Cloud Search Service

Quick Start

Issue 01

Date 2024-04-26

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process.* For details about this process, visit the following web page:
https://www.huawei.com/en/psirt/vul-response-process
For vulnerability information, enterprise customers can visit the following web page:
https://securitybulletin.huawei.com/enterprise/en/security-advisory

Contents

1 Getting Started with Cloud Search Service

This section provides an example of how to use CSS, including creating indexes, importing data, and searching for data.

Scenario Description

A women's clothing brand builds an e-commerce website. It uses traditional databases to provide a product search function for users. However, due to an increase in the number of users and business growth, the traditional databases have slow response and low accuracy. To improve user experience and user retention, the e-commerce website plans to use Cloud Search Service (CSS) to provide the product search function for users.

Assume that the e-commerce website provides the following data:

```
{
"products":[
{"productName":"Latest elegant shirts in autumn 2017","size":"L"}
{"productName":"Latest elegant shirts in autumn 2017","size":"M"}
{"productName":"Latest elegant shirts in autumn 2017","size":"S"}
{"productName":"Latest jeans in spring 2018","size":"M"}
{"productName":"Latest jeans in spring 2018","size":"S"}
{"productName":"Latest casual pants in spring 2017","size":"L"}
{"productName":"Latest casual pants in spring 2017","size":"S"}
]
}
```

Procedure

- **Step 1: Creating a Cluster**
- **Step 2: Importing Data**
- **Step 3: Searching for Data**
- **(Optional) Step 4: Deleting Indexes**
- **(Optional) Step 5: Deleting the Cluster**

Preparations

You have registered with Huawei Cloud, performed real-name authentication, and ensure your account is not frozen or in arrears before using CSS.

If you already have a HUAWEI ID, skip this step. If you do not have one, perform the following operations to create an account:

1. Visit the **Huawei Cloud official website**.
2. In the upper right corner of the page, click **Register** and complete the registration as prompted.
3. Select the service agreement and click **Enable**.
4. Perform real-name authentication.
   - If your account is an individual account, see **Individual Real-Name Authentication**.
   - If your account is an enterprise account, see **Enterprise Real-Name Authentication**.

Step 1: Creating a Cluster

In this example, you need to create an Elasticsearch cluster named **Sample-ESCluster**. This cluster is used only for getting started with Elasticsearch. For this cluster, you are advised to select **ess.spec-4u8g** for **Node Specifications**, **High I/O** for **Node Storage Type**, and **40 GB** for **Node Storage Capacity**.

1. Log in to the CSS management console.
2. Click **Create Cluster** in the upper right corner. The **Create Cluster** page is displayed.
3. The billing mode can be **Pay-per-use** or **Yearly/Monthly**. In this example, select **Pay-per-use**. You are billed by actual duration of use, with a billing cycle of one hour. For example, 58 minutes of usage will be rounded up to an hour and billed.
4. Specify **Region** and **AZ**.

Table 1-1 Region and AZ parameters

Parameter	Description
Region	Select the region where the cluster is located from the drop-down list box.
AZ	Select AZs associated with the cluster region. A maximum of three AZs can be configured. In this example, select one AZ.

5. Configure basic cluster information.

Figure 1-1 Configuring cluster information

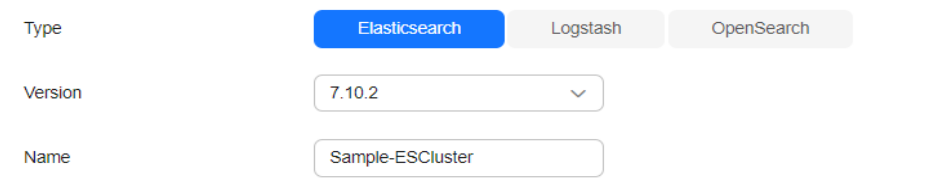


Table 1-2 Basic configuration parameters

Parameter	Description
Version	Select a cluster version from the drop-down list box. In this example, select the version **7.10.2**.
Name	User-defined cluster name. In this example, enter **Sample-ESCluster**.

6. Configure the cluster flavor.

Figure 1-2 Configuring the cluster flavors

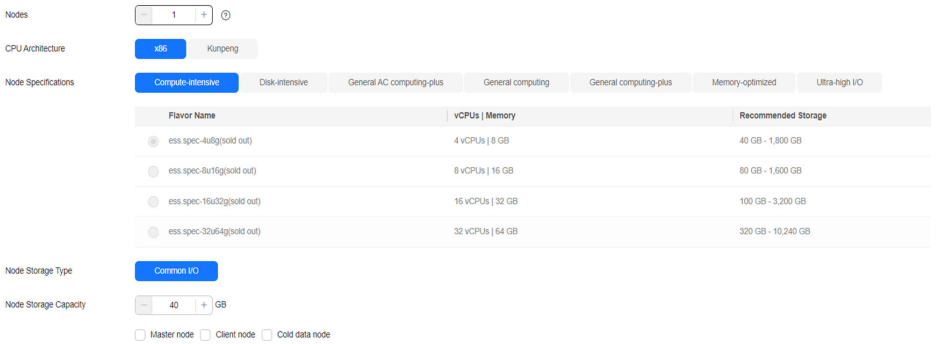


Table 1-3 Specification parameters

Parameter	Description
Nodes	Number of nodes in a cluster. Select a number from 1 to 32. In this example, select one node.
CPU Architecture	Currently, **x86** and **Kunpeng** are supported. The supported type is determined by the actual regional environment.
Node Specifications	Specifications of nodes in a cluster. In this example, select ess.spec-4u8g.
Node Storage Type	Select a storage type. In this example, select **High I/O**.
Node Storage Capacity	Storage space. Its value varies with node specifications. The node storage capacity must be a multiple of 20. In this example, select **40 GB**.
Master node	The master node manages all nodes in the cluster. If more than 20 nodes are required to store and analyze a large amount of data, you are advised to enable the master node to ensure cluster stability. In this scenario, the master node is not required.

Parameter	Description
Client node	The client node allows clients to access clusters and analyze data. If more than 20 nodes are required to store and analyze a large amount of data, you are advised to enable the client node to ensure cluster stability. In this example, the client node is not required.
Cold data node	The cold data node is used to store historical data, for which query responses can be returned in minutes. If you do not quire a quick query response, store historical data on cold data nodes to reduce costs. In this example, the cold data node is not required.

7. Set the enterprise project.

   When creating a CSS cluster, you can bind an enterprise project to the cluster if you have enabled the enterprise project function. You can select an enterprise project created by the current user from the drop-down list on the right or click **View Project Management** to go to the **Enterprise Project Management** console and create a new project or view existing projects.

8. Click **Next: Configure Network**. Configure the cluster network.

Figure 1-3 Configuring networking

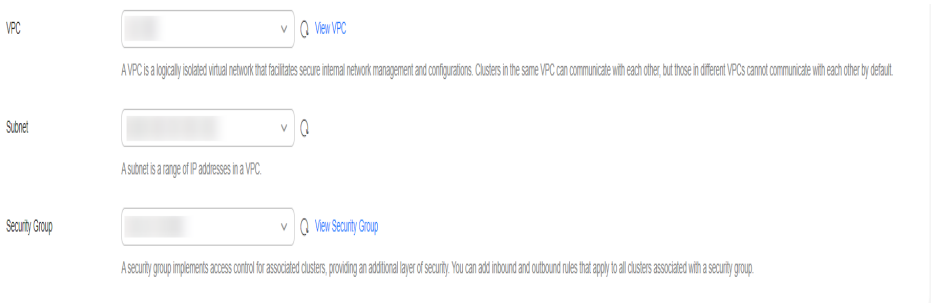


Table 1-4 Network configuration parameters

Parameter	Description
VPC	A VPC is a secure, isolated, and logical network environment. Select the target VPC. Click **View VPC** to enter the VPC management console and view the created VPC names and IDs. If no VPCs are available, create one. **NOTE** The VPC must contain CIDRs. Otherwise, cluster creation will fail. By default, a VPC will contain CIDRs.

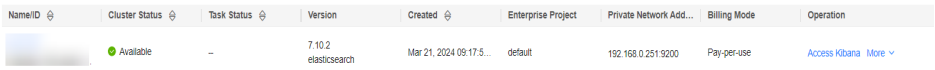
Parameter	Description
Subnet	A subnet provides dedicated network resources that are isolated from other networks, improving network security. Select the target subnet. You can access the VPC management console to view the names and IDs of the existing subnets in the VPC.
Security Group	A security group implements access control for ECSs that have the same security protection requirements in a VPC. Click **View Security Group** to learn security group details.
Security Mode	After the security mode is enabled, communication will be encrypted and authentication required for the cluster. In this example, disable the security mode.

9. Click **Next: Configure Advanced Settings**. Configure the automatic snapshot creation and other functions.

   This cluster is used only for getting started. Cluster snapshots and advanced functions are not required.

10. Click **Next: Confirm**. Check the configuration and click **Next** to create a cluster.
11. Click **Back to Cluster List** to switch to the **Clusters** page. The cluster you created is listed on the displayed page and its status is **Creating**. If the cluster is successfully created, its status will change to **Available**.

Figure 1-4 Creating a cluster



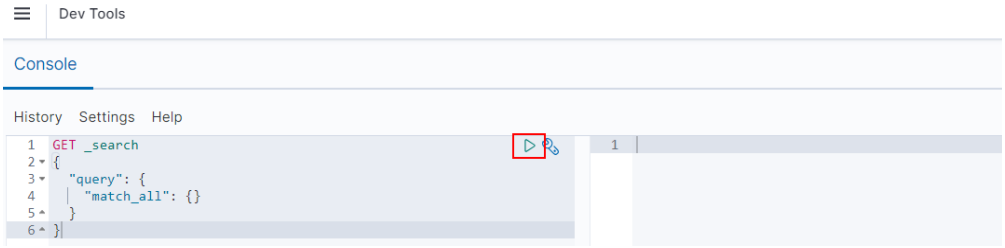
Step 2: Importing Data

CSS supports importing data to Elasticsearch using Cloud Data Migration (CDM), Logstash, Kibana, or APIs. Kibana is an open-source data analysis and visualization platform. You can use Kibana to search for and view data stored in Elasticsearch indexes and display data in charts and maps. By default, the Elasticsearch cluster of CSS provides the access channel to Kibana. The following procedure illustrates how to import data to Elasticsearch using Kibana.

1. On the cluster management page, select the created **Sample-ESCluster** cluster and click **Access Kibana** in the **Operation** column to access the Kibana console.
2. In the Kibana navigation pane on the left, choose **Dev Tools**.

   The text box on the left is the input box. The triangle icon in the upper right corner of the input box is the command execution button. The text box on the right area is the result output box.

Figure 1-5 Console page



3. On the **Console** page, run the following command to create index named **my_store**:

   Versions earlier than 7.*x*

   ```
   PUT /my_store
   {
     "settings": {
       "number_of_shards": 1
     },
     "mappings": {
       "products": {
         "properties": {
           "productName": {
             "type": "text",
             "analyzer": "ik_smart"
           },
           "size": {
             "type": "keyword"
           }
         }
       }
     }
   }
   ```

   Versions 7.*x* and later

   ```
   PUT /my_store
   {
     "settings": {
       "number_of_shards": 1
     },
     "mappings": {
       "properties": {
         "productName": {
           "type": "text",
           "analyzer": "ik_smart"
           },
           "size": {
             "type": "keyword"
           }
         }
       }
     }
   ```

   The command output is similar to the following:

   ```
   {
     "acknowledged" : true,
     "shards_acknowledged" : true,
     "index" : "my_store"
   }
   ```

4. On the **Console** page, run the following command to import data to index named **my_store**:

   Versions earlier than 7.*x*

   ```
   POST /my_store/products/_bulk
   {"index":{}}
   {"productName":"Latest elegant shirts in autumn 2017","size":"L"}
   ```



```
{"index":{}}
{"productName":"Latest elegant shirts in autumn 2017","size":"M"}
{"index":{}}
{"productName":"Latest elegant shirts in autumn 2017","size":"S"}
{"index":{}}
{"productName":"Latest jeans in spring 2018","size":"M"}
{"index":{}}
{"productName":"Latest jeans in spring 2018","size":"S"}
{"index":{}}
{"productName":"Latest casual pants in spring 2017","size":"L"}
{"index":{}}
{"productName":"Latest casual pants in spring 2017","size":"S"}
```

Versions 7.*x* and later

```
POST /my_store/_doc/_bulk
{"index":{}}
{"productName":"Latest elegant shirts in autumn 2017","size":"L"}
{"index":{}}
{"productName":"Latest elegant shirts in autumn 2017","size":"M"}
{"index":{}}
{"productName":"Latest elegant shirts in autumn 2017","size":"S"}
{"index":{}}
{"productName":"Latest jeans in spring 2018","size":"M"}
{"index":{}}
{"productName":"Latest jeans in spring 2018","size":"S"}
{"index":{}}
{"productName":"Latest casual pants in spring 2017","size":"L"}
{"index":{}}
{"productName":"Latest casual pants in spring 2017","size":"S"}
```

If the value of the **errors** field in the command output is **false**, the data is imported successfully.

Step 3: Searching for Data

- **Full-text search**

  If you access the e-commerce website and want to search for commodities whose names include "spring jeans", enter "spring jeans" to begin your search. The following example shows the command to be executed on Kibana and the command output.

  Run the following command:

  Versions earlier than 7.*x*

  ```
  GET /my_store/products/_search
  {
    "query": {"match": {
      "productName": "spring jeans"
    }}
  }
  ```

  Versions 7.*x* and later

  ```
  GET /my_store/_search
  {
    "query": {"match": {
      "productName": "spring jeans"
    }}
  }
  ```

  The command output is similar to the following:

  ```
  {
    "took" : 3,
    "timed_out" : false,
    "_shards" : {
      "total" : 1,
      "successful" : 1,
      "skipped" : 0,
  ```



```
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 4,
      "relation" : "eq"
    },
    "max_score" : 1.7965372,
    "hits" : [
      {
        "_index" : "my_store",
        "_type" : "_doc",
        "_id" : "9xf6VHIBfClt6SDjw7H5",
        "_score" : 1.7965372,
        "_source" : {
          "productName": "Latest jeans in spring 2018",
          "size" : "M"
        }
      },
      {
        "_index" : "my_store",
        "_type" : "_doc",
        "_id" : "-Bf6VHIBfClt6SDjw7H5",
        "_score" : 1.7965372,
        "_source" : {
          "productName": "Latest jeans in spring 2018",
          "size" : "S"
        }
      },
      {
        "_index" : "my_store",
        "_type" : "_doc",
        "_id" : "-Rf6VHIBfClt6SDjw7H5",
        "_score" : 0.5945667,
        "_source" : {
          "productName": "Latest casual pants in spring 2017",
          "size" : "L"
        }
      },
      {
        "_index" : "my_store",
        "_type" : "_doc",
        "_id" : "-hf6VHIBfClt6SDjw7H5",
        "_score" : 0.5945667,
        "_source" : {
          "productName": "Latest casual pants in spring 2017",
          "size" : "S"
        }
      }
    ]
  }
}
```


- Elasticsearch supports IK word segmentation. The preceding command segments "spring jeans" into "spring" and "jeans".
- Elasticsearch supports full-text search. The preceding command searches for the information about all commodities whose names include "spring" or "jeans".
- Unlike traditional databases, Elasticsearch can return results in milliseconds by using inverted indexes.
- Elasticsearch supports sorting by score. In the command output, information about the first two commodities contains both "spring" and "jeans", while that about the last two products contain only "spring". Therefore, the first two commodities rank prior to the last two due to high keyword match.

- **Aggregation result display**

  The e-commerce website provides the function of displaying aggregation results. For example, it classifies commodities corresponding to "spring" based on the size so that you can collect the number of products of different sizes. The following example shows the command to be executed on Kibana and the command output.

  Run the following command:

  Versions earlier than 7.*x*

  ```
  GET /my_store/products/_search
  {
  "query": {
  "match": { "productName": "spring" }
  },
  "size": 0,
  "aggs": {
  "sizes": {
  "terms": { "field": "size" }
  }
  }
  }
  ```

  Versions 7.*x* and later

  ```
  GET /my_store/_search
  {
  "query": {
  "match": { "productName": "spring" }
  },
  "size": 0,
  "aggs": {
  "sizes": {
  "terms": { "field": "size" }
  }
  }
  }
  ```

  The command output is similar to the following:

  Versions earlier than 7.*x*

  ```
  {
    "took" : 31,
    "timed_out" : false,
    "_shards" : {
      "total" : 1,
      "successful" : 1,
      "skipped" : 0,
      "failed" : 0
    },
    "hits" : {
      "total" : 4,
      "max_score" : 0.0,
      "hits" : [ ]
    },
    "aggregations" : {
      "sizes" : {
        "doc_count_error_upper_bound" : 0,
        "sum_other_doc_count" : 0,
        "buckets" : [
          {
            "key" : "S",
            "doc_count" : 2
          },
          {
            "key" : "L",
            "doc_count" : 1
          },
          {
  ```



```
        "key" : "M",
        "doc_count" : 1
      }
    ]
  }
 }
}
```

Versions 7.*x* and later

```
{
 "took" : 3,
 "timed_out" : false,
 "_shards" : {
  "total" : 1,
  "successful" : 1,
  "skipped" : 0,
  "failed" : 0
 },
 "hits" : {
  "total" : {
   "value" : 4,
   "relation" : "eq"
  },
  "max_score" : null,
  "hits" : [ ]
 },
 "aggregations" : {
  "sizes" : {
   "doc_count_error_upper_bound" : 0,
   "sum_other_doc_count" : 0,
   "buckets" : [
    {
     "key" : "S",
     "doc_count" : 2
    },
    {
     "key" : "L",
     "doc_count" : 1
    },
    {
     "key" : "M",
     "doc_count" : 1
    }
   ]
  }
 }
}
```

(Optional) Step 4: Deleting Indexes

If an index is no longer used, run the following command to delete the index to avoid resource waste:

```
DELETE /my_store
```

The command output is similar to the following:

```
{
 "acknowledged" : true
}
```

(Optional) Step 5: Deleting the Cluster

You can delete the cluster if you do not need it.

> **NOTE**
>
> After you delete a cluster, its data cannot be restored. Exercise caution when deleting a cluster.

1. Log in to the CSS management console. In the navigation pane on the left, choose **Clusters** > **Elasticsearch**.
2. Go to the cluster management page, locate the row that contains the target **Sample-ESCluster** cluster, and click **More** > **Delete** in the **Operation** column.
3. In the displayed dialog box, enter the name of the cluster to be deleted and click **OK**.

2 Getting Started with Common Practices

CSS provides you with best practices in different service scenarios and solution architectures, helping you select a proper practice solution based on your service scenario.

Table 2-1 Common practices

Scenario		Description
Migrating data	Migrating data from Elasticsearch	• **Migrating Data Through Backup and Restoration (from CSS Elasticsearch)** This practice describes how to migrate data between CSS Elasticsearch clusters by backing up and restoring cluster snapshots. • **Migrating Data Through Backup and Restoration (from Third-Party Elasticsearch)** This practice describes how to migrate data between custom ESs and Huawei Cloud ESs and between third-party ESs and Huawei Cloud ESs by backing up and restoring cluster snapshots
		Migrating the Entire Elasticsearch Database to CSS Huawei Cloud CDM is used to migrate clusters between different cloud services. This practice describes how to use CDM to migrate the entire local Elasticsearch database to CSS.

Scenario		Description
		Migrating Cluster Data Using Logstash Logstash: an official data cleaning tool provided by Elasticsearch. It is a part of the Elk ecosystem and provides powerful functions. It can migrate data between different data sources and Elasticsearch, and clean and process data.
	Migration from Kafka/MQ	In industries dealing with a large amount of data, such as IoT, news, public opinion analysis, and social networking, message middleware such as Kafka and MQ is used to balance traffic in peak and off-peak hours. The tools such as Flink and Logstash are then used to consume data, preprocess data, and import data to the search engine. This practice describes how to migrate clusters from Kafka/MQ.
	Migration from a Database	Elasticsearch supports full-text search and ad hoc queries. It is often used as a supplement to relational databases, such as MySQL and GaussDB(for MySQL), to improve the full-text search and high-concurrency ad hoc query capabilities of databases.
Accessing a cluster	**Accessing an Elasticsearch Cluster**	CSS clusters have built-in Kibana and Cerebro components. You can quickly access an Elasticsearch cluster through Kibana and Cerebro.
	Accessing a Cluster Using cURL Commands	If the CSS cluster and ECS are in the same VPC, you can run cURL commands on the ECS to directly access the Elasticsearch cluster. This method is mainly used to check whether the client that accesses the cluster can be connected to Elasticserach nodes.
	Accessing a cluster using Java	**Accessing a Cluster Through the Rest High Level Client** Elasticsearch provides SDK (Rest High Level Client) for connecting to a cluster. This client encapsulates Elasticsearch APIs. You only need to construct required structures to access the Elasticsearch cluster.

Scenario		Description
		Accessing a Cluster Through the Rest Low Level Client The high-level client is encapsulated based on the low-level client. If the method calls (such as .search and .bulk) in the high-level client cannot meet the requirements or has compatibility issues, you can use the low-level client. You can even use **HighLevelClient.getLowLevelClient()** to directly obtain a low-level client.
		Accessing the Cluster Through the Transport Client This practice describes how to use the native Transport Client of Elasticsearch to access to a cluster in non-security mode.
		Using Spring Boot to Access a Cluster This practice describes how to use Spring Boot to access a cluster.
	Accessing a Cluster Using Python	This practice describes how to access a CSS cluster using Python.
	Using ES-Hadoop to Read and Write Data in Elasticsearch Through Hive	This practice uses the ES-Hadoop of MRS as an example to describe how to connect to a CSS cluster. You can configure any other applications that need to use the Elasticsearch cluster. Ensure the network connection between the client and the Elasticsearch cluster is normal.
Optimizing cluster performance	**Optimizing Write Performance**	Before using a CSS cluster, you are advised to optimize the write performance of the cluster to improve efficiency.
	Optimizing Query Performance	Before using a CSS cluster, you are advised to optimize the query performance of the cluster to improve efficiency.
Managing the index lifecycle	**Configuring the Lifecycle to Automate Index Rollover**	Time series data is continuously written and increases index size. You can configure the lifecycle to periodically roll over to new indexes and delete old indexes.
	Automatic Storage and Compute Separation Using Lifecycles	CSS supports decoupled storage and compute. That is, indexes can be frozen in OBS to reduce the storage cost of cold data. This document describes how to use index lifecycle management to automatically freeze indexes at a specific time to decouple storage and compute.

Scenario		Description
Accelerated data query and analysis	**Using CSS to Accelerate Database Query and Analysis**	Elasticsearch is used as a supplement to relational databases, such as MySQL and GaussDB(for MySQL), to improve the full-text search and high-concurrency ad hoc query capabilities of the databases.
Unified log management platform	**Using CSS to Build a Unified Log Management Platform**	A unified log management platform built using CSS can manage logs in real time in a unified and convenient manner, enabling log-driven O&M and improving service management efficiency.
Querying customized scores	**Configuring Query Scoring in an Elasticsearch Cluster**	This practice describes how to customize scores for documents that match the search criteria.