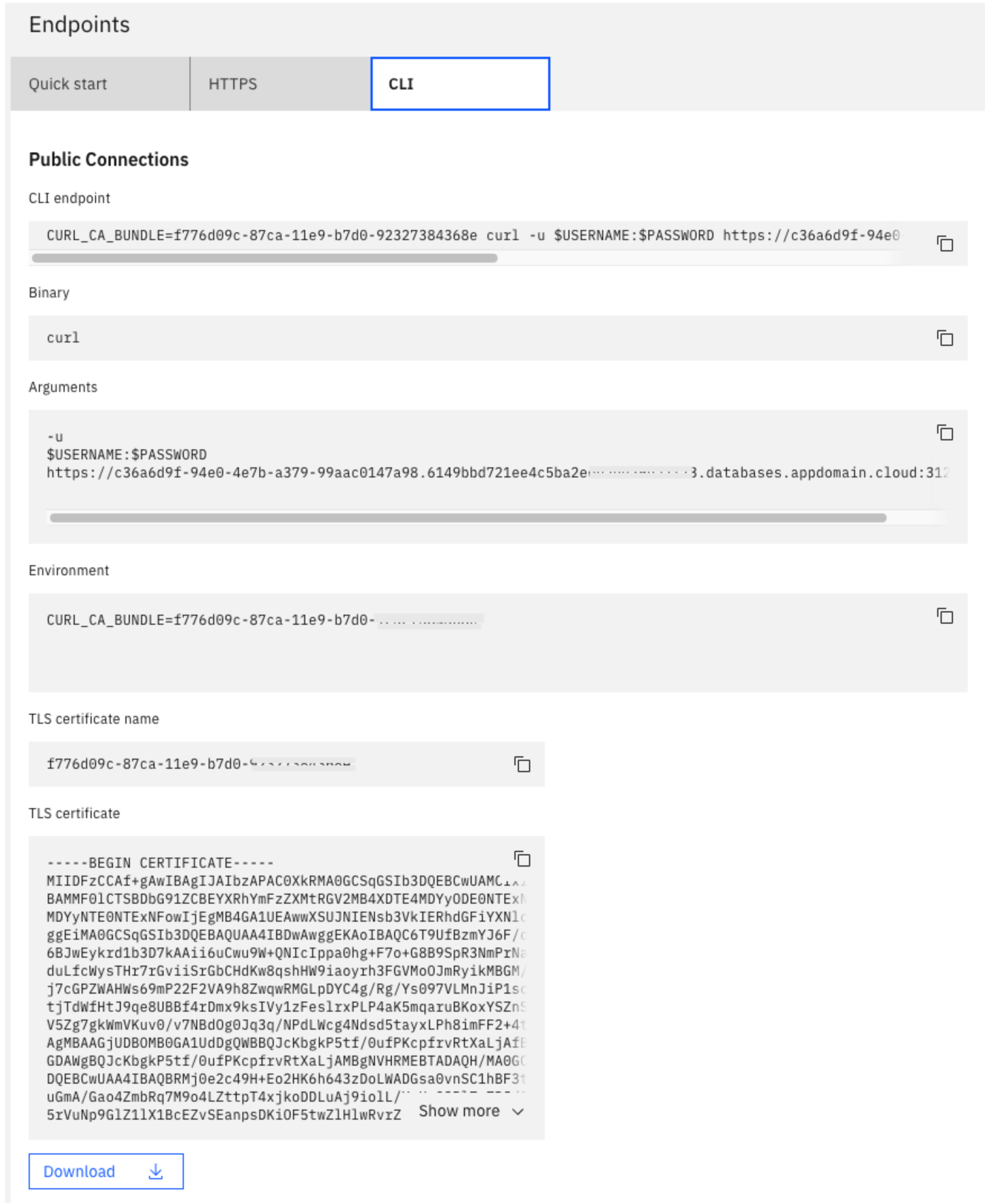
The Connection Strings page contains a reference of the JSON fields.

For more information, see the Kubernetes service docs.

Connecting with cURL

You can access your Elasticsearch database directly from a command-line terminal through cURL. Elasticsearch has a wide variety of REST APIs that allow for cluster monitoring, index management and searching within the database.

Connection strings are displayed in the *Endpoints* panel of your deployment's *Overview* page, and can also be retrieved from the cloud databases CLI plug-in, and the API.



Endpoints section, CLI tab

The information that you need to make a connection with cURL to your deployment is also in the "CLI" section of a credential created on the *Service credentials* page. The table contains a breakdown for reference.

Field Name	Index	Description
Bin		The recommended binary to create a connection; in this case it is `curl`.
Composed		A formatted command to establish a connection to your deployment. The command combines the `Bin` executable, `Environment` variable settings, and uses `Arguments` as command-line parameters.
Environment		A list of key/values you set as environment variables.
Arguments	0...	The information that is passed as arguments to the command shown in the Bin field.

`Certificate`	Base64	A service proprietary certificate that is used to confirm that an application is connecting to the appropriate server. It is base64 encoded.
`Certificate`	Name	The allocated name for the service proprietary certificate.
`Type`		The type of package that uses this connection information; in this case `cli`.

curl connection information

- `0...` indicates that there might be one or more of these entries in an array.

Elasticsearch API `cURL` example

```
$ CURL_CA_BUNDLE="/path-to/your_cert_file" curl -u admin:<password> 'https://d5eeee66-5bc4-499a-b73b-1307848f1eac.8f7bfd8f3faa4218aec56e069eb46187.databases.appdomain.cloud:31821/_cluster/health?pretty'
```

- `CURL_CA_BUNDLE` - curl performs SSL certificate verification by default. Since your deployment uses a service proprietary certificate, you must specify what certificate to use.
- `curl` - The command itself.
- `-u` - The parameter for the username and password, separated by a colon, to be used as credentials to log in to the Elasticsearch deployment.
- `https://...` - The parameter that specifies the endpoints where the `curl` command connects. It's composed of HTTPS protocol URLs and includes a port number.
- `/_cluster/health?pretty` - An Elasticsearch Cluster API endpoint that returns the status of your cluster.

Using the service proprietary certificate

1. Copy the certificate information from the *Endpoints* panel or the Base64 field of the service credential connection information.
2. If needed, decode the Base64 string into text.
3. Save the certificate to a file. (You can use the Name that is provided or your own file name).
4. Provide the path to the `CURL_CA_BUNDLE` variable.

CLI plug-in support for the service proprietary certificate

You can display the decoded certificate for your deployment with the CLI plug-in with the command `ibmcloud cdb deployment-cacert "your-service-name"`. It decodes the base64 into text. Copy and save the command's output to a file and provide the file's path to the `CURL_CA_BUNDLE` variable.

Managing resources

Adding disk, memory, and CPU

The Shared Compute hosting model supports more fine-grained resource allocations that are not shown in the UI to maintain clarity. For more information, see Hosting models.

To scale an Isolated Compute host flavor instance, set the relevant `hostflavor` parameter to the Isolated Compute size you're targeting, such as "b3c.4x16.encrypted". As this includes CPU and RAM allocation selections, do not separately select CPU and RAM.

To scale a Shared Compute host flavor instance between the minimum CPU value and 2 CPU, set the CPU to 0 and scale the RAM allocation using the following commands. The CPU value will scale as a ratio of 1 CPU : 8 GB RAM, up to 2 CPU. To scale above 2 CPU, set the CPU and RAM allocations to your target allocation. For both, make sure to include the relevant `hostflavor` parameter.

You can manually adjust the resources available to your IBM Cloud® Databases for Elasticsearch deployment to suit your workload and the size of your data.

To scale an Isolated Compute host flavor instance, set the relevant `host_flavor` parameter to the Isolated Compute size you're targeting, such as "b3c.4x16.encrypted". As this includes CPU and RAM allocation selections, do not separately select CPU and RAM.

To scale a Shared Compute host flavor instance between the minimum CPU value and 2 CPU, set the CPU to 0 and scale the RAM allocation using the following commands. The CPU value will scale as a ratio of 1 CPU : 8 GB RAM, up to 2 CPU. To scale above 2 CPU, set the CPU and RAM allocations to your target allocation. For both, make sure to include the relevant `host_flavor` parameter.

You can manually adjust the resources available to your IBM Cloud® Databases for Elasticsearch deployment to suit your workload and the size of your data.

To scale an Isolated Compute host flavor instance, set the relevant `host_flavor` parameter to the Isolated Compute size you're targeting, such as "b3c.4x16.encrypted". As this includes CPU and RAM allocation selections, do not separately select CPU and RAM.

To scale a Shared Compute host flavor instance between the minimum CPU value and 2 CPU, set the CPU to 0 and scale the RAM allocation using the following commands. The CPU value will scale as a ratio of 1 CPU : 8 GB RAM, up to 2 CPU. To scale above 2 CPU, set the CPU and RAM allocations to your target allocation. For both, make sure to include the relevant `host_flavor` parameter.

You can manually adjust the resources available to your IBM Cloud® Databases for Elasticsearch deployment to suit your workload and the size of your data.

Resource breakdown

A default Databases for Elasticsearch deployment runs with three data members in a cluster and resources are allocated to all three members equally. For example, the minimum storage of an Elasticsearch deployment is 15360 MB, which equates to an initial size of 5120 MB per member. The minimum RAM for an Elasticsearch deployment is 3072 MB, which equates to an initial allocation of 1028 MB per member.

Tip: Billing is based on the *total* resources that are allocated to the deployment.

Disk usage

Storage shows the amount of disk space that is allocated to your service. Each member gets an equal share of the allocated space. Your data is replicated across all the data members in the Elasticsearch cluster.

Disk allocation also affects the performance of the disk, with larger disks having higher performance. Baseline input/output operations per second (IOPS) performance for disk is 10 IOPS for each GB. Scale disk to increase the IOPS that your deployment can handle.

Tip: You cannot scale down storage. If your data set size has decreased, you can recover space by backing up and restoring to a new deployment.

RAM

If you find that your queries and database activity suffer from performance issues due to a lack of memory, you can scale the amount of RAM allocated to your service. If your database instance is on an Isolated Compute hosting model, select the CPU x RAM configuration that matches your resource needs. If your database instance is on a Shared Compute or Dedicated Core hosting model, select the RAM allocation that you want for your database. Note that Dedicated Core is deprecated, and will be removed in May 2025.

Adding memory to the total allocation adds memory to the members equally. Databases for Elasticsearch deployments have their memory allocation policy set at 50% heap and 50% system memory, so increasing the amount of RAM increases both heap and system memory. RAM can be scaled up or down.

vCPU

If you find that your database workloads need more CPU resources, you can scale the amount of CPU allocated to your service. If your database instance is on an Isolated Compute hosting model, select the CPU x RAM configuration that matches your resource needs. If your database instance is on a Shared Compute or Dedicated Core hosting model, select the CPU allocation that you want for your database. Note that Dedicated Core is deprecated, and will be removed in May 2025.

The default of 0 cores uses compute resources on multi-tenanted hosts. This style of multi-tenant is deprecated, and will be removed in September 2025 in favor of Shared Compute. CPU can be scaled up or down.

Scaling considerations

- Scaling up might cause your deployment to restart. If your deployment needs to be moved to a host with more capacity, the deployment is restarted as part of the move.
- Scaling down RAM or CPU does not trigger restarts.
- Disk cannot be scaled down.
- Scaling between hosting models (Shared Compute, Isolated Compute, and Dedicated Cores) moves your deployment to new hosts. Your databases are restarted as part of that move. As your deployment is moved to a new host, this can also take longer than just adding more resources. For more information, see Shared Compute and Isolated Compute.
- Similarly, drastically scaling up CPU, RAM, or disk can take longer to run than small resource increases to account for provisioning more underlying hardware resources.
- Scaling operations are logged in IBM Cloud® Activity Tracker Event Routing.
- If you find consistent trends in resource usage or want to scale when certain resource thresholds are reached, enable autoscaling on your deployment.
- Databases for Elasticsearch is designed to balance work load across a cluster and can benefit from being horizontally scaled. If you are concerned about performance, check out Adding Elasticsearch nodes.

Review current resources and hosting model

In the **Resources** tab, you find the **Hosting model** and **Resource allocations** tiles. These tiles reflect your current resources and hosting model. Selecting *Configure* allows you to adjust the settings in each tile.

Scaling in the UI

In the **Resources** tab of the UI, select *Configure* on the **Resource allocations** tile. This opens up a panel where you can adjust your resources.

If your database is on the Isolated Compute hosting model, you then see a "Host sizes" table, where you can select the vCPU and RAM configuration per member for your database.

If you are on the Shared Compute hosting model, you see the Small configuration, providing 0.5 vCPU and 4 GB RAM per member; the Small Custom option; or Custom configuration. Small Custom indicates that your database was scaled with the CLI, API, or Terraform, which provides more fine-grained resource scaling, along with an option for automatically allocated vCPU pro-rated against RAM value. On the UI, you can scale to Small and Custom, but are not able to scale to the fine-grained values provided by the CLI, API, or Terraform. With Custom, drag the slider or adjust the value in the input box to select your database's per member vCPU and RAM values.

The "Disk (GB/member)" slider is your disk selection per member. Drag the slider or adjust the number in the input box to change the number of GB disk. Note that Disk is tied to IOPS at 1 GB = 10 IOPS.

Members is the number of members of your database. For Elasticsearch, members are set to 3.

Review your total estimated cost in the calculator on the bottom. Note that if you have grandfathered costs, also known as legacy pricing structure, scaling your database instance will remove some or all of your legacy pricing. For more information on grandfathering and when it ends, see the Hosting models transition timeline.

After you are done, click *Apply changes"* to trigger the scaling operation.

Switch to and between hosting models in the UI

In the **Resources** tab of the UI, select *Configure* on the Hosting model tile. This opens up a panel where you can adjust your hosting model selection.

The first option available is "Select your hosting model". Here, you can switch to a different hosting model.

Below, you see the options to also adjust the resources of the new hosting model that you selected. Follow the instructions in the previous section, "Scaling in the UI" to adjust your resources.

Click *Apply changes* to trigger this scale operation.

Review current resources and hosting model

IBM Cloud CLI cloud databases plug-in supports viewing and scaling the resources on your deployment. Use the command `cdb deployment-groups` to see current resource information for your service, including which resource groups are adjustable. To scale any of the available resource groups, use `cdb deployment-groups-set` command.

For example, with the following command you can view the resource groups for a deployment named "example-deployment". Note that this command will also reveal if your database is a Shared Compute or Isolated Compute instance through the `hostflavor` attribute. If the `hostflavor` is null, it is on an old style hosting model.

```
$ ibmcloud cdb deployment-groups example-deployment
```

This produces the output:

```
$ Group   member
Count   3
|
+   Memory
|   Allocation            3072mb
|   Allocation per member   1024mb
|   Minimum               3072mb
|   Step Size             384mb
|   Adjustable            true
|   Cpu Enforcement Ratio Ceiling   49152mb
|   Cpu Enforcement Ratio           8192mb
|
+   CPU
|   Allocation            0
|   Allocation per member   0
|   Minimum               9
|   Step Size             3
|   Adjustable            true
|
+   HostFlavor
|   ID            multitenant
|   Name
|   HostingSize
|
+   Disk
|   Allocation            15360mb
|   Allocation per member   5120mb
|   Minimum               15360mb
|   Step Size             3072mb
|   Adjustable            true
```

The deployment has three members, with 3072 MB of RAM and 15360 MB of disk allocated in total. The "per member" allocation is 1024 MB of RAM and 5120 MB of disk. The minimum value is the lowest the total allocation can be set. The step size is the smallest amount by which the total allocation can be adjusted.

Resources and scaling in the CLI

The `cdb deployment-groups-set` command allows either the total RAM or total disk allocation to be set in MB. For example, to scale the memory of the "example-deployment" to 4096 MB of RAM for each memory member (for a total memory of 12288 MB), you use the command:

```
$ ibmcloud cdb deployment-groups-set example-deployment member --memory 12288
```

Determine the hosting model of your database in the CLI

Use the following command to review the value of the `hostflavor` attribute. This will be null if the database is on a deprecated hosting model (not Shared or Isolated Compute).

```
$ ibmcloud cdb groups <deployment_id> --json
```

Switching to and between Hosting Models in the CLI

If your database is a Shared Compute instance, you can adjust the memory, CPU, and disk options with the following command. If your database is not on Shared Compute, this command will also move a database from a different hosting model to the Shared Compute hosting model.

```
$ ibmcloud cdb deployment-groups-set <deploymentid> <groupid> [--memory <val>] [--cpu <val>] [--disk <val>] [--hostflavor multitenant]
```

For example, use the following to scale to a Shared Compute instance or scale up your Shared Compute instance:

```
$ ibmcloud cdb deployment-groups-set crn:abc ... xyz:: member  --memory 24576 --cpu 6  --hostflavor multitenant
```

If your database is an Isolated Compute instance, memory and CPU are adjusted together by selecting the Isolated Compute size (see all sizes in Table 1). Disk is scaled separately. If your database is not on Isolated Compute, this command will also move a database from a different hosting model to the Isolated Compute hosting model.

Note that since the host flavor selection includes CPU and RAM sizes (`b3c.4x16.encrypted` is 4 CPU and 16 RAM), this request does not accept both an Isolated size selection and separate CPU and RAM allocation selections.

```
$ ibmcloud cdb deployment-groups-set <deploymentid> <groupid> [--disk <val>] [--hostflavor <hostflavor>]
```

For example, use the following to scale to an Isolated Compute instance or scale up your Isolated Compute instance:

```
$ ibmcloud cdb deployment-groups-set crn:abc ... xyz:: member  --hostflavor b3c.8x32.encrypted
```

The `hostflavor` parameter

The `hostflavor` parameter defines your compute sizing. To provision a Shared Compute instance, specify `multitenant`. To provision an Isolated Compute instance, input the appropriate value for your desired CPU and RAM configuration.

Host flavor	hostflavor value
Shared Compute	`multitenant`
4 CPU x 16 RAM	`b3c.4x16.encrypted`
8 CPU x 32 RAM	`b3c.8x32.encrypted`
8 CPU x 64 RAM	`m3c.8x64.encrypted`
16 CPU x 64 RAM	`b3c.16x64.encrypted`
32 CPU x 128 RAM	`b3c.32x128.encrypted`
30 CPU x 240 RAM	`m3c.30x240.encrypted`

Host flavor sizing parameter

Review current resources and hosting model

The *Foundation Endpoint* that is shown on the *Overview* panel of your service provides the base URL to access this deployment through the API. Use it with the `/groups` endpoint if you need to manage or automate scaling programmatically.

To view the current and scalable resources on a deployment, use the /deployments/{id}/groups endpoint. Note that this command will also reveal if your database is a Shared Compute or Isolated Compute instance through the `host_flavor` attribute. If the `host_flavor` is null, it is on an old style hosting model.

```
$ curl -X GET -H "Authorization: Bearer $APIKEY" 'https://api.{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/groups'
```

Scaling with the API

To scale the memory of a deployment to 4096 MB of RAM for each member (there are 3 so a total memory of 12288 MB), use the /deployments/{id}/groups/{group_id} API endpoint.

```
$ curl -X PATCH 'https://api.{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/groups/member' \
-H "Authorization: Bearer $APIKEY" \
```

```
-H "Content-Type: application/json" \
-d '{"memory": {
      "allocation_mb": 12288
    }
  }'
```

Determine the hosting model of your database in the API

Use the following command to review the value of the `host_flavor` attribute. This will be null if the database is on a deprecated hosting model (not Shared or Isolated Compute).

```
$ curl -X GET https://api.{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/groups -H 'Authorization: Bearer <>' \
```

Switching to and between Hosting Models in the API

To scale any Cloud Databases Shared Compute instance, use the the following command, setting `host_flavor` to `multitenant`. If your database is not on Shared Compute, this command will also move a database from a different hosting model to the Shared Compute hosting model.

```
$ curl -X PATCH https://api.{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/groups/member
-H 'Authorization: Bearer <>'
-H 'Content-Type: application/json'
-d '{"host_flavor":
      {"id": "multitenant"},
    "cpu":
      {"allocation_count": 3},
    "memory":
      {"allocation_mb": 12288}
  }' \
```

To scale any instance into a Cloud Databases Isolated Compute instance or to scale to a different Isolated Compute size, use the `host_flavor` parameter, this time set to the desired Isolated Compute size. Available hosting sizes and their `host_flavor` value parameters are listed in Table 1. For example, `{"host_flavor": "b3c.4x16.encrypted"}`. Note that since the host flavor selection includes CPU and RAM sizes (`b3c.4x16.encrypted` is 4 CPU and 16 RAM), this request does not accept both an Isolated size selection and separate CPU and RAM allocation selections. Scale with the Cloud Databases API Scaling endpoint, with a command like:

```
$ curl -X PATCH https://api.{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/groups/member
-H 'Authorization: Bearer <>'
-H 'Content-Type: application/json'
-d '{"host_flavor": {"id": "b3c.4x16.encrypted"}}' \
```

Note: CPU and RAM autoscaling is not supported on Cloud Databases Isolated Compute. Disk autoscaling is available. If you have provisioned an Isolated instance or switched over from a deployment with autoscaling, keep an eye on your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

The `host_flavor` parameter

The `host_flavor` parameter defines your compute sizing. To provision a Shared Compute instance, specify `multitenant`. To provision an Isolated Compute instance, input the appropriate value for your desired CPU and RAM configuration.

Host flavor	host_flavor value
Shared Compute	`multitenant`
4 CPU x 16 RAM	`b3c.4x16.encrypted`
8 CPU x 32 RAM	`b3c.8x32.encrypted`
8 CPU x 64 RAM	`m3c.8x64.encrypted`
16 CPU x 64 RAM	`b3c.16x64.encrypted`
32 CPU x 128 RAM	`b3c.32x128.encrypted`

30 CPU x 240 RAM	`m3c.30x240.encrypted`

Table 1 Host flavor sizing parameter

Review current resources and hosting model

Review resource allocations to your database by checking your terraform scripts for `cpu { allocation_count = }`, `memory {allocation_mb = }`, and `disk { allocation_mb = }`. Review the `host_flavor` setting to determine if your database is a Shared Compute or Isolated Compute style hosting model. If `host_flavor` does not exist, your database is on an old style hosting model.

Scaling with Terraform

⚠ **Important:** Before executing a Terraform script on an existing instance, use the `terraform plan` command to compare the current infrastructure state with the desired state defined in your Terraform files. Any alteration to the `resource_group_id`, `service plan`, `version`, `key_protect_instance`, `key_protect_key`, `backup_encryption_key_crn` attributes recreates your instance. For a list of current argument references with the `Forces new resource` specification, see the ibm_database Terraform Registry.

Scale your instance by adjusting your Terraform script for the resource you're interested in. In the following example, `cpu`, `memory`, and `disk` allocations are specified. Note that if you have a host flavor selected (Isolated Compute or Shared Compute Multitenant), keep the host flavor selection in your script.

To implement your change, run `terraform apply`.

```
data "ibm_resource_group" "group" {
  name = "<your_group>"
}
resource "ibm_database" "<your_database>" {
  name              = "<your_database_name>"
  plan              = "standard"
  location          = "eu-gb"
  service           = "databases-for-elasticsearch"
  resource_group_id = data.ibm_resource_group.group.id
  tags              = ["tag1", "tag2"]
  adminpassword     = "password12"
  group {
    group_id = "member"
    cpu {
      allocation_count = 6
    }
    memory {
      allocation_mb = 24576
    }
    disk {
      allocation_mb = 256000
    }
  }
  users {
    name     = "user123"
    password = "password12"
  }
  allowlist {
    address     = "172.168.1.1/32"
    description = "desc"
  }
}
output "ICD Elasticsearch database connection string" {
  value = "http://${ibm_database.test_acc.ibm_database_connection.icd_conn}"
}
```

Alternatively, you can use pre-built, open-source, and enterprise-ready Terraform IBM Modules (TIM) for Databases for Elasticsearch that support the auto-scaling feature.

Switching to and Scaling Hosting Models in Terraform

Select the hosting model you want your database to be scaled to. You can change this later.

To scale your Databases for Elasticsearch instance to the Shared Compute hosting flavor, set the `"host_flavor"` parameter to `multitenant`. This works if you

want to scale to the Shared Compute hosting flavor, or if you want to keep the host flavor and scale your resources. To implement your change, run `terraform apply`. See the following example:

```
data "ibm_resource_group" "group" {
  name = "<your_group>"
}
resource "ibm_database" "<your_database>" {
  name              = "<your_database_name>"
  plan              = "standard"
  location          = "eu-gb"
  service           = "databases-for-elasticsearch"
  resource_group_id = data.ibm_resource_group.group.id
  tags              = ["tag1", "tag2"]
  adminpassword     = "password12"
  group {
    group_id = "member"
    host_flavor {
      id = "multitenant"
    },
    cpu {
      allocation_count = 6
    }
    memory {
      allocation_mb = 24576
    }
    disk {
      allocation_mb = 256000
    }
  }
  users {
    name     = "user123"
    password = "password12"
  }
  allowlist {
    address     = "172.168.1.1/32"
    description = "desc"
  }
}
output "ICD Elasticsearch database connection string" {
  value = "http://${ibm_database.test_acc.ibm_database_connection.icd_conn}"
}
```

Scale your Databases for Elasticsearch instance to Isolated Compute with the same `"host_flavor"` parameter, set to the desired Isolated size. This command works to scale your database instance to a different Isolated Compute size, as well as to move from another host flavor to the Isolated Compute host flavor. Available hosting sizes and their `host_flavor value` parameters are listed in Table 1. For example, `{"host_flavor": "b3c.4x16.encrypted"}`. Note that since the host flavor selection includes CPU and RAM sizes (`b3c.4x16.encrypted` is 4 CPU and 16 RAM), this request does not accept both an Isolated size selection and separate CPU and RAM allocation selections.

To implement your change, run `terraform apply`.

```
data "ibm_resource_group" "group" {
  name = "<your_group>"
}
resource "ibm_database" "<your_database>" {
  name              = "<your_database_name>"
  plan              = "standard"
  location          = "eu-gb"
  service           = "databases-for-elasticsearch"
  resource_group_id = data.ibm_resource_group.group.id
  tags              = ["tag1", "tag2"]
  adminpassword     = "password12"
  group {
    group_id = "member"
    host_flavor {
      id = "b3c.8x32.encrypted"
    }
    disk {
      allocation_mb = 256000
    }
```

```
  }
  users {
    name     = "user123"
    password = "password12"
  }
  allowlist {
    address     = "172.168.1.1/32"
    description = "desc"
  }
}
output "ICD Elasticsearch database connection string" {
  value = "http://${ibm_database.test_acc.ibm_database_connection.icd_conn}"
}
```

The host flavor parameter

The `host_flavor` parameter defines your compute sizing. To provision a Shared Compute instance, specify `multitenant`. To provision an Isolated Compute instance, input the appropriate value for your desired CPU and RAM configuration.

Host flavor	host_flavor value
Shared Compute	multitenant
4 CPU x 16 RAM	b3c.4x16.encrypted
8 CPU x 32 RAM	b3c.8x32.encrypted
8 CPU x 64 RAM	m3c.8x64.encrypted
16 CPU x 64 RAM	b3c.16x64.encrypted
32 CPU x 128 RAM	b3c.32x128.encrypted
30 CPU x 240 RAM	m3c.30x240.encrypted

Host flavor sizing parameter

Note: CPU and RAM autoscaling is not supported on Cloud Databases Isolated Compute. Disk autoscaling is available. If you have provisioned an Isolated instance or switched over from a deployment with autoscaling, keep an eye on your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

Autoscaling

Autoscaling is designed to respond to the short-to-medium term trends in resource usage on your IBM Cloud® Databases for Elasticsearch deployment. When enabled, your deployment is checked at the interval you specify. If it is running short on resources, more resources are added to the deployment. To keep an eye on your resources, use the IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization.

You can set your deployment to autoscale disk, RAM, or both.

General Autoscaling parameters:

- When to scale, based on usage over time.
- By how much to scale, as a percentage of the resources per member.
- How often to scale, measured either in seconds, minutes, or hours.
- A hard limit on scaling, your deployment stops scaling at the limit.

Memory - Memory autoscaling is based on Disk I/O utilization to provide more memory for disk caching as your read/write load increases. The benefit is that additional memory might alleviate pressure on disk I/O by supporting more caching. Autoscaling configurations based on memory usage are currently not available.

Disk - Disk autoscaling can scale when either disk usage reaches a certain threshold, Disk I/O utilization reach a certain threshold, or both. (The "or" in the UI operates as an `inclusive or`, `|`, `v`.) The amount of IOPS available to your deployment increases with disk size at a ratio of 10 IOPS for each GB.

> **Note:** CPU and RAM autoscaling is not supported on Isolated Compute. Disk autoscaling is available. If you provisioned an isolated instance or switched over from a deployment with autoscaling, monitor your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

The resource numbers refer to each database node in a deployment. For example, there are three members in an Elasticsearch cluster and if the deployment is scaled with 10 GB of disk and 1 GB of RAM, that means each member gets 10 GB of disk and 1 GB of RAM. The total resources added to your deployment is 30 GB of disk and 3 GB of RAM.

Autoscaling considerations

- Scaling your deployment up might cause your Elasticsearch to restart. If your scaled deployment needs to be moved to a host with more capacity, then the databases are restarted as part of the move.
- Disk cannot be scaled down.
- A few scaling operations can be more long running than others. Drastically increasing RAM or Disk can take longer than smaller increases to account for provisioning more underlying hardware resources.
- Autoscaling operations are logged in IBM Cloud® Activity Tracker Event Routing.
- Limits:
  - Can't set anything to scale in an interval less than 60 seconds.
  - Maximum Disk = 4 TB per member.
  - Maximum RAM = 112 GB per member.
- Autoscaling does not scale down deployments where disk or memory usage has shrunk. The RAM provisioned to your deployment remains for your future needs, or until you scale down your deployment manually. The disk provisioned to your deployment remains because disk cannot be scaled down.
- If you need to add resources to your deployment occasionally or rarely, you can manually scale your deployment.
- Elasticsearch is designed to balance work load across a cluster and can benefit from being horizontally scaled. If you are concerned about performance, check out Adding Elasticsearch Nodes.

Configuring Autoscaling in the UI

The Autoscaling panel is on the *Resources* tab of your deployment's *Manage* page. To enable scaling, enter your parameters. Then, check the boxes to enable the parameters you are using. Be sure to click **Save changes** for your configuration to be saved and your changes to take effect.

To disable autoscaling, clear the boxes for the parameters that you no longer want to use. If you clear all the boxes, autoscaling is disabled. Click **Save changes** to save the configuration.

Configuring Autoscaling in the CLI

You can get the autoscaling parameters for your deployment through the CLI by using the `cdb deployment-autoscaling` command.

```
$ ibmcloud cdb deployment-autoscaling <deployment name or CRN> member
```

To enable and set autoscaling parameters through the CLI, use a JSON object or file with the `cdb deployment-autoscaling-set` command.

```
$ ibmcloud cdb deployment-autoscaling-set <deployment name or CRN> member '{"autoscaling": { "memory": {"scalers": {"io_utilization": {"enabled": true, "over_period": "5m","above_percent": 90}},"rate": {"increase_percent": 10.0, "period_seconds": 300,"limit_mb_per_member": 125952,"units": "mb"}}}}'
```

CPU and RAM autoscaling is not supported on Isolated Compute. Disk autoscaling is available. If you provisioned an isolated instance or switched over from a deployment with autoscaling, monitor your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

Configuring Autoscaling in the API

You can get the autoscaling parameters for your deployment through the API by sending a `GET` request to the `/deployments/{id}/groups/{group_id}/autoscaling` endpoint.

```
$ curl -X GET -H "Authorization: Bearer $APIKEY" 'https://api.{region}.databases.cloud.ibm.com/v4/ibm/deployments/{id}/groups/member/autoscaling'
```

To enable and set the autoscaling parameters for your deployment through the API, send a `POST` request to the endpoint. Enabling autoscaling works by setting the `scalers` (`io_utilization` or `capacity`) to `true`.

```
$ curl -X PATCH https://api.{region}.databases.cloud.ibm.com/v4/ibm/deployments/{id}/groups/member/autoscaling -H 'Authorization: Bearer <>'
-H 'Content-Type: application/json'
-d '{"autoscaling": {
      "memory": {
        "scalers": {
          "io_utilization": {
            "enabled": true,
            "over_period": "5m",
            "above_percent": 90}
          },
          "limits": {
            "scale_increase_percent": 10.0,
            "scale_period_seconds": 30,
            "scale_maximum_mb": 125952,
            "units": "mb"
          }
      }
    }'
```

To disable autoscaling, send the PATCH request with the currently enabled scalers set to `false`. If all of them are set to `false`, then autoscaling is disabled on your deployment.

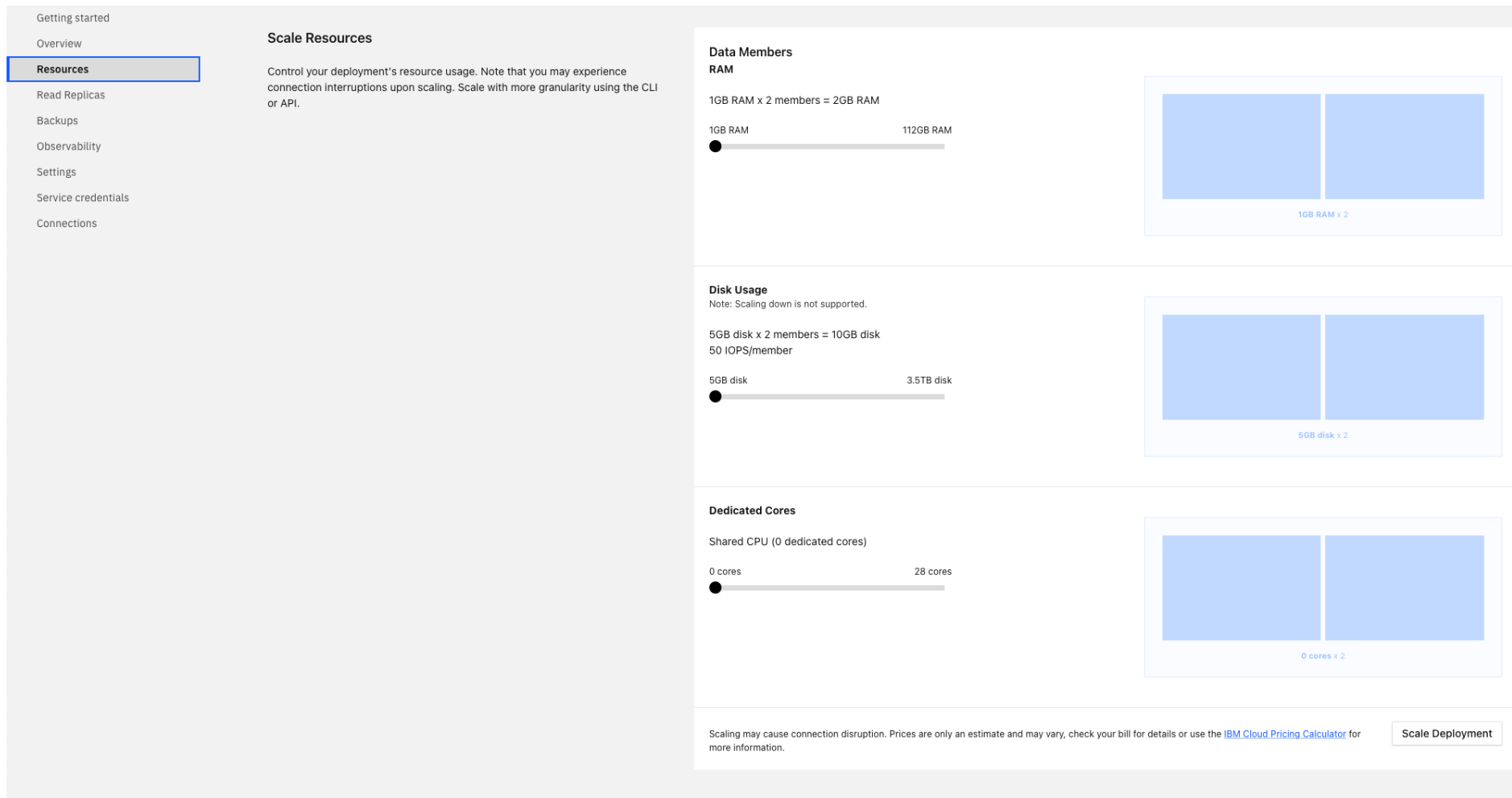
CPU and RAM autoscaling is not supported on Isolated Compute. Disk autoscaling is available. If you provisioned an isolated instance or switched over from a deployment with autoscaling, monitor your resources using IBM Cloud® Monitoring integration, which provides metrics for memory, disk space, and disk I/O utilization. To add resources to your instance, manually scale your deployment.

Adding Elasticsearch members

It is possible to scale your IBM Cloud® Databases for Elasticsearch deployment horizontally by adding more Elasticsearch members (also referred to as nodes). If your deployment starts to strain or slow down, adding members increases capacity and reliability. When a member is added, Elasticsearch automatically balances the workload across all the members in your deployment.

Note: It is not possible to decrease the amount of Elasticsearch members. As an alternative, you can use Backups and restore to a new instance with fewer members.

members that you add to your deployment are added with the amount of disk, memory, and CPU as the other members currently in your deployment. A visual representation of your data members and their resource allocation is available on the *Resources* tab of your deployment's *Manage* page. However, horizontal scaling is only available via the UI and the API.



The Scale Resources Pane

A default IBM Cloud® Databases for Elasticsearch deployment runs with three data members in a cluster, and resources are allocated to all three members equally. For example, the minimum storage of an Elasticsearch deployment is 15360 MB, which equates to an initial size of 5120 MB per member. The minimum RAM for an Elasticsearch deployment is 3072 MB, which equates to an initial allocation of 1028 MB per member. Adding a node adds another member with a size of 5120 MB of disk and 1028 MB of RAM, bringing your total resource usage for your deployment to 20480 MB of disk and 4096 MB of RAM.

Tip: Billing is based on the *total* amount of resources that are allocated to the service.

Adding members through the API

The *Foundation endpoint* that is shown on the *Overview* panel of your service provides the base URL to access this deployment through the API.

To view the current and scalable resources on a deployment, use the /deployments/{id}/groups.

```
$ curl -X GET -H "Authorization: Bearer $APIKEY" `https://api.{region}.databases.cloud.ibm.com/v4/ibm/deployments/{id}/groups'
```

To add members, use the /deployments/{id}/groups/{group_id} API endpoint, sending a PATCH request with the number of members you want in your deployment. The example request increases the number of members from the default of 3 to 5.

```
$ curl -X PATCH 'https://api.{region}.databases.cloud.ibm.com/v4/ibm/deployments/{id}/groups/member' \
-H 'Authorization: Bearer <>' \
-H 'Content-Type: application/json' \
-d '{"members": {"allocation_count": 5}}' \
```

Tip: When you use the CRN, remember to URL encode the CRN value as it might include the forward-slash (/) `%2F` character.

When properly encoded, a CRN that uses a forward-slash (/) character substitutes with a `%2F` character string. For example, see the following CRN.

```
$ crn:v1:bluemix:public:databases-for-redis:us-south:a/274074dce64e9c423ffc238516c755e1:29caf0e7-120f-4da8-9551-3abf57ebcfc7::
```

becomes

```
$ crn:v1:bluemix:public:databases-for-redis:us-south:a%2F274074dce64e9c423ffc238516c755e1:29caf0e7-120f-4da8-9551-3abf57ebcfc7::
```

Managing Cloud Databases backups

An automatically scheduled backup is taken of your database every day. You can also trigger on-demand backups at any time. Backups are encrypted either with an automatic key or your own key if you use Bring Your Own Key (BYOK). You can restore a backup to a new instance of Cloud Databases.

To access backups for Cloud Databases, go to your database instance's Dashboard, and see the *Backups and restore* tab.

Here is some additional general information about backups:

- Automatic backups are performed daily and kept with a simple retention schedule of 30 days.
- Backups cannot be deleted.
- If you delete your instance, its backups are deleted automatically.
- Daily backup scheduling is not configurable.
- Backups are restorable to other regions, except for `eu-de`, `eu-es`, and `par-01`, which can restore backups only between each other. For example, `par-01` backups can be restored to and between `eu-de` and `eu-es`.
- Backup storage is encrypted. To manage the encryption keys, see Key Protect integration. Otherwise, backups are encrypted with a key that is automatically generated for your instance.
- Backups are restorable across accounts, but only through the API and only if the user that is running the restore has access to both the source and destination accounts.
- Cloud Databases backups are not downloadable. If you need a local backup, use the appropriate software. For example, pg_dump is an effective tool for managing PostgreSQL backups.

For information on taking an on-demand backup, see Taking an on-demand backup.

Backups in the UI

In the UI, navigate to the *Backups and restore* tab where you see a table with all available backups for your database.

The backup types can be either *On-demand* or *Automatic*. Each backup is listed with its type and when the backup was taken.

Click the backup to reveal information for that specific backup, including its full ID. A **Restore** button or a pre-formatted CLI command is there for restore options.

Backups in the CLI

You can access the list of backups and individual backup information from the Cloud Databases CLI plug-in and the Cloud Databases API.

Use the `cdb deployment-backups-list` command to view the list of all available backups for your instance. To get details about a specific backup, use the `cdb backup-show` command.

For example, to view the backups for an instance named "example-instance", use the following command:

```
$ ibmcloud cdb deployment-backups-list <INSTANCE_NAME_OR_CRN>
```

To see the details of one of the backups from the list, take the ID from the `ID` field of the `deployment-backups-list` response and use it with the `backup-show` command:

```
$ ibmcloud cdb backup-show crn:v1:staging:public:cloud-databases:us-south:a/6284014dd5b487c87a716f48aeeaf99f:3b4537bf-a585-4594-8262-2b1e24e2701e:backup:a3364821-d061-413f-a0df-6ba0e2951566
```

Backups in the Cloud Databases API

For backups information in the Cloud Databases API, use the `/deployments/{id}/backups` endpoint to list the instance's backups. To get information about a specific backup, use the `/backups/{backup_id}` endpoint.

Taking an on-demand backup in the UI

If you plan to make major changes to your instance, like scaling or removing databases, tables, collections, on-demand backups are useful. It can also be useful if you need to back up on a schedule. On-demand backups are kept for 30 days.

> **Tip:** Instances come with backup storage equal to their total disk space at no cost. If your backup storage usage is greater than total disk space, each gigabyte is charged at an overage of $0.03/month. Backups are compressed, so even if you use on-demand backups, most instances do not exceed the allotted credit.

To create a manual backup in the UI, go to the *Backups and restore* tab of your instance then click **Create backup**. A message is displayed that a backup is in progress, and an on-demand backup is added to the list of available backups.

Taking an on-demand backup in the CLI

If you plan to make major changes to your instance, like scaling or removing databases, tables, collections, on-demand backups are useful. It can also be useful if you need to back up on a schedule. On-demand backups are kept for 30 days.

> **Tip:** Instances come with backup storage equal to their total disk space at no cost. If your backup storage usage is greater than total disk space, each gigabyte is charged at an overage of $0.03/month. Backups are compressed, so even if you use on-demand backups, most instances do not exceed the allotted credit.

In the CLI, an on-demand backup is triggered with the `cdb deployment-backup-now` command.

```
$ ibmcloud cdb deployment-backup-now <INSTANCE_NAME_OR_CRN>
```

Taking an on-demand backup in the API

If you plan to make major changes to your instance, like scaling or removing databases, tables, collections, on-demand backups are useful. It can also be useful if you need to back up on a schedule. On-demand backups are kept for 30 days.

> **Tip:** Instances come with backup storage equal to their total disk space at no cost. If your backup storage usage is greater than total disk space, each gigabyte is charged at an overage of $0.03/month. Backups are compressed, so even if you use on-demand backups, most instances do not exceed the allotted credit.

In the API, sending a POST to the `/deployments/{id}/backups` endpoint triggers an on-demand backup.

Restoring a backup

Backups are restored to a new instance. After the new instance finishes provisioning, your data in the backup file is restored into the new instance.

By default, the new instance is auto-sized to the same disk and memory allocation as the source instance at the time of the backup from which you are restoring. To adjust the resources that are allocated to the new instance, use the optional fields in the UI, CLI, or API to resize the new instance. Be sure to allocate enough for your data and workload; if the instance is not given enough resources, the restore fails.

Tip: Do not delete the source instance while the backup is restoring. Before you delete the old instance, wait until the new instance is provisioned and the backup is restored. Deleting an instance also deletes its backups.

Restoring a backup in the UI

To restore a backup to a new service instance,

1. Click in the corresponding row to expand the options for the backup that you want to restore.
2. Click **Restore**.
3. On the **Provisioning** page, select from some available options.
    - The new instance is automatically named `<name>-restore-[timestamp]`, but you can rename it.
    - You can also select the region where the new instance is located. Cross-region restores are supported, except for restoring into or out of the `eu-de` region.
    - You can choose the initial resource allocation, either to expand or shrink the resources on the new instance. You can also enable or disable dedicated cores. Note that if you decrease your resource amount, it may lead to provision failure or your database not functioning properly.
4. Click **Restore backup**. A "restore from backup started" message appears. Clicking **Your new instance is available now** takes you to your *Resources List*.

Restoring a backup in the CLI

The Resource Controller supports provisioning of database instances, and provisioning and restoring are the responsibility of the Resource Controller CLI. Use the resource service-instance-create command.

```
$ ibmcloud resource service-instance-create <INSTANCE_NAME> <SERVICE-ID> standard <REGION> --service-endpoints <ENDPOINT-TYPE> -p
'{"backup_id":"BACKUP_ID"}'
```

- Change the value of `instance_name` to the name that you want for your new instance.
- The `service-id` is the type of instance, such as *databases-for-postgresql* or *messages-for-rabbitmq*.
- The `region` is where you want the new instance to be located, which can be a different region from the source instance. Cross-region restores are supported, except for restoring to or from `eu-de` by using another region.
- `backup_id` is the backup that you want to restore.

The previous command will restore a backup to a machine of the same configuration and on the same hosting model as your original deployment.

Optional parameters

Optional parameters are available through the CLI. Use them if you need to customize resources, change the hosting model, or use a Key Protect key for BYOK encryption on the new instance. See the following example:

```
$ ibmcloud resource service-instance-create <INSTANCE_NAME> <SERVICE-ID> standard <REGION> -p
'{"backup_id":"BACKUP_ID","key_protect_key":"KEY_PROTECT_KEY_CRN", "members_disk_allocation_mb":"DESIRED_DISK_IN_MB", "members_host_flavor": "<VALUE>",
"members_memory_allocation_mb":"DESIRED_MEMORY_IN_MB", "members_cpu_allocation_count":"NUMBER_OF_CORES"}'
```

Note: The `members_host_flavor` value can be either "multitenant" or an appropriate-sized Isolated Compute host (see the list of available values). Only specify `members_memory_allocation_mb` or `members_cpu_allocation_count` if you use "multitenant" hosting.

Tip: A pre-formatted command for a specific backup is available in detailed view of the backup on the *Backups and restore* tab of your instance's dashboard.

By default, restoring from a backup provisions an instance with the preferred version of the database type, not the version of the instance you restore from. You can specify a version by adding the version in the parameters object, as in the following example.

```
$ `ibmcloud resource service-instance-create <INSTANCE_NAME> databases-for-mysql standard us-south -p '{"backup_id":"<BACKUP_ID>", "version": "<VERSION>"}'
```

To see a list of versions available, run `ibmcloud cdb deployables`.

Add `async_restore` parameter (new) - PostgreSQL only

A new optional parameter, `async_restore` was added to the restore `parameters` block.

`async_restore` (boolean) — default: false. When set to true, the restore is initiated as an asynchronous operation, which helps to reduce end-to-end restore time.

```
$ `ibmcloud resource service-instance-create <INSTANCE_NAME> databases-for-postgresql standard us-south -p '{"point_in_time_recovery_deployment_id":"
<SOURCE_CRN>", "point_in_time_recovery_time":"<PITR_TIME>", version": "<VERSION>", "async_restore": true }'
```

Example:

```
$ `ibmcloud resource service-instance-create <INSTANCE_NAME> databases-for-postgresql standard us-south -p '{"point_in_time_recovery_deployment_id":"test_crn",
"point_in_time_recovery_time":"2025-12-08T17:08:32Z", version": "17", "async_restore": true }'
```

An asynchronous restore can only be requested when the source and target PostgreSQL databases are running the same major version. Restores across different major versions are not supported. If the `async_restore` parameter is not specified, the service defaults to performing the restore synchronously, which is the current behavior.

Restoring a backup through the API

The Resource Controller API supports provisioning and restoring database instances. The create request is a `POST` to the `/resource_instances` endpoint.

```
$ curl -X POST \
  https://resource-controller.cloud.ibm.com/v2/resource_instances \
  -H 'Authorization: Bearer <>' \
  -H 'Content-Type: application/json' \
    -d '{
    "name": "<INSTANCE_NAME>",
    "target": "<REGION>",
    "resource_group": "<YOUR-RESOURCE-GROUP>",
    "resource_plan_id": "<SERVICE-ID>",
    "parameters":{
      "backup_id": "<BACKUP_ID>"
    }
  }'
```

> ⚠ **Important:** The parameters `name`, `target`, `resource_group`, and `resource_plan_id` are all required, and `backup_id` is the backup that you want to restore.

- Change the value of `name` to the name that you want for your new instance.
- The `resource_plan_id` is the type of instance, such as *databases-for-postgresql* or *messages-for-rabbitmq*.
- The `target` is the region where you want the new instance to be located, which can be a different region from the source instance. Cross-region restores are supported, except for restoring into or out of the `eu-de` region.
- `backup_id` is the backup that you want to restore.

The above command will restore a backup to a machine of the same configuration and on the same hosting model as your original deployment.

Optional parameters

Optional parameters are available through the API. Use them if you need to customize resources, change the hosting model, deploy to a specific version, or use a Key Protect key for BYOK encryption on the new instance.

If you need to adjust resources, add any of the optional parameters `key_protect_key`, `members_disk_allocation_mb`, `members_host_flavor`, `members_memory_allocation_mb`, `members_cpu_allocation_count`, or `version` and their preferred values to the body of the request. See the following example:

```
$ curl -X POST \
  https://resource-controller.cloud.ibm.com/v2/resource_instances \
  -H 'Authorization: Bearer <>' \
  -H 'Content-Type: application/json' \
    -d '{
    "name": "<INSTANCE_NAME>",
```

```
  "target": "<REGION>",
  "resource_group": "<YOUR-RESOURCE-GROUP>",
  "resource_plan_id": "<SERVICE-ID>",
  "parameters":{
    "backup_id": "<BACKUP_ID>",
    "members_host_flavor": "<members_host_flavor_value>",
    "version": "<VERSION_NUMBER>"
  }
}'
```

Note: The `members_host_flavor` value can be either "multitenant" or an appropriate-sized Isolated Compute host (see the list of available values). Only specify `members_memory_allocation_mb` or `members_cpu_allocation_count` if you use "multitenant" hosting.

Note: By default, restoring from a backup provisions an instance with the preferred version of the database type, not the version of the instance you restore from. You can specify a version by adding a `version` value in the parameters object.

Add async_restore parameter (new) - PostgreSQL only

A new optional parameter, `async_restore` was added to the restore `parameters` block.

`async_restore` (boolean) — default: false. When set to true, the restore is initiated as an asynchronous operation, which helps to reduce end-to-end restore time.

```
$ curl -X POST \
  https://resource-controller.cloud.ibm.com/v2/resource_instances \
  -H 'Authorization: Bearer <>' \
  -H 'Content-Type: application/json' \
    -d '{
    "name": "<INSTANCE_NAME>",
    "target": "<REGION>",
    "resource_group": "<YOUR-RESOURCE-GROUP>",
    "resource_plan_id": "<SERVICE-ID>",
    "parameters":{
      "point_in_time_recovery_deployment_id": "<SOURCE_CRN>",
      "point_in_time_recovery_time": "<PITR_TIME>",
      "version": "<VERSION_NUMBER>",
      "async_restore": true
    }
  }'
```

An asynchronous restore can only be requested when the source and target PostgreSQL databases are running the same major version. Restores across different major versions are not supported. If the `async_restore` parameter is not specified, the service defaults to performing the restore synchronously, which is the current behavior.

Restoring a backup through Terraform

Use Terraform to restore to a backup from an older version to a new version.

1. Set your `backup_id`. For more information, see `backup_id`.
2. Set your `version` in the version attribute. For more information, see `version`.

The code looks like:

```
resource "ibm_database" "<your-instance>" {
  name                          = "<your_database_name>"
  service                       = "<service>"
  plan                          = "<plan>"
  location                      = "<region>"
  version                       = "<version>"
  backup_id                     = "<backup_id>"
}
```

For more information, see the Cloud Databases Terraform Registry.

Fast PG Restore(async_restore) through Terraform - PostgreSQL only

1. A new optional parameter, `async_restore` was added to the block.
2. `async_restore` (boolean) — default: false. When set to true, the restore is initiated as an asynchronous operation, which helps to reduce end-to-end restore time.
3. This parameter is only applicable when restoring a PostgreSQL instance.

The code looks like the following:

```
data "ibm_resource_group" "group" {
  name = "<your_group>"
}

resource "ibm_database" "<your-instance>" {
  name                              = "<your_database_name>"
  location                          = "<region>"
  plan                              = "<plan>"
  service                           = "databases-for-postgresql"
  resource_group_id                 = data.ibm_resource_group.group.id
  service_endpoints                 = "private"
  async_restore                     = true
  point_in_time_recovery_time       = "<PITR_TIME>"
  point_in_time_recovery_deployment_id = "<SOURCE_CRN>"
  version                           = "<VERSION_NUMBER>"
}
```

An asynchronous restore can only be requested when the source and target PostgreSQL databases are running the same major version. Restores across different major versions are not supported. If the `async_restore` parameter is not specified, the service defaults to performing the restore synchronously, which is the current behavior.

Backups and restoration

- Cloud Databases are not responsible for restoration, timeliness, or validity of said backups.
- Actions that you take as a user can compromise the integrity of backups, such as under-allocating memory and disk. Users can monitor that backups are successful by using the API, and periodically restore a backup to ensure validity and integrity. Users can retrieve the most recent-scheduled backup details from the Cloud Databases CLI plug-in and the Cloud Databases API.
- As a managed service, Cloud Databases monitors the state of your backups and can attempt to remediate when possible. If you encounter issues from which you cannot recover, contact support for more help.

Backup locations

Backup location differs per database region. Ensure that the backup region location matches your data location requirements.

Instance region	Backup region
Dallas	US cross regional Object Storage
Washington D.C.	US cross regional Object Storage
London	EU cross regional Object Storage
Frankfurt	EU cross regional Object Storage
Tokyo	AP cross regional Object Storage
Osaka	AP cross regional Object Storage
Sydney	AP cross regional Object Storage
Toronto	Montreal Object Storage
Chennai	Chennai Object Storage

Sao Paolo	Sao Paolo Object Storage
Madrid	EU cross regional Object Storage

Instance and backup regions

For more details about Cloud Databases Object Storage locations, review the location's documentation.

Business continuity and disaster recovery

Cloud Databases provides mechanisms to protect your data and restore service functions. For more information (including Backup Storage Regions), see Understanding business continuity and disaster recovery for Cloud Databases .

Point-in-Time Recovery

With Point-in-Time Recovery (PITR), the instance continuously backs up incrementally and can replay transactions to bring a new instance that is restored from a backup to any point in the last 7 days. Cloud Databases offers Point-In-Time Recovery (PITR) for the following services:

- IBM Cloud® Databases for PostgreSQL
- IBM Cloud® Databases for MongoDB
- IBM Cloud® Databases for MySQL

Backups FAQ

For frequently asked questions about backups, see the Backups FAQ.

Configuring Elasticsearch

Elasticsearch plug-ins

Elasticsearch supports many plug-ins to extend its core functions. IBM Cloud® Databases for Elasticsearch comes with a selection of plug-ins that already enabled on your deployment. Plug-in management is handled by the service, and users cannot install, uninstall, enable, or disable plug-ins.

You can check the plug-ins that are installed in your Elasticsearch cluster by using the `/_nodes/plugins` cluster API endpoint. Example,

```
$ CURL_CA_BUNDLE=certificate.crt curl -u ibm_cloud_es_user:password https://f1b5e4a9-7179-4af2-a795-b7a0d71ec80a.974550db55eb4ec0983f023940bf637f.databases.appdomain.cloud:30909/_nodes/plugins
```

Available plug-ins

Name	Description
analysis-icu	The ICU Analysis plug-in integrates the Lucene ICU module into Elasticsearch, adding ICU-related analysis components.
analysis-kuromoji	The Japanese (kuromoji) Analysis plug-in integrates the Lucene kuromoji analysis module into Elasticsearch.
analysis-nori	The Korean (nori) Analysis plug-in integrates the Lucene nori analysis module into Elasticsearch.
analysis-phonetic	The Phonetic Analysis plug-in integrates a phonetic token filter analysis with Elasticsearch.
analysis-smartcn	Smart Chinese Analysis plug-in integrates the Lucene Smart Chinese analysis module into Elasticsearch.
analysis-stempel	The Stempel (Polish) Analysis plug-in integrates the Lucene stempel (polish) analysis module into Elasticsearch.
analysis-ukrainian	The Ukrainian Analysis plug-in integrates the Lucene UkrainianMorfologikAnalyzer into Elasticsearch.
ingest-attachment	An ingest processor that uses Apache Tika to extract contents.
mapper-size	The Mapper Size plug-in allows document to record their uncompressed size at index time.
repository-s3	The S3 repository plug-in adds the repositories that are used for automated backups of your data.

Available Elasticsearch plugins

More details on the plug-ins are available in the Elasticsearch documentation.

Uploading files to Elasticsearch

A few Elasticsearch features allow indexes to read from files on the file system, so IBM Cloud® Databases for Elasticsearch allows you to upload files to your deployment. The files are stored at a known location, and Elasticsearch is configured so that it is allowed to read files from the location.

Tip: Files that are uploaded to your deployment use disk resources, both in the index and on the file system. Make sure that you scale your deployment before uploading files.

Basic Process

1. You base64 encode the file on client side.
2. The base64 strings are stored as documents in an index named `ibm_file_sync` in your Elasticsearch deployment.
3. You trigger a file sync from the Cloud Databases API.
4. All nodes in your Elasticsearch cluster download the file contents from the index, decode the base64, and restore the files on the deployment's disk in the `/data/ibm_file_sync/current` directory.
5. At regular intervals, and on restarts, the files get resynced to assure they are present on all nodes.
6. Files that are on disk, but not in the index, get deleted. You can delete files from disk by removing them from the index.

The index in Elasticsearch is `ibm_file_sync`.
The location of the files on disk is `/data/ibm_file_sync/current`.

Note: In Elasticsearch 7 the API removes doc types. Refer to the Elasticsearch documentation for more details. This is especially good to keep in mind when updating from prior versions.

Uploading the files to the Index

The structure of the documents in the index is as follows: `name` is the file name of the file, `blob` is the base64-encoded file contents, and `md5` is an optional hash value over the file contents. The recommended mapping for the index is split out based on version.

For Elasticsearch 6:

```
$ curl -X PUT "https://user:password@host:port/ibm_file_sync" -H 'Content-Type: application/json' -d'
{
    "mappings": {
        "files": {
            "properties": {
                "name": {
                    "type": "text"
                },
                "blob": {
                    "type": "binary"
                },
                "md5": {
                    "type": "text"
                }
            }
        }
    }
}'
```

For Elasticsearch 7 (note the removal of the `files` section):

```
$ curl -X PUT "https://user:password@host:port/ibm_file_sync" -H 'Content-Type: application/json' -d'
{
  "mappings": {
    "properties": {
      "name": {
        "type": "text"
      },
      "blob": {
        "type": "binary"
      },
      "md5": {
        "type": "text"
      }
    }
  }
}'
```

Tip: The URL is the `https` connection string from your deployment.

To use the index, encode the file contents as base64. To encode an example file `README.md` in bash, `ENC=$(base64 -w 0 README.md)`. Then, build a checksum over the content, `HASH=$(md5sum README.md)`.

Tip: The download function compares the hash values on each sync run and if the values are unchanged since the last sync, no new download is attempted. If any document in the index has no md5 value, all downloads are attempted again.

Next, upload the document to the index. Note that the file name is supplied in the URL as well.

For Elasticsearch 6:

```
$ curl -X PUT "https://user:password@host:port/ibm_file_sync/files/README1.md" -H 'Content-Type: application/json' -d'
{
    "name": "README1.md",
    "blob": "\"$ENC\"",
    "md5": "\"$HASH\""
```

```
}'
```

For Elasticsearch 7 (note only the URL is changed):

```
$ curl -X PUT "https://user:password@host:port/ibm_file_sync/_doc/README1.md" -H 'Content-Type: application/json' -d'
{
    "name": "README1.md",
    "blob": '"\"$ENC\""',
    "md5": '"\"$HASH\""'
}'
```

You can verify the uploaded data.

For Elasticsearch 6:

```
$ curl https://user:password@host:port/ibm_file_sync/files/README.md?pretty
```

For Elasticsearch 7:

```
$ curl https://user:password@host:port/ibm_file_sync/_doc/README.md?pretty
```

If everything went smoothly, the returned data looks like this (shortened) example. The "md5" field can contain a file name alongside the hash.

For Elasticsearch 6:

```
$ {
  "_index" : "ibm_file_sync",
  "_type" : "files",
  "_id" : "README1.md",
  "_version" : 1,
  "found" : true,
  "_source" : {
    "name" : "README1.md",
    "blob" : "IyBF ... KWBgCg==",
    "md5" : "270f60e62d3d37add3702ced7f6969a1  README.md"
  }
}
```

For Elasticsearch 7:

```
$ {
  "_index" : "ibm_file_sync",
  "_id" : "README1.md",
  "_version" : 1,
  "found" : true,
  "_source" : {
    "name" : "README1.md",
    "blob" : "IyBF ... KWBgCg==",
    "md5" : "270f60e62d3d37add3702ced7f6969a1  README.md"
  }
}
```

Syncing files to disk

After the files are uploaded to the index, they can be synced to the disk. Call the `/elasticsearch/file_syncs` endpoint from the Cloud Databases API.

```
$ curl -X POST \
https://api.{region}.databases.cloud.ibm.com/v4/ibm/deployments/{id}/elasticsearch/file_syncs \
-H 'authorization: Bearer <token>'
```

The `region` is the region that your deployment is in, and the `id` (CRN) part of the URL needs to be url-encoded. More information is in the API Reference.

The call starts and returns a task so you can monitor its progress. After the returned task finishes, the contents in the index are present on all the nodes in your cluster.

Tip: Any number of files can be uploaded and synced. The contents of the files are not validated. Ensure that they can be processed by

Elasticsearch.

Using the files

Elasticsearch features that use files on the file system do so by accepting the path to the file when defining the index. An uploaded file `example.txt` is at `/data/ibm_file_sync/current/example.txt`. This list contains examples and is not exhaustive.

- Keep Words Token Filter
- Mapping Char Filter
- Compound Word Token Filters

Observability

Set up logging and monitoring

1. Use **IBM Cloud® Monitoring** to gain operational visibility into the performance and health of your applications, services, and platforms. For more information, see Cloud Databases IBM Cloud® Monitoring integration.
2. Use the **IBM® Cloud Logs** service to capture a record of your Cloud Databases activities and manage logs including audit and operational events. For more information, see Cloud Databases IBM Cloud® Activity Tracker Event Routing.
3. Use **IBM® Cloud Logs** to add log management capabilities to your Cloud Databases architecture. For more information, see Cloud Databases IBM® Cloud Logs.

Next steps

You provisioned Cloud Databases service, set up notifications, and set up monitoring. Now, work on Securing your service.

Logging for Cloud Databases

IBM Cloud® services, such as Cloud Databases, generate platform logs that you can use to investigate abnormal activity and critical actions in your account, and troubleshoot problems.

You can use **IBM® Cloud Logs Routing**, a platform service, to route platform logs in your account to a destination of your choice by configuring a tenant that defines where platform logs are sent. For more information, see About logs routing.

You can use **IBM® Cloud Logs** to visualize and alert on platform logs that are generated in your account and routed by IBM Cloud Logs Routing to an IBM Cloud Logs instance.

Locations where platform logs are generated

Locations where logs are sent by IBM Cloud Logs Routing

Cloud Databases sends logs by IBM Cloud Logs Routing in the regions that are indicated in the following table.

Dallas (us-south)	Washington (us-east)	Toronto (ca-tor)	Sao Paulo (br-sao)
Yes	Yes	Yes	Yes

Regions where platform logs are sent in Americas locations

Tokyo (jp-tok)	Sydney (au-syd)	Osaka (jp-osa)	Chennai (in-che)
Yes	Yes	Yes	No

Regions where platform logs are sent in Asia Pacific locations

Frankfurt (eu-de)	London (eu-gb)	Madrid (eu-es)	Paris (eu-par01)
Yes	Yes	Yes	No

Regions where platform logs are sent in Europe locations

Platform logs that are generated

Cloud Databases generates platform logs for the severity types debug, error, info, warning, and critical.

Platform logs from your database instances can be routed to any region supported as per the table above. Logs from Chennai (`in-che`) is routed to Tokyo (`jp-tok`) and Paris (`eu-par01`) is routed to Frankfurt (`eu-de`).

Enabling logging

Create IBM Cloud Logs and configure routing by setting the target between source location to target instance.

Viewing logs

Cloud Databases logs can be viewed in on the IBM Cloud Logs instance created. Go to the Logging instance page and click on *Dashboard*.

Launching IBM Cloud Logs from the Cloud Databases page

Users can visit the Cloud Databases instance. Click on *Overview* and scroll to the *Observability* section. Click on *IBM Cloud Logs* to view your logging instances. Click on *Dashboard* to access the logs.

Launching IBM Cloud Logs from the Observability page

For more information about launching the IBM Cloud Logs UI, see Launching the UI in the IBM Cloud Logs documentation.

Fields by log type

For information about fields included in every platform log, see Fields for platform logs.

Cloud Databases logs include the following fields.

Field	Type	Description
logSourceCRN	Required	Defines the account and flow log instance where the log is published.
saveServiceCopy	Required	Defines whether IBM saves a copy of the record for operational purposes.
message	Required	Description of the log that is generated.
messageID	Required	ID of the log that is generated.

Log record fields

Cloud Databases sends audit events as platform logs. For more information, see Activity tracking for Cloud Databases.

Analyzing Cloud Databases logs

In the IBM Cloud Logs Dashboard, users can filter based on *Application*, *Subsystem*, *Severity* to find logs specific to an instance. Users can also create a custom dashboard, view logs or write a query to search for a log data. example label.region:"us-south"

They can also create *alerts* in the IBM Cloud Logs.

Monitoring integration

Monitoring for Cloud Databases is provided through integration with the IBM Cloud® Monitoring service. Your instances forward select information so that you can monitor instance health and resource usage. To start collecting and viewing monitoring data, follow the instructions to enable Platform Metrics. Platform Metrics need to be enabled in the same region as your instance. Use the Metrics Router to configure which Sysdig instance your platform metrics flows to. For more information, see IBM Cloud Metrics Routing.

You can then access your monitoring dashboard for each region from the IBM Cloud Monitoring area in the Cloud console (under Observability).

> **Note:** IBM Cloud Monitoring is available for instances in every region. Instances in Multi-zone Regions (MZRs) - `eu-gb`, `eu-de`, `us-east`, `us-south`, `jp-tok`, `au-syd` - have their metrics in the same region. If you have instances that are in a Single-zone Region (SZR) (e.g. `che01`) then your logs are forwarded to an IBM Cloud Monitoring instance in another region. You need to provision monitoring instances in the region where your metrics are forwarded to. Metrics for instances in `che01` go to `jp-tok`.

Use IBM Cloud Monitoring dashboards to monitor your environments and applications. IBM Cloud Monitoring dashboards are designed around time. Select your dashboard based on specific data gathered over a set time range.

Common metrics

Here is a detailed description about two of the common metrics across all Cloud Databases offerings.

CPU cores used per member

The usage that is presented in this dashboard is the number of CPU cores used per member, which is measured in core seconds. This metric is available for all hosting models; you can monitor this metric for both, databases that are hosted either as a single-tenant on underlying hardware and databases running on multi-tenant hosts. We recommend that you use this metric to track historical CPU allocation over time, which can help you to decide how many

CPU cores to allocate for your database to match desired performance.

CPU used per member (data only available with dedicated cores)

The usage that is presented in this dashboard is a percentage of total CPU being used, based on the number of cores in your Cloud Databases instance. For example, if you have 8 cores and your usage is 12.5%, then that percentage reflects that your database member is using 1 core's worth of CPU seconds. However, this does not guarantee that your member's workload is pinned to 1 core – the workload might be distributed unevenly among your 8 cores. In the same example, 25% usage reflects that your database member is using 2 core's worth of CPU seconds out of your available 8 cores.

> **Note:** The title of this metric specifies "data only available with dedicated cores," which is an anachronism. This panel now displays information about instances using the Isolated Compute hosting model; however, it will not display information about any instances on the legacy hosting model once referred to as "dedicated cores." Dedicated cores were deprecated during the hosting model transition outlined here, so there should be no instances using the hosting model anymore. This panel contains a subset of the metrics that are visible in the newer panel, **CPU used per member (all instance types)**.

CPU used per member (all instance types)

The metrics in this panel cover instances of both the Isolated Compute and Shared Compute hosting models. The usage that is presented in this dashboard is a percentage of total CPU being used, based on the number of cores in your Cloud Databases instance. For example, if you have 8 cores and your usage is 12.5%, then that percentage reflects that your database member is using 1 core's worth of CPU seconds. However, this does not guarantee that your member's workload is pinned to 1 core – the workload might be distributed unevenly among your 8 cores. In the same example, 25% usage reflects that your database member is using 2 core's worth of CPU seconds out of your available 8 cores.

Metrics available by service plan

In addition to the above metrics, each database service has its own set of metrics that can be monitored.

MongoDB metrics

Metric name
MongoDB Average time spent acquiring locks in microseconds
MongoDB Average time spent acquiring locks in microseconds
MongoDB Connections
MongoDB Disk read latency mean
MongoDB Disk write latency mean
MongoDB IO utilization in percent 15-minute average
MongoDB IO utilization in percent 30-minute average
MongoDB IO utilization in percent 5-minute average
MongoDB IO utilization in percent 60-minute average
MongoDB IOPS read and write total count for an instance
MongoDB Maximum allowed memory for an instance
MongoDB Oplog gigabyte per hour
MongoDB Oplog used bytes
MongoDB Oplog used bytes percent of total

MongoDB Oplog window hours
MongoDB Page faults
MongoDB Process resident memory in bytes
MongoDB Process virtual memory in bytes
MongoDB Replica set member state
MongoDB Replication lag
MongoDB Total disk space for an instance
MongoDB Used CPU for an instance
MongoDB Used disk space for an instance
MongoDB Used disk space for an instance
MongoDB Used memory for an instance
MongoDB Used memory for an instance

Metrics available by plan names

MongoDB metric descriptions

MongoDB Average time spent acquiring locks in microseconds total W-average

Average time spent acquiring exclusive (W) locks in microseconds

Metadata	Description
Metric Name	ibm_databases_for_mongodb_locks_time_acquiring_microseconds_W_average
Metric Type	gauge
Value Type	second
Segment By	Service instance, Service instance name

Average time spent acquiring exclusive (W) locks in microseconds metric metadata

MongoDB Average time spent acquiring locks in microseconds total_average

Average time spent acquiring locks in microseconds

Metadata	Description
Metric Name	ibm_databases_for_mongodb_locks_time_acquiring_microseconds_total_average
Metric Type	gauge
Value Type	second
Segment By	Service instance, Service instance name

Average time spent acquiring locks in microseconds metric metadata

MongoDB Connections

The number of connections to the database

Metadata	Description
Metric Name	ibm_databases_for_mongodb_connections
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Connections metric metadata

MongoDB Disk read latency mean

Disk read latency mean

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_read_latency_mean
Metric Type	gauge
Frequency	60ms
Value Type	count
Segment By	Service instance, Service instance name

Disk read latency mean metric metadata

MongoDB Disk write latency mean

Disk write latency mean

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_write_latency_mean
Metric Type	gauge
Frequency	60ms
Value Type	count
Segment By	Service instance, Service instance name

Disk write latency mean metric metadata

MongoDB IO utilization in percent 15-minute average

How much disk I/O has been used over 15 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_io_utilization_percent_average_15m
Metric Type	gauge

Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 15 minute average metric metadata

MongoDB IO utilization in percent 30-minute average

How much disk I/O has been used over 30 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_io_utilization_percent_average_30m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 30 minute average metric metadata

MongoDB IO utilization in percent 5-minute average

How much disk I/O has been used over 5 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_io_utilization_percent_average_5m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 5 minute average metric metadata

MongoDB IO utilization in percent 60-minute average

How much disk I/O has been used over 60 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_io_utilization_percent_average_60m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 60 minute average metric metadata

MongoDB IOPS read & write total count for an instance

How many input-output operations per second your instance is performing

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_iops_read_write_total

Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

IOPS read & write total count for an instance metric metadata

MongoDB Maximum allowed memory for an instance

The maximum amount of memory available to your instance

Metadata	Description
Metric Name	ibm_databases_for_mongodb_memory_limit_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Maximum allowed memory for an instance metric metadata

MongoDB Oplog gigabyte per hour

The gigabytes of oplog per hour the primary generates

Metadata	Description
Metric Name	ibm_databases_for_mongodb_oplog_gb_per_hour
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Oplog gigabyte per hour metric metadata

MongoDB Oplog used bytes

The total amount of space used by the oplog in bytes.

Metadata	Description
Metric Name	ibm_databases_for_mongodb_oplog_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Oplog used bytes metric metadata

MongoDB Oplog used bytes percent of total

The total used oplog space in percent

Metadata	Description

Metric Name	ibm_databases_for_mongodb_oplog_used_bytes_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Oplog used bytes percent of total metric metadata

MongoDB Oplog window hours

The approximate number of hours available in the oplog.

Metadata	Description
Metric Name	ibm_databases_for_mongodb_oplog_window_hours
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Oplog window hours metric metadata

MongoDB Page faults

The number of times per second that MongoDB had to request data from disk. Scale RAM to reduce the number of disk requests

Metadata	Description
Metric Name	ibm_databases_for_mongodb_page_faults
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Page faults metric metadata

MongoDB Process resident memory in bytes

Amount of actual physical memory used by the MongoDB process.

Metadata	Description
Metric Name	ibm_databases_for_mongodb_process_resident_memory_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Process resident memory in bytes metric metadata

MongoDB Process virtual memory in bytes

Amount of virtual memory used by the MongoDB process.

Metadata	Description
Metric Name	ibm_databases_for_mongodb_process_virtual_memory_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Process virtual memory in bytes metric metadata

MongoDB Replica set member state

An integer between 0 and 10 that represents the replica state of the current member.

Metadata	Description
Metric Name	ibm_databases_for_mongodb_status
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

Replica set member state metric metadata

MongoDB Replication lag

The replication lag in seconds

Metadata	Description
Metric Name	ibm_databases_for_mongodb_replica_lag
Metric Type	gauge
Value Type	second
Segment By	Service instance, Service instance name

Replication lag metric metadata

MongoDB Total disk space for an instance

Represents the total amount of disk space available to your deployment

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_total_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Total disk space for an instance metric metadata

MongoDB Used CPU for an instance

How much CPU is used as a percentage of total CPU available. Only for deployments that have dedicated CPU

Metadata	Description
Metric Name	ibm_databases_for_mongodb_cpu_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used CPU for an instance metric metadata

MongoDB Used disk space for an instance bytes

How much disk space your instance is using

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

MongoDB Used disk space for an instance

How much disk space is used as a percentage of total disk available

Metadata	Description
Metric Name	ibm_databases_for_mongodb_disk_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

MongoDB Used memory for an instance

How much memory your instance is using

Metadata	Description
Metric Name	ibm_databases_for_mongodb_memory_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

MongoDB Used memory for an instance percent

How much memory is used as a percentage of total memory available

Metadata	Description
Metric Name	ibm_databases_for_mongodb_memory_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

PostgreSQL Metrics

Metric Name
PostgreSQL Cache hit ratio
PostgreSQL Disk read latency mean
PostgreSQL Disk write latency mean
PostgreSQL IO utilization in percent 15-minute average
PostgreSQL IO utilization in percent 30-minute average
PostgreSQL IO utilization in percent 5-minute average
PostgreSQL IO utilization in percent 60-minute average
PostgreSQL IOPS read & write total count for an instance
PostgreSQL Maximum allowed memory for an instance
PostgreSQL Read replica replication lag
PostgreSQL Successful archive rate
PostgreSQL Temporary files size in bytes
PostgreSQL The total number of PostgreSQL connections being used
PostgreSQL Total disk space for an instance
PostgreSQL Transaction commit count
PostgreSQL Transaction commit rate
PostgreSQL Transaction rollback count
PostgreSQL Transaction rollback rate
PostgreSQL Tuples deleted count

PostgreSQL Tuples deleted rate
PostgreSQL Tuples fetched count
PostgreSQL Tuples fetched rate
PostgreSQL Tuples inserted count
PostgreSQL Tuples inserted rate
PostgreSQL Tuples returned rate
PostgreSQL Tuples updated count
PostgreSQL Tuples updated rate
PostgreSQL Used CPU for an instance
PostgreSQL Used disk space for an instance
PostgreSQL Used disk space for an instance
PostgreSQL Used memory for an instance
PostgreSQL Used memory for an instance
PostgreSQL WAL logs used bytes
PostgreSQL Blocks hit rate
PostgreSQL Blocks read rate
PostgreSQL Buffers backend rate
PostgreSQL Buffers checkpoint rate
PostgreSQL Deadlocks count
PostgreSQL Deadlocks rate

Metrics Available by Plan Names

PostgreSQL Metric Descriptions

PostgreSQL Blocks hit rate

Blocks hit rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_blocks_hit_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Blocks hit rate metric metadata

PostgreSQL Blocks read rate

Blocks read rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_blocks_read_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Blocks read rate metric metadata

PostgreSQL Buffers backend rate

Buffers backend rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_buffers_backend_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Buffers backend rate metric metadata

PostgreSQL Buffers checkpoint rate

Buffers checkpoint rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_buffers_checkpoint_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Buffers checkpoint rate metric metadata

PostgreSQL Cache hit ratio

Cache hit ratio

Metadata	Description
Metric Name	ibm_databases_for_postgresql_cache_hit_ratio
Metric Type	gauge
Value Type	percent

Segment By	Service instance, Service instance name

Cache hit ratio metric metadata

PostgreSQL Deadlocks count

Deadlocks count

Metadata	Description
Metric Name	ibm_databases_for_postgresql_deadlocks_count
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Deadlocks count metric metadata

PostgreSQL Deadlocks rate

Deadlocks rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_deadlocks_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Deadlocks rate metric metadata

PostgreSQL Disk read latency mean

Disk read latency mean

Metadata	Description
Metric Name	ibm_databases_for_postgresql_disk_read_latency_mean
Metric Type	gauge
Frequency	60ms
Value Type	count
Segment By	Service instance, Service instance name

Disk read latency mean metric metadata

PostgreSQL Disk write latency mean

Disk write latency mean

Metadata	Description
Metric Name	ibm_databases_for_postgresql_disk_write_latency_mean

Metric Type	gauge
Frequency	60ms
Value Type	count
Segment By	Service instance, Service instance name

Disk write latency mean metric metadata

PostgreSQL IO utilization in percent 15-minute average

How much disk I/O has been used over 15 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_postgresql_disk_io_utilization_percent_average_15m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 15 minute average metric metadata

PostgreSQL IO utilization in percent 30-minute average

How much disk I/O has been used over 30 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_postgresql_disk_io_utilization_percent_average_30m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 30 minute average metric metadata

PostgreSQL IO utilization in percent 5-minute average

How much disk I/O has been used over 5 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_postgresql_disk_io_utilization_percent_average_5m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 5 minute average metric metadata

PostgreSQL IO utilization in percent 60-minute average

How much disk I/O has been used over 60 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_postgresql_disk_io_utilization_percent_average_60m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 60 minute average metric metadata

PostgreSQL IOPS read & write total count for an instance

How many input-output operations per second your instance is performing

Metadata	Description
Metric Name	ibm_databases_for_postgresql_disk_iops_read_write_total
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

IOPS read & write total count for an instance metric metadata

PostgreSQL Maximum allowed memory for an instance

The maximum amount of memory available to your instance

Metadata	Description
Metric Name	ibm_databases_for_postgresql_memory_limit_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Maximum allowed memory for an instance metric metadata

PostgreSQL Read replica replication lag

How far behind a PostgreSQL read-only replica is, in bytes

Metadata	Description
Metric Name	ibm_databases_for_postgresql_read_replica_replication_lag_bytes
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Read replica replication lag metric metadata

PostgreSQL Successful archive rate

Successful archive rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_successful_archive_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Successful archive rate metric metadata

PostgreSQL Temporary files size in bytes

Temporary files size in bytes

Metadata	Description
Metric Name	ibm_databases_for_postgresql_temp_bytes_count
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Temporary files size in bytes metric metadata

PostgreSQL The total number of PostgreSQL connections being used

The total number of PostgreSQL connections being used

Metadata	Description
Metric Name	ibm_databases_for_postgresql_total_connections
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The total number of PostgreSQL connections being used metric metadata

PostgreSQL Total disk space for an instance

Represents the total amount of disk space available to your deployment

Metadata	Description
Metric Name	ibm_databases_for_postgresql_disk_total_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Total disk space for an instance metric metadata

PostgreSQL Transaction commit count

Transaction commit count

Metadata	Description
Metric Name	ibm_databases_for_postgresql_transaction_commit_count
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Transaction commit count metric metadata

PostgreSQL Transaction commit rate

Transaction commit rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_transaction_commit_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Transaction commit rate metric metadata

PostgreSQL Transaction rollback count

Transaction rollback count

Metadata	Description
Metric Name	ibm_databases_for_postgresql_transaction_rollback_count
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Transaction rollback count metric metadata

PostgreSQL Transaction rollback rate

Transaction rollback rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_transaction_rollback_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Transaction rollback rate metric metadata

Transaction rollback rate metric metadata

PostgreSQL Tuples deleted count

Tuples deleted count

Metadata	Description
Metric Name	ibm_databases_for_postgresql_tuples_deleted_count
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Tuples deleted count metric metadata

PostgreSQL Tuples deleted rate

Tuples deleted rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_tuples_deleted_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Tuples deleted rate metric metadata

PostgreSQL Tuples fetched count

Tuples fetched count

Metadata	Description
Metric Name	ibm_databases_for_postgresql_tuples_fetched_count
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Tuples fetched count metric metadata

PostgreSQL Tuples fetched rate

Tuples fetched rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_tuples_fetched_rate
Metric Type	gauge
Value Type	rate

Segment By	Service instance, Service instance name

Tuples fetched rate metric metadata

PostgreSQL Tuples inserted count

Tuples inserted count

Metadata	Description
Metric Name	ibm_databases_for_postgresql_tuples_inserted_count
Metric Type	gauge
Value Type	none
Segment By	Service instance, Service instance name

Tuples inserted count metric metadata

PostgreSQL Tuples inserted rate

Tuples inserted rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_tuples_inserted_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Tuples inserted rate metric metadata

PostgreSQL Tuples returned rate

Tuples returned rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_tuples_returned_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Tuples returned rate metric metadata

PostgreSQL Tuples updated count

Tuples updated count

Metadata	Description
Metric Name	ibm_databases_for_postgresql_tuples_updated_count
Metric Type	gauge

Value Type	none
Segment By	Service instance, Service instance name

Tuples updated count metric metadata

PostgreSQL Tuples updated rate

Tuples updated rate

Metadata	Description
Metric Name	ibm_databases_for_postgresql_tuples_updated_rate
Metric Type	gauge
Value Type	rate
Segment By	Service instance, Service instance name

Tuples updated rate metric metadata

PostgreSQL Used CPU for an instance

How much CPU is used as a percentage of total CPU available. This metric is available only for the Dedicated Hosting Model. For deployments on the Shared Hosting Model, use the `ibm_databases_for_postgresql_cpu_usage_seconds` metric to assess database CPU utilization. Applying the `average_over_time` function to this metric can provide meaningful insights.

Metadata	Description
Metric Name	ibm_databases_for_postgresql_cpu_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used CPU for an instance metric metadata

PostgreSQL Used disk space for an instance bytes

How much disk space your instance is using

Metadata	Description
Metric Name	ibm_databases_for_postgresql_disk_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

PostgreSQL Used disk space for an instance

How much disk space is used as a percentage of total disk available

Metadata	Description

Metric Name	ibm_databases_for_postgresql_disk_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

PostgreSQL Used memory for an instance bytes

How much memory your instance is using

Metadata	Description
Metric Name	ibm_databases_for_postgresql_memory_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

PostgreSQL Used memory for an instance percent

How much memory is used as a percentage of total memory available

Metadata	Description
Metric Name	ibm_databases_for_postgresql_memory_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

PostgreSQL WAL logs used bytes

How much WAL log file uses, in bytes

Metadata	Description
Metric Name	ibm_databases_for_postgresql_wal_used_bytes
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

WAL logs used bytes metric metadata

MySQL Metrics

Metric Name

MySQL Cache hit ratio
MySQL Connection usage for an instance
MySQL Disk read latency mean
MySQL Disk write latency mean
MySQL IO utilization in percent 15-minute average
MySQL IO utilization in percent 30-minute average
MySQL IO utilization in percent 5-minute average
MySQL IO utilization in percent 60-minute average
MySQL IOPS read & write total count for an instance
MySQL Maximum allowed memory for an instance
MySQL The maximum permitted number of simultaneous client connections.
MySQL Percent of threads connected
MySQL Percent of threads running
MySQL The number of connections that were aborted because the client died without closing the connection properly
MySQL The number of threads created to handle connections
MySQL The number of threads in the thread cache
MySQL The number of threads in the thread cache
MySQL The open file usage
MySQL The pool hit rate
MySQL The pool utilization
MySQL The rate of bytes received from all clients
MySQL The rate of bytes sent to all clients
MySQL The rate of failed attempts to connect to the MySQL server
MySQL The rate of joins that did a full scan of the first table
MySQL The rate of joins that perform table scans because they do not use indexes
MySQL The rate of joins that used a range search on a reference table
MySQL The rate of joins that used ranges on the first table
MySQL The rate of joins without keys that check for key usage after each row

MySQL The rate of merge passes that the sort algorithm has had to do
MySQL The rate of queries that have taken more than long_query_time seconds
MySQL The rate of sorted rows
MySQL The rate of sorts that were done by scanning the table
MySQL The rate of sorts that were done using ranges
MySQL The rate of statements executed by the server
MySQL The rate of times that a request for a table lock could be granted immediately
MySQL The rate of times that a request for a table lock could not be granted immediately and a wait was needed
MySQL The rate of times that the log buffer was too small and a wait was required for it to be flushed before continuing
MySQL The rate of total command statements executed
MySQL Total disk space for an instance
MySQL Used CPU for an instance
MySQL Used disk space for an instance
MySQL Used disk space for an instance
MySQL Used memory for an instance
MySQL Used memory for an instance
MySQL Total active connections to the database
MySQL Replica lag
MySQL Replica state

Metrics Available by Plan Names

MySQL Metrics Descriptions

MySQLCache hit ratio

Cache hit ratio

Metadata	Description
Metric Name	ibm_databases_for_mysql_cache_hit_ratio
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Cache hit ratio metric metadata

MySQL Connection usage for an instance

Represents the connection usage for your deployment.

Metadata	Description
Metric Name	ibm_databases_for_mysql_connection_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Connection usage for an instance metric metadata

MySQL Disk read latency mean

Disk read latency mean

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_read_latency_mean
Metric Type	gauge
Frequency	60ms
Value Type	count
Segment By	Service instance, Service instance name

Disk read latency mean metric metadata

MySQL Disk write latency mean

Disk write latency mean

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_write_latency_mean
Metric Type	gauge
Frequency	60ms
Value Type	count
Segment By	Service instance, Service instance name

Disk write latency mean metric metadata

MySQL IO utilization in percent 15-minute average

How much disk I/O has been used over 15 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_io_utilization_percent_average_15m
Metric Type	gauge

Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 15 minute average metric metadata

MySQL IO utilization in percent 30-minute average

How much disk I/O has been used over 30 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_io_utilization_percent_average_30m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 30 minute average metric metadata

MySQL IO utilization in percent 5-minute average

How much disk I/O has been used over 5 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_io_utilization_percent_average_5m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 5 minute average metric metadata

MySQL IO utilization in percent 60-minute average

How much disk I/O has been used over 60 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_io_utilization_percent_average_60m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 60 minute average metric metadata

MySQL IOPS read & write total count for an instance

How many input-output operations per second your instance is performing

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_iops_read_write_total

Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

IOPS read & write total count for an instance metric metadata

MySQL Maximum allowed memory for an instance

The maximum amount of memory available to your instance

Metadata	Description
Metric Name	ibm_databases_for_mysql_memory_limit_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Maximum allowed memory for an instance metric metadata

MySQL Maximum permitted number of simultaneous client connections.

Represents the maximum permitted number of simultaneous client connections.

Metadata	Description
Metric Name	ibm_databases_for_mysql_max_connections
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The maximum permitted number of simultaneous client connections. metric metadata

MySQL Percent of threads connected

Percent of threads connected.

Metadata	Description
Metric Name	ibm_databases_for_mysql_threads_connected_usage
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Percent of threads connected metric metadata

MySQL Percent of threads running

Percent of threads running.

Metadata	Description

Metric Name	ibm_databases_for_mysql_threads_running_usage
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Percent of threads running metric metadata

MySQL The number of connections that were aborted because the client died without closing the connection properly

The number of connections that were aborted because the client died without closing the connection properly.

Metadata	Description
Metric Name	ibm_databases_for_mysql_aborted_clients_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The number of connections that were aborted because the client died without closing the connection properly metric metadata

MySQL The number of threads created to handle connections

The number of threads created to handle connections.

Metadata	Description
Metric Name	ibm_databases_for_mysql_threads_created
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The number of threads created to handle connections metric metadata

MySQL The number of threads in the thread cache size

The number of threads in the thread cache.

Metadata	Description
Metric Name	ibm_databases_for_mysql_thread_cache_size
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The number of threads in the thread cache metric metadata

MySQL The number of threads in the thread cache

The number of threads in the thread cache.

Metadata	Description
Metric Name	ibm_databases_for_mysql_threads_cached
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The number of threads in the thread cache metric metadata

MySQL The open file usage

The open file usage.

Metadata	Description
Metric Name	ibm_databases_for_mysql_open_file_usage
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

The open file usage metric metadata

MySQL The pool hit rate

The pool hit rate.

Metadata	Description
Metric Name	ibm_databases_for_mysql_pool_hit_rate
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

The pool hit rate metric metadata

MySQL The pool utilization

The pool utilization.

Metadata	Description
Metric Name	ibm_databases_for_mysql_pool_utilization
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

The pool utilization metric metadata

MySQL The rate of bytes received from all clients

The rate of bytes received from all clients.

Metadata	Description
Metric Name	ibm_databases_for_mysql_bytes_received_rate
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

The rate of bytes received from all clients metric metadata

MySQL The rate of bytes sent to all clients

The rate of bytes sent to all clients.

Metadata	Description
Metric Name	ibm_databases_for_mysql_bytes_sent_rate
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

The rate of bytes sent to all clients metric metadata

MySQL The rate of failed attempts to connect to the MySQL server

The rate of failed attempts to connect to the MySQL server.

Metadata	Description
Metric Name	ibm_databases_for_mysql_aborted_connects_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of failed attempts to connect to the MySQL server metric metadata

MySQL The rate of joins that did a full scan of the first table

The rate of joins that did a full scan of the first table.

Metadata	Description
Metric Name	ibm_databases_for_mysql_select_scan_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of joins that did a full scan of the first table metric metadata

MySQL The rate of joins that perform table scans because they do not use indexes

The rate of joins that perform table scans because they do not use indexes.

Metadata	Description
Metric Name	ibm_databases_for_mysql_select_full_join_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of joins that perform table scans because they do not use indexes metric metadata

MySQL The rate of joins that used a range search on a reference table

The rate of joins that used a range search on a reference table.

Metadata	Description
Metric Name	ibm_databases_for_mysql_select_full_range_join_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of joins that used a range search on a reference table metric metadata

MySQL The rate of joins that used ranges on the first table

The rate of joins that used ranges on the first table.

Metadata	Description
Metric Name	ibm_databases_for_mysql_select_range_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of joins that used ranges on the first table metric metadata

MySQL The rate of joins without keys that check for key usage after each row

The rate of joins without keys that check for key usage after each row.

Metadata	Description
Metric Name	ibm_databases_for_mysql_select_range_check_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of joins without keys that check for key usage after each row metric metadata

The rate of joins without keys that check for key usage after each row metric metadata

MySQL The rate of merge passes that the sort algorithm has had to do

The rate of merge passes that the sort algorithm has had to do.

Metadata	Description
Metric Name	ibm_databases_for_mysql_sort_merge_passes_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of merge passes that the sort algorithm has had to do metric metadata

MySQL The rate of queries that have taken more than long_query_time seconds

The rate of queries that have taken more than long_query_time seconds.

Metadata	Description
Metric Name	ibm_databases_for_mysql_slow_queries_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of queries that have taken more than long_query_time seconds metric metadata

MySQL The rate of sorted rows

The rate of sorted rows.

Metadata	Description
Metric Name	ibm_databases_for_mysql_sort_rows_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of sorted rows metric metadata

MySQL The rate of sorts that were done by scanning the table

The rate of sorts that were done by scanning the table.

Metadata	Description
Metric Name	ibm_databases_for_mysql_sort_scan_rate
Metric Type	gauge
Value Type	count

Segment By	Service instance, Service instance name

The rate of sorts that were done by scanning the table metric metadata

MySQL The rate of sorts that were done using ranges

The rate of sorts that were done using ranges.

Metadata	Description
Metric Name	ibm_databases_for_mysql_sort_range_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of sorts that were done using ranges metric metadata

MySQL The rate of statements executed by the server

The rate of statements executed by the server.

Metadata	Description
Metric Name	ibm_databases_for_mysql_questions_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of statements executed by the server metric metadata

MySQL The rate of times that a request for a table lock could be granted immediately

The rate of times that a request for a table lock could be granted immediately.

Metadata	Description
Metric Name	ibm_databases_for_mysql_table_locks_immediate_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of times that a request for a table lock could be granted immediately metric metadata

MySQL The rate of times that a request for a table lock could not be granted immediately and a wait was needed

The rate of times that a request for a table lock could not be granted immediately and a wait was needed.

Metadata	Description
Metric Name	ibm_databases_for_mysql_table_locks_waited_rate

Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of times that a request for a table lock could not be granted immediately and a wait was needed metric metadata

MySQL The rate of times that the log buffer was too small and a wait was required for it to be flushed before continuing

The rate of times that the log buffer was too small and a wait was required for it to be flushed before continuing.

Metadata	Description
Metric Name	ibm_databases_for_mysql_innodb_log_waits_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of times that the log buffer was too small and a wait was required for it to be flushed before continuing metric metadata

MySQL The rate of total command statements executed

The rate of total command statements executed.

Metadata	Description
Metric Name	ibm_databases_for_mysql_commands_total_rate
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

The rate of total command statements executed metric metadata

MySQL Total disk space for an instance

Represents the total amount of disk space available to your deployment

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_total_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Total disk space for an instance metric metadata

MySQL Used CPU for an instance

How much CPU is used as a percentage of total CPU available. Only for deployments that have dedicated CPU

Metadata	Description
Metric Name	ibm_databases_for_mysql_cpu_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used CPU for an instance metric metadata

MySQL Used disk space for an instance

How much disk space your instance is using

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

MySQL Used disk space for an instance percent

How much disk space is used as a percentage of total disk available

Metadata	Description
Metric Name	ibm_databases_for_mysql_disk_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

MySQL Used memory for an instance

How much memory your instance is using

Metadata	Description
Metric Name	ibm_databases_for_mysql_memory_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

MySQL Used memory for an instance percent

How much memory is used as a percentage of total memory available

Metadata	Description
Metric Name	ibm_databases_for_mysql_memory_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

MySQL Total active connections to the database

Represents the total number of active connections to the database

Metadata	Description
Metric Name	ibm_databases_for_mysql_total_connections
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

Replica lag

Represents the delay of a replica relative to the leader.

Metadata	Description
Metric Name	ibm_databases_for_mysql_replica_lag
Metric Type	gauge
Value Type	count
Segment By	Service instance, Resource Id, Service instance name

Replica lag metric metadata

Replica state

Represents the state of the replicas.

Metadata	Description
Metric Name	ibm_databases_for_mysql_replica_state
Metric Type	gauge
Value Type	count
Segment By	Service instance, Resource Id, Service instance name

Replica state metric metadata

Elasticsearch Metrics

Metric Name
Elasticsearch Cluster status
Elasticsearch Disk read latency mean
Elasticsearch Disk write latency mean
Elasticsearch GC Percentage
Elasticsearch IO utilization in percent 15-minute average
Elasticsearch IO utilization in percent 30-minute average
Elasticsearch IO utilization in percent 5-minute average
Elasticsearch IO utilization in percent 60-minute average
Elasticsearch IOPS read & write total count for an instance
Elasticsearch Maximum allowed memory for an instance
Elasticsearch Number of unassigned shards
Elasticsearch Total disk space for an instance
Elasticsearch Used CPU for an instance
Elasticsearch Used JVM heap for a database member of the instance in percent
Elasticsearch Used disk space for an instance
Elasticsearch Used disk space for an instance
Elasticsearch Used memory for an instance
Elasticsearch Used memory for an instance

Metrics Available by Plan Names

Elasticsearch Metrics Descriptions

Elasticsearch Cluster status

A number derived from the status value of the /_cluster/health endpoint. Possible Values: 'green' = 1.0, 'yellow' = 0.5, 'red' = 0, ERROR = -1

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_cluster_status
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

Cluster status metric metadata

Elasticsearch Disk read latency mean

Disk read latency mean

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_read_latency_mean
Metric Type	gauge
Frequency	60ms
Value Type	count
Segment By	Service instance, Service instance name

Disk read latency mean metric metadata

Elasticsearch Disk write latency mean

Disk write latency mean

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_write_latency_mean
Metric Type	gauge
Frequency	60ms
Value Type	count
Segment By	Service instance, Service instance name

Disk write latency mean metric metadata

Elasticsearch GC Percentage

Percentage of time the Elasticsearch JVM spends on garbage collection

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_garbage_collection_percent_average_15m
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

GC Percentage metric metadata

Elasticsearch IO utilization in percent 15-minute average

How much disk I/O has been used over 15 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_io_utilization_percent_average_15m
Metric Type	gauge

Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 15 minute average metric metadata

Elasticsearch IO utilization in percent 30-minute average

How much disk I/O has been used over 30 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_io_utilization_percent_average_30m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 30 minute average metric metadata

Elasticsearch IO utilization in percent 5-minute average

How much disk I/O has been used over 5 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_io_utilization_percent_average_5m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 5 minute average metric metadata

Elasticsearch IO utilization in percent 60-minute average

How much disk I/O has been used over 60 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_io_utilization_percent_average_60m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 60 minute average metric metadata

Elasticsearch IOPS read & write total count for an instance

How many input-output operations per second your instance is performing

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_iops_read_write_total

Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

IOPS read & write total count for an instance metric metadata

Elasticsearch Maximum allowed memory for an instance

The maximum amount of memory available to your instance

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_memory_limit_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Maximum allowed memory for an instance metric metadata

Elasticsearch Number of unassigned shards

Number of unassigned shards

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_unassigned_shards_total
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

Number of unassigned shards metric metadata

Elasticsearch Total disk space for an instance

Represents the total amount of disk space available to your deployment

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_total_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Total disk space for an instance metric metadata

Elasticsearch Used CPU for an instance

How much CPU is used as a percentage of total CPU available. Only for deployments that have dedicated CPU

Metadata	Description

Metric Name	ibm_databases_for_elasticsearch_cpu_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used CPU for an instance metric metadata

Elasticsearch Used JVM heap for a database member of the instance in percent

How much JVM heap is used as a percentage of total JVM heap is available

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_jvm_heap_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used JVM heap for a database member of the instance in percent metric metadata

Elasticsearch Used disk space for an instance bytes

How much disk space your instance is using

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

Elasticsearch Used disk space for an instance percent

How much disk space is used as a percentage of total disk available

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_disk_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

Elasticsearch Used memory for an instance

How much memory your instance is using

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_memory_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

Elasticsearch Used memory for an instance percent

How much memory is used as a percentage of total memory available

Metadata	Description
Metric Name	ibm_databases_for_elasticsearch_memory_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

Redis metrics

Metric name
Redis IO utilization in percent 15-minute average
Redis IO utilization in percent 30-minute average
Redis IO utilization in percent 5-minute average
Redis IO utilization in percent 60-minute average
Redis IOPS read & write total count for an instance
Redis maximum allowed memory for an instance
Redis total disk space for an instance
Redis used CPU for an instance
Redis used disk space for an instance
Redis used disk space for an instance
Redis used memory for an instance
Redis used memory for an instance
Redis blocked clients
Redis connected clients

Redis rejected connections
Redis instantaneous ops
Redis total commands processed
Redis AOF current file size
Redis RDB current file size
Redis changes since last snapshot
Redis last RDB save duration (sec)
Redis last AOF rewrite duration (sec)
Redis cache hit ratio
Redis total reads processed
Redis total writes processed
Redis total evicted keys
Redis reads per second
Redis writes per second
Redis operations per second

Metrics available by plan names

Redis metrics descriptions

Redis IO utilization in percent 15-minute average

How much disk I/O has been used over 15 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_redis_disk_io_utilization_percent_average_15m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 15 minute average metric metadata

Redis IO utilization in percent 30-minute average

How much disk I/O has been used over 30 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_redis_disk_io_utilization_percent_average_30m
Metric Type	gauge

Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 30 minute average metric metadata

Redis IO utilization in percent 5-minute average

How much disk I/O has been used over 5 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_redis_disk_io_utilization_percent_average_5m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 5 minute average metric metadata

Redis IO utilization in percent 60-minute average

How much disk I/O has been used over 60 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_databases_for_redis_disk_io_utilization_percent_average_60m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 60 minute average metric metadata

Redis IOPS read & write total count for an instance

How many input-output operations per second your instance is performing

Metadata	Description
Metric Name	ibm_databases_for_redis_disk_iops_read_write_total
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

IOPS read & write total count for an instance metric metadata

Redis maximum allowed memory for an instance

The maximum amount of memory available to your instance

Metadata	Description
Metric Name	ibm_databases_for_redis_memory_limit_bytes

Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Maximum allowed memory for an instance metric metadata

Redis total disk space for an instance

Represents the total amount of disk space available to your deployment

Metadata	Description
Metric Name	ibm_databases_for_redis_disk_total_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Total disk space for an instance metric metadata

Redis used CPU for an instance

How much CPU is used as a percentage of total CPU available. Only for deployments that have dedicated CPU

Metadata	Description
Metric Name	ibm_databases_for_redis_cpu_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used CPU for an instance metric metadata

Redis used disk space for an instance

How much disk space your instance is using

Metadata	Description
Metric Name	ibm_databases_for_redis_disk_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

Redis used disk space for an instance percent

How much disk space is used as a percentage of total disk available

Metadata	Description

Metric Name	ibm_databases_for_redis_disk_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

Redis used memory for an instance bytes

How much memory your instance is using

Metadata	Description
Metric Name	ibm_databases_for_redis_memory_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

Redis used memory for an instance percent

Memory used as a percentage of total memory available.

Metadata	Description
Metric Name	ibm_databases_for_redis_memory_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

Redis blocked clients

Number of clients pending on a blocking call.

Metadata	Description
Metric Name	ibm_databases_for_redis_blocked_clients
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

Redis blocked clients

Redis connected clients

Number of client connections.

Note: Higher number of client connections can impact performance of the Redis instance, as aggregate memory consumption can be extremely high, leading to out-of-memory errors. It is recommended to use Connection pooling.

Metadata	Description
Metric Name	ibm_databases_for_redis_connected_clients
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

Redis connected clients

Redis rejected connections

Number of connections rejected because of maxclients limit. Read more about Managing connections.

Metadata	Description
Metric Name	ibm_databases_for_redis_rejected_connections
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

Redis rejected connections

Redis instantaneous ops

Number of commands processed per second. Note: Operations per second is averaged over a minute scale, and is disaplyed in metrics.

Metadata	Description
Metric Name	ibm_databases_for_redis_instantaneous_ops
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

Commands proceesed by Redis per second

Redis total commands processed

Total number of commands processed by the server. Note: This is an incremental number which resets when your Redis instance restarts.

Metadata	Description
Metric Name	ibm_databases_for_redis_total_commands_processed
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

Total commands processed by Redis

Redis AOF current file size

Shows the AOF current file size.

Metadata	Description
Metric Name	ibm_databases_for_redis_aof_current_size
Metric Type	gauge
Value Type	byte
Segment By	Service instance

AOF current file size

Redis RDB current file size

Shows the RDB current file size.

Metadata	Description
Metric Name	ibm_databases_for_redis_rdb_current_size
Metric Type	gauge
Value Type	byte
Segment By	Service instance

RDB current file size

Redis changes since last snapshot

Shows the number of write operations performed since the last RDB snapshot was saved.

Metadata	Description
Metric Name	ibm_databases_for_redis_rdb_changes_since_last_save
Metric Type	gauge
Value Type	count
Segment By	Service instance

Changes since last snapshot

Redis last RDB save duration (sec)

Represents the duration in seconds taken by the last background RDB save operation. A higher value may indicate performance issues during snapshotting.

Metadata	Description
Metric Name	ibm_databases_for_redis_rdb_last_bgsave_time_sec
Metric Type	gauge
Value Type	count

Segment By	Service instance

Last RDB save duration (sec)

Redis last AOF rewrite duration (sec)

Shows the duration in seconds taken by the last AOF (Append-Only File) rewrite operation. Longer durations may indicate performance bottlenecks during log rewriting.

Metadata	Description
Metric Name	ibm_databases_for_redis_aof_last_rewrite_time_sec
Metric Type	gauge
Value Type	count
Segment By	Service instance

Last AOF rewrite duration (sec)

Redis cache hit ratio

Indicates the efficiency of key lookups by showing the ratio of successful key hits to total lookups in Redis. A higher ratio reflects better cache performance.

Metadata	Description
Metric Name	ibm_databases_for_redis_cache_hit_ratio
Metric Type	gauge
Value Type	count
Segment By	Service instance

Cache hit ratio

Redis total reads processed

Shows the total number of successful key lookups in the main Redis dictionary, representing all read operations processed over time.

Metadata	Description
Metric Name	ibm_databases_for_redis_total_reads_processed
Metric Type	gauge
Value Type	count
Segment By	Service instance

Total reads processed

Redis total writes processed

Shows the total number of successful key modifications in the main Redis dictionary, indicating the number of write operations processed over time.

Metadata	Description
Metric Name	ibm_databases_for_redis_total_writes_processed

Metric Type	gauge
Value Type	count
Segment By	Service instance

Total writes processed

Redis total Evicted Keys

Represents the total number of keys evicted from Redis due to memory constraints, typically when the max memory limit is reached and keys are removed based on the eviction policy.

Metadata	Description
Metric Name	ibm_databases_for_redis_evicted_keys
Metric Type	gauge
Value Type	count
Segment By	Service instance

Total evicted keys

Redis reads Per Second

Shows the rate of read operations processed by Redis each second, indicating the current read workload and query throughput.

Metadata	Description
Metric Name	ibm_databases_for_redis_reads_per_second
Metric Type	gauge
Value Type	count
Segment By	Service instance

Reads per second

Redis writes Per Second

Shows the rate of write operations processed by Redis each second, indicating the current write workload and data modification throughput.

Metadata	Description
Metric Name	ibm_databases_for_redis_writes_per_second
Metric Type	gauge
Value Type	count
Segment By	Service instance

Writes per second

Redis operations Per Second

Shows the rate of read and write operations processed by Redis each second, reflecting the overall command throughput and server workload.

Metadata	Description
Metric Name	ibm_databases_for_redis_operations_per_second
Metric Type	gauge
Value Type	count
Segment By	Service instance

Operations per second

Messages for RabbitMQ Metrics

Metric Name
Messages for RabbitMQ IO utilization in percent 15-minute average
Messages for RabbitMQ IO utilization in percent 30-minute average
Messages for RabbitMQ IO utilization in percent 5-minute average
Messages for RabbitMQ IO utilization in percent 60-minute average
Messages for RabbitMQ IOPS read & write total count for an instance
Messages for RabbitMQ Maximum allowed memory for an instance
Messages for RabbitMQ Total disk space for an instance
Messages for RabbitMQ Used CPU for an instance
Messages for RabbitMQ Used disk space for an instance
Messages for RabbitMQ Used disk space for an instance
Messages for RabbitMQ Used memory for an instance
Messages for RabbitMQ Used memory for an instance

Metrics Available by Plan Names

Messages for RabbitMQ Metrics Descriptions

Messages for RabbitMQ IO utilization in percent 15-minute average

How much disk I/O has been used over 15 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_disk_io_utilization_percent_average_15m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 15 minute average metric metadata

Messages for RabbitMQ IO utilization in percent 30-minute average

How much disk I/O has been used over 30 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_disk_io_utilization_percent_average_30m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 30 minute average metric metadata

Messages for RabbitMQ IO utilization in percent 5-minute average

How much disk I/O has been used over 5 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_disk_io_utilization_percent_average_5m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 5 minute average metric metadata

Messages for RabbitMQ IO utilization in percent 60-minute average

How much disk I/O has been used over 60 minutes as a percentage of total disk I/O available

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_disk_io_utilization_percent_average_60m
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

IO utilization in percent 60 minute average metric metadata

Messages for RabbitMQ IOPS read & write total count for an instance

How many input-output operations per second your instance is performing

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_disk_iops_read_write_total
Metric Type	gauge
Value Type	count
Segment By	Service instance, Service instance name

IOPS read & write total count for an instance metric metadata

IOPS read & write total count for an instance metric metadata

Messages for RabbitMQ Maximum allowed memory for an instance

The maximum amount of memory available to your instance

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_memory_limit_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Maximum allowed memory for an instance metric metadata

Messages for RabbitMQ Total disk space for an instance

Represents the total amount of disk space available to your deployment

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_disk_total_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Total disk space for an instance metric metadata

Messages for RabbitMQ Used CPU for an instance

How much CPU is used as a percentage of total CPU available. Only for deployments that have dedicated CPU

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_cpu_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used CPU for an instance metric metadata

Messages for RabbitMQ Used disk space for an instance bytes

How much disk space your instance is using

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_disk_used_bytes
Metric Type	gauge
Value Type	byte

Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

Messages for RabbitMQ Used disk space for an instance percent

How much disk space is used as a percentage of total disk available

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_disk_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used disk space for an instance metric metadata

Messages for RabbitMQ Used memory for an instance bytes

How much memory your instance is using

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_memory_used_bytes
Metric Type	gauge
Value Type	byte
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

Messages for RabbitMQ Used memory for an instance percent

How much memory is used as a percentage of total memory available

Metadata	Description
Metric Name	ibm_messages_for_rabbitmq_memory_used_percent
Metric Type	gauge
Value Type	percent
Segment By	Service instance, Service instance name

Used memory for an instance metric metadata

Attributes for segmentation

Global attributes

The following attributes are available for segmenting all of the metrics listed above

Attribute	Attribute Name	Attribute Description
Cloud Type	ibm_ctype	The cloud type is a value of public, dedicated or local

Location	ibm_location	The location of the monitored resource - this may be a region, data center or global
Resource	ibm_resource	The resource being measured by the service - typically a indentifying name or GUID
Resource Type	ibm_resource_type	The type of the resource being measured by the service
Scope	ibm_scope	The scope is the account, organization or space GUID associated with this metric
Service name	ibm_service_name	Name of the service generating this metric

Global segmentation attributes

Additional Attributes

The following attributes are available for segmenting one or more attributes as described in the reference above. Please see the individual metrics for segmentation options.

Attribute	Attribute Name	Attribute Description
Service instance	ibm_service_instance	The service instance segment identifies the instance the metric is associated with

Additional Segmentation Attributes

Activity tracking events for Cloud Databases

IBM Cloud® services, such as Cloud Databases, generate activity tracking events.

Activity tracking events report on activities that change the state of a service in IBM Cloud. You can use the events to investigate abnormal activity and critical actions and to comply with regulatory audit requirements.

You can use **IBM Cloud® Activity Tracker Event Routing** , a platform service, to route auditing events in your account to destinations of your choice by configuring targets and routes that define where activity tracking events are sent. For more information, see About IBM Cloud Activity Tracker Event Routing.

You can use **IBM® Cloud Logs** to visualize and alert on events that are generated in your account and routed by IBM Cloud Activity Tracker Event Routing to an IBM Cloud Logs instance.

Locations where activity tracking events are generated

Locations where activity tracking events are sent by IBM Cloud Activity Tracker Event Routing

Cloud Databases sends activity tracking events by IBM Cloud Activity Tracker Event Routing in the regions that are indicated in the following table.

Dallas (us-south)	Washington (us-east)	Toronto (ca-tor)	Sao Paulo (br-sao)
Yes	Yes	Yes	Yes

Regions where activity tracking events are sent in Americas locations

Tokyo (jp-tok)	Sydney (au-syd)	Osaka (jp-osa)	Chennai (in-che)
Yes	Yes	Yes	Yes

Regions where activity tracking events are sent in Asia Pacific locations

Frankfurt (eu-de)	London (eu-gb)	Madrid (eu-es)	Paris (eu-par01)
Yes	Yes	Yes	No

Regions where activity tracking events are sent in Europe locations

Enabling activity tracking events for Cloud Databases

Create an IBM Cloud Logs instance and configure IBM Cloud Activity Tracker Event Routing by setting the routing rule between the Cloud Databases instance and the IBM Cloud Logs target instance.

Viewing activity tracking events for Cloud Databases

You can use IBM Cloud Logs to visualize and alert on events that are generated in your account and routed by IBM Cloud Activity Tracker Event Routing to an IBM Cloud Logs instance.

Launching IBM Cloud Logs from the Cloud Databases dashboard

You can visit the Cloud Databases instance. Click on *Overview* and scroll to the *Observability* section. Click on *IBM Cloud Logs* to view your logging instances. Click on *Open Dashboard* to access the logs.

Launching IBM Cloud Logs from the Observability page

For information on launching the IBM Cloud Logs UI, see Launching the UI in the IBM Cloud Logs documentation.

List of platform events

The following table lists the activity tracking event actions that Cloud Databases and IBM Cloud generate.

Action name	Legacy action name	Description
`<service_name>.deployment-backup.create`	`<service_id>.backup-ondemand.create`	An on-demand backup of your instance was created. If the backup failed, a "-failure" flag is included in the message.
`<service_name>.deployment-backup-scheduled.create`	`<service_id>.backup-scheduled.create`	A scheduled backup of your instance was created. If the backup failed, a "-failure" flag is included in the message.
`<service_name>.deployment-user.update`	`<service_id>.user-password.update`	A user's password was updated. A "-failure" flag is included in the message if the attempt to update a user's password failed.
`<service_name>.deployment-user.create`	`<service_id>.user.create`	A user was created. A "-failure" flag is included in the message if the attempt to create a user failed.
`<service_name>.deployment-user.delete`	`<service_id>.user.delete`	A user was deleted. A "-failure" flag is included in the message if the attempt to delete a user failed.
`<service_name>.deployment-group.update`	`<service_id>.resources.scale`	A scaling operation was performed. If the scaling operation failed, a "-failure" flag is included in the message.
`<service_name>.deployment-allowlist-ip-addresses.update`	`<service_id>.whitelisted-ips-list.update`	The allowlist was modified. A "-failure" flag is included in the message if the attempt to modify the allowlist failed.
`<service_name>.deployment.update`	`<service_id>.serviceendpoints.update`	A change was made to the service endpoints configuration. If the operation failed, a "-failure" flag is included in the message.
`<service_name>.deployment-group-autoscaling.update`	`<service_id>.autoscaling.update`	An autoscaling configuration change or an autoscaling operation was performed. If an autoscaling operation was performed the message includes `autoscale resources for instance <deployment-id>`. If the autoscaling operation or the configuration change failed, a "-failure" flag is included in the message.
`<service_name>.deployment-volumes.update`	`<service_id>.volumes.update`	An activity was performed on the encryption key that is used by the database, such as rotation or shredding. Details of the action are in the event.
`<service_name>.deployment-version.update`	`<service_id>.version.update`	A version update operation was performed on your instance. If the update failed, a "-failure" flag is included in the message.

List of events and event descriptions by Cloud Databases

The `service_name` field indicates the type of Cloud Databases instance. For example, `databases-for-postgresql` or `messages-for-rabbitmq`.

Auditing of global events, such as `<service_name>.instance.create`, is covered by the IBM Cloud global event. For more resource-related global events, see

Auditing events for service instances.

Enhancing security

Managing security and compliance with Cloud Databases

Cloud Databases is integrated with the Compliance Manager to help you manage security and compliance for your organization.

With the Compliance Manager, you can monitor for controls and goals that pertain to Cloud Databases.

Monitoring security and compliance posture with Cloud Databases

As a security or compliance focal, you can use the Cloud Databases goals to help ensure that your organization is adhering to the external and internal standards for your industry. By using the Compliance Manager to validate the resource configurations in your account against a *profile*, you can identify potential issues as they arise.

Note: All of the goals for Cloud Databases are added to the IBM Cloud Control Library profile, but can also be mapped to other profiles.

To start monitoring your resources, check out Getting started with Compliance Manager

Available goals for Cloud Databases

- **Check whether Cloud Databases is enabled with IBM-managed or customer-managed encryption.** All Cloud Databases instances are automatically encrypted at rest with IBM-managed keys. For more information, see Key Protect Integration.
- **Check whether Cloud Databases is accessible only through TLS.** All Cloud Databases connections use TLS/SSL encryption for data in transit. The current supported version of this encryption is TLS 1.2.
- **Check whether Cloud Databases is accessible only by using private endpoints.** Customers can disable public endpoints at provision time. For more information, see Service Endpoints Integration.
- **Check whether Cloud Databases network access is restricted to a specific IP range.** For more information see Context-based restrictions or Allowlisting.

Security and Compliance

Protection Against Unauthorized Access

IBM Cloud® Databases for Elasticsearch use the following methods to protect data in transit or in storage.

- All Databases for Elasticsearch connections use TLS/SSL encryption for data in transit. The current supported version of this encryption is TLS 1.2.
- Access to the Account, Management Console UI, and API is secured through Identity and Access Management (IAM).
- Access to the database is secured through the standard access controls provided by the database. These access controls are configured to require valid database-level credentials that are obtainable only through prior access to the database or through our Management Console UI or API.
- All Databases for Elasticsearch storage is provided on storage encrypted with LUKS using AES-256. The default keys are managed by Key Protect. Bring-your-own-key (BYOK) for encryption is also available through Key Protect Integration.
- IP allowlisting - All deployments support allowlisting IP addresses to restrict access to the service.
- Public and Private Networking - Databases for Elasticsearch is integrated with Service Endpoints. You can select whether to use connections over the public network, the IBM Cloud internal network, or both.
- Dedicated Cores - Allocating dedicated cores to your deployment introduces hypervisor-level isolation to your database instance, by using isolated virtual machines to ensure that your data processing remains separated from other customers. It also provides a minimum number of CPUs to your deployment. Deployments with dedicated cores in the same Resource Group and IBM Cloud Region can share a virtual machine.

Data Resilience

- Backups are included in the service. Databases for Elasticsearch backups reside in IBM Cloud Object Storage and are also encrypted.
- Databases for Elasticsearch deployments are configured with replication. Deployments contain a cluster with three nodes where all three are data nodes and any node can be the primary node. You can also add nodes to provide more stability in a multi-node failure, since you can lose more nodes and maintain a quorum.
- If you deploy to an IBM Cloud Single-Zone Region (SZR), each database node resides on a different host in the data center.
- If you deploy to an IBM Cloud Multi-Zone Region (MZR), the nodes are spread over the region's availability zone locations.

SOC 2 Type 2 Certification

IBM provides a Service Organization Controls (SOC) 2 Type 2 report for Databases for Elasticsearch. The reports evaluate IBM's operational controls

according to the criteria set by the American Institute of Certified Public Accountants (AICPA) Trust Services Principles. The Trust Services Principles define adequate control systems and establish industry standards for service providers such as IBM Cloud to safeguard their customers' data and information.

You can request an SOC 2 Type 2 report from the customer portal or contact your sales representative. Alternatively, you can open a support ticket with IBM Cloud support

ISO 27017, ISO 27018

Databases for Elasticsearch conforms to the guidelines for information security controls applicable to the provision and use of cloud services that are defined in ISO 27017 and ISO 27018.

General Data Protection Regulation (GDPR)

If you have an account with IBM Cloud, your personal data is held by IBM Cloud. The IBM Data Processing Addendum (Addendum) applies to the processing of client's personal data by IBM on behalf of client in order to provide IBM standard services.
IBM DPA

Databases for Elasticsearch processes limited client Personal Information (PI) in the course of running the service and optimizing the user experience.

Databases for Elasticsearch provides a Data Sheet Addendum (DSA) with its policies as a Data Processor regarding content and data protection.

HIPAA

Databases for Elasticsearch meets the required IBM controls that are commensurate with the Health Insurance Portability and Accountability Act of 1996 (HIPAA) Security and Privacy Rule requirements. These requirements include the appropriate administrative, physical, and technical safeguards required of Business Associates in 45 CFR Part 160 and Subparts A and C of Part 164. HIPAA must be requested at the time of provisioning and requires a representative to sign a Business Associate Addendum (BAA) agreement with IBM.

PCI DSS

Databases for Elasticsearch are compliant with the Payment Card Industry Data Security Standard (PCI DSS). IBM Cloud completes annual PCI DSS assessments by using an approved Qualified Security Assessor (QSA), and the resulting Attestations of Compliance (AOCs) and Service Responsibility Matrix (SRM) guides are available upon customer request. Auditors reviewed Databases for Elasticsearch for compliance under PCI DSS version 3.2.1 at Service Provider Level 1.

Customers are responsible for the storing, processing, and transmission of their cardholder data, and can create cardholder data environments (CDEs) that can store, transmit, or process cardholder data by using Databases for Elasticsearch. Customers can request and use the IBM Cloud AOCs and SRM guides when they seek their own PCI DSS certifications. It is the responsibility of the customer to document and operate CDEs and applications that are built by using IBM Cloud Platform services in a PCI DSS-compliant manner.

It is the customer's responsibility to familiarize themselves with these processes and to manage data retention and removal from the service according to the customer's policies.

A full list of PCI DSS-ready IBM Cloud Platform services, and options to request a PCI DSS AOC and SRM guide, can be found at the IBM Cloud compliance page.

Terms

- The IBM Privacy Policy
- The IBM Cloud Notices and Terms of Use

⚠ **Important:** This document outlines the process for using context-based restrictions to protect your Cloud Databases resources. Use this document to prepare your resources for context-based restrictions. Cloud Databases doesn't offer scoping rules to the control plane in this current phase of implementation.

Context-based restrictions

Context-based restrictions give account owners and administrators the ability to define and enforce access restrictions for IBM Cloud® resources based on the context of access requests. Access to Cloud Databases resources can be controlled with context-based restrictions and Identity and Access Management (IAM) policies.

These restrictions work with traditional IAM policies, which are based on identity, to provide an extra layer of protection. Unlike IAM policies, context-based restrictions don't assign access. Context-based restrictions check that an access request comes from an allowed context that you configure. Since both

IAM access and context-based restrictions enforce access, context-based restrictions offer protection even in the face of compromised or mismanaged credentials. For more information, see What are context-based restrictions.

> **Note:** A user must have the Administrator role on the Cloud Databases service to create, update, or delete rules. A user must also have either the Editor or Administrator role on the context-based restrictions service to create, update, or delete network zones. A user with the Viewer role on the context-based restrictions service can add network zones to a rule.

Any IBM Cloud Activity Tracker or audit log events generated come from the context-based restrictions service, not Cloud Databases. Cloud Databases supports audit events only for customer interactions with context-based restrictions-protected platform endpoint calls. Cloud Databases does not support audit events when you enable context-based restrictions rules on the control plane API for your instances. For more information, see Monitoring context-based restrictions.

To start protecting your Cloud Databases resources with context-based restrictions, see the tutorial for Leveraging context-based restrictions to secure your resources.

How Cloud Databases integrates with context-based restrictions

You can create context-based restrictions for the Cloud Databases service, specific resources, and specific APIs.

Protecting Cloud Databases resources

You can create context-based restrictions rules to protect specific **regions**, **resource groups**, and **instances**.

Region

Protects Cloud Databases resources in a specific region. If you include a region in your context-based restrictions rule, resources in the network zones that you associate with the rule can interact with resources only in that region. If you use the CLI, you can specify the `--region` option to protect resources in a specific region. If you use the UI, you can specify *Region* in the resource attributes.

Resource groups

Protects a specific resource group. If you include a resource group in your context-based restrictions rule, resources in the network zones that you associate with the rule can interact only with resources in that resource group. Scoping a rule to a specific resource group is available only for rules that protect the cluster API type. If you use the CLI, you can specify the `--resource-group-id` option to protect resources in a specific resource group. If you use the UI, you can specify the *Resource group* in the resource attributes.

Instance

Protects a specific instance. If you include an instance in your context-based restrictions rule, resources in the network zones that you associate with the rule can interact only with resources in that instance. Scoping a rule to a specific instance is available only for rules that protect the cluster API type. If you use the CLI, you can specify the `--service-instance` option to protect instances in a specific resource group. If you use the UI, you can specify the *Service instance* in the resource attributes.

Using the Command Line Interface (CLI)

You can create and manage context-based restrictions with the IBM Cloud CLI by installing the context-based restrictions CLI plug-in.

Creating network zones

A network zone represents an allowlist of IP addresses where an access request is created. It defines a set of one or more network locations that are specified by the following attributes:

- IP addresses, which include individual addresses, ranges, or subnets.
- VPCs.

Creating network zones in the UI

1. Go to **Manage** > **Context-based restrictions** in the IBM Cloud® console.
2. Select **Network zones**.
3. Click **Create**.
4. Name your network zone and provide a description.
5. Enter your *Allowed IP addresses.* You can enter a single IP address, a range of IP addresses, or a single CIDR.

> **Note:** The **Denied IP addresses** field is optional and should include only exceptions that are contained within the IP ranges you provide in the allowed IP addresses field.

6. Choose your *Allowed VPCs*, selecting as many as you like.
7. **Reference a service**: You can select Cloud Databases as a source service for context-based restrictions, but not as a target service. For example, you can provision a Cloud Databases instance using BYOK from IBM® Key Protect for IBM Cloud®. In this example, Cloud Databases is the source formation and IBM® Key Protect for IBM Cloud® is the target formation. Then, you would create a network zone with a Cloud Databases service reference and create a rule associated with the network zone that targets IBM® Key Protect for IBM Cloud®. To add a Cloud Databases service reference, for *Service Type*, IAM services is autoselected. In the *Service* dropdown, select a specific Cloud Databases service. If the zone you create is associated with a rule targeting Cloud Databases, then a service reference is not allowed.

> **Important:** Service references function only from Key Protect service to Cloud Databases.

Create network zones in the CLI

To create network zones in the CLI, use the `cbr-zone-create` command to add resources to network zones. For more information, see the context-based restrictions CLI reference.

Create a zone by using a command like:

```
$ ibmcloud cbr zone-create --addresses=1.1.1.1,5.5.5.5 --name=<NAME>
```

Creating network zones in Terraform

To create zones in Terraform, follow the instructions in the IBM Cloud Terraform provider documentation.

Example Terraform script to create a CBR zone:

```
$ resource "ibm_cbr_zone" "cbr_zone" {
  account_id = "12ab34cd56ef78ab90cd12ef34ab56cd"
  addresses {
    type = "ipAddress"
    value = "169.23.56.234"
  }
  addresses {
    type = "ipRange"
    value = "169.23.22.0-169.23.22.255"
  }
  excluded {
    type  = "ipAddress"
    value = "169.23.22.10"
  }
  excluded {
    type  = "ipAddress"
    value = "169.23.22.11"
  }
  description = "this is an example of zone"
  excluded {
       type = "ipAddress"
       value = "value"
  }
  name = "an example of zone"
}
```

Alternatively, you can also use Terraform IBM Modules (TIM) for CBR Zone to create a zone for context-based restrictions or update addresses in an existing zone.

An example for creating a CBR zone using Terraform IBM Modules (TIM):

```
module "ibm_cbr" "zone" {
  source          = "terraform-ibm-modules/cbr/ibm//modules/cbr-zone-module"
  version         = "X.X.X" # Replace "X.X.X" with a release version to lock into a specific release
  name            = "zone_for_pg_access"
  account_id      = "defc0df06b644a9cabc6e44f55b3880s"
```

```
  zone_description = "Zone created from terraform"
  addresses        = [{type  = "vpc",value = "vpc_crn"}]
}
```

Update network zones in the CLI

Update a zone by using a command like:

```
$ ibmcloud cbr zone-update <ZONE-ID> --addresses=1.2.3.4 --name=<NAME>
```

Updating requires the `ZONE-ID`, not the zone name. Use the following command to list your zones and retrieve the relevant `ZONE-ID`:

```
$ ibmcloud cbr zones
```

⚠ **Important:** The `zone-update` command is an overwrite. Include all of the fields that are required as if you are creating the rule from scratch. If you omit any required fields, the rule overwrites those missing fields as empty, and the rule might fail because some of those fields are required, regardless of whether they are changing the rule.

Delete network zones in the CLI

Delete a zone by using a command like:

```
$ ibmcloud cbr zone-delete <ZONE-ID>
```

Creating rules

Rules restrict access to specific cloud resources based on resource attributes and contexts. A created rule can accept up to 2,000 IP/CIDR values for private endpoints and up to 2,000 IP/CIDR values for public endpoints. This limit is specfic to Cloud Databases. Other IBM Cloud® service limits may vary.

Cloud Databases does not support IPv6 addresses. If an IPv6 address is included, it will be ignored.

⚠ **Important:** Full closure of access to non-allowlisted endpoints: To provide a more robust security framework, we have implemented a significant change in access control for public and private endpoints. Going forward, access to both public and private endpoints that are not explicitly allowlisted will be fully closed. This restriction ensures only authorized access to your endpoints, minimizing the risk of unauthorized access.

Creating rules in the UI

1. Go to **Manage** > **Context-based restrictions** in the IBM Cloud® console.
2. Select **Rules**.
3. Click **Create**.
4. Under **Service**, select the service you want to target with your rule.
5. Under **APIs**, select `Data plane`. Currently any other selection results in an error.

   Note: 0 actions are expected for the Data Plane API type.

   Note: Cloud Databases does not currently support **Control plane** as an option.
6. Under **Resources**, scope the rule to **All resources** or **Specific resources**. For more information, see Protecting Cloud Databases resources.
7. Click **Continue**.
8. Define the allowed endpoint types.
   - Keep the toggle set to **No** to allow all endpoint types.
   - Set the toggle to **Yes** to allow only specific endpoint types, then choose from the list.
9. Select a network zone or zones that you have already created, or create a new network zone by clicking **Create**.

   Tip: Contexts define from where your resources can be accessed, effectively linking your network zone to your rule.

10. Click **Add** to add your configuration to the summary.
11. Click **Next**.
12. Name your rule.
13. Select how you want to enforce the rule.

> ⚠ **Important:** *Report-only* is not available for Cloud Databases.

Create rules in the CLI

To create a rule in the CLI, you need the appropriate Cloud Databases `service_name`:

- `databases-for-postgresql`
- `databases-for-mongodb`
- `databases-for-redis`
- `databases-for-elasticsearch`
- `database-for-mysql`
- `messages-for-rabbitmq`

All the other parameters that follow are explained in the CBR plugin reference guide.

Example command for creating a CBR Rule:

```
$ ibmcloud cbr rule-create --enforcement-mode enabled --context-attributes "networkZoneId=<ZONE-ID>" --resource-group-id <RESOURCE_GROUP_ID> --service-name <SERVICE-NAME> --service-instance <SERVICE-INSTANCE> --api-types crn:v1:bluemix:public:context-based-restrictions::::api-type:data-plane --description <DESCRIPTION>
```

> **Note:** The only `api-type` option currently supported by Cloud Databases is **Data plane**.

> ⚠ **Important:** *Report-only* is not available for Cloud Databases.

Update rules in the CLI

Example command for updating a CBR rule:

```
$ ibmcloud cbr rule-update <RULE-ID> --enforcement-mode disabled --context-attributes="networkZoneId=<ZONE-ID>" --resource-group-id <RESOURCE_GROUP_ID> --service-name <SERVICE_NAME> --api-types crn:v1:bluemix:public:context-based-restrictions::::api-type:data-plane --description <DESCRIPTION>
```

> ⚠ **Important:** The `rule-update` command is an overwrite. Include all of the fields that are required as if you are creating the rule from scratch. If you omit any required fields, the rule overwrites those missing fields as empty, and the rule might fail because some of those fields are required, regardless of whether they are changing the rule.

Updating requires the `RULE-ID`, not the rule name. Use the following command to list your rules and retrieve the relevant `RULE-ID`:

```
$ ibmcloud cbr rules
```

Delete rules in the CLI

Delete a rule by using a command like:

```
$ ibmcloud cbr rule-delete <RULE-ID>
```

> **Tip:** Use `ibmcloud cbr <command> — help` for a full list of options and parameters. For example, `ibmcloud cbr rule-create — help` outputs parameters for rule creation.

Creating rules in Terraform

To create rules in Terraform, follow the instructions in the IBM Cloud Terraform provider documentation.

To create a rule, you need the appropriate Cloud Databases `service_name`:

- `databases-for-postgresql`
- `databases-for-mongodb`
- `databases-for-redis`
- `databases-for-elasticsearch`
- `database-for-mysql`
- `messages-for-rabbitmq`

Create a rule by using a command like:

```
$ resource "ibm_cbr_rule" "cbr_rule" {
  contexts {
      attributes {
         name = "networkZoneId"
         value = "559052eb8f43302824e7ae490c0281eb"
      }
      attributes {
           name = "endpointType"
           value = "private"
    }
  }
  description = "this is an example of a rule with one context one zone"
  enforcement_mode = "enabled"
  operations {
      api_types {
         api_type_id = "api_type_id"
      }
  }
  resources {
      attributes {
         name = "accountId"
         value = "12ab34cd56ef78ab90cd12ef34ab56cd"
      }
      attributes {
           name = "serviceName"
           value = "network-policy-enabled"
      }
      tags {
           name     = "tag_name"
           value    = "tag_value"
      }
  }
}
```

Verifying your rule

To verify that your rule is applied, go to the IBM Cloud® Dashboard and select the relevant instance from your *Resource List*. Within **Recent Tasks**, you see your rule's status.

> **Note:** The task of creating or modifying a rule goes into your instance's task queue. Depending on workload, it might take some time for your rule enforcement to complete.

Example context-based restrictions scenarios

With context-based restrictions, account owners and administrators can define and enforce access restrictions for IBM Cloud® resources, based on the context of access requests. Access to Cloud Databases resources can be controlled with context-based restrictions and identity and access management policies. For more information, see Protecting Cloud Databases resources with context-based restrictions.

Restrict traffic to your deployment by using Cloud Databases Allowlisting

In this example scenario, you use context-based restrictions to restrict traffic to your IBM Cloud® Databases for MySQL cluster in the `in-che` region by allowing only the set of subnets from the Cloud Databases Allowlist page to connect to your deployment.

In the following steps, you start by creating a network zone, or allowlist, that includes your subnets. Then, you create a context-based restrictions rule for your deployment. When you create the rule, you associate it with the network zone that contains the individual IP address.

Prerequisites

Before beginning this tutorial, make sure you have created or installed the following resources and tools.

- An IBM Cloud account. For more information, see Creating an account.
- The [Cloud Databases CLI plug-in/docs/cloud-databases?topic=cloud-databases-cdb-reference) - the CLI interface to interact with the Cloud Databases API. For more information, see Getting started with the IBM Cloud CLI.
- A Cloud Databases deployment. For more information, see Provisioning.
- Review the What are context-based restrictions docs to get an understanding of concepts of network zones and rules.
- Review the Protecting Cloud Databases resources with context-based restrictions docs to understand how you can leverage CBR for your Cloud Databases resources.

Step 1: Creating your network zone in the CLI

Provision your service from the IBM Cloud Catalog and choose your IP addresses from the Cloud Databases Allowlist page.

1. Run the following example command to create a network that includes a range of allowed IP addresses.

```
$ ibmcloud cbr zone-create --
addresses=169.38.95.127/27,169.38.121.159/28,169.38.132.127/25,169.38.136.255/26,169.38.73.151/29,169.38.105.79/29,10.162.8.127/26,10.163.20.127/25,10.162.115.103/29,10.162
9 --name=tutorial_zone
```

2. Verify the network zone was created.

```
$ ibmcloud cbr zones
```

Step 2: Creating your CBR rule in the CLI

1. After you create your network zone (allowlist), create a context-based restrictions rule and add the network zone you created in the previous step. The following example creates a rule that uses the `data-plane` API type. Replace `ZONE-ID` with the ID of the `tutorial_zone` network zone that you created in Step 1.

```
$ ibmcloud cbr rule-create --enforcement-mode enabled --context-attributes networkZoneId=<ZONE-ID> --resource-group-id <RESOURCE_GROUP_ID> --service-name
databases-for-mysql --service-instance <SERVICE-INSTANCE> --api-types crn:v1:bluemix:public:context-based-restrictions::::api-type:data-plane --description
<DESCRIPTION>
```

Understanding the command options.

`--context-attributes (string)`

Contexts to add to the rule. Can only pass in one `networkZoneId` field at a time. This option can be repeated to add multiple network zones. Please use this if you need to specify the `endpointType` with the `networkZoneId`.

`--zone-id (string)`

Shorthand for adding context attribute `networkZoneId` to the context without specifying endpoint.

`--resource-attributes (string)`

Resources to add to the rule.

`--resource-group-id (string)`

Shorthand for creating IBM Cloud resource attribute `resourceGroupId`. Used to restrict the rule to a single resource group.

`--region (string)`

Shorthand for creating IBM Cloud resource attribute `region`. Used to restrict the rule to a single region.

`--service-name (string)`

Shorthand for creating IBM Cloud resource attribute `serviceName`.

`--service-instance (string)`

GUID of the service instance to scope the context to. This option can be omitted if the context applies to more than one of you service instances. This option is exclusive with the `--file` option.

2. Verify the rule was created.

```
$ ibmcloud cbr rules
```

Step 3: Testing your context-based restrictions

To test your context-based restrictions setup, try connecting to your deployment from an IP address other than the IP addresses that you allowlisted in your network zone. With this setup, only the IP addresses in your network zone can connect to your deployment.

Allowlisting

⊖ **Deprecated:** Cloud Databases now supports context-based restrictions, which give account owners and administrators the ability to define and enforce access restrictions for IBM Cloud® resources based on the context of access requests. Access to Cloud Databases resources can be controlled with context-based restrictions and identity and access management (IAM) policies. Context-based restrictions check that an access request comes from an allowed context that you configure. Since both IAM access and context-based restrictions enforce access, context-based restrictions offer protection even in the face of compromised or mismanaged credentials. For more information, see Context-based restrictions.

To restrict access to your databases, allowlist specific IP addresses or ranges of IP addresses on your deployment.

If you use allowlists in your environment, allowlist our services by using the list of subnets for each region.

Note: While IBM values the use of inclusive language, terms that are outside of IBM's direct influence are sometimes required for the sake of maintaining user understanding. As other industry leaders join IBM in embracing the use of inclusive language, IBM continues to update terminology to reflect those changes. We updated documentation to reflect changes in terminology from `whitelist` to `allowlist`. You might encounter continued references to this former terminology while we work to implement these deeper changes to code, API, and CLI commands.

Using IP allowlists on your Deployment

When you create an allowlist, only IP addresses that match the allowlist or are in the range of IP addresses in the allowlist can connect to your deployment. Allowlists can be enabled for both public endpoints and private endpoints. When no IP addresses are in the allowlist, the allowlist is disabled and the deployment accepts connections from any IP address.

⚠ **Important:** Each deployment is limited to 100 allowlist entries.

Note: Even if not explicitly allowlisted, IBM Cloud management services are still able to connect.

Setting an allowlist in the UI

The UI for managing allowlists is on the *Settings* tab of your *Deployment Overview*.

IP addresses

The *IP* field can take a single complete IPv4 address with or without a netmask. Without a netmask, incoming connections must come from exactly that IP address. To allow a connection from a specified range of IP addresses, use a netmask. The IP address must be fully specified. That means entering, for example, 192.168.1.0/24 rather than 192.168.1/24.

Tip: IPv6 is not currently supported.

Description

The *Description* can be any user-significant text for identifying the allowlist entry - a customer name, project identifier, or employee number, for example. The description field is required.

Setting an allowlist through the CLI

The Cloud Databases CLI plug-in offers a set of commands for managing allowlists. Use `cdb deployment-whitelist-add` to add an allowlist.

To add a single IP address, use a command like:

```
$ ibmcloud cdb deployment-whitelist-add <INSTANCE_NAME_OR_CRN> 198.51.100.1 "Allowlisted for testing"
```

To add an IP address range, use a command like:

```
$ ibmcloud cdb deployment-whitelist-add <INSTANCE_NAME_OR_CRN> 198.51.100.0/24 "Testing range is now open"
```

The `cdb deployment-whitelist-list` command allows you to view the current allowlist.

Use a command like:

```
$ ibmcloud cdb deployment-whitelist-list <INSTANCE_NAME_OR_CRN>
```

For more information, see the Cloud Databases CLI plug-in reference page.

Setting an allowlist through the API

Manage your allowlist with the Cloud Databases API Authorization endpoint. Retrieve the current allowlist, add entries to the allowlist, and also bulk upload IP addresses to the allowlist from the API with this endpoint.

To add an address, use a command like:

```
$ curl -X POST https://api.{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/allowlists/ip_addresses -H 'Authorization: Bearer <>' -H 'Content-Type: application/json' -d
'{"ip_address": {"address": "http://172.16.254.1/16", "description": "Dev IP space 3"}}' \
```

For more information, see Cloud Databases API Security.

Removing an allowlist in the UI

In the UI, remove an IP address or netmask from the allowlist by clicking *Remove*.

Removing an allowlist in the CLI

The `cdb deployment-whitelist-delete` command removes an IP address or range from the current allowlist for a deployment.

The command looks like:

```
$ ibmcloud cdb deployment-allowlist-delete <INSTANCE_NAME_OR_CRN> <ALLOWLIST ADDRESS OR RANGE> [--nowait] [--json]
```

When all entries on the allowlist are removed, the allowlist is disabled and all IP addresses are accepted by your deployment.

Removing an allowlist in the API

Delete an IP address or range with the Cloud Databases API Authorization endpoint.

The command looks like:

```
$ curl -X DELETE https://api.{region}.databases.cloud.ibm.com/v5/ibm/deployments/{id}/allowlists/ip_addresses/{ipaddress} -H 'Authorization: Bearer <>' \
```

When all entries in the allowlist are removed, the allowlist is disabled and all IP addresses are accepted by your deployment.

Allowlist Cloud Databases in your Environment

If you use allowlists to control connections in your environment, use the following IP lists to allowlist Cloud Databases deployments. We recommend you allowlist all of the subnet ranges for the *entire* region that your deployments live in.

in-che List

Public Subnets

Location	Region	Data center	Subnet	First IP
Chennai	in-che	che01	169.38.95.127/27	169.38.95.97
Chennai	in-che	che01	169.38.121.159/28	169.38.121.145

Chennai	in-che	che01	169.38.132.127/25	169.38.132.1
Chennai	in-che	che01	169.38.136.255/26	169.38.136.193
Chennai	in-che	che01	169.38.73.151/29	169.38.73.145
Chennai	in-che	che01	169.38.105.79/29	169.38.73.145

in-che Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Chennai	in-che	che01	10.162.8.127/26	10.162.8.65
Chennai	in-che	che01	10.163.20.127/25	10.163.20.1
Chennai	in-che	che01	10.162.115.103/29	10.162.115.97
Chennai	in-che	che01	10.162.132.79/29	10.162.132.73

in-che Private Subnets

`ca-tor` List

Public Subnets

Location	Region	Data center	Subnet	First IP
Toronto	ca-tor	tor01	158.85.91.111/28	158.85.91.97
Toronto	ca-tor	tor01	158.85.120.63/26	158.85.120.1
Toronto	ca-tor	tor01	169.55.136.127/25	169.55.136.1
Toronto	ca-tor	tor01	169.55.142.191/27	169.55.142.161
Toronto	ca-tor	tor01	158.85.95.183/29	158.85.95.177
Toronto	ca-tor	tor01	169.55.130.215/29	169.55.130.209
Toronto	ca-tor	tor04	163.74.68.95/28	163.74.68.81
Toronto	ca-tor	tor04	163.74.69.159/27	163.74.69.129
Toronto	ca-tor	tor04	163.74.72.127/26	163.74.72.65
Toronto	ca-tor	tor04	163.74.73.255/25	163.74.73.129
Toronto	ca-tor	tor04	163.74.68.55/29	163.74.68.49
Toronto	ca-tor	tor04	163.74.68.63/29	163.74.68.57
Toronto	ca-tor	tor05	163.75.67.111/28	163.75.67.97
Toronto	ca-tor	tor05	163.75.68.95/27	163.75.68.65

Toronto	ca-tor	tor05	163.75.75.63/26	163.75.75.1
Toronto	ca-tor	tor05	163.75.96.255/25	163.75.96.129
Toronto	ca-tor	tor05	163.75.67.7/29	163.75.67.1
Toronto	ca-tor	tor05	163.75.67.119/29	163.75.67.113

ca-tor Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Toronto	ca-tor	tor01	10.114.100.127/26	10.114.100.65
Toronto	ca-tor	tor01	10.114.225.127/25	10.114.225.1
Toronto	ca-tor	tor01	10.114.79.63/29	10.114.79.57
Toronto	ca-tor	tor01	10.115.88.79/29	10.115.88.73
Toronto	ca-tor	tor04	10.11.22.127/25	10.11.22.1
Toronto	ca-tor	tor04	10.11.25.191/26	10.11.25.129
Toronto	ca-tor	tor04	10.11.12.47/29	10.11.12.41
Toronto	ca-tor	tor04	10.11.12.55/29	10.11.12.49
Toronto	ca-tor	tor05	10.243.14.255/26	10.243.14.193
Toronto	ca-tor	tor05	10.243.102.255/25	10.243.102.129
Toronto	ca-tor	tor05	10.243.23.135/29	10.243.23.129
Toronto	ca-tor	tor05	10.243.23.159/29	10.243.23.153

ca-tor Private Subnets

`br-sao` List

Public Subnets

Location	Region	Data center	Subnet	First IP
Sao Paolo	br-sao	sao01	169.57.154.239/28	169.57.154.225
Sao Paolo	br-sao	sao01	169.57.191.63/26	169.57.191.1
Sao Paolo	br-sao	sao01	169.57.198.255/25	169.57.198.129
Sao Paolo	br-sao	sao01	169.57.225.127/27	169.57.225.97
Sao Paolo	br-sao	sao01	169.57.167.95/29	169.57.167.89
Sao Paolo	br-sao	sao01	169.57.199.23/29	169.57.199.17

Sao Paolo	br-sao	sao04	163.107.67.63/28	163.107.67.49
Sao Paolo	br-sao	sao04	163.107.69.63/27	163.107.69.33
Sao Paolo	br-sao	sao04	163.107.72.127/26	163.107.72.65
Sao Paolo	br-sao	sao04	163.107.75.255/25	163.107.75.129
Sao Paolo	br-sao	sao04	163.107.68.87/29	163.107.68.81
Sao Paolo	br-sao	sao04	163.107.68.95/29	163.107.68.89
Sao Paolo	br-sao	sao05	163.109.68.63/26	163.109.68.1
Sao Paolo	br-sao	sao05	163.109.68.95/27	163.109.68.65
Sao Paolo	br-sao	sao05	163.109.68.175/28	163.109.68.161
Sao Paolo	br-sao	sao05	163.109.76.127/25	163.109.76.1
Sao Paolo	br-sao	sao05	163.109.65.119/29	163.109.65.113
Sao Paolo	br-sao	sao05	163.109.65.127/29	163.109.65.121

br-sao Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Sao Paolo	br-sao	sao01	10.150.175.255/26	10.150.175.193
Sao Paolo	br-sao	sao01	10.151.137.127/25	10.151.137.1
Sao Paolo	br-sao	sao01	10.150.169.167/29	10.150.169.161
Sao Paolo	br-sao	sao01	10.150.207.55/29	10.150.207.49
Sao Paolo	br-sao	sao04	10.14.19.127/26	10.14.19.65
Sao Paolo	br-sao	sao04	10.14.41.255/25	10.14.41.129
Sao Paolo	br-sao	sao04	10.14.20.63/29	10.14.20.57
Sao Paolo	br-sao	sao04	10.14.21.23/29	10.14.21.17
Sao Paolo	br-sao	sao05	10.15.20.191/26	10.15.20.129
Sao Paolo	br-sao	sao05	10.15.31.127/25	10.15.31.1
Sao Paolo	br-sao	sao05	10.15.18.183/29	10.15.18.177
Sao Paolo	br-sao	sao05	10.15.18.191/29	10.15.18.185

br-sao Private Subnets

eu-gb List

Public Subnets

Location	Region	Data center	Subnet	First IP
London	eu-gb	lon04	158.175.64.95/28	158.175.64.81
London	eu-gb	lon04	158.175.139.191/27	158.175.139.161
London	eu-gb	lon04	158.175.141.255/24	158.175.141.1
London	eu-gb	lon04	158.175.147.63/26	158.175.147.1
London	eu-gb	lon04	158.175.151.127/25	158.175.151.1
London	eu-gb	lon04	158.175.156.255/24	158.175.156.1
London	eu-gb	lon04	158.175.81.111/29	158.175.81.105
London	eu-gb	lon04	158.175.91.151/29	158.175.91.145
London	eu-gb	lon04	158.175.92.175/29	158.175.92.169
London	eu-gb	lon04	158.175.97.7/29	158.175.97.1
London	eu-gb	lon04	158.175.97.167/29	158.175.97.161
London	eu-gb	lon04	158.175.125.79/29	158.175.125.73
London	eu-gb	lon05	141.125.69.63/27	141.125.69.33
London	eu-gb	lon05	141.125.71.15/28	141.125.71.1
London	eu-gb	lon05	141.125.83.127/26	141.125.83.65
London	eu-gb	lon05	141.125.88.127/25	141.125.88.1
London	eu-gb	lon05	141.125.93.255/24	141.125.93.1
London	eu-gb	lon05	141.125.146.255/24	141.125.146.1
London	eu-gb	lon05	141.125.69.31/29	141.125.69.25
London	eu-gb	lon05	141.125.70.7/29	141.125.70.1
London	eu-gb	lon05	141.125.77.191/29	141.125.77.185
London	eu-gb	lon05	141.125.85.87/29	141.125.85.81
London	eu-gb	lon05	141.125.87.55/29	141.125.87.49
London	eu-gb	lon05	141.125.98.103/29	141.125.98.97
London	eu-gb	lon06	158.176.109.207/28	158.176.109.193
London	eu-gb	lon06	158.176.109.255/27	158.176.109.225
London	eu-gb	lon06	158.176.122.63/26	158.176.122.1

London	eu-gb	lon06	158.176.139.255/25	158.176.139.129
London	eu-gb	lon06	158.176.148.255/24	158.176.148.1
London	eu-gb	lon06	158.176.199.255/24	158.176.199.1
London	eu-gb	lon06	158.176.71.255/29	158.176.71.249
London	eu-gb	lon06	158.176.87.135/29	158.176.87.129
London	eu-gb	lon06	158.176.113.143/29	158.176.113.137
London	eu-gb	lon06	158.176.122.239/29	158.176.122.233
London	eu-gb	lon06	158.176.124.223/29	158.176.124.217
London	eu-gb	lon06	158.176.131.15/29	158.176.131.9

eu-gb Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
London	eu-gb	lon04	10.45.36.63/26	10.45.36.1
London	eu-gb	lon04	10.45.110.255/24	10.45.110.1
London	eu-gb	lon04	10.45.235.255/25	10.45.235.129
London	eu-gb	lon04	10.45.245.255/24	10.45.245.1
London	eu-gb	lon04	10.45.250.255/24	10.45.250.1
London	eu-gb	lon04	10.45.17.7/29	10.45.17.1
London	eu-gb	lon04	10.45.17.135/29	10.45.17.129
London	eu-gb	lon04	10.45.63.79/29	10.45.63.73
London	eu-gb	lon04	10.45.63.127/29	10.45.63.121
London	eu-gb	lon04	10.45.131.175/29	10.45.131.169
London	eu-gb	lon04	10.45.142.247/29	10.45.142.241
London	eu-gb	lon05	10.196.18.63/26	10.196.18.1
London	eu-gb	lon05	10.196.54.255/25	10.196.54.129
London	eu-gb	lon05	10.196.60.255/24	10.196.60.1
London	eu-gb	lon05	10.196.148.255/24	10.196.148.1
London	eu-gb	lon05	10.196.149.255/24	10.196.149.1
London	eu-gb	lon05	10.196.4.183/29	10.196.4.177

London	eu-gb	lon05	10.196.12.191/29	10.196.12.185
London	eu-gb	lon05	10.196.13.71/29	10.196.13.65
London	eu-gb	lon05	10.196.43.231/29	10.196.43.225
London	eu-gb	lon05	10.196.50.39/29	10.196.50.33
London	eu-gb	lon05	10.196.50.47/29	10.196.50.41
London	eu-gb	lon06	10.72.162.191/26	10.72.162.129
London	eu-gb	lon06	10.72.196.255/24	10.72.196.1
London	eu-gb	lon06	10.72.211.255/25	10.72.211.129
London	eu-gb	lon06	10.242.60.255/24	10.242.60.1
London	eu-gb	lon06	10.242.63.255/24	10.242.63.1
London	eu-gb	lon06	10.72.9.63/29	10.72.9.57
London	eu-gb	lon06	10.72.25.95/29	10.72.25.89
London	eu-gb	lon06	10.72.42.215/29	10.72.42.209
London	eu-gb	lon06	10.72.95.167/29	10.72.95.161
London	eu-gb	lon06	10.72.127.111/29	10.72.127.105
London	eu-gb	lon06	10.72.187.247/29	10.72.187.241

eu-gb Private Subnets

eu-es List

Public Subnets

Location	Region	Data center	Subnet	First IP
Madrid	eu-es	mad02	13.120.67.15/28	13.120.67.1
Madrid	eu-es	mad02	13.120.67.159/27	13.120.67.129
Madrid	eu-es	mad02	13.120.66.175/29	13.120.66.169
Madrid	eu-es	mad02	13.120.66.231/29	13.120.66.225
Madrid	eu-es	mad04	13.121.64.63/28	13.121.64.49
Madrid	eu-es	mad04	13.121.66.127/27	13.121.66.97
Madrid	eu-es	mad04	13.121.66.47/29	13.121.66.41
Madrid	eu-es	mad04	13.121.66.183/29	13.121.66.177
Madrid	eu-es	mad05	13.122.64.15/28	13.122.64.1

Madrid	eu-es	mad05	13.122.66.255/27	13.122.66.225
Madrid	eu-es	mad05	13.122.64.55/29	13.122.64.49
Madrid	eu-es	mad05	13.122.65.247/29	13.122.65.241

eu-es Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Madrid	eu-es	mad02	10.118.13.255/26	10.118.13.193
Madrid	eu-es	mad02	10.118.8.231/29	10.118.8.225
Madrid	eu-es	mad02	10.118.8.239/29	10.118.8.233
Madrid	eu-es	mad04	10.118.75.127/26	10.118.75.65
Madrid	eu-es	mad04	10.118.67.119/29	10.118.67.113
Madrid	eu-es	mad04	10.118.70.87/29	10.118.70.81
Madrid	eu-es	mad05	10.118.138.255/26	10.118.138.193
Madrid	eu-es	mad05	10.118.131.247/29	10.118.131.241
Madrid	eu-es	mad05	10.118.136.39/29	10.118.136.33

eu-es Private Subnets

au-syd List

Public Subnets

Location	Region	Data center	Subnet	First IP
Sydney	au-syd	syd01	168.1.13.127/26	168.1.13.65
Sydney	au-syd	syd01	168.1.29.63/28	168.1.29.49
Sydney	au-syd	syd01	168.1.216.127/27	168.1.216.97
Sydney	au-syd	syd01	168.1.220.127/25	168.1.220.1
Sydney	au-syd	syd01	168.1.32.79/29	168.1.32.73
Sydney	au-syd	syd01	168.1.36.95/29	168.1.36.89
Sydney	au-syd	syd01	168.1.62.231/29	168.1.62.225
Sydney	au-syd	syd04	130.198.102.63/26	130.198.102.1
Sydney	au-syd	syd04	130.198.102.159/28	130.198.102.145
Sydney	au-syd	syd04	130.198.102.191/27	130.198.102.161

Sydney	au-syd	syd04	168.1.88.255/24	168.1.88.1
Sydney	au-syd	syd04	168.1.108.127/25	168.1.108.1
Sydney	au-syd	syd04	130.198.70.239/29	130.198.70.233
Sydney	au-syd	syd04	130.198.93.135/29	130.198.93.129
Sydney	au-syd	syd04	130.198.99.71/29	130.198.99.65
Sydney	au-syd	syd05	135.90.68.111/28	135.90.68.97
Sydney	au-syd	syd05	135.90.69.127/27	135.90.69.97
Sydney	au-syd	syd05	135.90.83.63/26	135.90.83.1
Sydney	au-syd	syd05	135.90.95.127/25	135.90.95.1
Sydney	au-syd	syd05	135.90.108.255/24	135.90.108.1
Sydney	au-syd	syd05	135.90.67.31/29	135.90.67.25
Sydney	au-syd	syd05	135.90.67.135/29	135.90.67.129
Sydney	au-syd	syd05	135.90.68.39/29	135.90.68.33

au-syd Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Sydney	au-syd	syd01	10.138.220.255/25	10.138.220.129
Sydney	au-syd	syd01	10.139.40.63/26	10.139.40.1
Sydney	au-syd	syd01	10.139.61.255/24	10.139.61.1
Sydney	au-syd	syd01	10.138.152.63/29	10.138.152.57
Sydney	au-syd	syd01	10.138.172.175/29	10.138.172.169
Sydney	au-syd	syd01	10.138.172.183/29	10.138.172.177
Sydney	au-syd	syd04	10.63.110.255/25	10.63.110.129
Sydney	au-syd	syd04	10.63.212.127/26	10.63.212.65
Sydney	au-syd	syd04	10.63.244.255/24	10.63.244.1
Sydney	au-syd	syd04	10.63.4.255/29	10.63.4.249
Sydney	au-syd	syd04	10.63.44.103/29	10.63.44.97
Sydney	au-syd	syd04	10.63.253.111/29	10.63.253.105
Sydney	au-syd	syd05	10.195.4.255/26	10.195.4.193

Sydney	au-syd	syd05	10.195.81.127/25	10.195.81.1
Sydney	au-syd	syd05	10.195.128.255/24	10.195.128.1
Sydney	au-syd	syd05	10.195.7.239/29	10.195.7.233
Sydney	au-syd	syd05	10.195.7.247/29	10.195.7.241
Sydney	au-syd	syd05	10.195.46.111/29	10.195.46.105

au-syd Private Subnets

jp-tok List

Public Subnets

Location	Region	Data center	Subnet	First IP
Tokyo	jp-tok	tok02	161.202.140.191/27	161.202.140.161
Tokyo	jp-tok	tok02	169.56.7.255/26	169.56.7.193
Tokyo	jp-tok	tok02	169.56.30.255/25	169.56.30.129
Tokyo	jp-tok	tok02	169.56.31.255/24	169.56.31.1
Tokyo	jp-tok	tok02	169.56.45.175/28	169.56.45.161
Tokyo	jp-tok	tok02	161.202.102.87/29	161.202.102.81
Tokyo	jp-tok	tok02	161.202.145.63/29	161.202.145.57
Tokyo	jp-tok	tok02	161.202.234.223/29	161.202.234.217
Tokyo	jp-tok	tok02	161.202.239.191/29	161.202.239.185
Tokyo	jp-tok	tok04	128.168.72.63/27	128.168.72.33
Tokyo	jp-tok	tok04	128.168.72.191/28	128.168.72.177
Tokyo	jp-tok	tok04	128.168.93.127/26	128.168.93.65
Tokyo	jp-tok	tok04	128.168.105.127/25	128.168.105.1
Tokyo	jp-tok	tok04	128.168.157.255/24	128.168.157.1
Tokyo	jp-tok	tok04	128.168.69.79/29	128.168.69.73
Tokyo	jp-tok	tok04	128.168.71.87/29	128.168.71.81
Tokyo	jp-tok	tok04	128.168.71.95/29	128.168.71.89
Tokyo	jp-tok	tok04	128.168.111.87/29	128.168.111.81
Tokyo	jp-tok	tok05	165.192.71.47/28	165.192.71.33
Tokyo	jp-tok	tok05	165.192.71.127/27	165.192.71.97

Tokyo	jp-tok	tok05	165.192.89.255/26	165.192.89.193
Tokyo	jp-tok	tok05	165.192.102.255/25	165.192.102.129
Tokyo	jp-tok	tok05	165.192.157.255/24	165.192.157.1
Tokyo	jp-tok	tok05	165.192.66.7/29	165.192.66.1
Tokyo	jp-tok	tok05	165.192.70.175/29	165.192.70.169
Tokyo	jp-tok	tok05	165.192.71.239/29	165.192.71.233
Tokyo	jp-tok	tok05	165.192.148.63/29	165.192.148.57

tok Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Tokyo	jp-tok	tok02	10.129.16.255/24	10.129.16.1
Tokyo	jp-tok	tok02	10.129.132.127/25	10.129.132.1
Tokyo	jp-tok	tok02	10.129.223.255/24	10.129.223.1
Tokyo	jp-tok	tok02	10.133.170.255/26	10.133.170.193
Tokyo	jp-tok	tok02	10.129.121.239/29	10.129.121.233
Tokyo	jp-tok	tok02	10.132.40.151/29	10.132.40.145
Tokyo	jp-tok	tok02	10.133.92.159/29	10.133.92.153
Tokyo	jp-tok	tok02	10.133.227.231/29	10.133.227.225
Tokyo	jp-tok	tok04	10.192.23.191/26	10.192.23.129
Tokyo	jp-tok	tok04	10.192.112.127/25	10.192.112.1
Tokyo	jp-tok	tok04	10.192.131.255/24	10.192.131.1
Tokyo	jp-tok	tok04	10.192.184.255/24	10.192.184.1
Tokyo	jp-tok	tok04	10.192.18.87/29	10.192.18.81
Tokyo	jp-tok	tok04	10.192.22.79/29	10.192.22.73
Tokyo	jp-tok	tok04	10.192.69.7/29	10.192.69.1
Tokyo	jp-tok	tok04	10.192.138.15/29	10.192.138.9
Tokyo	jp-tok	tok05	10.193.12.191/26	10.193.12.129
Tokyo	jp-tok	tok05	10.193.105.255/25	10.193.105.129
Tokyo	jp-tok	tok05	10.193.193.255/24	10.193.193.1

Tokyo	jp-tok	tok05	10.193.15.167/29	10.193.15.161
Tokyo	jp-tok	tok05	10.193.19.39/29	10.193.19.33
Tokyo	jp-tok	tok05	10.193.76.255/29	10.193.76.249
Tokyo	jp-tok	tok05	10.193.103.175/29	10.193.103.169

tok Private Subnets

`jp-osa` List

Public Subnets

Location	Region	Data center	Subnet	First IP
Osaka	jp-osa	osa21	163.68.68.127/28	163.68.68.113
Osaka	jp-osa	osa21	163.68.73.127/27	163.68.73.97
Osaka	jp-osa	osa21	163.68.96.191/26	163.68.96.129
Osaka	jp-osa	osa21	163.68.67.119/29	163.68.67.113
Osaka	jp-osa	osa21	163.68.70.63/29	163.68.70.57
Osaka	jp-osa	osa22	163.69.65.111/28	163.69.65.97
Osaka	jp-osa	osa22	163.69.68.31/27	163.69.68.1
Osaka	jp-osa	osa22	163.69.71.255/26	163.69.71.193
Osaka	jp-osa	osa22	163.69.65.55/29	163.69.65.49
Osaka	jp-osa	osa22	163.69.65.63/29	163.69.65.57
Osaka	jp-osa	osa23	163.73.65.175/28	163.73.65.161
Osaka	jp-osa	osa23	163.73.68.95/27	163.73.68.65
Osaka	jp-osa	osa23	163.73.71.63/26	163.73.71.1
Osaka	jp-osa	osa23	163.73.67.191/29	163.73.67.185
Osaka	jp-osa	osa23	163.73.67.231/29	163.73.67.225

jp-osa Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Osaka	jp-osa	osa21	10.8.22.255/26	10.8.22.193
Osaka	jp-osa	osa21	10.8.66.127/25	10.8.66.1
Osaka	jp-osa	osa21	10.8.16.159/29	10.8.16.153

Osaka	jp-osa	osa21	10.8.16.167/29	10.8.16.161
Osaka	jp-osa	osa22	10.9.12.127/26	10.9.12.65
Osaka	jp-osa	osa22	10.9.32.255/25	10.9.32.129
Osaka	jp-osa	osa22	10.9.6.207/29	10.9.6.201
Osaka	jp-osa	osa22	10.9.6.215/29	10.9.6.209
Osaka	jp-osa	osa23	10.10.12.255/26	10.10.12.193
Osaka	jp-osa	osa23	10.10.28.255/25	10.10.28.129
Osaka	jp-osa	osa23	10.10.8.7/29	10.10.8.1
Osaka	jp-osa	osa23	10.10.8.15/29	10.10.8.9

jp-osa Private Subnets

`us-east` List

Public Subnets

Location	Region	Data center	Subnet	First IP
Washington DC	us-east	wdc04	52.116.78.127/25	52.116.78.1
Washington DC	us-east	wdc04	52.116.115.255/24	52.116.115.1
Washington DC	us-east	wdc04	150.239.70.255/24	150.239.70.1
Washington DC	us-east	wdc04	150.239.101.255/24	150.239.101.1
Washington DC	us-east	wdc04	150.239.107.255/24	150.239.107.1
Washington DC	us-east	wdc04	169.47.179.63/26	169.47.179.1
Washington DC	us-east	wdc04	169.63.72.175/28	169.63.72.161
Washington DC	us-east	wdc04	169.63.83.255/24	169.63.83.1
Washington DC	us-east	wdc04	169.63.121.159/27	169.63.121.129
Washington DC	us-east	wdc04	52.116.73.231/29	52.116.73.225
Washington DC	us-east	wdc04	169.47.179.231/29	169.47.179.225
Washington DC	us-east	wdc04	169.63.86.31/29	169.63.86.25
Washington DC	us-east	wdc04	169.63.95.247/29	169.63.95.241
Washington DC	us-east	wdc04	169.63.111.119/29	169.63.111.113
Washington DC	us-east	wdc04	169.63.121.55/29	169.63.121.49
Washington DC	us-east	wdc04	169.63.121.63/29	169.63.121.57

Washington DC	us-east	wdc04	169.63.123.167/29	169.63.123.161
Washington DC	us-east	wdc04	169.63.125.223/29	169.63.125.217
Washington DC	us-east	wdc06	169.59.138.255/24	169.59.138.1
Washington DC	us-east	wdc06	169.59.144.255/24	169.59.144.1
Washington DC	us-east	wdc06	169.59.151.255/24	169.59.151.1
Washington DC	us-east	wdc06	169.59.158.255/24	169.59.158.1
Washington DC	us-east	wdc06	169.59.183.255/24	169.59.183.1
Washington DC	us-east	wdc06	169.63.128.191/27	169.63.128.161
Washington DC	us-east	wdc06	169.63.135.191/28	169.63.135.177
Washington DC	us-east	wdc06	169.63.139.255/26	169.63.139.193
Washington DC	us-east	wdc06	169.63.172.127/25	169.63.172.1
Washington DC	us-east	wdc06	169.59.145.239/29	169.59.145.233
Washington DC	us-east	wdc06	169.60.65.55/29	169.60.65.49
Washington DC	us-east	wdc06	169.60.89.7/29	169.60.89.1
Washington DC	us-east	wdc06	169.60.93.7/29	169.60.93.1
Washington DC	us-east	wdc06	169.60.95.151/29	169.60.95.145
Washington DC	us-east	wdc06	169.63.129.119/29	169.63.129.113
Washington DC	us-east	wdc06	169.63.141.239/29	169.63.141.233
Washington DC	us-east	wdc06	169.63.149.199/29	169.63.149.193
Washington DC	us-east	wdc07	52.117.104.255/24	52.117.104.1
Washington DC	us-east	wdc07	150.239.194.255/24	150.239.194.1
Washington DC	us-east	wdc07	150.239.231.255/24	150.239.231.1
Washington DC	us-east	wdc07	150.239.232.255/24	150.239.232.1
Washington DC	us-east	wdc07	169.61.123.95/28	169.61.123.81
Washington DC	us-east	wdc07	169.62.36.255/24	169.62.36.1
Washington DC	us-east	wdc07	169.62.42.63/27	169.62.42.33
Washington DC	us-east	wdc07	169.62.54.191/26	169.62.54.129
Washington DC	us-east	wdc07	169.62.60.127/25	169.62.60.1

Washington DC	us-east	wdc07	52.117.76.47/29	52.117.76.41
Washington DC	us-east	wdc07	52.117.85.71/29	52.117.85.65
Washington DC	us-east	wdc07	169.61.73.231/29	169.61.73.225
Washington DC	us-east	wdc07	169.61.95.15/29	169.61.95.9
Washington DC	us-east	wdc07	169.61.113.39/29	169.61.113.33
Washington DC	us-east	wdc07	169.61.113.47/29	169.61.113.41
Washington DC	us-east	wdc07	169.61.122.159/29	169.61.122.153
Washington DC	us-east	wdc07	169.62.6.87/29	169.62.6.81

us-east Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Washington DC	us-east	wdc04	10.65.53.255/24	10.65.53.1
Washington DC	us-east	wdc04	10.65.119.255/25	10.65.119.129
Washington DC	us-east	wdc04	10.65.145.255/26	10.65.145.193
Washington DC	us-east	wdc04	10.211.67.255/24	10.211.67.1
Washington DC	us-east	wdc04	10.211.96.255/24	10.211.96.1
Washington DC	us-east	wdc04	10.211.126.255/24	10.211.126.1
Washington DC	us-east	wdc04	10.211.198.255/24	10.211.198.1
Washington DC	us-east	wdc04	10.65.0.7/29	10.65.0.1
Washington DC	us-east	wdc04	10.65.99.103/29	10.65.99.97
Washington DC	us-east	wdc04	10.65.145.95/29	10.65.145.89
Washington DC	us-east	wdc04	10.65.152.223/29	10.65.152.217
Washington DC	us-east	wdc04	10.65.179.159/29	10.65.179.153
Washington DC	us-east	wdc04	10.65.218.247/29	10.65.218.241
Washington DC	us-east	wdc04	10.65.224.15/29	10.65.224.9
Washington DC	us-east	wdc04	10.65.224.23/29	10.65.224.17
Washington DC	us-east	wdc04	10.65.224.47/29	10.65.224.41
Washington DC	us-east	wdc06	10.188.113.255/24	10.188.113.1
Washington DC	us-east	wdc06	10.188.150.255/24	10.188.150.1

Washington DC	us-east	wdc06	10.188.160.127/26	10.188.160.65
Washington DC	us-east	wdc06	10.189.7.127/25	10.189.7.1
Washington DC	us-east	wdc06	10.189.91.255/24	10.189.91.1
Washington DC	us-east	wdc06	10.189.159.255/24	10.189.159.1
Washington DC	us-east	wdc06	10.189.223.255/24	10.189.223.1
Washington DC	us-east	wdc06	10.188.34.31/29	10.188.34.25
Washington DC	us-east	wdc06	10.188.35.231/29	10.188.35.225
Washington DC	us-east	wdc06	10.188.53.127/29	10.188.53.121
Washington DC	us-east	wdc06	10.188.137.247/29	10.188.137.241
Washington DC	us-east	wdc06	10.188.140.143/29	10.188.140.137
Washington DC	us-east	wdc06	10.188.199.95/29	10.188.199.89
Washington DC	us-east	wdc06	10.189.13.23/29	10.189.13.17
Washington DC	us-east	wdc06	10.189.45.223/29	10.189.45.217
Washington DC	us-east	wdc07	10.39.58.255/24	10.39.58.1
Washington DC	us-east	wdc07	10.39.62.255/24	10.39.62.1
Washington DC	us-east	wdc07	10.39.119.255/24	10.39.119.1
Washington DC	us-east	wdc07	10.190.86.191/26	10.190.86.129
Washington DC	us-east	wdc07	10.190.231.255/24	10.190.231.1
Washington DC	us-east	wdc07	10.191.126.127/25	10.191.126.1
Washington DC	us-east	wdc07	10.191.213.255/24	10.191.213.1
Washington DC	us-east	wdc07	10.190.22.175/29	10.190.22.169
Washington DC	us-east	wdc07	10.190.123.143/29	10.190.123.137
Washington DC	us-east	wdc07	10.190.166.39/29	10.190.166.33
Washington DC	us-east	wdc07	10.190.233.215/29	10.190.233.209
Washington DC	us-east	wdc07	10.191.11.127/29	10.191.11.121
Washington DC	us-east	wdc07	10.191.12.175/29	10.191.12.169
Washington DC	us-east	wdc07	10.191.49.247/29	10.191.49.241
Washington DC	us-east	wdc07	10.191.115.7/29	10.191.115.1

us-east Private Subnets

eu-de List

Public Subnets

Location	Region	Data center	Subnet	First IP
Frankfurt	eu-de	fra02	158.177.41.255/24	158.177.41.1
Frankfurt	eu-de	fra02	158.177.56.255/24	158.177.56.1
Frankfurt	eu-de	fra02	158.177.61.255/24	158.177.61.1
Frankfurt	eu-de	fra02	158.177.77.63/27	158.177.77.33
Frankfurt	eu-de	fra02	158.177.87.255/25	158.177.87.129
Frankfurt	eu-de	fra02	158.177.155.63/26	158.177.155.1
Frankfurt	eu-de	fra02	158.177.185.255/24	128.0.0.1
Frankfurt	eu-de	fra02	158.177.186.255/24	128.0.0.1
Frankfurt	eu-de	fra02	158.177.187.255/24	128.0.0.1
Frankfurt	eu-de	fra02	158.177.241.255/24	128.0.0.1
Frankfurt	eu-de	fra02	158.177.110.111/29	128.0.0.1
Frankfurt	eu-de	fra02	158.177.221.175/29	128.0.0.1
Frankfurt	eu-de	fra02	159.122.89.23/29	159.122.89.17
Frankfurt	eu-de	fra02	159.122.97.23/29	159.122.97.17
Frankfurt	eu-de	fra02	159.122.97.47/29	159.122.97.41
Frankfurt	eu-de	fra02	159.122.108.215/29	128.0.0.1
Frankfurt	eu-de	fra02	159.122.108.223/29	128.0.0.1
Frankfurt	eu-de	fra02	169.50.10.15/29	169.50.10.9
Frankfurt	eu-de	fra02	169.50.13.207/29	169.50.13.201
Frankfurt	eu-de	fra02	169.50.15.31/29	169.50.15.25
Frankfurt	eu-de	fra02	169.50.15.191/29	169.50.15.185
Frankfurt	eu-de	fra02	169.50.25.231/29	169.50.25.225
Frankfurt	eu-de	fra02	169.50.35.111/29	169.50.35.105
Frankfurt	eu-de	fra02	169.50.35.239/29	169.50.35.233
Frankfurt	eu-de	fra02	169.50.53.247/29	169.50.53.241
Frankfurt	eu-de	fra02	169.50.54.23/29	169.50.54.17

Frankfurt	eu-de	fra04	161.156.2.255/24	161.156.2.1
Frankfurt	eu-de	fra04	161.156.15.255/24	161.156.15.1
Frankfurt	eu-de	fra04	161.156.16.255/24	161.156.16.1
Frankfurt	eu-de	fra04	161.156.25.255/24	161.156.25.1
Frankfurt	eu-de	fra04	161.156.51.255/24	161.156.51.1
Frankfurt	eu-de	fra04	161.156.56.255/24	161.156.56.1
Frankfurt	eu-de	fra04	161.156.95.223/27	161.156.95.193
Frankfurt	eu-de	fra04	161.156.97.63/26	161.156.97.1
Frankfurt	eu-de	fra04	161.156.131.255/25	128.0.0.1
Frankfurt	eu-de	fra04	161.156.152.255/24	128.0.0.1
Frankfurt	eu-de	fra04	161.156.1.79/29	161.156.1.73
Frankfurt	eu-de	fra04	161.156.8.47/29	161.156.8.41
Frankfurt	eu-de	fra04	161.156.38.207/29	161.156.38.201
Frankfurt	eu-de	fra04	161.156.67.247/29	161.156.67.241
Frankfurt	eu-de	fra04	161.156.69.135/29	161.156.69.129
Frankfurt	eu-de	fra04	161.156.69.143/29	161.156.69.137
Frankfurt	eu-de	fra04	161.156.107.159/29	128.0.0.1
Frankfurt	eu-de	fra04	161.156.111.55/29	161.156.111.49
Frankfurt	eu-de	fra04	161.156.111.79/29	161.156.111.73
Frankfurt	eu-de	fra04	161.156.122.239/29	128.0.0.1
Frankfurt	eu-de	fra04	161.156.132.103/29	128.0.0.1
Frankfurt	eu-de	fra04	161.156.148.215/29	128.0.0.1
Frankfurt	eu-de	fra04	161.156.157.135/29	128.0.0.1
Frankfurt	eu-de	fra04	161.156.177.183/29	128.0.0.1
Frankfurt	eu-de	fra04	161.156.185.31/29	161.156.185.25
Frankfurt	eu-de	fra04	161.156.190.127/29	128.0.0.1
Frankfurt	eu-de	fra05	149.81.73.159/28	149.81.73.145
Frankfurt	eu-de	fra05	149.81.80.255/27	149.81.80.225

Frankfurt	eu-de	fra05	149.81.114.63/26	149.81.114.1
Frankfurt	eu-de	fra05	149.81.132.127/25	149.81.132.1
Frankfurt	eu-de	fra05	149.81.139.255/24	149.81.139.1
Frankfurt	eu-de	fra05	149.81.150.255/24	149.81.150.1
Frankfurt	eu-de	fra05	149.81.183.255/24	149.81.183.1
Frankfurt	eu-de	fra05	149.81.206.255/24	149.81.206.1
Frankfurt	eu-de	fra05	149.81.208.255/24	149.81.208.1
Frankfurt	eu-de	fra05	149.81.220.255/24	149.81.220.1
Frankfurt	eu-de	fra05	149.81.225.255/24	149.81.225.1
Frankfurt	eu-de	fra05	149.81.65.255/29	149.81.65.249
Frankfurt	eu-de	fra05	149.81.77.207/29	149.81.77.201
Frankfurt	eu-de	fra05	149.81.79.183/29	149.81.79.177
Frankfurt	eu-de	fra05	149.81.83.215/29	149.81.83.209
Frankfurt	eu-de	fra05	149.81.84.103/29	149.81.84.97
Frankfurt	eu-de	fra05	149.81.87.143/29	149.81.87.137
Frankfurt	eu-de	fra05	149.81.87.151/29	149.81.87.145
Frankfurt	eu-de	fra05	149.81.98.167/29	149.81.98.161
Frankfurt	eu-de	fra05	149.81.100.207/29	149.81.100.201
Frankfurt	eu-de	fra05	149.81.101.119/29	149.81.101.113
Frankfurt	eu-de	fra05	149.81.106.239/29	149.81.106.233
Frankfurt	eu-de	fra05	149.81.114.175/29	149.81.114.169
Frankfurt	eu-de	fra05	149.81.142.47/29	149.81.142.41
Frankfurt	eu-de	fra05	149.81.148.79/29	149.81.148.73
Frankfurt	eu-de	fra05	149.81.171.231/29	149.81.171.225
Frankfurt	eu-de	fra05	149.81.180.207/29	149.81.180.201
Paris	eu-de	par01	159.8.70.111/28	159.8.70.97
Paris	eu-de	par01	159.8.90.127/26	159.8.90.65
Paris	eu-de	par01	159.8.114.127/27	159.8.114.97

Paris	eu-de	par01	159.8.78.239/29	159.8.78.233
Paris	eu-de	par01	159.8.94.95/29	159.8.94.89

eu-de Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Frankfurt	eu-de	fra02	10.20.34.255/24	10.20.34.1
Frankfurt	eu-de	fra02	10.134.228.255/24	10.134.228.1
Frankfurt	eu-de	fra02	10.135.51.255/24	10.135.51.1
Frankfurt	eu-de	fra02	10.135.164.255/24	10.135.164.1
Frankfurt	eu-de	fra02	10.194.5.63/26	10.194.5.1
Frankfurt	eu-de	fra02	10.194.98.255/25	10.194.98.129
Frankfurt	eu-de	fra02	10.194.205.255/24	10.194.205.1
Frankfurt	eu-de	fra02	10.215.194.255/24	10.215.194.1
Frankfurt	eu-de	fra02	10.215.250.255/24	10.215.250.1
Frankfurt	eu-de	fra02	10.134.41.111/29	10.134.41.105
Frankfurt	eu-de	fra02	10.134.57.143/29	10.134.57.137
Frankfurt	eu-de	fra02	10.134.68.47/29	10.134.68.41
Frankfurt	eu-de	fra02	10.134.191.167/29	10.134.191.161
Frankfurt	eu-de	fra02	10.134.195.207/29	10.134.195.201
Frankfurt	eu-de	fra02	10.134.199.247/29	10.134.199.241
Frankfurt	eu-de	fra02	10.134.236.207/29	10.134.236.201
Frankfurt	eu-de	fra02	10.134.247.207/29	10.134.247.201
Frankfurt	eu-de	fra02	10.135.17.103/29	10.135.17.97
Frankfurt	eu-de	fra02	10.135.61.239/29	10.135.61.233
Frankfurt	eu-de	fra02	10.135.84.111/29	10.135.84.105
Frankfurt	eu-de	fra02	10.135.87.103/29	10.135.87.97
Frankfurt	eu-de	fra02	10.135.120.143/29	10.135.120.137
Frankfurt	eu-de	fra02	10.135.123.31/29	10.135.123.25
Frankfurt	eu-de	fra02	10.135.158.47/29	10.135.158.41

Frankfurt	eu-de	fra02	10.135.180.7/29	10.135.180.1
Frankfurt	eu-de	fra02	10.135.180.15/29	10.135.180.9
Frankfurt	eu-de	fra04	10.21.72.255/24	10.21.72.1
Frankfurt	eu-de	fra04	10.75.27.255/24	10.75.27.1
Frankfurt	eu-de	fra04	10.75.68.191/26	10.75.68.129
Frankfurt	eu-de	fra04	10.75.134.127/25	10.75.134.1
Frankfurt	eu-de	fra04	10.75.234.255/24	10.75.234.1
Frankfurt	eu-de	fra04	10.240.59.255/24	10.240.59.1
Frankfurt	eu-de	fra04	10.240.146.255/24	10.240.146.1
Frankfurt	eu-de	fra04	10.240.181.255/24	10.240.181.1
Frankfurt	eu-de	fra04	10.240.213.255/24	10.240.213.1
Frankfurt	eu-de	fra04	10.75.35.55/29	10.75.35.49
Frankfurt	eu-de	fra04	10.75.74.151/29	10.75.74.145
Frankfurt	eu-de	fra04	10.75.80.239/29	10.75.80.233
Frankfurt	eu-de	fra04	10.75.82.23/29	10.75.82.17
Frankfurt	eu-de	fra04	10.75.82.39/29	10.75.82.33
Frankfurt	eu-de	fra04	10.75.82.47/29	10.75.82.41
Frankfurt	eu-de	fra04	10.75.127.79/29	10.75.127.73
Frankfurt	eu-de	fra04	10.75.166.47/29	10.75.166.41
Frankfurt	eu-de	fra04	10.75.166.79/29	10.75.166.73
Frankfurt	eu-de	fra04	10.75.210.39/29	10.75.210.33
Frankfurt	eu-de	fra04	10.75.219.103/29	10.75.219.97
Frankfurt	eu-de	fra04	10.75.221.103/29	10.75.221.97
Frankfurt	eu-de	fra04	10.240.20.103/29	10.240.20.97
Frankfurt	eu-de	fra04	10.240.53.215/29	10.240.53.209
Frankfurt	eu-de	fra04	10.240.160.31/29	10.240.160.25
Frankfurt	eu-de	fra04	10.240.166.183/29	10.240.166.177
Frankfurt	eu-de	fra04	10.240.228.71/29	10.240.228.65

Frankfurt	eu-de	fra05	10.13.21.255/24	10.13.21.1
Frankfurt	eu-de	fra05	10.13.175.255/24	10.13.175.1
Frankfurt	eu-de	fra05	10.13.205.255/24	10.13.205.1
Frankfurt	eu-de	fra05	10.13.217.255/24	10.13.217.1
Frankfurt	eu-de	fra05	10.13.224.255/24	10.13.224.1
Frankfurt	eu-de	fra05	10.123.14.63/26	10.123.14.1
Frankfurt	eu-de	fra05	10.123.55.127/25	10.123.55.1
Frankfurt	eu-de	fra05	10.123.167.255/24	10.123.167.1
Frankfurt	eu-de	fra05	10.123.241.255/24	10.123.241.1
Frankfurt	eu-de	fra05	10.13.10.151/29	10.13.10.145
Frankfurt	eu-de	fra05	10.13.35.71/29	10.13.35.65
Frankfurt	eu-de	fra05	10.123.22.151/29	10.123.22.145
Frankfurt	eu-de	fra05	10.123.53.15/29	10.123.53.9
Frankfurt	eu-de	fra05	10.123.59.103/29	10.123.59.97
Frankfurt	eu-de	fra05	10.123.60.167/29	10.123.60.161
Frankfurt	eu-de	fra05	10.123.63.199/29	10.123.63.193
Frankfurt	eu-de	fra05	10.123.64.199/29	10.123.64.193
Frankfurt	eu-de	fra05	10.123.64.247/29	10.123.64.241
Frankfurt	eu-de	fra05	10.123.87.191/29	10.123.87.185
Frankfurt	eu-de	fra05	10.123.99.127/29	10.123.99.121
Frankfurt	eu-de	fra05	10.123.108.111/29	10.123.108.105
Frankfurt	eu-de	fra05	10.123.123.151/29	10.123.123.145
Frankfurt	eu-de	fra05	10.123.179.15/29	10.123.179.9
Frankfurt	eu-de	fra05	10.123.182.127/29	10.123.182.121
Frankfurt	eu-de	fra05	10.123.208.95/29	10.123.208.89
Frankfurt	eu-de	fra05	10.123.253.207/29	10.123.253.201
Paris	eu-de	par01	10.126.152.127/25	10.126.152.1
Paris	eu-de	par01	10.127.213.63/26	10.127.213.1

Paris	eu-de	par01	10.126.23.255/29	10.126.23.249
Paris	eu-de	par01	10.126.100.135/29	10.126.100.129

eu-de Private Subnets

us-south List

Public Subnets

Location	Region	Data center	Subnet	First IP
Dallas	us-south	dal10	52.116.167.255/24	52.116.167.1
Dallas	us-south	dal10	52.116.179.175/28	52.116.179.161
Dallas	us-south	dal10	52.116.190.63/26	52.116.190.1
Dallas	us-south	dal10	52.117.150.127/25	52.117.150.1
Dallas	us-south	dal10	52.117.184.255/24	52.117.184.1
Dallas	us-south	dal10	52.118.4.255/24	52.118.4.1
Dallas	us-south	dal10	52.118.10.255/24	52.118.10.1
Dallas	us-south	dal10	52.118.48.255/24	52.118.48.1
Dallas	us-south	dal10	52.118.138.255/24	52.118.138.1
Dallas	us-south	dal10	52.118.163.127/25	52.118.163.1
Dallas	us-south	dal10	150.238.244.255/24	150.238.244.1
Dallas	us-south	dal10	150.239.18.63/26	150.239.18.1
Dallas	us-south	dal10	150.239.46.255/24	150.239.46.1
Dallas	us-south	dal10	150.239.63.255/24	150.239.63.1
Dallas	us-south	dal10	150.240.131.255/24	150.240.131.1
Dallas	us-south	dal10	150.240.158.255/24	150.240.158.1
Dallas	us-south	dal10	169.46.57.31/27	169.46.57.1
Dallas	us-south	dal10	169.47.212.191/27	169.47.212.161
Dallas	us-south	dal10	169.48.177.159/28	169.48.177.145
Dallas	us-south	dal10	169.60.22.255/27	169.60.22.225
Dallas	us-south	dal10	169.61.212.255/26	169.61.212.193
Dallas	us-south	dal10	169.61.219.239/28	169.61.219.225
Dallas	us-south	dal10	169.63.201.255/25	169.63.201.129

Dallas	us-south	dal10	169.63.204.255/24	169.63.204.1
Dallas	us-south	dal10	52.116.160.151/29	52.116.160.145
Dallas	us-south	dal10	52.116.190.135/29	52.116.190.129
Dallas	us-south	dal10	52.116.190.239/29	52.116.190.233
Dallas	us-south	dal10	52.117.135.111/29	52.117.135.105
Dallas	us-south	dal10	52.117.148.31/29	52.117.148.25
Dallas	us-south	dal10	52.117.166.239/29	52.117.166.233
Dallas	us-south	dal10	52.118.41.103/29	52.118.41.97
Dallas	us-south	dal10	52.118.153.159/29	52.118.153.153
Dallas	us-south	dal10	150.238.4.151/29	150.238.4.145
Dallas	us-south	dal10	150.238.229.39/29	150.238.229.33
Dallas	us-south	dal10	150.238.247.167/29	150.238.247.161
Dallas	us-south	dal10	169.46.13.239/29	169.46.13.233
Dallas	us-south	dal10	169.46.17.87/29	169.46.17.81
Dallas	us-south	dal10	169.46.18.23/29	169.46.18.17
Dallas	us-south	dal10	169.46.21.175/29	169.46.21.169
Dallas	us-south	dal10	169.46.21.183/29	169.46.21.177
Dallas	us-south	dal10	169.46.25.103/29	169.46.25.97
Dallas	us-south	dal10	169.46.31.175/29	169.46.31.169
Dallas	us-south	dal10	169.46.39.199/29	169.46.39.193
Dallas	us-south	dal10	169.46.40.7/29	169.46.40.1
Dallas	us-south	dal10	169.46.42.215/29	169.46.42.209
Dallas	us-south	dal10	169.46.50.239/29	169.46.50.233
Dallas	us-south	dal10	169.46.51.63/29	169.46.51.57
Dallas	us-south	dal10	169.46.51.111/29	169.46.51.105
Dallas	us-south	dal10	169.46.60.23/29	169.46.60.17
Dallas	us-south	dal10	169.46.65.191/29	169.46.65.185
Dallas	us-south	dal10	169.46.71.143/29	169.46.71.137

Dallas	us-south	dal10	169.46.76.71/29	169.46.76.65
Dallas	us-south	dal10	169.46.78.15/29	169.46.78.9
Dallas	us-south	dal10	169.46.81.175/29	169.46.81.169
Dallas	us-south	dal10	169.46.91.55/29	169.46.91.49
Dallas	us-south	dal10	169.46.97.79/29	169.46.97.73
Dallas	us-south	dal10	169.46.99.103/29	169.46.99.97
Dallas	us-south	dal10	169.46.99.175/29	169.46.99.169
Dallas	us-south	dal10	169.46.110.7/29	169.46.110.1
Dallas	us-south	dal10	169.46.110.87/29	169.46.110.81
Dallas	us-south	dal10	169.46.120.183/29	169.46.120.177
Dallas	us-south	dal10	169.46.121.215/29	169.46.121.209
Dallas	us-south	dal10	169.47.193.79/29	169.47.193.73
Dallas	us-south	dal10	169.47.194.95/29	169.47.194.89
Dallas	us-south	dal10	169.47.195.47/29	169.47.195.41
Dallas	us-south	dal10	169.47.195.111/29	169.47.195.105
Dallas	us-south	dal10	169.47.197.207/29	169.47.197.201
Dallas	us-south	dal10	169.47.203.167/29	169.47.203.161
Dallas	us-south	dal10	169.47.205.31/29	169.47.205.25
Dallas	us-south	dal10	169.47.214.167/29	169.47.214.161
Dallas	us-south	dal10	169.47.228.255/29	169.47.228.249
Dallas	us-south	dal10	169.47.229.119/29	169.47.229.113
Dallas	us-south	dal10	169.47.233.167/29	169.47.233.161
Dallas	us-south	dal10	169.47.236.111/29	169.47.236.105
Dallas	us-south	dal10	169.47.246.39/29	169.47.246.33
Dallas	us-south	dal10	169.47.251.87/29	169.47.251.81
Dallas	us-south	dal10	169.47.252.135/29	169.47.252.129
Dallas	us-south	dal10	169.48.135.79/29	169.48.135.73
Dallas	us-south	dal10	169.48.139.79/29	169.48.139.73

Dallas	us-south	dal10	169.48.140.143/29	169.48.140.137
Dallas	us-south	dal10	169.48.161.199/29	169.48.161.193
Dallas	us-south	dal10	169.48.163.135/29	169.48.163.129
Dallas	us-south	dal10	169.48.164.151/29	169.48.164.145
Dallas	us-south	dal10	169.48.171.87/29	169.48.171.81
Dallas	us-south	dal10	169.48.173.135/29	169.48.173.129
Dallas	us-south	dal10	169.48.176.15/29	169.48.176.9
Dallas	us-south	dal10	169.48.177.103/29	169.48.177.97
Dallas	us-south	dal10	169.48.184.175/29	169.48.184.169
Dallas	us-south	dal10	169.60.28.103/29	169.60.28.97
Dallas	us-south	dal10	169.60.31.55/29	169.60.31.49
Dallas	us-south	dal10	169.60.37.167/29	169.60.37.161
Dallas	us-south	dal10	169.60.43.79/29	169.60.43.73
Dallas	us-south	dal10	169.60.61.223/29	169.60.61.217
Dallas	us-south	dal10	169.60.193.191/29	169.60.193.185
Dallas	us-south	dal10	169.60.196.47/29	169.60.196.41
Dallas	us-south	dal10	169.60.198.143/29	169.60.198.137
Dallas	us-south	dal10	169.60.199.127/29	169.60.199.121
Dallas	us-south	dal10	169.60.200.95/29	169.60.200.89
Dallas	us-south	dal10	169.60.202.255/29	169.60.202.249
Dallas	us-south	dal10	169.60.206.7/29	169.60.206.1
Dallas	us-south	dal10	169.60.224.175/29	169.60.224.169
Dallas	us-south	dal10	169.60.229.175/29	169.60.229.169
Dallas	us-south	dal10	169.60.235.247/29	169.60.235.241
Dallas	us-south	dal10	169.60.239.175/29	169.60.239.169
Dallas	us-south	dal10	169.60.240.47/29	169.60.240.41
Dallas	us-south	dal10	169.60.241.23/29	169.60.241.17
Dallas	us-south	dal10	169.61.193.127/29	169.61.193.121

Dallas	us-south	dal10	169.61.196.215/29	169.61.196.209
Dallas	us-south	dal10	169.61.214.175/29	169.61.214.169
Dallas	us-south	dal10	169.61.216.143/29	169.61.216.137
Dallas	us-south	dal10	169.61.218.23/29	169.61.218.17
Dallas	us-south	dal10	169.61.235.63/29	169.61.235.57
Dallas	us-south	dal10	169.63.194.159/29	169.63.194.153
Dallas	us-south	dal10	169.63.195.39/29	169.63.195.33
Dallas	us-south	dal10	169.63.218.119/29	169.63.218.113
Dallas	us-south	dal10	169.63.220.39/29	169.63.220.33
Dallas	us-south	dal12	50.22.135.255/24	50.22.135.1
Dallas	us-south	dal12	52.116.212.255/24	52.116.212.1
Dallas	us-south	dal12	52.116.218.191/27	52.116.218.161
Dallas	us-south	dal12	52.116.229.255/25	52.116.229.129
Dallas	us-south	dal12	52.116.252.255/24	52.116.252.1
Dallas	us-south	dal12	52.118.193.255/24	52.118.193.1
Dallas	us-south	dal12	52.118.197.255/24	52.118.197.1
Dallas	us-south	dal12	52.118.203.255/24	52.118.203.1
Dallas	us-south	dal12	52.118.233.255/24	52.118.233.1
Dallas	us-south	dal12	52.118.236.255/24	52.118.236.1
Dallas	us-south	dal12	52.118.238.255/24	52.118.238.1
Dallas	us-south	dal12	150.239.149.255/24	150.239.149.1
Dallas	us-south	dal12	150.239.187.255/24	150.239.187.1
Dallas	us-south	dal12	169.47.108.255/26	169.47.108.193
Dallas	us-south	dal12	169.48.215.191/27	169.48.215.161
Dallas	us-south	dal12	169.59.194.255/24	169.59.194.1
Dallas	us-south	dal12	169.59.234.255/24	169.59.234.1
Dallas	us-south	dal12	169.61.157.255/25	169.61.157.129
Dallas	us-south	dal12	169.61.167.31/28	169.61.167.17

Dallas	us-south	dal12	169.61.169.63/28	169.61.169.49
Dallas	us-south	dal12	169.63.57.127/26	169.63.57.65
Dallas	us-south	dal12	50.22.153.191/29	50.22.153.185
Dallas	us-south	dal12	50.22.175.255/24	50.22.175.1
Dallas	us-south	dal12	52.116.234.199/29	52.116.234.193
Dallas	us-south	dal12	52.116.244.119/29	52.116.244.113
Dallas	us-south	dal12	52.116.254.31/29	52.116.254.25
Dallas	us-south	dal12	169.47.101.7/29	169.47.101.1
Dallas	us-south	dal12	169.47.101.15/29	169.47.101.9
Dallas	us-south	dal12	169.47.101.63/29	169.47.101.57
Dallas	us-south	dal12	169.47.107.95/29	169.47.107.89
Dallas	us-south	dal12	169.47.107.151/29	169.47.107.145
Dallas	us-south	dal12	169.47.109.87/29	169.47.109.81
Dallas	us-south	dal12	169.47.111.79/29	169.47.111.73
Dallas	us-south	dal12	169.47.113.223/29	169.47.113.217
Dallas	us-south	dal12	169.47.124.95/29	169.47.124.89
Dallas	us-south	dal12	169.48.207.175/29	169.48.207.169
Dallas	us-south	dal12	169.48.216.47/29	169.48.216.41
Dallas	us-south	dal12	169.48.218.143/29	169.48.218.137
Dallas	us-south	dal12	169.48.222.135/29	169.48.222.129
Dallas	us-south	dal12	169.48.236.183/29	169.48.236.177
Dallas	us-south	dal12	169.48.240.199/29	169.48.240.193
Dallas	us-south	dal12	169.48.241.167/29	169.48.241.161
Dallas	us-south	dal12	169.59.203.47/29	169.59.203.41
Dallas	us-south	dal12	169.59.203.247/29	169.59.203.241
Dallas	us-south	dal12	169.59.225.191/29	169.59.225.185
Dallas	us-south	dal12	169.59.255.119/29	169.59.255.113
Dallas	us-south	dal12	169.61.133.159/29	169.61.133.153

Dallas	us-south	dal12	169.61.133.223/29	169.61.133.217
Dallas	us-south	dal12	169.61.136.175/29	169.61.136.169
Dallas	us-south	dal12	169.61.138.231/29	169.61.138.225
Dallas	us-south	dal12	169.61.139.79/29	169.61.139.73
Dallas	us-south	dal12	169.61.139.103/29	169.61.139.97
Dallas	us-south	dal12	169.61.150.223/29	169.61.150.217
Dallas	us-south	dal12	169.61.170.151/29	169.61.170.145
Dallas	us-south	dal12	169.61.178.111/29	169.61.178.105
Dallas	us-south	dal12	169.61.190.103/29	169.61.190.97
Dallas	us-south	dal12	169.63.1.79/29	169.63.1.73
Dallas	us-south	dal12	169.63.2.79/29	169.63.2.73
Dallas	us-south	dal12	169.63.18.71/29	169.63.18.65
Dallas	us-south	dal12	169.63.21.175/29	169.63.21.169
Dallas	us-south	dal12	169.63.21.191/29	169.63.21.185
Dallas	us-south	dal12	169.63.25.7/29	169.63.25.1
Dallas	us-south	dal12	169.63.26.15/29	169.63.26.9
Dallas	us-south	dal12	169.63.26.255/29	169.63.26.249
Dallas	us-south	dal12	169.63.27.7/29	169.63.27.1
Dallas	us-south	dal12	169.63.29.47/29	169.63.29.41
Dallas	us-south	dal12	169.63.31.7/29	169.63.31.1
Dallas	us-south	dal12	169.63.34.207/29	169.63.34.201
Dallas	us-south	dal12	169.63.47.87/29	169.63.47.81
Dallas	us-south	dal12	169.63.54.175/29	169.63.54.169
Dallas	us-south	dal12	169.63.55.143/29	169.63.55.137
Dallas	us-south	dal12	169.63.58.71/29	169.63.58.65
Dallas	us-south	dal12	169.63.58.79/29	169.63.58.73
Dallas	us-south	dal13	52.116.16.127/25	52.116.16.1
Dallas	us-south	dal13	52.116.24.63/27	52.116.24.33

Dallas	us-south	dal13	52.116.47.255/24	52.116.47.1
Dallas	us-south	dal13	52.116.60.255/24	52.116.60.1
Dallas	us-south	dal13	52.117.49.255/24	52.117.49.1
Dallas	us-south	dal13	52.117.198.191/26	52.117.198.129
Dallas	us-south	dal13	52.117.202.255/25	52.117.202.129
Dallas	us-south	dal13	52.117.205.255/24	52.117.205.1
Dallas	us-south	dal13	52.117.230.255/24	52.117.230.1
Dallas	us-south	dal13	52.117.235.255/24	52.117.235.1
Dallas	us-south	dal13	52.117.254.255/24	52.117.254.1
Dallas	us-south	dal13	52.118.127.255/24	52.118.127.1
Dallas	us-south	dal13	67.228.229.255/24	67.228.229.1
Dallas	us-south	dal13	67.228.234.255/24	67.228.234.1
Dallas	us-south	dal13	150.238.107.255/24	150.238.107.1
Dallas	us-south	dal13	150.238.113.255/24	150.238.113.1
Dallas	us-south	dal13	169.48.76.63/27	169.48.76.33
Dallas	us-south	dal13	169.59.14.255/24	169.59.14.1
Dallas	us-south	dal13	169.59.16.255/24	169.59.16.1
Dallas	us-south	dal13	169.59.22.255/24	169.59.22.1
Dallas	us-south	dal13	169.62.218.223/28	169.62.218.209
Dallas	us-south	dal13	169.62.237.63/26	169.62.237.1
Dallas	us-south	dal13	52.116.17.159/29	52.116.17.153
Dallas	us-south	dal13	52.116.19.183/29	52.116.19.177
Dallas	us-south	dal13	52.116.19.191/29	52.116.19.185
Dallas	us-south	dal13	52.116.25.247/29	52.116.25.241
Dallas	us-south	dal13	52.116.32.39/29	52.116.32.33
Dallas	us-south	dal13	52.116.54.239/29	52.116.54.233
Dallas	us-south	dal13	52.117.22.247/29	52.117.22.241
Dallas	us-south	dal13	52.117.23.143/29	52.117.23.137

Dallas	us-south	dal13	52.117.31.119/29	52.117.31.113
Dallas	us-south	dal13	52.117.35.231/29	52.117.35.225
Dallas	us-south	dal13	52.117.55.95/29	52.117.55.89
Dallas	us-south	dal13	52.117.55.119/29	52.117.55.113
Dallas	us-south	dal13	52.117.62.127/29	52.117.62.121
Dallas	us-south	dal13	52.117.234.183/29	52.117.234.177
Dallas	us-south	dal13	52.117.252.207/29	52.117.252.201
Dallas	us-south	dal13	150.238.100.247/29	150.238.100.241
Dallas	us-south	dal13	169.48.71.167/29	169.48.71.161
Dallas	us-south	dal13	169.48.71.175/29	169.48.71.169
Dallas	us-south	dal13	169.48.78.95/29	169.48.78.89
Dallas	us-south	dal13	169.48.96.71/29	169.48.96.65
Dallas	us-south	dal13	169.48.98.199/29	169.48.98.193
Dallas	us-south	dal13	169.48.98.231/29	169.48.98.225
Dallas	us-south	dal13	169.48.99.119/29	169.48.99.113
Dallas	us-south	dal13	169.48.107.79/29	169.48.107.73
Dallas	us-south	dal13	169.48.114.247/29	169.48.114.241
Dallas	us-south	dal13	169.48.123.151/29	169.48.123.145
Dallas	us-south	dal13	169.59.10.199/29	169.59.10.193
Dallas	us-south	dal13	169.60.131.167/29	169.60.131.161
Dallas	us-south	dal13	169.60.137.87/29	169.60.137.81
Dallas	us-south	dal13	169.60.138.223/29	169.60.138.217
Dallas	us-south	dal13	169.60.150.239/29	169.60.150.233
Dallas	us-south	dal13	169.60.159.183/29	169.60.159.177
Dallas	us-south	dal13	169.60.164.31/29	169.60.164.25
Dallas	us-south	dal13	169.60.184.63/29	169.60.184.57
Dallas	us-south	dal13	169.60.186.71/29	169.60.186.65
Dallas	us-south	dal13	169.61.23.199/29	169.61.23.193

Dallas	us-south	dal13	169.61.48.207/29	169.61.48.201
Dallas	us-south	dal13	169.61.48.239/29	169.61.48.233
Dallas	us-south	dal13	169.61.49.143/29	169.61.49.137
Dallas	us-south	dal13	169.61.56.223/29	169.61.56.217
Dallas	us-south	dal13	169.61.59.103/29	169.61.59.97
Dallas	us-south	dal13	169.61.60.39/29	169.61.60.33
Dallas	us-south	dal13	169.61.60.47/29	169.61.60.41
Dallas	us-south	dal13	169.62.134.103/29	169.62.134.97
Dallas	us-south	dal13	169.62.144.71/29	169.62.144.65
Dallas	us-south	dal13	169.62.151.47/29	169.62.151.41
Dallas	us-south	dal13	169.62.158.95/29	169.62.158.89
Dallas	us-south	dal13	169.62.184.207/29	169.62.184.201
Dallas	us-south	dal13	169.62.187.159/29	169.62.187.153
Dallas	us-south	dal13	169.62.193.183/29	169.62.193.177
Dallas	us-south	dal13	169.62.206.31/29	169.62.206.25
Dallas	us-south	dal13	169.62.238.151/29	169.62.238.145
Dallas	us-south	dal13	169.62.239.47/29	169.62.239.41
Dallas	us-south	dal13	169.62.240.223/29	169.62.240.217
Dallas	us-south	dal13	169.62.240.231/29	169.62.240.225
Dallas	us-south	dal13	174.36.70.143/29	174.36.70.137

us-south Public Subnets

Private Subnets

Location	Region	Data center	Subnet	First IP
Dallas	us-south	dal10	10.5.5.255/24	10.5.5.1
Dallas	us-south	dal10	10.5.124.255/24	10.5.124.1
Dallas	us-south	dal10	10.5.204.255/24	10.5.204.1
Dallas	us-south	dal10	10.23.68.255/24	10.23.68.1
Dallas	us-south	dal10	10.38.177.255/24	10.38.177.1
Dallas	us-south	dal10	10.94.35.255/24	10.94.35.1

Dallas	us-south	dal10	10.95.108.191/26	10.95.108.129
Dallas	us-south	dal10	10.95.123.255/24	10.95.123.1
Dallas	us-south	dal10	10.95.170.255/25	10.95.170.129
Dallas	us-south	dal10	10.95.191.255/24	10.95.191.1
Dallas	us-south	dal10	10.95.225.255/24	10.95.225.1
Dallas	us-south	dal10	10.171.25.255/24	10.171.25.1
Dallas	us-south	dal10	10.171.186.255/24	10.171.186.1
Dallas	us-south	dal10	10.171.209.191/26	10.171.209.129
Dallas	us-south	dal10	10.177.3.255/24	10.177.3.1
Dallas	us-south	dal10	10.177.109.127/25	10.177.109.1
Dallas	us-south	dal10	10.177.238.255/24	10.177.238.1
Dallas	us-south	dal10	10.177.253.255/26	10.177.253.193
Dallas	us-south	dal10	10.221.13.255/25	10.221.13.129
Dallas	us-south	dal10	10.221.69.255/24	10.221.69.1
Dallas	us-south	dal10	10.93.5.7/29	10.93.5.1
Dallas	us-south	dal10	10.93.16.247/29	10.93.16.241
Dallas	us-south	dal10	10.93.21.79/29	10.93.21.73
Dallas	us-south	dal10	10.93.24.55/29	10.93.24.49
Dallas	us-south	dal10	10.93.27.127/29	10.93.27.121
Dallas	us-south	dal10	10.93.28.127/29	10.93.28.121
Dallas	us-south	dal10	10.93.40.47/29	10.93.40.41
Dallas	us-south	dal10	10.93.51.55/29	10.93.51.49
Dallas	us-south	dal10	10.93.70.255/29	10.93.70.249
Dallas	us-south	dal10	10.93.71.15/29	10.93.71.9
Dallas	us-south	dal10	10.93.82.79/29	10.93.82.73
Dallas	us-south	dal10	10.93.92.79/29	10.93.92.73
Dallas	us-south	dal10	10.93.97.127/29	10.93.97.121
Dallas	us-south	dal10	10.93.107.231/29	10.93.107.225

Dallas	us-south	dal10	10.93.110.207/29	10.93.110.201
Dallas	us-south	dal10	10.93.116.23/29	10.93.116.17
Dallas	us-south	dal10	10.93.139.135/29	10.93.139.129
Dallas	us-south	dal10	10.93.184.63/29	10.93.184.57
Dallas	us-south	dal10	10.93.186.191/29	10.93.186.185
Dallas	us-south	dal10	10.93.203.63/29	10.93.203.57
Dallas	us-south	dal10	10.93.204.207/29	10.93.204.201
Dallas	us-south	dal10	10.93.224.47/29	10.93.224.41
Dallas	us-south	dal10	10.93.250.215/29	10.93.250.209
Dallas	us-south	dal10	10.171.1.55/29	10.171.1.49
Dallas	us-south	dal10	10.171.9.95/29	10.171.9.89
Dallas	us-south	dal10	10.171.13.119/29	10.171.13.113
Dallas	us-south	dal10	10.171.13.223/29	10.171.13.217
Dallas	us-south	dal10	10.171.20.215/29	10.171.20.209
Dallas	us-south	dal10	10.171.31.135/29	10.171.31.129
Dallas	us-south	dal10	10.171.31.151/29	10.171.31.145
Dallas	us-south	dal10	10.171.33.23/29	10.171.33.17
Dallas	us-south	dal10	10.171.49.63/29	10.171.49.57
Dallas	us-south	dal10	10.171.51.151/29	10.171.51.145
Dallas	us-south	dal10	10.171.53.47/29	10.171.53.41
Dallas	us-south	dal10	10.171.62.47/29	10.171.62.41
Dallas	us-south	dal10	10.171.68.23/29	10.171.68.17
Dallas	us-south	dal10	10.171.70.207/29	10.171.70.201
Dallas	us-south	dal10	10.171.70.255/29	10.171.70.249
Dallas	us-south	dal10	10.171.82.255/29	10.171.82.249
Dallas	us-south	dal10	10.171.108.255/29	10.171.108.249
Dallas	us-south	dal10	10.171.112.103/29	10.171.112.97
Dallas	us-south	dal10	10.171.115.191/29	10.171.115.185

Dallas	us-south	dal10	10.171.131.39/29	10.171.131.33
Dallas	us-south	dal10	10.171.131.47/29	10.171.131.41
Dallas	us-south	dal10	10.171.164.23/29	10.171.164.17
Dallas	us-south	dal10	10.171.166.79/29	10.171.166.73
Dallas	us-south	dal10	10.171.168.159/29	10.171.168.153
Dallas	us-south	dal10	10.171.172.7/29	10.171.172.1
Dallas	us-south	dal10	10.171.176.119/29	10.171.176.113
Dallas	us-south	dal10	10.171.178.31/29	10.171.178.25
Dallas	us-south	dal10	10.171.205.15/29	10.171.205.9
Dallas	us-south	dal10	10.171.207.183/29	10.171.207.177
Dallas	us-south	dal10	10.171.223.199/29	10.171.223.193
Dallas	us-south	dal10	10.171.233.47/29	10.171.233.41
Dallas	us-south	dal10	10.171.234.175/29	10.171.234.169
Dallas	us-south	dal10	10.171.238.239/29	10.171.238.233
Dallas	us-south	dal10	10.171.241.215/29	10.171.241.209
Dallas	us-south	dal10	10.171.253.39/29	10.171.253.33
Dallas	us-south	dal10	10.176.13.71/29	10.176.13.65
Dallas	us-south	dal10	10.176.18.23/29	10.176.18.17
Dallas	us-south	dal10	10.176.18.31/29	10.176.18.25
Dallas	us-south	dal10	10.176.18.47/29	10.176.18.41
Dallas	us-south	dal10	10.176.25.15/29	10.176.25.9
Dallas	us-south	dal10	10.176.31.71/29	10.176.31.65
Dallas	us-south	dal10	10.176.33.111/29	10.176.33.105
Dallas	us-south	dal10	10.176.35.159/29	10.176.35.153
Dallas	us-south	dal10	10.176.39.87/29	10.176.39.81
Dallas	us-south	dal10	10.176.61.39/29	10.176.61.33
Dallas	us-south	dal10	10.176.69.191/29	10.176.69.185
Dallas	us-south	dal10	10.176.120.207/29	10.176.120.201

Dallas	us-south	dal10	10.176.136.55/29	10.176.136.49
Dallas	us-south	dal10	10.176.158.87/29	10.176.158.81
Dallas	us-south	dal10	10.176.202.175/29	10.176.202.169
Dallas	us-south	dal10	10.176.205.239/29	10.176.205.233
Dallas	us-south	dal10	10.176.212.79/29	10.176.212.73
Dallas	us-south	dal10	10.176.222.223/29	10.176.222.217
Dallas	us-south	dal10	10.176.227.167/29	10.176.227.161
Dallas	us-south	dal10	10.176.248.55/29	10.176.248.49
Dallas	us-south	dal10	10.177.46.199/29	10.177.46.193
Dallas	us-south	dal10	10.177.65.15/29	10.177.65.9
Dallas	us-south	dal10	10.177.143.63/29	10.177.143.57
Dallas	us-south	dal10	10.177.144.191/29	10.177.144.185
Dallas	us-south	dal10	10.177.146.239/29	10.177.146.233
Dallas	us-south	dal10	10.177.165.119/29	10.177.165.113
Dallas	us-south	dal10	10.177.165.143/29	10.177.165.137
Dallas	us-south	dal10	10.177.194.15/29	10.177.194.9
Dallas	us-south	dal10	10.177.200.23/29	10.177.200.17
Dallas	us-south	dal10	10.177.204.71/29	10.177.204.65
Dallas	us-south	dal10	10.177.212.79/29	10.177.212.73
Dallas	us-south	dal10	10.177.212.199/29	10.177.212.193
Dallas	us-south	dal10	10.177.255.207/29	10.177.255.201
Dallas	us-south	dal10	10.177.255.231/29	10.177.255.225
Dallas	us-south	dal12	10.48.97.255/24	10.48.97.1
Dallas	us-south	dal12	10.48.107.255/24	10.48.107.1
Dallas	us-south	dal12	10.48.129.255/24	10.48.129.1
Dallas	us-south	dal12	10.48.136.255/24	10.48.136.1
Dallas	us-south	dal12	10.48.152.255/24	10.48.152.1
Dallas	us-south	dal12	10.48.176.255/24	10.48.176.1

Dallas	us-south	dal12	10.74.184.191/26	10.74.184.129
Dallas	us-south	dal12	10.74.213.127/25	10.74.213.1
Dallas	us-south	dal12	10.74.239.255/24	10.74.239.1
Dallas	us-south	dal12	10.184.148.255/25	10.184.148.129
Dallas	us-south	dal12	10.184.210.255/24	10.184.210.1
Dallas	us-south	dal12	10.185.6.255/26	10.185.6.193
Dallas	us-south	dal12	10.185.146.255/24	10.185.146.1
Dallas	us-south	dal12	10.185.241.255/24	10.185.241.1
Dallas	us-south	dal12	10.241.47.255/24	10.241.47.1
Dallas	us-south	dal12	10.241.186.255/24	10.241.186.1
Dallas	us-south	dal12	10.241.187.255/24	10.241.187.1
Dallas	us-south	dal12	10.241.222.255/24	10.241.222.1
Dallas	us-south	dal12	10.48.47.167/29	10.48.47.161
Dallas	us-south	dal12	10.48.63.167/29	10.48.63.161
Dallas	us-south	dal12	10.48.71.71/29	10.48.71.65
Dallas	us-south	dal12	10.74.42.255/29	10.74.42.249
Dallas	us-south	dal12	10.74.44.191/29	10.74.44.185
Dallas	us-south	dal12	10.74.53.159/29	10.74.53.153
Dallas	us-south	dal12	10.74.57.255/29	10.74.57.249
Dallas	us-south	dal12	10.74.64.191/29	10.74.64.185
Dallas	us-south	dal12	10.74.86.39/29	10.74.86.33
Dallas	us-south	dal12	10.74.87.231/29	10.74.87.225
Dallas	us-south	dal12	10.74.87.239/29	10.74.87.233
Dallas	us-south	dal12	10.74.87.247/29	10.74.87.241
Dallas	us-south	dal12	10.74.94.31/29	10.74.94.25
Dallas	us-south	dal12	10.74.103.47/29	10.74.103.41
Dallas	us-south	dal12	10.74.104.167/29	10.74.104.161
Dallas	us-south	dal12	10.74.123.15/29	10.74.123.9

Dallas	us-south	dal12	10.74.135.15/29	10.74.135.9
Dallas	us-south	dal12	10.74.153.135/29	10.74.153.129
Dallas	us-south	dal12	10.74.168.151/29	10.74.168.145
Dallas	us-south	dal12	10.74.171.255/29	10.74.171.249
Dallas	us-south	dal12	10.74.178.207/29	10.74.178.201
Dallas	us-south	dal12	10.74.181.71/29	10.74.181.65
Dallas	us-south	dal12	10.74.181.87/29	10.74.181.81
Dallas	us-south	dal12	10.74.189.87/29	10.74.189.81
Dallas	us-south	dal12	10.74.196.247/29	10.74.196.241
Dallas	us-south	dal12	10.74.203.151/29	10.74.203.145
Dallas	us-south	dal12	10.74.204.247/29	10.74.204.241
Dallas	us-south	dal12	10.74.205.31/29	10.74.205.25
Dallas	us-south	dal12	10.74.224.215/29	10.74.224.209
Dallas	us-south	dal12	10.74.226.111/29	10.74.226.105
Dallas	us-south	dal12	10.74.230.223/29	10.74.230.217
Dallas	us-south	dal12	10.74.234.255/29	10.74.234.249
Dallas	us-south	dal12	10.74.237.215/29	10.74.237.209
Dallas	us-south	dal12	10.184.5.71/29	10.184.5.65
Dallas	us-south	dal12	10.184.12.183/29	10.184.12.177
Dallas	us-south	dal12	10.184.16.191/29	10.184.16.185
Dallas	us-south	dal12	10.184.41.63/29	10.184.41.57
Dallas	us-south	dal12	10.184.121.39/29	10.184.121.33
Dallas	us-south	dal12	10.184.129.199/29	10.184.129.193
Dallas	us-south	dal12	10.184.168.199/29	10.184.168.193
Dallas	us-south	dal12	10.184.180.15/29	10.184.180.9
Dallas	us-south	dal12	10.184.250.175/29	10.184.250.169
Dallas	us-south	dal12	10.185.5.63/29	10.185.5.57
Dallas	us-south	dal12	10.185.11.175/29	10.185.11.169

Dallas	us-south	dal12	10.185.35.103/29	10.185.35.97
Dallas	us-south	dal12	10.185.44.55/29	10.185.44.49
Dallas	us-south	dal12	10.185.89.215/29	10.185.89.209
Dallas	us-south	dal12	10.185.89.231/29	10.185.89.225
Dallas	us-south	dal12	10.185.177.239/29	10.185.177.233
Dallas	us-south	dal12	10.241.69.7/29	10.241.69.1
Dallas	us-south	dal12	10.241.111.79/29	10.241.111.73
Dallas	us-south	dal13	10.36.35.255/24	10.36.35.1
Dallas	us-south	dal13	10.36.46.255/24	10.36.46.1
Dallas	us-south	dal13	10.36.70.255/24	10.36.70.1
Dallas	us-south	dal13	10.36.143.255/24	10.36.143.1
Dallas	us-south	dal13	10.36.157.255/24	10.36.157.1
Dallas	us-south	dal13	10.186.201.255/24	10.186.201.1
Dallas	us-south	dal13	10.187.55.63/26	10.187.55.1
Dallas	us-south	dal13	10.208.7.255/24	10.208.7.1
Dallas	us-south	dal13	10.208.16.127/25	10.208.16.1
Dallas	us-south	dal13	10.208.206.255/24	10.208.206.1
Dallas	us-south	dal13	10.209.179.255/26	10.209.179.193
Dallas	us-south	dal13	10.220.14.255/25	10.220.14.129
Dallas	us-south	dal13	10.220.18.255/24	10.220.18.1
Dallas	us-south	dal13	10.220.20.255/24	10.220.20.1
Dallas	us-south	dal13	10.220.53.255/24	10.220.53.1
Dallas	us-south	dal13	10.220.91.255/24	10.220.91.1
Dallas	us-south	dal13	10.220.117.255/24	10.220.117.1
Dallas	us-south	dal13	10.220.119.255/24	10.220.119.1
Dallas	us-south	dal13	10.220.152.255/24	10.220.152.1
Dallas	us-south	dal13	10.36.53.7/29	10.36.53.1
Dallas	us-south	dal13	10.36.94.119/29	10.36.94.113

Dallas	us-south	dal13	10.36.126.95/29	10.36.126.89
Dallas	us-south	dal13	10.36.127.23/29	10.36.127.17
Dallas	us-south	dal13	10.73.22.15/29	10.73.22.9
Dallas	us-south	dal13	10.73.84.143/29	10.73.84.137
Dallas	us-south	dal13	10.73.84.255/29	10.73.84.249
Dallas	us-south	dal13	10.73.104.159/29	10.73.104.153
Dallas	us-south	dal13	10.73.136.167/29	10.73.136.161
Dallas	us-south	dal13	10.73.136.183/29	10.73.136.177
Dallas	us-south	dal13	10.73.136.191/29	10.73.136.185
Dallas	us-south	dal13	10.73.209.23/29	10.73.209.17
Dallas	us-south	dal13	10.73.209.39/29	10.73.209.33
Dallas	us-south	dal13	10.73.209.55/29	10.73.209.49
Dallas	us-south	dal13	10.186.70.135/29	10.186.70.129
Dallas	us-south	dal13	10.186.108.207/29	10.186.108.201
Dallas	us-south	dal13	10.186.109.135/29	10.186.109.129
Dallas	us-south	dal13	10.186.124.55/29	10.186.124.49
Dallas	us-south	dal13	10.186.237.119/29	10.186.237.113
Dallas	us-south	dal13	10.186.244.143/29	10.186.244.137
Dallas	us-south	dal13	10.187.3.55/29	10.187.3.49
Dallas	us-south	dal13	10.187.46.95/29	10.187.46.89
Dallas	us-south	dal13	10.187.61.103/29	10.187.61.97
Dallas	us-south	dal13	10.187.63.63/29	10.187.63.57
Dallas	us-south	dal13	10.187.64.55/29	10.187.64.49
Dallas	us-south	dal13	10.187.64.63/29	10.187.64.57
Dallas	us-south	dal13	10.187.75.247/29	10.187.75.241
Dallas	us-south	dal13	10.187.78.63/29	10.187.78.57
Dallas	us-south	dal13	10.187.92.175/29	10.187.92.169
Dallas	us-south	dal13	10.187.134.79/29	10.187.134.73

Dallas	us-south	dal13	10.187.134.95/29	10.187.134.89
Dallas	us-south	dal13	10.187.145.119/29	10.187.145.113
Dallas	us-south	dal13	10.187.158.199/29	10.187.158.193
Dallas	us-south	dal13	10.187.158.239/29	10.187.158.233
Dallas	us-south	dal13	10.208.40.175/29	10.208.40.169
Dallas	us-south	dal13	10.208.142.199/29	10.208.142.193
Dallas	us-south	dal13	10.209.0.47/29	10.209.0.41
Dallas	us-south	dal13	10.209.9.199/29	10.209.9.193
Dallas	us-south	dal13	10.209.17.39/29	10.209.17.33
Dallas	us-south	dal13	10.209.45.247/29	10.209.45.241
Dallas	us-south	dal13	10.209.53.167/29	10.209.53.161
Dallas	us-south	dal13	10.209.76.95/29	10.209.76.89
Dallas	us-south	dal13	10.209.86.151/29	10.209.86.145
Dallas	us-south	dal13	10.209.104.79/29	10.209.104.73
Dallas	us-south	dal13	10.209.200.23/29	10.209.200.17
Dallas	us-south	dal13	10.209.222.199/29	10.209.222.193
Dallas	us-south	dal13	10.209.222.247/29	10.209.222.241
Dallas	us-south	dal13	10.209.233.215/29	10.209.233.209
Dallas	us-south	dal13	10.209.236.207/29	10.209.236.201
Dallas	us-south	dal13	10.209.239.199/29	10.209.239.193
Dallas	us-south	dal13	10.209.239.207/29	10.209.239.201
Dallas	us-south	dal13	10.220.5.151/29	10.220.5.145
Dallas	us-south	dal13	10.220.30.23/29	10.220.30.17
Dallas	us-south	dal13	10.220.30.31/29	10.220.30.25
Dallas	us-south	dal13	10.220.162.167/29	10.220.162.161
Dallas	us-south	dal13	10.220.202.239/29	10.220.202.233

us-south Private Subnets

Service endpoints integration

All Cloud Databases deployments offer integration with IBM Cloud service endpoints to enable connections to your deployments from the public internet

and over the IBM Cloud private network.

Service endpoints are available in all IBM Cloud multizone regions and some single-campus multizone regions. Deployments in all other regions are able to use service endpoints.

Private endpoints

A deployment with a service endpoint on the private network gets an endpoint that is not accessible from the public internet. At provision, this is the default option for all deployments. All traffic is routed to hardware dedicated to Cloud Databases deployments and remains on the IBM Cloud private network. All traffic to and from this endpoint is free and unmetered on the condition that the traffic remains in IBM Cloud. After your environment has access to the IBM Cloud private network, an internet connection is not required to connect to your deployment.

For more information, see Secure access to services using service endpoints.

> ⚠ **Important:** Deployments with private endpoints are reachable from any account within the private network and access to each instance requires authentication. To restrict this access to specific IP addresses, or ranges of IP addresses, configure Context-based restrictions.

Public endpoints

Public endpoints provide a connection to your deployment on the public network. Your environment needs to have internet access to connect to a deployment.

> ⚠ **Important:** For enhanced security, it is recommended that users connect to their Cloud Databases deployments using private endpoints instead of public endpoints.

Enabling service endpoints

To use connections over the public internet, you do not have to enable service endpoints on your IBM Cloud account. To enable private networking on your deployments, follow the instructions at Enabling VRF and service endpoints.

Currently, enabling virtual routing and forwarding (VRF) on your account in classic is a manual step that is handled by support ticket. VRF is automatically enabled for VPC. After you complete the request, check on the status of the ticket by going to your Support page on IBM Cloud.

Provisioning with service endpoints through the UI

To configure your deployment's endpoints on provision, use the **Endpoints** field on the **Provisioning** page. Select from the following available options:

- Private network
- Public network
- Both public and private network

> ⚠ **Important:** A MongoDB deployment cannot support both public and private endpoints simultaneously. *This cannot be changed after provisioning*.

Provisioning with service endpoints through the CLI

Service endpoints are specified using a required flag when you provision through the CLI. Provisioning is handled by the Resource Controller. You can change the endpoints by passing the `--service-endpoints` flag with one of the following values: `public`, `private`, or `public-and-private`. It is recommended to use *private* endpoints.

```
$ ibmcloud resource service-instance-create <INSTANCE_NAME> <SERVICE_NAME> <SERVICE_PLAN_NAME> <LOCATION> <SERVICE_ENDPOINTS_TYPE>
<RESOURCE_GROUP> -p '{"members_host_flavor": "<host_flavor value>"}' --service-endpoints=<ENDPOINT>
```

> ☑ **Tip:** Cloud Databases deployments except Databases for MongoDB allow for both public and private networking to be enabled at the same time.

Provisioning with service endpoints through the API

Service endpoints are enabled through a required parameter when you provision through the API. Provisioning is handled by the Resource Controller. Pass the `service-endpoints` parameter with one of the following options: `public`, `private`, or `public-and-private`. It is recommended to use *private* endpoints.

```
$ curl -X POST    https://resource-controller.cloud.ibm.com/v2/resource_instances   -H "Authorization: Bearer <TOKEN>"   -H 'Content-Type: application/json'     -d '{
   "name": "<INSTANCE_NAME",
```

```
  "location": "<LOCATION>",
  "resource_group": "RESOURCE_GROUP_ID",
  "resource_plan_id": "<SERVICE_PLAN_NAME>"
  "parameters": {
     "service-endpoints": "private"
 }
}'
```

Tip: Cloud Databases deployments except Databases for MongoDB allow for both public and private networking to be enabled at the same time.

Changing service endpoints

After you deploy, it is possible to change your public and private service endpoints configuration, except for Databases for MongoDB.

Changing service endpoints through the UI

In the **Settings** tab of your deployment's dashboard, go to the **Service endpoints** section. Toggle which types of connections are available to your deployment.

Changing the type of endpoints available on your deployment does not cause any downtime from a database perspective. However, if you disable an endpoint that is being used by you or your applications, those connections are dropped.

Changing service endpoints through the CLI

Use the ibmcloud resource service-instance-update command in the CLI, specifying the endpoint with the `--service-endpoints` flag.

```
$ ibmcloud resource service-instance-update <INSTANCE_NAME_OR_CRN> --service-endpoints <ENDPOINT-TYPE>
```

Changing the type of endpoints available on your deployment does not cause any downtime from a database perspective. However, if you disable an endpoint that is being used by you or your applications, those connections are dropped.

Changing service endpoints through the API

Use the Resource Controller API, with a `PATCH` request to the /resource_instances/{id} endpoint.

Changing the type of endpoints available on your deployment does not cause any downtime from a database perspective. However, if you disable an endpoint that is being used by you or your applications, those connections are dropped.

Credentials for private endpoints

Use either public or private connection strings with any set of credentials that you make on your deployment. By default, the connection strings for a set of credentials are filled with strings for connecting over a public endpoint. If you are using private endpoints, specify connection strings that contain the private endpoint to be generated instead.

When you create credentials in the *Service credentials* UI, use either the `{ "service-endpoints": "public" }` or the `{ "service-endpoints": "private" }` parameter to specify which endpoint gets filled into the connection strings. For the steps to follow to create credentials, see the topic *Managing users and roles* in the documentation for your chosen service.

In the API, use the /deployments/{id}/users/{userid}/connections/{endpoint_type} to retrieve connection strings for both public or private endpoints.

If you have only private endpoints on your deployments, then all new credentials have private endpoints in the connection strings.

Connecting through private endpoints

Cloud Databases offers both private and public cloud service endpoints. To run your application or access the endpoint from a browser that is not on the private network, take the following additional steps:

- Ensure your Cloud IaaS or SL account is enabled for private endpoints.
- Create a virtual machine (VSI) that runs Linux.
- Configure a user account with SSH access.
- From your workstation, run `ssh -D 2345 user@vsi-host` to start an SSH session and open a SOCKS proxy on port `2345` that forwards all traffic through the VSI.
- Configure your browser or application to use a SOCKS5 proxy on `localhost:2345`.
- Run your application or open the preferred private endpoint in your browser (for example, a management UI).

Using virtual private endpoints

For more information, see Virtual private endpoints.

Virtual Private Endpoints

> **Note:** This document covers all the Cloud Databases: Databases for PostgreSQL, Databases for MongoDB, Databases for Redis, Databases for Elasticsearch, IBM Cloud® Databases for MySQL, and Messages for RabbitMQ.

IBM Cloud® Virtual Private Endpoint (VPE) provides connection points to IBM services on the IBM private network from your VPC network.

Using Virtual Private Endpoints

> **Note:** Virtual Private Endpoints (VPEs) are generally available in all regions.

Before you begin

- Log in to the IBM Cloud console.
- You need to have a Cloud Databases deployment. You can provision one from the IBM Cloud catalog. Give your deployment a memorable name that appears in your account's Resource List.

Setting up your VPE

1. Create an IBM Cloud® Virtual Private Cloud. Follow the getting started instructions.
2. Make sure that your VPC has at least one virtual server instance (VSI), and that the VPC can connect to the VSI. You can use the UI, CLI, and API to provision a VSI. Follow the getting started instructions.
3. Make sure your Cloud Databases deployment's private endpoint is enabled.
4. In the IBM Cloud console, click the menu icon and select -> VPC Infrastructure -> Network -> Virtual private endpoint gateways. Create a VPE for your Cloud Databases instances with these instructions.
5. After you create your VPE, it might take a few minutes for the new VPE and pDNS to complete the process and begin working for your VPC. Completion is confirmed when you see an IP address set in the details view of the VPE.
6. To make sure pDNS is functioning for your VPE, `ssh` into your VSI and run the following:

```
$ nslookup <instance_hostname>
```

The following example shows the output from running `nslookup` on instance hostnames of `host-0.private.databases.appdomain.cloud`, `host-1.private.databases.appdomain.cloud`, and `host-2.private.databases.appdomain.cloud`:

```
$ root@test-vpc-vsi:~# nslookup host-0.private.databases.appdomain.cloud
Server:  127.0.0.53
Address: 127.0.0.53#53
Non-authoritative answer:
Name: host-0.private.databases.appdomain.cloud
Address: 10.240.64.6
```

```
$ root@test-vpc-vsi:~# nslookup host-1.private.databases.appdomain.cloud
Server:  127.0.0.53
Address: 127.0.0.53#53
Non-authoritative answer:
Name: host-1.private.databases.appdomain.cloud
Address: 10.240.64.6
```

```
$ root@test-vpc-vsi:~# nslookup host-2.private.databases.appdomain.cloud
Server:  127.0.0.53
Address: 127.0.0.53#53
Non-authoritative answer:
Name: host-2.private.databases.appdomain.cloud
Address: 10.240.64.6    < ---- your VPE IP address
```

7. You can now use your Cloud Databases instance in the virtual server instance (VSI). See Connecting an external application to choose the appropriate service documentation for viewing example commands.

   For example, if you have a Databases for MongoDB instance, refer to the documentation topic Connecting with the MongoDB Shell and use a command like the following:

```
$ mongosh -u $USERNAME -p $PASSWORD --tls --tlsCAFile /root/ c--authenticationDatabase admin --host replset host-0.private.databaseappdomain.cloud:30066,host-1.private.databases.appdomain.cloud:30066,host-private.databases.appdomain.cloud:30066
```

VPE discoverability

Following the previous steps results in a database instance with private endpoints that is reachable with the Virtual Private Endpoints from your VPC network.

⚠ **Important:** Database instances with private endpoints are reachable from any account within the private network and access to each instance requires authentication. To restrict this access to specific IP addresses, or ranges of IP addresses, configure Context-based restrictions or allowlisting.

⚠ **Important:** A MongoDB deployment cannot support both public and private endpoints simultaneously. *This cannot be changed after provisioning*.

Tip: For more information, see Secure access to services by using service endpoints.

More resources

- Planning for virtual private endpoint gateways.
- Creating an endpoint gateway.
- For further assistance, see the FAQs for virtual private endpoints, and the **Troubleshooting VPE gateways** documentation that includes how to fix communications issues.

Identity and Access Management integration

Access to IBM Cloud® Databases service instances for users in your account is controlled by IBM Cloud Identity and Access Management (IAM).

Note: This document covers the integration of IAM with Cloud Databases: Databases for PostgreSQL, Databases for MongoDB, Databases for Redis, Databases for Elasticsearch, IBM Cloud® Databases for MySQL, and Messages for RabbitMQ.

IAM is only integrated with high-level service access, which governs privileges and operations available in the Cloud Databases API and the Cloud Databases CLI plug-in. It does not govern database-level users and privileges. Database access is governed by the standard access controls provided by the database. IAM does not control database users.

For more information about assigning user roles in IBM Cloud, see Managing IAM access.

The following table provides a general overview of actions that are mapped to service management roles. Service management roles enable users to perform tasks on service resources at the service level. For example, assign user access for the service, create or delete service IDs, create instances, and bind instances to applications.

Service management role	Description of actions	Example actions
Viewer	As a viewer, you can view database instances but you can't make configuration changes.	View service overview and view alerts.
Operator	As an operator, you can view database instances and make configuration changes that include managing database credentials.	Scale a deployment and change a deployment's password.
Editor	As an editor, you can perform all platform actions (including making configuration changes and managing credentials) except for managing the account and assigning access policies.	Scale a deployment and change a deployment's password.

Administrator	As an administrator, you can perform all platform actions, including assigning access policies to other users.	Scale a deployment, change a deployment's password, and assign access policies.

IAM user roles and actions

Actions for Cloud Databases API

Access to certain API endpoints and requests is governed by role. The following lists the access policy for each role for IBM Cloud® Databases.

Viewer

The allowed actions for the Viewer role.

```
$ GET /v5/ibm/deployables
Read Deployables
---
GET /v5/ibm/regions
Read Discover available regions
---
GET /v5/ibm/tasks/:task_id
Read a Task
---
GET /v5/ibm/backups/:backup_id
Read a Backup
---
GET /v5/ibm/deployments/:deployment_id
Read a Deployment
---
GET /v5/ibm/deployables/:deployable_id/groups
Read deployable group
---
GET /v5/ibm/deployments/:deployment_id/point_in_time_recovery_data
Read all deployment point-in-time-recovery data
---
GET /v5/ibm/deployments/:deployment_id/tasks
Read all deployment tasks
---
GET /v5/ibm/deployments/:deployment_id/backups
Read all deployment backups
---
GET /v5/ibm/deployments/:deployment_id/remotes
Read all deployment remotes
---
GET /v5/ibm/deployables/:deployable_id/groups
Read all deployment groups
---
GET /v5/ibm/deployments/:deployment_id/configuration/schema
Read deployment configuration schema
---
GET /v5/ibm/deployments/:deployment_id/users/:user_type/:user_id/connections/:endpoint_type
Read deployment user connections
---
POST /v5/ibm/deployments/:deployment_id/users/:user_type/:user_id/connections/:endpoint_type
Create deployment user connections
---
GET /v5/ibm/deployments/:deployment_id/allowlists/ip_addresses
Read Allowlisted IP Addresses
```

Operator and Editor

The Operator and Editor roles are functionally the same for Cloud Databases. This list contains allowed actions for the Operator and the Editor roles.

```
$ GET /v5/ibm/deployables
Read Deployables
---
GET /v5/ibm/regions
Read Discover available regions
---
```

GET /v5/ibm/tasks/:task_id
Read a Task

GET /v5/ibm/backups/:backup_id
Read a Backup

GET /v5/ibm/deployments/:deployment_id
Read a Deployment

GET /v5/ibm/deployables/:deployable_id/groups
Read deployable group

GET /v5/ibm/deployments/:deployment_id/point_in_time_recovery_data
Read all deployment point-in-time-recovery data

GET /v5/ibm/deployments/:deployment_id/tasks
Read all deployment tasks

GET /v5/ibm/deployments/:deployment_id/backups
Read all deployment backups

POST /v5/ibm/deployments/:deployment_id/backups
Create an on-demand backup

GET /v5/ibm/deployments/:deployment_id/remotes
Read all deployment remotes

POST /v5/ibm/deployments/:deployment_id/remotes/resync
Resync remote replica

GET /v5/ibm/deployables/:deployable_id/groups
Read all deployment groups

PATCH /v5/ibm/deployments/:deployment_id/groups/:group_id
Set scaling values on a specified group.

DELETE /v5/ibm/deployments/:deployment_id/management/database_connections
Closes all the connections on a deployment. Available for PostgreSQL ONLY.

PATCH /v5/ibm/deployments/:deployment_id/configuration
Update deployment configuration

GET /v5/ibm/deployments/:deployment_id/configuration/schema
Read deployment configuration schema

POST /v5/ibm/deployments/:deployment_id/users/:user_type
Create a user based on user type

DELETE /v5/ibm/deployments/:deployment_id/users/:user_type/:user_id
Remove a user based on user type

GET /v5/ibm/deployments/:deployment_id/users/:user_type/:user_id/connections/:endpoint_type
Read deployment user connections

POST /v5/ibm/deployments/:deployment_id/users/:user_type/:user_id/connections/:endpoint_type
Create deployment user connections

GET /v5/ibm/deployments/:deployment_id/allowlists/ip_addresses
Read Allowlisted IP Addresses

POST /v5/ibm/deployments/:deployment_id/allowlists/ip_addresses
Create an Allowlisted IP Addresses

DELETE /v5/ibm/deployments/:deployment_id/allowlists/ip_addresses/:ip_address_id
Remove an Allowlisted IP Addresses

PUT /v5/ibm/deployments/:deployment_id/allowlists/ip_addresses
Bulk allowlist IP addresses

POST /v5/ibm/deployments/:deployment_id/elasticsearch/file_syncs
Create elasticsearch file sync

Administrator

The allowed actions for the Administrator role.

```
$ GET /v5/ibm/deployables
Read Deployables
---
GET /v5/ibm/regions
Read Discover available regions
---
GET /v5/ibm/tasks/:task_id
Read a Task
---
GET /v5/ibm/backups/:backup_id
Read a Backup
---
GET /v5/ibm/deployments/:deployment_id
Read a Deployment
---
GET /v5/ibm/deployables/:deployable_id/groups
Read deployable group
---
GET /v5/ibm/deployments/:deployment_id/point_in_time_recovery_data
Read all deployment point-in-time-recovery data
---
GET /v5/ibm/deployments/:deployment_id/tasks
Read all deployment tasks
---
GET /v5/ibm/backups/:backup_id
Read all deployment backups
---
POST /v5/ibm/deployments/:deployment_id/backups
Create an on-demand backup
---
GET /v5/ibm/deployments/:deployment_id/backups
Read all deployment remotes
---
POST /v5/ibm/deployments/:deployment_id/remotes/resync
Resync remote replica
---
GET /v5/ibm/deployables/:deployable_id/groups
Read all deployment groups
---
PATCH /v5/ibm/deployments/:deployment_id/groups/:group_id
Read deployment group
---
DELETE /v5/ibm/deployments/:deployment_id/management/database_connections
Kill all database connections
---
PATCH /v5/ibm/deployments/:deployment_id/configuration
Update deployment configuration
---
GET /v5/ibm/deployments/:deployment_id/configuration/schema
Read deployment configuration schema
---
POST /v5/ibm/deployments/:deployment_id/users/:user_type
Create a user based on user type
---
PATCH /v5/ibm/deployments/:deployment_id/users/:user_type/:user_id
Update a DeploymentUser
---
DELETE /v5/ibm/deployments/:deployment_id/users/:user_type/:user_id
Remove a user based on user type
---
GET /v5/ibm/deployments/:deployment_id/users/:user_type/:user_id/connections/:endpoint_type
Read deployment user connections
---
POST /v5/ibm/deployments/:deployment_id/users/:user_type/:user_id/connections/:endpoint_type
Create deployment user connections
---
GET /v5/ibm/deployments/:deployment_id/allowlists/ip_addresses
```

```
Read Allowlisted IP Addresses
---
POST /v5/ibm/deployments/:deployment_id/allowlists/ip_addresses
Create an Allowlisted IP Addresses
---
DELETE /v5/ibm/deployments/:deployment_id/allowlists/ip_addresses/:ip_address_id
Remove an Allowlisted IP Addresses
---
PUT /v5/ibm/deployments/:deployment_id/allowlists/ip_addresses
Bulk allowlist IP addresses
---
POST /v5/ibm/deployments/:deployment_id/elasticsearch/file_syncs
Create elasticsearch file sync
```

Key Protect integration

The data that you store in IBM Cloud® Databases is encrypted by default by using randomly generated keys. To control the encryption keys, you can Bring Your Own Key (BYOK) through IBM Key Protect and use one of your own keys to encrypt your databases and backups.

Note: This document covers the integration of Key Protect with Cloud Databases, which includes Databases for PostgreSQL, Databases for MongoDB, Databases for Redis, Databases for Elasticsearch, IBM Cloud® Databases for MySQL, and Messages for RabbitMQ.

To get started, you need Key Protect provisioned on your IBM Cloud account.

Creating or adding a key in Key Protect

Go to your instance of Key Protect and generate or enter a key.

Granting service authorization in the UI

Authorize Key Protect for use with Cloud Databases deployments:

1. Open your IBM Cloud dashboard.
2. From the menu bar, click **Manage** -> **Access (IAM)**.
3. In the side navigation, click **Authorizations**.
4. Click **Create**.
5. In the **Source service** menu, select the service of the deployment. For example, **Databases for PostgreSQL** or **Messages for RabbitMQ**
6. In the **Source service resources** menu, select **All resources**.
7. In the **Target service** menu, select **Key Protect**.
8. Select or retain the default value **Account** as the resource group for the **Target Service**
9. In the Target service **Instance ID** menu, select the service instances to authorize.
10. Enable the **Reader** role.
11. To use "Bring your own key" (BYOK) for backups, Select the **Enable authorizations to be delegated** box in the **Authorize dependent services** section.
12. Click **Authorize**.

If the service authorization is not present before provisioning your deployment with a key, the provision fails.

Using the Key Protect key

After you grant your Cloud Databases deployments permission to use your keys, you supply the key name or CRN when you provision a deployment. The deployment uses your encryption key to encrypt your data.

Using the Key Protect key in the UI

If provisioning from the catalog page, select the Key Protect instance and key from the dropdown menus.

Using the Key Protect key in the CLI

In the CLI, use the `disk_encryption_key_crn` parameter in the parameters JSON object.

```
$ ibmcloud resource service-instance-create <INSTANCE_NAME> <SERVICE-NAME> standard us-south \
```

```
-p \ '{
  "disk_encryption_key_crn": "crn:v1:<...>:key:<id>"
}'
```

Tip: The Key Protect key needs to be identified by its full CRN, not just its ID. A key protect CRN is in the format `crn:v1:<...>:key:<id>`.

Using the Key Protect key in the API

In the API, use the `disk_encryption_key` parameter in the body of the request.

```
$ curl -X POST \
  https://resource-controller.cloud.ibm.com/v2/resource_instances \
  -H 'Authorization: Bearer <>' \
  -H 'Content-Type: application/json' \
    -d '{
    "name": "my-instance",
    "target": "blue-us-south",
    "resource_group": "5g9f447903254bb58972a2f3f5a4c711",
    "resource_plan_id": "databases-for-x-standard",
    "disk_encryption_key_crn": "crn:v1:<...>:key:<id>"
}'
```

Tip: The Key Protect key needs to be identified by its full CRN, not just its ID. A key protect CRN is in the format `crn:v1:<...>:key:<id>`.

Key rotation

Key Protect offers manual and automatic key rotation and key rotation is supported by Cloud Databases deployments. When you rotate a key, the process initiates a *syncing KMS state* task, and your deployment is reencrypted with the new key. The task is displayed on the *Tasks* page on your deployment's *Overview* and the associated Key Protect and Cloud Databases events are sent to Activity Tracker.

For more information, see Rotating manually or automatically.

Deleting the deployment

If you delete a deployment that is protected with a Key Protect key, the deployment remains registered against the key during the soft-deletion period (up to 9 days). To delete the key in the soft-deletion period, force delete the key. After the soft-deletion period, the key can be deleted without the force. To determine when you can delete the key, check the association between the key and your deployment.

Cryptoshredding

Important: Cryptoshredding is a destructive action. When the key is deleted, your data is unrecoverable.

Key Protect allows you to initiate a force delete of a key that is in use by IBM Cloud® services, including your Cloud Databases deployments. This action is called cryptoshredding. Deleting a key that is in use on your deployment locks the disks that contain your data and disables your deployment. You are still able to access the UI and some metadata such as security settings in the UI, CLI, and API but you are not able to access any of the databases or data that is contained within them. Key deletion is sent to the Activity Tracker Event Routing as `kms.secrets.delete`.

Bring your own key for backups

If you use Key Protect, when you provision a database you can also designate a key to encrypt the Cloud Object Storage disk that holds your deployment's backups.

Note: BYOK for backups is available only in US regions `us-south` and `us-east`, and `eu-de`.

Important: Only keys in the `us-south` and `eu-de` are durable to region failures. To ensure that your backups are available even if a region failure occurs, you must use a key from `us-south` or `eu-de`, regardless of your deployment's location.

Granting the delegation authorization

To enable your deployment to use the Key Protect key, you need to Enable authorization to be delegated when granting the service authorizations. If the

delegation authorization is not present before provisioning your deployment with a key, the provision fails.

Using the key at provision in the CLI

After the appropriate authorization and delegation is granted, you supply the key name or CRN when you provision a deployment.

In the CLI, use the `backup_encryption_key_crn` parameter in the parameters JSON object.

```
$ ibmcloud resource service-instance-create <INSTANCE_NAME> <SERVICE-NAME> standard us-south \
-p \ '{
  "backup_encryption_key_crn": "crn:v1:<...>:key:<id>"
}'
```

Using the key at provision in the API

In the API, use the `backup_encryption_key_crn` parameter in the body of the request.

```
$ curl -X POST \
  https://resource-controller.cloud.ibm.com/v2/resource_instances \
  -H 'Authorization: Bearer <>' \
  -H 'Content-Type: application/json' \
    -d '{
  "name": "my-instance",
  "target": "blue-us-south",
  "resource_group": "5g9f447903254bb58972a2f3f5a4c711",
  "resource_plan_id": "databases-for-x-standard",
  "backup_encryption_key_crn": "crn:v1:<...>:key:<id>"
}'
```

After you enable delegation and provisioned your deployment, two entries appear in your *Authorizations* in IAM. One is the entry for the deployment that lists its status as delegator. It is "User Created".

Role	Source	Target	Type
AuthorizationDelegator, Reader	`<cloud-databases>` Service	Key Protect Service	User defined

Example delegator Key Protect Authorization

And one for the Cloud Object Storage bucket for its backups, where the deployment is the initiator.

Role	Source	Target	Type
Reader	Cloud Object Storage service	Key Protect Service	Created by `<cloud-databases-crn>`

Example Key Protect Authorization for Cloud Object Storage from Cloud Databases

Removing keys

IAM/Key Protect does not stop you from removing the policy between the key and Cloud Object Storage (the second example), but doing so can make your backups unrestorable. To prevent this, if you delete the Cloud Object Storage policy that governs the ability of Cloud Databases to use the key for Cloud Object Storage, the policy is re-created to continue backing up your deployment.

Be careful when removing keys and authorizations. If you have multiple deployments that use the same keys, it is possible to inadvertently destroy backups to **all** of those deployments by revoking the delegation authorization. If possible, do not use the same key for multiple deployment's backups.

If you want to shred the backups, you can delete the key. Cloud Object Storage ensures that the storage is unreadable and unwriteable. However, any other deployments that use that same key for backups encounter subsequent backup failures.

If you do require that the same key to be used for multiple deployment's backups, removing keys and authorizations can have the following side effects.

- If you delete just the Cloud Object Storage authorization (as seen in Table 2), then not only is the deployment that is shown as the creator affected, but any deployments that also use the same key are also affected. Those deployments can encounter temporary backup failures until the policy is automatically re-created. There should be no lasting effects, except for missing backups.
- If you delete just Cloud Databases delegator authorization, which is created by you (as seen in Table 1), nothing immediately breaks because the second authorization is still in place. However, if the Cloud Object Storage authorization is ever removed, it cannot be re-created, and can lead to multiple deployments that use the same key losing the ability to back up.
- If you delete both the Cloud Object Storage authorization **AND** the Cloud Databases delegator authorization, all deployments that use the same key

will immediately not have the ability to back up and the correct authorizations will not be able to be re-created, effectively destroying the backups for all deployments that use that key.

⚠ **Important:** Use caution if you reuse keys.

Hyper Protect Crypto Services integration

The data that you store in IBM Cloud® Databases is encrypted by default by using randomly generated keys. If you need to control the encryption keys, you can Bring Your Own Key (BYOK) through Hyper Protect Crypto Services, and use one of your own keys to encrypt your databases. Take note that Hyper Protect Crypto Services for IBM Cloud® Databases backups is currently not supported for the majority of regions and not recommended to be used without careful considerations of the impact to disaster recovery.

Note: This document covers the integration of Hyper Protect Crypto Services (HPCS) with Cloud Databases, which includes Databases for Elasticsearch, Databases for MongoDB, Databases for PostgreSQL, Databases for Redis, IBM Cloud® Databases for MySQL, and Messages for RabbitMQ.

To get started, you need Hyper Protect Crypto Services provisioned on your IBM Cloud account.

Creating or adding a key in Hyper Protect Crypto Services

Navigate to your instance of Hyper Protect Crypto Services and generate or enter a key.

Granting service authorization

Authorize Hyper Protect Crypto Services for use with Cloud Databases deployments:

1. Open your IBM Cloud dashboard.
2. From the menu bar, click **Manage** > **Access (IAM)**.
3. In the side navigation, click **Authorizations**.
4. Click **Create**.
5. In the **Source service** menu, select the service of the deployment. For example, **Databases for PostgreSQL** or **Messages for RabbitMQ**
6. In the **Source service instance** menu, select **All instances**.
7. In the **Target service** menu, select **HPCS**.
8. Select or retain the default value `Account` as the resource group for the **Target Service**
9. In the Target service **Instance ID** menu, select the service instances to authorize.
10. Enable the **Reader** role.
11. Click **Authorize**.

If the service authorization is not present before provisioning your deployment with a key, the provision fails.

Using the HPCS key

After you grant your Cloud Databases deployments permission to use your keys, you supply the key name or CRN when you provision a deployment. The deployment uses your encryption key to encrypt your data.

If provisioning from the catalog page, select the HPCS instance and key from the drop-down menu.

In the CLI, use the `disk_encryption_key_crn` parameter in the parameter's JSON object.

```
ibmcloud resource service-instance-create <INSTANCE_NAME> <SERVICE-NAME> standard us-south \
-p \ '{
  "disk_encryption_key_crn": "crn:v1:<...>:key:<id>"
}'
```

In the API, use the `disk-encryption-key` parameter in the body of the request.

```
curl -X POST \
  https://resource-controller.cloud.ibm.com/v2/resource_instances \
  -H 'Authorization: Bearer <>' \
  -H 'Content-Type: application/json' \
    -d '{
```

```
  "name": "my-instance",
  "target": "blue-us-south",
  "resource_group": "5g9f447903254bb58972a2f3f5a4c711",
  "resource_plan_id": "databases-for-x-standard",
  "disk_encryption_key_crn": "crn:v1:<...>:key:<id>"
}'
```

Tip: If you provision a deployment through the CLI or API, the HPCS key must be identified by its full CRN, not just its ID. An HPCS CRN has the format `crn:v1:<...>:key:<id>`.

Using the HPCS key for backup encryption

Note: This feature is only supported in the eu-es and br-sao regions. Encrypting backups with HPCS in a single region renders the backups inaccessible, if availability of HPCS is disrupted in this region. Taking a backup and restoring from backups will fail for the period that HPCS is unavailable. Therefore, encrypting backups with HPCS is not recommended. Use IBM® Key Protect to encrypt backups.

Tip: If you encrypted the backup with HPCS, encrypt the disk also with HPCS.

After you grant your Cloud Databases deployments permission to use your keys, you supply the key name or CRN when you provision a deployment. The deployment uses your encryption key to encrypt your data.

If you provision from the Catalog, select the HPCS instance and key from the drop-down menu.

In the CLI, use the `backup_encryption_key_crn` parameter in the parameter's JSON object.

```
ibmcloud resource service-instance-create <INSTANCE_NAME> <SERVICE-NAME> standard eu-es \
-p \ '{
  "backup_encryption_key_crn": "crn:v1:<...>:key:<id>"
}'
```

In the API, use the `backup-encryption-key` parameter in the body of the request.

```
curl -X POST \
  https://resource-controller.cloud.ibm.com/v2/resource_instances \
  -H 'Authorization: Bearer <>' \
  -H 'Content-Type: application/json' \
    -d '{
    "name": "my-instance",
    "target": "blue-us-south",
    "resource_group": "5g9f447903254bb58972a2f3f5a4c711",
    "resource_plan_id": "databases-for-x-standard",
    "parameters": {
      "backup_encryption_key_crn": "crn:v1:<...>:key:<id>"
    }
}'
```

Tip: If you provision a deployment through the CLI or API, the HPCS key must be identified by its full CRN, not just its ID. An HPCS CRN has the format `crn:v1:<...>:key:<id>`.

Key rotation

HPCS offers manual and automatic key rotation and key rotation is supported by Cloud Databases deployments. When you rotate a key, the process initiates a *Syncing KMS state* task, and your deployment is reencrypted with the new key. The task is displayed on the *Tasks* panel on your deployment's *Overview* and the associated HPCS and Cloud Databases events are sent to Activity Tracker.

Deleting the deployment

If you delete a deployment that is protected with an HPCS key, the deployment remains registered against the key during the soft-deletion period (up to 9 days). If you need to delete the key in the soft-deletion period, you must force delete the key. After the soft-deletion period, the key can be deleted without the force. You can check the association between the key and your deployment to determine when you can delete the key.

Cryptoshredding

Important: Cryptoshredding is a destructive action. When the key is deleted, your data is unrecoverable.

Hyper Protect Crypto Services enables initiation of a force delete of a key that is in use by IBM Cloud® services, including your Cloud Databases deployments. This action is called cryptoshredding. Deleting a key that is in use on your deployment locks the disks that contain your data and disables your deployment. You are still able to access the UI and some metadata such as security settings in the UI, CLI, and API but you are not able to access any of the databases or data that is contained within them. Key deletion is sent to the Activity Tracker Event Routing as `hs-crypto.secrets.delete`.

Deleting your deployment and removing your data

When an IBM Cloud® Databases deployment is deleted, it is put in a soft-deleted state for 3 days after which it is removed. Soft-deleted deployments can be recovered by following the steps below.

Note: After the 3-day soft-deletion period has ended, or if a hard deletion is issued, deployments are no longer recoverable.

Deleting your deployment in the user interface

To delete your deployment from the Resource list section dashboard of the IBM Cloud® dashboard, select your deployment. Then, in the overflow menu click **Delete service** from the drop-down list.

Restoring a soft-deleted instance using the UI

Soft-deleted instances must be restored using the CLI.

Deleting your deployment using the CLI

By using the CLI, you can delete your existing IBM Cloud® Databases deployment with the `ibmcloud resource service-instance-delete` command:

```
$ ibmcloud resource service-instance-delete <INSTANCE_NAME_OR_CRN>
```

You can further hard-delete an instance by using `ibmcloud resource reclamations` to list soft-deleted instances, followed by `ibmcloud resource reclamation-delete <RECLAMATION ID>` to hard-delete it.

Important: Instances hard-deleted by `ibmcloud resource reclamation-delete` are unrecoverable and cannot be restored.

Restoring a soft-deleted instance using the CLI

You can use the following command to list the soft-deleted instances that are available for reclamation:

```
$ ibmcloud resource reclamations
```

Then, use the following command to recover them:

```
$ ibmcloud resource reclamation-restore <RECLAMATION ID>
```

Note: Restoring an IBM Cloud® Databases deployment from a soft-deleted state may take several hours.

Cryptoshredding keys

Key Protect provides cryptoshredding, which is deletion of a key that is in use by IBM Cloud® services, including your Cloud Databases deployments.

Important: Cryptoshredding is a destructive action. When the key is deleted, your data is unrecoverable even from a soft delete state.

Backups removal

Backups cannot be manually deleted or removed. However, if you delete your deployment, its backups are deleted automatically.

Databases for Elasticsearch tutorials

Configuring Kibana and Enterprise Search server with a Databases for Elasticsearch instance

This tutorial will guide you through the steps to configure a functional Enterprise Search server integrated with an IBM Cloud® Databases for Elasticsearch instance. Elasticsearch is a powerful and versatile search and analytics engine that helps you store, search, and analyze large volumes of data quickly and in near-real-time.

Databases for Elasticsearch is an Elasticsearch service that is offered by IBM Cloud that provides a managed and scalable solution for deploying and running Elasticsearch clusters.

Kibana complements Elasticsearch by offering a flexible visualization platform. It allows you to explore, visualize, and share insights from your data, enabling you to create custom dashboards and visualizations to better understand your information.

Enterprise Search extends the capabilities of Databases for Elasticsearch to provide a unified search experience across various data sources, including documents, emails, databases, and more.

By integrating Enterprise Search with your Databases for Elasticsearch instance, you gain a comprehensive search solution that uses the strengths of both platforms to efficiently discover insights from your data.

Kibana and Enterprise Search will be deployed on IBM Code Engine, a fully-managed serverless platform that can be used to host cloud native applications such as web apps.

Before you start

Before you begin, ensure you have the following:

- An IBM Cloud account
- The IBM Cloud CLI
- Terraform - to deploy infrastructure

Step 1: Obtain an API key to deploy infrastructure to your account

Follow these steps to create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

> **Note:** For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Step 2: Clone the project

```
$ git clone https://github.com/IBM/elasticsearch-kibana-enterprise-search.git
```

Step 3: Install your infrastructure

1. Navigate into the terraform folder of the cloned project.

   ```
   $ cd elasticsearch-kibana-codeengine/terraform
   ```

2. On your machine, create a document that is named `terraform.tfvars`, with the following fields:

   ```
   $ ibmcloud_api_key = "<your api key>"
   region = "<an ibm cloud region>" #e.g. eu-gb
   es_username = "admin"
   es_password = "<make up a password>" #Passwords have a 15 character minimum and must contain a number. Other acceptable characters are A-Z, a-z, 0-9, -, _
   es_version="<a supported major version>" # eg 8.12
   ```

   > **Important:** The `terraform.tfvars` document contains variables that you might want to keep secret.

3. Install the infrastructure with the following command:

   ```
   $ terraform init
   ```

```
terraform apply --auto-approve
```

Step 4: Visit your Kibana deployment

The previous step will output the URL of the Kibana deployment, for example something like:

```
$ kibana_endpoint = "https://kibana-app.1dqmr45rt678g05.eu-gb.codeengine.appdomain.cloud"
```

Log in at this URL with the username and password you supplied above.

Once logged in, you can configure Enterprise Search by visiting the following URL.

```
$ https://kibana-app.1dqmr45rt678g05.eu-gb.codeengine.appdomain.cloud/app/enterprise_search/app_search/engines
```

You can find more information on the many features of Enterprise search on the Elastic website.

The output of the previous step also contains the URL of the Elasticsearch deployment itself, which can be used to connect it to WatsonX Assistant or other applications.

Step 5: Wrapping up

Your Databases for Elasticsearch incurs charges, as does the Code Engine resources that host Kibana and Enterprise Search. After you finish this tutorial, you can remove all the infrastructure by going to the `terraform` directory of the project and using the command:

```
$ terraform destroy
```

Deploy Kibana using Code Engine and connect to your Databases for Elasticsearch instance

With this tutorial, deploy Kibana using Code Engine and connect to your Databases for Elasticsearch instance. Kibana is a web interface that allows you to visualise the data in Elasticsearch instances. Code Engine is a a fully managed, serverless platform that allows you to run workloads without worrying about deploying infrastructure. Elasticsearch is a NoSQL database with powerful search capabilities.

Databases for Elasticsearch does not include a managed Kibana service, but with this tutorial you can provision a Kibana instance and connect to your Databases for Elasticsearch instance within a few minutes and still using the managed service model of IBM Cloud.

NOTE: Code Engine is a paid-for service, so following this tutorial will incur charges.

Before you start

Before you begin, ensure you have the following:

- An IBM Cloud Account.
- Terraform - to deploy infrastructure
- A Databases for Elasticsearch instance

Step 1: Obtain an API key to deploy infrastructure to your account

Follow these steps to create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

> **Note:** For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Step 2: Clone the project

```
$ git clone https://github.com/IBM/elasticsearch-kibana-codeengine.git
```

Step 3: Upload and Analyze your Data

1. Navigate into the terraform folder of the cloned project.

```
$ cd elasticsearch-kibana-codeengine/terraform
```

2. On your machine, create a document that is named `terraform.tfvars`, with the following fields:

```
$  ibmcloud_api_key = "<your_api_key_from_step_1>"
 region = "<the IBM region where you will deploy the Code Engine application>"
 es_host = "<the hostname of your elasticsearch deployment>"
 es_port = "<the port number of your elasticsearch deployment>"
 es_username = "<the username of your elasticsearch deployment>"
 es_password = "<the password of your elasticsearch user>"
```

> ⚠ **Important:** The `terraform.tfvars` document contains variables that you might want to keep secret.

> **Note:** Kibana runs with `ELASTICSEARCH_SSL_VERIFICATIONMODE` set to `none`, so although the traffic between Kibana and Elasticsearch is encrypted, the Elasticsearch service is not verified against the CA certificate provided by IBM Databases for Elasticsearch credentials.

3. Install the infrastructure with the following command:

```
$ terraform init
terraform apply --auto-approve
```

Step 4: Visit your Kibana deployment

The previous step produces a URL, which is the public URL of your Kibana deployment. It looks something like: `https://kibana-app.1834dcfgrtygbg.eu-gb.codeengine.appdomain.cloud`.

Visit that URL in your web browser. You should see the Kibana login screen where you can log in with your credentials and have access to your Elasticsearch deployment!

Your Databases for Elasticsearch incurs charges. After you finish this tutorial, you can remove all the infrastructure by going to the `terraform` directory of the project and using the command:

```
$ terraform destroy
```

Elasticsearch data migration using snapshot and restore

Migrate data between two instances of Elasticsearch with the help of Object Storage. Using the snapshot and restore method, take snapshots from your source instance, store them in Object Storage, and then restore those snapshots from Object Storage into your target instance.

This tutorial uses snapshot and restore, IBM Cloud Object Storage and two instances of Databases for Elasticsearch. However, this process is applicable to any S3-compatible object storage solution and any deployment of Elasticsearch.

We've simplified the process by using Terraform and shell scripts. Simply follow the procedure outlined on this page, plugging in the necessary variables as you go.

> ⚠ **Important:** If you are using the snapshot and restore process to upgrade from Elasticsearch 7.9/7.10 to version 7.17, you must change your plan from Standard to Enterprise. Change your plan by clicking *Restore* on the Provisioning page on the backup you want to restore, then select 7.17 from the dropdown menu.

Getting Productive

Before you migrate your data, install Terraform to codify and deploy necessary infrastructure. You also need an IBM Cloud account.

Step 1: Obtain an IBM Cloud API key and clone the GitHub repository

Follow these steps to create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

> ⚠ **Important:** For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Next, clone the Elasticsearch Snapshot/Restore GitHub Repository to your local machine.

```
$ git clone https://github.com/IBM/elasticsearch-cos-snapshot-restore.git
```

After cloning this folder, navigate to the newly created project folder on your local machine.

Step 2: Install and run the Terraform script

The terraform folder contains files that create the necessary infrastructure to create and restore your snapshots:

- cos.tf
- elastic.tf
- main.tf
- variables.tf

cos.tf

cos.tf creates a Cloud Object Storage instance and a bucket. Update your resources for the `restoreCOSInstance`, `restoreBucket`, `resourceKey`. This script then outputs the `bucket_credentials` and `bucket_name`.

elastic.tf

elastic.tf creates a source, target, and necessary configurations. Input the variables for the `resource "ibm_database" "esSource"` and `resource "ibm_database" "esTarget"`. This script then outputs the necessary configuration variables.

main.tf

main.tf contains the main set of configuration for your module. For the `ibmcloud_api_key` variable, create or retrieve an IBM Cloud® API key. Then, specify the Resource Group and `ibm_resource_group` variable, which outputs the `resource_group_name`.

variables.tf

variables.tf contains the variable definitions `ibmcloud_api_key`, `region`, and `elastic_password`. Update these variables with your API key, preferred region, and your Elasticsearch password.

After setting up your resources, configurations, and variables, go ahead and run your Terraform script. Navigate to your terraform folder and install the infrastructure with the following command:

```
$ terraform init
terraform apply --auto-approve
```

The Terraform script outputs configuration data that is needed to run the application, so copy it into the root folder:

```
$ terraform output -json >../config.json
```

Step 3: Run the shell snapshot script

Run the *migrate.sh* file in the main project folder. This shell script uses the information that is provided by the `config.json` file to perform the necessary migration steps.

- Create different snapshot names by adding a timestamp.
- Get database and S3/COS parameters.
- Mount S3/COS bucket on source deployment.
- Mount S3/COS bucket on target deployment.
- Close all indexes on the target so the restore can be run without touching the `icd-auth index`, which is protected by Cloud Databases.

Run *migrate.sh* as many times as necessary to fully back up your data.

> **Note:** Snapshots are incremental, so the first snapshot takes longer than the rest.

Once your COS bucket has all the necessary snapshots, stop any writes to the source. Then, run *migrate.sh* one more time to take a final snapshot and restore it to the target. All of your data is now in the target. Point your applications to the target database and your upgrade is complete.

Retrieve and update user passwords

If you're restoring to Elasticsearch 7.17 as an update from an earlier version, existing user passwords will be invalidated and must be reset after the restore. For more information, see Retrieve and update user passwords.

Configuring the Index Lifecycle Management capabilities of Databases for Elasticsearch

Index Lifecycle Management (ILM) is a great feature of Elasticsearch. It allows you to proactively manage your indices to make efficient use of resources, both in terms of storage and search capabilities. For example, if an application needs to store 30 days of events, ILM can be used to create an index called “events”, which can be written to and queried easily, but in reality consists of thirty separate indices in the background. Older indices can be made “read only” and optionally optimized, and when they reach the age of 30 days, deleted.

Lifecycle rules can be defined, including:

- When creating a new index: by age, data volume, or document count.
- Whether to make older indices “read only”.
- Whether to change the shard count of older indices.
- Setting the priority of each index, which defines the order in which indices are restored on node reboots.
- If and when older data is deleted.

ILM is very flexible and feature-rich. For a full description of its capabilities, see the ILM overview.

In this tutorial, you will get familiar with ILM by creating a set of simple rules and then watching how they get implemented. Although the set up is relatively simple, you will need to let the rules take their course over a couple of days to see the full effect.

Note: Databases for Elasticsearch is a paid-for service, so following this tutorial will incur charges.

Before you start

Before you begin, ensure that you have the following:

- IBM Cloud account
- Terraform - To deploy infrastructure.

Obtain an API key to deploy infrastructure to your account

Follow these steps to create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

Note: For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Clone the project

To clone the project, run the following command:

```
$ git clone https://github.com/IBM/elasticsearch-index-lifecycle-management.git
```

Install the Elasticsearch cluster

1. Navigate into the Terraform folder of the cloned project.

   ```
   cd elasticsearch-index-lifecycle-management/terraform
   ```

2. Create a document that is named `terraform.tfvars`, with the following fields:

   ```
   ibmcloud_api_key = "<your_api_key_from_step_1>"
   region = "<your_region>"
   elastic_password = "<make-up-a-password>"
   ```

   Important: The `terraform.tfvars` document contains variables that you may want to keep secret, so it is excluded from the public Github repository.

3. Install the infrastructure with the following command:

   ```
   terraform init
   terraform apply --auto-approve
   ```

4. Finally, export the database access URL to your terminal environment (it will be required by subsequent steps).

```
terraform output --json
export ES="<the url value obtained from the output>"
```

Create an ILM process

Let’s assume that you have logs coming from your applications into an Elasticsearch instance. The logs are very important on day one because you are checking for anomalies. After day two, the logs become less useful but you still need them around for things like trying to spot trends. After three days, these logs are stale and you have no further use for them. So, create a three day lifecycle for your index:

- Day 1: Your data is in the Hot Tier, meaning it is readily available for search. It is your most recent, most searched data.
- Day 2: Your data is in the Warm Tier. After day 1, your data is rolled over to a “warm” state. In this tier, it is optimized for search rather than indexing. In this tier, you will force a merge to reduce the number of segments in the index’s shards, for more efficient searching.
- Day 3: Delete. After day 3 your data is no longer needed, so it gets deleted.

Create an Index Lifecycle policy

First, create an ILM policy that defines the appropriate phases and actions as described above.

```
$ curl -kX PUT -H 'Content-Type: application/json' -d'{"policy":{"phases":{"hot":{"actions":{"rollover":{"max_age":"1d"},"set_priority":{"priority":100},"forcemerge":{"max_num_segments":1},"shrink":{"number_of_shards":1},"readonly":{}},"min_age":"0ms"},"warm":{"min_age":"1d","actions":{"set_priority":{"priority":50}}},"delete":{"min_age":"3d","actions":{"delete":{}}}}}}' $ES/_ilm/policy/ilm-test-1
```

Create an index template

An index template defines how indices are going to be created. Because our lifecycle policy above moves data from hot to warm every day, a new index will be created each day. This template tells Elasticsearch what patterns to use to create these new indices: in this case all related indices will be called “logs-”, followed by an incrementing number. The template also has other settings, such as what lifecycle policy to use for the index. Let's use the one we created in the previous step.

Index templates are created in two steps. First, you create one or more “component” templates. These are reusable blocks that can be combined later to create multiple templates. In this very simple example you create only one component template.

```
$ curl -kX PUT -H 'Content-Type: application/json' -d'{"template":{"mappings":{"properties":{"@timestamp":{"type":"date"}}}}}' $ES/_component_template/component_template1
```

(This component template is basically empty except for a mapping to a timestamp field (all logs have a timestamp) but a real-world use case can contain complex mappings and other instructions).

Second, you create the index template itself, making use of your component template(s). This one tells Elasticsearch about the index pattern and other data, such as for example that every new index must have two shards and two replicas.

```
$ curl -kX PUT -H 'Content-Type: application/json' -d'{"index_patterns":["logs-*"],"template":{"settings":{"number_of_shards":2,"number_of_replicas":2,"index.lifecycle.name":"ilm-test-1","index.lifecycle.rollover_alias":"logs"}},"priority":500,"composed_of":["component_template1"],"version":3,"_meta":{"description":"my custom template"}}' $ES/_index_template/my_index_template
```

Create an index

The last step is to create an Elasticsearch index that will use your template (and therefore your lifecycle policy). By calling it “logs-000001” and aliasing it to the alias defined in the template, you ensure that the right template is used and that it is incremented numerically.

```
$ curl -kX PUT -H 'Content-Type: application/json' -d'{"aliases": {"logs": { "is_write_index": true } } }' $ES/logs-000001
```

Add documents to the index

Documents can be added without knowing the name of the current `logs` index, we simply write to `logs`.

```
$ curl -kX PUT -d’{document goes here}’ $ES/logs/_doc/mydocid
```

Query the index

Although Elasticsearch is storing data in multiple indices, it can still be queried as if it were one.

```
$ curl -X POST -d’{query goes here}’ $ES/logs/_search
```

Watch your index manage itself

Now you have to wait a bit. On day one you will be able to see an index called `logs-000001`.

```
$ curl -kX GET $ES/_cat/indices | grep logs-
```

But on day two you will see another index called `logs-000002` appear, if you run the above command again. And on day three, `logs-000001` should disappear (because it was deleted) but you should see `logs-000003` appear. You can always search the entire contents of your logs at any point by using the alias `logs`. Your indices are managing themselves!

Tear down your infrastructure

Your Databases for Elasticsearch incurs charges. After you finish this tutorial, you can remove all the infrastructure by going to the `terraform` directory of the project and using the command:

```
$ terraform destroy
```

Next Steps

ILM is very feature-rich and in this tutorial you have only explored the basics of it. ILM can help you manage your data in an efficient way. For more information, see ILM: Manage the index lifecycle.

Elasticsearch machine learning tutorials

Use Elasticsearch vector search capabilities

Objectives

In this tutorial, you deploy an instance of Databases for Elasticsearch and use it to store vector representations of images that you are then able to search to find similarities with new, unseen, images.

These vector representations, known as embeddings, are created using Machine learning algorithms. Machine learning is a branch of artificial intelligence (AI) and computer science that focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. By using statistical methods, algorithms are trained to make classifications or predictions, and to uncover key insights in data mining projects.

These learning algorithms are known as “models”. In this tutorial we make use of one such model, OpenAI's CLIP. CLIP (Contrastive Language-Image Pre-Training) is a neural network trained on a variety of (image, text) pairs.

Traditionally, finding similarities between images, something that is relatively straightforward to the human eye, has been difficult for a computer to do. Machine Learning has transformed this field of search.

See the other tutorials in this Elasticsearch machine learning series:

- Use ELSER, Elastic's Natural Language Processing model
- Use machine learning models with Elasticsearch to tag content

⚠ **Important:** Databases for Elasticsearch is a paid-for service, so following this tutorial will incur charges.

Getting productive

To begin the provisioning process, install some must-have productivity tools:

- You need to have an IBM Cloud account.
- Terraform - To codify and provision infrastructure.
- Python

You also need a dataset of images that provide the corpus for doing similarity search. For example, you might have a dataset of car images or of bird images. You typically need thousands of these images. We cannot provide one here because of copyright restrictions on images.

Note: In this tutorial you won't upload the images themselves to your database. The vector representations will be calculated locally and only those will be uploaded. In a real-life scenario you would probably have the images stored somewhere (like an Object Storage bucket) and a reference to the image location would be stored alongside the vector representation, for retrieval.

Step 1: Obtain an API key

Create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

⚠ **Important:** For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Step 2: Clone the project

Clone the project from the GitHub repository.

```
$ git clone https://github.com/IBM/elasticsearch-ml-vector-search-tutorial.git
```

Step 3: Install the Elasticsearch cluster

1. Navigate into the terraform folder of the cloned project.

   ```
   $ cd elasticsearch-ml-vector-search-tutorial/terraform
   ```

2. On your machine, create a document that is named `terraform.tfvars`, with the following fields:

```
ibmcloud_api_key = "<your_api_key_from_step_1>"
region = "<your_region>"
elastic_password = "<make-up-a-password>"
```

> ⚠ **Important:** The `terraform.tfvars` document contains variables that you might want to keep secret, so it is excluded from the public Github repository.

3. Install the infrastructure with the following command:

```
$ terraform init
terraform apply --auto-approve
```

4. Finally, export the database access URL to your terminal environment. It will be required by subsequent steps.

```
$  terraform output --json
 export ES_URL="<the url value obtained from the output>"
```

Step 4: Install dependencies

You need to install some Python dependencies:

```
$    pip3 install elasticsearch
   pip3 install Pillow
   pip3 install imgbeddings
   pip3 install requests
```

Step 5: Generate vector embeddings for your images

Create a folder called `images` at the root of the project folder structure. Inside it, create one or more folders with different images. For example, if you have a dataset of cars then you may want to create folders for different types of car, for example `fordescort` and `fordcortina`. This is not strictly necessary (all images could go in a folder called `cars`), but organizing folders may make it easier to identify search matches later on.

You are ready to run the `create.py` script. In the root of the project type:

```
$ python3 create.py
```

This script creates an Elasticsearch index called `images` in your Databases for Elasticsearch database.

Then, it cycles through the `images` folder and for each image it finds it create a set of embeddings using this open source Python package, which uses the open source CLIP model from OpenAI. With those embeddings and a small amount of metadata (file path and file id), the script creates a document that is then uploaded to the Elasticsearch index.

Depending on the size of your dataset, this process could take multiple hours to complete.

Step 6: Search your dataset

You are now ready to test your new dataset. To do this you need to find another image of a car (if you are using cars) that is not part of your original dataset. Save this image (for example, `myimage.jpg`) to the root of the project.

Run the `search.py` script, passing in the image name:

```
$ python3 search.py myimage.jpg
```

The script generates a set of embeddings for the supplied image using the same algorithm as before. It attempts a known nearest neighbor search on the dataset to find the closest match in the dataset to the image that was supplied in the search. It returns the details of the closest match.


```
{"took": 4947, "timed_out": false, "_shards": {"total": 1, "successful": 1, "skipped": 0, "failed": 0}, "hits": {"total": {"value": 1, "relation": "eq"}, "max_score": 0.97320974, "hits": [{"_index": "images", "_id": "5c910de5357cb9a3f1b43e6618b141afa6666bfca8676269d5e10a14e1688819", "_score": 0.97320974, "fields": {"file_path": ["./images/Bald_Eagle/26897.jpg"], "desc": ["Bald_Eagle"]}}]}}
```


You can repeat this process with other images.

Step 7: Image similarity, not bird similarity

What you have built is more of an image similarity search than a bird (or car) similarity search. The vector search algorithm is analyzing the whole image and looking for matches. An image of a warbler sitting on a tree branch might be more similar to an image of a tree sparrow sitting on a tree branch than to an image of a warbler in flight. Nevertheless, it is a powerful tool to make relationships between objects that (until recently) were difficult, if not impossible.

Step 8: Tear down your infrastructure

Your Databases for Elasticsearch incurs charges. After you finish this tutorial, you can remove all the infrastructure by going to the `terraform` directory of the project and using the command:

```
$ terraform destroy
```

Next Steps

If you are ready to explore further, you can also use Databases for Elasticsearch to not only store vector embeddings but to generate them, as well. That is the subject of our next tutorial in this series.

Use ELSER, Elastic's Natural Language Processing model

Elastic Learned Sparse EncodeR (ELSER) is a natural language processing (NLP) model trained by Elastic that enables you to perform semantic search by using sparse vector representation. Instead of literal matching on search terms, semantic search retrieves results based on the intent and the contextual meaning of a search query.

In this tutorial you provision an instance of Databases for Elasticsearch Platinum and apply the ELSER model to a body of text and see how it enhances the quality of search results.

This is the second of three tutorials exploring the capabilities of Elasticsearch around Machine learning. Machine Learning is a branch of artificial intelligence (AI) and computer science that focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. By using statistical methods, algorithms are trained to make classifications or predictions, and to uncover key insights in data mining projects.

See the other tutorials in this Elasticsearch machine learning series:

- Use Elasticsearch vector search capabilities
- Use machine learning models with Elasticsearch to tag content

Note: Databases for Elasticsearch is a paid-for service, so following this tutorial will incur charges.

Step 1: Before you start

To begin the provisioning process, install some must-have tools:

- IBM Cloud account
- Terraform - To codify and provision infrastructure.
- Python
- jq - To process configuration files.

Step 2: Obtain an API key to deploy infrastructure to your account

Create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

Important: For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Step 3: Clone the project

Clone the project from the GitHub repository.

```
$ git clone https://github.com/IBM/elasticsearch-ml-elser-tutorial.git
```

Step 4: Install the Elasticsearch cluster

1. Navigate into the terraform folder of the cloned project.

```
$ cd elasticsearch-ml-elser-tutorial/terraform
```

2. On your machine, create a document that is named `terraform.tfvars`, with the following fields:

```
ibmcloud_api_key = "<your_api_key_from_step_1>"
region = "<your_region>"
elastic_password = "<make-up-a-password>"
```

> ⚠ **Important:** The `terraform.tfvars` document contains variables that you might want to keep secret, so it is excluded from the public Github repository.

3. Install the infrastructure with the following command:

```
$ terraform init
terraform apply --auto-approve
```

Finally, export the database access URL to your terminal environment (it will be required by subsequent steps)

```
$  terraform output --json
export ES="<the url value obtained from the output>"
cd ..
```

Step 5: Install the ELSER model

The ELSER model needs to be installed and started before you can use it. Install it with a command like:

```
$ curl -kX PUT "$ES/_ml/trained_models/.elser_model_1?pretty" -H 'Content-Type: application/json' -d'
{
  "input": {
 "field_names": ["text_field"]
  }
}
'
```

Next, start it by typing:

```
$ curl -kX POST "$ES/_ml/trained_models/.elser_model_1/deployment/_start?deployment_id=for_search&pretty"
```

> 🔖 **Note:** You may have to wait a few minutes for the model to finish installing before the start command can execute.

Step 6: Create an index and mappings for your data

You should be in the root of your project folder. In the terminal, use a command like:

```
$ curl -kXPUT -H"Content-Type: application/json" -d@mapping.json $ES/test_data
```

This creates an index called `test_data` in your ES instance. It will also create some explicit mappings. This is the way Elastic has of defining how a document, and the fields it contains, are stored and indexed. The mapping definition is stored in the `mapping.json` file. It defines a field called `ml.tokens` which will be used to store all the additional data (tokens) that will be created my the machine learning algorithm when it analyses the data you upload. That data will be contained in a field called `text`, which is also explicitly defined in the mapping document.

Step 7: Create an ingest pipeline for analysing your data

The ELSER model comes pre-installed in all Platinum deployments of Databases for Elasticsearch. All you have to do is create a pipeline that uses it to analyze your incoming data.

```
$ curl -kX PUT -H"Content-Type: application/json" -d@pipeline.json $ES/_ingest/pipeline/elser-v1-test
```

The `pipeline.json` document has the pipeline definition. It describes what processing will happen to documents when they are uploaded. In this case it defines the use of the ELSER model and tells ES to put any resulting data in the `ml.tokens` field of the index.

This is a very simple pipeline. For example it does not deal with any ingest errors, just discards them. But it is sufficient for the purposes of this demo.

Step 8: Upload your data

The `import.json` document contains data from the `msmarco-passagetest2019-top1000` data set, which is a subset of the MS MARCO Passage Ranking data set. It consists of 200 queries, each accompanied by a list of relevant text passages. All unique passages, along with their IDs, have been extracted from that data set and made ready for bulk upload to Elasticsearch.

In this step, you upload the data to Elasticsearch and pass it through the ELSER analyzer as it uploads.

Doing Natural Language Processing analysis is compute-intensive, so this process will take multiple hours (up to 12 hours). We will be doing the bulk upload in batches of 100 entries to ensure that the pipeline buffer is not overrun.

So first, split the `import.json` document into smaller documents of 100 entries each (every entry is two lines in the document):

```
$ split -l 200 -a 4  import.json
```

This creates several hundred small documents. Import them by running the `upload.sh` script:

```
$ ./upload.sh
```

For each uploaded document, the ELSER model tries to infer what the text is about, and will generate a group of words that it thinks are relevant, along with a relevancy score. Here's one example of an ingested document:


```
{
  "_index": "test_data",
  "_id": "1",
  "_version": 3,
  "_seq_no": 1310,
  "_primary_term": 1,
  "found": true,
  "_source": {
    "text": "This is the definition of RNA along with examples of types of RNA molecules. This is the definition of RNA along with examples of types of RNA molecules. RNA Definition",
    "ml": {
      "tokens": {
        "molecules": 0.9468944,
        "cell": 0.31727117,
        "type": 0.1881346,
        "lab": 0.14050934,
        "example": 0.27423903,
        "strand": 0.11950072,
        "dna": 0.82435495,
        "protein": 0.25066674,
        "term": 0.5622088,
        "definition": 0.21340553,
        "nuclear": 0.03731725,
        "different": 0.026590228,
        "element": 0.42168307,
        "genetic": 0.36145726,
        "types": 0.7244048,
        "characteristics": 0.24549837,
        "adam": 0.15298073,
        "rna": 1.949169,
        "organism": 0.27345642,
        "gene": 0.57350045,
        "substance": 0.006454099,
        "mrna": 1.002312,
        "bond": 0.096653655,
        "structure": 0.1670035,
        "genome": 0.24130683,
        "sequence": 0.21949387,
        "q": 0.028950738,
        "unit": 0.1926791,
        "examples": 1.0590855,
        "material": 0.034060754,
        "chemical": 0.454947,
        "science": 0.20523745,
        "biological": 0.47201967,
        "molecule": 0.92732555,
```

```
      "atom": 0.021480415,
      "word": 0.2971402
    },
    "model_id": ".elser_model_1"
  }
 }
}
```

You can see that it has created relationships beyond the actual words on the text. For example, it has infered that this snippet is about `science`, `genes` and `dna`.

Sit back and relax. Your import will take a bit of time.

Step 9: Search your dataset and compare results

Once your dataset is uploaded, you are ready to test it, in particular the difference that the ELSER NLP processor can make to the quality of your search results. The following search query will make use the generated tokens to return results that appear to be most relevant:

```
$ curl -k -H"Content-Type: application/json" -d@elserquery.json $ES/test_data/_search | jq .
```

Compare that with a query that does NOT use the tokens and is simply searching over the original text:

```
$ curl -k -H"Content-Type: application/json" -d@query.json $ES/test_data/_search | jq .
```

You can try other queries by changing the text in the `query.json` and `elserquery.json` files. Try things like:

`what is the best exercise for stiff limbs?`

or

`explain how the US president is elected`

In all these cases, the ELSER-enhanced results are much more relevant.

Of course, that does not apply to all searches. A search for `taylor swift`, for example, will produce similar results on both. So it all depends on the type of search and the data in your index. But you can see that many searches could be significantly improved by applying the ELSER ML model.

Step 10: Tear down your infrastructure

Your Databases for Elasticsearch incurs charges. After you finish this tutorial, you can remove all the infrastructure by going to the `terraform` directory of the project and using the command:

```
$ terraform destroy
```

Step 11: Ready for more?

ELSER is just one ML model. There are many others and with the Databases for Elasticsearch Platinum Plan you can easily deploy others and analyze your data using them instead. We have put together a tutorial that shows you how to do just that

Use machine learning models with Elasticsearch to tag content

Objectives

Databases for Elasticsearch supports machine learning workloads. In this tutorial, you learn how to provision a machine learning model to a Databases for Elasticsearch instance and then use it to extract meaningful additional information from a test data set. Only some basic knowledge of terminal commands is required to provision and understand this tutorial.

Machine learning is a branch of artificial intelligence (AI) and computer science that focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. By using statistical methods, algorithms are trained to make classifications or predictions, and to uncover key insights in data mining projects.

These learning algorithms are known as “models”. In this tutorial, we use a pre-built Natural Language Processing (NLP) model, which extracts meaning out of sentences in written (Natural) language. Specifically, we use the *distilbert-base-uncased-finetuned-conll03-english* model that tries to identify the names of people, locations, and organizations within text.

Many other models specialize in analyzing other forms of data, like text extraction from images, speech conversion to text, or object identification in

images. For a full list of Elastic stack supported models, see Compatible third party NLP models. Models can be trained on domain-specific knowledge, but the training of new models is beyond the scope of this tutorial.

This tutorial guides you through the process of:

- Provisioning a Databases for Elasticsearch instance
- Uploading a machine learning model
- Uploading a data set of headlines and summaries from news articles
- Passing the data set through the NLP model
- Querying the augmented data to see the results of the model on the data.

See the other tutorials in this Elasticsearch machine learning series:

- Use ELSER, Elastic's Natural Language Processing model
- Use Elasticsearch vector search capabilities

Getting productive

To begin the provisioning process, install some must-have productivity tools:

- You need to have an IBM Cloud account.
- Terraform - To codify and provision infrastructure.
- Python
- jq - To process configuration files.

Step 1: Obtain an API key to provision infrastructure to your account

Follow these steps to create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

> ⚠ **Important:** For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Step 2: Clone the project

Clone the project from the GitHub repository.

```
$ git clone https://github.com/IBM/elasticsearch-nlp-ml-tutorial.git
cd elasticsearch-nlp-ml-tutorial/terraform
```

Step 3: Install the infrastructure

Provision your Databases for Elasticsearch instance.

1. On your machine, create a document that is named `terraform.tfvars`, with the following fields:

   ```
   ibmcloud_api_key = "<your_api_key_from_step_1>"
   region = "<your_region>"
   es_password  = "<make_up_a_password>"
   ```

   > **Note:** The `terraform.tfvars` document contains variables that you might want to keep secret so it is ignored by the GitHub repository.

2. Install the infrastructure with the following command:

   ```
   $ terraform init
   terraform apply --auto-approve
   ```

   The Terraform script outputs the configuration data that is needed to run the application, so copy it into the `scripts` folder:

   ```
   $ terraform output -json >../scripts/config.json
   cd ../scripts
   ```

Step 4: Install Eland

Eland is a Python Elasticsearch client for exploring and analyzing data in Elasticsearch. Install it with a command like:

```
$ python3 -m pip install 'eland[pytorch]'
```

Upload and analyze data

This tutorial uses a small data set of 132 articles that are obtained from the BBC News and Guardian websites through their RSS feeds.

These have been transformed into a json file that is formatted as required by the Elasticsearch bulk upload method.

Run the `upload.sh` script, which does the following:

- Uploads the NLP model to Elasticsearch.
- Creates a data processing pipeline in Elasticsearch that takes any incoming data and analyzes it for meaningful terms.
- Uploads the pre-formatted data to Elasticsearch and passes it through the pipeline for analysis.

⚠ **Important:** Since this is a demo, we connect nonsecurely to your database. For production, use secure connections.

To run the script, make sure you are in the `scripts` directory and use the command:

```
$ ES_PASSWORD=`cat config.json | jq -r .es_password.value`
ES_PORT=`cat config.json | jq -r .es_port.value`
ES_HOST=`cat config.json | jq -r .es_host.value`
export ES="https://admin:${ES_PASSWORD}@${ES_HOST}:${ES_PORT}"
./upload.sh
```

Step 5: Query your data

You now have an index that is named `test` that has 132 records. Each of these records contains the news data (article ID, headline and summary) and an additional object called `tags`. This is where the machine learning model has inserted any people (PER), locations (LOC) or organizations (ORG) that it has found in the text.

The index also contains another object, called `ml`, which shows some of how the model works. For example, it tells you what accuracy probability it assigned to each of the terms it found.

For example, use this query to retrieve a single record to inspect:

```
$ curl -k "$ES/test/_search?size=1" | jq .
```

This machine learning model has generated valuable information. For example, if you run a news website, you might generate pages of tag-based content. If, for example, your users go to a page like www.mynewssite.com/tag/rishi-sunak, all your system would have to do is search by that tag in the index of news articles to retrieve a list of those that mention Rishi Sunak:

```
$ curl -kX POST -d@body.json -H "Content-Type: application/json" "$ES/test/_search" | jq .
```

There is a `body.json` file in the `scripts` directory that you can play around with to make different searches.

Conclusion

This tutorial shows how to use Databases for Elasticsearch to harness the power of machine learning to generate valuable additional information from your data. We hope you can use it as a springboard to explore other ways to augment and create value from your data.

To stop incurring charges, don't forget to remove all your deployed infrastructure. In your terraform directory, use the command:

```
$ terraform destroy
```

Build an Elasticsearch chatbot

Objectives

This tutorial shows how an IBM watsonx.ai model can be enhanced with knowledge gleaned by spidering content from your website to produce a chatbot

that is capable of answering questions related to your knowledge base. This technique is known as Retrieval-Augmented Generation (RAG). Pre-trained large language models have good *general knowledge*, being trained with a large corpus of public content, but they lack *domain-specific knowledge* about your business, such as:

- "Can I get a refund if the box is opened?"
- "What is the waiting list for treatment?"
- "Do you deliver on Saturdays?"

We can build a chatbot using the following IBM Cloud services:

- Databases for Elasticsearch that runs the ELSER Natural Languge Processing (NLP) model to enhance the incoming data before being stored in an Elasticsearch index. An *ingest pipeline* is used to allow data to feed into ELSER before the enhanced data is stored.
- Elastic Enterprise Search is deployed on IBM Cloud® Code Engine and is used to spider your website to collect domain-specific data and feed it into Elasticsearch's ingest pipeline.
- Kibana is deployed on IBM Cloud Code Engine and becomes the web UI for Elasticsearch and Elastic Enterprise Search. It is used to specify and to set off the web crawler.
- IBM watsonx.ai runs a pre-trained machine learning model to answer chatbot requests. The model's API is used to produce chatbot responses given the user prompt and the contextual data collected by running the prompt against the spidered and enhanced data in Elasticsearch.
- A simple Python app is deployed on IBM Cloud Code Engine to provide a chatbot web interface. It collects prompts from users, queries the Elasticsearch data and then uses IBM watsonx.ai to produce the response.

⚠ **Important:** Databases for Elasticsearch, IBM Cloud Code Engine, and IBM watsonx are paid products so this tutorial incurs charges.

Prerequisites

- An IBM Cloud account.
- Terraform - to deploy infrastructure.
- A website containing public-facing text content that we will "spider" to bolster the chatbot's expertise.
- Docker running on your local machine.

Follow these steps to create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

Note: For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Set up an IBM watsonx.ai project

Most of the infrastructure is deployed through Terraform, but IBM watsonx.ai must be set up manually.

IBM watsonx.ai is a studio of integrated tools for working with generative AI capabilities that is powered by foundation models for building machine learning applications. The IBM watsonx.ai component provides a secure and collaborative environment where you access your organization's trusted data, automate AI processes, and deliver AI in your applications.

Follow these steps to set up IBM watsonx.ai:

1. Sign Up for IBM watsonx.ai as a Service and click on the **Get Started** link for watsonx.ai. Select a region and log in.
2. Create a project within watsonx.ai. In the **Projects** box, click **+** and **Create Project**. Name you project and supply the IBM Cloud® Object Storage instance that will be used to store the project's state. (If you don't have a Object Storage instance, create one.
3. In the project's **Manage** tab, in the General page, make a note of the "Project ID".
4. In the project's **Manage** tab, in the **Services and Integrations** page, click **Associate Service**. Then, click **New Service** and choose the **Watson Machine Learning** option. You will use the Lite plan, so just click **Create**.

Provision the infrastructure with Terraform

Clone the repo:

```
$ git clone https://github.com/IBM/icd-elastic-bot.git
cd icd-elastic-bot
cd terraform
```

In this directory, create a file called `terraform.tfvars` containing the following data, but replacing the placeholder (`MY_*` values) with your own:

```
$ ibmcloud_api_key="MY_IBM_CLOUD_API_KEY"
region="eu-gb"
es_username="admin"
es_password="MY_ELASTICSEARCH_PASSWORD"
es_version="8.12"
wx_project_id="MY_WATSONX_PROJECT_ID"
```

Pick a secure Elasticsearch password that together with the Elasticsearch username will become the credentials required to access Elasticsearch and the Kibana web user interface.

Now deploy the infrastructure with:

```
$ terraform init
terraform apply --auto-approve
```

Terraform will output:

- the URL of your Kibana instance.
- the URL of the Python app.

Make a note of these values for the next steps.

Feed the data

This step feeds your website's data to your Databases for Elasticsearch instance. You will use the Elastic web crawler. This feature, accessed through Kibana, is used to extract data from any website.

Follow the steps to add your data:

1. Navigate to your Kibana URL - Terraform output this URL from the previous section. Log in with the Elasticsearch username and password you chose.
2. In the Kibana UI, in the Search section, choose the **Overview** option. Click on **Crawl URL**.
3. Name your index *search-bot* (the `search-` prefix is already present). Click **Create index**.
4. Add your website's URL in the **Manage domain** section of the index. Click **Validate domain** and then **Add domain**.
5. Click Add Inference Pipeline in the Machine Learning Inference Pipeline section and follow the steps. Select .elser_model_1 for the trained ML Model and make sure the model is in a *Started* state. Select the title field in Select field mappings step and click **Add**, then **Continue**, and **Create pipeline**.
6. Click on **Crawl all domains** and then **Crawl all domains on this index** . Then, wait until the data is collected.

Query the data

Navigate to the Python app's URL in your web browser - this can be found in the output from Terraform as `python_endpoint` from previous steps.

Start interacting with the model by asking questions, and you will receive answers generated by IBM watsonx.ai. The Python app takes your prompt and searches the Elasticsearch index populated with spidered and enhanced data from the web crawl. It then uses the IBM watsonx.ai API to produce a response from the context provided by the Elasticsearch result.

Conclusion

In this tutorial you created an Databases for Elasticsearch instance that is paired with Kibana and Elastic Enterprise Search hosted on IBM Cloud Code Engine. You then configured Elasticsearch to use its Elser model in a pipeline so that when you spidered a website's contents, its JSON documents were augmented with *sparse vector* data. You also deployed a Python app that takes user prompts, queries the spidered data in Elasticsearch to gather domain-specific context before sending the prompt and the context to a watsonx.ai large language model to formulate a response to the prompt.

Now you can create your own chat applications using watsonx.ai, enhanced by modelling your own domain-specfic data held in Databases for Elasticsearch.

Your Databases for Elasticsearch incurs charges. After you finish this tutorial, you can remove all the infrastructure by going to the `terraform` directory of the project and using the command:

```
$ terraform destroy
```

Cloud Databases tutorials

Deploying and connecting a Cloud Databases instance

Objectives

This tutorial guides you through the process of deploying a Cloud Databases instance and connecting it to a web front end by creating a webpage that allows visitors to input a word and its definition. These values are then stored in a database running on Cloud Databases. You install the database infrastructure by using Terraform and your web application uses the popular Express framework. The application can then be run locally, or by using Docker.

Getting productive

To begin the deployment process, install some must-have productivity tools:

- You need to have an IBM Cloud account.
- Node.js and npm - to install packages from public npm registries
- Terraform - to codify and deploy infrastructure
- *Optional* Docker - to run your application nonlocally

Step 1: Obtain an API key to deploy infrastructure to your account

Follow these steps to create an IBM Cloud API key that enables Terraform to provision infrastructure into your account. You can create up to 20 API keys.

> ⚠ **Important:** For security reasons, the API key is only available to be copied or downloaded at the time of creation. If the API key is lost, you must create a new API key.

Step 2: Clone the project

Clone the project from the Cloud Databases Hello World project GitHub repository.

```
$ git clone https://github.com/IBM-Cloud/clouddatabases-helloworld-examples.git
```

Step 3: Install the infrastructure

In this step, you deploy an instance of the database service you want to use. The GitHub repository contains folders for various Cloud Databases resources.

1. From the main GitHub project folder, navigate into the `terraform` service folder of your choice, for example, `mysql/terraform`.
2. On your machine, create a document that is named `terraform.tfvars`, with the following fields:

   ```
   $ ibmcloud_api_key = "<YOUR_API_KEY_FROM_STEP_1>"
   region = "<YOUR_REGION>"
   admin_password  = "<CREATE_15_CHARACTER_PASSWORD>"
   ```

   The `terraform.tfvars` document contains variables that you might want to keep secret so it is ignored by the GitHub repository.
3. Install the infrastructure with the following command:

   ```
   $ terraform init
   terraform apply --auto-approve
   ```

   The Terraform script outputs configuration data that is needed to run the application, so copy it into the root folder:

   ```
   $ terraform output -json >../config.json
   ```

Step 4: Run your app locally

1. To connect to the database from your local machine, ensure that you are in your service folder, then install the node dependencies and run the service with the following commands:

   ```
   $ npm install
   ```

```
$ npm run start
```

If successful, the output shows you are connected:

```
$ #Connected!
#Server is listening on port 8080
```

2. Open a browser and visit http://localhost:8080. You are greeted by a welcome page with a database logo that is displayed in your browser window.
3. To test the interface, enter a word and its definition. The data pair is added to the database and appears in a list at the bottom of the page.

Step 5: Run the app from a Docker container (optional)

The first step toward hosting your application from a service like Code Engine is to containerize the app code inside a Docker container and run it from there.

1. Make sure you are logged in to your Docker account. In the service folder of your chosen database, enter the following command:

```
$ docker build -t database-hello-world:1.0 .
docker run -p 8080:8080 database-hello-world:1.0
```

2. Open a browser and visit http://localhost:8080 to see the same welcome page from the Step 4.

Congratulations, you've created an app with a front end that feeds data into your Cloud Databases deployment!

Connect a Cloud Databases deployment to an IBM Cloud Kubernetes Service application

The Cloud Databases "Hello World" Kubernetes examples repository holds sample IBM Cloud® applications, which are written in various programming languages, that detail how to connect a Cloud Databases deployment to an IBM Cloud Kubernetes Service application.

Each Git branch of the examples repository corresponds to samples in a particular programming language, either JavaScript that uses Node.js, or python. The files in each folder correspond either to a database or a message queue.

Trying out the sample applications

Clone the respective repo you want to use. For instance, you can clone the **Node** repository by selecting the **Node** branch. Then, click **Clone or download** to get the URL you need to clone by using SSH or HTTPS. This command looks like:

```
$ git clone -b node git@github.com:IBM-Cloud/clouddatabases-helloworld-kubernetes-examples.git
```

Or, clone by using HTTPS:

```
$ git clone -b node https://github.com/IBM-Cloud/clouddatabases-helloworld-kubernetes-examples.git
```

After the branch is cloned, select the appropriate directory for the database you want to try. Each database has its own copy of these instructions on how to provision and deploy a database or message queue and an application on IBM Cloud Kubernetes Service.

Running on IBM Cloud

1. If you do not already have an IBM Cloud account, sign up here.
2. Download and install IBM Cloud CLI. The IBM Cloud CLI tool enables you to communicate with IBM Cloud from your console or CLI.
3. Install the Kubernetes Service CLI plug-in and the Container Registry CLI plug-in

```
$ ibmcloud plugin install container-service
ibmcloud plugin install container-registry
```

To verify their installation, run

```
$ ibmcloud plugin list
```

You receive a response like this:

```
Listing installed plug-ins...
```

```
Plugin Name                          Version   Status
container-registry                   0.1.382
container-service/kubernetes-service 0.3.34
```

4. Download and install the Kubernetes CLI.

   Follow the instructions for downloading and installing the Kubernetes CLI for the platform that you're using.

5. Connect to IBM Cloud in the CLI tool and follow the prompts to log in.

```
$ ibmcloud login
```

> **Note:** If you have a federated user ID, use the `ibmcloud login --sso` command to log in with your single sign-on ID.

Creating your database

> **Note:** This process creates a standard database instance in the service that you specify that might incur extra charges in your selected plan.

1. You must target a resource group, by using the following command:

```
$ ibmcloud target -g <RESOURCE_GROUP>
```

> **Tip:** For more information, see Working with resources and resource groups (ibmcloud resource).

1. The database can be created from the CLI by using the `ibmcloud resource service-instance-create` command. The command takes a service instance name, a service name, plan name, and location.
2. The service name is one of the Cloud Databases services, `databases-for-elasticsearch`, `databases-for-mongodb`, `databases-for-postgresql`, `databases-for-redis`, `messages-for-rabbitmq`, or `databases-for-mysql`.

```
$ ibmcloud resource service-instance-create <INSTANCE_NAME> <SERVICE_NAME> standard <REGION>
```

> **Tip:** Remember the database instance name. Find your region identifier here.

> **Note:** The previous example provisions a Shared Compute instance. For more information, see the Hosting models overview.

Configuring the Kubernetes app

1. Create a Kubernetes Service. Choose the location and resource group in which you'd like to set up your cluster. Select the cluster type that you want to use. This example requires only the lite plan, which comes with one worker node. After a cluster is provisioned, you are given a list of steps to access your cluster and set the environment variables under the *Access* tab. You are also able to verify that your deployment is provisioned and running normally.
2. Make sure that you are targeting the correct IBM Cloud resource group of your Kubernetes Service.

   If your resource group is named anything other than `default`, use the following command to target your cluster resource group:

```
$ ibmcloud target -g <RESOURCE_GROUP_NAME>
```

   This example uses the `default` resource group.

3. Create your own private image repository in Container Registry to store your application's Docker image. Since we want the images to be private, we need to create a namespace, which creates a unique URL to your image repository.

```
$ ibmcloud cr namespace-add <YOUR_NAMESPACE>
```

4. Add the Cloud Databases deployment to your cluster.

```
$ ibmcloud ks cluster service bind --cluster <YOUR_CLUSTER_NAME> --namespace default    --service <INSTANCE_NAME_OR_CRN>
```

> **Note:** The "default" namespace refers to the Kubernetes instance and not the user-created image store namespace. Likewise, if your database uses both public and private endpoints, your public endpoint is used by default. Therefore, if you want to select the private endpoint, first you need to create a service key for your database so Kubernetes can use it when binding to the database. You set up a service key by using the command:

```
$ ibmcloud resource service-key-create <YOUR-PRIVATE-KEY> --instance-name    <INSTANCE_NAME_OR_CRN> --service-endpoint private
```

The private service endpoint is selected with `--service-endpoint private`. After that, you bind the database to the Kubernetes cluster through the private endpoint by using the command

```
$ ibmcloud ks cluster service bind <YOUR_CLUSTER_NAME> default    <INSTANCE_NAME_OR_CRN> --key <YOUR-PRIVATE-KEY>
```

5. Verify that the Kubernetes secret was created in your cluster namespace. Kubernetes uses secrets to store confidential information like the IBM Cloud Identity and Access Management (IAM) API key and the URL that the container uses to gain access. To set the cluster as the context for this session and then get the API key for accessing the instance of your deployment, run the following commands

   ```
   $ ibmcloud ks cluster config --cluster <CLUSTER_NAME_OR_ID>
   ```

   Then

   ```
   $ kubectl get secrets --namespace=default
   ```

   > **Note:** Save the name of the secret that was generated when you bound `your_database_name` to your Kubernetes service.

6. If you haven't already, clone the app in one of the available languages to your local environment from your console by using the following command

   ```
   $ git clone -b <LANGUAGE> git@github.com:IBM-Cloud/clouddatabases-helloworld-kubernetes-examples.git
   ```

7. `cd` into this newly created directory, and `cd` into the database folder. The code for connecting to the service, and reading from and updating the database can be found in `server.js`. See Code Structure and the code comments for information on the app's functions. A `public` directory contains the html, stylesheets, and JavaScript for the web app. However, to get the application to work, we first need to push the Docker image of this application to our Container Registry.

8. Build and push the application's Docker image to your Container Registry. Specify the appropriate region and give the container a name.

   ```
   $ ibmcloud cr build -t <REGION>.icr.io/<NAMESPACE>/<CONTAINER_NAME> .
   ```

   You can view the image in container registry by using

   ```
   $ ibmcloud cr images
   ```

   You get something like the following response

   ```
   REPOSITORY                          TAG     DIGEST       NAMESPACE   CREATED       SIZE    SECURITY STATUS
   <region>.icr.io/mynamespace/container_name latest   81c3959ea657  mynamespace 4 hours ago   28 MB   No Issues
   ```

9. Update the Kubernetes deployment configuration file `clouddb-deployment.yaml`.

   Change the `image` name to the repository name that you got from the previous step:

   ```
   $ image: "<REGION>.icr.io/mynamespace/<container_name>" # Edit me
   ```

   Now, under `secretKeyRef`, change the name of `<db-secret-name>` to match the name of the secret that was created when you bound your database deployment to your Kubernetes cluster.

   ```
   $ secretKeyRef:
     name: <DB-SECRET-NAME> # Edit me
   ```

   As for the `service` configuration at the end of the file, nodePort indicates the port that the application can be accessed from. You have ports in the range 30000 - 32767 that you can use, but we chose 30081. The TCP port is set to 8080, which is the port the Node.js application runs on in the container.

Deploying your Kubernetes app

1. Deploy the application to Kubernetes Service. When you deploy the application, it is automatically bound to your Kubernetes cluster.

   ```
   $ kubectl apply -f clouddb-deployment.yaml
   ```

2. Get the IP for the application.

   ```
   $ ibmcloud ks workers -c <CLUSTER_NAME>
   ```

   The result is something like:

   ```
   ID                                                 Public IP       PrivateIP      Machine Type   State    Status   Zone    Version
   kube-hou02-pa1a59e9fd92f44af9b4147a27a31db5c4-w1   199.199.99.999   10.76202.188   free           normal   Ready    hou02   1.10.11_1536
   ```

   Now you can access the application from the Public IP from port 30082.

   The clouddatabases-helloworld app displays the contents of an *examples* database. To demonstrate that the app is connected to your service, add some words to the database. The words are displayed as you add them, with the most recently added words displayed first.

Code structure

File	Description
server.js	Establishes a connection to the database by using credentials from BINDING (the name that we created in the Kubernetes deployment file to display the credentials) and handles create and read operations on the database.
main.js	Handles user input for a PUT command and parses the results of a GET command to output the contents of the database.

Code structure

The app uses a PUT and a GET operation:

- PUT
  - Takes user input from main.js.
  - Adds the user input to the database.
- GET
  - Retrieves the contents of the database.
  - Returns the response of the database command to main.js.

Example context-based restrictions scenarios

With context-based restrictions, account owners and administrators can define and enforce access restrictions for IBM Cloud® resources, based on the context of access requests. Access to Cloud Databases resources can be controlled with context-based restrictions and identity and access management policies. For more information, see Protecting Cloud Databases resources with context-based restrictions.

Restrict traffic to your deployment by using Cloud Databases Allowlisting

In this example scenario, you use context-based restrictions to restrict traffic to your IBM Cloud® Databases for MySQL cluster in the `in-che` region by allowing only the set of subnets from the Cloud Databases Allowlist page to connect to your deployment.

In the following steps, you start by creating a network zone, or allowlist, that includes your subnets. Then, you create a context-based restrictions rule for your deployment. When you create the rule, you associate it with the network zone that contains the individual IP address.

Prerequisites

Before beginning this tutorial, make sure you have created or installed the following resources and tools.

- An IBM Cloud account. For more information, see Creating an account.
- The [Cloud Databases CLI plug-in/docs/cloud-databases?topic=cloud-databases-cdb-reference) - the CLI interface to interact with the Cloud Databases API. For more information, see Getting started with the IBM Cloud CLI.

- A Cloud Databases deployment. For more information, see Provisioning.
- Review the What are context-based restrictions docs to get an understanding of concepts of network zones and rules.
- Review the Protecting Cloud Databases resources with context-based restrictions docs to understand how you can leverage CBR for your Cloud Databases resources.

Step 1: Creating your network zone in the CLI

Provision your service from the IBM Cloud Catalog and choose your IP addresses from the Cloud Databases Allowlist page.

1. Run the following example command to create a network that includes a range of allowed IP addresses.

```
$ ibmcloud cbr zone-create --
addresses=169.38.95.127/27,169.38.121.159/28,169.38.132.127/25,169.38.136.255/26,169.38.73.151/29,169.38.105.79/29,10.162.8.127/26,10.163.20.127/25,10.162.115.103/29,10.162
9 --name=tutorial_zone
```

2. Verify the network zone was created.

```
$ ibmcloud cbr zones
```

Step 2: Creating your CBR rule in the CLI

1. After you create your network zone (allowlist), create a context-based restrictions rule and add the network zone you created in the previous step. The following example creates a rule that uses the `data-plane` API type. Replace `ZONE-ID` with the ID of the `tutorial_zone` network zone that you created in Step 1.

```
$ ibmcloud cbr rule-create --enforcement-mode enabled --context-attributes networkZoneId=<ZONE-ID> --resource-group-id <RESOURCE_GROUP_ID> --service-name
databases-for-mysql --service-instance <SERVICE-INSTANCE> --api-types crn:v1:bluemix:public:context-based-restrictions::::api-type:data-plane --description
<DESCRIPTION>
```

Understanding the command options.

`--context-attributes (string)`

Contexts to add to the rule. Can only pass in one `networkZoneId` field at a time. This option can be repeated to add multiple network zones. Please use this if you need to specify the `endpointType` with the `networkZoneId`.

`--zone-id (string)`

Shorthand for adding context attribute `networkZoneId` to the context without specifying endpoint.

`--resource-attributes (string)`

Resources to add to the rule.

`--resource-group-id (string)`

Shorthand for creating IBM Cloud resource attribute `resourceGroupId`. Used to restrict the rule to a single resource group.

`--region (string)`

Shorthand for creating IBM Cloud resource attribute `region`. Used to restrict the rule to a single region.

`--service-name (string)`

Shorthand for creating IBM Cloud resource attribute `serviceName`.

`--service-instance (string)`

GUID of the service instance to scope the context to. This option can be omitted if the context applies to more than one of you service instances. This option is exclusive with the `--file` option.

2. Verify the rule was created.

```
$ ibmcloud cbr rules
```

Step 3: Testing your context-based restrictions

To test your context-based restrictions setup, try connecting to your deployment from an IP address other than the IP addresses that you allowlisted in your network zone. With this setup, only the IP addresses in your network zone can connect to your deployment.

Setting up disk alerts for disk utilization

Objectives

Getting timely alerts about resource utilization is key to managing your database, avoiding problems, and mitigating downtime. If you know in advance that your database is running out of disk, take steps to scale those resources.

In this tutorial, you use the IBM Cloud API and the IBM Cloud CLI to set up an alert that emails you whenever the disk utilization of your database exceeds 90%. This specific example creates an alert on a Databases for Elasticsearch deployment, but it is applicable to all the databases in the IBM Cloud Databases catalog.

Getting productive

Set up monitoring instance and Platform Metrics

To get started, you need access to IBM Cloud® Monitoring in your database region, and you need to have a monitoring instance available. This monitoring instance must be in the same region as the database target.

You also must have Platform Metrics enabled.

Install Command Line Interface tools

Next, you need the IBM Cloud Monitoring CLI and Cloud Databases CLI plug-in.

Install the IBM Cloud Monitoring CLI by running the following command:

```
$ ibmcloud plugin install monitoring
```

Install the Cloud Databases CLI plug-in by running the following command:

```
$ ibmcloud plugin install cloud-databases
```

You are now ready to retrieve and monitor your service instance.

Step 1: Retrieve your monitoring service instance

In this step, you retrieve the necessary credentials to gain access to your monitoring instance.

Begin by logging in to the IBM Cloud CLI with the following command:

```
$ ibmcloud login -sso
```

Follow the on-screen instructions to log in.

Next, target the appropriate region for your instance:

```
$ ibmcloud target -r <REGION>
```

Then, list the existing monitoring instances in that region with the following command:

```
$ ibmcloud monitoring service-instances
```

Note: Note the service instance name that has “Platform Metrics” enabled.

Now create an authorization token for the API, by using a command like:

Important: The following steps work only on Bash.

```
$ AUTH_TOKEN=$(ibmcloud iam oauth-tokens | awk '{print $4}')
```

You can now retrieve the ID of your monitoring service instance for the API with the following command:

```
$ GUID=$(ibmcloud resource service-instance <instance_name_from_step_above> --output json | jq -r '.[].guid')
```

Step 2: Set up the notification channel

Configure a notification channel to be notified when an alert is triggered. To set up your notification channel, use the following command:

```
$ curl -X POST https://<region>.monitoring.cloud.ibm.com/api/notificationChannels -H "Authorization: Bearer $AUTH_TOKEN" -H "IBMInstanceID: $GUID" -H "content-type: application/json"  --data-raw '{"notificationChannel":{"id":null,"version":null,"teamId":"","name":"<notification_channel>","type":"EMAIL","enabled":true,"sendTestNotification":true,"options":{"notifyOnOk":true,"notifyOnResolve":true,"emailRecipients":["email@email.com"]}}}'
```

You see the following output:

```
$ {"notificationChannel":{"id":39209,"version":1,"customerId":34292,"enabled":true,"sendTestNotification":true,"createdOn":1678967870764,"modifiedOn":1678967870764,"name":"thursTest","options":{"notifyOnOk":true,"emailRecipients":["email@email.com"],"notifyOnResolve":true},"type":"EMAIL"}}%
```

You have now created a notification channel for your alerts.

⚠ **Important:** Make a note of the `id` field that is returned by the API call.

Step 3: Create the alert

Now that you have a notification channel, create your alert rule. Your alert rule describes the metric query to be monitored, the threshold value, and the action to take when the threshold is crossed. In this case, you're monitoring your `ibm_service_instance_name` to ensure that `max` utilization doesn't exceed 90%. If that happens, an alert is triggered and you're notified.

Note: This alert is triggered at 90% disk utilization. However, 50-70% disk utilization is preferred.

To retrieve the name of the database instance you want to set up the alert for, list all your database instances with a command like:

```
$ ibmcloud cdb ls
```

⚠ **Important:** Make sure to select a database in the same region as the monitoring instance.

You see output like the following:

```
$ Retrieving instances for all database types in all resource groups in all locations under IBM as ...
OK
Name                          Location   State
Databases for PostgreSQL-76   us-south   inactive
testelastic                   eu-gb      active
Databases for MySQL-9j        us-south   active
```

Now, use the name of your database to create the alert by using a command like:

```
$ curl --request POST \
  --url https://<region>.monitoring.cloud.ibm.com/api/alerts \
  --header "Authorization: Bearer  $AUTH_TOKEN" \
  --header 'Content-Type: application/json' \
  --header "IBMInstanceID: $GUID" \
  --data-raw  '{
 "alert":
  {
   "type": "MANUAL",
   "name": "Disk Alert",
   "description": "",
   "enabled": true,
   "severity": 1,
   "timespan": 60000000,
   "notificationChannelIds": [
    <id_from_previous_step>
   ],
   "filter": "ibm_service_instance_name in (\"<db_instance_name_from_previous_step>\")",
```

```
    "condition": "max(max(ibm_databases_for_elasticsearch_disk_used_percent)) > 0.9"
  }
}'
```

Note: This command takes the max disk utilization of any member available, regardless of the number of members.

Step 4: Check that your alert is created

An alert is created whenever you reach 90% of disk size. You receive the alert to the same email that you created in the notification channel. You can also use the following command to check current active alerts:

```
$ ibmcloud monitoring alert list --name <monitoring instance name>
```

You now receive an alert whenever your Databases for Elasticsearch instance disk utilization exceeds 90%, so you can act before the disk is too full.

Next steps

To modify your alert or find out more about Monitoring, see Getting started withIBM Cloud Monitoring.

Scaling resources

If you receive an alert that your disk utilization exceeds 90%, scale your disk so that you do not exceed 50-70% usage. Manually manage your service's resources or autoscale.

Service	Managing resources	Autoscaling
IBM Cloud® Databases for MongoDB	Scaling Disk, RAM, and CPU	Autoscaling
IBM Cloud® Databases for PostgreSQL	Scaling Disk, RAM, and CPU	Autoscaling
IBM Cloud® Databases for Redis	Scaling Disk, RAM, and CPU	Autoscaling
IBM Cloud® Databases for MySQL	Scaling Disk, RAM, and CPU	Autoscaling
IBM Cloud® Messages for RabbitMQ	Scaling Disk, RAM, and CPU	Autoscaling

Scaling Resources

Cloud Databases service metrics

This tutorial uses IBM Cloud® Databases for Elasticsearch. However, the same process applies to other Cloud Databases services:

- Databases for MongoDB
- Databases for PostgreSQL
- Databases for Redis
- Databases for MySQL
- Messages for RabbitMQ

Cloud Databases through the CLI

The Cloud Databases Command Line Interface (CLI) plug-in offers extra methods of accessing the capabilities of the Cloud Databases services.

The IBM Cloud CLI

The IBM Cloud® CLI provides commands for managing resources in IBM Cloud. When you install the standalone IBM Cloud CLI, you get only the CLI itself without any recommended plug-ins or tools. For more information, see Installing the stand-alone IBM Cloud CLI.

Installing the Cloud Databases CLI plug-in

After you install the IBM Cloud CLI, log in and install the Cloud Databases CLI plug-in, using a command like:

```
$ ibmcloud plugin install cloud-databases
```

For a list of commands and usage information, use a command like:

```
$ ibmcloud cdb help
```

On its own, the `ibmcloud cdb help` command displays the available top-level commands. When followed by another command, it displays specific help for that command.

```
$ ibmcloud cdb help [<command>]
```

Cloud Databases service-specific CLIs

Service	CLI client
Databases for MongoDB	The mongo Shell
Databases for Elasticsearch	
Databases for Redis	The Redis CLI
Databases for PostgreSQL	psql
Databases for MySQL	mysql
Messages for RabbitMQ	

Cloud Databases service-specific CLI clients

Versioning policy

When you provision a Cloud Databases instance, you can choose from the versions currently available on IBM Cloud®. Find the latest versions from the catalog pages, the Cloud Databases CLI plug-in, or the Cloud Databases API.

Major versions defined

Service	Cloud Databases versioning schema	Next known end of life version and date	Preferred major version	End of life procedure [1]
Databases for MongoDB	Cloud Databases major versions are the first two numbers in a `major.x.patch` version number. In cases where `x` is even, it is a stable release suitable for production. Even `x` versions are the only ones available on Cloud Databases.	v7, 25 Aug 2027	v8.0	Automatically upgraded in place to next Major version
Databases for Elasticsearch	Cloud Databases major versions are the first two numbers in a `release.version`.maintenance version number.	v8.7, v8.10, v8.12, v8.15, 30 June 2026	v8.19	Automatically upgraded in-place to next major version.
Databases for Redis	Cloud Databases major versions are the first number in a `major.minor.patch` version number.	v7.2, 19 August 2026	v8.2	Automatically upgraded in place to next Major version
Databases for PostgreSQL	Cloud Databases major version is defined by the first number in the version number.	v14, 21 October 2026	v18	Automatically upgraded in place to next major version, Customer-initiated in-place upgrade from v14 to v15 supported
Databases for MySQL	Cloud Databases major versions are the first two numbers in a `major.x.patch` version number.	v8.0, 29 July 2026	v8.0, v8.4 (Preview)	Backup taken and access removed
Messages for RabbitMQ	Cloud Databases Major versions are the first two numbers in a `major.x.patch` version number.	v3.13, 20 May 2026, v4.1, 12 August 2026	v4.2	Backup taken and access removed until v3.13, Automatically upgraded in place to next Major version starting v4.x

Major versions for Cloud Databases

End of life procedure

This approach is not recommended for the following reasons:

- We provide no SLAs for this type of forced upgrade.
- Data loss may occur.
- Applications may experience downtime.
- Applications may stop working if they have any incompatibilities with the new database version.
- You cannot control the timing of the forced upgrade of your instances.
- There is no rollback process for forced upgrades.
- We strongly recommend upgrading Cloud Databases instances to the latest version available as soon as possible after that version is made available.

Additional information on upgrade methods for each database type:

- Upgrading Databases for MongoDB major versions
- Upgrading Databases for Elasticsearch major versions
- Upgrading Databases for Redis major versions
- Upgrading Databases for PostgreSQL major versions
- Upgrading Databases for MySQL major versions
- Upgrading Messages for RabbitMQ major versions)

Subscribe for version updates

Availability of a new major database version in IBM Cloud is communicated via the release notes and IBM Cloud status page. Set up IBM Cloud status notifications, as described in the documentation, in order to receive a notification when new release notes are published.

Major version end of life procedures

End of life dates for major database versions in Cloud Databases are determined after considering two primary factors.

1. The date when the open-source community or vendor that provides the database stops maintaining that version.
2. Industry best practices for security that generally prohibit the use of software that is no longer maintained because bugs and security vulnerabilities are unlikely to be addressed in such a version.

Because the frequency of major releases and the maintenance lifecycle policies associated with each database offered in the IBM Cloud portfolio is different, the time between general availability of a major release in IBM Cloud and the end of life for that version in IBM Cloud varies across different databases and over time.

When the IBM Cloud end of life date for a major version is defined, a notification is provided via the IBM Cloud status announcements page. During the time between notification and the end of life date for a major version, you are strongly advised to initiate an upgrade to the most recent major version.

At the end of life date, any database instances that remain on the deprecated major version are handled as described in the end of life procedure column in Table 1. If the end of life procedure includes taking a backup of the instance, the backup is available to be restored into a new supported version for 30 days, after which the backup is deleted.

Requests to re-enable disabled formations of end-of-life versions are not accommodated.

Note: End of life procedures and related actions happen over several days following the end of life date. We try, but cannot guarantee, to complete these actions outside of business hours in the local region. If you want more control over the upgrade process of your instance, we recommend that you upgrade before the EOL date of your version.

Version tags

Version tag	Description
Preferred	The recommended and default version for all new instances. It's the most stable, up-to-date version from both an instance-level and service-level perspective.
Preview	A preview version is released for a limited time to try available functions. Often it is the newest available version available from the project maintainers in preparation for making it the "Preferred" version. While deployable, preview versions are not suitable for production, as they are excluded from service-level agreements and support. Also, a preview version isn't guaranteed to become a production-level release. IBM reserves the right to ask a customer to delete an instance that uses a preview version.
Deprecated	Old versions and versions near their end of life dates are marked as "Deprecated". Provisions and restores of instances that run a deprecated version are still available and instances that run a deprecated version continue to be supported. However, you are encouraged to upgrade to the new "Preferred" version as deprecated versions are eventually removed from IBM Cloud and are no longer provisionable, restorable, or supported.
Untagged	Untagged versions are fully supported and deployable versions. They are usually slightly older than the current preferred version, but they are still supported by the project maintainers. They continue to be supported on Cloud Databases instances until their deprecation is announced.
Hidden	A hidden version cannot be provisioned. Existing instances that are using a version marked as hidden are still able to be restored to the hidden version.

Cloud Databases version tags

Minor versions

IBM Cloud is committed to providing secure, up-to-date versions of services. As updates are released by project maintainers, they are tested, evaluated, and released to Cloud Databases instances. Your instance's minor version and patch updates are handled automatically and are not user configurable.

Major versioning end of life notification

The ability to provide advance notification of the end of life date for major database versions to IBM Cloud Database users is limited by the advance notice provided by the associated open-source community or vendor with respect to the date that maintenance of a version will end.

For those databases where the open-source community or vendor provides advance notice of the end of maintenance date for major versions, multiple notifications will be sent to inform users of upcoming end of life dates. You can typically expect:

1. A Cloud status page announcement, for example: End of support notices.
2. An announcement in your service's Release Notes, for example: IBM Cloud® Databases for PostgreSQL version 12 end of life on January 22, 2025.
3. A notification by email if account notification have been correctly configured to include email addresses. This email contains a *Notifications* link that takes you to a Notifications Management page. **Make sure that these announcements are not being caught by your email service's spam filter.** For more information, see Setting up distribution lists for IBM Cloud notifications){:external} and Setting email preferences for notifications.

Ensure that your account is enabled to receive notifications and announcements. You must enable receipt of both platform and resource updates.

- Turn on major and minor toggle under the **Platform tab > Announcements > Major and minor** .
- Turn on service updates under the **Resource tab > Resource activity > Service updates** .

Customers are also encouraged to proactively check the database version status of all IBM Cloud Database instances programmatically via either the CLI or API. For more information, see Programmatic methods for checking version status.

Database pecific information

[IBM Cloud Databases for Elasticsearch]

Elastic publishes the maintenance policy for Elasticsearch versions here. According to this policy, three versions are maintained by Elastic at any point in time, the most recent release (X.Y), the previous release (X.Y-1), and the last release for the previous major version (X-1.last, 8.19 for example). When a new release is made (X.Y+1), maintenance for release X.Y-1 ends immediately. Customers can choose between two approaches to upgrading the Elasticsearch versions they use. The first approach is to always upgrade to the latest Elasticsearch version soon after it is released, making the frequency of upgrades equal to the frequency of releases by Elastic. The second approach is to stay on the last release of the previous major version as long as it continues to be maintained by Elastic in order to reduce the frequency of required version upgrades during that period. An IBM Cloud notification will be sent shortly after each Elasticsearch major version release communicating that Elastic maintenance has ended for an additional major version and that this major version will reach end of life in IBM Cloud Databases in 5 weeks.

Programmatic methods for checking version status

On the CLI the following Cloud Databases `deployables-show` command shows deployable service types, specifically the available versions and their `preferred` or `stable` status.

```
$ ibmcloud cdb deployables-show [--stable] [--preferred] [--json]
```

Check the status of a major version by reviewing the output of the `deployable` command, specifically *Status* and *Preferred*. The following output example shows that version 7 is the `Preferred` version and version 6 *Status* is `deprecated`.

```
$ Service Type:  mongodb
Version   Status      Preferred
7         stable      true
6         deprecated  false
```

On the Cloud Databases API the `deployables` endpoint returns all deployable services. Use the `version` parameter to return the version number.

```
$ GET /v5/ibm/deployables
```

Major versions and Terraform

⚠ **Important:** Note that you cannot currently upgrade to a new major version using Terraform. Changing the version number on a Terraform script could lead to your data being destroyed. The recommended method of version upgrade is restoring a backup into a new deployment with the latest

version. For more information, see Restoring a backup.

---

1. This column describes the actions that will be taken by the IBM Cloud® team on database instances that have not been upgraded to a new version prior to the version EoL date. This approach is not recommended. For more information, see End of life procedure. ↩

Getting to production for Cloud Databases

Prework

- ☐ To ensure cloud-native alignment, complete your data modeling and architectural reviews. For help with data modeling and architecture, contact the IBM Garage.
- ☐ Determine the best method for your initial setup, including Terraform, API, CLI, or UI methods.
- ☐ Determine the hosting model and therefore the single- or multi-tenancy of your database.
- ☐ To manage your database's encryption key for data-at-rest, you must Bring your own encryption key (BYOK) when creating your database. This setup cannot be changed after your instance is provisioned.
- ☐ Make sure that IAM access policies and resource groups are set up correctly for your business protocols.
- ☐ Ensure that your account is VRF-enabled.
- ☐ Understand your database's high availability model. This is covered in the "High-Availability" section of each database's documentation.
- ☐ To ensure you receive messages, enroll in IBM Cloud notifications, specifically the "Resource Activity" notification. We directly notify you when your database version is approaching end of life. In addition, you can bookmark our Database version lifecycle policy, which is kept up to date with end of life dates and resources for all databases.

Sample "Getting to production" checklist

- ☐ Create a database with the required hosting model, disk, RAM, and virtual CPUs. While these scaling parameters can be changed after the initial provisioning, disks *cannot be scaled down*.
- ☐ If you would like hypervisor level isolation, or if you want to guarantee a number of vCPUs, ensure that you provision Isolated Compute instances.
- ☐ Add users. See the related documentation for your Cloud Databases instance.
- ☐ Change the `Admin` Password.
- ☐ Set up auto scaling policies, if appropriate. Default auto scaling logic is the suggested baseline. Tune these parameters to your budget and use case. The recommended disk space reflects the minimum amounts that are needed, but note that disk capacity cannot be scaled down without a backup and restore. RAM and virtual CPUs (vCPUs) can scale up and down. Memory auto scaling works based on disk I/O usage to optimize page cache performance. Databases will not auto-scale when in-use memory approaches 100%; this is often the desired state.

  For more information, see your Cloud Databases instance.
- ☐ Set up monitoring with IBM Cloud® Monitoring, IBM Cloud® Activity Tracker Event Routing, and IBM® Cloud Logs. At minimum, set alerts on:
  - IBM Cloud Monitoring - when disk usage is greater than 80% of provisioned capacity (we encourage you to use auto scaling for disk capacity). We also encourage you to use, understand, and alert on all provided metrics like disk I/O or CPU usage.
  - IBM Cloud Activity Tracker Event Routing audit events for control plane actions, such as failed backups, IP allowlist updates, and auto scaling.
  - IBM Cloud Logs - any particular database-specific logs you want to be notified about, such as slow query logs.
  - If available, turn on granular in-database auditing (only available for Databases for PostgreSQL and Databases for MongoDB Enterprise Edition).
- ☐ Set up context-based restrictions, which give account owners and administrators the ability to define and enforce access restrictions for IBM Cloud® resources based on the context of access requests. For more information, see Protecting Cloud Databases resources with context-based restrictions.
- ☐ Set Private endpoints. You might also choose to disable public endpoints (highly recommended if no connection is expected from outside IBM Cloud®).
- ☐ Make sure that your application uses TLS for connecting to the database. Insecure connections to Cloud Databases are not allowed.
- ☐ Thoroughly load test, and then load test again.
- ☐ Validate the application's reconnect logic. For some applications retry is not enough and you must reconnect. Review the article, "Unresponsive Redis Service" for an example of implementation on IBM Cloud® Databases for Redis.
- ☐ Set up development and testing environments as separate instances, then work through this checklist again. Depending on your requirements, you might not want to use Isolated Compute for these test environments. Not using Isolated Compute helps to keep costs lower.
- ☐ Complete Disaster recovery testing. Test restoring your application to a different IBM Cloud region. Ensure you are able to connect to a "restored" database with new connection details.
  - Understand your Recovery Point Objective (RPO) and Recovery Time Objective (RTO) requirements and ensure that you can meet them with

your database's configuration.

Tip: To resolve a database's UUID, use the command `ibmcloud resource search <UUID>`. Other useful CLI commands are: `ibmcloud cdb about` and `ibmcloud cdb cxn`. For more information, see the CLI plug-in reference documentation.

Shared responsibilities for Cloud Databases

Learn about the management responsibilities and terms and conditions that you have when you use IBM Cloud® Cloud Databases. For a high-level view of the service types in IBM Cloud and the breakdown of responsibilities between the client and IBM for each type, see Shared responsibilities for IBM Cloud offerings.

Review the following sections for the specific responsibilities for you and for IBM when you use Cloud Databases. For the overall terms of use, see IBM Cloud Terms and notices.

Incident and operations management

Task	IBM responsibilities	Your responsibilities
Monitoring	Cloud Databases is responsible for hosting monitoring and health services.	The Client is responsible for integrating with the IBM Cloud® Monitoring, IBM Cloud® Activity Tracker Event Routing, or IBM® Cloud Logs.
High Availability	Cloud Databases is responsible for deploying databases across availability zones in a Multi-Zone Region (MZR), or across hosts in a single-campus multizone region, and storing backups in cross-region Cloud Object Store instances. Cloud Databases provides replication, fail-over features, and infrastructure maintenance or updates. High availability varies based on each database type, refer to database-specific documentation for details.	The Client is responsible for designing application logic to retry connections caused by temporary connection failures (during regular database maintenance and updates).
Database performance	Cloud Databases is responsible for hosting and maintaining database infrastructure.	The Client is responsible for the data model and performance, including tuning the data model, queries, and scaling the database as necessary for application needs.
Operating System	Cloud Databases is responsible for hosting and maintaining database Operating System infrastructure.	The Client is not responsible for, nor has access to, Operating System level activities.

Responsibilities for incident and operations

Change management

Task	IBM responsibilities	Your responsibilities
Scaling	Cloud Databases is responsible for scaling infrastructure to meet client requests.	The Client is responsible for choosing, monitoring, and scaling disk, memory, and CPU core allocation for their deployments by using the UI or API. If a database deployment runs out of disk space, it might go down and must be restored from backup.
Version management	Cloud Databases is responsible for maintaining minor versions and patches as described in the version lifecycle policy. Cloud Databases is also responsible for providing availability and tools for database major version upgrades.	The Client is responsible for running major database version upgrades. The Client is also responsible for monitoring for EOL announcements and moving off EOL versions before the end of support date that is announced by Cloud Databases also described in the version lifecycle policy.

Responsibilities for change management

Security and regulation compliance

Task	IBM responsibilities	Your responsibilities
Encryption	Cloud Databases is responsible for the encryption of data on disk, in motion, and in backups.	The Client is responsible for choosing and managing appropriate additional security features. If using Key Protect and Bring Your Own Key (BYOK), the client is responsible for managing Key Protect authorizations and keys.

Security	Cloud Databases is responsible for ensuring the security of data on disk and data in motion within our infrastructure.	The Client is responsible for managing IBM Cloud passwords and database passwords, and keeping passwords secure. The Client is also responsible for configuring appropriate network security or isolation (for example, IP allowlists or private endpoints).
Compliance	Cloud Databases is responsible for ensuring adherence, auditing, and certification of compliances listed at the IBM Cloud compliance page.	The Client is responsible for the storing, processing, and transmission of their data. More information on these specific responsibilities can be found in each of the Cloud Databases offerings' specific Security Compliance documentation.

Responsibilities for security and regulation compliance

Disaster recovery

Note: For new hosting models, PITR is currently available through the CLI, API, and Terraform.

Task	IBM responsibilities	Your responsibilities
Backups and restore	Cloud Databases is responsible for automatic daily backups, as well as monitoring the state of client backups.	The Client is responsible for restoration, timeliness, validity of backups, and alerting of failed backups via IBM Cloud® Activity Tracker Event Routing. For more information, see Managing Cloud Databases backups.
Read-only replicas (*Databases for PostgreSQL and Databases for MySQL ONLY*)	Databases for PostgreSQL and Databases for MySQL are responsible for providing the capability of deploying read-only replicas across regions (except for replicating data into or outside of eu-de).	The Client is responsible for provisioning, configuring, monitoring, and promoting read-only replicas.

Responsibilities for disaster recovery

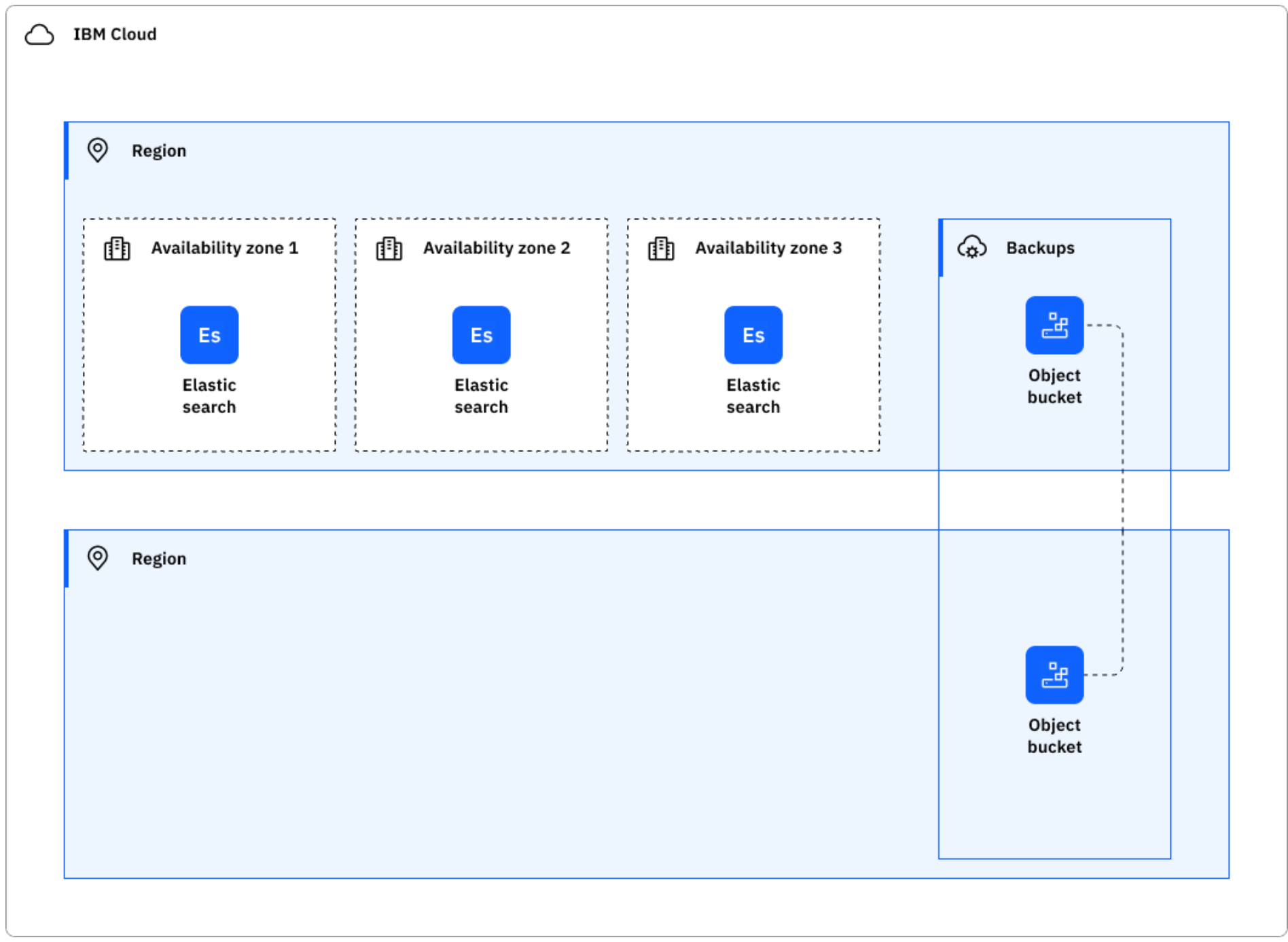
Understanding high availability and disaster recovery for Databases for Elasticsearch

High availability (HA) is the ability for a service to remain operational and accessible in the presence of unexpected failures. *Disaster recovery* is the process of recovering the service instance to a working state.

Databases for Elasticsearch is a regional service that fulfills the defined Service Level Objectives (SLO) with the Standard plan.

For more information, see Service Level Agreement (SLA). For more information about the available IBM Cloud regions and data centers for Databases for Elasticsearch, see Service and infrastructure availability by location.

High availability architecture



Elasticsearch high availability architecture

Databases for Elasticsearch provides replication, failover, and high-availability features to protect your databases and data from infrastructure maintenance, upgrades, and some failures. Deployments contain a cluster with three data members. Elasticsearch uses indexes to store data and each index has a primary shard and a replica shard. Elasticsearch uses a data replication model based on the primary-backup model. The primary shard serves as the main entry point for indexing operations, while the other copies are known as replica shards. Asynchronous replication is employed to keep the replica shards up to date. Elasticsearch ensures cluster state stability and facilitates failover. If the master node becomes unavailable, a new master is elected and the replica shards on the new master get promoted to master. The leader and replica shards are geographically distributed across different zones within the cluster to mitigate the risk of simultaneous failures.

You can extend high availability further by adding more replicas to the indexes however, this comes with additional storage costs that should be taken into consideration.

Review the Elasticsearch documentation on replication techniques to understand the constraints and tradeoffs that are associated with the asynchronous replication strategy that is deployed by default.

High availability features

Databases for Elasticsearch supports the following high availability features:

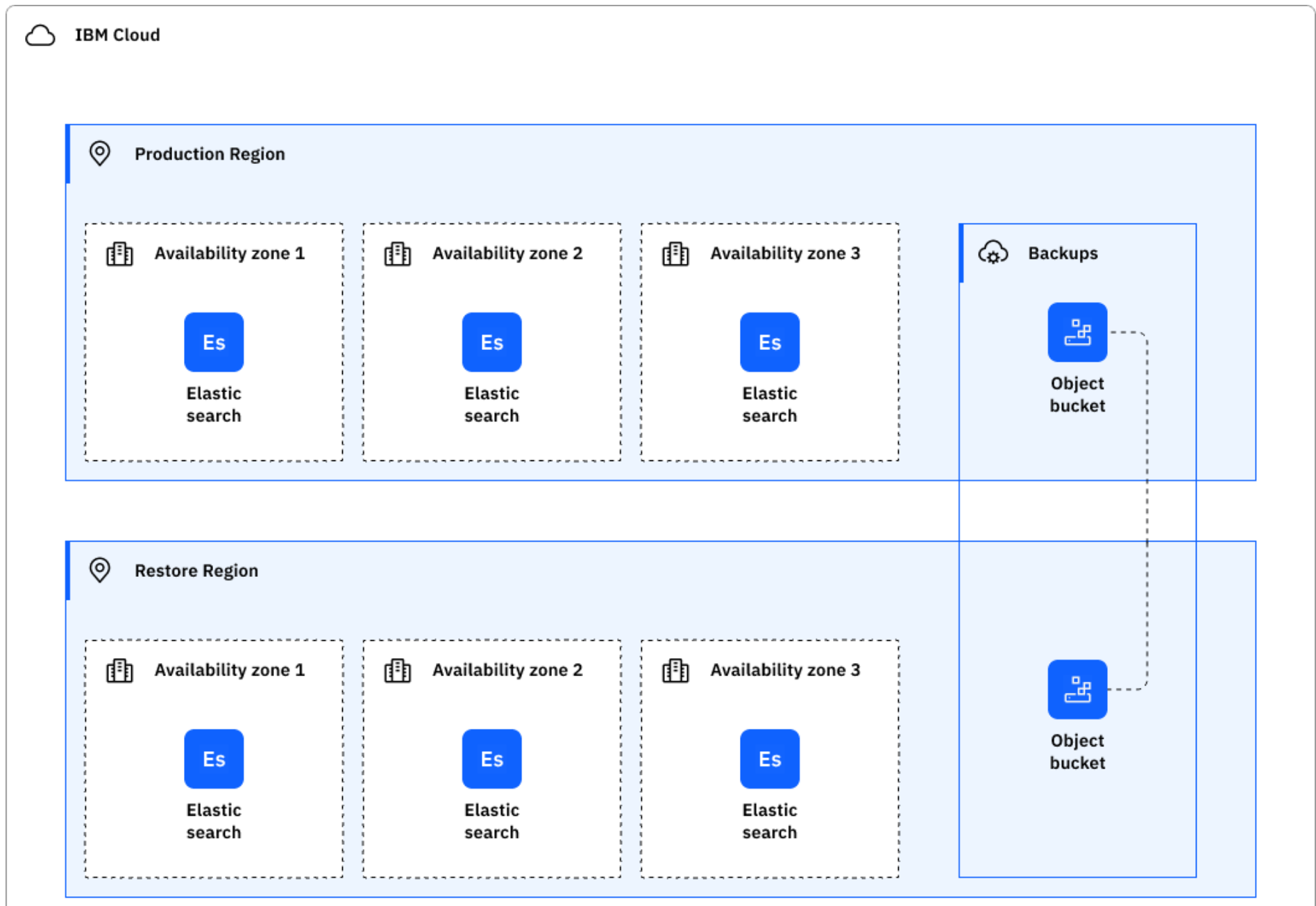
Feature	Description	Consideration

Automatic failover	Standard on all clusters and resilient against a zone or single member failure.
Member count	Minimum 3 members. Default is a Standard three deployment.
Horizontal scaling	It is possible to scale your IBM Cloud Databases for Elasticsearch deployment horizontally by adding more Elasticsearch nodes (also referred to as members). Adding more nodes increases capacity and reliability.

High availability features

Disaster recovery architecture

Disaster recovery features



Elasticsearch disaster recovery architecture

Databases for Elasticsearch supports the following disaster recovery features:

Feature	Description	Consideration
Backup restore	Create a database from previously created backup. For more information, see Managing Cloud Databases backups.	New connection strings for the restored database must be referenced throughout the workload.
Horizontal scaling	It is possible to scale your IBM Cloud Databases for Elasticsearch deployment horizontally by adding more Elasticsearch nodes (also referred to as members). Adding more nodes will make your database more resilient to multiple zone failures.	
Snapshots	You can store snapshots in a snapshot repository that is accessible from multiple regions. However, snapshots are not real-time and are typically used for backup and restore purposes rather than immediate failover. In case of a failure, you must restore from a snapshot manually.	

Planning for DR

The disaster recovery steps must be practiced regularly. As you build your plan, consider the following failure scenarios and resolutions.

Failure	Resolution
Hardware failure (single point)	IBM provides a database that is resilient from a single point of hardware failure within a zone - no configuration is required.
Zone failure	Automatic failover. The database members are distributed between zones. Configuring three members provides additional resiliency to multiple zone failures.
Data corruption	Backup restore. Use the restored database in production or for source data to correct the corruption in the restored database.
Regional failure	Backup restore. Use the restored database in production.

Failure scenarios and resolutions

Application-level high availability

Applications that communicate over networks and cloud services are subject to transient connection failures. Design your applications to retry connections when errors are caused by a temporary loss in connectivity to your deployment or to IBM Cloud.

As Databases for Elasticsearch is a managed service, regular updates and database maintenance occur as part of normal operations. This can occasionally cause short intervals where your database is unavailable. It can also cause the database to trigger a graceful failover, retry, and reconnect. It takes a short time for the database to determine which member is a replica and which is the leader, so you might also see a short connection interruption. Failovers generally take less than 30 seconds.

Your applications must be designed to handle temporary interruptions to the database, implement error handling for failed database commands, and implement retry logic to recover from a temporary interruption.

Several minutes of database unavailability or connection interruption are not expected. Open a support case with details if you have periods longer than a minute with no connectivity so we can investigate.

Connection limits

Databases for Elasticsearch does not have any restrictions on the number of connections you open to the database as it uses REST API to interact with the database. Elasticsearch's default settings provide a good out-of-box experience for basic operations, such as full text search, highlighting, aggregations, and indexing.

If you want more performance out of our database, see the Optimize Elasticsearch page for more information.

Your responsibilities for HA and DR

The following information can help you create and continuously practice your plan for HA and DR.

When restoring a database from backups or using point-in-time restore, a new database is created with new connection strings. Existing workloads and processes must be adjusted to consume the new connection strings.

A recovered database may also need the same customer-created dependencies of the disaster database. Ensure that these and other services exist in the recovered region:

- IBM® Key Protect for IBM Cloud®
- Hyper Protect Crypto Services

Remember that deleting a database also deletes its associated backups. However, deleted databases may be recoverable within a limited timeframe. For more information, see the Backup FAQ documentation for specific details on database recovery procedures.

It is not possible to copy backups off the IBM Cloud, so consider using the database-specific tools for additional backups. It may be required to recover from malicious database deletion followed by a reclamation-delete of a database. Careful management of IAM access to databases can help reduce exposure to this problem.

The following checklist associated with each feature can help you create and practice your plan.

- Backup restore
  - Verify backups are available at the desired frequency to meet RPO requirements. Managing Cloud Databases backups documents backup frequency.
  - There are some restrictions on database restore regions - verify your restore goals can be achieved by reading managing Cloud Databases backups.
  - Verify the retention period of the backups meet your requirements.
  - Schedule test restores regularly to verify that the actual restored times meet the defined RTO. Remember that database size significantly impacts restore time. Please consider strategies to minimize restore times, such as breaking down large databases into smaller, more manageable units and purging unused data.
  - Verify the Key Protect service.

To find out more about responsibility ownership between the customer and IBM Cloud for using Databases for Elasticsearch, see Shared responsibilities for Cloud Databases.

Stay informed: IBM notifications

Updates affecting customer workloads are communicated through IBM Cloud notifications. To stay informed about planned maintenance, announcements, and release notes related to this service, refer to the Monitoring notifications and status page. In addition, regularly review the Version policy page for the latest updates on End-of-Life versions and dates.

Additional guidance

- Understanding high availability for Cloud Databases
- Understanding business continuity and disaster recovery for Cloud Databases
- Horizontal scaling

Understanding data portability for Databases for Elasticsearch

Data Portability involves a set of tools, and procedures that enable customers to export the digital artifacts that would be needed to implement similar workload and data processing on different service providers or on-prem software. It includes procedures for copying and storing the service customer's content, including the related configuration used by the service to store and process the data, on customer's own location.

Responsibilities

IBM Cloud® services provide interfaces and instructions to guide the customer to copy and store the service customer content, including the related configuration, on their own selected location.

The customer then is responsible for the use of the exported data and configuration for the purpose of data portability to other infrastructures. This can involve the following:

- Planning and execution for setting up alternate infrastructure on on different cloud providers or on-prem software that provide similar capabilities to the IBM services.
- Planning and execution for the porting of the required application code on the alternate infrastructure, including the adaptation of customer's application code, and deployment automation.
- Conversion of the exported data and configuration to format required by the alternate infrastructure and adapted applications.

For more information about your responsibilities when using Databases for Elasticsearch, see Shared responsibilities for Databases for Elasticsearch.

Note: If you're deploying Databases for Elasticsearch by using the Cloud automation for Databases for Elasticsearch deployable architecture, you can additionally refer to Understanding your responsibilities when you use IBM deployable architectures.

Data export procedures

Databases for Elasticsearch provides mechanisms to export your content that has been uploaded, stored, and processed using the service.

You can take snapshots of your data using the procedure described in Creating snapshots.

Exported data formats

The format of the data exported using the snapshots is a binary file with Elasticsearch propriatory format and can be used only with Elasticsearch instances. Exporting to other data formats is supported using Logstash.

Data ownership

All exported data are classified as Customer content and therefore apply to them the full customer ownership and licensing rights, as stated in IBM Cloud Service Agreement.

Troubleshooting

Why can’t I connect to my Elasticsearch deployment?

If you encounter errors when connecting to your IBM Cloud® Databases for Elasticsearch deployment, review these common causes and resolutions.

You receive an error message or fail to connect to a Databases for Elasticsearch deployment. If reviewing application logs, you might see errors that mention intermittent connection timeouts or unable to connect.

Review the following information to troubleshoot and resolve common connectivity problems:

- An unsecured connection is a common cause of connectivity errors. All Databases for Elasticsearch connections use TLS/SSL encryption; Databases for Elasticsearch rejects unsecured connections. To avoid errors, make sure you configured a secure connection. Refer to Getting started for an example of a secure connection.
- If you are using a private endpoint, make sure that you specify connection strings that contain the private endpoint and that you followed the steps in Connecting through Private Endpoints.
- If your application log captures a short connection interruption, that behavior is expected as a normal part of operations for this managed service. You want to design your applications to retry connections when errors are caused by a temporary loss in connectivity to your deployment or to IBM Cloud. However, if you experience several minutes of connection interruption check the Cloud Status for the service. For more information, see Application-level high availability.

Why did Terraform replace my instance?

Your Terraform script deleted your Cloud Databases instance. Why did this happen, and what can I do?

What’s happening

You ran your Terraform script and now your instance has been deleted. You may have seen an output that looks like:

```
$ ibm_database.database_instance must be replaced
```

Why it’s happening

Terraform can force a new resource when certain attributes are modified. Altering certain attributes recreates your instance, for example: `resource_group_id`, `service plan`, `version`, `key_protect_instance`, `key_protect_key`, and `backup_encryption_key_crn`.

How to fix it

Before executing a Terraform script on an existing instance, use the `terraform plan` command to compare the current infrastructure state with the desired state defined in your Terraform files. Any alteration to the `resource_group_id`, `service plan`, `version`, `key_protect_instance`, `key_protect_key`, `backup_encryption_key_crn` attributes recreates your instance. For a list of current argument references with the `Forces new resource` specification, see the ibm_database Terraform Registry.

FAQs

IBM Cloud® platform and service FAQ

The IBM Cloud® Status page is the central place to find details about major incidents that affect the IBM Cloud® platform and services. Other incidents, planned maintenance, announcements, release notes, and security bulletins are posted on the Notifications page, where you can easily view them.

For more information, see Viewing cloud status.

Subscribing to an RSS feed

Stay up to date with events by subscribing to the RSS feed for the Status page. For more information, see Subscribing to an RSS feed.

Checking incident reports

The IBM Cloud® Incident reports page provides a way for you to review technical details of major outages for IBM Cloud services. For more information, see Checking incident reports.

Resource configuration FAQ

How do I retrieve the resource configuration for a Cloud Databases instance?

Retrieving resource configuration through the UI

After you have selected the specific IBM Cloud Databases instance from the Resource List, navigate to the **Resources** tab, which displays your current resource configuration.

Retrieving resource configuration through the CLI

The Cloud Databases CLI plug-in can retrieve your instance's current resource configuration, by using the `ibmcloud cdb deployment-groups` command. The `ibmcloud cdb deployment-groups` displays the scaling group values for a deployment's members. The scaling groups relate to Memory, CPU, and Disk. The default group is named "member". Use a command like:

```
$ ibmcloud cdb deployment-groups <INSTANCE_NAME_OR_CRN>
```

The command will return a value that looks like:

```
$ Group   member
Count   3
|
+   Memory
|   Allocation              3072mb
|   Allocation per member   1024mb
|   Minimum                 3072mb
|   Step Size               384mb
|   Adjustable              true
|
+   CPU
|   Allocation              0
|   Allocation per member   0
|   Minimum                 9
|   Step Size               3
|   Adjustable              true
|
+   Disk
|   Allocation              30720mb
|   Allocation per member   10240mb
|   Minimum                 30720mb
|   Step Size               3072mb
|   Adjustable              true
```

For more information, see ibmcloud cdb deployment-groups.

Availability zones FAQ

You can create a Cloud Databases instance on IBM Cloud in a multi-zone or single-zone region.

See the documentation for provisioning a specific service Cloud Databases instance:

- Provisioning IBM Cloud® Databases for PostgreSQL
- Provisioning IBM Cloud® Databases for MongoDB
- Provisioning IBM Cloud® Databases for Redis
- Provisioning IBM Cloud® Databases for Elasticsearch
- Provisioning IBM Cloud® Databases for MySQL
- Provisioning IBM Cloud® Messages for RabbitMQ

What is an availability zone?

IBM Cloud® has a resilient global network of locations to host your highly available cloud workload. You can create resources in different locations, such as a region or data center, but with the same billing and usage view. You can also deploy your apps to the location that is nearest to your customers to achieve low application latency. IBM Cloud provides three tiers of regions: *multizone regions*, *single-campus multizone regions*, and *data centers*.

For more information, see IBM Cloud® Region and data center locations for resource deployment.

Certificates FAQ

Cloud Databases certificates are authenticated by unique TLS certificates. To verify this, you can inspect the certificate that your service presents when opening a connection. The certificate contains the hostname for a single instance. Cloud Databases data plane clusters possess their own distinctive root certificate. When you're engaged in certificate validation procedures, exercise caution and select the correct root certificate for each instance. For more information, see Connecting an external application.

Connecting an external application documentation

Choose the appropriate service documentation for connecting an external application:

- Databases for PostgreSQL
- Databases for MongoDB
- Databases for Redis
- Databases for Elasticsearch
- Databases for MySQL
- Messages for RabbitMQ

Mutual TLS

Cloud Databases does not support mutual TLS for client connections. Presenting client certificates or configuring trusted root certificate authorities (CAs) for client certificates is not supported.

IBM Corporation
New Orchard Road
Armonk, NY 10504

Produced in the United States of America
2026-03-22