Alibaba Cloud Elasticsearch

ベストプラクティス

Document Version20200612

目次

1 Elasticsearchの移行

1.1 ユーザーが作成したElasticsearchクラスターからデータを移行する

1.1.1 外部 Elasticsearch インスタンスから Alibaba Cloud Elasticsearch インスタンスへのデータの移行

このトピックでは、外部 Elasticsearch インスタンスから Alibaba Cloud Elasticsearch インスタンスにデータを移行し、データのインデックスを再作成する方法について説明します。 外部 Elasticsearch インスタンスは、Elastic Compute Service (ECS) インスタンスで実行されています。

手順

データ移行手順には、次の手順が含まれます。

1. インデックスの作成
2. データの移行

このトピックでは、よくある質問とそれに対する解決策についても説明します。 詳細については、「FAQ」をご参照ください。

前提条件

このトピックの手順に従ってデータを移行する前に、次の要件が満たされていることを確認してください。 これらの要件を満たしていない場合は、別のデータ移行計画を選択してください。

- 外部 Elasticsearch インスタンスをホストする ECS インスタンスは、VPC ネットワークにデプロイされている必要があります。 ClassicLink を介して VPC ネットワークに接続された ECS インスタンスを使用することはできません。 外部 Elasticsearch インスタンスと Alibaba Cloud Elasticsearch インスタンスが、同じ VPC ネットワークに接続されていることを確認する必要があります。
- このトピックのスクリプトは、中間サーバーで実行できます。 中間サーバーがポート **9200** を介して両方の Elasticsearch インスタンスに接続されていることを確認します。
- Alibaba Cloud Elasticsearch インスタンスの IP アドレスを、外部 Elasticsearch インスタンスをホストする ECS インスタンスの VPC セキュリティグループに追加し、ポート 9200 を有効にします。

- Alibaba Cloud Elasticsearch インスタンスのノードの IP アドレスを、外部 Elasticsearch インスタンスをホストする ECS インスタンスの VPC セキュリティグループに追加します。これにより、ノードは外部 Elasticsearch インスタンスに接続できます。 Alibaba Cloud Elasticsearch インスタンスの Kibana コンソールからノードの IP アドレスをクエリできます。
- 外部 Elasticsearch インスタンスと Alibaba Cloud Elasticsearch インスタンスは相互接続されています。 `curl -XGET http://<host>:9200` コマンドを実行して、中間サーバーで接続をテストできます。

インデックスの作成

外部 Elasticsearch インスタンスの既存のインデックス設定に従って、Alibaba Cloud Elasticsearch インスタンスに新しいインデックスを作成します。 Alibaba Cloud Elasticsearch インスタンスの自動インデックスを有効にして、インデックスとマッピングを自動的に作成することもできます。 ただし、自動インデックスを使用しないことを推奨します。

次の例は、外部 Elasticsearch インスタンスから Alibaba Cloud Elasticsearch インスタンスにインデックスのバッチを移行するための Python スクリプトです。 デフォルトでは、新しいインデックスのレプリカは作成されません。

```
#! /usr/bin/python
# -*- coding: UTF-8 -*-
# File name: indiceCreate.py
import sys
import base64
import time
import httplib
import json
## The ECS instance that hosts the source Elasticsearch instance (IP address + Port).
oldClusterHost = "old-cluster.com"
# The username of the source Elasticsearch instance. The username field can be left
empty.
oldClusterUserName = "old-username"
## The password of the source Elasticsearch instance. The password field can be left
empty.
oldClusterPassword = "old-password"
## The ECS instance that hosts the destination Elasticsearch instance (IP address + Port).
newClusterHost = "new-cluster.com"
## The username of the destination Elasticsearch instance. The username field can be
left empty.
newClusterUser = "new-username"
## The password of the destination Elasticsearch instance. The password field can be left
 empty.
newClusterPassword = "new-password"
DEFAULT_REPLICAS = 0
def httpRequest(method, host, endpoint, params="", username="", password=""):
    conn = httplib.HTTPConnection(host)
    headers = {}
    if (username ! = "") :
        'Hello {name}, your age is {age} !'.format(name = 'Tom', age = '20')
        base64string = base64.encodestring('{username}:{password}'.format(username =
username, password = password)).replace('\n', '')
```

```
        headers["Authorization"] = "Basic %s" % base64string;
    if "GET" == method:
        headers["Content-Type"] = "application/x-www-form-urlencoded"
        conn.request(method=method, url=endpoint, headers=headers)
    else :
        headers["Content-Type"] = "application/json"
        conn.request(method=method, url=endpoint, body=params, headers=headers)
    response = conn.getresponse()
    res = response.read()
    return res
def httpGet(host, endpoint, username="", password=""):
    return httpRequest("GET", host, endpoint, "", username, password)
def httpPost(host, endpoint, params, username="", password=""):
    return httpRequest("POST", host, endpoint, params, username, password)
def httpPut(host, endpoint, params, username="", password=""):
    return httpRequest("PUT", host, endpoint, params, username, password)
def getIndices(host, username="", password=""):
    endpoint = "/_cat/indices"
    indicesResult = httpGet(oldClusterHost, endpoint, oldClusterUserName, oldCluster
Password)
    indicesList = indicesResult.split("\n")
    indexList = []
    for indices in indicesList:
        if (indices.find("open") > 0):
            indexList.append(indices.split()[2])
    return indexList
def getSettings(index, host, username="", password=""):
    endpoint = "/" + index + "/_settings"
    indexSettings = httpGet(host, endpoint, username, password)
    print index + "  The original settings: \n" + indexSettings
    settingsDict = json.loads(indexSettings)
    ## The number of shards equals those of the indexes on the source Elasticsearch
instance by default.
    number_of_shards = settingsDict[index]["settings"]["index"]["number_of_shards"]
    ## The default number of replicas is 0.
    number_of_replicas = DEFAULT_REPLICAS
    newSetting = "\"settings\": {\"number_of_shards\": %s, \"number_of_replicas\": %s}"
 % (number_of_shards, number_of_replicas)
    return newSetting
def getMapping(index, host, username="", password=""):
    endpoint = "/" + index + "/_mapping"
    indexMapping = httpGet(host, endpoint, username, password)
    print index + "The original mappings: \n" + indexMapping
    mappingDict = json.loads(indexMapping)
    mappings = json.dumps(mappingDict[index]["mappings"])
    newMapping = "\"mappings\" : " + mappings
    return newMapping
def createIndexStatement(oldIndexName):
    settingStr = getSettings(oldIndexName, oldClusterHost, oldClusterUserName,
oldClusterPassword)
    mappingStr = getMapping(oldIndexName, oldClusterHost, oldClusterUserName,
oldClusterPassword)
    createstatement = "{\n" + str(settingStr) + ",\n" + str(mappingStr) + "\n}"
    return createstatement
def createIndex(oldIndexName, newIndexName=""):
    if (newIndexName == "") :
        newIndexName = oldIndexName
    createstatement = createIndexStatement(oldIndexName)
    print "New index" + newIndexName + "Index settings and mappings: \n" + createstat
ement
    endpoint = "/" + newIndexName
    createResult = httpPut(newClusterHost, endpoint, createstatement, newClusterUser,
newClusterPassword)
    print "New index" + newIndexName + "Creation result:" + createResult
```

```
## main
indexList = getIndices(oldClusterHost, oldClusterUserName, oldClusterPassword)
systemIndex = []
for index in indexList:
    if (index.startswith(".")):
        systemIndex.append(index)
    else :
        createIndex(index, index)
if (len(systemIndex) > 0) :
    for index in systemIndex:
        print index + "It may be a system index that will not be recreated. You can manually
recreate the index as needed."
```

データの移行

後述のいずれかの方法でデータを移行できます。 移行するデータの量に基づいて適切な方法を選択します。

 :

- データの整合性を確保するには、データを移行する前にソース Elasticsearch インスタンスへのデータの書き込みを停止する必要があります。 ただし、ソース Elasticsearch インスタンスからデータを読み取ることはできます。 データが移行された後、ターゲット Elasticsearch インスタンスに切り替えて、データを読み書きできます。 ソース Elasticsearch インスタンスへのデータの書き込みを停止しない場合、ターゲット Elasticsearch インスタンスのデータはソース Elasticsearch インスタンスのデータと一致しない可能性があります。
- 後述のいずれかの方法でデータを移行する場合、IP + Port を使用してソース Elasticsearch インスタンスを接続するには、最初にターゲット Elasticsearch インスタンスの YML 設定を変更する必要があります。 ソースインスタンスの IP アドレスを reindex ホワイトリストに追加します。 例：reindex.remote.whitelist: 1.1.1.1:9200,1.2. *. *:*。
- エンドポイントを使用してソース Elasticsearch インスタンスに接続する場合、 http://host:port/path 形式を使用しないでください。 ドメイン名に path を含めないでください。

- 少量のデータを移行する

  reindex.sh スクリプトを実行します。

  ```
  #! /bin/bash
  # file:reindex.sh
  indexName="The name of the index"
  newClusterUser="The username of the destination Elasticsearch instance"
  Newclusterpass = "The password of the destination Elasticsearch instance"
  newClusterHost="The ECS instance that hosts the destination Elasticsearch instance"
  Oldclusteruser = "The username of the source Elasticsearch instance"
  Oldclusterpass = "The password of the source Elasticsearch instance"
  # The address of the ECS instance that hosts the source Elasticsearch instance must be
   in this format: [scheme]://[host]:[port]. Example: http://10.37.1.1:9200.
  Oldclusterhost = "The ECS instance that hosts the source Elasticsearch instance"
  curl -u ${newClusterUser}:${newClusterPass} -XPOST "http://${newClusterHost}/
  _reindex? pretty" -H "Content-Type: application/json" -d'{
  ```

```
  "source": {
    "remote": {
      "host": "'${oldClusterHost}'",
      "username": "'${oldClusterUser}'",
      "password": "'${oldClusterPass}'"
    },
    "index": "'${indexName}'",
    "query": {
      "match_all": {}
    }
  },
  "dest": {
    "index": "'${indexName}'"
  }
}'
```

- 大量のデータを移行する (削除操作なし、ローリング更新時間ありの場合)

  削除操作なしで大量のデータを移行するには、ローリング更新を使用すると、書き込み操作の中断期間を短縮できます。 ローリング更新では、スクリプトに時系列フィールドを追加して、更新時間を定義する必要があります。 ソース Elasticsearch インスタンスからターゲット Elasticsearch インスタンスにデータを移行した後、ソース Elasticsearch インスタンスへのデータの書き込みを停止できます。 次に、ローリング更新を使用して、データ移行中に更新されたデータを同期します。 ローリング更新が完了したら、ターゲット Elasticsearch インスタンスに切り替え、読み取り操作と書き込み操作を再開します。

```
#! /bin/bash
# file: circleReindex.sh
# CONTROLLING STARTUP:
# This is a script that uses the Reindex operation to remotely reindex the data.
Requirements:
#1. You have created indexes on the destination Elasticsearch instance, or the
destination instance supports auto-indexing and dynamic mapping.
# 2. You must configure the IP whitelist in the YML configuration of the destination
Elasticsearch instance: reindex.remote.whitelist: 172.16.123. *:9200
#3. The specified ECS instance address must be in the following format: [scheme]://[
host]:[port].
USAGE="Usage: sh circleReindex.sh <count>
      count: the number of times to perform the reindex operation. A negative number
indicates loop execution. You can set this parameter to perform the reindex operation
only once or multiple times.
Example:
       sh circleReindex.sh 1
       sh circleReindex.sh 5
       sh circleReindex.sh -1"
indexName="The name of the index"
newClusterUser="The username of the destination Elasticsearch instance"
newClusterPass="The password of the destination Elasticsearch instance"
oldClusterUser="The username of the source Elasticsearch instance"
oldClusterPass="The password of the source Elasticsearch instance"
## http://myescluster.com
newClusterHost="The ECS instance that hosts the destination Elasticsearch instance"
# You need to specify address of the ECS instance that hosts the source Elasticsearch
instance in the following format: [scheme]://[host]:[port]. Example: http://10.37.1.1:
9200
oldClusterHost="The ECS instance that hosts the source Elasticsearch instance"
timeField="The field that specifies the time window during which the incremental data
 is migrated"
```

```
reindexTimes=0
lastTimestamp=0
curTimestamp=`date +%s`
hasError=false
function reIndexOP() {
    reindexTimes=$[${reindexTimes} + 1]
    curTimestamp=`date +%s`
    ret=`curl -u ${newClusterUser}:${newClusterPass} -XPOST "${newClusterHost}/
_reindex? pretty" -H "Content-Type: application/json" -d '{
        "source": {
          "remote": {
            "host": "'${oldClusterHost}'",
            "username": "'${oldClusterUser}'",
            "password": "'${oldClusterPass}'"
          },
          "index": "'${indexName}'",
          "query": {
            "range" : {
              "'${timeField}'" : {
                "gte" : '${lastTimestamp}',
                "lt" : '${curTimestamp}'
              }
            }
          }
        },
        "dest": {
          "index": "'${indexName}'"
        }
    }'`
    lastTimestamp=${curTimestamp}
    echo "${reindexTimes} reindex operations have been performed. The last reindex
operation is completed at ${lastTimestamp} Result: ${ret}"
    if [[ ${ret} == *error* ]]; then
        hasError=true
        echo "An unknown error occurred while performing this operation. All subsequent
 operations have been suspended."
    fi
}
function start() {
    ## A negative number indicates loop execution.
    if [[ $1 -lt 0 ]]; then
        while :
        do
            reIndexOP
        done
    elif [[ $1 -gt 0 ]]; then
        k=0
        while [[ k -lt $1 ]] && [[ ${hasError} == false ]]; do
            reIndexOP
            let ++k
        done
    fi
}
## main
if [ $# -lt 1 ]; then
    echo "$USAGE"
    exit 1
fi
echo "Start the reindex operation for index ${indexName}"
start $1
```

```
echo "You have performed ${reindexTimes} reindex operations"
```

- 大量のデータを移行する (削除操作またはローリング更新時間なしの場合)

  大量のデータを移行する必要があり、マッピングに更新時間フィールドが定義されていない場合、ソースインスタンスのワークロードのスクリプトに更新時間フィールドを追加する必要があります。 フィールドを追加した後、既存のデータを移行してから、前述のデータ移行計画で説明されているローリング更新を使用して増分データを移行できます。

```
#! /bin/bash
# file:miss.sh
indexName="The name of the index"
newClusterUser="The username of the destination Elasticsearch instance"
Newclusterpass = "The password of the destination Elasticsearch instance"
newClusterHost="The ECS instance that hosts the destination Elasticsearch instance"
Oldclusteruser = "The username of the source Elasticsearch instance"
Oldclusterpass = "The password of the source Elasticsearch instance"
# The address of the ECS instance that hosts the source Elasticsearch instance must be
 in this format: [scheme]://[host]:[port]. Example: http://10.37.1.1:9200.
oldClusterHost="The ECS instance that hosts the source Elasticsearch instance"
timeField="updatetime"
curl -u ${newClusterUser}:${newClusterPass} -XPOST "http://${newClusterHost}/
_reindex? pretty" -H "Content-Type: application/json" -d '{
   "source": {
      "remote": {
         "host": "'${oldClusterHost}'",
         "username": "'${oldClusterUser}'",
         "password": "'${oldClusterPass}'"
      },
      "index": "'${indexName}'",
      "query": {
         "bool": {
            "must_not": {
               "exists": {
                  "field": "'${timeField}'"
               }
            }
         }
      }
   },
   "dest": {
     "index": "'${indexName}'"
   }
}'
```

- 書き込み操作を停止せずにデータを移行する

  このデータ移行計画は今後提供される予定です。

FAQ

- 問題：curl コマンドを実行すると、{"error":"Content-Type header [application/x-www-form-urlencoded] is not supported","status":406} が表示される。

  解決策：-H "Content-Type: application/json" パラメーターを **curl** コマンドに追加し、もう一度お試しください。

```
// Obtain all the indexes on the source instance. If you do not have the required permissions, remove the "-u user:pass" parameter. Make sure that you have replaced oldClusterHost with the name of the ECS instance that hosts the source Elasticsearch instance.
curl -u user:pass -XGET http://oldClusterHost/_cat/indices | awk '{print $3}'
// Based on the returned indexes, obtain the setting and mapping of the index that you need to migrate for the specified user. Make sure that you have replaced indexName with the index name that you need to query.
curl -u user:pass -XGET http://oldClusterHost/indexName/_settings,_mapping?pretty=true
// Create a new index on the destination Elasticsearch instance according to the _settings and _mapping settings that you have obtained from the preceding step. You can set the number of index replicas to 0 to accelerate the data synchronization process, and change the number of replicas to one after the migration is complete.
// ewClusterHost indicates the ECS instance that hosts the destination Elasticsearch instance, testindex indicates the name of the index that you have created, and testtype indicates the type of the index.
curl -u user:pass -XPUT http://<newClusterHost>/<testindex> -d '{
  "testindex" : {
    "settings" : {
      "number_of_shards" : "5", //Specify the number of shards for the corresponding index on the source Elasticsearch instance, for example, 5
      "number_of_replicas" : "0" //Set the number of index replicas to zero
    }
    },
    "mappings" : { //Set the mapping for the index on the source Elasticsearch instance. For example, you can set the mapping as follows
      "testtype" : {
        "properties" : {
          "uid" : {
            "type" : "long"
          },
          "name" : {
            "type" : "text"
          },
          "create_time" : {
            "type" : "long"
          }
        }
      }
    }
  }
```

```
}'
```

- 問題：データ移行プロセスが遅すぎる。

  解決策：インデックスが大きすぎる場合、移行前にレプリカの数を 0 に、更新間隔を -1 に設定します。 データの移行後、レプリカと更新の設定を元の値に戻します。 これにより、同期プロセスが高速化されます。

```
// You can set the number of index replicas to 0 and disable refresh to accelerate the migration process.
curl -u user:password -XPUT 'http://<host:port>/indexName/_settings' -d' {
      "number_of_replicas" : 0,
      "refresh_interval" : "-1"
}'
// After the data is migrated, set the number of index replicas to 1 and the refresh interval to 1s (default value, which means 1 second).
curl -u user:password -XPUT 'http://<host:port>/indexName/_settings' -d' {
      "number_of_replicas" : 1,
      "refresh_interval" : "1s"
}'
```

注：

このトピックのコンテンツの一部は、『Reindex API』から引用しています。

1.2 サードパーティのElasticsearchインスタンスからAlibaba Cloud Elasticsearchにデータを移行する

2 データ移行

2.1 RDS同期

2.1.1 MySQL データの Elasticsearch との同期

Alibaba Cloud は、幅広いクラウドストレージとデータベースサービスを提供します。 DataWorks の Data Integration を使用して、ストレージサービスやデータベースサービスと Alibaba Cloud Elasticsearch (ES) との間でデータを同期し、データのクエリや分析をすることができます。 Data Integration を使用すると、最小 5 分間隔でデータを同期できます。

:

データの同期には、インターネットデータ転送料金が発生する場合があります。

概要

次の手順に従って、データベースに保存されているデータを分析・検索します。

1. データベースを作成します。 ApsaraDB RDS for RDS データベースを使用するか、ローカルサーバーにデータベースを作成することができます。 このトピックでは、ApsaraDB RDS

for MySQL データベースを使用します。 2 つの MySQL テーブルを結合してから、データを Elasticsearch と同期します。 次の図に、2 つの MySQL テーブルを示します。

図 2-1 : テーブル 1

	id	stu_id	c_name	grade
1	1	901	compute	98
2	2	901	english	80
3	3	902	compute	65
4	4	902	chinese	88
5	5	903	chinese	95
6	6	904	compute	70
7	7	904	english	92
8	8	905	english	94
9	9	906	compute	90
10	10	906	english	85

図 2-2 : テーブル 2

	id	name	sex	birth	department	address
1	901	zhangda	man	1985	compute	beijing
2	902	zhanger	man	1986	chinese	beijing
3	903	zhangsan	woman	1990	chinese	hunan
4	904	lisi	man	1990	english	liaoning
5	905	wangwu	woman	1991	english	fujian
6	906	wangliu	man	1988	compute	hunan

2. VPC ネットワークを介して Elasticsearch インスタンスに接続可能な Elastic Compute Service (ECS) インスタンスを購入します。 ECS インスタンスを使用して、MySQL データベースからデータを読み取り、同期タスクを実行して Elasticsearch にデータを書き込みます。 同期タスクは Data Integration によってディスパッチされます。
3. Data Integration を有効化し、同期タスクを実行するためのリソースとして ECS インスタンスを Data Integration に追加します。
4. データ同期スクリプトを作成し、定期的に実行します。
5. Data Integration で同期されたデータを保存するための Elasticsearch インスタンスを作成します。

準備

1. VPC ネットワークと VSwitch を作成します。
2. #unique_9を行います。

   :

   Elaticsearch インスタンスの **[**リージョン**]**、**[VPC** ネットワーク**]**、**[VSwitch]** は、手順 1 で指定したものと同じでなければなりません。

3. Elasticsearch インスタンスと同じ VPC ネットワークに接続されている ECS インスタンスを購入し、パブリック IP アドレスか Elastic IP アドレス (EIP) を割り当てます。

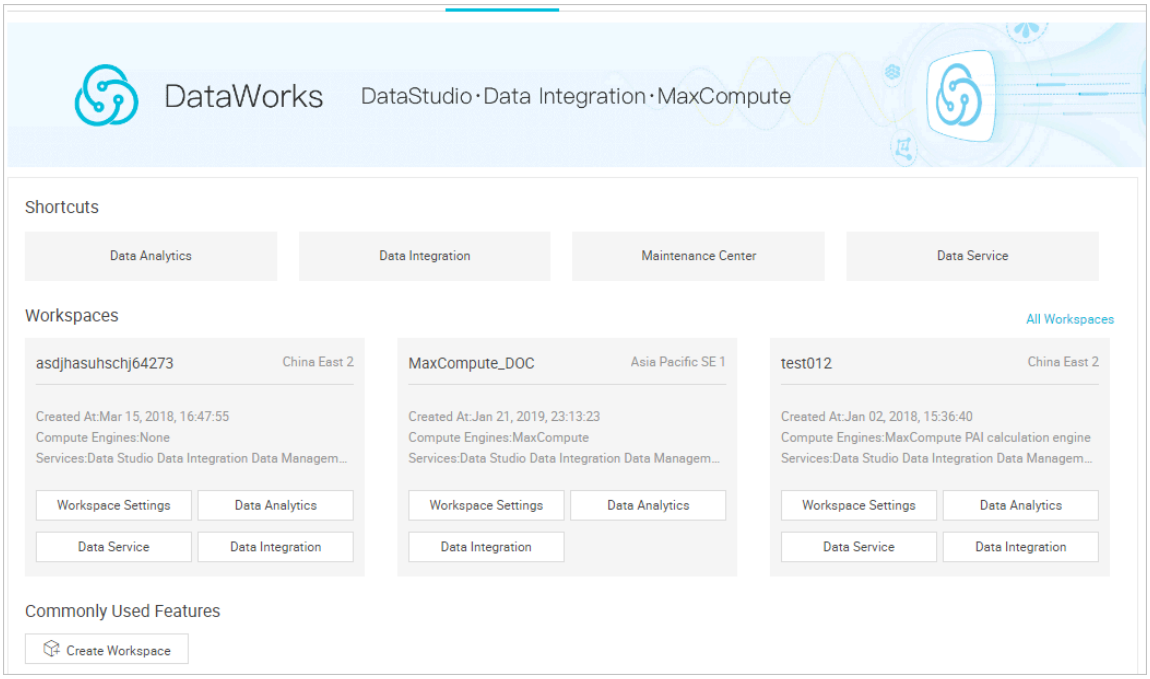
   注 **:**

   - 既存の ECS インスタンスを使用することもできます。
   - ECS インスタンスには、CentOS 6、CentOS 7、または AliyunOS を選択することを推奨します。
   - ECS インスタンスで MaxCompute または同期タスクを実行する場合、ECS インスタンスが Python V2.6 または V2.7 を実行していることを確認します。 CentOS 5 は Python V2.4 を使用します。 他のオペレーティングシステムは、Python バージョン V2.6 以降を使用します。
   - ECS インスタンスにパブリック IP アドレスが割り当てられていることを確認します。

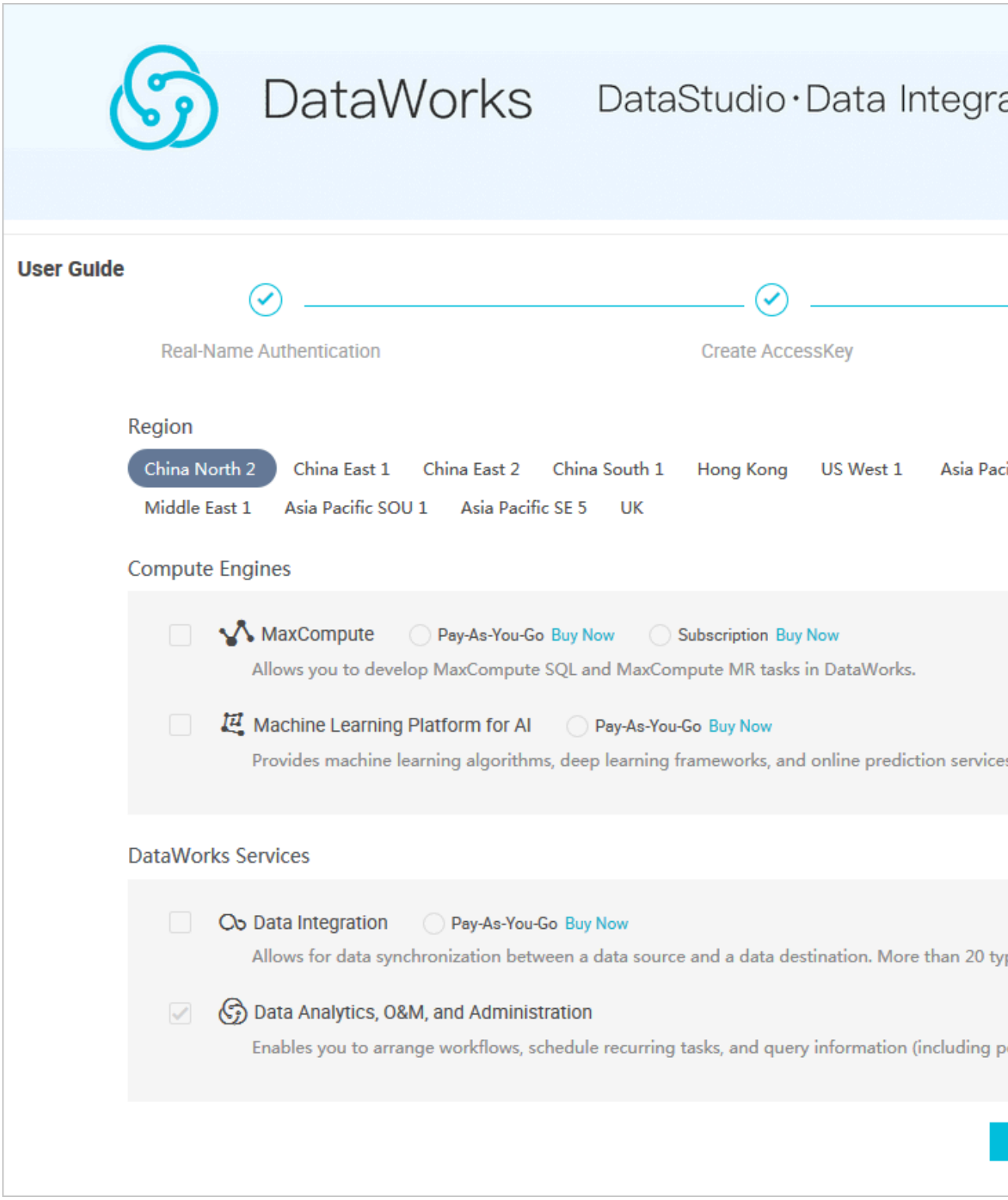
リソースグループの作成とデータソースの追加

1. DataWorks コンソールにログインし、**[Workspaces]** タブをクリックします。

   - Data Integration が既に有効化されている場合、次のページが表示されます。



   - Data Integration が有効化されていない場合、次のページが表示されます。 手順に従って Data Integration を有効化します。 Data Integration は有料サービスです。 ページに表示される課金項目に基づいて料金を見積もることができます。

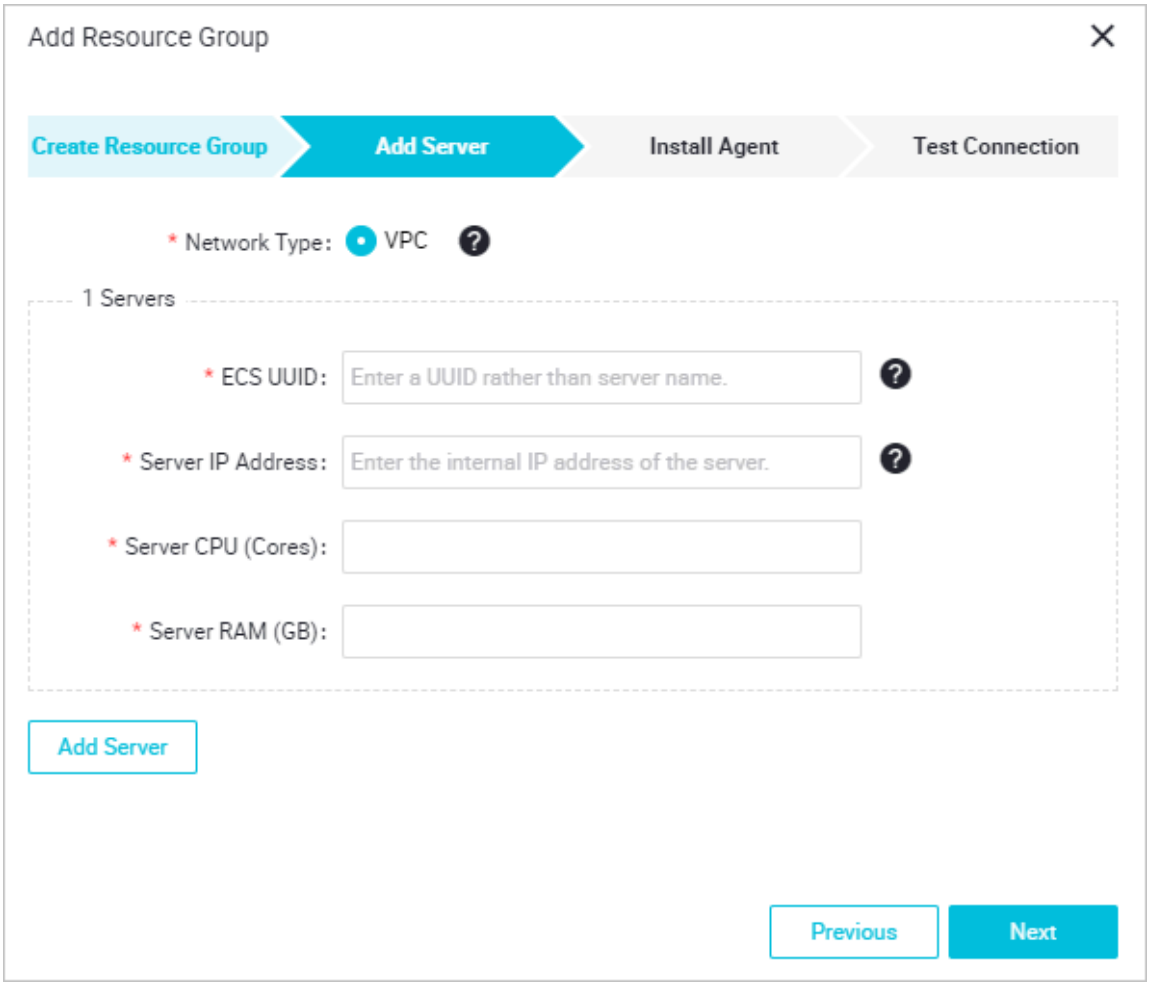


2. 購入したワークスペースの [Actions] 列の **[Data Integration]** をクリックします。
3. **[Data Integration]** ページの左側のナビゲーションペインで **[Resource Group]** を選択し、右上隅の **[Add Resource Group]** をクリックします。

4. リソースグループを追加します。

   a. サーバーを追加します。

   リソースグループ名を入力し、サーバー情報を指定します。 この例では、購入した ECS インスタンスを追加します。 サーバー情報を次に示します。



フィールド	フィールドの説明
ECS UUID	#unique_10を行い、dmidecode | grep UUID を実行します。戻り値をこのフィールドに入力します。

フィールド	フィールドの説明
Server IP Address/Server CPU (Cores)/Server RAM (GB)	ECS インスタンスのパブリック IP アドレス、CPU コア、メモリサイズ。 必要な情報を取得するには、ECS コンソールにログインし、ECS インスタンス名をクリックします。 **[**設定情報**]** フィールドに、情報が表示されます。

b. **Security Center** エージェントをインストールします。

ページの指示に従って、エージェントをインストールします。 手順 5 で、ECS インスタンスのポート 8000 を開きます。 この手順をスキップすると、デフォルト設定が使用されます。

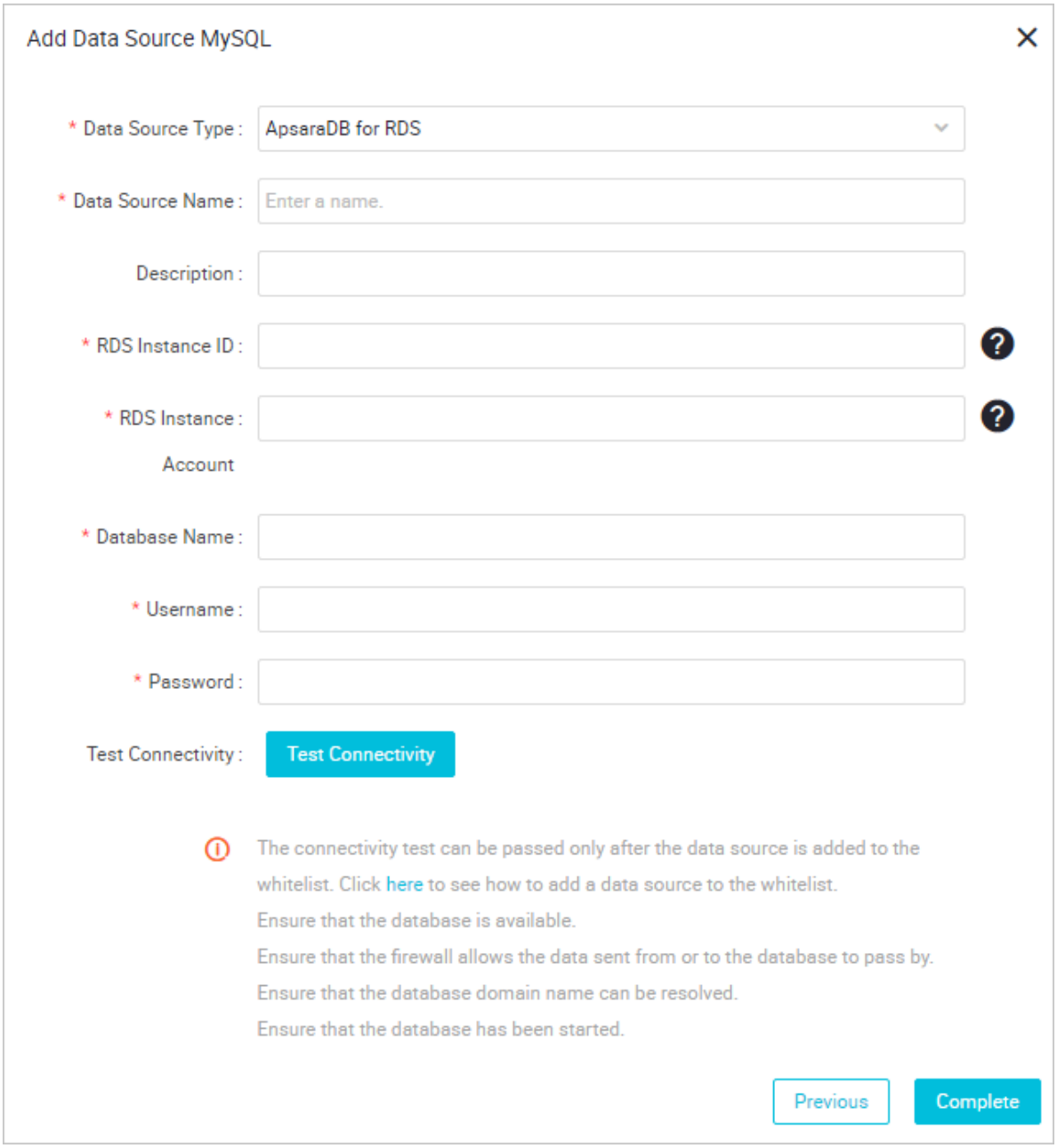
c. 接続をテストします。

5. MySQL データベースと Elasticsearch ホワイトリストを設定します。

ECS インスタンスの IP アドレスを MySQL データベースと Elasticsearch インスタンスのホワイトリストに追加して、ECS インスタンスが MySQL データベースと Elasticsearch インスタンスと通信できるようにします。

6. 左側のナビゲーションペインで **[Connections]** を選択し、**[Add Connection]** をクリックします。

7. **[MySQL]** を選択します。 **[Add MySQL Connection]** ページで、必要な情報を入力します。



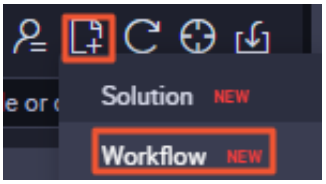
Connect To：この例では、**ApsaraDB RDS for MySQL** データベースを使用します。 **[User-created Data Store with Public IP Addresses]** か **[User-created Data Store without Public IP Addresses]** から選択できます。 パラメーターの詳細については、「#unique_11」をご参照ください。

:

ECS インスタンスがデータベースへの接続に失敗した場合、データベースのホワイトリストを確認してください。

同期タスクの作成

1. DataWorks コンソールにノード所有者としてログインします。
2. **[Workspaces]** タブで、**[Actions]** 列の **[Data Analytics]** をクリックします。
3. **[Data Studio]** ページで、**[Create]** > **[Workflow]** を選択します。



4. **[Create Workflow]** ダイアログボックスで、**[Workflow Name]** と**[Description]** を指定し、**[Create]** をクリックします。
5. 作成したワークフローを左側のワークフローリストで展開し、**[Data Integration]** を右クリックし、**[Create Data Integration Node]** > **[Sync]** を選択します。
6. **[Create Node]** ダイアログボックスで、**[Node Name]** を指定し、**[Commit]** をクリックします。
7. [node] タブの上部のツールバーで、**[Switch to Code Editor]** アイコン をクリックします。
8. 操作を確認して、コードエディターに切り替えます。

   コードエディターの使用方法については、「#unique_12」をご参照ください。

   次のスクリプトは、2 つのテーブルから学生と試験の情報を取得する例です。

```
{
  "type": "job",
  "steps": [
    {
      "stepType": "mysql",
      "parameter": {
        "column": [
          &amp;quot;id&amp;quot;、
          "name",
          "sex",
          "birth",
          "department",
          "address": {
        ],
        "connection": [
          {
            "querysql":["SELECT student.id,name,sex,birth,department,address,
c_name,grade FROM student JOIN score on student.id=score.stu_id;"],
            "datasource": "zl_****_rdsmysql",
            "table": [
              "score"
            ]
          }
        ],
        "where": "",
```

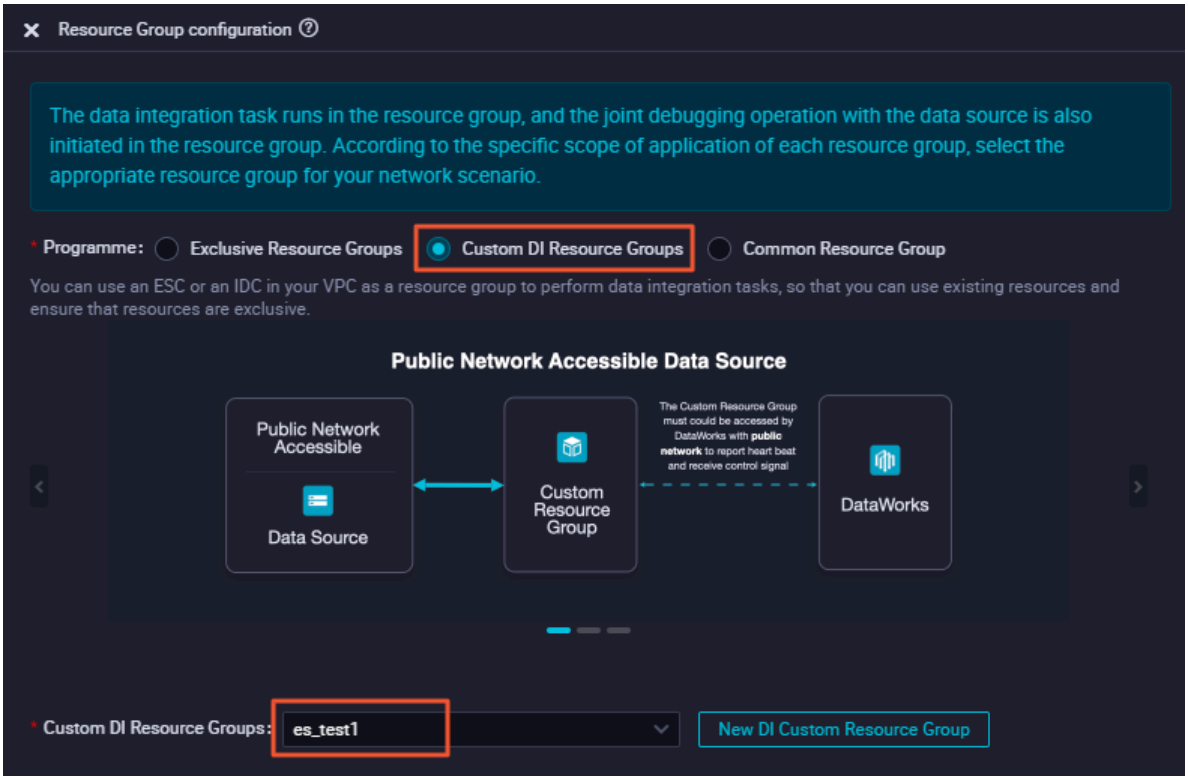
```
            "splitPk": "",
            "encoding": "UTF-8"
        },
        "name":"Reader",
        "category": "reader"
    },
    {
        "stepType": "elasticsearch",
        "parameter": {
            "accessId": "elastic",
            "endpoint": "http://es-cn-0p*********2dpxtx.elasticsearch.aliyuncs.com:9200",
            "indexType": "score",
            "accessKey": "******",
            "cleanup": true,
            "discovery": false,
            "column": [
                {
                    "name":"student_id",
                    "type":"id"
                },
                 {
                    "name": "sex",
                    "type": "text"
                },
                {
                    "name": "name",
                    "type": "text"
                },
                {
                    "name": "birth",
                    "type": "integer"
                },
                {
                    "name": "quyu",
                    "type": "text"
                },
                {
                    "name": "address",
                    "type": "text"
                },
                {
                    "name": "cname",
                    "type": "text"
                },
                {
                    "name": "grades",
                    "type": "integer"
                }
            ],
            "index": "mysqljoin",
            "batchSize": 1000,
            "splitter": ","
        },
        "name": "Writer",
        "category": "writer"
    }
],
"version": "2.0",
"order": {
    "hops": [
        {
            "from": "Reader",
            "to": "Writer"
        }
```

```
      ]
    },
    "setting": {
      "jvmOption": "-Xms1024m -Xmx1024m",
      "errorLimit": {
        "record": ""
      },
      "speed": {
        "throttle": false,
        "concurrent": 1
      }
    }
}
```

スクリプトには 3 つのセクションがあります。

- Reader：このセクションは、#unique_13に使用されます。querysql を使用して、特定の条件に基づいてデータを取得する SQL 文を定義します。 querysql が設定されている場合、MySQL Reader は table、column、where、splitPk 条件を無視します。 これは、querysql の優先順位は table、column、where、splitPk よりも高いためです。 datasource は querysql を使用して、ユーザー名とパスワード情報を解析します。
- Writer：このセクションは、#unique_14に使用されます。
  - endpoint：Elasticsearch インスタンスのパブリックまたはプライベートネットワークエンドポイント。 Elasticsearch インスタンスに接続するには、Elasticsearch インスタンスの **[セキュリティ]** ページでパブリックまたはプライベートネットワークのホワイトリストを設定する必要があります。
  - accessId/accessKey：Elasticsearch インスタンスのユーザー名とパスワード。 デフォルトのユーザー名は、elastic です。
  - index ：Elasticsearch インスタンスのインデックスの名前。 インデックスに保存されたデータにアクセスするには、インデックス名を指定する必要があります。
  - Reader と Writer 内の列は、同じ順序で定義する必要があります。 Reader は、指定された列からデータを読み取り、データを配列に保存します。 次に、Writer は配列からデータを取り出し、定義された列に順番にデータを書き込むためです。
- setting：このセクションは、パケット損失や最大同時数などの設定に使用されます。

9. スクリプトの設定後、右上隅の **[Resource Group]** をクリックし、作成したリソースグループを選択し、上部のツールバーで **[Run]** アイコンをクリックして MySQL データを Elasticsearch と同期します。



同期結果の確認

1. Elasticsearch インスタンスの Kibana コンソールにログインします。
2. 左側のナビゲーションペインで、**[Dev Tools]** を選択します。
3. **[Console]** タブで、次のコマンドを実行して同期データをクエリします。

```
POST /mysqljoin/_search? pretty
{
"query": { "match_all": {}}
}
```

文字列 **mysqljoin** は、同期データが保存されている **index** の名前です。

FAQ

- Q：データベース接続エラーを解決するにはどうすればよいですか。

  A：リソースグループ内の ECS インスタンスのパブリックまたはプライベート IP アドレスが、データベースのホワイトリストに追加されているかどうかを確認してください。 IP アドレスがホワイトリストに追加されていない場合は、追加してください。

- Q：Elasticsearch インスタンスの接続エラーを解決するにはどうすればよいですか。

  A：次の手順に従って原因を特定します。

  1. コードエディターの右上隅にある **[Resource Group]** をクリックし、前の手順で選択したリソースグループが選択されているかどうかを確認します。
     - 選択されている場合、 次の手順に進みます。
     - 選択されていない場合、 **[Resource Group]** をクリックし、作成したリソースグループを選択します。 次に、**[Run]** アイコンをクリックしてスクリプトを実行します。
  2. リソースグループ内の ECS インスタンスの IP アドレスが Elasticsearch インスタンスのホワイトリストに追加されているかどうかを確認します。
     - 追加されている場合、 次の手順に進みます。
     - 追加されていない場合、 ECS インスタンスの IP アドレスを Elasticsearch インスタンスのホワイトリストに追加してください。

     > **:**
     > ECS インスタンスのプライベート IP アドレスが使用されている場合、Elasticsearch インスタンスの **[セキュリティ]** ページに移動し、IP アドレスを Elasticsearch システムのホワイトリストに追加します。 ECS インスタンスのパブリックIPアドレスが使用されている場合、Elasticsearch インスタンスの **[セキュリティ]** ページに移動し、IP アドレスをパブリックネットワークホワイトリストに追加します。
  3. スクリプトの設定が正しいかどうかを確認します。 確認する必要があるフィールドには、`endpoint` (Elasticsearch インスタンスのパブリックまたはプライベートネットワークエンドポイント)、`accessId` (Elasticsearch インスタンスのユーザー名。 デフォルトのユーザー名は elastic)、`accessKey` (Elasticsearch インスタンスのパスワード) があります。

2.1.2 RDS for MySQL から ES へのリアルタイムデータ同期

このセクションでは、Data Transmission Service (DTS) を使用して、RDS for MySQL インスタンスから Elasticsearch (ES) インスタンスへのリアルタイムデータ同期タスクを迅速に作成する方法を説明します。 DTS はこの同期機能を使用して RDS for MySQL データを ES インスタンスに同期し、データをリアルタイムでクエリします。

リアルタイム同期タイプ

同じ Alibaba Cloud アカウントの DTS インスタンスで、RDS for MySQL から ES 。

SQL 操作タイプ

サポートされている主な SQL 操作タイプは次のとおりです。

- Insert
- Delete
- Update

> 注：
> DTS では、DDL 文を使用してデータを同期することはできません。 データを同期する場合、DDL 操作は無視されます。

RDS for MySQL インスタンスで DDL を使用するテーブルが検出されると、対応するテーブルの DML 操作が失敗する可能性があります。 この問題を解決するには、以下の手順を実行します。

1. 同期リストからオブジェクトを削除します。 詳細は、「」「同期オブジェクトの削除」「」をご参照ください。
2. ES インスタンスのこのテーブルに対応するインデックスを削除します。
3. テーブルを同期リストに追加し直し、再初期化してください。 詳細は、「」「同期オブジェクトの追加」「」をご参照ください。

DDL を使用して列を追加したりテーブルを変更したりする場合、DDL 操作順序は次のとおりです。

1. ES インスタンスのマッピングと新しい列を手動で変更します。
2. テーブルスキーマを変更し、ソース RDS for MySQL インスタンスに新しいスキーマを追加します。
3. DTS でインスタンスの同期を停止します。DTS 同期インスタンスを再起動して ES で変更されたマッピング関係を再ロードします。

データ同期の設定

RDS for MySQL インスタンスから ES インスタンスにデータを同期するには、以下の手順を実行します。

1. DTS 同期インスタンスの購入

   Data Transmission Service コンソールにログインし、**[データ同期]** ウィンドウに移動します。 右上隅にある **[同期タスクの作成]** をクリックして同期インスタンスを購入します。 同期インスタンスを設定できるようになります。

   > 注：
   > 設定する前に同期インスタンスを購入する必要があります。 サブスクリプションと従量課金の 2 つの課金方法がサポートされています。.

   購入ページのパラメーター

   - 機能

     [データ同期] を選択します。
   - ソースインスタンス

     [MySQL] を選択します。
   - ソースリージョン
     - この例では、RDS for MySQL インスタンスを使用するため、RDS for MySQL インスタンスが存在するリージョンを選択する必要があります。
   - ターゲットインスタンス

     [Elasticsearch] を選択します。
   - ターゲットリージョン

     Elasticsearch インスタンスが存在するリージョンを選択します。 同期インスタンスを購入した後、リージョンを変更することはできません。
   - 仕様

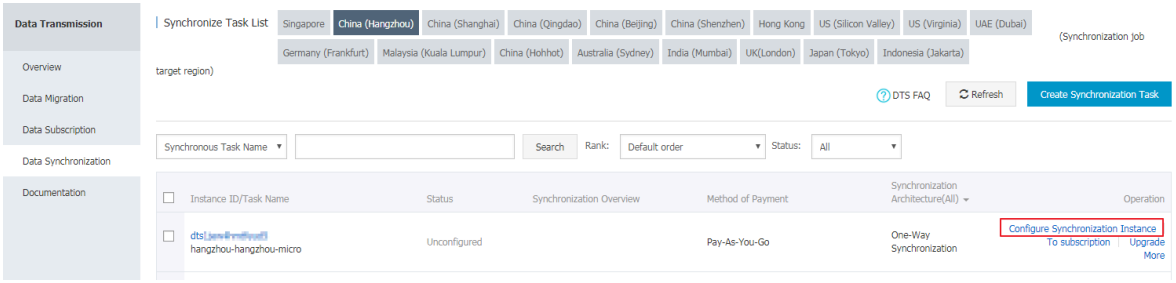
     インスタンス仕様は、同期インスタンスのパフォーマンスに対応しています。 詳細は、「」「データ同期仕様」「」をご参照ください。
   - 購入期間
     - 同期インスタンスがサブスクリプションの場合、デフォルトの購入期間は 1 ヶ月です。
   - 数

     デフォルトの数量は 1 です。

     > 注：

DTS 同期インスタンスのリージョンは、選択したターゲットリージョンです。 たとえば、[中国 (杭州)] リージョンの RDS for MySQL から [中国 (杭州)] リージョンの Elasticsearch への同期インスタンスの場合、DTS 同期インスタンスのリージョンは [中国 (杭州)] です。 同期インスタンスを設定するには、DTS でそのリージョンのインスタンスリストに移動し、購入したばかりの同期インスタンスを検索して、右上隅の **[**同期インスタンスの設定**]** をクリックします。

2. 同期インスタンスの設定

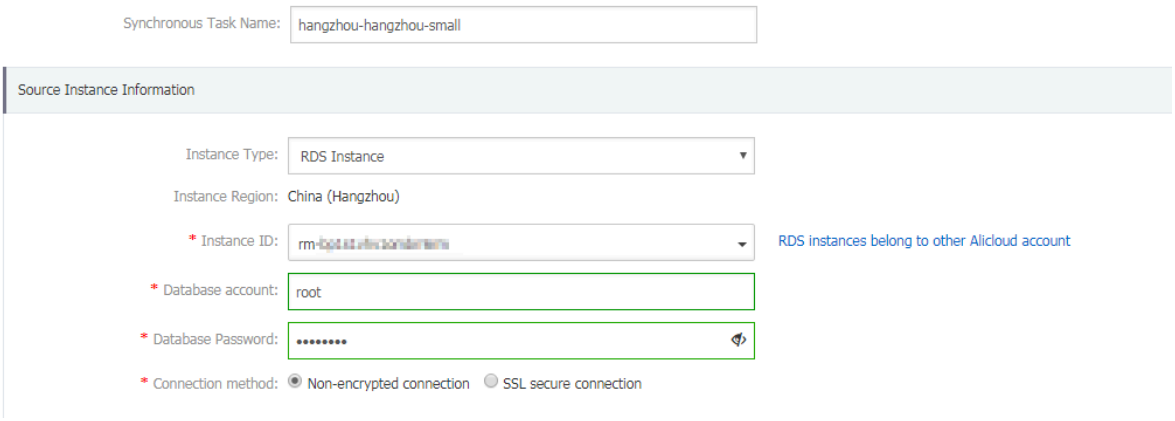


同期タスク名

同期インスタンスの名前に関する要件はありません。

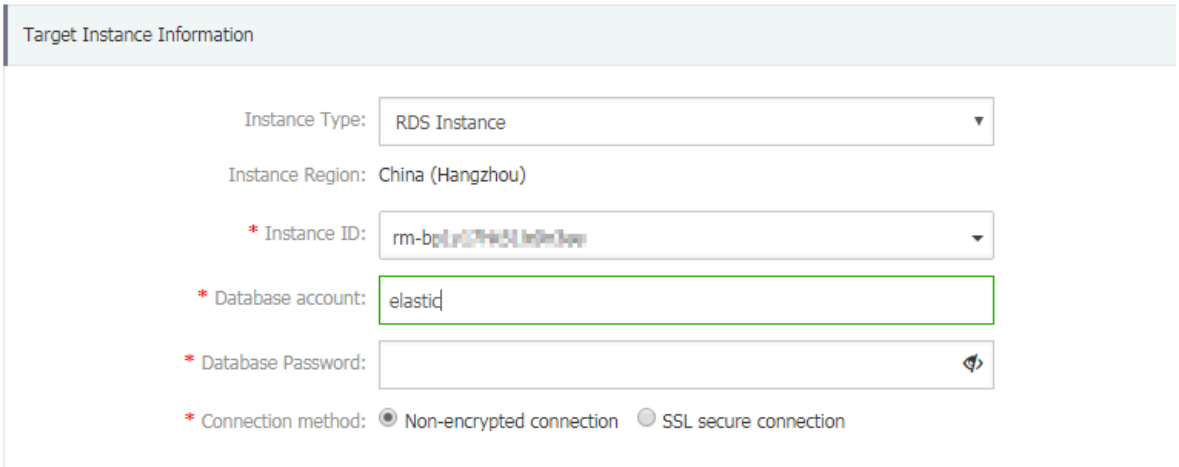
ソースインスタンス

この例では、データソースとして RDS for MySQL を使用します。 インスタンスのタイプ、リージョン、ID、およびデータベースのアカウントとパスワードを設定する必要があります。



ターゲットインスタンス

ES インスタンスの ID、アカウント、パスワードを設定する必要があります。



これらの設定が完了したら、**[ホワイトリストを承認して次のステップに進む]** をクリックして、RDS for MySQL と ES インスタンスのホワイトリストに IPを追加します。

3. インスタンスのホワイトリストの承認

> 注 **:**
> ソースインスタンスが RDS for MySQL の場合、自動的に IP がホワイトリストに追加されるか、セキュリティグループが追加されます。

ソースインスタンスが RDS for MySQL の場合、インスタンス IP が RDS インスタンスのホワイトリストのセキュリティグループに追加されます。 これにより、同期タスクを作成するとき、DTS インスタンスと RDS データベース間の切断によって引き起こされるエラーを回避することができます。 同期タスクの安定性を確保するため、RDS インスタンスからインスタンス IP を削除しないでください。

ホワイトリストの承認後、**[次へ]** をクリックして同期アカウントを作成します。

4. 同期オブジェクトの選択

同期オブジェクトとインデックスの命名規則を設定するには、次の手順を実行します。

a. インデックスの命名規則を [テーブル名] または [データベース名_テーブル名] から選択します。

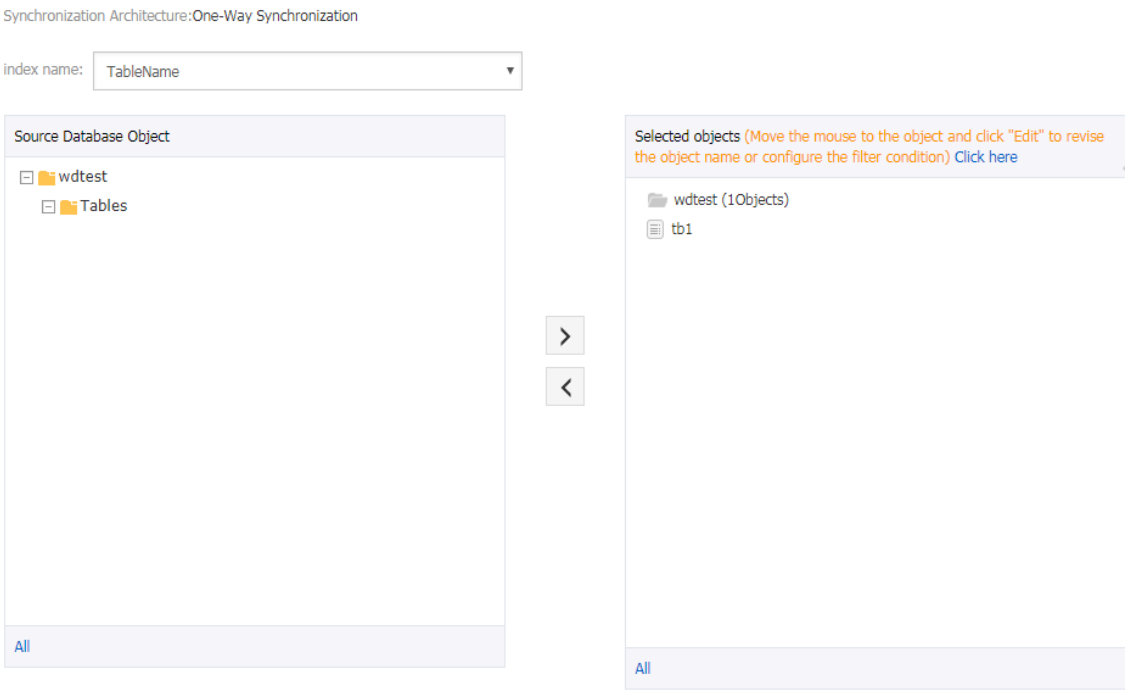
- [テーブル名] を選択した場合、インデックスの名前はテーブルの名前です。
- [データベース名_テーブル名] を選択した場合、インデックスの命名規則はデータベース名_テーブル名です。 たとえば、データベース名が dbtest、テーブル名が sbtest1 の場

合、テーブルが ES インスタンスと同期された後にインデックス名は dbtest_sbtest1 になります。

- 同じ名前の 2 つのテーブルが、それぞれ異なるデータベースにある場合、インデックス名をデータベース名_テーブル名に設定することを推奨します。

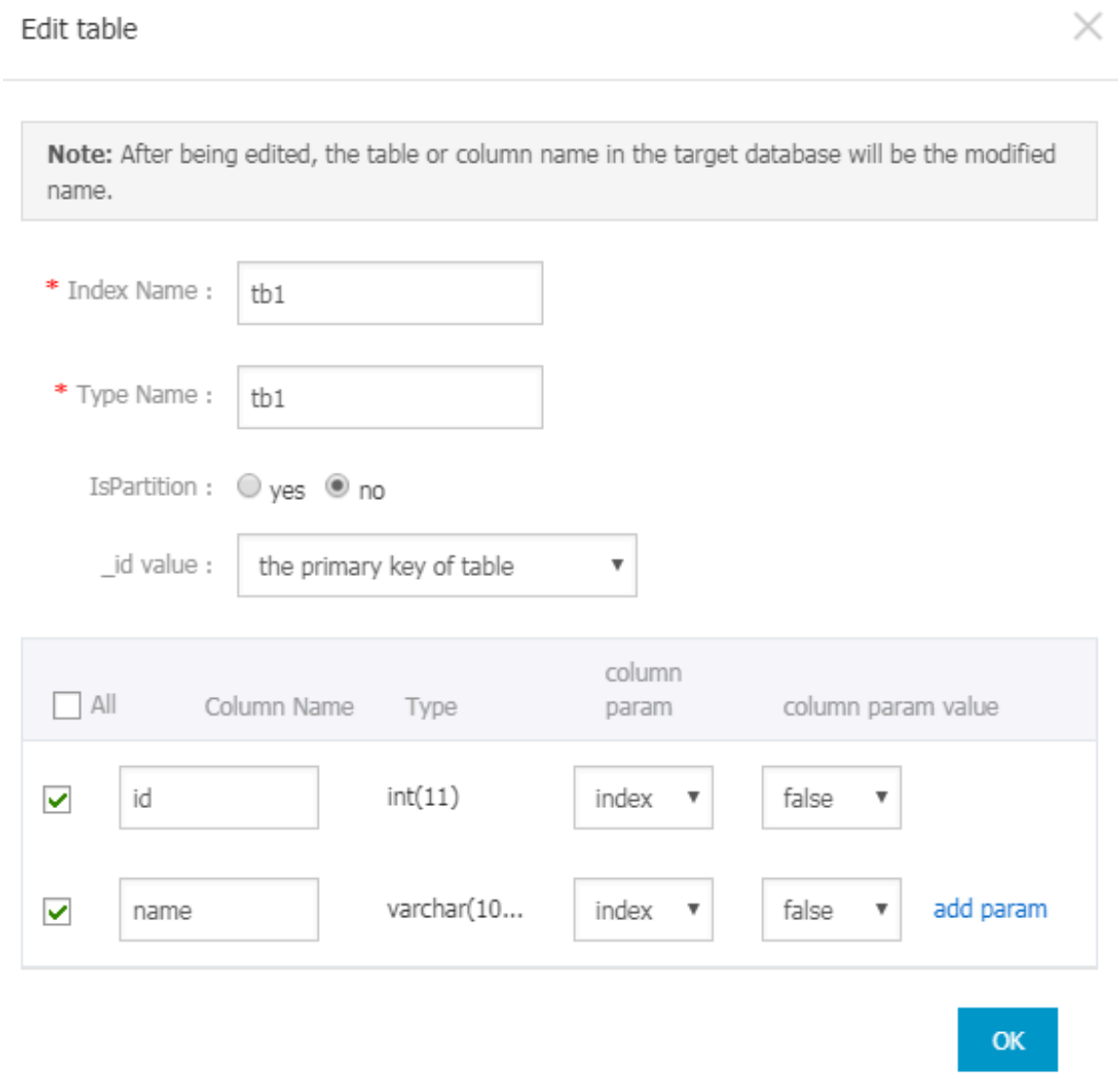
b. 特定のデータベース、テーブル、列を選択します。 同期オブジェクトの選択可能粒度は、テーブルレベルの操作をサポートします。 つまり、複数のデータベースとテーブルを同期できます。

同期オブジェクトの選択可能粒度は、テーブルレベルの操作をサポートします。 つまり、複数のデータベースとテーブルを同期できます。



c. デフォルトでは、すべてのテーブルの docid はプライマリキーです。 プライマリキーを持たないテーブルの場合、ソーステーブルの列に対応する docid を設定します。 右側の [選

択したオブジェクト] のボックスで、対応するテーブルの上にポインターを移動し、**[編集]** をクリックして [詳細設定] ウィンドウに移動します。



d. [詳細設定] では、インデックス名、タイプ名、パーティション列、数量、および _id 値列を設定できます。 _id の値がビジネスプライマリキーに設定されている場合、対応するビジネスプライマリキー列を選択する必要があります。

e. 同期オブジェクトを設定したら、詳細設定に進みます。

5. 詳細設定

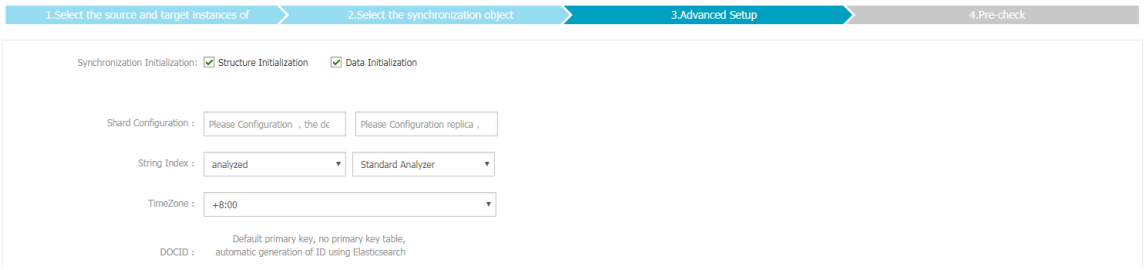
主な設定

a. 同期の初期化：**[構造体の初期化]** と **[データの初期化]** を選択することを推奨します。これにより、自動的にインデックスが作成され、データが初期化されます。 [スキーマの初期化] を選択しない場合、同期する前に ES のインデックスのマッピングを手動で定義する必要が

あります。 [全データの初期化] を選択しない場合、増分 DTS データ同期の開始時刻は同期が開始される時刻です。

b. シャード設定：デフォルトで 5 つのパーティションと 1 つのレプリカがあります。 設定の調整後、すべてのインデックスがこの設定に従ってパーティションを定義します。

c. 文字列インデックス：文字列を選択できるアナライザーです。 デフォルトは、**[Standard Analyzer]** です。 その他の値は、**[Simple Analyzer]**、**[Whitespace Analyzer]**、**[Stop Analyzer]**、**[Keyword Analyzer]**、**[English Analyzer]**、**[Fingerprint Analyzer]** です。 すべてのインデックスの文字列フィールドは、この設定に従ってアナライザーを定義します。



d. **Time Zone**：ES インスタンスに同期している時間フィールドが保存されている場所です。 中国のデフォルトのタイムゾーンは UTC (UTC + 8) です。

6. 事前チェック

同期タスクの設定が完了したら、DTS は事前チェックを実行します。 事前チェックが確認されたら、**[開始]** をクリックして同期タスクを開始します。

同期タスクが開始したら、同期ジョブリストに移動し、タスクのステータスが **[同期初期化]** かどうかを確認します。 初期化にかかる時間は、ソースインスタンス内の同期オブジェクトのデータ量によって異なります。 初期化が完了すると、同期インスタンスのステータスは **[同期中]** になります。 ソースインスタンスとターゲットインスタンス間に同期リンクが確立されます。

7. データの検証

上記のすべての手順を完了したら、ES コンソールにログインして、ES インスタンスに作成されたインデックスと同期データを確認します。

2.1.3 Canal による Alibaba Cloud Elasticsearch とのデータ同期

このトピックでは、Canal を使用して、ApsaraDB RDS for MySQL の増分データを Alibaba Cloud Elasticsearch と同期させる方法について説明します。

Canal を使用する前に、次の前提条件を満たしていることを確認してください。

:

ApsaraDB RDS for MySQL、Alibaba Cloud Elasticsearch、Elastic Compute Service (ECS) を有効化するとき、これらのサービスに同じリージョン、ゾーン、**VPC** ネットワーク、**VSwitch**、セキュリティ グループが指定されていることを確認します。

- ApsaraDB RDS for MySQL

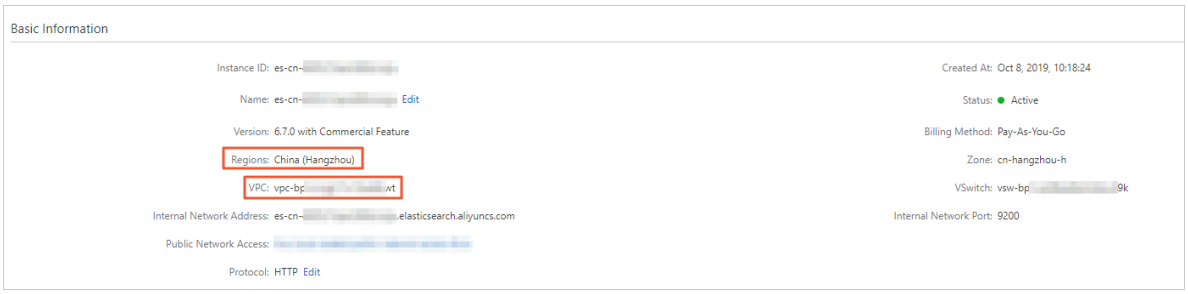
  ApsaraDB RDS for MySQL を使用して、同期対象のソースデータと増分データを保存します。ApsaraDB RDS for MySQL を有効化する方法の詳細は、「#unique_18」をご参照ください。次の図に、このトピックで使用する ApsaraDB RDS for MySQL の設定を示します。

- canal.adapter-1.1.4.tar.gz とcanal.deployer-1.1.4.tar.gz

  Canal パッケージです。 Canal は、GitHub オープンソースの extract-transform-load (ETL) ツールです。このツールを使用してデータベースログを解析し、同期対象の増分データを取得します。 詳細は、「Canal」をご参照ください。

- Alibaba Cloud Elasticsearch

  Alibaba Cloud Elasticsearch を使用して、同期対象の増分データを受信します。 Alibaba Cloud Elasticsearch を有効化する方法の詳細は、「#unique_9」をご参照ください。 次の図に、このトピックで使用する Elasticsearch の設定を示します。

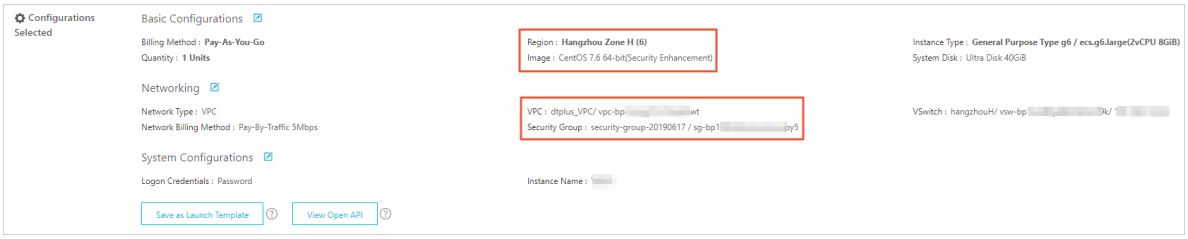


- Alibaba Cloud ECS

  Alibaba Cloud ECS を使用して、ApsaraDB RDS for MySQL と Alibaba Cloud Elasticsearch を接続します。 また、Canal deployer と Canal adapter を Alibaba Cloud ECS にデプロイし

ます。 Alibaba Cloud ECS を有効化する方法の詳細は、「#unique_19」をご参照ください。

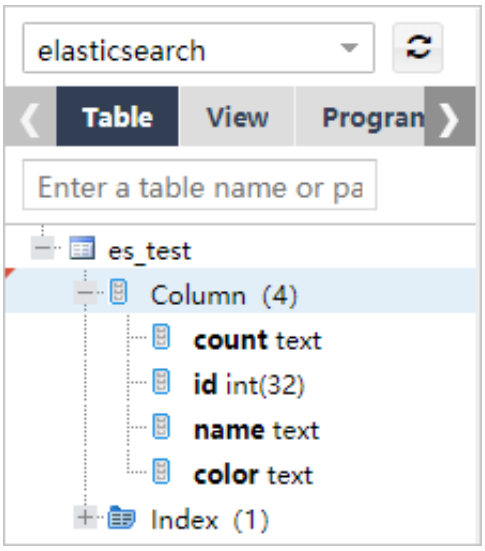
次の図に、このトピックで使用する ECS の設定を示します。



テーブル、フィールド、インデックスの作成

1. ApsaraDB RDS for MySQL にテーブルを作成し、このテーブルにフィールドを追加します。

   このトピックでは、テーブル **es_test** を作成します。 次の図に、テーブルに追加するフィールドを示します。



2. Elasticsearch インスタンスにインデックスを作成し、マッピングを設定します。

   #unique_15を実行し、[**Dev Tools**] ページ の [**Console**] で次のコマンドを実行してインデックスを作成し、マッピングを設定します。

   :

   次のコマンドで指定されているテーブル名、フィールド名、フィールドタイプは、MySQL テーブルのテーブル名、フィールド名、フィールドタイプと同じでなければなりません。

```
PUT es_test? include_type_name=true
{

  "settings" : {
    "index" : {
      "number_of_shards" : "5",
      "number_of_replicas" : "1"
    }
  },
```

```
  "mappings" : {
    "_doc" : {
      "properties" : {
       "count": {
          "type": "text"
       },
       "id": {
          "type": "long"
       },
        "name" : {
          "type" : "text"
        },
        "color" : {
          "type" : "text"
        }
      }
    }
  }
}
```

テーブルとマッピングが正常に作成されると、次の結果が返されます。

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

MySQL のインストール

1. Alibaba Cloud ECS インスタンスに接続します。
2. MySQL ソースインストールパッケージをダウンロードします。

   ```
   wget http://dev.mysql.com/get/mysql57-community-release-el7-11.noarch.rpm
   ```

3. MySQL ソースをインストールします。

   ```
   yum -y install mysql57-community-release-el7-11.noarch.rpm
   ```

4. MySQL ソースが正常にインストールされたかどうかを確認します。

   ```
   yum repolist enabled | grep mysql. *
   ```

   MySQL ソースが正常にインストールされると、次の結果が返されます。

   ```
   [root@VM01 ~]# yum repolist enabled | grep mysql.*
   mysql-connectors-community/x86_64 MySQL Connectors Community                118
   mysql-tools-community/x86_64      MySQL Tools Community                      95
   mysql57-community/x86_64          MySQL 5.7 Community Server                364
   ```

5. MySQL サーバーをインストールします。

   ```
   yum install mysql-community-server
   ```

6. MySQL サービスを起動し、MySQL サービスのステータスを確認します。

   ```
   systemctl start mysqld.service
   ```

```
systemctl status mysqld.service
```

MySQL サービスが起動されていると、次の結果が返されます。

```
[root@VM01 ~]# systemctl start mysqld.service
systemctl status mysqld.service
[root@VM01 ~]# systemctl status mysqld.service
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2019-09-05 11:12:21 CST; 8ms ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
  Process: 1683 ExecStart=/usr/sbin/mysqld --daemonize --pid-file=/var/run/mysqld/mysqld.pid $MYSQLD_OPTS (code=ex
/SUCCESS)
  Process: 1609 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
 Main PID: 1687 (mysqld)
   CGroup: /system.slice/mysqld.service
           └─1687 /usr/sbin/mysqld --daemonize --pid-file=/var/run/mysqld/mysqld.pid

Sep 05 11:12:17 VM01 systemd[1]: Starting MySQL Server...
Sep 05 11:12:21 VM01 systemd[1]: Started MySQL Server.
```

7. ApsaraDB RDS for MySQL データベースに接続します。

:

- 次のコマンドを実行して ApsaraDB RDS for MySQL データベースに接続する前に、ECS インスタンスのプライベート **IP** アドレスを ApsaraDB RDS for MySQL ホワイトリストに追加する必要があります。 詳細は、「#unique_20」をご参照ください。
- Canal を使用するには、MySQL バイナリログモードを有効にする必要があります。 ApsaraDB RDS for MySQL の場合、このモードはデフォルトで有効になっています。 次のコマンドを実行すると、バイナリログモードのステータスを照会できます。

  ```
  show variables like '%log_bin%';
  ```

  バイナリログモードが有効になっている場合、次の結果が返されます。

```
mysql> show variables like '%log_bin%';
+---------------------------------+-----------------------------------------------+
| Variable_name                   | Value                                         |
+---------------------------------+-----------------------------------------------+
| log_bin                         | ON                                            |
| log_bin_basename                | /home/mysql/data3001/mysql/mysql-bin          |
| log_bin_index                   | /home/mysql/data3001/mysql/master-log-bin.index |
| log_bin_trust_function_creators | ON                                            |
| log_bin_use_v1_row_events       | ON                                            |
| sql_log_bin                     | ON                                            |
+---------------------------------+-----------------------------------------------+
6 rows in set (0.00 sec)
```

```
mysql -h<hostname> -P<port> -u<username> -p<password> -D<database>
```

変数	説明
<hostname>	ApsaraDB RDS for MySQL インスタンスのイントラネットアドレス。 イントラネットアドレスの情報は、インスタンスの [基本情報] で確認できます。

変数	説明
<port>	ApsaraDB RDS for MySQL インスタンスの内部ポート。 デフォルトのポート番号は、**3306** です。 内部ポートの情報は、インスタンスの [基本情報] ページで確認できます。
<username>	ApsaraDB RDS for MySQL データベースのユーザー名。 アカウント情報は、インスタンスの [アカウント管理] ページで確認できます。 使用可能なアカウントがない場合、アカウントを作成する必要があります。 詳細は、「#unique_21」をご参照ください。
<database>	ApsaraDB RDS for MySQL データベースの名前。 データベース名は、インスタンスの [データベース管理] ページで確認できます。 使用可能なデータベースがない場合、データベースを作成する必要があります。 詳細は、「#unique_21」をご参照ください。
<password>	ApsaraDB RDS for MySQL データベースのパスワード。

コマンド例：

```
mysql -hrm-bp1u1xxxxxxxxx6ph.mysql.rds.aliyuncs.com -P3306 -ues -pmima -Delasticsearch
```

ApsaraDB RDS for MySQL に接続されていると、次の結果が返されます。

```
[root@VM01 ~]# mysql -hrm-bp                h.mysql.rds.aliyuncs.com -P3306 -u   -p          3 -Delasticsearch
mysql: [Warning] Using a password on the command line interface can be insecure.
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 688823
Server version: 5.7.25-log Source distribution

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| elasticsearch      |
| mysql              |
| panqbi             |
| xuyongqbi          |
+--------------------+
5 rows in set (0.00 sec)

mysql> _
```

JDK のインストール

1. ECS インスタンスに接続し、使用可能な JDK パッケージを照会します。

   ```
   yum search java | grep -i --color JDK
   ```

2. バージョンを選択し、JDK をインストールします。 このトピックでは、**java-1.8.0-openjdk-devel.x86_64** を選択しています。

   ```
   yum install java-1.8.0-openjdk-devel.x86_64
   ```

3. 環境変数を設定します。

   a) etc フォルダーの profile ファイルを開きます。

   ```
   vi /etc/profile
   ```

   b) 次の環境変数をファイルに追加します。

   ```
   export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.71-2.b15.el7_2.x86_64
   export CLASSPATH=.:$JAVA_HOME/jre/lib/rt.jar:$JAVA_HOME/lib/dt.jar:$JAVA_HOME
   /lib/tools.jar
   export PATH=$PATH:$JAVA_HOME/bin
   ```

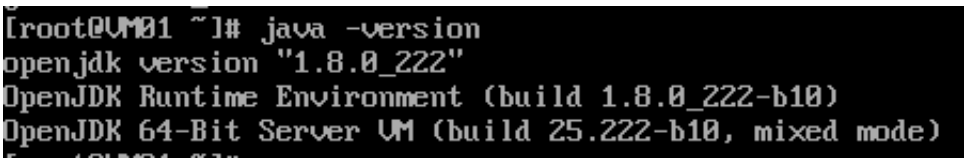
   c) `:wq` と入力してファイルを保存し、vi モードを終了します。続いて、次のコマンドを実行して変更を適用します。

   ```
   source /etc/profile
   ```

4. 次のコマンドを実行して、JDK が正常にインストールされたかどうかを確認します。

   - `java`
   - `javac`
   - `java -version`

   JDK が正常にインストールされていると、次の結果が返されます。

Canal サーバーのインストールと起動

1. ECS インスタンスに接続し、Canal deployer パッケージをダウンロードして解凍します。 このトピックでは、Canal deployer V1.1.4 を使用しています。

```
wget https://github.com/alibaba/canal/releases/download/canal-1.1.4/canal.deployer-1.1.4.tar.gz
```

2. canal.deployer-1.1.4.tar.gz パッケージを解凍します。

```
tar -zxvf canal.deployer-1.1.4.tar.gz
```

3. conf/example/instance.properties ファイルを変更します。

```
vi conf/example/instance.properties
```

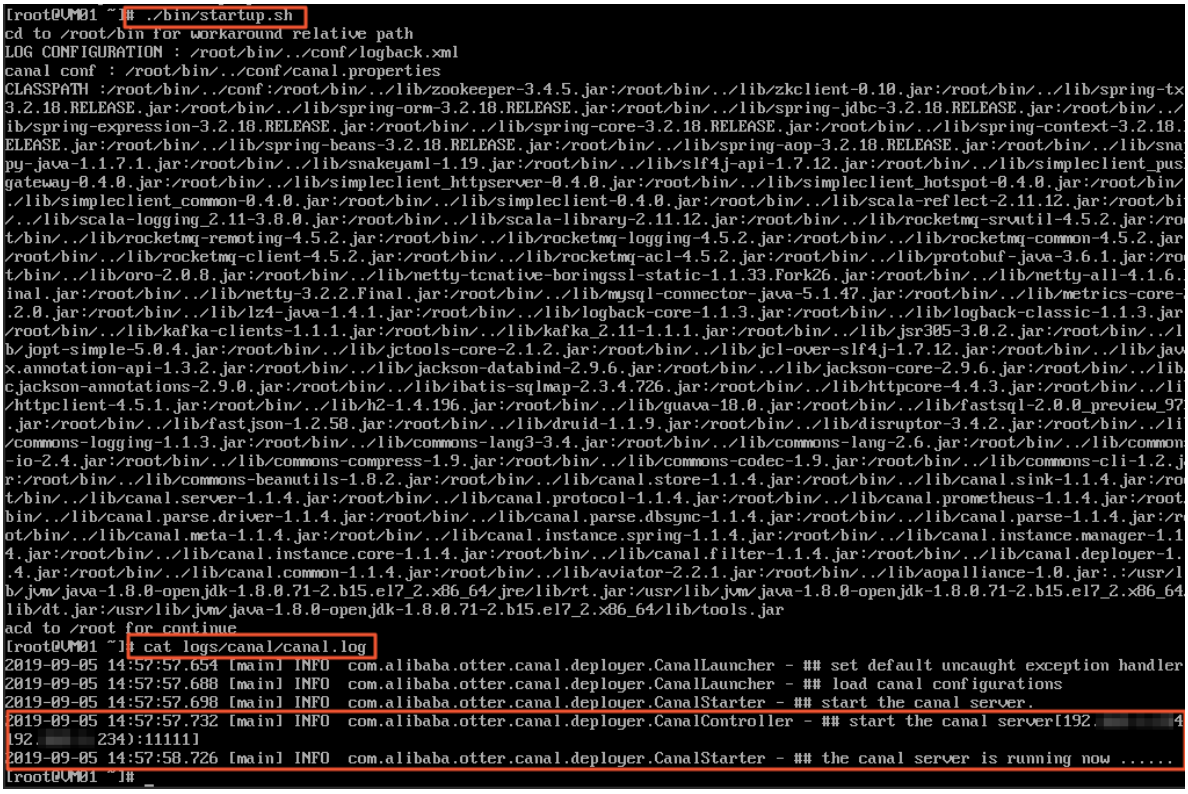
```
#################################################
## mysql serverId , v1.0.26+ will autoGen
# canal.instance.mysql.slaveId=0

# enable gtid use true/false
canal.instance.gtidon=false

# position info
canal.instance.master.address=rm-bp[redacted]d6ph.mysql.rds.aliyuncs.com:3306
canal.instance.master.journal.name=
canal.instance.master.position=
canal.instance.master.timestamp=
canal.instance.master.gtid=

# rds oss binlog
canal.instance.rds.accesskey=
canal.instance.rds.secretkey=
canal.instance.rds.instanceId=

# table meta tsdb info
canal.instance.tsdb.enable=true
#canal.instance.tsdb.url=jdbc:mysql://127.0.0.1:3306/canal_tsdb
#canal.instance.tsdb.dbUsername=canal
#canal.instance.tsdb.dbPassword=canal

#canal.instance.standby.address =
#canal.instance.standby.journal.name =
#canal.instance.standby.position =
#canal.instance.standby.timestamp =
#canal.instance.standby.gtid=

# username/password
canal.instance.dbUsername=[redacted]
canal.instance.dbPassword=[redacted]
canal.instance.connectionCharset = UTF-8
# enable druid Decrypt database password
canal.instance.enableDruid=false
```

パラメーター	説明
canal.instance.master.address	<ApsaraDB RDS for MySQL instance internal endpoint>:<internal port>。 必要な情報は、ApsaraDB RDS for MySQL インスタンスの [基本情報] ページで確認できます。 例：rm-bp1u1xxxxxxxxx6ph.mysql.rds.aliyuncs.com:3306

パラメーター	説明
canal.instance.dbUsername	ApsaraDB RDS for MySQL データベースのユーザー名。 アカウント情報は、 ApsaraDB RDS for MySQL インスタンスの [アカウント管理] ページで確認できます。
canal.instance.dbPassword	ApsaraDB RDS for MySQL データベースのパスワード。

4. **:wq** と入力してファイルを保存し、vi モードを終了します。
5. Canal サーバーを起動し、Canal ログを表示します。

```
./bin/startup.sh
cat logs/canal/canal.log
```



Canal adapter のインストールと起動

1. ECS インスタンスに接続し、Canal adapter パッケージをダウンロードして解凍します。 このトピックでは、Canal adapter V1.1.4 を使用しています。

```
wget https://github.com/alibaba/canal/releases/download/canal-1.1.4/canal.adapter-1.1.4.tar.gz
```

2. canal.adapter-1.1.4.tar.gz パッケージを解凍します。

```
tar -zxvf canal.adapter-1.1.4.tar.gz
```

3. conf/application.yml ファイルを変更します。

```
vi conf/application.yml
```

```
canal.conf:
  mode: tcp # kafka rocketMQ
  canalServerHost: 127.0.0.1:11111
#  zookeeperHosts: slave1:2181
#  mqServers: 127.0.0.1:9092 #or rocketmq
#  flatMessage: true
  batchSize: 500
  syncBatchSize: 1000
  retries: 0
  timeout:
  accessKey:
  secretKey:
  srcDataSources:
    defaultDS:
      url: jdbc:mysql://rm-bp[redacted]ph.mysql.rds.aliyuncs.com:3306/elasticsearch?useUnicode=true
      username: [redacted]
      password: p[redacted]
  canalAdapters:
  - instance: example # canal instance Name or mq topic name
    groups:
    - groupId: g1
      outerAdapters:
      - name: logger
#      - name: rdb
#        key: mysql1
#        properties:
#          jdbc.driverClassName: com.mysql.jdbc.Driver
#          jdbc.url: jdbc:mysql://127.0.0.1:3306/mytest2?useUnicode=true
#          jdbc.username: root
#          jdbc.password: 121212
#      - name: rdb
#        key: oracle1
#        properties:
#          jdbc.driverClassName: oracle.jdbc.OracleDriver
#          jdbc.url: jdbc:oracle:thin:@localhost:49161:XE
#          jdbc.username: mytest
#          jdbc.password: m121212
#      - name: rdb
#        key: postgres1
#        properties:
#          jdbc.driverClassName: org.postgresql.Driver
#          jdbc.url: jdbc:postgresql://localhost:5432/postgres
#          jdbc.username: postgres
#          jdbc.password: 121212
#          threads: 1
#          commitSize: 3000
#      - name: hbase
#        properties:
#          hbase.zookeeper.quorum: 127.0.0.1
#          hbase.zookeeper.property.clientPort: 2181
#          zookeeper.znode.parent: /hbase
      - name: es
        hosts: es-cn-v6[redacted]dp.elasticsearch.aliyuncs.com:9200
        properties:
          mode: rest # or transport
          security.auth: elastic:[redacted]3
          cluster.name: es-cn-v6[redacted]dp
```

パラメーター	説明
canal.conf.canalServerHost	Canal deployer のエンドポイント。 デフォルトの設定 127.0.0.1:11111 を使用します。
canal.conf.srcDataSources.defaultDS.url	jdbc:mysql://<ApsaraDB RDS for MySQL instance internal endpoint>:<internal port>/<database name>? useUnicode=true 。 必要な情報は、ApsaraDB RDS for MySQL インスタンスの [基本情報] ページで確認できます。 例：jdbc:mysql://rm-bp1xxxxxxxxxnd6ph.mysql.rds.aliyuncs.com:3306/elasticsearch?useUnicode=true
canal.conf.srcDataSources.defaultDS.username	ApsaraDB RDS for MySQL データベースのユーザー名。 アカウント情報は、ApsaraDB RDS for MySQL インスタンスの [基本情報] ページで確認できます。
canal.conf.srcDataSources.defaultDS.password	ApsaraDB RDS for MySQL データベースのパスワード。
canal.conf.canalAdapters.groups.outerAdapters.hosts	name:es を見つけ、**hosts** を <Alibaba Cloud Elasticsearch instance internal network address>:<internal network port> に置き換えます。 必要な情報は、Elasticsearch インスタンスの [#unique_22] ページで確認できます。 例：es-cn-v64xxxxxxxx3medp.elasticsearch.aliyuncs.com:9200
canal.conf.canalAdapters.groups.outerAdapters.mode	値を rest に設定します。

パラメーター	説明
canal.conf.canalAdapters.groups.outerAdapters.properties.security.auth	値を <Alibaba Cloud Elasticsearch instance username>:<password> に設定します。 例：elastic:es_password
canal.conf.canalAdapters.groups.outerAdapters.properties.cluster.name	Alibaba Cloud Elasticsearch インスタンスの ID。 ID は、Elasticsearch インスタンスの [#unique_22] ページで確認できます。 例：es-cn-v64xxxxxxxx3medp

4. **:wq** と入力してファイルを保存し、vi モードを終了します。
5. 同じ手順で conf/es/*.yml ファイルを変更し、ApsaraDB RDS for MySQL から Alibaba Cloud Elasticsearch にマッピングするフィールドを指定します。

```
dataSourceKey: defaultDS
destination: example
groupId: g1
esMapping:
  _index: es_test
  _type: _doc
  _id: _id
  #pk: id
  sql: "select t.id as _id,t.id,t.count,t.name,t.color from es_test t"
  commitBatch: 3000
~
```

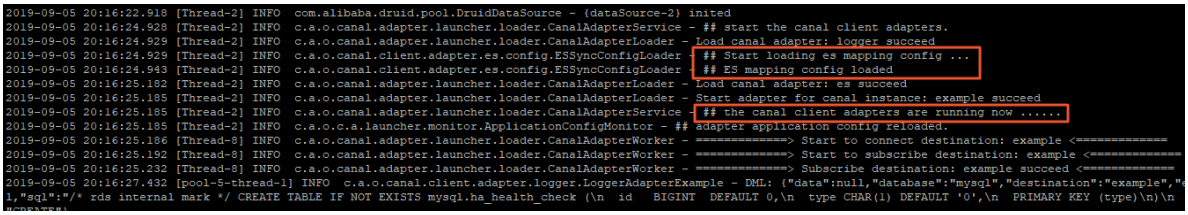
パラメーター	説明
esMapping._index	この値には、「テーブル、フィールド、インデックスの作成」で Elasticsearch インスタンスに作成したインデックス名を設定します。 このトピックでは、値に **es_test** を設定しています。
esMapping._type	この値には、「テーブル、フィールド、インデックスの作成」で Elasticsearch インスタンスに作成したインデックスタイプを設定します。 このトピックでは、値に **_doc** を設定しています。
esMapping._id	Elasticsearch インスタンスに同期させるドキュメントの ID。 独自のドキュメント ID を指定できます。 このトピックでは、値に **_id** を設定しています。

パラメーター	説明
esMapping.sql	Elasticsearch インスタンスに同期させるフィールドを取得するための SQL 文。 このトピックでは、`select t.id as _id,t.id,t.count,t.name,t.color from es_test t;` 文を使用しています。

6. Canal adapter サービスを起動し、Canal ログを表示します。

```
./bin/startup.sh
cat logs/adapter/adapter.log
```

Canal adapter サービスが起動されていると、次の結果が返されます。



Canal の増分データ同期機能の検証

1. ApsaraDB RDS for MySQL データベースのテーブル **es_test** に対して、データの追加、変更、または削除を行います。

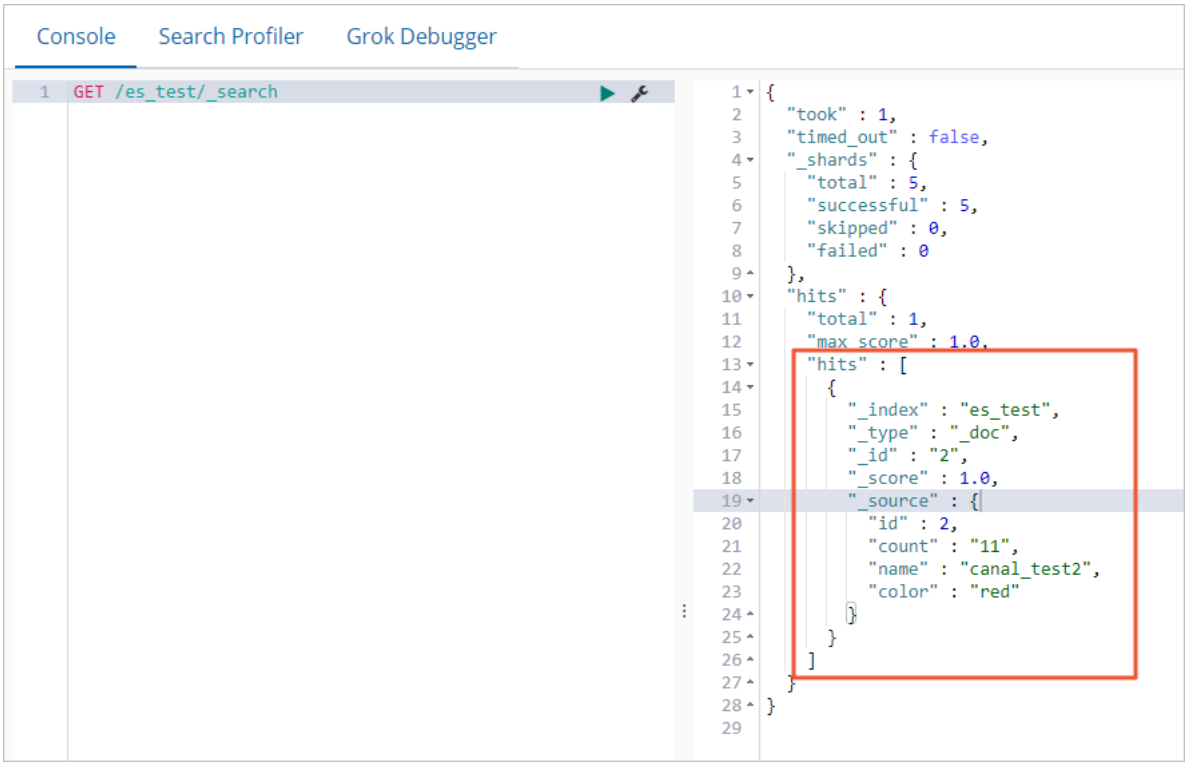
```
insert `elasticsearch`.`es_test`(`count`,`id`,`name`,`color`) values('11',2,'canal_test2','
red');
```

2. Alibaba Cloud Elasticsearch コンソールにログインし、#unique_15を実行します。

3. Kibana コンソールの [**Dev Tools**] ページの [**Console**] で、同期対象の増分データを照会します。

```
GET /es_test/_search
```

増分データが正常に同期されていると、次の結果が返されます。



3 データ移行

3.1 ユーザー作成の Logstash インスタンスを使用して、データを Alibaba Cloud Elasticsearch と同期

Logstash は、データソースからデータを動的に取り込むことができるオープンソースのデータ取り込みエンジンです。 Logstash では、カスタムルールを使用して、さまざまなデータソースから取り込まれたデータをフィルタリングし、そのデータをターゲットサービスに出力できます。 このトピックでは、Elastic Compute Service (ECS) に Logstash をデプロイする方法について説明し、Logstash から Alibaba Cloud Elasticsearch (ES) にデータを移行する例を示します。

前提条件

Logstash をデプロイして使用する前に、Alibaba Cloud ECS インスタンスと Elasticsearch インスタンスを作成し、設定する必要があります。

Alibaba Cloud Elasticsearch インスタンスの作成と設定

1. #unique_9を実行します。

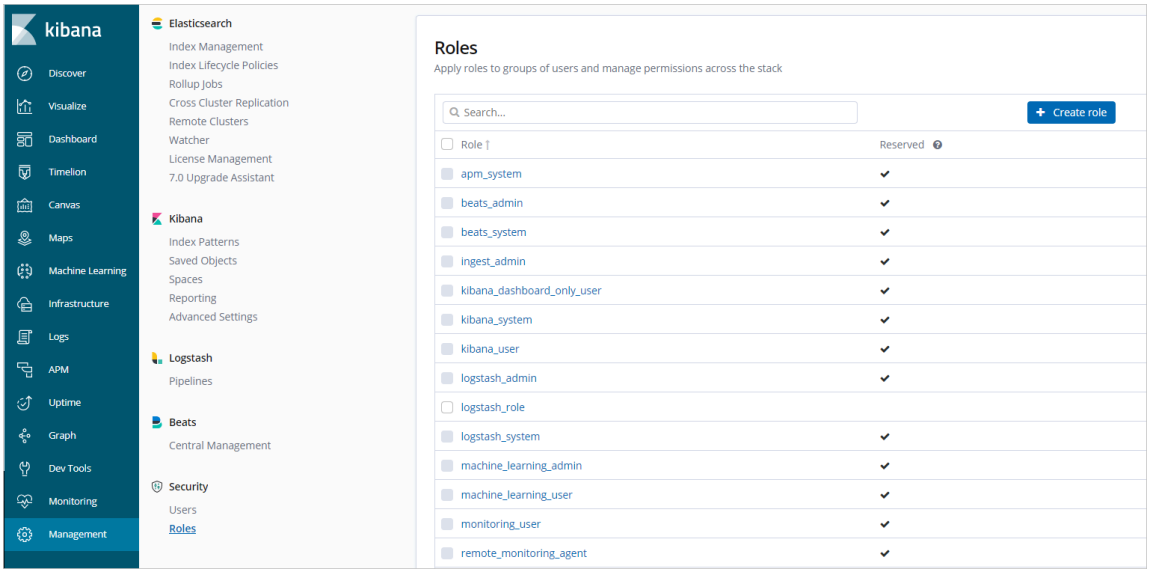
   この例では、Elasticsearch インスタンスのバージョンは V6.7.0 です。 次の図に Elasticsearch インスタンスの設定を示します。

2. Alibaba Cloud Elasticsearch コンソールにログインし、自動インデックスを有効化します。

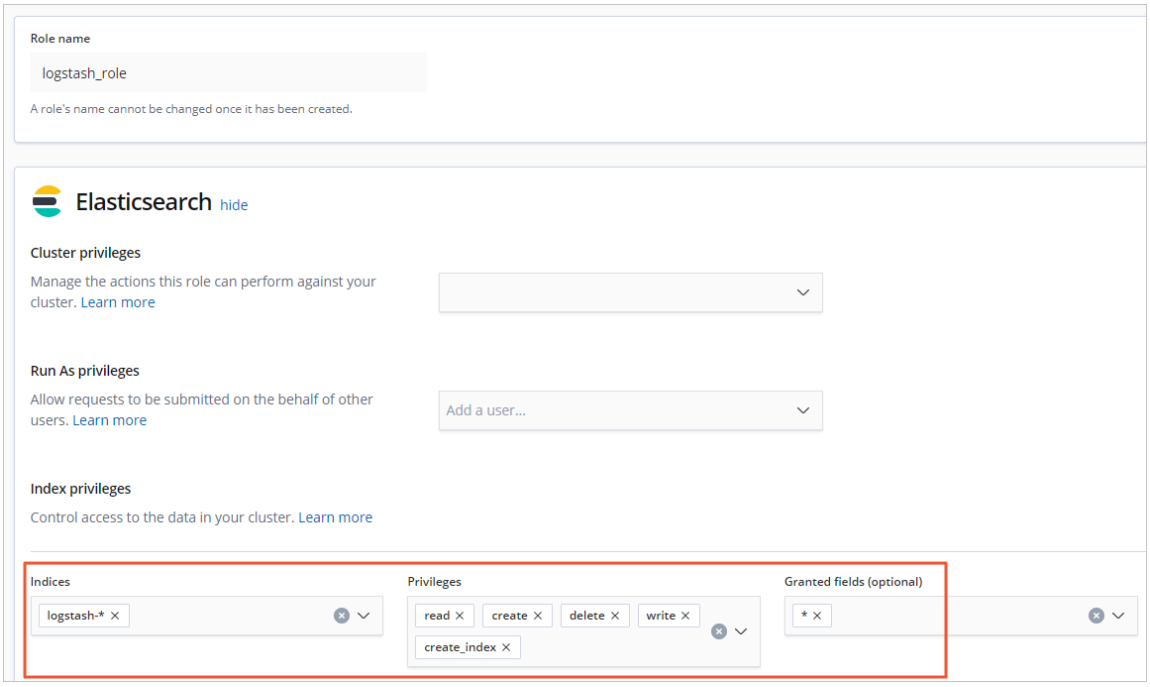
3. Elasticsearch インスタンスの Kibana コンソールにログインし、読み書き権限が付与されているロールを追加します。**logstash-***

   a. #unique_15を行います。

   b. **[Management]** > **[Roles]** > **[Create role]** を選択します。



   c. **[Create role]** ページで、関連するパラメーターを設定します。



パラメーター	説明
Role name	ロールの名前。
Indices	移行するインデックスファイル。 **logstash-*** を入力します。

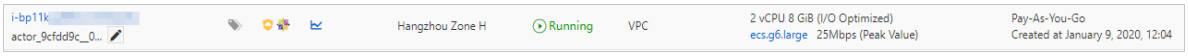
パラメーター	説明
Privileges	ロールに付与する権限。 read、write、create、delete、create_index 権限を追加します。
Granted fields	ロールがアクセスできるフィールド。 このパラメーターはオプションです。 この例では、* を入力します。

d. **[Create role]** をクリックしてロールを作成します。

Alibaba Cloud ECS インスタンスの作成と設定

1. #unique_19を実行します。 ECS インスタンスが Logstash インスタンスと Elasticsearch インスタンスにアクセスできることを確認してください。 要件を満たす購入済み ECS インスタンスを使用することもできます。

   次の図に ECS インスタンスの設定を示します。



   > 注：
   > Elasticsearch インスタンスと同じゾーンかつ同じ VPC ネットワークの ECS インスタンスを購入することを推奨します。 クラシックネットワークに接続された ECS インスタンスを購入することもできます。 ただし、クラシックネットワークが Elasticsearch インスタンスの VPC ネットワークに接続されていることを最初に確認する必要があります。

2. ECS インスタンスに JDK をインストールします。 JDK のバージョンは、V1.8 以降でなければなりません。

   JDK をインストールする方法の詳細については、「JDK のインストール」をご参照ください。

Logstash のインストール

1. Logstash V6.7.0 をダウンロードします。

   公式 Elasticsearch Web サイトにアクセスし、Elasticsearch バージョンと同じバージョンの Logstash をダウンロードします。 Logstash V6.7.0 をダウンロードすることを推奨します。

   ```
   wget https://artifacts.elastic.co/downloads/logstash/logstash-6.7.0.tar.gz
   ```

2. Logstash パッケージを解凍します。

   ```
   tar -xzvf logstash-6.7.0.tar.gz
   ```

Logstash での増分データの同期

1. ECS インスタンスに接続してから、Logstash ディレクトリに切り替えます。

   ```
   cd logstash-6.7.0
   ```

2. test という名前の .conf ファイルを作成します。

   ```
   touch test.conf
   ```

3. test.conf ファイルを設定します。 以下に例を示します。

   ```
   input {
      file {
         path => "/your/file/path/xxx"
         }
   }
   filter {
   }
   output {
     elasticsearch {
       hosts => ["http://instanceId.elasticsearch.aliyuncs.com:9200"]
       user => "user-name"
       password => "logstash-password"
     }
   }
   ```

パラメーター	説明
path	ログファイルのパス。 この例では、パスは /var/log/meaasges です。
hosts	Elasticsearch インスタンスのエンドポイント。 instanceId を Elasticsearch インスタンスの ID に置き換えます。 Elasticsearch インスタンスの [基本情報] ページでインスタンス ID を確認できます。 例：http://es-cn-45xxxxxxxxxxxxju.elasticsearch.aliyuncs.com:9200

パラメーター	説明
user	Elasticsearch インスタンスのユーザー名。 デフォルトのユーザー名は、elastic です。 カスタムアカウントを使用する場合、まずアカウントのロールを作成し、そのロールに必要な権限を付与する必要があります。 詳細については、「ユーザーの作成」と「ロールの作成」をご参照ください。 **:** ユーザー名は、二重引用符 (") のペアで囲む必要があります。 これにより、Logstash の起動時にユーザー名に含まれる特殊文字が原因で発生するエラーを回避できます。
password	Elasticsearch インスタンスのパスワード。 パスワードは、二重引用符 (") のペアで囲む必要があります。 Elasticsearch インスタンスの作成後にパスワードを変更することもできます。 **:** パスワードは、二重引用符 (") のペアで囲む必要があります。 これにより、Logstash の起動時にパスワードに含まれる特殊文字が原因で発生するエラーを回避できます。

Logstash には、さまざまな入力、フィルター、出力のプラグインがあります。 これらのプラグインは、データの簡単な取り込み、変換、出力に役立ちます。 詳細については、「設定ファイルの構造」をご参照ください。

4. **logstash** コマンドを実行します。

   手順 3 に従って、.conf ファイルを設定し、**logstash** コマンドを実行します。

   ```
   bin/logstash -f test.conf
   ```

   コマンドが実行されると、システムは自動的に Logstash を使用してログファイルから新しいコンテンツを取り込み、Elasticsearch インスタンスに出力します。 Logstash は、ログファイル内の変更を Elasticsearch インスタンスにインデックス付けします。

5. 結果を確認します。

   同期結果を確認する前に、ロールを作成し、**logstash-*** インデックスの管理をロールに許可する必要があります。 必要な権限は、read、write、create、delete、create_index です。詳細については、「ロールの作成」をご参照ください。

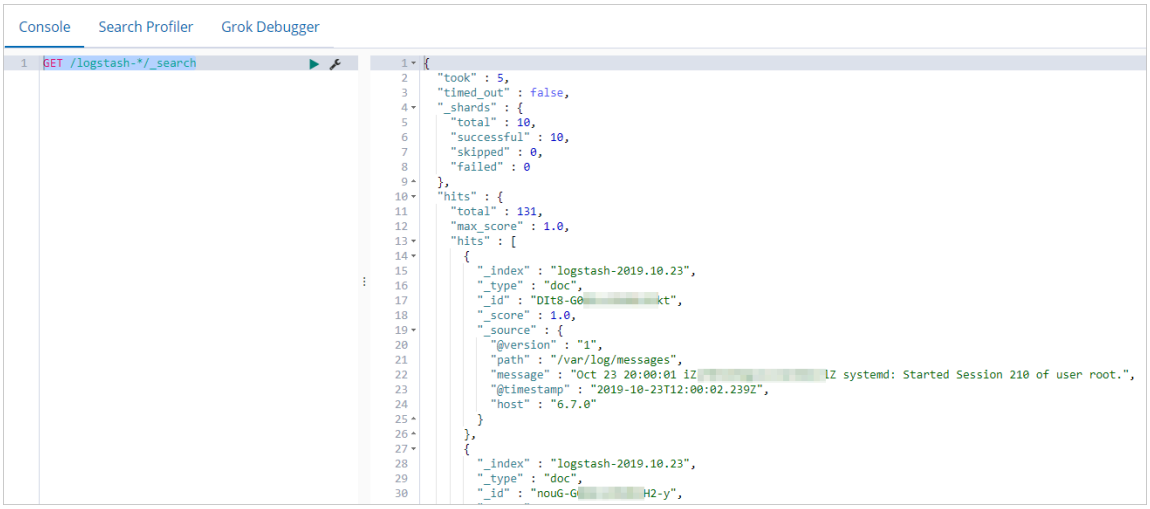
   a. Elasticsearch インスタンスの Kibana コンソールにログインします。

   b. 左側のナビゲーションペインで、**[Dev Tools]** を選択します。

   c. **[Dev Tools]** ページの **[Console]** タブで、次のコマンドを実行します。

   ```
   GET /logstash-*/_search
   ```

   コマンドの実行後、次の結果が返されます。



Logstash ノードのモニタリング

Logstash ノードをモニタリングし、モニタリングデータを収集するには、次の手順に従います。

1. Logstash ディレクトリの config フォルダーに切り替えます。

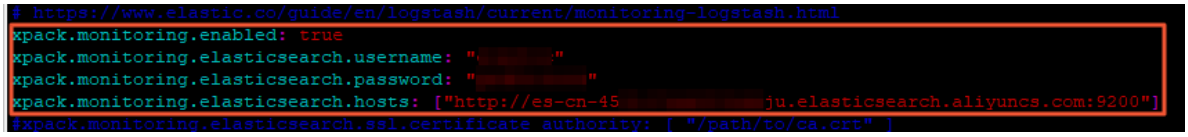
```
cd /logstash-6.7.0/config
```

2. logstash.yml ファイルを設定します。

   logstash.yml ファイルを開きます。

```
vim logstash.yml
```

   logstash.yml ファイルの次のパラメーターの注釈を削除し、これらのパラメーターに値を割り当てます。

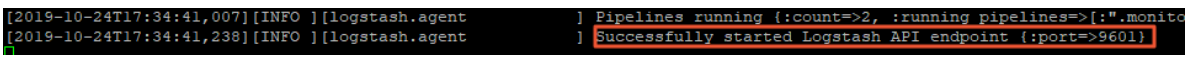


パラメーター	説明
xpack.monitoring.enable	このパラメーターを true に設定します。 デフォルト値は false です。
xpack.monitoring.elasticsearch.username	Logstash をモニタリングするユーザーを作成します。 詳細については、「ユーザーの作成」をご参照ください。 **:** elastic アカウントを使用することもできます。 ただし、システムのセキュリティリスクが生じる可能性があるため、本番環境では elastic アカウントを使用しないことを推奨します。
xpack.monitoring.elasticsearch.password	Logstash のモニタリングに使用されるユーザーのパスワード。
xpack.monitoring.elasticsearch.hosts	Elasticsearch インスタンスのエンドポイント。 例：http://es-cn-45xxxxxxxxxxxju.elasticsearch.aliyuncs.com:9200。

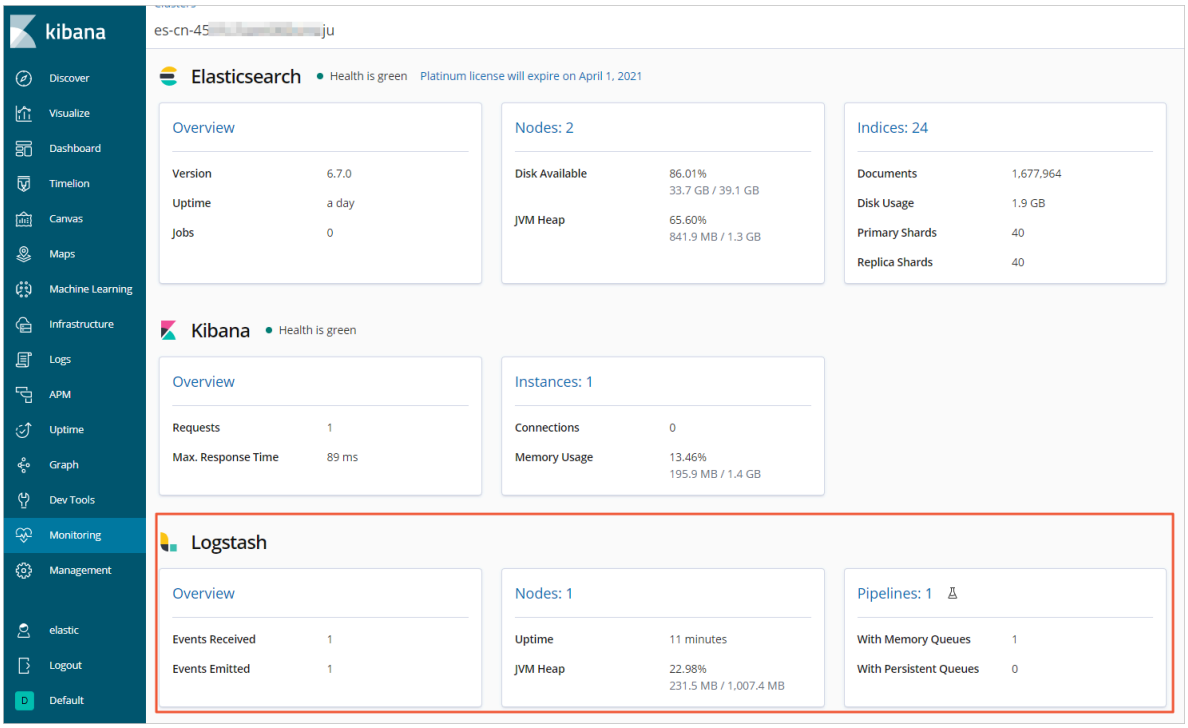
3. Logstash のディレクトリに戻り、Logstash サービスを起動します。

```
cd ../
bin/logstash -f test.conf
```

   Logstash の起動後、次の結果が返されます。



4. Elasticsearch インスタンスの Kibana コンソールにログインします。 左側のナビゲーションペインで **[Monitoring]** を選択して、Logstash モニタリングデータを表示します。



ユーザーの作成

このセクションでは、CLI または Kibana コンソールから Logstash をモニタリングするユーザーの作成方法について説明します。

:

デフォルトでは、logstash_system ユーザーは無効化されているので、logstash_system ユーザーを作成できません。 したがって、ogstash_system ロールを引き受けるユーザーを作成する必要があります。

CLI でのユーザーの作成

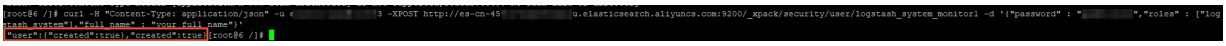
ECS インスタンスに接続し、次のコマンドを実行してユーザーを作成します。

```
curl -u elastic:es-password -XPOST http://instanceId.elasticsearch.aliyuncs.com:9200/_xpack/security/user/logstash_system_monitor -d '{"password" : "logstash-monitor-password","roles" : ["logstash_system"],"full_name" : "your full name"}'
```

パラメーター	説明
es-password	Elasticsearch インスタンスのパスワード。 このパスワードは、Kibana コンソールへのログインにも使用されます。

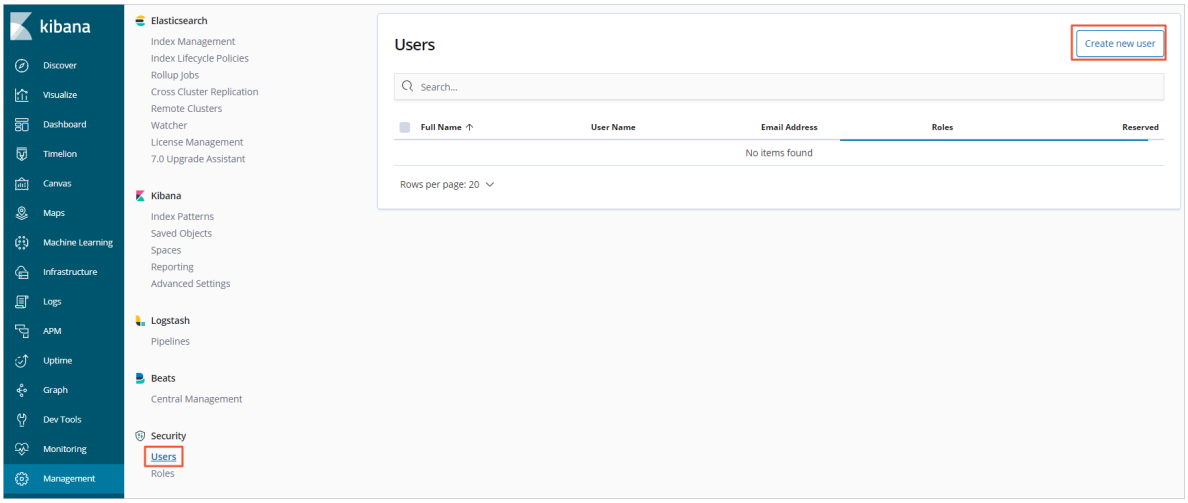
パラメーター	説明
instanceId	Elasticsearch インスタンスの ID。 Elasticsearch インスタンスの [基本情報] ページでインスタンス ID を確認できます。
logstash-monitor-password	logstash_system_monitor ユーザーのパスワード。
your full name	ユーザーのフルネーム。

ユーザーが作成されると、次の結果が返されます。

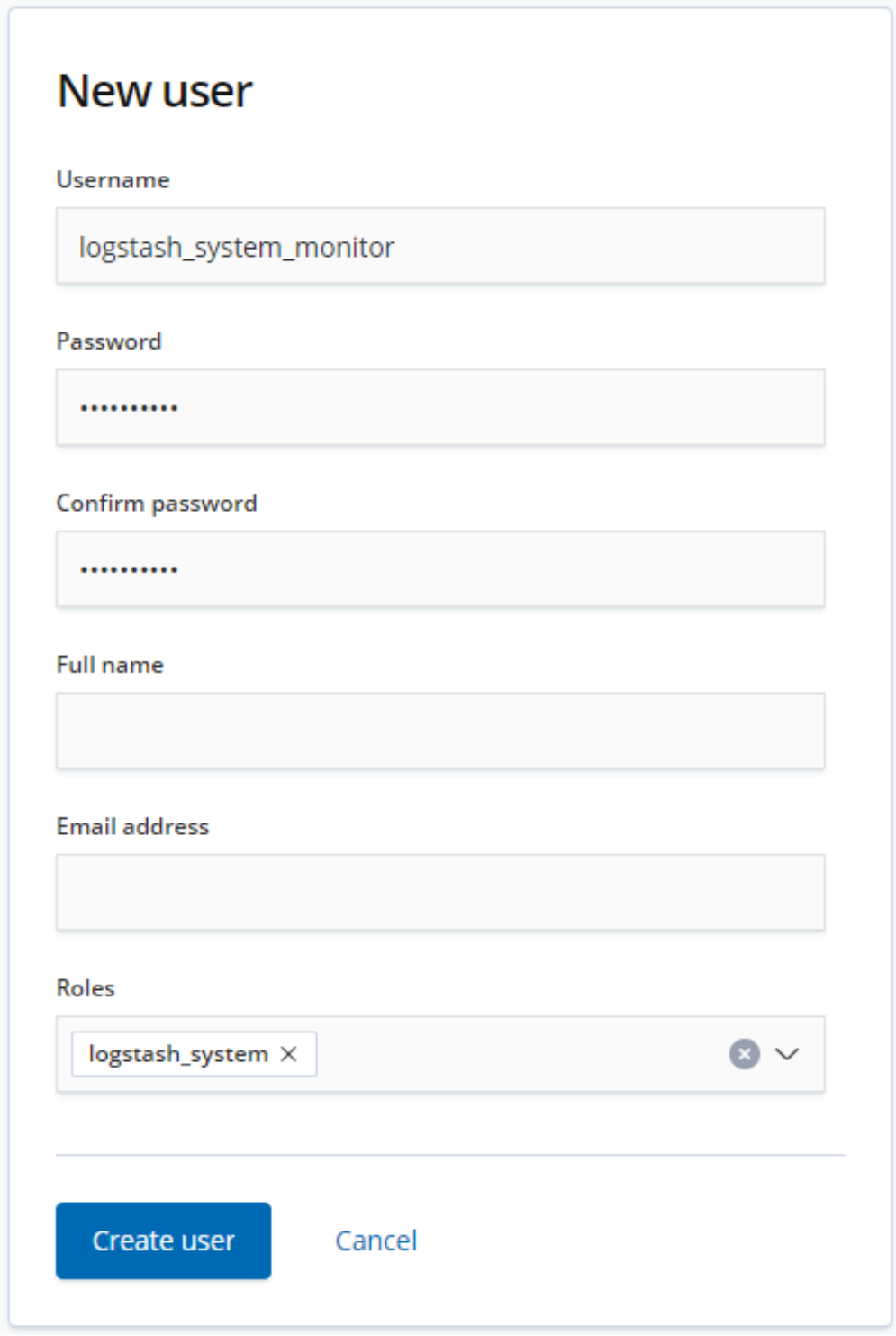


Kibana コンソールでのユーザーの作成

1. #unique_15を行います。
2. **[Management]** > **[Users]** > **[Create new user]** を選択します。



3. **[New user]** ページで、ユーザー情報を入力します。



パラメーター	説明
Username	ユーザーの名前。 カスタム名を入力することができます。 このセクションでは、ユーザー **logstash_system_monitor** を作成して、Logstash をモニタリングします。
Password	ユーザーのパスワード。

パラメーター	説明
Confirm password	確認用のパスワードをもう一度入力します。
Full name	ユーザーのフルネーム。 このパラメーターはオプションです。
Email address	ユーザーのメールアドレス。 このパラメーターはオプションです。
Roles	ユーザーが引き受けるロール。 **logstash_system** ロールを指定します。

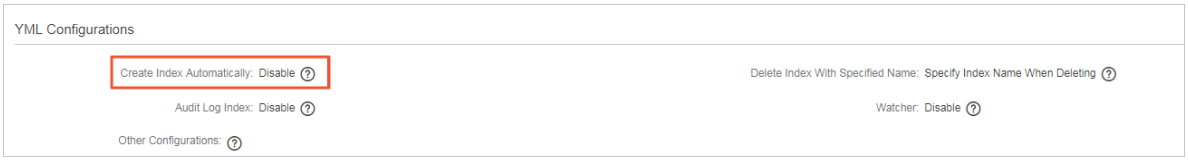
4. **[Create user]** をクリックして、ユーザーを作成します。

FAQ

- Q：Logstash を使用して Elasticsearch にデータを出力する前に、Elasticsearch インスタンスの自動インデックスを有効にする必要があるのはなぜですか。

  A：データのセキュリティを確保するため、Alibaba Cloud Elasticsearch はデフォルトで自動インデックスを無効にしています。

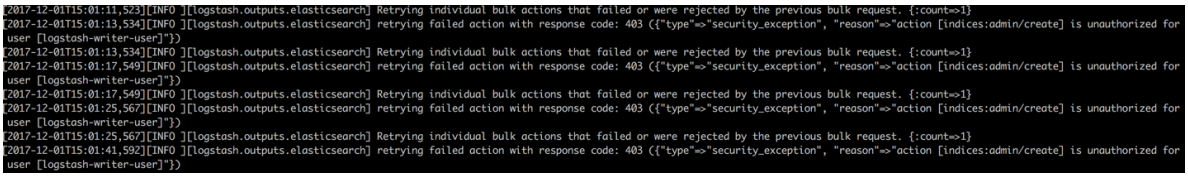
  Logstash は Create index 操作を呼び出さずに、データを Elasticsearch に送信します。 その後、Elasticsearch はデータに自動的にインデックスを付けます。 したがって、Logstash を使用してデータを Elasticsearch に出力する前に、Elasticsearch インスタンスの自動インデックスを有効にする必要があります。



:

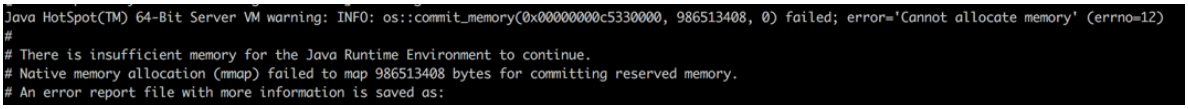
自動インデックスを有効にし、操作を確定すると、Elasticsearch インスタンスは再起動されます。 操作を確定する前に、インスタンスで実行されているワークロードが悪影響を受けないことを確認してください。

- Q：インデックスを作成する権限がないというメッセージが表示された場合、どうすればよいですか。



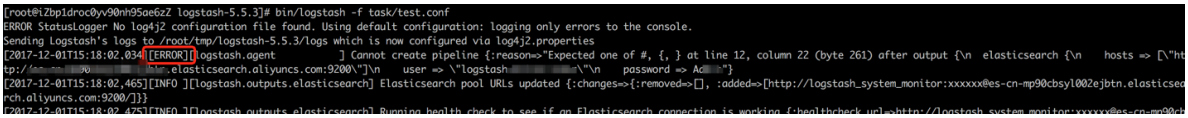
A：データの受信に使用される Elasticsearch アカウントのロールを確認する必要があります。 ロールに **write**、**delete**、**create_index** 権限があることを確認してください。

- Q：メモリ不足のエラーを解決するにはどうすればよいですか。



A：デフォルトでは、1 GB のメモリが Logstash に割り当てられています。 購入した ECS インスタンスに Logstash 用のメモリが不足している場合、config/jvm.options を変更して Logstash に割り当てられるメモリ量を減らします。

- Q：Logstash コマンドの実行時に次のエラーが発生します。このエラーを解決するにはどうすればよいですか。



A：test.conf ファイル内のユーザー名とパスワードに特殊文字が含まれていないことを確認する必要があります。 ユーザー名かパスワードに特殊文字が含まれている場合、二重引用符 (") のペアで囲む必要があります。

4 ビッグデータの同期

4.1 データの同期と移行

4.1.1 クラウドデータのインポート

Alibaba Cloud から ES へのデータのインポート (オフライン)

Alibaba Cloud は、豊富なクラウドストレージとデータベースプロダクトを提供しています。これらのプロダクトのデータの分析や検索をする場合、Data Integration を使用してください。Data Integration では、オフラインデータを **5** 分ごとに Elasticsearch に同期させることができます。

サポートされているデータソース

- Alibaba Cloud データベース (MySQL、PostgreSQL、SQL Server、PPAS、MongoDB、HBase)
- DRDS
- MaxCompute
- OSS
- Table Store
- 自社開発の HDFS、Oracle、FTP、DB2、および以前のクラウドデータベースの自社開発バージョン

> 注：
> データ同期により、パブリックネットワークのトラフィック料金が発生することがあります。

手順

オフラインデータをインポートするには、次の手順を実行します。

- VPC 内で Elasticsearch とやり取り可能な ECS インスタンスを準備します。 この ECS インスタンスはデータソースを取得し、ES データを書き込むジョブを実行します (このジョブは Data Integration によって一元的に実行されます)。
- Data Integration サービスを有効化して、実行可能ジョブリソースとして ECS インスタンスを Data Integration サービスに登録する必要があります。
- データ同期スクリプトを設定し、定期的に実行させます。

ステップ

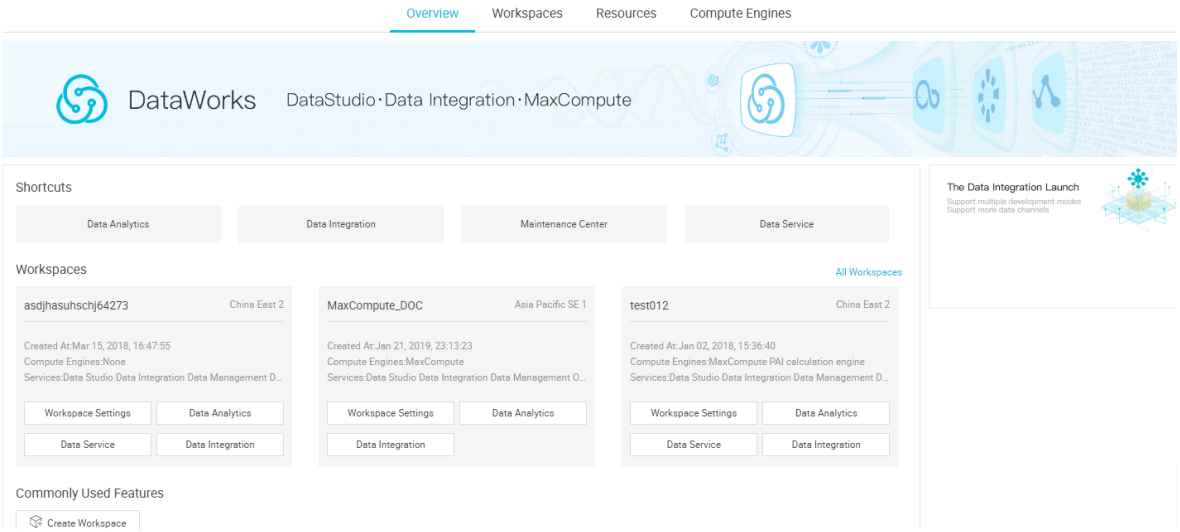
1. Elasticsearch サービスと同じ VPC にある ECS インスタンスを購入します。 パブリック IP アドレスを ECS インスタンスに割り当てるか、ECS インスタンスの Elastic IP アドレスを有効にします。 コストを抑えるため、既存の ECS インスタンスを使用できます。 ECS インスタンスの購入方法は、「#unique_19」をご参照ください。

   注：

   - CentOS 6、CentOS 7、AliyunOS を推奨します。
   - 追加された ECS インスタンスで MaxCompute または同期タスクを実行する必要がある場合、ECS インスタンスの現在の Python バージョンが 2.6 または 2.7 かどうかを確認します (CentOS 5 の Python バージョンは 2.4 ですが、他のオペレーティングシステムの Python バージョンは 2.6 以降)。
   - ECS インスタンスにパブリック IP アドレスが割り当てられていることを確認します。

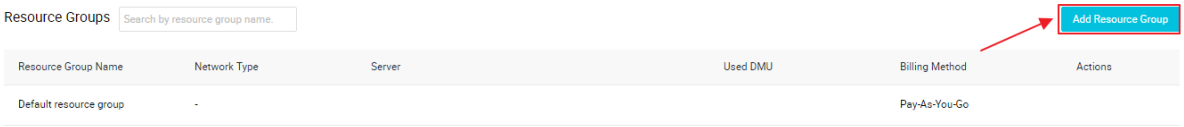
2. Data Integration コンソールにログインして、ワークベンチを開きます。

   Data Integration または DataWorks が有効化されている場合、次のように表示されます。



   Data Integration または DataWorks が有効化されていない場合、次のメッセージが表示されます。 手順に従って、Data Integration サービスを有効化します。 有料サービスなので、見積もり価格を予算と照らし合わせてチェックしてください。

3. Data Integration サービスの [Project Management-Scheduling Resource Management] ページに移動し、VPC の ECS インスタンスをスケジューリングリソースとして設定します。詳細は、「#unique_32」をご参照ください。



4. Data Integration サービスでデータ同期スクリプトを設定します。 設定手順は、「#unique_12」をご参照ください。 Elasticsearch の設定方法は、「#unique_14」をご参照ください。

> 注：
>
> - 同期スクリプトの設定は 3 つの部分に分けられます。Reader は、アップストリームデータソース (データ同期のクラウドプロダクト) の設定、Writer は ES の設定、setting はパケットロス率や最大同時性などの同期の設定です。
> - ES Writer の accessId と accessKey は、それぞれ Elasticsearch のユーザー名とパスワードです。

5. スクリプトを設定したら、データ同期ジョブを送信します。 ジョブの実行サイクルを設定し、**[OK]** をクリックします。

> 注：
>
> - 定期スケジュールを設定する場合、このポップアップウィンドウでジョブ開始時間、実行間隔、ジョブライフサイクルなどのパラメーターを設定します。
> - 設定したルールに従って、翌日の 00:00 に定期ジョブが実行されます。

6. 送信後、[O&M Center-Task Scheduling] ページに移動して送信されたジョブを見つけ、デフォルトのスケジューリングリソースから、設定したスケジューリングリソースに変更します。

リアルタイムデータのインポート

この機能は現在開発中です。今後利用可能になる予定です。

4.1.2 DataWorks による MaxCompute と Elasticsearch 間のデータ同期

Alibaba Cloud は、幅広いクラウドストレージとデータベースサービスを提供します。 これらのサービスに保存されているデータの分析や検索をする場合、Data Integration を使用してデータ

を Elasticsearch にレプリケートしてから、データのクエリや分析をします。 Data Integration を使用すると、最小 5 分間隔でデータをレプリケートできます。

> 注：
> データのレプリケートにより、パブリックネットワークトラフィックが生成され、課金が発生する可能性があります。

前提条件

オンプレミスデータの分析や検索をするには、次の手順に従います。

- #unique_34 、およびデータのインポートを行います。 Hadoop から MaxCompute へデータを移行してから、データを同期します。 この例では、以下のテーブルスキームとデータを使用します。

Column Name	Type
create_time	STRING
category	STRING
brand	STRING
buyer_id	STRING
trans_num	BIGINT
trans_amount	DOUBLE
click_cnt	BIGINT
pt	STRING

create_time	category	brand	buyer_id	trans_num	trans_amount	click_cnt	pt
2018-08-21 00:00:00	[illegible]	[illegible]A	null	null	null	null	1
2018-08-22 00:00:00	[illegible]	[illegible]B	null	null	null	null	1
2018-08-22 00:00:00	[illegible]	[illegible]C	null	null	null	null	1
	[illegible]	[illegible]A	null	null	null	null	1
2018-08-22 00:00:00	[illegible]	[illegible]D	null	null	null	null	1
2018-08-23 00:00:00	[illegible]	[illegible]B	null	null	null	null	1
2018-08-23 00:00:00	[illegible]	[illegible]A	null	null	null	null	1
2018-08-23 00:00:00	[illegible]	[illegible]E	null	null	null	null	1
2018-08-24 00:00:00	[illegible]	[illegible]G	null	null	null	null	1
2018-08-24 00:00:00	[illegible]	[illegible]F	null	null	null	null	1
2018-08-24 00:00:00	[illegible]	[illegible]A	null	null	null	null	1
2018-08-24 00:00:00	[illegible]	[illegible]G	null	null	null	null	1
2018-08-24 00:00:00	[illegible]	[illegible]C	null	null	null	null	1

- Data Integration でレプリケートされたデータを保存するための Elasticsearch インスタンスを作成します。

- Elasticsearch と同じ VPC を共有する ECS インスタンスを購入します。 この ECS インスタンスでデータを取得し、Elasticsearch タスクを実行します (これらのタスクは、Data Integration によって送信されます)。
- Data Integration を有効化し、タスクを実行可能なリソースとして ECS インスタンスを Data Integration に登録します。
- データ同期スクリプトを設定し、定期的に実行します。

手順

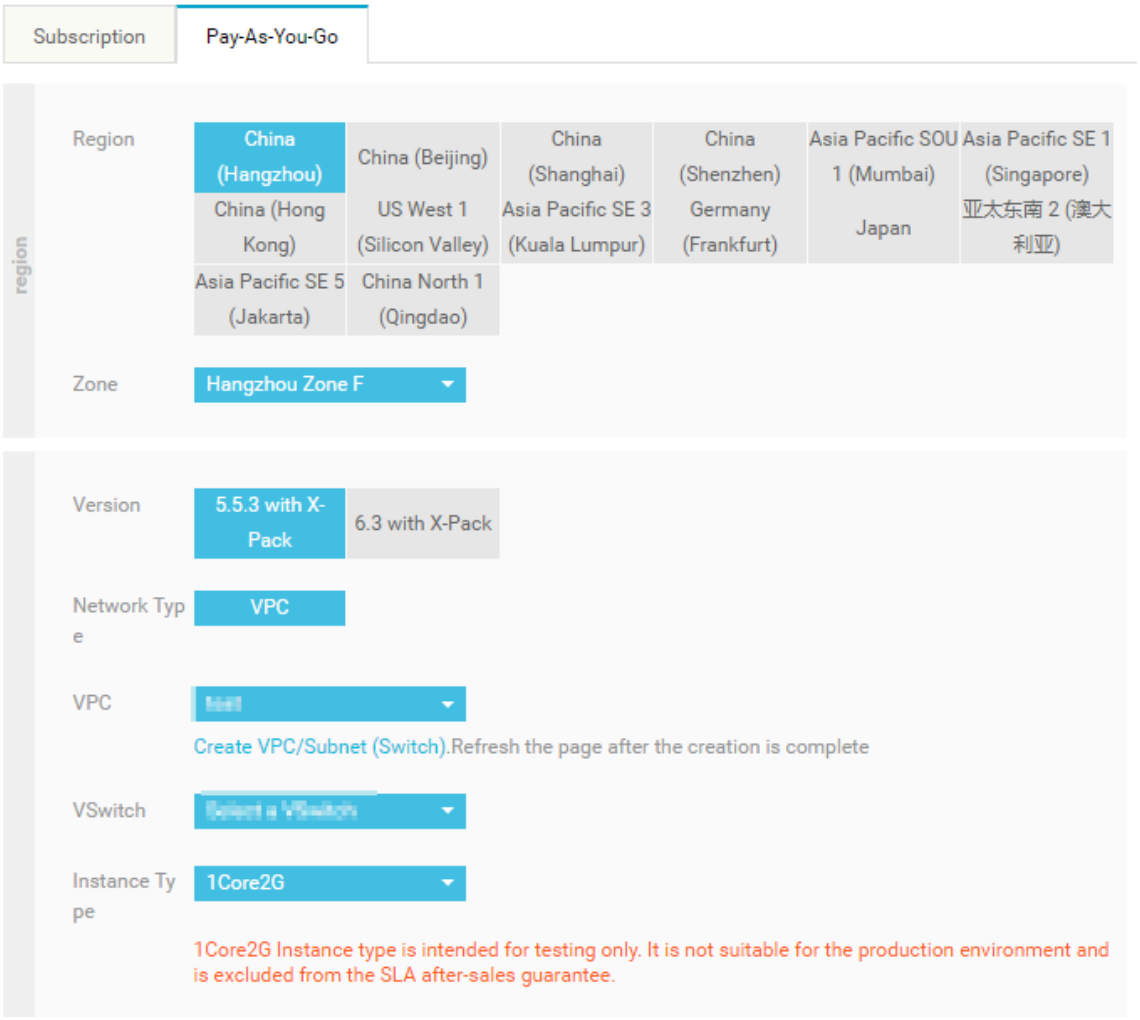
1. Elasticsearch インスタンスと ECS インスタンスの作成

   a. #unique_37を行います。 この例では、VPC を中国 **(**杭州**)** リージョンに作成します。 インスタンス名は **es_test_vpc**、VSwitch 名は **es_test_switch** です。

   b. Elasticsearch コンソールにログインし 、Elasticsearch インスタンスを作成します。

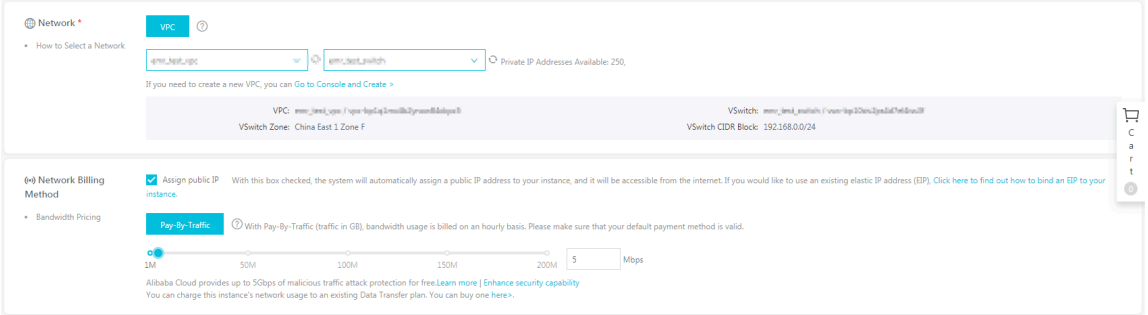
   注 **:**

前の手順で作成した VPC と同じリージョン、**VPC**、 **VSwitch** を選択するようにします。



c. Elasticsearch インスタンスと同じ VPC にある ECS インスタンスを購入し、パブリック IP アドレスを割り当てるか、EIP を有効化します。 コストを節約するため、要件を満たす既存の ECS インスタンスを使用することを推奨します。

この例では、杭州 **(**中国東部 1 **)** ゾーン **F** に ECS インスタンスを作成します。 **[64-bit CentOS 7.4]** と **[**パブリック **IP** の割り当て**]** を選択して、ネットワークを設定します (次図)。



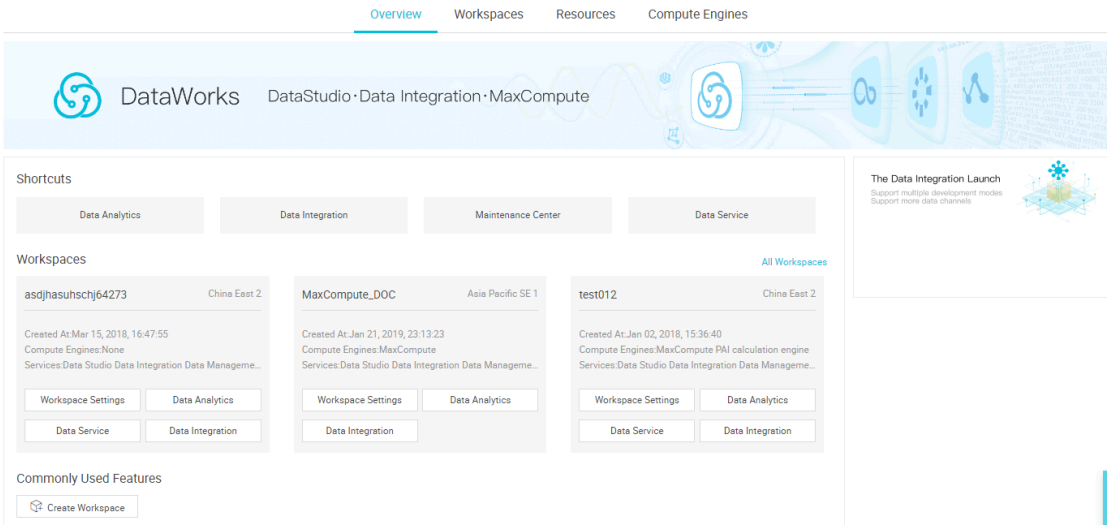
注：

- CentOS 6、CentOS 7、または Aliyun Linux を使用することを推奨します。
- ECS インスタンスで MaxCompute タスクまたはデータ同期タスクを実行する場合、ECS インスタンスの Python バージョンが 2.6 か 2.7 であることを確認する必要があります。 CentOS 5 をインストールすると、Python 2.4 もインストールされます。 他の CentOS バージョンの Python は 2.6 以降です。
- ECS インスタンスにパブリック IP アドレスが割り当てられていることを確認します。

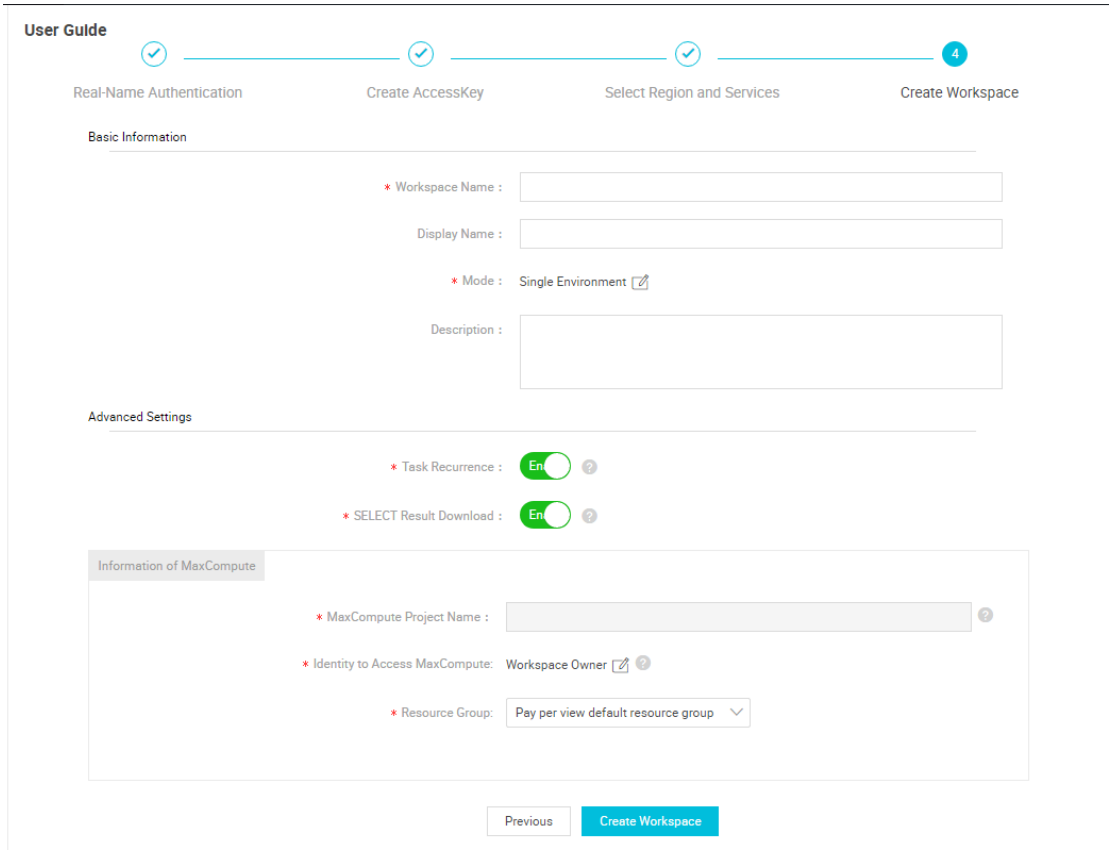
2. データ同期タスクの設定

   a. DataWorks コンソールにログインしてプロジェクトを作成します。 この例では、**bigdata_DOC** という名前の DataWorks プロジェクトを使用します。

   - Data Integration が既に有効化されている場合、次のページが表示されます。



   - Data Integration が有効化されていない場合、次のページが表示されます。Data Integration を有効化するには、次の手順を実行します。 このサービスを有効化すると、料金が発生します。料金設定ルールに基づいて見積もることができます。



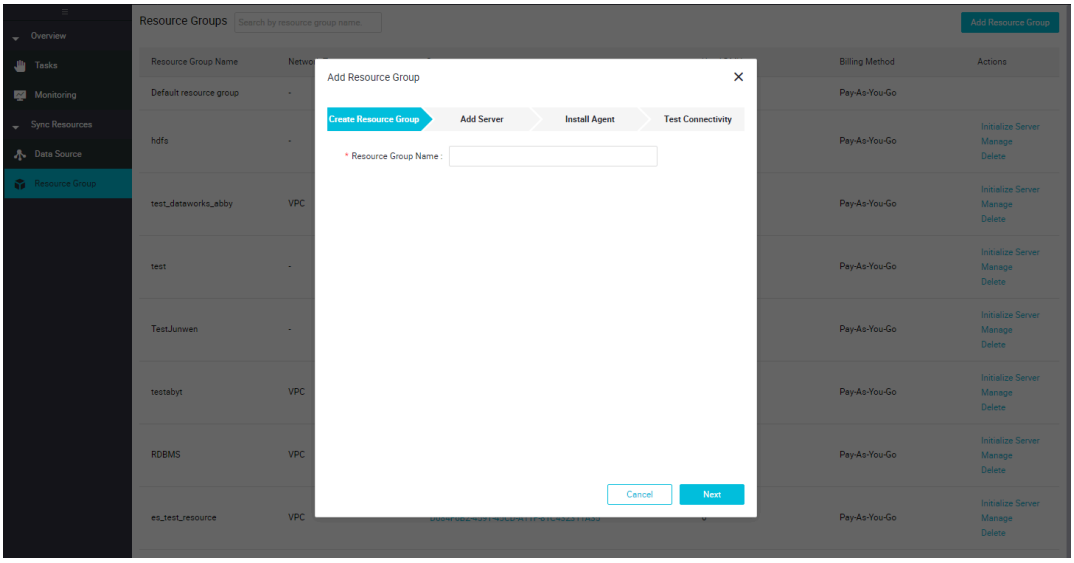
   b. DataWorks プロジェクトの下にある **[Data Integration]** をクリックします。

c. リソースグループを作成します。

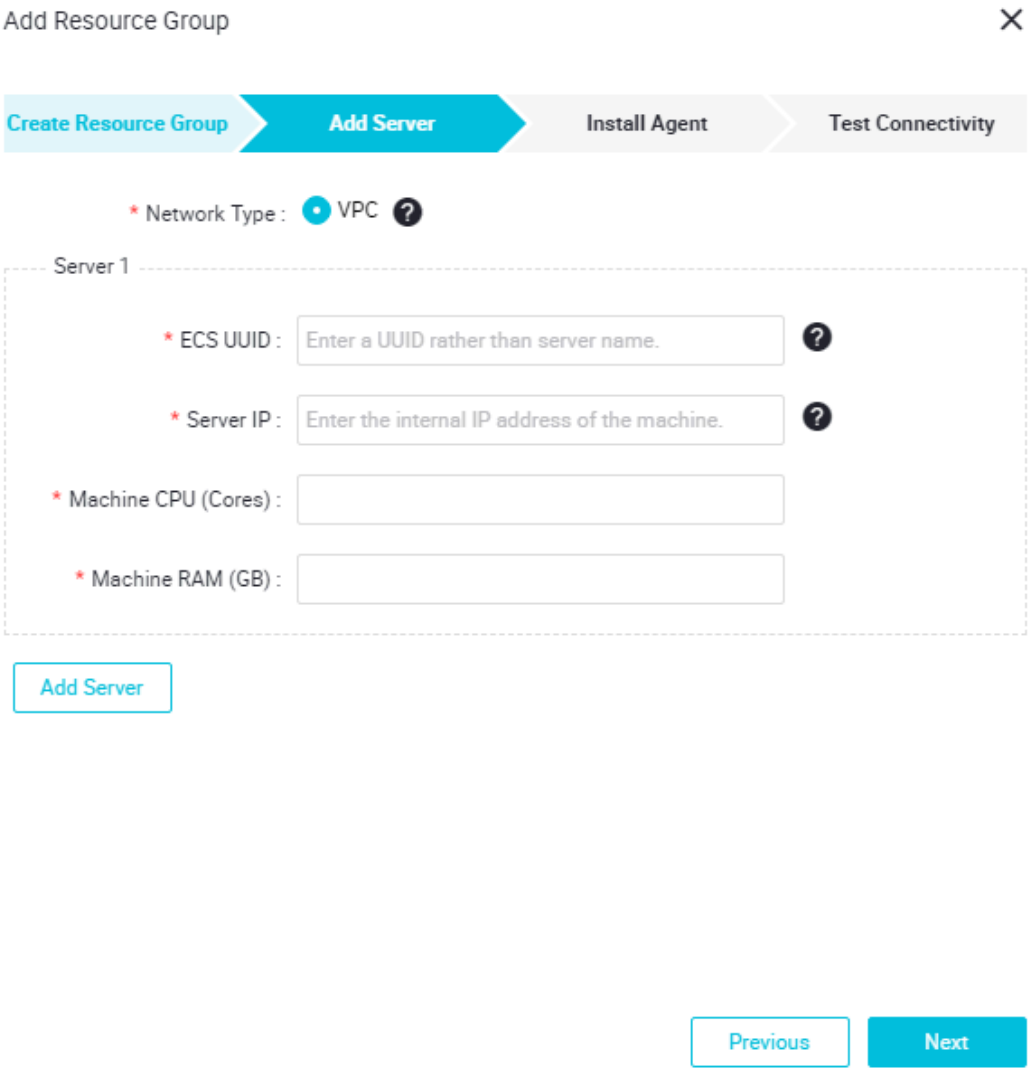
A. **[Data Integration]** ページの左側のナビゲーションウィンドウで、**[Resource Group]** を選択し、**[Add Resource Group]** をクリックします。

B. リソースグループを追加するには、次の手順に従います。

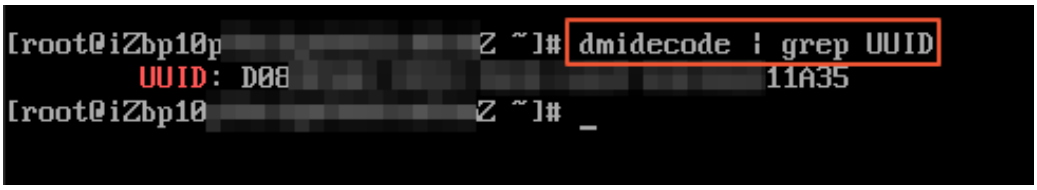
A. リソースグループを作成します。リソースグループ名を入力します。 この例では、リソースグループに **es_test_resource** という名前を付けます。



B. サーバーを追加します。



- ECS UUID：#unique_10。 ECS インスタンスにログインし、dmidecode | grep UUID コマンドを実行して戻り値を取得します。



- Machine IP/Machine CPU (Cores)/Memory Size (GB)：ECS インスタンスのパブリック IP アドレス、CPU コア、メモリサイズを指定します。 ECS コンソーにログ

インし、ECS インスタンス名をクリックすると、**[**設定情報**]** に、情報が表示されます。

C. エージェントをインストールします。次の手順に従って、エージェントのインストールを完了します。 この例では、VPC を使用します。 したがって、インスタンスのポート 8000 を開く必要はありません。

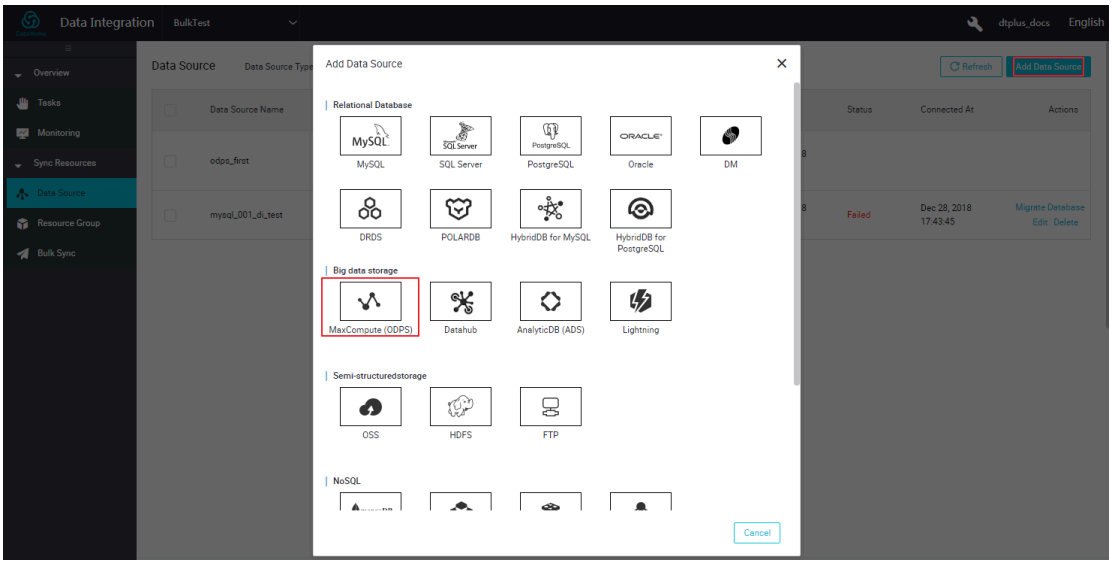
D. 接続を確認します。接続が正常に確立されると、ステータスが **[Available]** に変わります。 ステータスが **[Unavailable]** の場合、ECS インスタンスにログインし、`tail -f /home/admin/alisatasknode/logs/heartbeat.log` コマンドを実行して DataWorks

と ECS インスタンス間のハートビートメッセージがタイムアウトしたかどうかを確認します。

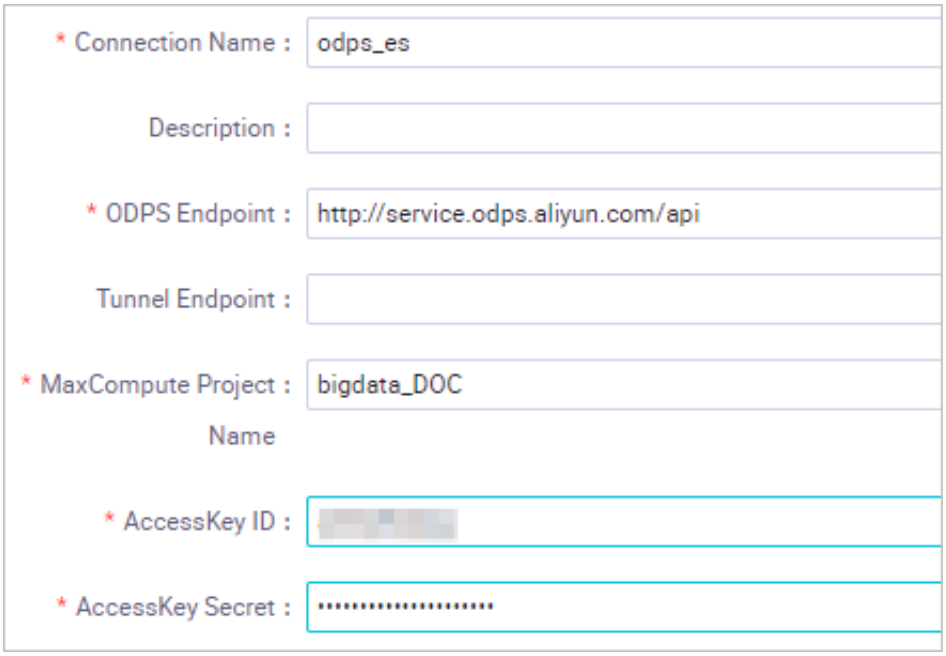
d. データソースを追加します。

A. **[Data Integration]** ページの左側のナビゲーションウィンドウで、**[Data Source]** を選択し、**[Add Data Source]** をクリックします。

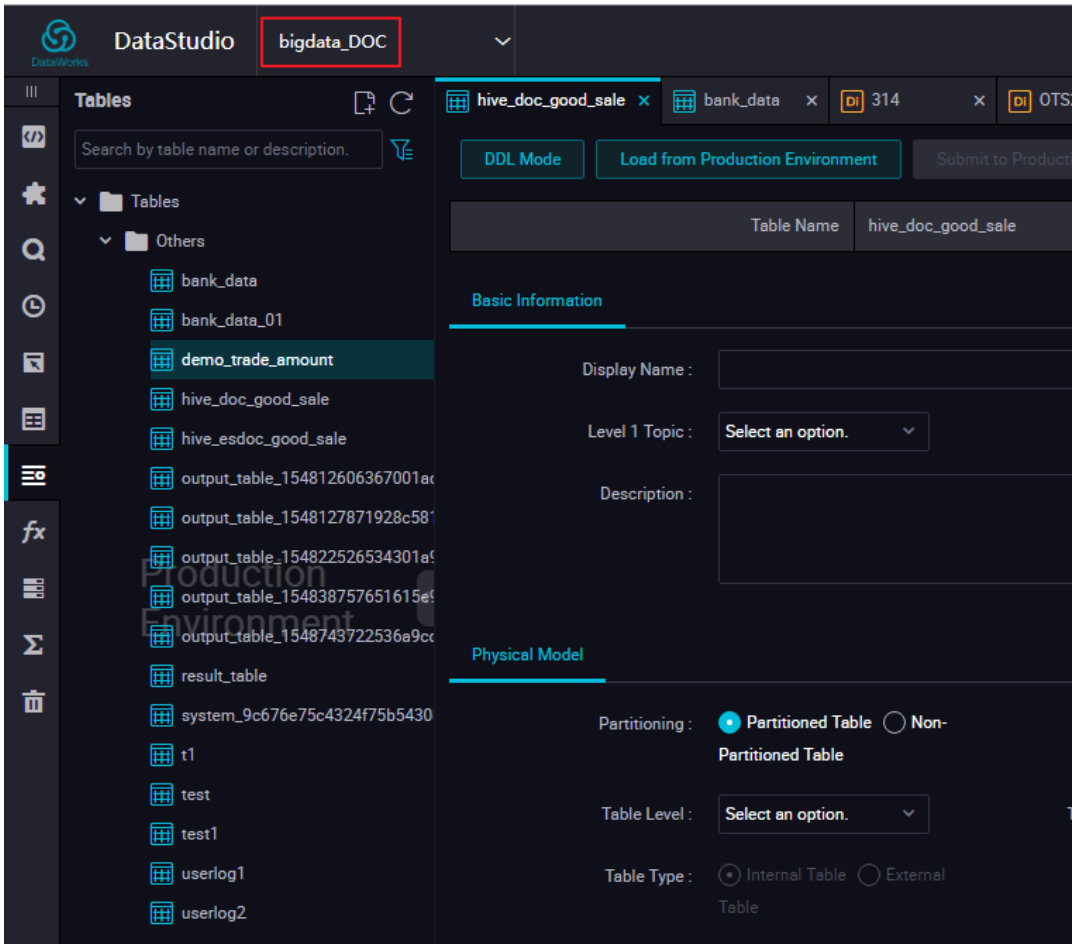
B. ソースタイプとして **[MaxCompute]** を選択します。



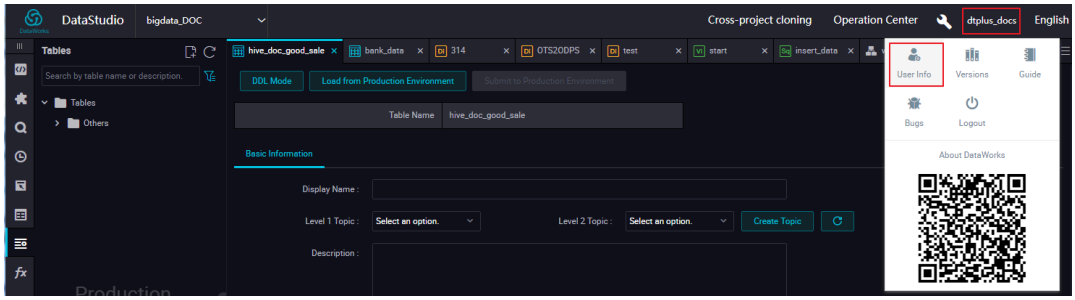
C. データソースに関する情報を入力します。 この例では、**odps_es** という名前のデータソースを作成します (次図)。



- **ODPS workspace name**：DataWorks の **[Data Analytics]** ページの左上隅のアイコンの右側に、テーブルのワークスペース名が表示されます (次図) 。



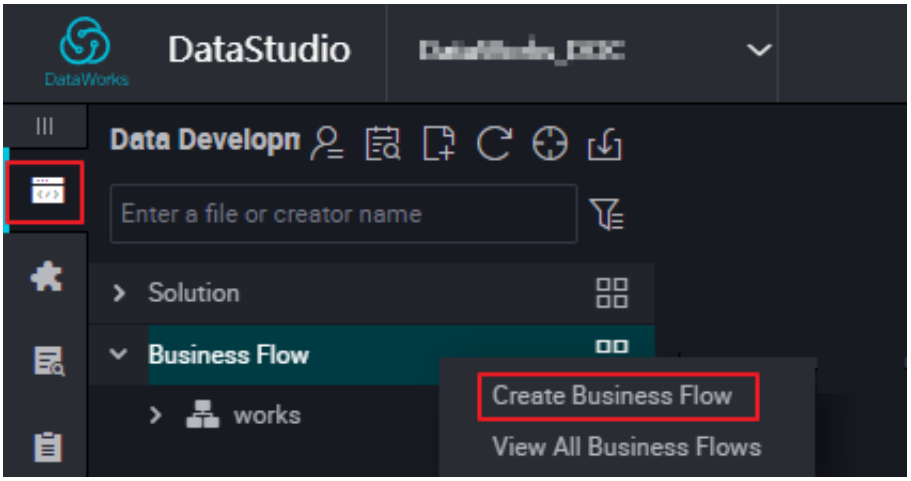
- **AccessKeyId /AccessKeySecrete**：ユーザー名の上にポインターを移動し、**[User Info]** を選択します (次図)。



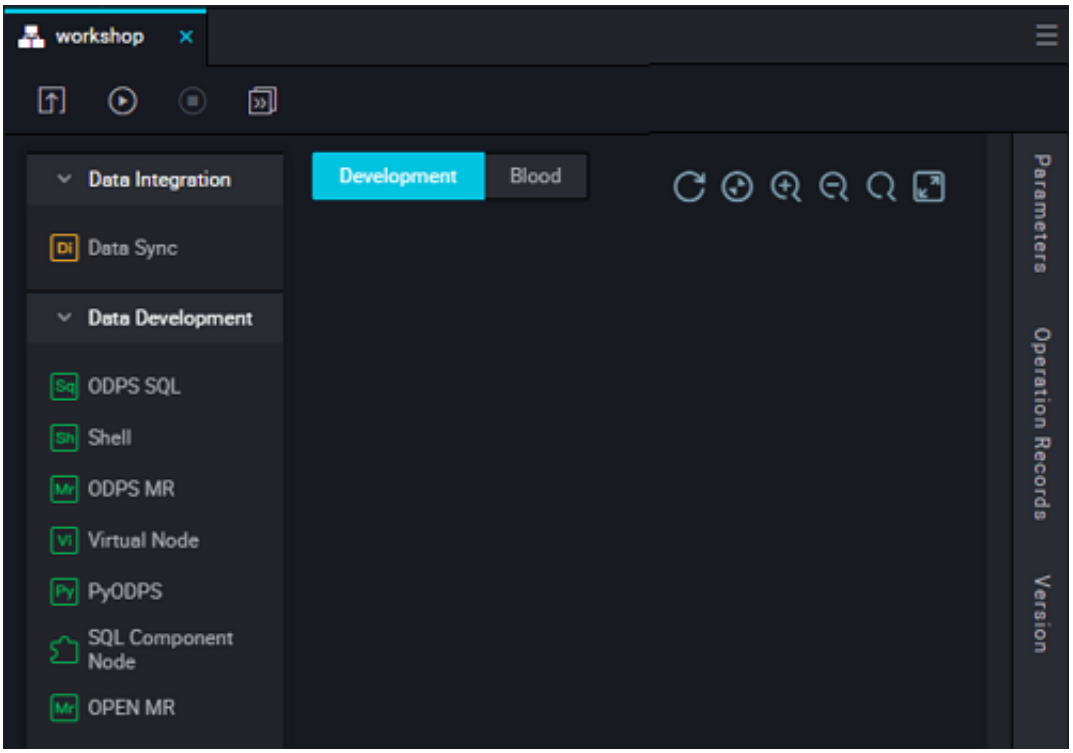
[Personal Account] ページでアバターの上にポインターを移動し、**[accesskeys]** をクリックします (次図)。

e. 同期タスクを設定します。

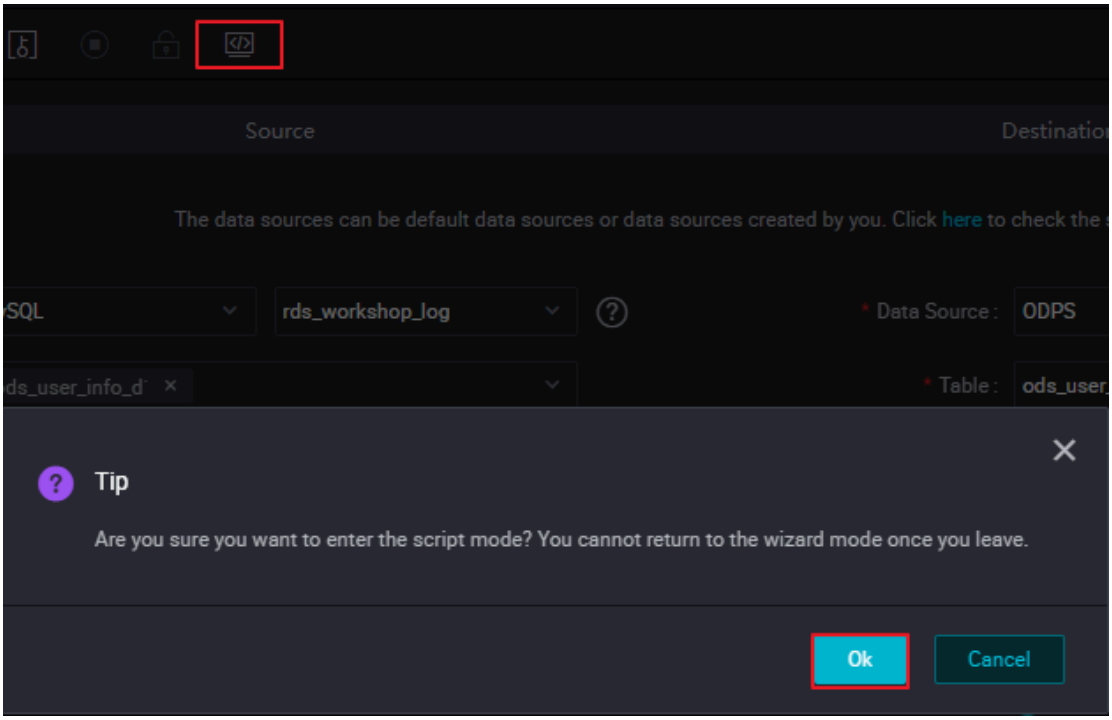
A. [Data Analytics] ページの左側のナビゲーションウィンドウで、**[Data Analytics]** アイコンをクリックし、**[Business Flow]** をクリックします。



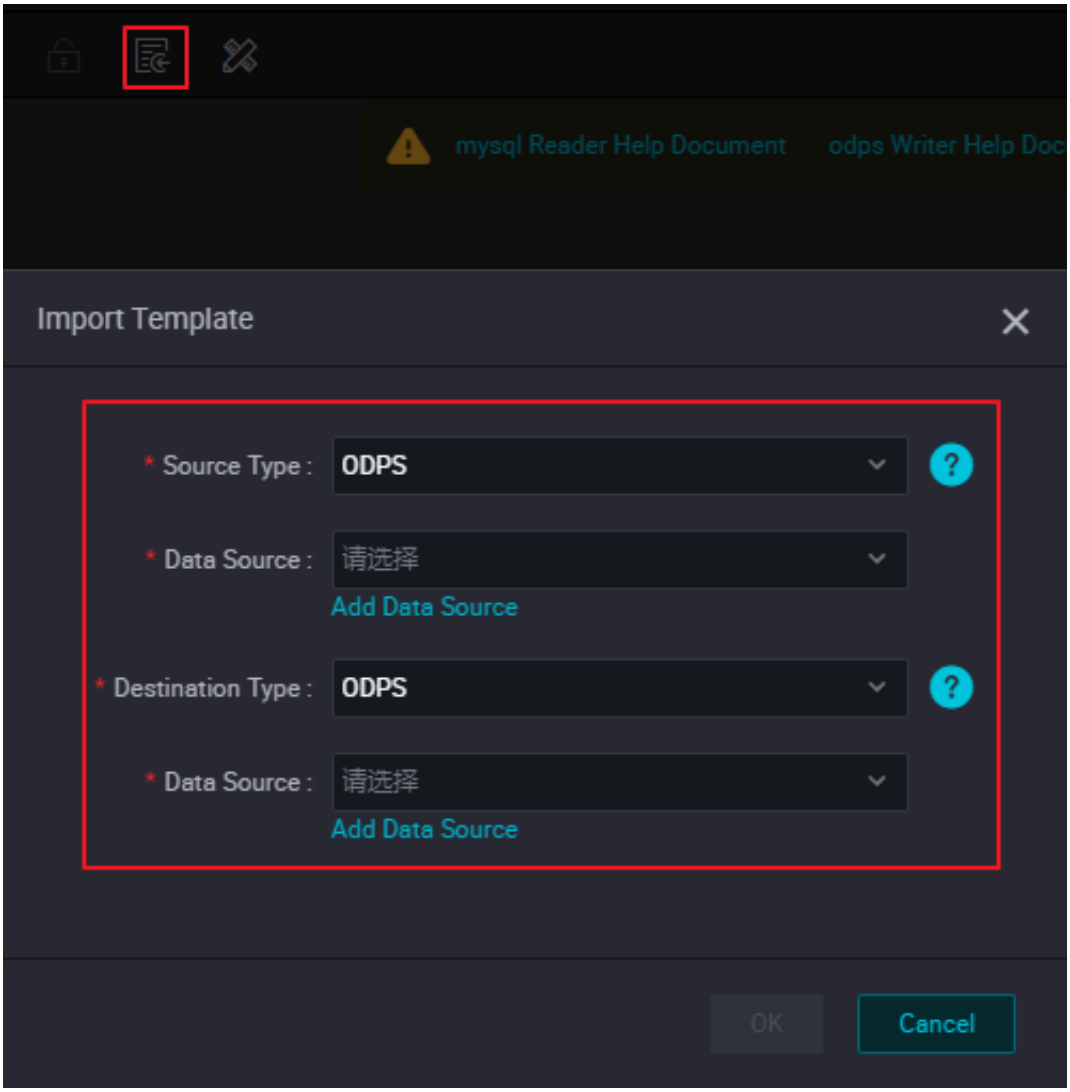
B. ターゲットビジネスフローをクリックし、**[Data Integration]** を選択し、**[Create Data Integration Node]** > **[Data Sync]** を選択してから、同期ノード名を入力します。



C. 同期ノードが正常に作成されたら、新しい同期ノードページの上部にある **[Switch to Script Mode]** アイコンをクリックし、**[Confirm]** を選択します。



D. [Script Mode] ページの上部にある **[Apply Template]** アイコンをクリックします。[Source Type]、[Data Source]、[Destination Type]、[Data Source] のオプションに情報を入力し、**[OK]** をクリックして初期スクリプトを生成します。



E. データ同期スクリプトの設定を行います。 Elasticsearch の設定ルールの詳細は、「writer プラグインの設定」をご参照ください。

注：

- 同期スクリプトの設定には、Reader、Writer、Setting の 3 つの部分があります。**Reader** では、データを同期するソースクラウドサービスを設定します。 **Write** では、Elasticsearch の設定ファイルを設定します。 **Setting** では、パケットロスと最大同時タスクを設定します。

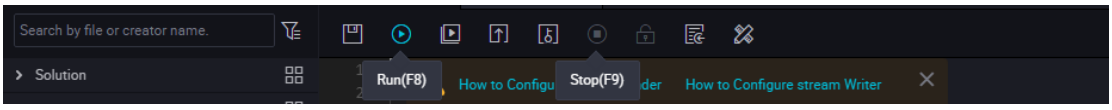
- **Endpoint** には、Elasticsearch インスタンスのプライベートまたはパブリック IP アドレスを指定します。 この例では、プライベート IP アドレスを使用します。 したがって、ホワイトリストは不要です。 外部 IP アドレスを使用する場合、Elasticsearch の **[**ネットワークとスナップショット**]** ページで、Elasticsearch へのアクセスが許可されているパブリック IP アドレスを含むホワイトリストを設定する必要があります。 ホワイトリストには、DataWorks サーバーの IP アドレスと、使用するリソースグループを指定する必要があります。
- Elasticsearch インスタンスへのログインに使用されるユーザー名とパスワードを Elasticsearch Writer の **accessId** と **accesskey** に設定する必要があります。
- Elasticsearch インスタンスのインデックス名を **index** に入力します。 Elasticsearch インスタンスのデータにアクセスするには、このインデックス名を使用する必要があります。 この例では、**es_index** という名前の インデックスを使用します。
- MaxCompute テーブルがパーティションテーブルの場合、**partition** フィールドにパーティション情報を設定する必要があります。 この例のパーティション情報は、**pt=1** です。

サンプル設定コード：

```
{
"configuration": {
"reader": {
"plugin": "odps",
"parameter": {
 "partition": "pt=1",
 "datasource": "odps_es",
 "column": [
  "create_time",
  "category",
  "brand",
  "buyer_id",
  "trans_num",
  "trans_amount",
  "click_cnt"
 ],
 "table": "hive_doc_good_sale"
}
},
"writer": {
"plugin": "elasticsearch",
"parameter": {
 "accessId": "elastic",
 "endpoint": "http://es-cn-mpXXXXXXX.elasticsearch.aliyuncs.com:9200",
 "indexType": "elasticsearch",
 "accessKey": "XXXXXX",
 "cleanup": true,
 "discovery": false,
 "column": [
  {
```

```
      "name": "create_time",
      "type": "string"
     },
     {
      "name": "category",
      "type": "string"
     },
     {
      "name": "brand",
      "type": "string"
     },
     {
      "name": "buyer_id",
      "type": "string"
     },
     {
      "name": "trans_num",
      "type": "long"
     },
     {
      "name": "trans_amount",
      "type": "double"
     },
     {
      "name": "click_cnt",
      "type": "long"
     }
    ],
    "index": "es_index",
    "batchSize": 1000,
    "splitter": ",",
  }
  },
  "setting": {
  "errorLimit": {
    "record": "0"
  },
  "speed": {
    "throttle": false,
    "concurrent": 1,
    "mbps": "1",
    "dmu": 1
  }
  }
  },
  "Type": "job ",
  "version": "1.0"
  }
```

F. スクリプトが同期されたら、**[Run]** をクリックして、ODPS データを Elasticsearch と同期させます。



3. 結果の検証

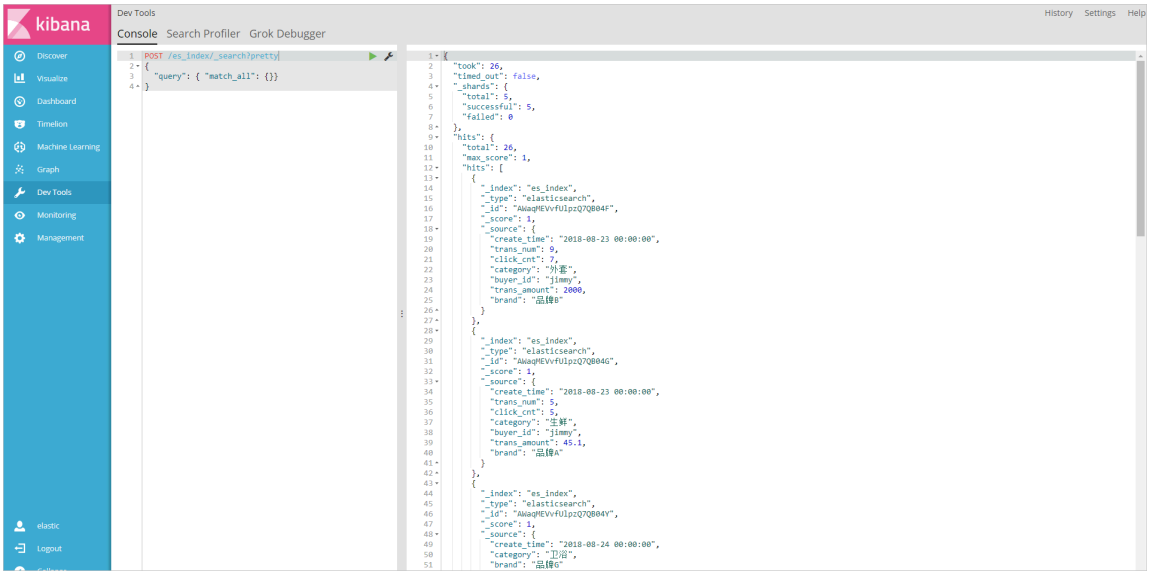
   a. Elasticsearch コンソールにログインし、右上隅の **Kibana** コンソールをクリックし、**[Dev Tools]** を選択します。

   b. 次のコマンドを実行して、データが Elasticsearch に正常にレプリケートされたことを確認します。

```
POST /es_index/_search? pretty
{
"query": { "match_all": {}}
}
```

   es_index は、データ同期中の **index** フィールドの値です。

   データが正常に同期されると、次のページが表示されます。



   c. 次のコマンドを実行して、**trans_num** フィールドを基準にドキュメントをソートします。

```
POST /es_index/_search? pretty
{
"query": { "match_all": {} },
"sort": { "trans_num": { "order": "desc" } }
}
```

   d. 次のコマンドを実行して、ドキュメント内の **category** フィールドと **brand** フィールドを検索します。

```
POST /es_index/_search? pretty
{
"query": { "match_all": {} },
"_source": ["category", "brand"]
}
```

   e. 次のコマンドを実行して、 **category** が **fresh** のドキュメントをクエリします。

```
POST /es_index/_search? pretty
```

```
{
"query": { "match": {"category":"fresh"} }
}
```

詳細は、「#unique_39」と『Elastic help center』をご参照ください。

よくある質問

Elasticsearch インスタンスに接続するときにエラーが発生します

1. 同期スクリプトを実行する前に、前の手順で作成したリソースグループが、右側の **[configuration tasks resources group]** メニューで選択されているかどうかを確認してください。
   - リソースグループが選択されている場合、次の手順に進みます。
   - リソースグループが選択されていない場合、右側の **[configuration tasks resources group]** メニューで、作成したリソースグループを選択し、**[Run]** をクリックします。
2. endpoint、accessId、accesskey など、同期スクリプトの設定が正しいかどうかを確認してください。 endpoint には、Elasticsearch インスタンスのプライベートまたはパブリック IP アドレスを指定します。 パブリック IP アドレスを使用する場合、ホワイトリストを設定してください。 accessId には、Elasticsearch インスタンスへのアクセスに使用されるユーザー名を指定します。デフォルトは elastic です。 accesskey には、Elasticsearch インスタンスへのアクセスに使用されるパスワードを指定します。

4.2 Alibaba Cloud Realtime Compute および Alibaba Cloud Elasticsearch のベストプラクティス

このドキュメントでは、Alibaba Cloud Realtime Compute を使用してデータを処理し、データを Alibaba Cloud Elasticsearch (ES) にインポートする方法について説明します。

- Alibaba Cloud Realtime Compute の有効化とプロジェクトの作成
- Alibaba Cloud Elasticsearch の有効化
- このドキュメントでは、Alibaba Cloud Log Service を例として使用しています。 そのため、Alibaba Cloud Log Service の有効化、 #unique_42/unique_42_Connect_42_section_ahq_ggx_ndb 、および #unique_43/unique_43_Connect_42_section_v52_2jx_ndb も必要です。

Alibaba Cloud Realtime Compute は、Alibaba Cloud Flink プロダクトです。 Kafka や Elasticsearch など、複数のデータソースとデータ消費サービスをサポートしています。#unique_44 Alibaba Cloud Realtime Compute および Elasticsearch を使用して、ログ取得シナリオのビジネス要件を満たすことができます。

Kafka または Log Service のログは、単純または複雑な Flink SQL 文を使用して Realtime Compute によって処理され、検索サービスのソースデータとして Alibaba Cloud Elasticsearch にインポートされます。 Realtime Compute の強力なコンピューティング機能と Alibaba Cloud Elasticsearch の検索機能により、リアルタイムのデータ処理と検索を実現し、ビジネスをリアルタイムサービスに変換できます。 さらに、Realtime Compute は Alibaba Cloud Elasticsearch と簡単に連携できます。 次の例は、Realtime Compute を Alibaba Cloud Elasticsearch と連携する方法を示しています。

たとえば、ログまたはデータが Log Service にインポートされ、Alibaba Cloud Elasticsearch にインポートする前にデータを処理する必要があると仮定します。 次の図は、データ消費のパイプラインを示しています。

Realtime Compute ジョブを作成する

1. Realtime Compute コンソール にログインし、ジョブを作成します。
2. Flink SQL 文を記述します。

   a) Log Service テーブルを作成します。

```
create table sls_stream(
  a int,
  b int,
  c VARCHAR
)
WITH (
  type ='sls',
  endPoint ='<yourEndpoint>',
  accessId ='<yourAccessId>',
  accessKey ='<yourAccessKey>',
  startTime = '<yourStartTime>',
  project ='<yourProjectName>',
  logStore ='<yourLogStoreName>',
  consumerGroup ='<yourConsumerGroupName>'
```

```
);
```

WITH 変数の説明は次のとおりです。

変数	説明
<yourEndpoint>	Alibaba Cloud Log Service のパブリックエンドポイント。エンドポイントは、Log Service のプロジェクトや内部ログデータにアクセスするために使用される URL です。 詳細は、「#unique_45」をご参照ください。 たとえば、中国 (杭州) の Log Service のエンドポイントは http://cn-hangzhou.log.aliyuncs.com です。 エンドポイントが http:// で始まることを確認してください。
<yourAccessId>	Log Service へのアクセスに使用される AccessKey ID。
<yourAccessKey>	Log Service へのアクセスに使用される AccessKey シークレット。
<yourStartTime>	ログデータが消費される時間範囲の開始時間。 Realtime Compute ジョブを実行するときは、この変数で指定された開始時間よりも早い時間を指定する必要があります。
<yourProjectName>	Log Service プロジェクトの名前。
<yourLogStoreName>	プロジェクト内の Logstore の名前。
<yourConsumerGroupName>	Log Service コンシューマグループの名前。

WITH 変数の詳細については、「#unique_46」をご参照ください。

b) Elasticsearch 結果テーブルを作成します。

:

- Realtime Compute V3.2.2 以降のバージョンでは、Elasticsearch 結果テーブルの対応が追加されています。 Realtime Compute ジョブを作成するときに、正しいバージョンを選択してください。
- Elasticsearch の結果テーブルは RESTful API に基づいています。 したがって、すべての Elasticsearch バージョンと互換性があります。

```
CREATE TABLE es_stream_sink(
  a int,
  cnt BIGINT,
  PRIMARY KEY(a)
)
WITH(
  type ='elasticsearch',
  endPoint = 'http://<instanceid>.public.elasticsearch.aliyuncs.com:<port>',
```

```
  accessId = '<yourAccessId>',
  accessKey = '<yourAccessSecret>',
  index = '<yourIndex>',
  typeName = '<yourTypeName>'
);
```

WITH 変数の説明は次のとおりです。

変数	説明
<Instances>	Elasticsearch インスタンスの ID。 インスタンスの [基本情報] ページでインスタンス ID を確認できます。 例：es-cn-45xxxxxxxxxxxxk1q
<port>	Elasticsearch インスタンスのパブリックネットワークポート。 インスタンスの [基本情報] ページでパブリックネットワークポートを確認できます。 デフォルトのポート番号は、9200 です。
<yourAccessId>	Elasticsearch インスタンスへのアクセスと Kibana コンソールへのログインに使用されるユーザー名。 デフォルトのユーザー名は "elastic" です。
<yourAccessKey>	Elasticsearch インスタンスにアクセスし、 Kibana コンソールにログインするために使用されるパスワード。 パスワードは、Elasticsearch インスタンスを作成するときに指定されます。
<yourIndex>	Elasticsearch インスタンス上のドキュメントのインデックス。 インデックスはデータベース名のようなものです。 ドキュメントのインデックスが作成されていない場合は、最初にインデックスを作成してください。 詳細は、「#unique_47」をご参照ください。
<yourTypeName>	インデックスのタイプ。 タイプは、データベース内のテーブルの名前のようなものです。 インデックスにタイプが指定されていない場合は、最初にタイプを指定します。 詳細は、「#unique_47」をご参照ください。

WITH 変数の詳細については、「#unique_48」をご参照ください。

- Elasticsearch は、PRIMARY KEY フィールドに含まれるドキュメント ID に従ったドキュメントの更新をサポートしています。 PRIMARY KEY フィールドとして指定できるフィールドは 1 つだけです。 PRIMARY KEY フィールドを指定すると、フィールドの値がドキュメント ID として使用されます。 ドキュメント ID は、PRIMARY KEY フィールドの

ないドキュメントに対してランダムに生成されます。 詳細については『Index API』をご参照ください。

- Elasticsearch は複数の更新モードをサポートしています。 **updateMode** パラメーターを設定して、更新モードを指定できます。
  - updateMode=full の場合、新しいドキュメントが既存のドキュメントを上書きします。
  - updateMode=inc の場合、新しい値が、関連するフィールドの既存の値を上書きします。
- Elasticsearch のすべての更新は、データの挿入または更新を意味する UPSERT 構文に従います。

c) データ消費ロジックを作成し、データを同期します。

```
INSERT INTO es_stream_sink
SELECT
  a,
  count(*) as cnt
FROM sls_stream GROUP BY a
```

3. ジョブを送信して実行します。

ジョブを送信して実行すると、Log Service に保存されたデータが集約され、Elasticsearch にインポートされます。 Realtime Compute は他の計算操作もサポートしています。たとえば、データビューやユーザー定義拡張機能 (UDX) を作成できます。 詳細は、「#unique_49」をご参照ください。

まとめ

Alibaba Cloud Realtime Compute と Elasticsearch を使用すると、独自のリアルタイム検索サービスをすばやく作成できます。 Alibaba Cloud Elasticsearch にデータをインポートするために、より複雑なロジックが必要な場合は、Realtime Compute のカスタムシンク機能を使用してください。 詳細は、「#unique_50」をご参照ください。

5 データを収集します

5.1 Beats による可視化された O&M システムの構築

背景

Beats はデータシッパー専用のプラットフォームです。 Beats をインストールすると、軽量の Beats エージェントから Logstash や Elasticsearch などのターゲットオブジェクトにインスタンスのデータを送信できます。

Beats エージェントおよび軽量シッパーとして、Metricbeat はシステムやサービスからメトリックを収集し、そのメトリックを Elasticsearch などのターゲットオブジェクトに送信するように設計されています。 Metricbeat は、システムやサービスの統計情報を CPU からメモリ、Redis から NGINX などに送信するための軽量な方法です。

このトピックでは、Metricbeat を使用して MacBook からメトリックを収集し、そのメトリックを Elasticsearch インスタンスに送信し、Kibana にダッシュボードを生成する方法について説明します。

 注：

Linux または Windows システムを実行しているコンピューターからメトリックを収集し、そのメトリックを Elasticserach インスタンスに送信する手順もほぼ同じです。

1. Elasticsearch インスタンスの購入と設定

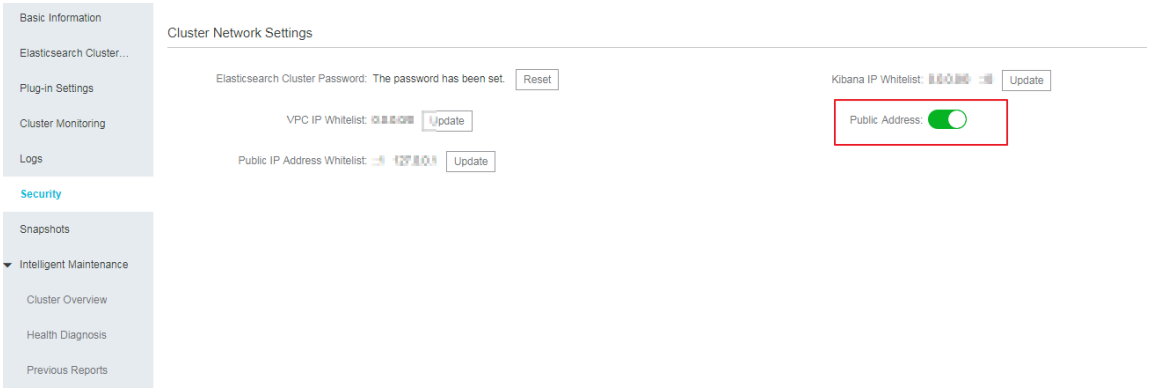
   Elasticsearch インスタンスがない場合は、Elasticsearch を有効化してインスタンスを作成する必要があります (#unique_53)。 その後、インスタンスの内部またはパブリック IP アドレスを介して、MacBook で収集したデータを Elasticsearch インスタンスに送信できるようになります。

    注：

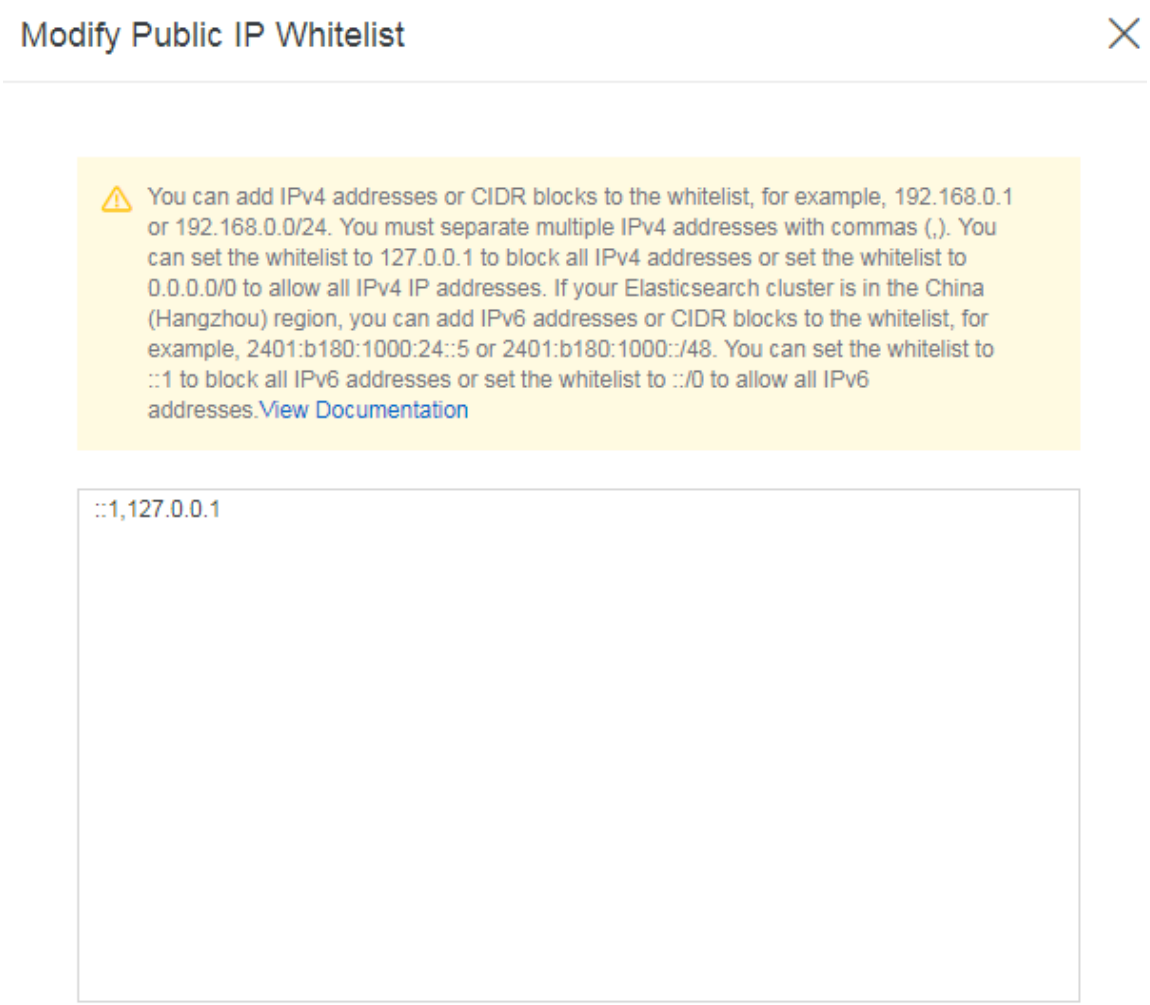
   - パブリック IP アドレスを介して Elasticsearch インスタンスにアクセスする場合、[セキュリティ] ページで [パブリックアドレス] のスイッチをオンにし、**[パブリック IP アドレスのホワイトリスト]** を設定する必要があります。

- 内部 IP アドレスを介して Elasticsearch インスタンスにアクセスする場合、Elasticsearch インスタンスと同じ **VPC** とリージョン に Elastic Compute Service (ECS) を作成して、Elasticsearch へのアクセスを管理する必要があります。

a. Elasticsearch コンソールにログインし、インスタンスの名前か ID をクリックしてから、左側のナビゲーションウィンドウで [セキュリティ] をクリックします。 [セキュリティ] ページで、[パブリックアドレス] のスイッチをオンにします。



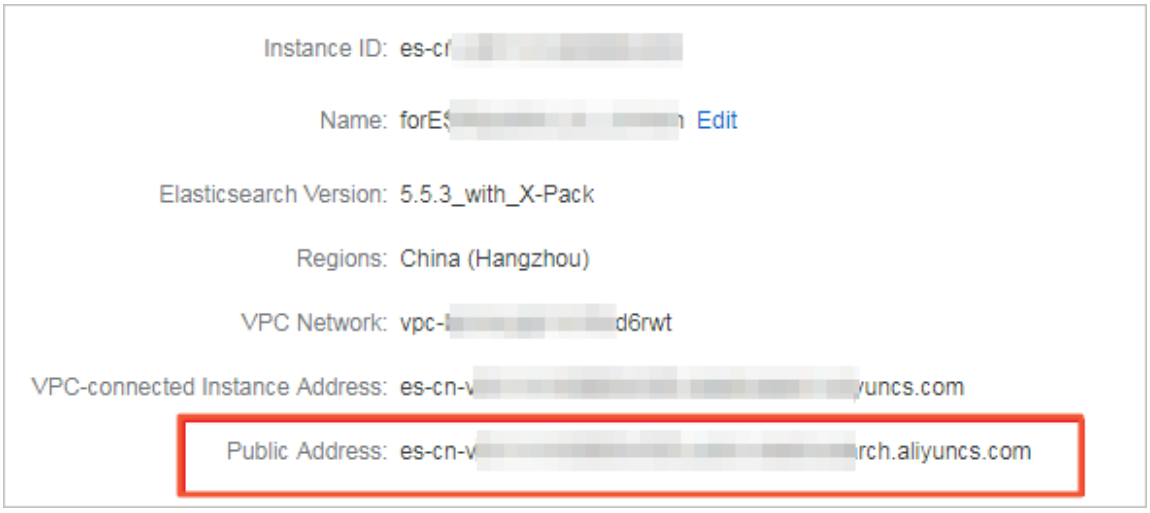
b. MacBook のパブリック IP アドレスをホワイトリストに追加します。



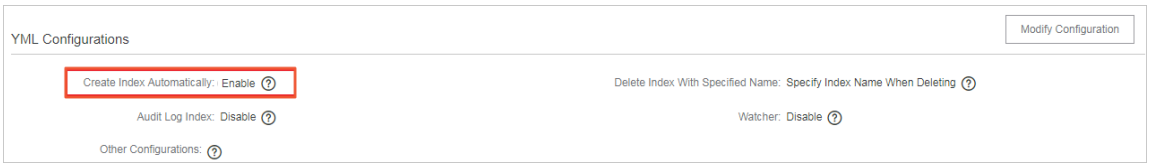
:

パブリックネットワークを使用する場合、パブリックネットワークのアウトバウンドネットワークトラフィックを制御するジャンプサーバーの IP アドレスをホワイトリストに追加します。 ジャンプサーバーの IP アドレスを取得できない場合、**0.0.0.0/1,128.0.0.0/1** をホワイトリストに追加して、特定の IP アドレスを許可します。 この設定により、**Elasticsearch** インスタンスはパブリックネットワークに公開されます。 リスクを評価し、慎重に進めてください。

c. 設定が完了したら、左側のナビゲーションペインで [基本情報]をクリックし、Elaticsearch インスタンスのパブリック IP アドレスをコピーします。



d. YML 設定を変更します。 [YML 設定] ページで、**[自動インデックス]** を有効化します。 デフォルトで、この機能は無効化されています。 この操作により、Elasticsearch インスタンスが再起動され、有効になるまでに時間がかかります。



2. Metricbeat のダウンロードと設定

- Metricbeat インストールパッケージ (Mac オペレーティングシステム用)
- Metricbeat インストールパッケージ (32 ビット Linux オペレーティングシステム用)
- Metricbeat インストールパッケージ (64 ビット Linux オペレーティングシステム用)
- Metricbeat インストールパッケージ (32 ビット Windows オペレーティングシステム用)
- Metricbeat インストールパッケージ (64 ビット Windows オペレーティングシステム用)

a. Metricbeat ファイルをダウンロード、解凍し、開きます。

```
[[redacted]MacBook-Pro:Desktop [redacted]$ cd metricbeat-6.3.1-darwin-x86_64
[[redacted]MacBook-Pro:metricbeat-6.3.1-darwin-x86_64 [redacted]$ ls
LICENSE.txt          data                 logs                     metricbeat.yml
NOTICE.txt           fields.yml           metricbeat               modules.d
README.md            kibana               metricbeat.reference.yml
[redacted]deMacBook-Pro:metricbeat-6.3.1-darwin-x86_64 [redacted]$
```

b. metricbeat.yml ファイルの **Elasticsearch output** セクションを開き、編集します。 該当するコンテンツのコメントを解除する必要があります。

```
#================================ Outputs =====================================

# Configure what output to use when sending the data collected by the beat.

#-------------------------- Elasticsearch output ------------------------------
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["es-cn-[redacted].public.elasticsearch.aliyuncs.com:9200"]

  # Optional protocol and basic auth credentials.
  protocol: "http"
  username: "[redacted]"
  password: "[redacted]"
```

注：

Elasticsearch のアクセス制御情報は、以下のとおりです。

- hosts：Elasticsearch インスタンスのパブリックまたは内部 IP アドレス。 この例では、パブリック IP アドレスを使用しています。
- protocol：**http** を設定します。
- username：デフォルトのユーザー名は **elastic** です。
- password：Elasticsearch へのログインに使用されるパスワード。

3. Metricbeat の有効化

   次のコマンドを実行して Metricbeat を有効化し、Metricbeat を使用して Elasticsearch インスタンスにデータを送信します。

   ```
   ./metricbeat -e -c metricbeat.yml
   ```

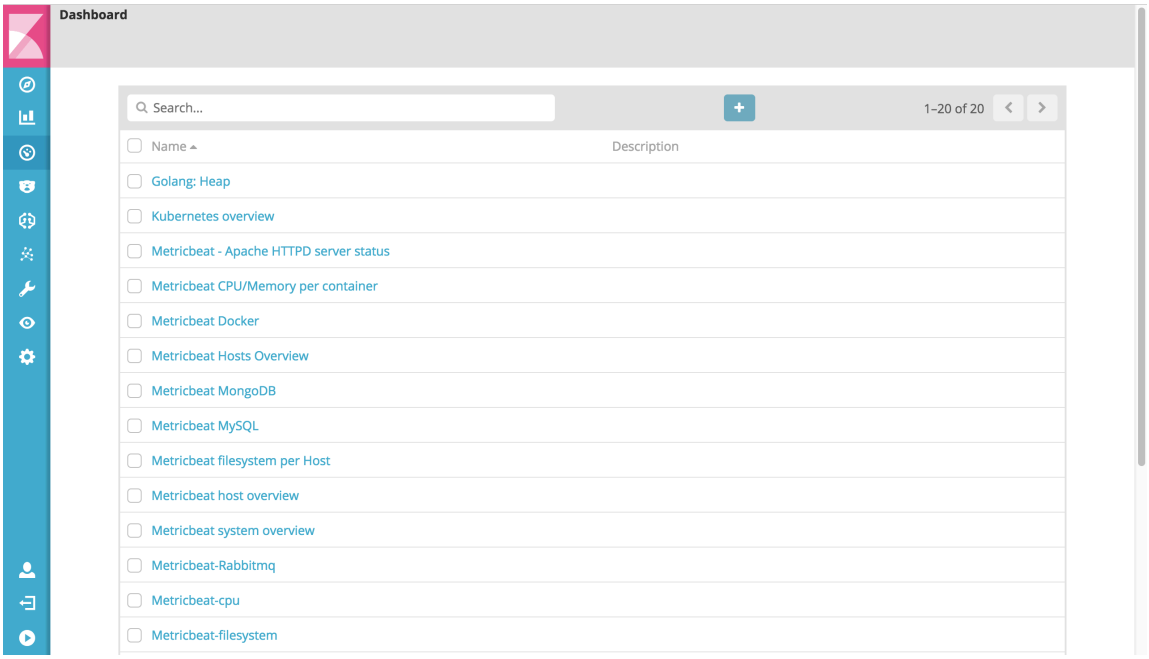
4. Kibana でのダッシュボードの表示

   Elasticsearch コンソールの右上にある Kibana コンソールをクリックします。 Dashboard ページに移動します (次図)。

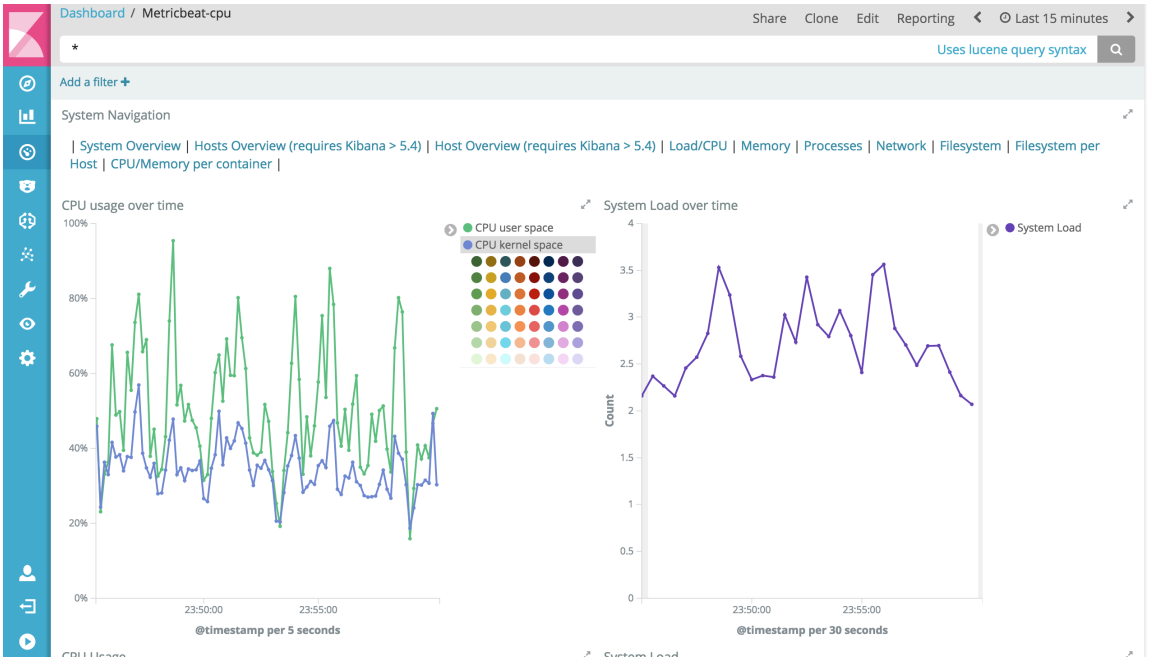
   注 :

Kibana コンソールでインデックスパターンを作成していない場合、ダッシュボードに情報が表示されないことがあります。 この問題を解決するには、インデックスパターンを作成して、[ダッシュボード]ページで情報をもう一度表示します。

a. メトリックのリスト。



b. CPU メトリック。



注：

5 秒ごとにデータを更新してレポートを生成するようシステムをスケジュールしたり、例外が発生したときにアラートを送信するよう webhook を設定したりすることができます。

6 Elasticsearchアプリケーション

7 Java high-level REST Client を使用したドキュメント API の呼び出し

7.1 概要

このベストプラクティスは、Java API 6.3.x に基づいています。 Java high-level REST Client を使用して Alibaba Cloud Elasticsearch インスタンスに接続し、Elasticsearch ドキュメント API を呼び出す方法について説明します。 ドキュメント API を呼び出して、ローカル Java プログラムでタスクを実行できます。たとえば、ドキュメントを作成、取得、または更新できます。

このベストプラクティスは Java API 6.3.x に基づいており、Alibaba Cloud Elasticsearch V6.3 インスタンスを使用しています。 Java high-level REST Client を使用して Alibaba Cloud Elasticsearch インスタンスに接続し、Elasticsearch ドキュメント API を呼び出す方法について説明します。 ベストプラクティスには次のトピックが含まれます。

1. Alibaba Cloud Elasticsearch インスタンスの作成と設定：Alibaba Cloud Elasticsearch インスタンスを作成し、ホワイトリストを設定して、自動インデックス作成を有効にする
2. ドキュメント API の呼び出し：Alibaba Cloud Elasticsearch インスタンスに接続し、ドキュメントを作成、取得、更新、削除する

7.2 Alibaba Cloud Elasticsearch インスタンスの作成と設定

このドキュメントでは、インスタンスの購入、パブリックネットワークアクセスの有効化、ホワイトリストの設定、自動インデックス設定など、Alibaba Cloud Elasticsearch インスタンスを作成して設定する方法について説明します。

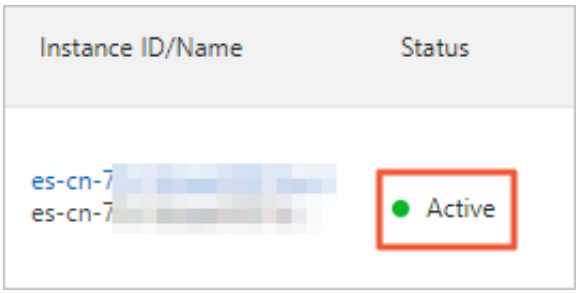
Alibaba Cloud Elasticsearch は VPC ネットワークのみをサポートします。 Elasticsearch インスタンスを作成する前に、VPC ネットワークと VSwitch を作成する 必要があります。

1. Alibaba Cloud Elasticsearch コンソール にログインし、Elasticsearch インスタンスを購入します。

   次の図は、この例で購入した Elasticsearch インスタンスの情報を示しています。
   Elasticsearch のバージョンを 6.3 に設定します。 他のバージョンには非互換性の問題がある可能性があります。

   注：
   Elasticsearch インスタンスを購入するとき、パスワードを指定する必要があります。 パスワードを記録し、安全な場所に保管してください。 このパスワードは、Java API を使用して Elasticsearch インスタンスに接続するときに使用します。

   Elasticsearch インスタンスを購入すると、[インスタンス] ページに移動します。 インスタンスの **[ステータス]** が **[有効]** に代わるまで待ち、次のタスクを続けます。



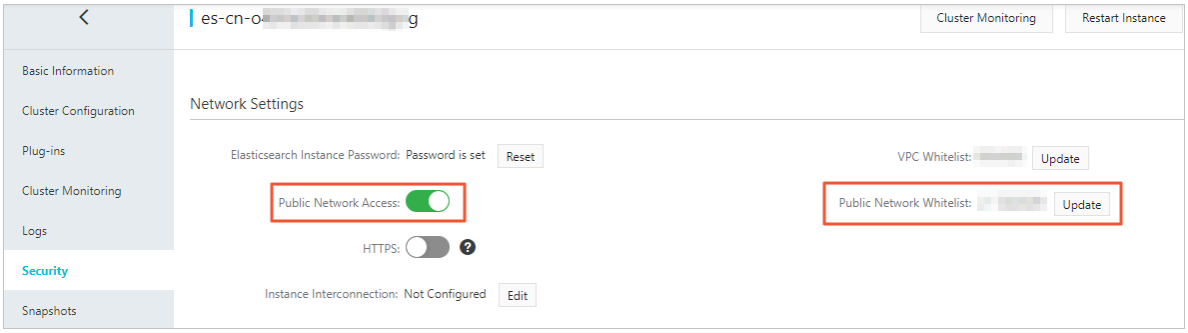
   注：
   Elasticsearch がインスタンスを有効化するのに約 30 分かかる場合があります。

2. パブリックネットワークアクセスを有効にします。

   インスタンス名をクリックします。 左側のナビゲーションウィンドウで、**[セキュリティ]** を選択し、 **[インターネットアドレス]** スイッチをクリックしてこの機能を有効にします。

   注：
   [インターネットアドレス] を有効にしたら、 **[基本情報]** ページでインスタンスのパブリックエンドポイントを確認できます。 パブリックエンドポイントを記録してください。 このエンドポイントは、Java API を呼び出して Elasticsearch インスタンスに接続するときに使用します。

3. ホストのパブリック IP アドレスを Elasticsearch インスタンスのパブリックネットワークホワイトリストに追加します。



ホストのパブリック IP アドレスを照会するには、www.google.com にアクセスし、"What Is My IP Address" と入力して、関連するリンクをクリックします。

4. 自動インデックス作成を有効にします。

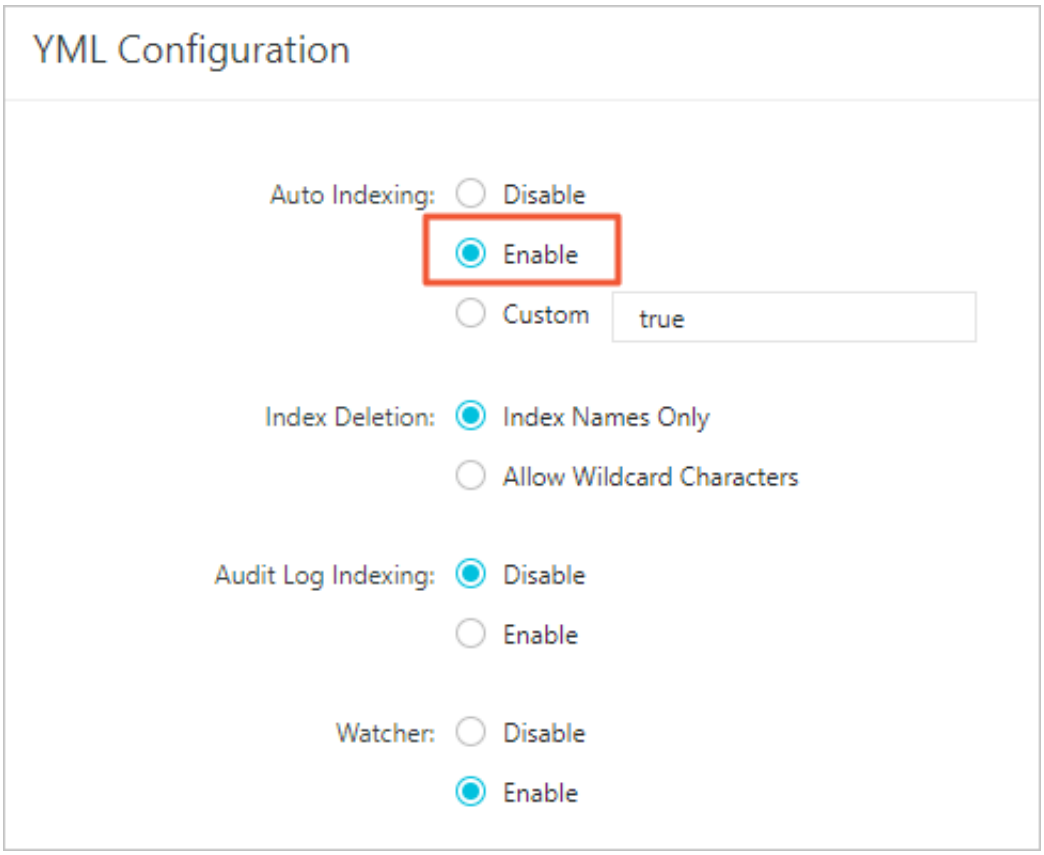
:

この機能を有効にするには、Elasticsearch で Elasticsearch インスタンスを再起動する必要があります。 操作を実行する前に、ビジネスが悪影響を受けないことを確認してください。

a) Elasticsearch インスタンスの **[クラスター設定]** ページに移動し、**[YML 設定]** の右側にある **[設定の編集]** をクリックします。

b) **[YML パラメーター設定]** ページで、 **[自動インデックス]** を **[自動インデックスを有効にする]** に設定します。

c) **[この操作は、インスタンスを再起動します。確認してから操作を行ってください。]** チェックボックスをオンにし、 **[OK]** をクリックします。



Elasticsearch インスタンスの再起動後、Java high-level REST Client を使用してドキュメント API の呼び出し を実行します。

7.3 ドキュメント API の呼び出し

このドキュメントでは、Java high-level REST Client を使用して Elasticsearch ドキュメント API を呼び出す方法について説明します。 また、Alibaba Cloud Elasticsearch クラスターに接続し、インデックスやドキュメントを作成したり、ドキュメントを取得するためのサンプルコードも提供します。

- JDK をインストールし、Java 環境変数を設定します。

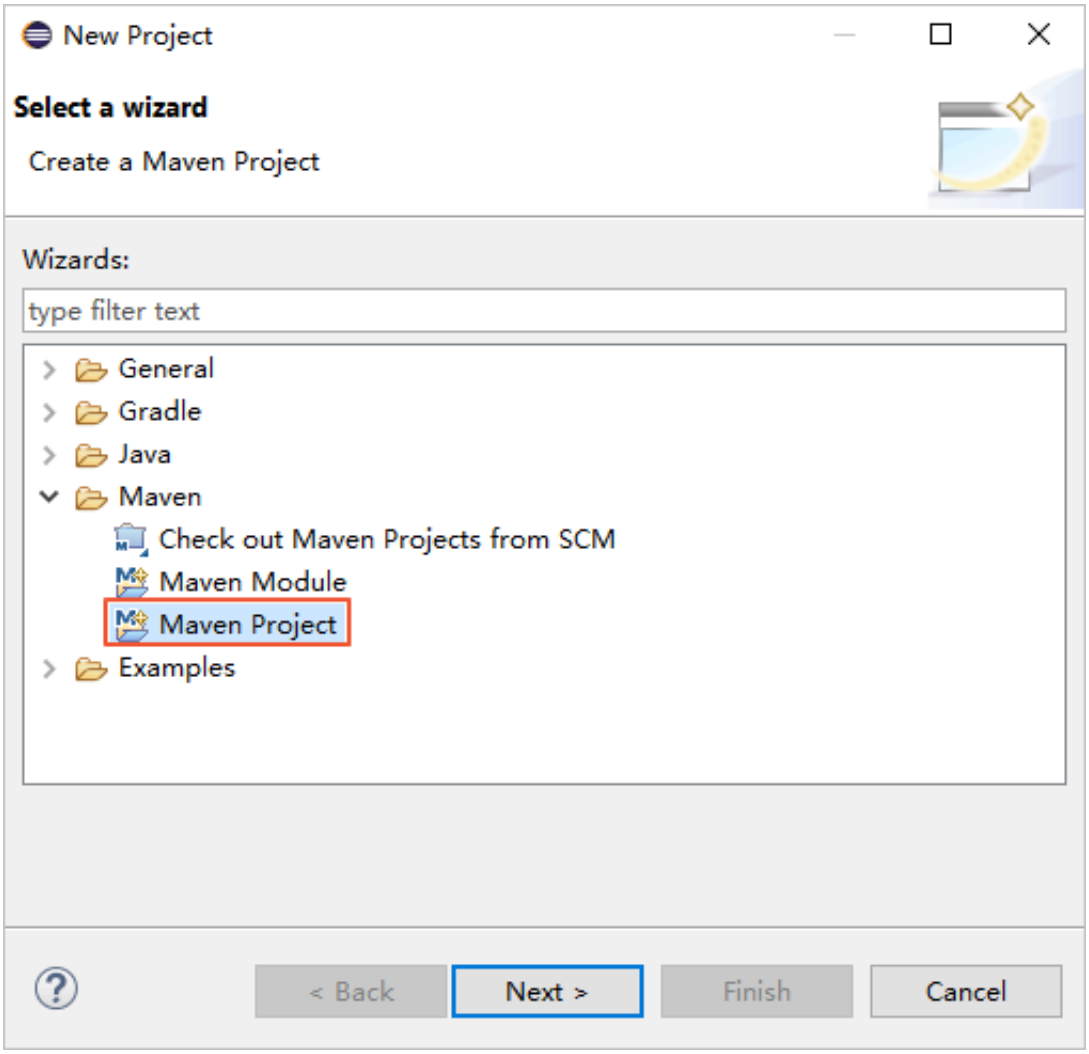
  この例では、jdk1.8.0_211 を使用します。 他の JDK バージョンの互換性は保証されません。公式 Web サイト から JDK をダウンロードできます。

- Java プログラム開発ツールを準備します。

  この例では、Eclipse を使用しています。 その他のツールも選択できます。

このドキュメントで提供するサンプルコードは、Java high-level REST Client 6.3.x に基づいています。 他のクライアントバージョンの互換性は保証されません。

1. Maven プロジェクトを作成します。



次の図は、Maven プロジェクトの設定を示しています。

図 7-1：プロジェクト名と場所を選択します。

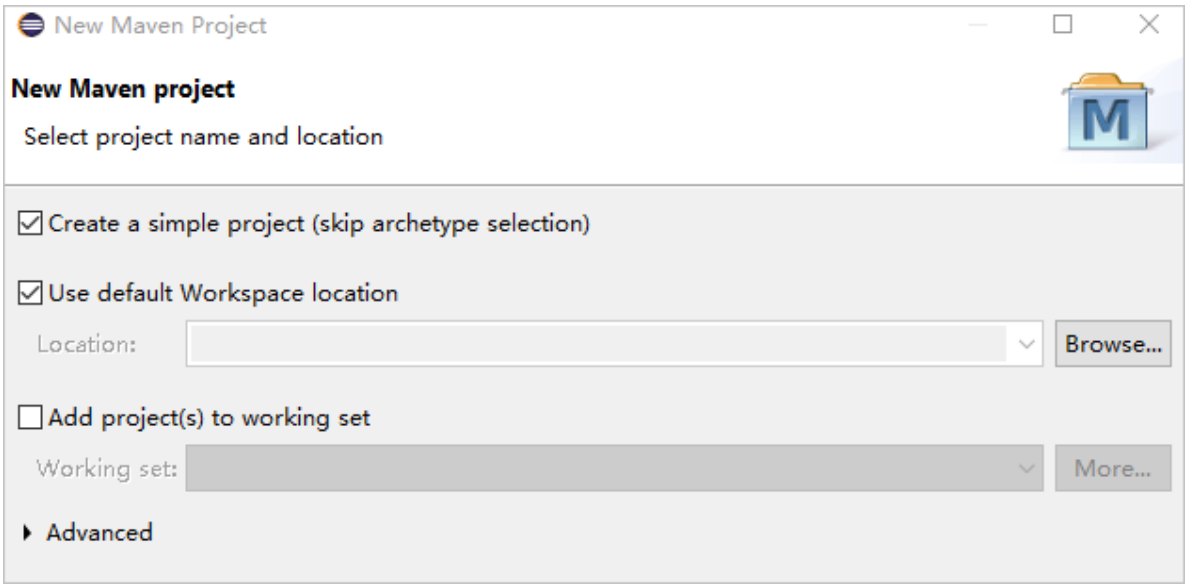
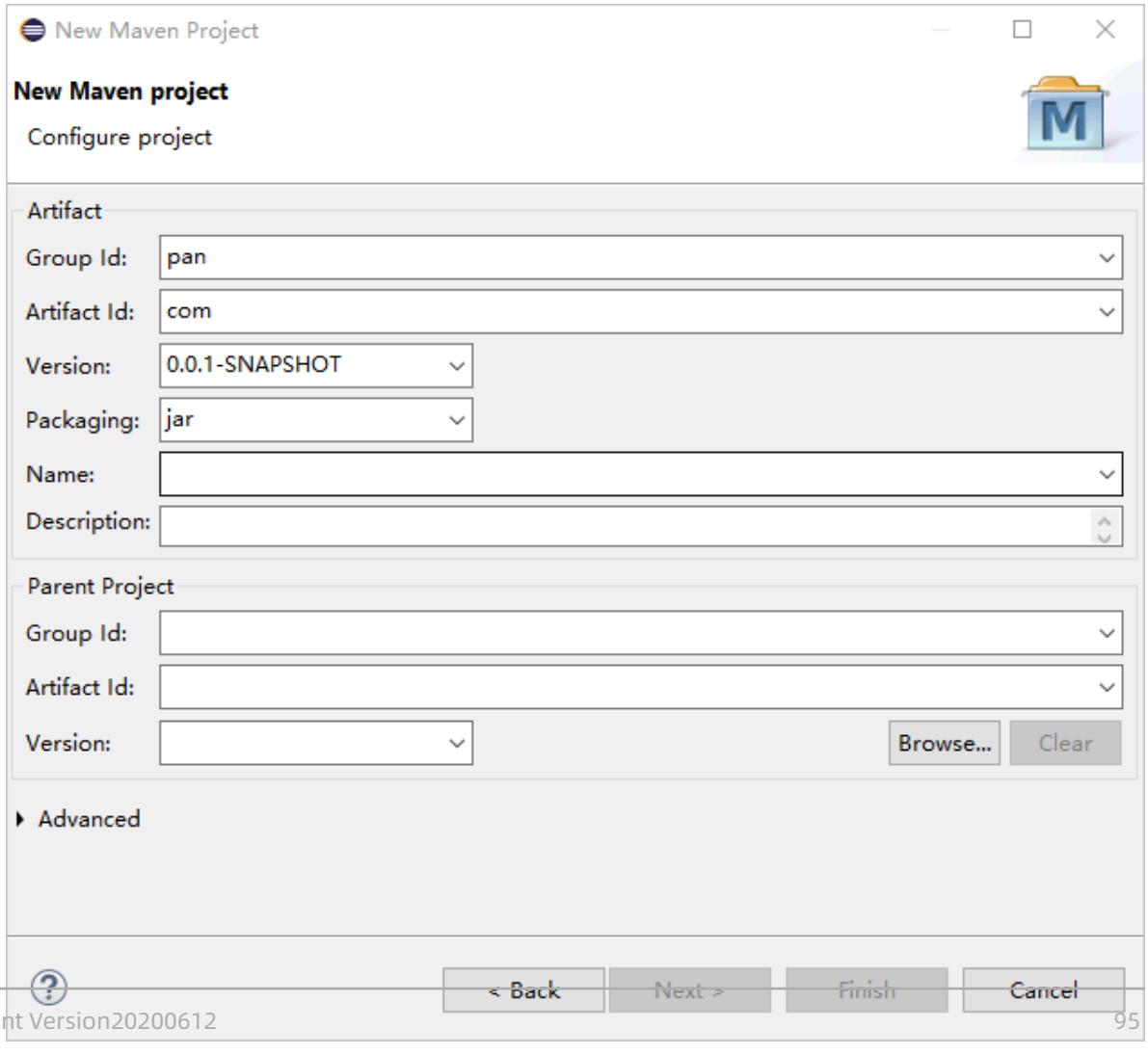
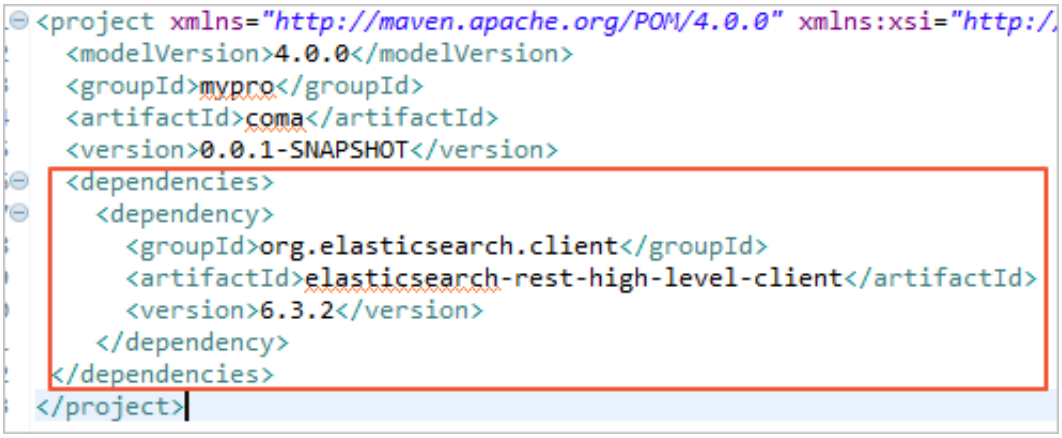


図 7-2：プロジェクトを設定します。



2. 次の依存関係を pom.xml ファイルに追加します。

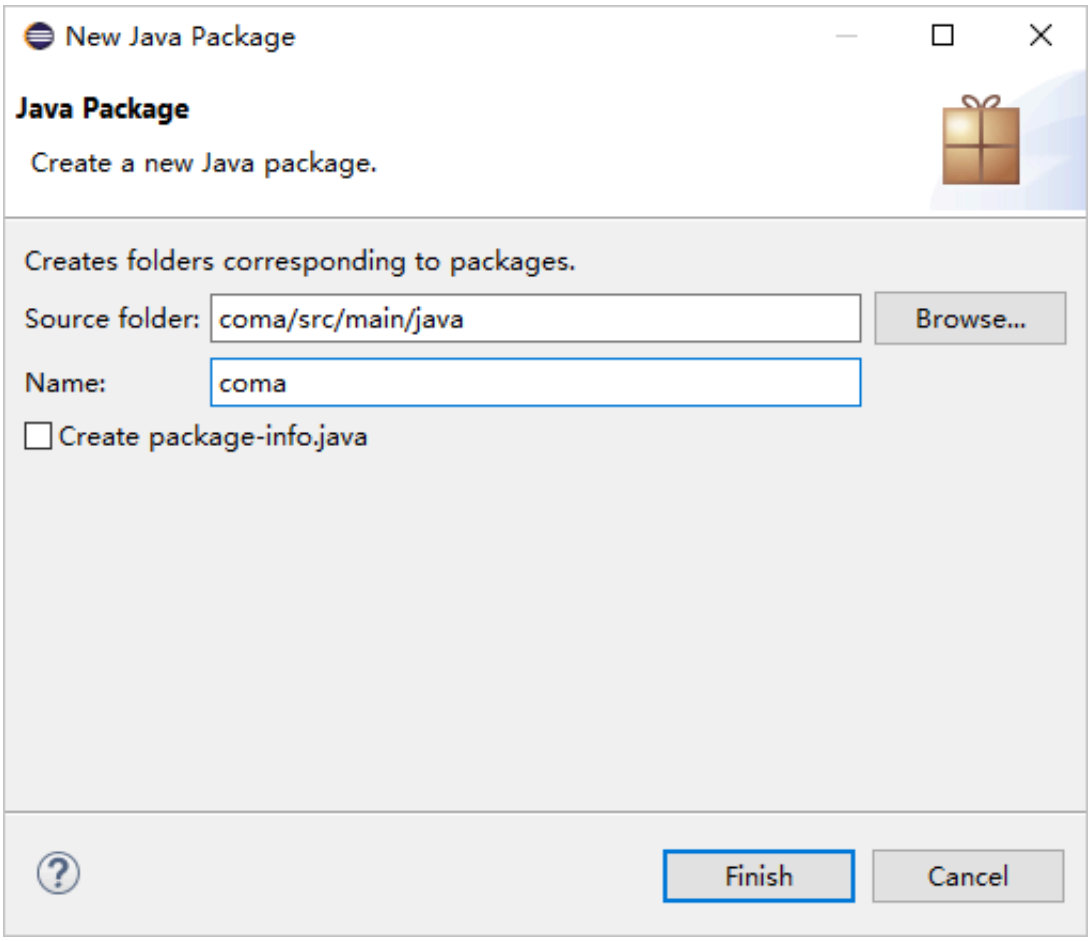



```
<dependency>
    <groupId>org.elasticsearch.client</groupId>
    <artifactId>elasticsearch-rest-high-level-client</artifactId>
    <version>6.3.2</version>
```

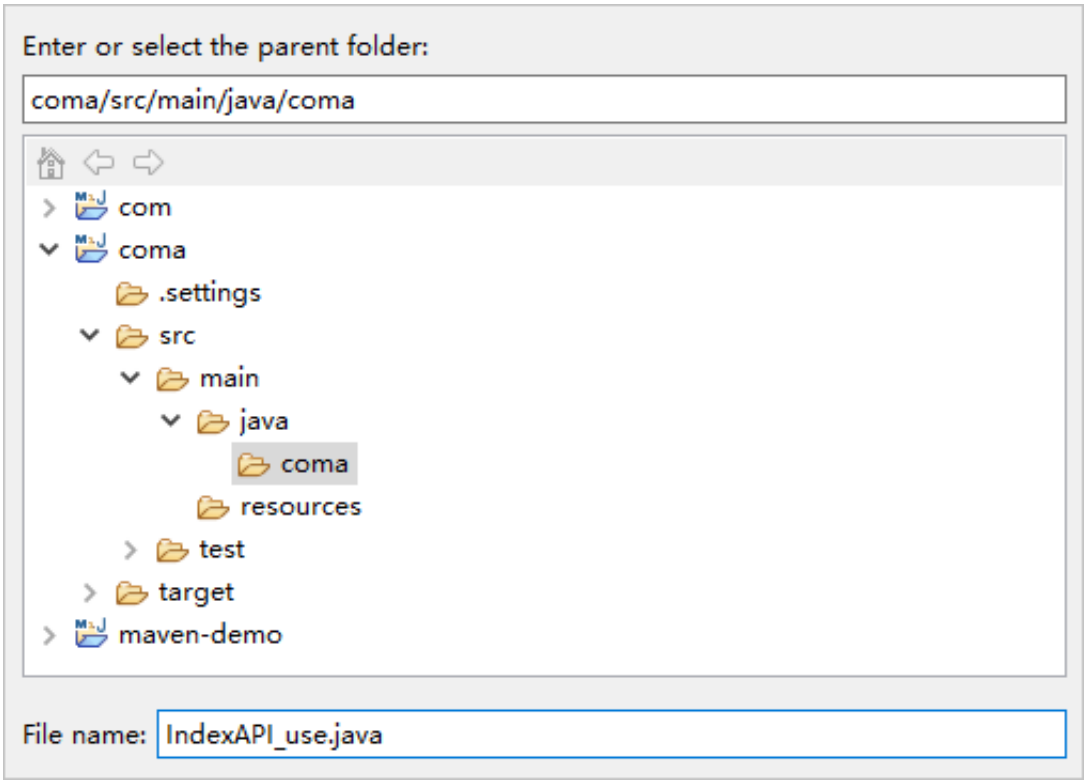
```
</dependency>
```

3. パッケージを作成し、パッケージに必要な Java ファイルを作成します。

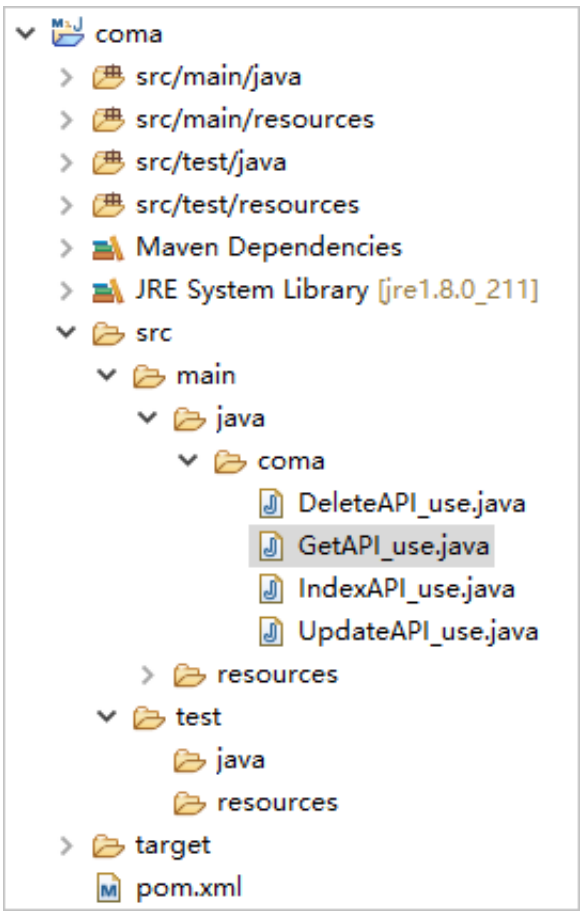
   パッケージの設定は以下のとおりです。



   次の図は、必要な Java ファイルを示しています。



次の図は、Maven プロジェクトのディレクトリを示しています。



- IndexAPI_use.java ：インデックスとドキュメントを作成します。
- UpdateAPI_use.java：ドキュメントを更新します。
- GetAPI_use.java：ドキュメントを取得します。
- DeleteAPI_use.java：ドキュメントを削除します。

注：

この例は参照用のみです。 ビジネス要件を満たすために、他の Java ファイルを作成することもできます。

4. 次のコードを IndexAPI_use.java ファイルに追加して Alibaba Cloud Elasticsearch クラスターに接続します。次に、Java IndexRequest 操作を呼び出して、インデックスを作成します。

   コードスニペットは次のとおりです。

```
//Connect to the Elasticsearch cluster.
final CredentialsProvider credentialsProvider = new BasicCredentialsProvider();

      credentialsProvider.setCredentials(AuthScope.ANY,
            new UsernamePasswordCredentials("elastic", "Your password"));


      RestClientBuilder builder = RestClient.builder(new HttpHost("Your Elasticsearch
instance ID.public.elasticsearch.aliyuncs.com", 9200))
            .setHttpClientConfigCallback(new RestClientBuilder.HttpClientConfigCallback
() {

                public HttpAsyncClientBuilder customizeHttpClient(HttpAsyncClientBuilder
httpClientBuilder) {
                  return httpClientBuilder.setDefaultCredentialsProvider(credential
sProvider);
                }
            });
```

```
//Call the Java IndexRequest operation to create an index.
RestHighLevelClient client = new RestHighLevelClient(builder);
      try {
         IndexRequest request = new IndexRequest();
         request.index("apitest_index");
         request.type("apitest_type");
         request.id("1");
         Map<String, Object> source = new HashMap<>();
         source.put("user", "kimchy");
         source.put("post_date", new Date());
         source.put("message", "trying out Elasticsearch");
         request.source(source);
         try {
            IndexResponse result = client.index(request);
            System.out.println(result);
         } catch (IOException e) {
            e.printStackTrace();
         }
      } finally {
         client.close();
```

```
    }
```

完全なサンプルコードは、 サンプルコード からダウンロードできます。 返される結果は次のとおりです。

```
IndexResponse
[
index=apitest_index,
type=apitest_type,
id=1,
version=1,
result=created,
seqNo=0,
primaryTerm=1,
shards={"total":2,"successful":1,"failed":0}
]
```

5. 次のコードを UpdateAPI_use.java ファイルに追加し、Java UpdateRequest 操作を呼び出してドキュメントを更新します。

   コードスニペットは次のとおりです。

```
RestHighLevelClient client = new RestHighLevelClient(builder);
    try {

        UpdateRequest updateRequest = new UpdateRequest("apitest_index", "
apitest_type", "1");
        IndexRequest indexRequest = new IndexRequest("apitest_index", "apitest_type",
 "1");
        Map<String, String> source = new HashMap<>();
        source.put("user", "dingw2");
        indexRequest.source（source）;
        updateRequest.doc(indexRequest);
        UpdateResponse result = client.update(updateRequest);
        System.out.println(result);

    }catch (IOException e) {
        e.printStackTrace();
    } finally {
        client.close();
    }
```

完全なサンプルコードは、 サンプルコード からダウンロードできます。 返される結果は次のとおりです。

```
UpdateResponse
[
index=apitest_index,
type=apitest_type,
id=1,
version=2,
seqNo=1,
primaryTerm=1,
result=updated,
shards=ShardInfo{
   total=2,
   successful=2,
   failures=[]
```

```
  }
]
```

6. 次のコードを GetAPI_use.java ファイルに追加し、Java GetRequest 操作を呼び出してドキュメントを取得します。

   コードスニペットは次のとおりです。

```
RestHighLevelClient client = new RestHighLevelClient(builder);
    try {
          GetRequest request = new GetRequest("apitest_index", "apitest_type", "1");
          GetResponse result = client.get(request);
          System.out.println(result);

       }catch (IOException e) {
             e.printStackTrace();
       } finally {
          client.close();
       }
```

   完全なサンプルコードは、 サンプルコード からダウンロードできます。 返される結果は次のとおりです。

```
{
   "_index": "apitest_index",
   "_type": "apitest_type",
   "_id": "1",
   "_version": 2,
   "found": true,
   "_source": {
      "post_date": "2019-06-10T05:50:52.752Z",
      "message": "trying out Elasticsearch",
      "user": "dingw2"
   }
}
```

7. 次のコードを DeleteAPI_use.java ファイルに追加し、Java DeleteRequest 操作を呼び出してドキュメントを削除します。

   コードスニペットは次のとおりです。

```
RestHighLevelClient client = new RestHighLevelClient(builder);
    try {
          DeleteRequest request = new DeleteRequest("apitest_index", "apitest_type", "1
");
          DeleteResponse result = client.delete(request);
          System.out.println(result);
       } catch(Throwable e) {
          e.printStackTrace();
       } finally {
          client.close();
```

```
    }
```

完全なサンプルコードは、 サンプルコード からダウンロードできます。 返される結果は次のとおりです。

```
DeleteResponse
[
index=apitest_index,
type=apitest_type,
id=1,
version=3,
result=deleted,
shards=ShardInfo{
   total=2,
   successful=2,
   failures=[]
 }
]
```

ビジネス要件を満たすために、その他のドキュメント API 操作を呼び出すことができます。 その他のドキュメント API 操作の呼び出しの詳細については、「ドキュメント API 」をご参照ください。

8 インデックス管理

8.1 Curator の使用

Elasticsearch Curator のインストール

1. Elasticsearch インスタンスと同じ VPC ネットワークにある ECS インスタンスを購入します。この例では、CentOS 7.3 64 ビットを実行する ECS インスタンスを使用します。
2. 以下のコマンドを実行します。

   a. Elasticsearch Curator のインストール

   ```
   pip install elasticsearch-curator
   ```

   > 注 :
   > - Elasticsearch Curator 5.6.0 をインストールすることを推奨します。 このバージョンは、Elasticsearch 5.5.3 と 6.3.2 をサポートします。
   > - Curator と Elasticsearch のバージョンの互換性

   b. Curator のバージョンの表示

   ```
   curator --version
   ```

   返されるバージョン情報

   ```
   curator, version 5.6.0
   ```

Singleton コマンドラインインターフェイス

- `curator_cli` を使用して、操作を実行します。
- Singleton コマンドラインインターフェイス

> 注 :
> - 一度に実行できる操作は 1 つだけです。
> - Curator を使用してすべての操作を実行できるわけではありません (Alias や Restore など)。

Crontab によるタスクのスケジュール

crontab コマンドと curator コマンドを使用して、複数の操作を実行するようにタスクをスケジュールすることができます。

Curator コマンド：

```
curator [OPTIONS] ACTION_FILE
オプション：
 --config PATH 設定ファイルへのパス。 デフォルト：~/.curator/curator.yml
 --dry-run 何も変更しません。
 --version  バージョンを表示して終了します。
 --help     このメッセージを表示して終了します。
```

- curator コマンドを実行するとき、config.yml ファイル (公式リファレンス) を指定する必要があります。
- curator コマンドを実行するとき、 action.yml ファイル (公式リファレンス) を指定する必要があります。

Hot-Warm アーキテクチャの演習

Curator を使用して hot ノードから warm ノードにインデックスを移行します (公式リファレンス)。

hot ノードから warm ノードへのインデックスの移行

1. 以下のように、**config.yml** ファイルをパス /usr/curator/ に作成します。

    :

   - **hosts**：アクセスする Elasticsearch インスタンスのアドレスに置き換えます。 この例では、Elasticsearch インスタンスのプライベートアドレスを使用しています。
   - **http_auth**：Elasticsearch インスタンスへのログインに使用するユーザー名とパスワードに置き換えます。

   ```
   client:
     hosts:
       - http://es-cn-0pp0z9p2v00031234.elasticsearch.aliyuncs.com
     port: 9200
     url_prefix:
     use_ssl: False
     certificate:
     client_cert:
     client_key:
     ssl_no_validate: False
     http_auth: user:password
     timeout: 30
     master_only: False
   logging:
     loglevel: INFO
   ```

```
  logfile:
  logformat: default
  blacklist: ['elasticsearch', 'urllib3']
```

2. 以下のように、**action.yml** ファイルをパス /usr/curator/ に作成します。

注 :

- 以下のコンテンツでは、30 分前に作成され、**logstash-** で始まるインデックスを **hot** ノードから **warm** ノードに移行します。
- 以下のコンテンツは、ビジネスニーズに合わせてカスタマイズできます。

```
actions:
  1:
    action: allocation
    description: "Apply shard allocation filtering rules to the specified indices"
    options:
      key: box_type
      value: warm
      allocation_type: require
      wait_for_completion: true
      timeout_override:
      continue_if_exception: false
      disable_action: false
    filters:
    - filtertype: pattern
      kind: prefix
      value: logstash-
    - filtertype: age
      source: creation_date
      direction: older
      timestring: '%Y-%m-%dT%H:%M:%S'
      unit: minutes
      unit_count: 30
```

3. curator コマンドが正常に実行されるかどうか確認します。

```
curator --config /usr/curator/config.yml /usr/curator/action.yml
```

コマンドが正常に実行されると、以下の情報が返されます。

```
2019-02-12 20:11:30,607 INFO      Preparing Action ID: 1, "allocation"
2019-02-12 20:11:30,612 INFO      Trying Action ID: 1, "allocation": Apply shard
allocation filtering rules to the specified indices
2019-02-12 20:11:30,693 INFO      Updating index setting {'index.routing.allocation.
require.box_type': 'warm'}
2019-02-12 20:12:57,925 INFO      Health Check for all provided keys passed.
2019-02-12 20:12:57,925 INFO      Action ID: 1, "allocation" completed.
2019-02-12 20:12:57,925 INFO      Job completed.
```

4. crontab コマンドを実行して、15 分間隔で curator コマンドを実行します。

```
*/15 * * * * curator --config /usr/curator/config.yml /usr/curator/action.yml
```

9 ベクトル検索プラグインのベストプラクティス

aliyun-knn プラグインは、Alibaba Cloud Elasticsearch (ES) が設計したベクトル検索エンジンです。 Alibaba DAMO Academy で設計されたベクトル検索エンジンである proxima のベクトルデータベースを使用しています。 このプラグインは、画像検索、動画フィンガープリンティング、顔認識、音声認識、商品推奨などのシナリオで、ベクトル空間に基づくデータ検索要件への対応に役立ちます。

Alibaba Cloud Elasticsearch のベクトル検索エンジンは、Pailitao、Image Search Cloud、Youku 動画フィンガープリンティング、Qutoutiao 動画フィンガープリンティング、Taobao 商品推奨、カスタマイズ検索、CrossMedia 検索など、Alibaba Group 内の本番環境シナリオで継続的に使用されています。

Alibaba Cloud Elasticsearch は、ベクトル検索エンジンにアクセスするための aliyun-knn プラグインを提供します。 これにより、エンジンの使用方法を学ぶ必要がなくなります。 aliyun-knn プラグインは、すべてのネイティブ Elasticsearch バージョンと互換性があります。 このベクトル検索エンジンは、リアルタイム増分同期とほぼリアルタイム (NRT) 検索に加えて、マルチレプリカ、復元、スナップショットなど、分散検索でのネイティブ Elasticsearch の他の機能もサポートしています。

Alibaba Cloud Elasticsearch のベクトル検索エンジンは、Hierarchical Navigable Small World (HNSW) と Linear Search のアルゴリズムをサポートします。 これらのアルゴリズムは、インメモリストレージから少量のデータを処理するのに適しています。 次の表は、2 つのアルゴリズムのパフォーマンスを比較しています。

表 9-1 : HNSW と Linear Search のパフォーマンス比較

2 つのアルゴリズムのパフォーマンスは、Alibaba Cloud Elasticsearch V6.7.0 で測定しています。 テスト環境の設定を次に示します。

- ノード設定：2 つのデータノードと 100 GB のクラウド SSD。 各ノードは 16 コア、64 GB メモリを搭載。
- データセット：SIFT 128 ディメンション float 型ベクトル (http://corpus-texmex.irisa.fr/)。
- 総サンプル数：2000 万。
- インデックス設定：デフォルト設定。

パフォーマンス指標	HNSW	Linear Search
リコール率トップ 10	98.6%	100%
リコール率トップ 50	97.9%	100%
リコール率トップ 100	97.4%	100%
遅延 (p99)	0.093 秒	0.934 秒
遅延 (p90)	0.018 秒	0.305 秒

 注：

p はパーセントの略です。 たとえば、遅延 (p99) は、99% のクエリのレスポンスに要する秒数を示します。

インデックス計画

アルゴリズム	シナリオ	インメモリストレージ	備考
HNSW	• ノードごとに少量のデータのみが保存されます。 • 低レイテンシのレスポンスを必要とします。 • 高いリコール率を必要とします。	はい	• HNSW は、貪欲な検索に基づいており、三角不等式に従います。 三角不等式は、A から B、B から C の辺の合計が A から C の辺よりも大きくなければならないことを示しています。内積空間は三角不等式に従いません。 したがって、HNSW アルゴリズムを適用する前に、ユークリッド空間か球状空間に変換する必要があります。 • Elasticsearch にデータをインポートした後、オフピーク時に定期的に強制マージ API 操作を呼び出してシャード内のセグメントをマージすることを推奨します。 これにより、レスポンスのレイテンシを短縮できます。

アルゴリズム	シナリオ	インメモリストレージ	備考
Linear Search	• ブルートフォース検索。 • 100% のリコール率を必要とします。 • レイテンシは、処理するデータ量とともに増加します。 • 効果の比較。	はい	なし

クラスターサイズ

アイテム	説明
データノード仕様 (必須)	本番環境の最小データノード仕様は、4 コア、16 GB でなければなりません。 2 コア、8 GB の仕様は、テスト目的でのみ使用できます。
ノードあたりの最大データ量	各ノードに保存される最大データ量は、ノードのメモリ容量の 50% です。
入力調整	ベクトルインデックスは、CPU を大量消費するジョブです。 高い入力レートを維持しないことを推奨します。 16 コア、64 GB メモリを備えたデータノードの場合、1 秒あたり 5,000 トランザクション未満のピーク入力レートを推奨します。 Elasticsearch でクエリを処理する際、すべてのインデックスファイルがノードメモリに読み込まれます。 ノードのメモリが不足している場合、Elasticsearch はシャードを再割り当てします。 したがって、クエリの処理中に大量のデータを Elasticsearch にインポートしないでください。

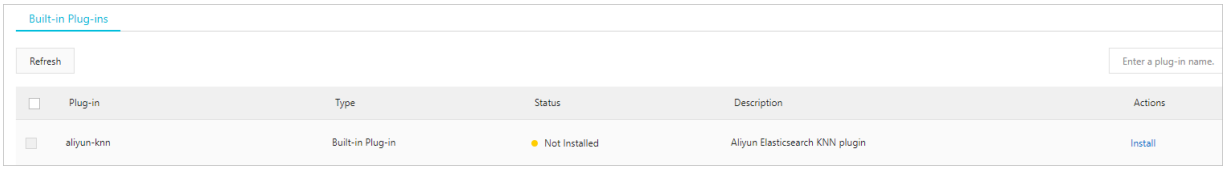
ベクトル検索プラグインのインストール

:

- ベクトル検索プラグインは、Alibaba Cloud Elasticsearch V6.7 以降のバージョンでのみサポートされています。

- プラグインをインストールする前に、データノード仕様が 2 コア、8 GB より高いことを確認してください。 2 コア、8 GB はテスト目的のみです。 本番環境の最小仕様は 4 コア、16 GB です。 Elasticsearch インスタンスが要件を満たしていない場合、最初にデータノードを 2 コア、8 GB より上にアップグレードする必要があります。 詳細については、「#unique_62」をご参照ください。

Alibaba Cloud Elasticsearch コンソールにログインし、**[インスタンス ID]** > **[プラグイン設定]** > **[ビルトインプラグインリスト]** を選択します。 **[ビルトインプラグインリスト]** から **[aliyun-knn]** プラグインを見つけ、インストールします。 詳細については、「#unique_63/unique_63_Connect_42_section_d0y_kyx_fu0」をご参照ください。



:

デフォルトでは、**[aliyun-knn]** プラグインのステータスは **[未インストール]** です。

ベクトル検索プラグインの使用

ベクトル検索プラグインのインストール、Elasticsearch インスタンスの Kibana コンソールへのログインの後、次の手順に従ってプラグインを使用します。

:

次のサンプルコードは、Elasticsearch V6.7 にのみ適用できます。 Elasticsearch V7.4 については、公式ドキュメントをご参照ください。

1. インデックスを作成します。

```
PUT test
{
 "settings": {
  "index.codec": "proxima",
  "index.vector.algorithm": "hnsw"
 },
 "mappings": {
  "_doc": {
   "properties": {
    "feature": {
     "type": "proxima_vector",
     "dim": 2
    },
    "id": {
     "type": "keyword"
    }
   }
  }
```

```
  }
}
```

パラメーター	説明
index.vector.algorithm	有効値：hnsw と linear。
type	フィールドのタイプ。 ベクトル型フィールドを指定するには、値を proxima_vector に設定します。
dim	ベクトルのディメンション。 有効値：1～2048。

上記のサンプルコードでは、test という名前のインデックスが作成されます。 インデックスのタイプは _doc です。 インデックスには 2 つのフィールド、feature と id が含まれます。 必要に応じて、インデックスとフィールドの名前を変更できます。

2. インデックスにドキュメントを追加します。

```
POST test/_doc
{
  "feature": [1.0, 2.0],
  "id": 1
}
```

:

feature フィールドの値は、float 配列でなければなりません。 配列の長さは、mapping の dim パラメーターに指定した長さと同じでなければなりません。

3. インデックス内のデータをクエリします。

```
GET test/_search
{
  "query": {
    "hnsw": {
      "feature": {
        "vector": [1.5, 2.5],
        "size": 10
      }
    }
  }
}
```

パラメーター	説明
hnsw	algorithm は、インデックスの作成時に指定したものと同じでなければなりません。
vector	float 配列。 配列の長さは、mapping の dim パラメーターに指定した長さと同じでなければなりません。
size	返される上位のドキュメント数。

パラメーター

表 9-2：アルゴリズムのパラメーター

パラメーター	説明	デフォルト
index.vector.algorithm	インデックスの作成に使用されるアルゴリズム。 有効値：hnsw と linear。	hnsw

表 9-3：HNSW の入力パラメーター

パラメーター	説明	デフォルト
index.vector.hnsw.builder.max_scan_num	最悪の事態が起きた場合、グラフ作成中にスキャンされる最近傍の最大数。	100000
index.vector.hnsw.builder.neighbor_cnt	レイヤー 0 での各ノードの最近傍の最大数。 値を 100 に設定することを推奨します。 非アクティブなインデックスを保存するために消費されるストレージの量。 グラフの品質は、このパラメーターの値とともに増加します。	100
index.vector.hnsw.builder.upper_neighbor_cnt	レイヤー 0 以外のレイヤーでの各ノードの最近傍の最大数。 このパラメーターを neighbor_cnt の 50% に設定することを推奨します。	50
index.vector.hnsw.builder.efconstruction	グラフ作成中にスキャンされる最近傍の数。 スキャンされる最近傍の数が多いほど、グラフの品質は向上します。 ただし、インデックスの作成には時間がかかります。 このパラメーターを 400 に設定することを推奨します。	400
index.vector.hnsw.builder.max_level	レイヤー 0 を含むレイヤーの総数。 1,000 万件のドキュメントがあり、 scaling_factor パラメーターが 30 に設定されているとします。 30 を基数として使用し、10000000 の対数を最も近い整数に切り上げます。 デフォルトは 5 です。	6
index.vector.hnsw.builder.scaling_factor	スケーリング係数。 レイヤーのデータ量は、上位レイヤーのデータ量にスケーリング係数を乗じたものに等しくなります。 10～100 の値を指定してください。 レイヤーの数は、scaling_factor の値とともに減少します。 このパラメーターを 50 に設定することを推奨します。	50

表 9-4：HNSW の検索パラメーター

パラメーター	説明	デフォルト
ef	オンライン検索中にスキャンされる最近傍の数。 値を大きくすると、リコール率は高くなりますが、検索は遅くなります。 有効値：100～1000。	100

サンプルリクエスト：

```
GET test/_search
{
  "query": {
    "hnsw": {
      "feature": {
        "vector": [1.5, 2.5],
        "size": 10,
        "ef": 100
      }
    }
  }
}
```

表 9-5：サーキットブレーカーのパラメーター

パラメーター	説明	デフォルト
indices.breaker.vector.native.indexing.limit	オフヒープメモリ使用量がこのパラメーターで指定された値を超えると、書き込み操作は中断されます。 Elasticsearch がインデックスを作成し、メモリをリリースすると、書き込み操作が再開されます。 このサーキットブレーカーがトリガーされた場合、システムメモリの消費量が多いことを意味します。 入力レートを調整することを推奨します。 初級ユーザーの場合は、デフォルト設定を使用することを推奨します。	70%
indices.breaker.vector.native.total.limit	ベクトルインデックスの作成に使用されるオフヒープメモリの最大割合。 実際のオフヒープメモリ使用量が、指定された割合を超えると、Elasticsearch はシャードを再割り当てする場合があります。 初級ユーザーの場合は、デフォルト設定を使用することを推奨します。	80%

FAQ

- Q：ドキュメントのリコール率をどのように求めればよいですか。

  A：2 つのインデックスを作成し、1 つはHNSW アルゴリズム、もう 1 つは Linear Search アルゴリズムを 使用します。その他のインデックス設定は、どちらのインデックスも同じにします。 クライアントから両方のインデックスに同じベクトルデータを追加し、インデックスを更新します。 同じクエリベクトルに対し、HNSW インデックスと Linear Search インデックスから返されるドキュメント ID を比較し、同じドキュメント ID を見つけます。

   注：
  どちらのインデックスからも返されたドキュメント ID の数を、返されたドキュメント ID の総数で除算して、ドキュメントのリコール率を算出します。

- Q：データを Elasticsearch にインポートするときに circuitBreakingException エラーが発生した場合、このエラーを解決するにはどうすればよいですか。

  A：このエラーは、オフヒープメモリ使用量が、ndices.breaker.vector.native.indexing.limit パラメーターで指定された割合を超えたことを示します。 デフォルトの割合は 70% です。 書き込み操作は中断されます。 ほとんどの場合、Elasticsearch がインデックスを作成し、メモリをリリースすると、自動的に書き込み操作が再開されます。 クライアントのデータインポートスクリプトに再試行メカニズムを追加することを推奨します。

- Q：書き込み操作が中断された後、CPU が動作し続けているのはなぜですか。

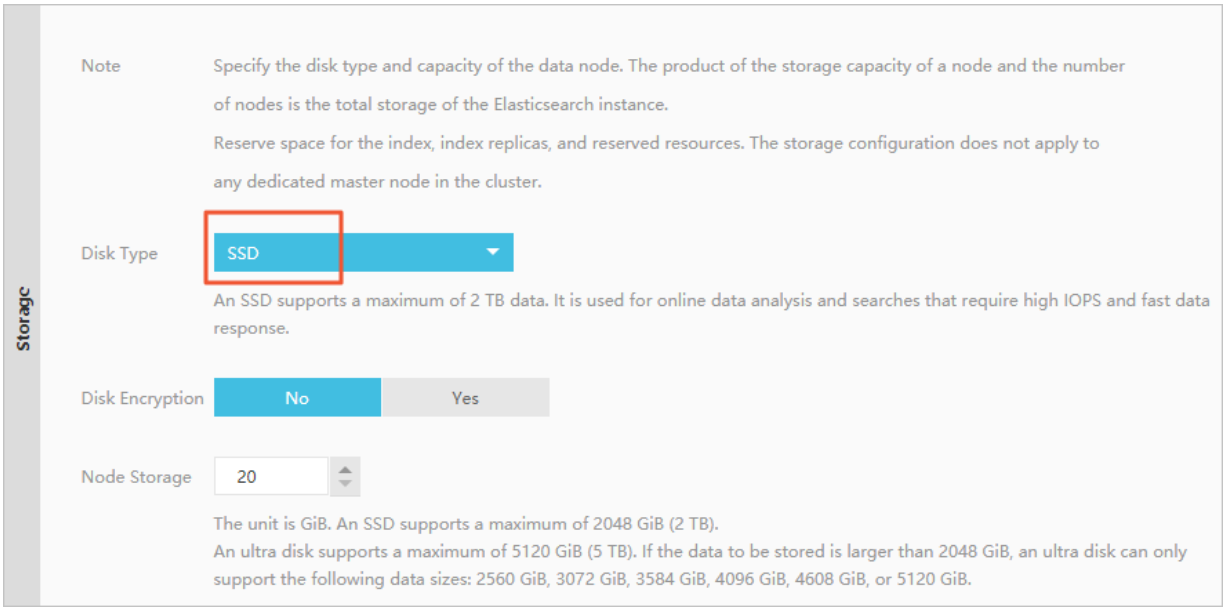
  A：Elasticsearch は、更新またはフラッシュ中にベクトルインデックスを作成します。 書き込み操作が中断された場合でも、ベクトルインデックス作成タスクは実行されている場合があります。 更新の最終ラウンドが完了すると、コンピューティングリソースはリリースされます。

10 Alibaba Cloud Elasticsearch サイジングのベストプラクティス

Alibaba Cloud Elasticsearch を使用する前に、必要なリソースの合計量を見積もる必要があります。 テスト結果とユーザーのフィードバックに基づいて、Alibaba Cloud は Elasticsearch リソースの量を推定し計算する一般的な方法をいくつか提供します。 これらのメソッドは参照専用です。

該当するディスクタイプ

このベストプラクティスは、**[ディスクタイプ]** が **[SSD クラウドディスク]** に設定されている Alibaba Cloud Elasticsearch インスタンスに適用できます。



ディスクのサイジング

Alibaba Cloud Elasticsearch インスタンスのディスク容量は、次の要因によって決まります。

- レプリカの数。 各インデックスには、少なくとも 1 つのレプリカが必要です。
- インデックス作成のオーバーヘッド。 一般的に、インデックス作成のオーバーヘッドは、ソースデータより **10**% 大きくなります。 **_all** パラメーターのインデックス作成のオーバーヘッドは含まれていません。
- オペレーティングシステムの予約領域。 オペレーティングシステムは、重要なプロセス、システムリカバリ、およびディスクフラグメントのために、ディスク容量の **5**% をデフォルトで予約します。

- Alibaba Cloud Elasticsearch のオーバーヘッド。 Alibaba Cloud Elasticsearch は、セグメントのマージ、ログ、およびその他の内部的な操作のために、ディスク容量の **20**% を予約します。
- セキュリティしきい値のオーバーヘッド。 ディスク容量の最低 **15**% をセキュリティのしきい値として予約する必要があります。

これらの要因に基づいて、必要な最小ディスク容量は次のように計算されます。最小ディスク容量 **=** ソースデータのサイズ **× 3.4**

```
合計ディスク容量 = ソースデータのサイズ × (1 + レプリカの数) × (1 + インデックス作成のオーバーヘッド)/(1 - オペレーティングシステムの予約領域)/(1 - Elasticsearch のオーバーヘッド)/(1 - セキュリティしきい値のオーバーヘッド)
= ソースデータのサイズ × (1 + レプリカの数) × 1.7
= ソースデータのサイズ × 3.4
```

:

- ビジネスで必要でない限り、 **_all** パラメーターを有効にしないことを推奨します。
- このパラメーターを有効にしたインデックスでは、ディスク使用率に大きなオーバーヘッドが発生します。 テスト結果と実践に基づいて、推定ディスク容量の 50% を最終的なディスク容量に追加することを推奨します。

```
合計ディスク容量 = ソースデータのサイズ × (1 + レプリカの数) × 1.7 × (1 + 0.5)
= ソースデータのサイズ × 5.1
```

クラスター仕様を選択する

Alibaba Cloud Elasticsearch クラスターのパフォーマンスは、クラスター内の Elasticsearch ノードの仕様によって決まります。 Elasticsearch を使用する前に、クラスターのサイズを見積もり、ノードを追加するか、クラスターをアップグレードすることを推奨します。 テスト結果と実践に基づいて、次の提案をします。

- `クラスターあたりの最大ノード数 = 各ノードの CPU コア数 × 5`
- Elasticsearch ノードが保存できるデータの最大量は、シナリオごとに異なります。
  - データクエリの高速化と集約：`ノードあたりの最大データ量 = 各ノードのメモリ (GB) × 10`
  - ログデータのインポートとオフライン分析：`ノードあたりの最大データ量 = 各ノードのメモリ (GB) × 50`
  - 通常のシナリオ：`ノードあたりの最大データ量 = 各ノードのメモリ (GB) × 30`

表 10-1：推奨クラスター仕様

仕様	クラスターノードの最大数	各ノードの最大メモリ (クエリ)	各ノードの最大メモリ (ログ)	各ノードの最大メモリ (共通)
2 コア 4 GB	10	40 GB	200 GB	100 GB
2 コア 8 GB	10	80 GB	400 GB	200 GB
4 コア 16 GB	20	160 GB	800 GB	512 GB
8 コア 32 GB	40	320 GB	1.5 TB	1 TB
16 コア 64 GB	50	640 GB	2 TB	2 TB

シャードサイジング

シャードの数と各シャードのサイズの両方が、Alibaba Cloud Elasticsearch クラスターの安定性とパフォーマンスに影響します。 Elasticsearch の各インデックスは、特定の数のシャードに分割されます。 デフォルトでは、インデックスは 5 つのシャードに分割されます。

- 小規模な Elasticsearch ノードの場合、各シャードのサイズは 30 GB 以下にすることを推奨します。 大規模な Elasticsearchノードの場合、各シャードのサイズは 50 GB 以下にすることを推奨します。
- ログ分析または非常に大きなインデックスの場合、各シャードのサイズは 100 GB 以下にすることを推奨します。
- レプリカを含むシャードの数は、ノードの数に等しいか、ノード数の等倍にしなければなりません。
- ノードのインデックスに最大 5 つのシャードを指定することを推奨します。

注：

- ユーザーによって、データスキーマ、クエリの複雑さ、データサイズ、パフォーマンス、およびデータの変更に関する要件が異なる場合があります。 このドキュメントでは、Elasticsearch のサイジングに関する参照のみを提供しています。
- 可能であれば、実際のデータとサービスのシナリオに基づいて、Elasticsearch クラスターのサイズを測定することを推奨します。
- 重いワークロードに対処する必要がある場合、Alibaba Cloud Elasticsearch のエラスティックスケーリング機能を使用して、ディスクの拡張、ノードの追加、またはノードのアップグレードを行うことができます。