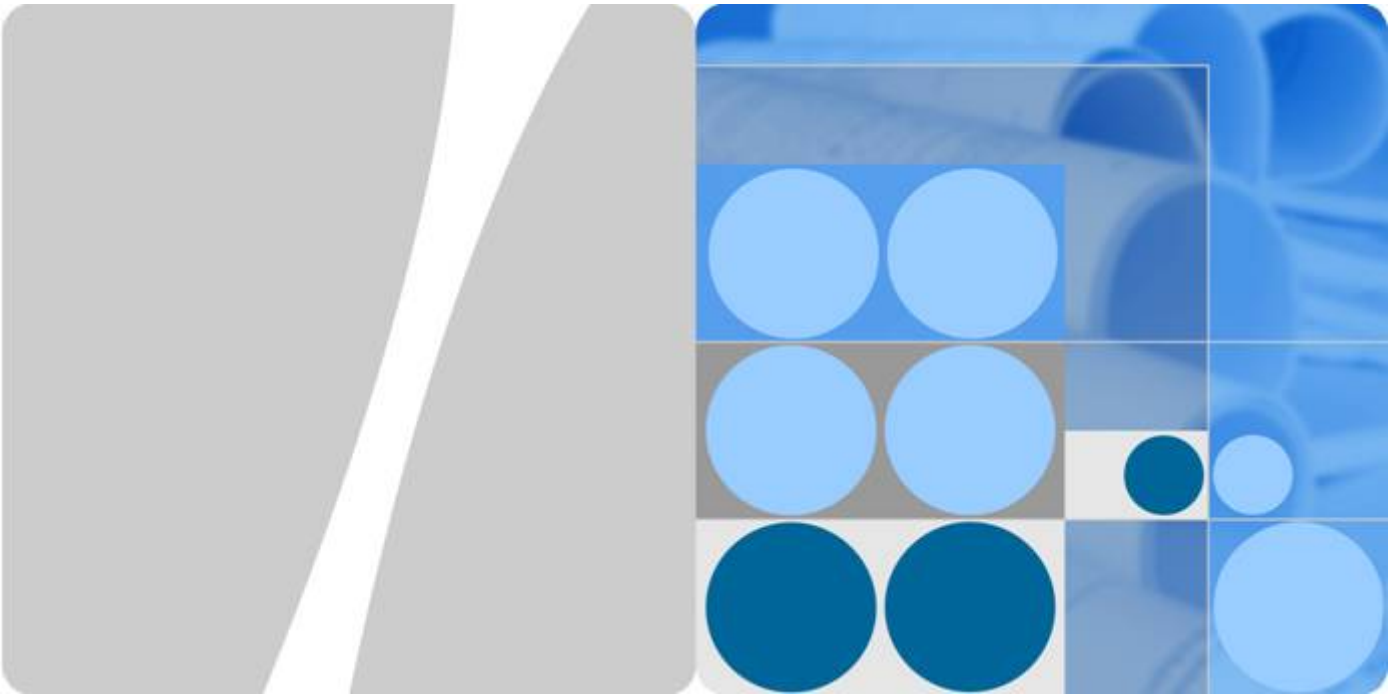


Application Orchestration Service

User Guide

Date **2020-11-05**

Contents

1 Service Overview

1.1 Introduction

Application Orchestration Service (AOS) enables enterprises to automate application cloudification. By orchestrating mainstream cloud services, you can create, replicate, and migrate your applications and provision required cloud resources with a few clicks.

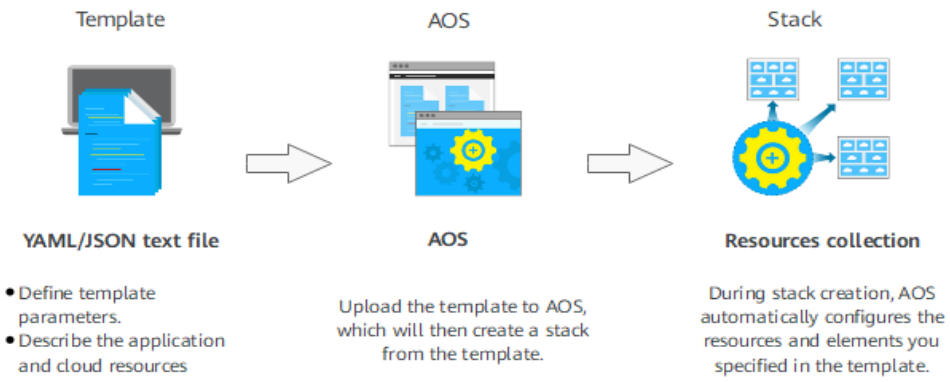
To work with AOS, all you need to do is create a template describing the applications and cloud resources that you would like, including their dependencies and references. AOS will then set up these applications and resources as specified in your template. For example, suppose you want to create an Elastic Cloud Server (ECS), together with a Virtual Private Cloud (VPC) and a subnet on which the ECS runs, you simply create a template defining an ECS, a VPC, a subnet, and their dependencies. AOS will then create a stack, namely, a collection of resources you specified in the template. After the stack has been successfully created, the ECS, VPC, and subnet are available to use.

AOS templates are text files that are easy to read and write. You can edit template files in YAML or JSON format.

AOS manages cloud resources and applications in a unified manner through stacks. During stack creation, AOS automatically configures the cloud resources and applications specified in the template. You can view the status and alarms of cloud resources or applications in a stack. You can create, delete, and copy cloud resources and applications by performing corresponding operations on stacks as a unit.

You can work with AOS on Console or through API.

Figure 1-1 How AOS works



Features

- **Automatic orchestration of resources**

  AOS provides automatic orchestration of mainstream public cloud services. For list on the cloud resources that can be orchestrated, see **Resource Indexes**. AOS also provides lifecycle management including resource scheduling, application design, deployment, and modification to reduce O&M costs through automation.

- **Hybrid orchestration of applications and cloud service resources**

  You can use standard languages, namely YAML and JSON, to describe required basic resources, application systems, upper-layer services, and their relationships. Based on your description, resource provision, application deployment, and service loading can be automatically performed in the order specified by dependencies with a few clicks. You can perform unified management on deployed resources and applications like deletion, scaling, replication, and migration.

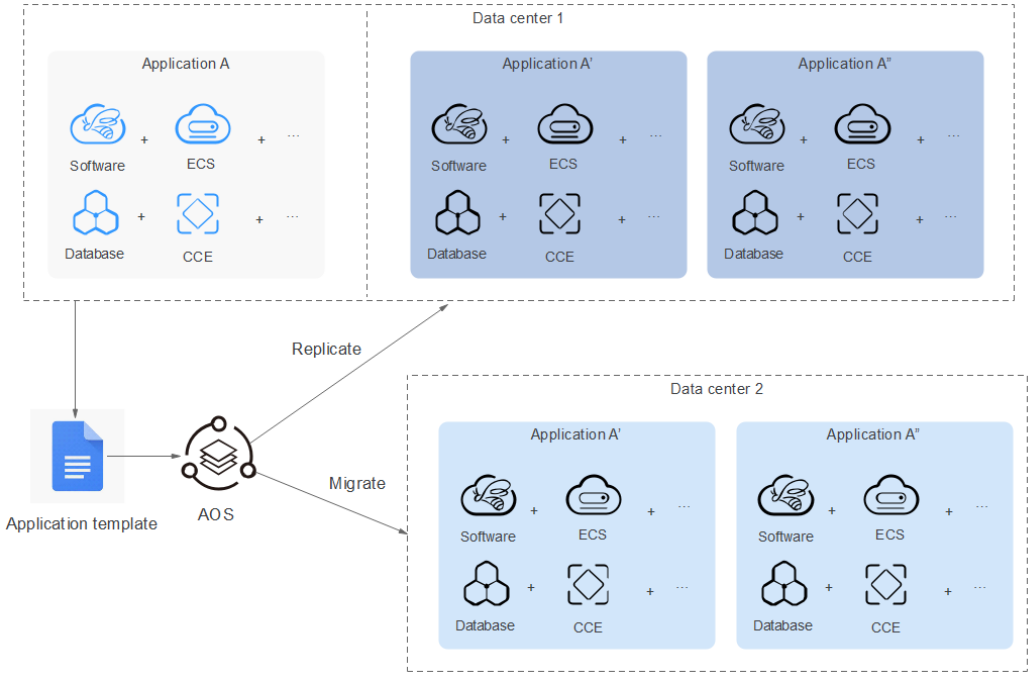
1.2 Advantages

Unified Orchestration of Cloud Services

Currently, AOS allows unified orchestration of mainstream public cloud services. By creating stacks, applications and cloud resources of different types and configurations can be automatically created in batches. In this way, you can perform unified orchestration conveniently and efficiently.

Fast Replication and Migration

AOS allows you to automatically replicate and migrate your business among different regions to ensure your business consistency across different environments. AOS templates allow you to delete and recreate resources and applications freely without any inconsistency. In this way, higher efficiency and reliability can be achieved.



User-Friendly Orchestration Language

- Both YAML and JSON syntax are supported when you are defining template elements.
- To define the configurations, number of instances, and operations of deployed objects, simply modify the input parameters. In this way, AOS enables you to reuse your templates conveniently.
- You can refer to variables, including input parameters, element attributes, and mapping tables during orchestration.
  - Input parameter reference: used to obtain the values of input parameters defined in the **inputs** sections of template files.
  - Element attribute reference: used to obtain the initialization results of the elements other than those in the **inputs** sections in a template. For example, when you are creating an ECS after creating a VPC, you can refer to the ID of the VPC you just created. This method can be used to build dependencies between resources and control the order of resource creation.
  - Mapping table reference: used to obtain the content in mapping tables.

1.3 Application Scenarios

Migrating Applications to the Cloud

Challenges

Migrating applications to the cloud involves repetitive work, such as the destruction and rebuild of environments and manually configuring new instances one by one when scaling out applications. Some operations could be time-consuming, such as creating databases or VMs, which usually take minutes to

finish. You may have to wait longer when these demanding operations need to be performed one by one. In this case, automating the whole process can improve the migrating efficiency and free you from tedious work.

Solution

AOS enables you to schedule resources, define applications, and deploy services at the same time. With a few clicks, operations such as deployment and destruction can be automatically performed. The only thing you need to do is define your applications and corresponding resources through templates.

Advantages

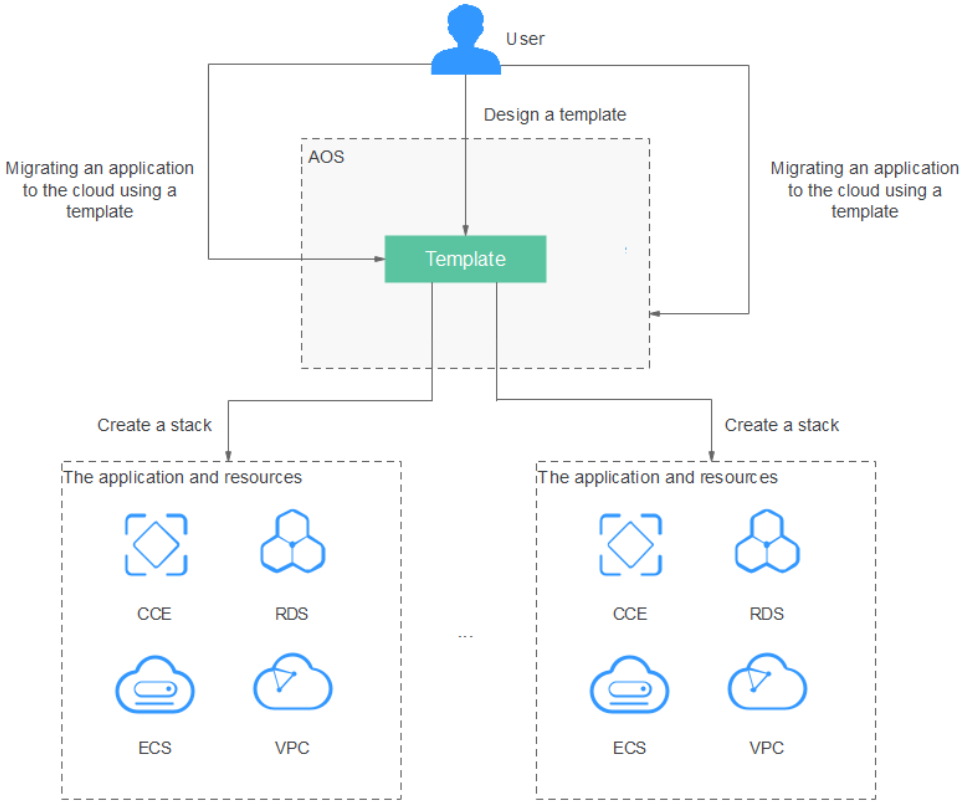
- **Easy to Use**

  Design your applications and schedule resources by writing templates. Organize and manage the service easily and efficiently.
- **Highly efficient**

  Automatically deploy services and destruct environments with a few clicks. Get rid of repetitive work.
- **Quick replication of applications**

  Applications and resources can be quickly replicated and deployed across different data centers. Templates allow you to quickly create identical applications.

Figure 1-2 Migrating applications to the cloud



ISV Service Provisioning

Challenges

Independent software vendors (ISVs) deliver software to their customers and have the software deployed in the cloud to provide services. The traditional delivery method is that ISVs provide the software code and platform building guides on their official websites for customers to download. This could be time demanding and costly, because customers have to create resources, configure networks, perform O&M, and manage updates all on themselves. In addition, the traditional method is complex and error-prone, as all the installation is performed manually.

Benefits

AOS templates enable ISVs to deliver software and required resources in a standard manner. By writing templates and deploying application through AOS, software can be easily delivered and efficiently deployed with a few clicks.

Advantages

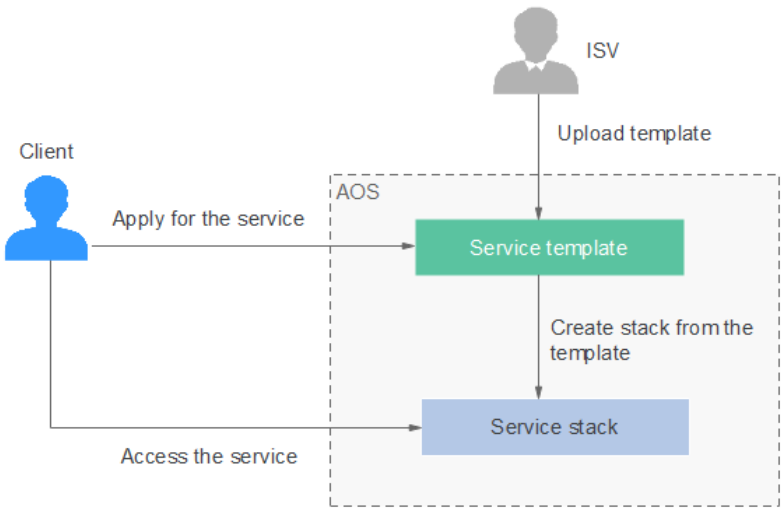
- **Fast delivery**

  AOS automatically deploys the software and provisions the resources as specified in the templates you write. The whole process only takes minutes to accomplish.
- **Error proofing creation**

  All the required resources for the software are defined in the template, which is fixed during delivery and deployment. In this way, mistakes introduced through manual work can be effectively prevented.
- **Unified O&M**

  AOS enables you to perform software lifecycle operations, including updating and scaling, in a unified and easy manner.

Figure 1-3 ISV software delivery



Creating Resources in Batches

Challenges

Assume that you need to create a web application which runs on ten ECSs of different specifications, or you want to create ten databases, you have to create each of them one by one separately, and make sure they can properly work together. The whole process could be complicated and time consuming.

Benefits

With AOS, you can define multiple resources of different services and different specifications in batches in templates, which highly boost the deployment efficiency and brings much more flexibility during configuration.

Advantages

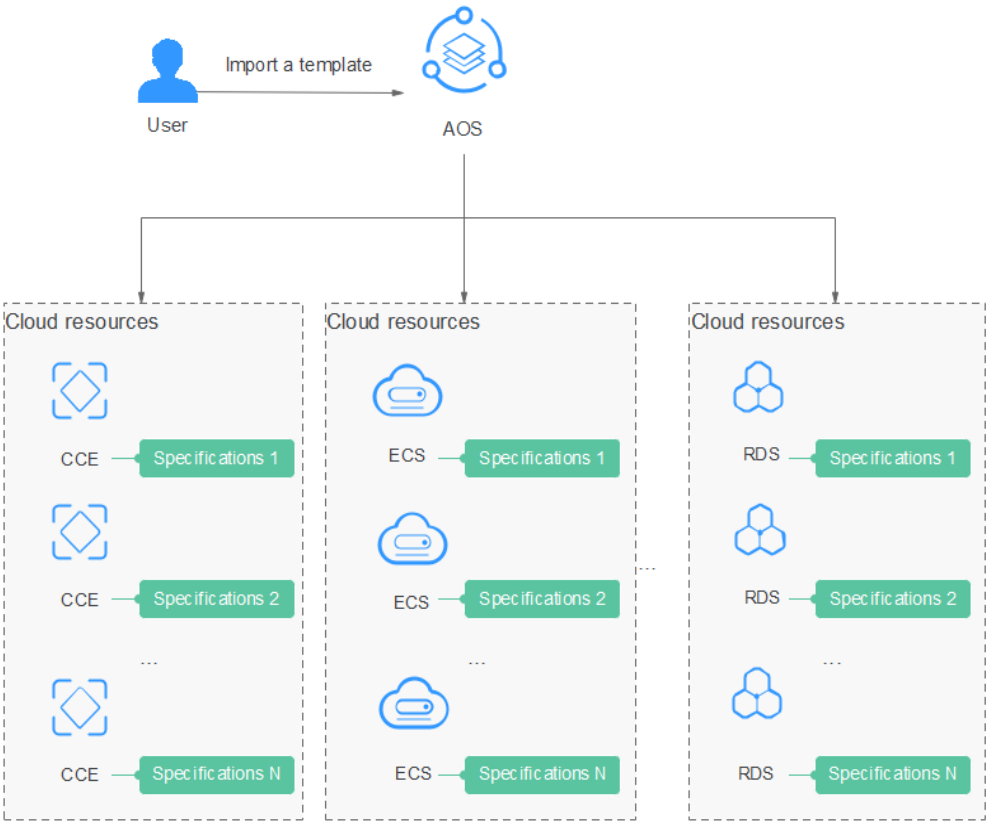
- **Quick deployment**

  Multiple cloud resources of various types or the same type of resources of different specifications can be created concurrently.
- **Flexible configuration**

  A wide variety of template syntax allows you to flexibly create and configure resources of various types and specifications.
- **Automatic rollback**

  When the creation of resources in batches fails, you can choose to perform automatic rollback to save costs.

Figure 1-4 Creating resources in batches



1.4 Basic Concepts

Template

Templates are text files in YAML or JSON format. They describe the cloud objects that you want, including applications, resources, and services. AOS creates various cloud objects automatically from AOS templates. For more information about templates, see **Template Reference**.

Stack

Stacks are collections of applications and cloud resources. The applications or cloud services in a stack are treated as a unit when being created or deleted.

1.5 Notes and Constraints

Quota

Quotas are imposed on the number of templates and stacks a user can create. For details, see **Table 1-1**.

Table 1-1 AOS resource quotas

Resource	Quota
Templates	100
Stacks	200

2 Getting Started

2.1 Writing a Template to Create an ECS

This section describes how to create an Elastic Cloud Server (ECS), including a Virtual Private Cloud (VPC) and subnet by writing a template. An ECS is a computing server equipped with CPUs, memory, images, and Elastic Volume Service (EVS) disks. ECSs can be created on demand and supports auto scaling. A VPC provides logically isolated, configurable, and manageable virtual networks for your ECSs. One or more subnets are automatically created when you create a VPC.

At the end of this walkthrough, you will see the newly created ECS on the Cloud Server Console.

NOTE

In this walkthrough, you will write a template in the YAML language. For more information about templates, see **Template Reference**.

In this section, you will complete the following steps:

1. **Step 1: Write a Template**: Use the YAML language to write a template for creating an ECS, VPC, and subnet.
2. **Step 2: Create an ECS**: Use the template to create an ECS, VPC, and subnet.
3. **Step 3: Delete Unnecessary Resources**: Delete unnecessary stack to avoid unwanted charges.

Step 1: Write a Template

Step 1 Write a simple template to create a VPC.

```
tosca_definitions_version: cloud_tosca_version_1_0   #Template version information
node_templates:                #Element object definition
  myvpc:                      #VPC
    type: Cloud.VPC.VPC
    properties:
      name: my-vpc             #Name of the VPC
      cidr: '192.168.0.0/16'   #VPC CIDR
```

This template includes:

1. **tosca_definitions_version**: specifies the version of a template. Currently, only **cloud_tosca_version_1_0** is supported by AOS.
2. **node_templates**: defines the set of objects to be orchestrated in a template. In AOS, objects are used interchangeably with elements. An object can be an application or cloud service resource. In the preceding template, **node_templates** defines the **myvpc** VPC.
3. **type**: specifies the type of an orchestration object. The value comes from the element type list and can be set to **Cloud.****** (***** indicates the element name in the **Resource Indexes**). In the preceding template, the **myvpc** VPC type is **Cloud.VPC.VPC**.
4. **properties**: defines element properties, which vary with element types. In the preceding template, the **myvpc** VPC has the **names** and **cidr** properties, which indicate the name and network segment of the VPC, respectively. For more information, see **VPC.VPC**.

Step 2 Define a subnet in the VPC. A VPC is a large network segment and is usually divided into several subnets. Define a subnet in the created VPC based on the preceding template.

```
tosca_definitions_version: cloud_tosca_version_1_0   #Template version information
node_templates:                #Element object definition
  myvpc:                     #VPC
    type: Cloud.VPC.VPC
    properties:
      name: my-vpc             #Name of the VPC
      cidr: '192.168.0.0/16'   #VPC CIDR
  mysubnet:                  #Subnet
    type: Cloud.VPC.Subnet
    properties:
      name: my-subnet          #Name of the subnet
      cidr: '192.168.1.0/24'   #Subnet CIDR
      gateway: 192.168.1.1     #Gateway of the subnet
      vpcId:                 #ID of the VPC to which the subnet belongs
        get_reference: myvpc
      dhcpEnable: true         #Determines whether to enable the DHCP function for the subnet in the VPC.
    requirements:              #Dependency between the subnet and VPC.
      - vpcId:
          node: myvpc
```

The **requirements** section specifies the elements that have dependencies with the current element. For example, define **myvpc** as a dependent node in the **requirements** section of the subnet because a subnet depends on a VPC.

Step 3 Define an ECS in the template.

```
tosca_definitions_version: cloud_tosca_version_1_0   #Template version information
node_templates:                #Element object definition
  myvpc:                     #VPC
    type: Cloud.VPC.VPC
    properties:
      name: my-vpc             #Name of the VPC
      cidr: '192.168.0.0/16'   #VPC CIDR
  mysubnet:                  #Subnet
    type: Cloud.VPC.Subnet
    properties:
      name: my-subnet          #Name of the subnet
      cidr: '192.168.1.0/24'   #Subnet CIDR
      gateway: 192.168.1.1     #Gateway of the subnet
      vpcId:                 #ID of the VPC to which the subnet belongs
        get_reference: myvpc
      dhcpEnable: true         #Determines whether to enable the DHCP function for the subnet in the VPC.
    requirements:              #Dependency between the subnet and VPC.
      - vpcId:
          node: myvpc
```

```
 myecs:                        #ECS
   type: Cloud.ECS.CloudServer
   properties:
     name: my-ecs              #Name of the ECS
     instances: 1              #Number of created ECSs
     imageId: 60e757e9-1924-413e-b71f-b7b49bacd2ca    #Image ID used by the ECS. In this template, the image ID is the ID of the system disk based on 64-bit CentOS 7.2.
     flavor: c2.large          #Specifications of the ECS
     vpcId:                    #ID of the VPC to which the ECS belongs. Either a new or an existing VPC ID can be used.
       get_reference: myvpc    #Obtains the dynamic attribute value of the associated element.
     availabilityZone: ae-ad-1a      #AZ to which the ECS belongs
     nics:                           #NIC of the ECS
       - subnetId:
           get_reference: mysubnet
     rootVolume:               #System disk configuration of the ECS
       volumeType: SATA        #Common I/O disk type
       size: 40                #System disk size (unit: GB)
   requirements:               #Dependency among the ECS, VPC, and subnet.
     - vpcId:
         node: myvpc
     - nics.subnetId:
         node: mysubnet
```

Step 4 Save the template as a local file **myecs.yaml**.

Step 5 Log in to the AOS console.

Step 6 In the navigation pane, choose **My Templates**, and then click **Create Template**.

Step 7 On the **Upload File** tab page, specify the following parameters, upload a local YAML file, and then click **Create**. The template details page is then displayed, showing the template information.

- **Template**: Enter a template name. Each template name must be globally unique. For example, set this parameter to **myecs**.
- **Version**: Set this parameter to **1.0**.
- **Select File**: Upload the **myecs.yaml** file.

----End

Step 2: Create an ECS

Step 1 Log in to the AOS console.

Step 2 In the navigation pane, choose **My Templates**. The **myecs** template is displayed in the template list.

Step 3 Click **Create Stack** in the **Operation** column of the **myecs** template.

Step 4 Set the stack information.

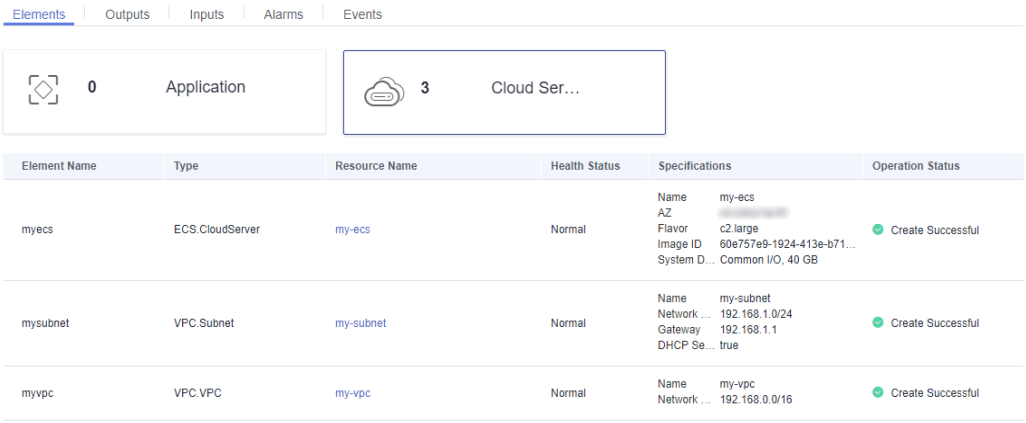
- **Stack Name**: Enter a unique stack name, for example, **aos-ecs**.
- **Description**: The description can be left blank.

Step 5 Click **Next** and check the stack information. If the stack information is correct, click **Create Stack**.

The stack details page is displayed, showing that the stack is being created. The stack includes a VPC, a subnet, and an ECS. It will take about 6 minutes to create the stack.

Step 6 Wait until the stack status becomes **Normal**. The VPC, subnet, and ECS are created and displayed in the stack element list.

Figure 2-1 Stack successfully created



Step 7 View the created cloud services.

1. Log in to the management console.
2. Choose **Service List** > **Computing** > **Elastic Cloud Server**. You will see the newly created ECS on the ECS list.
3. Choose **Service List** > **Network** > **Virtual Private Cloud**. You will see the newly created VPC on the VPC list.
4. Click the VPC name to show more details about the VPC. On the VPC details page, you will see that the subnet has been created in the VPC.

----End

Step 3: Delete Unnecessary Resources

Delete unnecessary stack resources to avoid unwanted charges.

Step 1 Log in to the AOS console.

Step 2 In the navigation pane, click **My Stacks**.

Step 3 Select the stack that will no longer be used, and click **Delete** to delete the stack.

----End

3 Stack Management

Stack management consists of two aspects. One is lifecycle management of created stacks, including deleting and changing. The other is viewing stack details to obtain stack running status.

Table 3-1 describes stack lifecycle status.

Table 3-1 Status description

Status	Description
Normal	Both the stack and its instances run properly.
Abnormal	The stack runs abnormally. Some or all stack instances run abnormally and cannot provide functions.
Initializing	Stack instances have not been installed or have been uninstalled. The stack does not provide functions.
Processing	A stack lifecycle action is being performed. The status of stack instances is unknown.
Unknown error	An unknown stack error occurs.

Changing a Stack

After a stack is created successfully (that is, in the normal status), you can change input parameters based on your service requirements.

Step 1 Log in to the Application Orchestration Service (AOS) console.

Step 2 In the navigation pane, click **My Stacks**.

Step 3 In the stack list, click the stack to be changed.

Step 4 On the stack details page, click **Change**.

Step 5 Change the template version or input parameters, and click **Next**.

Step 6 Confirm the configurations and then click **Change**.

On the **Events** tab page, view the detailed operation events related to stack change.

----End

Template change rules:

1. During template change, only the following elements are allowed to be added or deleted.
   - CCE.Addon.AutoScaler, CCE.HelmRelease, CCE.NodePool, CCE.Storage.OBS, and CCE.Storage.SFS
   - ECS.CloudServer
2. Do not modify the policies of the template.
3. Do not modify the association relationship between existing elements.
4. Do not delete the association relationship between existing elements alone. If necessary, delete both the elements and their relationship.
5. Do not add a relationship between a new element and an existing element.

Deleting a Stack

Deleted stacks cannot be restored. Exercise caution when deleting a stack.

Step 1 Log in to the AOS console.

Step 2 In the navigation pane, click **My Stacks**.

Step 3 In the stack list, select the stack to be deleted and click **Delete**.

Step 4 In the dialog box that is displayed, click **Yes**.

Check the stack name carefully. The deletion cannot be revoked.

On the **Events** tab page, view the detailed operation events related to stack deletion.

NOTE

If the stack status remains **Deleting** until a timeout message is displayed and the stack status becomes **Abnormal**, try to forcibly delete the stack.

----End

Viewing Stack Details

After a stack is created, view its data and resources on the stack details page.

- Stack elements

  The elements of a stack, such as applications and cloud services are displayed.

  Element health status:
  - Healthy: The resource is running properly.
  - Unknown: The AOS fails to obtain the resource status because an error occurs during the health check.
  - Abnormal: The AOS successfully calls the health check API of the resource, but the resource status is abnormal.

- Output parameters

  Output parameters and their values in the stack template are displayed.
- Input parameters

  Input parameters and their values in the stack template are displayed.
- Alarms

  Alarm information of the stack is displayed.
- Events

  View stack events to monitor stack operation progress. For example, when you create a stack, all important steps during the stack creation are displayed on the **Events** tab page. The events are sorted in chronological order with the latest event being displayed at the top.

4 CTS

4.1 AOS Operations Supported by CTS

Cloud Trace Service (CTS) records all operations performed on cloud services, providing data support for customers in fault locating, resource management, and security auditing. When you enable CTS, it begins to record operations performed on Application Orchestration Service (AOS) resources. CTS stores operation records from the last seven days.

Table 4-1 AOS operations supported by CTS

Operation	Description
CreateTemplate	Creating a template
DeleteTemplate	Deleting a template
UpdateTemplate	Updating a stack
PreviewStack	Previewing a stack
CreateStack	Creating a stack
DeleteStack	Deleting a stack
UpdateStack	Updating a stack
ExecuteStackAction	Executing a stack lifecycle action
CleanupResources	Cleaning a resource
UpdateTenantState	Freezing or unfreezing an account
GetBillingData	Generating billing data

4.2 Viewing Logs in CTS

When you enable CTS, operations performed on Application Orchestration Service (AOS) resources begin to be recorded. On the CTS console, you can query operation records from the last 7 days by performing the following operations.

Procedure

Step 1 Log in to the CTS console.

Step 2 In the left navigation pane, click **Trace List**.

Step 3 Filter the desired operation events.

The trace list supports four filter types:

- **Trace Source**, **Resource Type**, and **Search By**

  Select the search criteria from the drop-down lists. For example, select **AOS** from the **Trace Source** drop-down list box.

  From the **Search By** drop-down list, specify a trace name. From the **Search By** drop-down list, select or enter a specific resource ID. From the **Search By** drop-down list, select or enter a specific resource name.
- **Trace Status**: Select one of **All trace statuses**, **Normal**, **Warning**, and **Incident**.
- **Operator**: Select a specific operator (at the user level rather than the account level).
- **Start Date** and **End Date**: You can specify a time period to query traces.

Step 4 On the left of the to-be-queried record, click to view details.

Step 5 Click **View Trace** in the **Operation** column. On the displayed **View Trace** dialog box, the trace structure details are displayed.


```
{
    "service_type": "AOS",
    "user": {
        "domain": {
            "name": "***",
            "id": "6c389820d2fd46489c8987e5eb2675cc"
        },
        "id": "19652d0b0ff1407a9432b85b9e12f9eb",
        "name": "***"
    },
    "time": "2018/04/26 16:16:53 GMT+08:00",
    "code": 200,
    "resource_type": "AOS",
    "resource_name": "Stack",
    "resource_id": "19652d0b0ff1407a9432b85b9e12f9eb",
    "source_ip": "192.168.12.22",
    "trace_name": "PreviewStack",
    "trace_type": "ApiCall",
    "request": {},
    "api_version": "3.0.0",
    "message": "Preview stack successfully. Project id: 1e19d41bb1f24b5da4a98107607aac0f, stack name:
jhgdjh, template id: cea9ee29-3b39-f7be-d093-aff126b250e8, cluster id: . ",
    "record_time": "2018/04/26 16:16:53 GMT+08:00",
    "trace_id": "2da40c60-492a-11e8-a065-286ed488cbe3",
```

```
    "trace_status": "warning"
}
```

----End

5 Template Reference

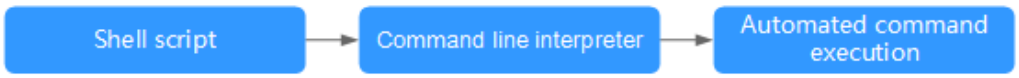
5.1 Template Introduction

5.1.1 Templates (Cloud-Based Automation Scripts)

AOS templates are text files in YAML or JSON format. They describe the cloud objects that you want, including applications, resources, and services. AOS creates various cloud objects automatically from AOS templates.

Each automated process requires a descriptive language to control its execution flow. For example, a shell script (text file) describes how to automatically run commands. Similarly, an AOS template describes the process of creating and deleting cloud objects.

The following is an example execution logic of a shell script:



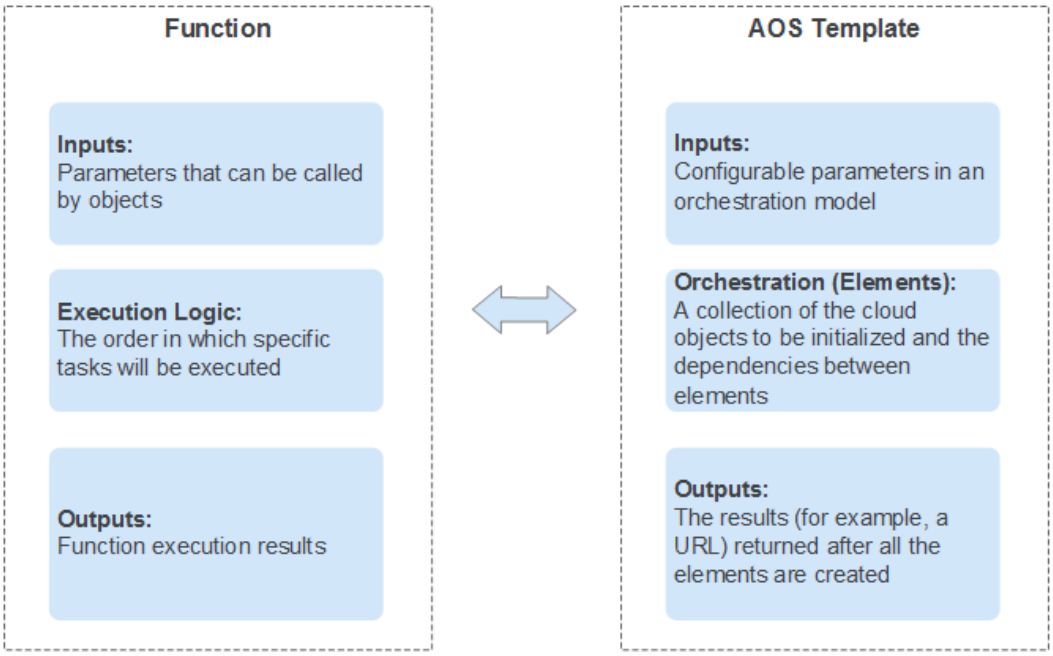
A shell script has the following features:

- A script is a text file.
- If a script is properly written, it can be reused.

An AOS template has the same execution logic as a shell script. The AOS service functions as the interpreter of AOS templates and executes actions according to templates. An AOS template can be considered as cloud automation standards.

A good shell script or function should have inputs, execution logic, and returned values. Likewise, a good template also should have **inputs**, **orchestration**, and **outputs**. A good template eases knowledge transfer and sharing.

Figure 5-1 Comparison between the function and template



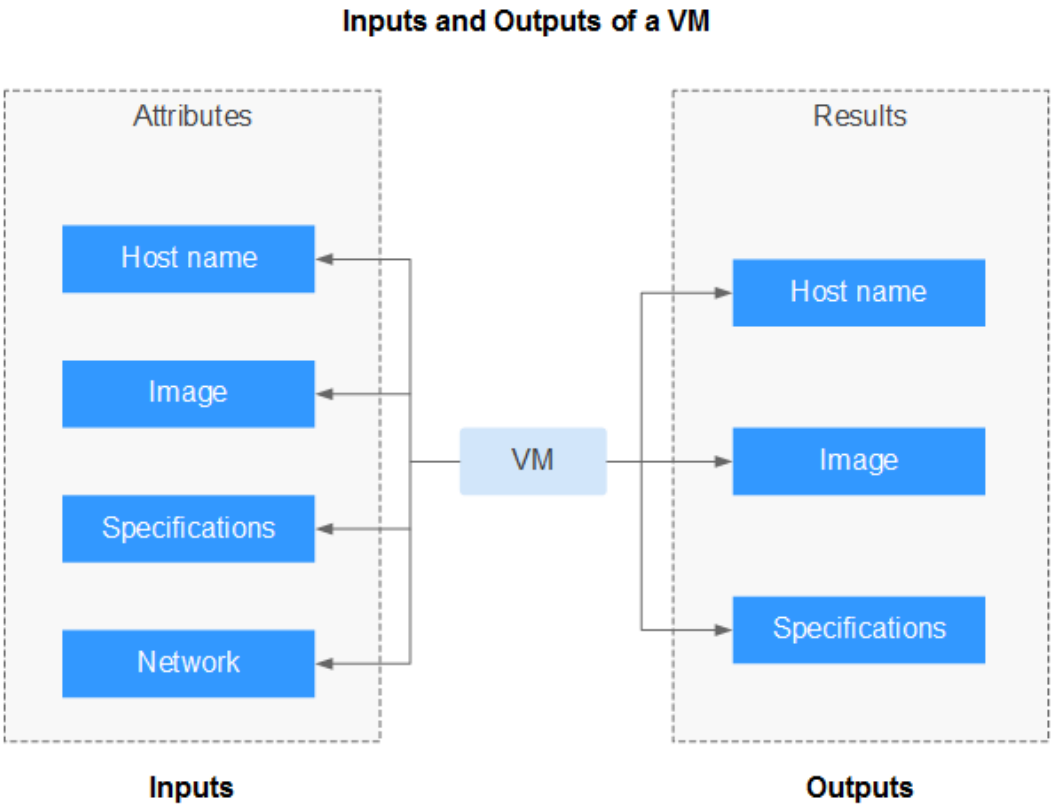
Elements (Cloud Objects)

Cloud objects can be cloud resources, services, or applications. Cloud resources are the most common cloud objects. AOS treats cloud objects as elements. A template is a collection of elements.

- Cloud resources: including resources such as Elastic Cloud Server (ECS) and Virtual Private Cloud (VPC).
- Cloud services: including services such as Distributed Cache Service (DCS).
- Cloud applications: including applications such as containerized applications in Cloud Container Engine (CCE).

You need to set inputs to create any cloud object. After a cloud object is created, a result is displayed. The following figure uses an ECS (VM) as an example.

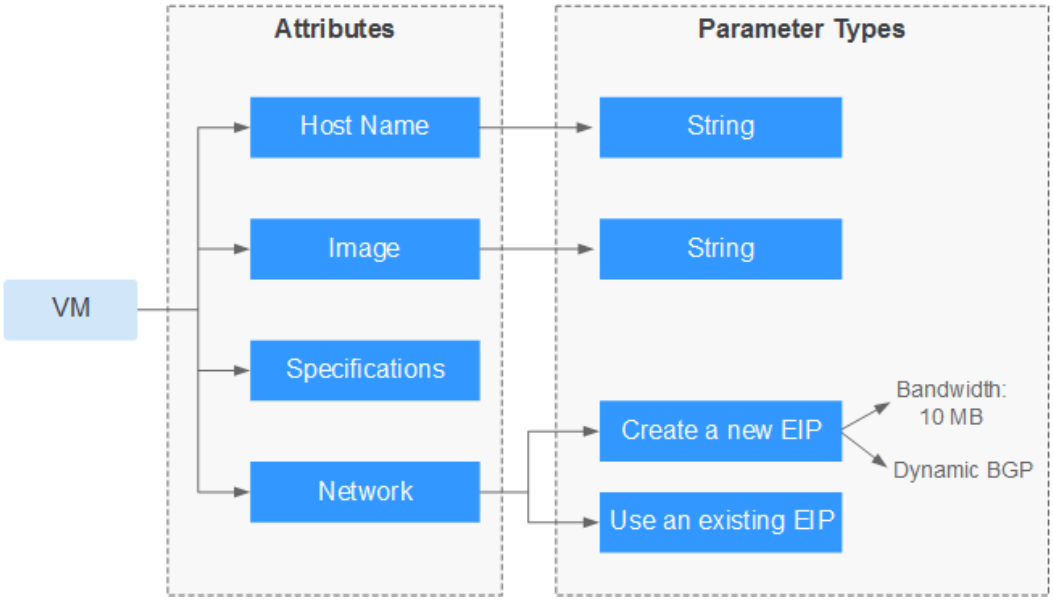
Figure 5-2 Inputs and outputs of a cloud object



Inputs (Properties)

Inputs are requirements or parameters involved during the creation of a cloud object. The parameters required by a cloud object are determined by the characteristics of the object. Some objects require many parameters, for example, VMs. Some objects can be created with a few parameters or without parameters, for example, Object Storage Service (OBS) buckets. Some input parameters are complex and consist of multiple basic parameters, for example, network attributes of VMs.

Figure 5-3 Inputs



The syntax is as follows:

```
Cloud object (element):
  description: description of the cloud object
  properties: # Parameters of the cloud object
    Property 1:  # Parameter 1
    Property 2: # Parameter 2
    Property....:  # Parameter...
```

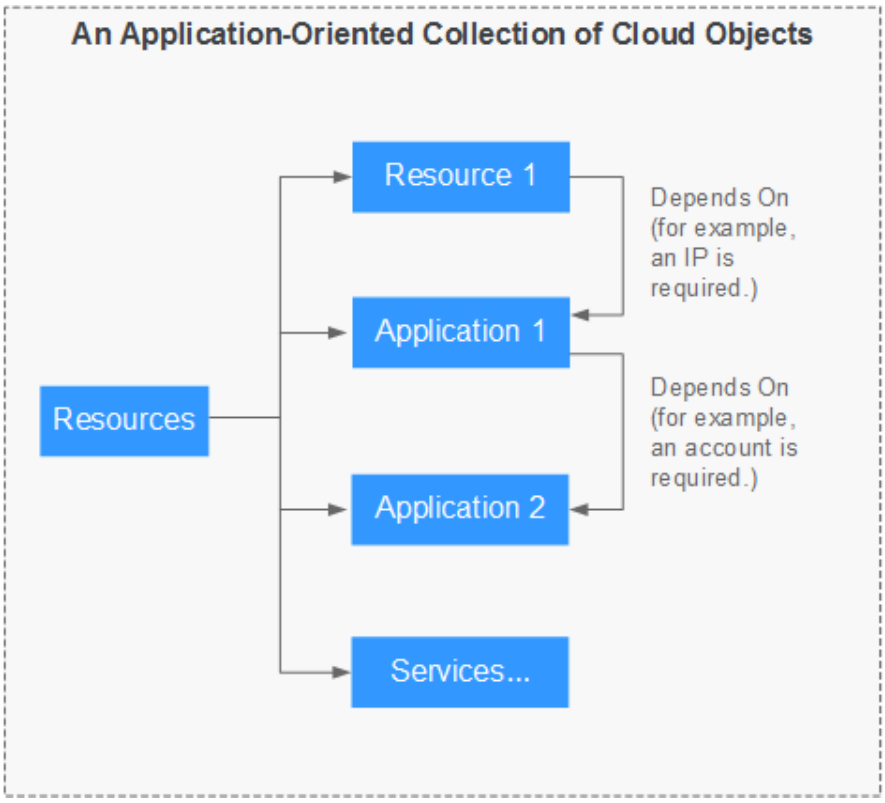
Orchestration (Elements)

If elements are initialized just one by one according to the order in which they are arranged, no orchestration is required. AOS supports orchestration of elements with complex dependencies between them. The initialization (input) of an element depends on the result (output) of another element. Such a relationship can be specified by using an AOS template.

In an AOS template, you can specify the output of any element as the input of another element. The initialization process can be controlled freely, which is called orchestration. Only orchestration can meet various automation requirements.

An AOS template is the collection of objects that you want to orchestrate. To be more specific, an AOS template is a collection of objects that you want to control during the initialization process.

Figure 5-4 Orchestration



The relationship between elements is classified into two types: dependency and inclusion.

- Dependency: The input of an element depends on the output of another element. If element A depends on element B, element A can be created only after element B is successfully created.
- Inclusion: An element is a part of another element. If element A contains element B, element B can be created only after element A is successfully created.

Outputs (Return Values)

Outputs are the results returned after a cloud object is successfully created. The returned results of a cloud object are determined by the characteristics of the object. Some objects have many results, and some objects have few results.

The output of a cloud object is used in the following two scenarios. Generally, it is used together with the **get_attribute** built-in function.

- The output is used as an input of another cloud object.
- The output is used as the result of the entire stack.

The syntax is as follows:

```
# Result of another ECS. The service name is Service.
value: {get_attribute: [ecs, Service, ports, 0, nodePort]}
```

5.1.2 Template Structure

Sample Template

```
# Version of the application template
tosca_definitions_version: cloud_tosca_version_1_0
# Description of the application template
description: template for deploying an ECS
# Definitions of input parameters
inputs:
  instance:
    default: 1
    description: number of ECSs to be created
  subnet:
    description: ID of the subnet to which the ECS belongs
  vpc:
    description: ID of the VPC to which the ECS belongs
mappings:
  regionMap:
    ae-ad-1:
      flavor: c2.medium
      image_id: f2003c7b-99c4-4616-be19-334beaca81b1
# Definitions of element objects
node_templates:
  myecs:
    type: Cloud.ECS.CloudServer
    properties:
      availabilityZone: ae-ad-1a
      flavor:
        get_in_map:
          - regionMap
          - get_input: Cloud.Region
          - flavor
      imageId:
        get_in_map:
          - regionMap
          - get_input: Cloud.Region
          - image_id
      instances:
        get_input: instance
      name: my-ecs
      nics:
        - subnetId:
            get_input: subnet
      publicIP:
        eip:
          bandwidth:
            shareType: PER
            size: 1
          ipType: 5_sbgp
      rootVolume:
        size: 40
        volumeType: SATA
      vpcId:
        get_input: vpc
# Definitions of output parameters
outputs:
  ecs-eip:
    description: elastic IP address of the ECS
    value:
      get_attribute:
        - myecs
        - publicIps
  south-flavor:
    description: specifications of the VM to be created
    value:
      get_in_map:
        - regionMap
```

```
    - ae-ad-1
    - flavor
```

Template Composition

An AOS template consists of the following sections:

1. **tosca_definitions_version**: (mandatory) specifies the version of the template.

   NOTE

   Currently, only **cloud_tosca_version_1_0** is supported by AOS.

2. **node_templates**: (mandatory) defines the set of objects, which are all elements, to be orchestrated in a template. For more information, see **node_templates**.
3. **description**: (optional) describes the template. The maximum length is 1,024 characters.
4. **inputs**: (optional) defines the input parameters used during stack creation. For more information, see **inputs**.
5. **outputs**: (optional) defines the output parameters during stack running. For more information, see **outputs**.
6. **mappings**: (optional) defines a mapping table. For more information, see **mappings**.
7. **conditions**: (optional) defines conditions. For more information, see **conditions**.

5.1.3 node_templates

The **node_templates** section is mandatory. It defines the set of objects, which are all elements, to be orchestrated in a template. An element can be an application or a cloud service resource.

Format of the **node_templates** section:

```
<Element name>:
  type: <Element type>
  properties: <Element properties>
  requirements: <Element dependency>
  condition: <Condition name>
```

Table 5-1 Parameter property description

Property	Mandatory	Type	Value Constraint	Description
Element name	Yes	String	Enter 1 to 48 characters. Only lowercase letters, digits, and hyphens (-) are allowed.	Each element name must be unique.

Property	Mandatory	Type	Value Constraint	Description
Element type	Yes	Cloud.*** (*** indicates an element names in **Resource Indexes**)	-	This parameter is used to specify the type of an orchestration object.
Element property	No	-	Property information is expanded based on element types. Each element type has its properties. For more information, see **Resource Indexes**.	The variable of a property can be obtained from the **inputs** section or by using the **get_attribute** function. If an element does not require a special property, you do not need to define **properties**.

Element dependency	No	-	This parameter is used to specify the name of another element that has a dependency with the current element.	If there is no relationship between elements, you do not need to define this parameter. The dependency between elements is based on the defined element type. Related dependencies can be defined for specific types.
				NOTE
				For example, when a subnet depends on a VPC, define the VPC as a dependent node in the requirements of the subnet.
				`requirements:`
				`- vpcId:`
				`node: myvpc`

Condition name	No	String	Enter 1 to 64 characters. Only letters, digits, and hyphens (-) are allowed.	If a condition is defined, the element is deployed only when the condition is met. For more information, see **conditions**.

Sample **node_templates**:

```
# Definitions of element objects
node_templates:
  myecs:
    type: Cloud.ECS.CloudServer
    properties:
      availabilityZone: ae-ad-1a
      flavor: c1.medium
      imageId: a3934478-bfeb-4a02-b257-9089779f0380
      instances: 1
      name: my-ecs
      nics:
        - subnetId:
            get_input: subnet
      rootVolume:
        size: 40
        volumeType: SATA
      vpcId:
        get_input: vpc
```

5.1.4 inputs

To enable a template to be more commonly used, do not set all parameter values of the elements to fixed values. For example, it is advised to set the image ID of a VM as an input of a template. In this case, you can select different images every time you use the template to create a stack.

The **inputs** section is optional and defines the inputs of a stack created based on a template. A maximum of 60 input parameters can be defined in a template. Each input parameter must have a unique name so that the value can be obtained by using the **get_input** built-in function. If an input parameter is defined repeatedly, the latest definition will overwrite the previous one.

Function scope: **node_templates** and **outputs** sections. That is, input parameters can be transferred in the properties of **node_templates** and values of **outputs**.

Format of the **inputs** section:

```
<Input parameter name>:
  type: <Type>
  default: <Default value>
  constraints: <Constraints>
  description: <Description>
  label: <Label>
  invisible: <Whether command outputs are visible>
```

In addition to the reusability of a template, methods of restricting and verifying user inputs also need to be considered during template input design. Template designers must understand parameter statements.

Table 5-2 Parameter property description

Property	Mandatory	Type	Value Constraint	Description
Input parameter name	Yes	String	The value must be 1 to 20 characters long. Only lowercase letters, digits, and hyphens (-) are allowed.	A maximum of 60 input parameter names can be defined and each name must be unique.

Property	Mandatory	Type	Value Constraint	Description
type	Yes	• **string**: character string • **integer**: number • **float**: floating-point number • **boolean**: boolean value • **password**: password	When the **type** is set to **password**, no output is visible. Currently, only the passwords entered at the system level can be decrypted. If a common parameter is defined as a password, encrypted information may be obtained and such information fails to be decrypted.	Parameter type.
description	No	String	The value must be 0 to 255 characters long.	Parameter description information.
default	No	String	When creating a stack, you can enter a value to replace the default value. If no default value is set, you must enter the value of this parameter. **NOTICE** The default value type must be the same as the defined parameter type. If they are inconsistent, the parser may perform automatic conversion, resulting in an unexpected result.	Default parameter value.
label	No	String	The value is a string of 0 to 64 characters.	Label of a parameter. The label defined here can be displayed by category during stack creation.

Property	Mandatory	Type	Value Constraint	Description
constraints	No	String	There are several constraints. You can define only one rule for each condition of an input parameter. If any of the constraints is not met, the parameter is considered invalid. • **equal**: The value of this parameter must be equal to the specified value. For example, if the value of an input parameter is not **aos**, the value is regarded as invalid. **constraints:** **equal: 'aos'** • **valid_values**: valid value range. This parameter is used to define an array. For example, the value of an output parameter can be **TCP** or **UDP**. **constraints:** **valid_values: ['TCP', 'UDP']** • **regex**: The parameter must meet a certain regular expression and must be of the string type. For example, if the input parameter does not meet the regular expression, the parameter is regarded as invalid. **constraints:** **regex: "^[-_a-zA-Z0-9]*$"** • **invalid_values**: invalid value range. If you set a parameter to a value which is within the	Parameter constraints, which are used to restrict the valid value range of an input parameter.

Property	Mandatory	Type	Value Constraint	Description
			invalid value range, such a value is regarded as invalid and an error is reported. For example, if the value of an input parameter is set to **1** or **12**, the value is regarded as invalid. **constraints:** **invalid_values: ['1', '12']**	
invisible	No	-	When **invisible** of an input parameter is set to **true**, ****** is displayed.	Whether the output is visible.

Example configuration of **inputs**:

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs: # defines the variables of a stack created based on a template.
  instance:
    description: number of ECSs to be created
    default: 1
  image:
    description: ID of the image used by the ECS
    type: Cloud.ECS.Image.Id
  vpc:
    description: ID of the VPC to which the ECS belongs
  subnet:
    description: ID of the subnet to which the ECS belongs
```

5.1.5 outputs

After you create a stack using a template, all objects defined in the template will be created. To view the deployment results more intuitively, write the results in the output section of the template. Generally, common outputs include the access address+port number, application URL, and initial account password.

The **outputs** section is optional and defines the output parameters during stack running. Each output parameter must have a unique name.

Format of the **outputs** section:

```
<Output parameter name>:
  description: <Description>
  value: <Value>
```

Table 5-3 Parameter property description

Property	Mandatory	Type	Value Constraint	Description
Output parameter name	Yes	String	The value must be 1 to 20 characters long. Only lowercase letters, digits, and hyphens (-) are allowed.	Name of an output parameter, which must be unique.
Description	No	Text string	Text string, supporting a maximum of 255 characters	Name of a mapping object, which must be unique.
value	Yes	-	-	**value** is used to define an output value. It can be a text, string, or number. The value can be concatenated by the **concat** and **get_attribute** built-in functions, or be obtained from input parameters. **NOTE** A parameter that begins with a hyphen (-) can be considered as an array.

Example configuration of **outputs**:

```
outputs:
  ecs-eip:
    description: elastic IP address of the ECS
    value:
      get_attribute:
        - myecs
        - publicIps
```

5.1.6 mappings

The **mappings** section is optional and defines a mapping table. When creating a stack based on a template, you can use the **get_in_map** function to extract the content corresponding to a specific variable. A maximum of 10 mappings can be defined in a template.

Format of the **mappings** section:

```
<Mapping name>:
  <Mapping object name>:
```

```
    <Mapping object property name>: <Mapping object property value>
    <Mapping object property name>: <Mapping object property value>
    ...
  ...
...
```

Table 5-4 Parameter property description

Property	Mandatory	Type	Value Constraint	Description
Mapping name	Yes	String	The value must be 1 to 64 characters long. Only letters, digits, and hyphens (-) are allowed.	A maximum of 10 mapping names can be defined and each name must be unique.
Mapping object name	Yes	String	The value must be 1 to 64 characters long. Only letters, digits, and hyphens (-) are allowed.	Name of a mapping object, which must be unique.
Mapping object property name	Yes	String	The value must be 1 to 64 characters long. Only letters, digits, and hyphens (-) are allowed.	Property name of a mapping object. Each name must be unique in the same mapping object.
Mapping object property value	Yes	String or digit	String or digit	Property value corresponding to a mapping object.

Example configuration of **mappings**:

```
mappings:
  imageMap:
    old:
      image: '192.168.1.86:20202/test/mysql-server:5.6.35'
    new:
      image: '192.168.1.90:20202/test/mysql-server:5.7.1'
```

Usage mode of **mappings**:

The defined mappings can be used in **node_templates** or **outputs**.

- Use the **get_in_map** function to extract the mapping content from **node_templates**.

  For example, the **myecs** object is defined in **node_templates**, and its properties include the image ID and VM specifications. The image ID and VM

specifications must have been predefined in **mappings**. During stack creation based on the template, the required image and VM specifications of the corresponding region will be used.

```
node_templates:
  myecs:
    type: Cloud.ECS.CloudServer
    properties:
      availabilityZone: ae-ad-1a
      flavor:
        get_in_map:
          - regionMap
          - get_input: Cloud.Region
          - flavor
      imageId:
        get_in_map:
          - regionMap
          - get_input: Cloud.Region
          - image_id
      ...
```

- Use the **get_in_map** function to extract the mapping content from **outputs**.

```
outputs:
  south-flavor:
    description: specifications of the VM to be created
    value:
      get_in_map:
        - regionMap
        - ae-ad-1
        - flavor
```

5.1.7 conditions

The **conditions** section is optional and defines conditions. By specifying conditions, you can determine whether to create and deploy elements defined in **node_templates**.

Format of the **conditions** section:

```
<Condition name>:
   <Built-in condition function>
...
```

The following shows how to specify conditions to control the effectiveness of properties in **node_templates**:

```
node_templates:
 <Element name>:
   condition: <Condition name>
...
```

Table 5-5 Parameter property description

Property	Mandatory	Type	Value Constraint	Description
Condition name	Yes	String	The value must be 1 to 64 characters long. Only letters, digits, and hyphens (-) are allowed.	Name of the condition, which must be unique.

Property	Mandatory	Type	Value Constraint	Description
Built-in function	Yes	-	-	Built-in condition functions are used to define conditions. For details, see **Condition Function**.
Element name	Yes	String	The value must be 1 to 48 characters long. Only lowercase letters, digits, and hyphens (-) are allowed.	Name of the element, which must be unique.
Condition name	Yes	String	The value must be 1 to 64 characters long. Only letters, digits, and hyphens (-) are allowed.	Condition name defined in **conditions**.

Example configuration of **conditions**:

When specifying conditions to determine whether to create and deploy elements, you need to define reference relationships in multiple sections such as **inputs**, **conditions**, and **node_templates**.

```
tosca_definitions_version: cloud_tosca_version_1_0
conditions:
  condition_vm_deploy:    #The conditions can be met only when inputs parameters are matched.
    cond_eq:
      - get_input: vm_deploy
      - true
inputs:
  image:
    description: ID of the image used by the ECS
    type: Cloud.ECS.Image.Id
  instance:
    default: 1
    description: number of ECSs to be created
  subnet:
    description: ID of the subnet to which the ECS belongs
..vm_deploy:    #Determines whether to deploy the VM.
    default: true
    type: boolean
  vpc:
    description: ID of the VPC to which the ECS belongs
node_templates:
  vm:
    condition: condition_vm_deploy     # The VM will be deployed only when the conditions are met.
    type: Cloud.ECS.CloudServer
    properties:
      availabilityZone: ae-ad-1a
      imageId:
        get_input: image
      flavor: s3.small.1
      instances:
```

```
      get_input: instance
    name: my-ecs
    nics:
      - subnetId:
          get_input: subnet
    rootVolume:
      size: 40
      volumeType: SATA
    vpcId:
      get_input: vpc
myecs:
  type: Cloud.ECS.CloudServer
  properties:
    name: my-ecs
    instances:
      get_input: instance
    imageId:
      get_input: image
    flavor: s3.small.1
    vpcId:
      get_input: vpc
    availabilityZone: ae-ad-1a
    nics:
      - subnetId:
          get_input: subnet
    rootVolume:
      volumeType: SSD
      size: 40
```

5.1.8 Template Compilation Skills

Waiting for Component Start-up

Assume that the "A" (application) and "S" (service) components need to be started, "A" depends on "S", and "A" needs to connect to "S" to provide services. In the following example, "A" is Tomcat and "S" is MySQL.

During Application Orchestration Service (AOS) orchestration, "S" is first started based on the template. After "S" is started successfully (its process is started successfully, but its service function is still unavailable), "A" is then started. If "A" is connected to "S" before the "S" service function is completely started, "A" fails to be started. As a result, the entire stack fails to be started. Therefore, you may need to wait for a period of time before starting "A".

Currently, the waiting logic is not supported in the template syntax. To solve the problem, add the waiting logic to the service process.

The following is an example of waiting for a period of time before starting a component:

```
name:  # Parameter name
  type: string  # Parameter type
  description: resource name  # Parameter description
Task-Name: # Task name (user defined)
  description: sleep before business
  actions:
    poststart: # Execute scripts before startup.
      command: "/bin/sh, -c, sleep
```

Converting Numbers into Strings

In many cases, variables are defined as strings, but they sometimes need to be referenced as numbers. For example, when the port number is used as an

environment variable, the value must be a string. When the port number is used as a microservice attribute, the value must be a number.

To solve the preceding problem, use either of the following methods:

- Method 1: Define two variables.

  Define the **PORT-i** and **PORT-s** variables. **PORT-s** is a string, while **PORT-i** is a number. This method can directly be used to solve the preceding problem, but the effect is not ideal. Due to duplication, the maintainability and usability of the template deteriorate.

- Method 2: Use the **concat** built-in function.

  Use the **concat** built-in function to combine multiple small strings into a longer and more complete string. The parameters of the **concat** built-in function can be any type of variable, supporting the combination of numbers and strings. Example command:

  First, define variables as follows:

  ```
  magento-EPORT:
    type: integer
    default: 32080
  ```

  When the parameter indicates a ULR, ensure that its value is a string:

  ```
  name: MAGENTO_URL
    value:
      concat:
      - "http://"
      - {get_input: magento-EIP}
      - ":"
      - {get_input: magento-EPORT}  #Convert a number to a string.
  ```

  When the parameter indicates a microservice attribute, ensure that its value is a number:

  ```
  serviceSpec:
    ports:
      - port: {get_input: magento-container-port}
        nodePort: {get_input: magento-EPORT}  #The value must be a number.
  ```

5.1.9 Built-In Functions

5.1.9.1 Variable Reference

During template compilation, you can reference a defined variable or reference a member variable of another object, just like the variable reference during function compilation. You can also reference other existing values in an AOS template.

To ease template compilation, different reference methods are used based on the reference objects:

- To reference input parameters, use **get_input**.
- To reference element properties, use **get_attribute** or **get_reference**.
- To reference mapping tables, use **get_in_map**.

The preceding reference methods are also called built-in functions. In addition to reference functions, built-in functions also include many other functions. For more information, see **Table 5-6**.

Table 5-6 AOS built-in functions

Built-In Function	Description
get_input	Used to obtain the values of input parameters in the **inputs** section of the template file.
get_attribute	Used to obtain the initialization results of other elements in the template.
get_reference	Simplified form of the **get_attribute** function. When the attribute information ends with **id** or **name**, use the **get_attribute** (**refID** or **refName**) function.
get_in_map	Used to obtain the content in mapping tables.
Condition function	Used to define whether elements need to be deployed, including **cond_eq**, **cond_not**, **cond_and**, **cond_or**, and **cond_if**.
base64_encode	Used to encode character strings in base64 mode.
concat	Used to convert description fields into strings and concatenate them. It can be embedded with the **get_attribute** and **get_input** functions.
split	Used together with the **select/get_list_length** function in most cases. The **split** function is mainly used in the following scenarios: • A string is split into a group of strings so that specific elements can be easily obtained from the result string list. • A result string array is directly used.
select	Used to obtain the object with a specified subscript from an array structure. Generally, this function is used together with the **split** function.
get_list_length	Used to calculate the number of elements in an array structure. Generally, this function is used together with the **split** function.

5.1.9.2 get_input

The **get_input** function is used to obtain the value of the input parameter defined in the **inputs** section of the template file. In addition, it can be used to reference system pseudo parameters. For details, see **System Pseudo Parameters**.

Syntax

```
get_input: [paramName ]
```

Parameter Description

Table 5-7 Parameter description

Parameter	Mandatory	Description
paramName	Yes	Name of the input parameter defined in the **inputs** section of the template file.

Return Value

Value of the parameter

Examples

The following shows how to use the **get_input** function to retrieve the value of a parameter in the **inputs** section:

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  name:
    default: test-vpc
  cidr:
    default: 10.0.0.0/8
node_templates:
  my-first-vpc:
    type: Cloud.VPC.VPC
    properties:
      name: {get_input: name}
      cidr: {get_input: cidr}
```

System Pseudo Parameters

In addition to the parameters defined in the template, the **get_input** function can also reference system pseudo parameters. Currently, the following system pseudo parameters are supported:

- **Cloud.UserId**: obtaining the user ID of the current stack creator.
- **Cloud.ProjectId**: obtaining the ID of the project to which the current stack belongs.
- **Cloud.DomainId**: obtaining the ID of the account to which the current stack belongs.
- **Cloud.Region**: obtaining the ID of the region where the current stack resides.
- **Cloud.StackName**: obtaining the name of the current stack.

System pseudo parameters can be used together with the **mappings** and **get_in_map** functions to obtain predefined configuration information.

For example, an ECS VM can be deployed in different regions. You can predefine images and VM specifications for different regions in the mapping table. During stack creation, you can run **{get_input: Cloud.Region}** to obtain the region where the current stack resides and obtain configuration information such as images and specifications from the mapping table.

```
mappings:
  regionMap:
    ae-ad-1:
      flavor: c2.medium
      image_id: f2003c7b-99c4-4616-be19-334beaca81b1
node_templates:
  myecs:
    type: Cloud.ECS.CloudServer
    properties:
      availabilityZone: ae-ad-1a
      flavor:
        get_in_map:
          - regionMap
          - get_input: Cloud.Region
          - flavor
      imageId:
        get_in_map:
          - regionMap
          - get_input: Cloud.Region
          - image_id
      ...
```

5.1.9.3 get_attribute

The **get_attribute** function is used to obtain the initialization results of other elements in a template.

Syntax

```
get_attribute: [resourceName, attributeName ]
```

If the content corresponding to **attributeName** is a structure body and contains multiple **key-value** fields, you can extend the definition. The format is as follows:

```
get_attribute: [resourceName, attributeName1, attributeName2, ...]
```

Parameter Description

Table 5-8 Parameter description

Parameter	Mandatory	Description
resourceName	Yes	Name of a resource customized in the template.
attributeName	Yes	Attribute name of the desired resource. For details about the attribute name, see the **outputs** section of the element object. If the attribute name defined in the template does not exist, no information is returned. Currently, for most elements, only their **refID** and **refName** can be obtained. • **refID**: unique ID generated after a resource is created. • **refName**: resource name.

Return Value

Attribute value obtained

- When a single resource is created, the return values of **refID** and **refName** are strings.
- When multiple resources are created (for example, multiple ECS VMs are created at a time), the return values of **refID** and **refName** are string arrays.

Examples

- Obtaining parameters and assign values to the parameters in the **outputs** section.

  Example: Obtaining the ID of the created **my-first-vpc**, and then assign it to the **vpc_id** output parameter of a stack.

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  name:
    default: test-vpc
  cidr:
    default: 10.0.0.0/8
node_templates:
  my-first-vpc:
    type: Cloud.VPC.VPC
    properties:
      name: {get_input: name}
      ciddr: {get_input: cidr}
outputs:
  vpc_id:
    value: {get_attribute: [my-first-vpc,refID]}
```

- Obtaining parameters and use them as input parameters for creating other resources.

  Example: Obtaining the ID of the created **my-second-vpc** and assign a value to the subnet resource as the input for creating the subnet resource. In this way, multiple resources can be created in one **blueprint** file.

```
node_templates:
  my-subnet:
    type: Cloud.VPC.Subnet
    properties:
      name: {get_input: subnet-name}
      cidr: {get_input: vpc-cidr}
      gateway: {get_input: subnet-gateway}
      dnsList: {get_input: dnsList}
      vpc: {get_attribute: [my-second-vpc,refID]}
      availabilityZone: {get_input: az}
    requirements:
      - vpcId:
          node: my-vpc
  my-second-vpc:
    type: Cloud.VPC.VPC
    properties:
      name: {get_input: vpc-name}
      cidr: {get_input: vpc-cidr}
```

5.1.9.4 get_reference

The **get_reference** function is the simplified form of the **get_attribute** function. When the attribute information ends with **id** or **name**, use the **get_attribute** (**refID** or **refName**) function.

Syntax

```
get_reference: [elementName ]
```

Parameters

Table 5-9 Parameters

Parameter	Mandatory	Description
elementName	Yes	Element name defined in the **node_templates** section of the **blueprint** file.

Return Value

Value of the parameter

Examples

The following describes how to use the **get_reference** function to obtain the dynamic attributes of associated elements:

```
node_templates:
  my-first-vpc:
    type: Cloud.VPC.VPC
    properties:
      name: {get_input: name}
      ciddr: {get_input: cidr}
  my-first-subnet:
    type: Cloud.VPC.VPC
    properties:
      vpcId: {get_reference: my-first-vpc } # Corresponding to {get_attribute: [my-first-vpc, refID] }
      ...
```

5.1.9.5 get_in_map

If a mapping table is defined in the template, you can use the **get_in_map** function to obtain the mapping table content from the **node_templates** and **outputs** sections.

Syntax

```
get_in_map: [map_name, top_level_key, second_level_key]
```

Parameter Description

Table 5-10 Parameter description

Parameter	Mandatory	Description
map_name	Yes	Mapping name
top_level_key	Yes	Mapping object name
second_level_key	Yes	Mapping object property

Return Value

Value of the corresponding field in the mapping table.

Examples

Obtaining the mapped content using the **get_in_map** function:

```
mappings:
  regionMap:
    ae-ad-1:
      flavor: c2.medium
      image_id: f2003c7b-99c4-4616-be19-334beaca81b1
node_templates:
  myecs:
    type: Cloud.ECS.CloudServer
    properties:
      availabilityZone: ae-ad-1a
      flavor:
        get_in_map:
          - regionMap
          - get_input: Cloud.Region
          - flavor
      imageId:
        get_in_map:
          - regionMap
          - get_input: Cloud.Region
          - image_id
      ...
```

5.1.9.6 Condition Function

Condition functions are usually used to define whether elements need to be deployed, including **cond_eq**, **cond_not**, **cond_and**, **cond_or**, and **cond_if**. Except **cond_if**, other condition functions can only be used in the **conditions** section. The **cond_if** function can be used in the **conditions**, **node_templates**, and **outputs** sections.

For example, **vm_deploy** is used to determine whether to deploy a VM.

```
tosca_definitions_version: cloud_tosca_version_1_0
conditions:
  condition_vm_deploy:    #The conditions can be met only when inputs parameters are matched.
    cond_eq:
      - get_input: vm_deploy
```

```
    - true
inputs:
  image:
    description: ID of the image used by the ECS
    type: Cloud.ECS.Image.Id
  instance:
    default: 1
    description: number of ECSs to be created
  subnet:
    description: ID of the subnet to which the ECS belongs
..vm_deploy:    #Determines whether to deploy the VM.
    default: true
    type: boolean
  vpc:
    description: ID of the VPC to which the ECS belongs
node_templates:
  vm:
    condition: condition_vm_deploy     # The VM will be deployed only when the conditions are met.
    type: Cloud.ECS.CloudServer
    properties:
      availabilityZone: ae-ad-1a
      imageId:
        get_input: image
      flavor: s3.small.1
      instances:
        get_input: instance
      name: my-ecs
      nics:
        - subnetId:
            get_input: subnet
      rootVolume:
        size: 40
        volumeType: SATA
      vpcId:
        get_input: vpc
  myecs:
    type: Cloud.ECS.CloudServer
    properties:
      name: my-ecs
      instances:
        get_input: instance
      imageId:
        get_input: image
      flavor: s3.small.1
      vpcId:
        get_input: vpc
      availabilityZone: ae-ad-1a
      nics:
        - subnetId:
            get_input: subnet
      rootVolume:
        volumeType: SSD
        size: 40
```

cond_eq

The **cond_eq** function is used to determine whether an equal condition is met. It is generally used to determine whether an input parameter is consistent with an expected value.

Table 5-11 cond_eq

Syntax	Parameter Description	Return Value
cond_eq: [cond1, cond2]	• **cond1**: Condition 1, which can be a number, string, Boolean value, or variable obtained using the **get_input** function. • **cond2**: Condition 2, which can be a number, string, Boolean value, or variable obtained using the **get_input** function.	When the value of **cond1** is the same as that of **cond2**, **true** is returned; otherwise, **false** is returned.

The following describes how to use the **cond_eq** function to determine whether the input parameter is consistent with an expected value:

```
inputs:
  a:
    type: string
    default: 10
conditions:
  matchA:
    cond_eq: [{get_input: a}, 10]
```

cond_not

The **cond_not** function is used to reverse the calculation result and is usually nested with other condition functions.

Table 5-12 cond_not

Syntax	Parameter Description	Return Value
cond_not: cond	• **cond**: Condition expression, which can be a Boolean value, Boolean variable obtained using the **get_input** function, or nested condition function such as **cond_eq** or **cond_not**.	If the calculation result of the condition expression is **true**, **false** is returned. If the result is **false**, **true** is returned.

The following describes how to use the **cond_not** function to determine whether the input parameter is consistent with an expected value:

```
inputs:
  a:
    type: boolean
    default: true
conditions:
  matchA:
    cond_not: {get_input: a}
```

cond_and

The **cond_and** function is used to check whether multiple conditions are met. This function supports 2 to 10 conditions.

Table 5-13 cond_and

Syntax	Parameter Description	Return Value
cond_and: [cond1, cond2...condn]	• **cond1**: Condition 1, which can be a Boolean value, Boolean variable obtained using the **get_input** function, or nested condition function such as **cond_eq** or **cond_not**. • **cond2**: Condition 2, which can be a Boolean value, Boolean variable obtained using the **get_input** function, or nested condition function such as **cond_eq** or **cond_not**. • **condn**: Condition n ($3 \leq n \leq 10$), which is optional and can be defined as required. The parameter type is the same as that of **cond1** or **cond2**.	If all parameter conditions are met, **true** is returned; otherwise, **false** is returned.

The following describes how to use the **cond_and** function to check whether the combination conditions are met:

```
inputs:
  a:
    type: integer
    default: 10
  b:
    type: string
    default: debug
conditions:
  matchAnd:
    cond_and: [{cond_eq: [{get_input: a}, 10]}, {cond_eq: [{get_input: b}, debug]}]   # The condition of matchAnd can be met only when both conditions 1 and 2 are met.
```

cond_or

The **cond_or** function is used to determine whether any of multiple conditions is met. This function supports 2 to 10 conditions.

Table 5-14 cond_or

Syntax	Parameter Description	Return Value
cond_or: [cond1, cond2...condn]	• **cond1**: Condition 1, which can be a Boolean value, Boolean variable obtained using the **get_input** function, or nested condition function such as **cond_eq** or **cond_not**. • **cond2**: Condition 2, which can be a Boolean value, Boolean variable obtained using the **get_input** function, or nested condition function such as **cond_eq** or **cond_not**. • **condn**: Condition n (3 ≤ n ≤ 10), which is optional and can be defined as required. The parameter type is the same as that of **cond1** or **cond2**.	If any condition is met, **true** is returned. If no condition is met, **false** is returned.

The following describes how to use the **cond_or** function to check whether the combination conditions are met:

```
inputs:
  a:
    type: integer
    default: 10
  b:
    type: string
    default: debug
conditions:
  matchOr:
    cond_or: [{cond_eq: [{get_input: a}, 8]}, {cond_eq: [{get_input: b}, debug]}]  # The condition of matchOr can be met when either condition is met.
```

cond_if

The **cond_if** function is a triplet expression used to assign values to properties. It is generally used in the property structure of **node_templates**.

Table 5-15 cond_if

Syntax	Parameter Description	Return Value
cond_if: [condition, value_true, value_false]	• **condition**: Condition name, which must be defined in the **conditions** section. • **value_true**: Value assigned when a condition is met. • **value_false**: Value assigned when a condition is not met.	If the condition is met, **value_true** is returned. If the condition is not met, **value_false** is returned.

The following describes how to use the **cond_if** function to define property values:

```
inputs:
  a:
    type: integer
    default: 10
  b:
    type: string
    default: debug
conditions:
  matchOr:
    cond_or: [{cond_eq: [{get_input: a}, 8]}, {cond_eq: [{get_input: b}, debug]}]  # The condition of matchOr can be met when either condition is met.
node_templates:
  vm:
    type: Cloud.ECS.CloudServer
    properties:
      vpcId: vpc-id-123
      name: myvm
      nics:
        - subnetId: subnet-id-123
      imageId: {cond_if: [matchOr, image-debug, image-product]}   # cond_if is used to define a condition. If the debugging mode is used, debugging images are used; otherwise, product images are used.
      instances: 1
      availabilityZone: ae-ad-1a
      rootVolume:
        volumeType: SATA
        size: 40
      flavor: flavor-1
```

5.1.9.7 base64_encode

The **base64_encode** function is used to encode character strings in base64 mode.

Syntax

```
base64_encode: param
```

Parameter Description

Table 5-16 Parameter description

Parameter	Description
param	Character string to be encoded.

Return Value

Base64-encoded result.

Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  ecs_availabilityZone:
    description: AZ to which the ECS belongs
    label: ''
  ecs_flavor:
    description: ECS specifications
    label: ''
  ecs_imageId:
    description: ID of the image used by the ECS
    label: ''
  ecs_nics_0_subnetId:
    description: NIC information about the ECS to be created
    label: ''
  ecs-key:
    description: SSH key pair used for login
    label: ''
  user-name:
    default: test
  password:
    label: ''
  ecs_vpcId:
    description: ID of the VPC to which the ECS belongs
    label: ''
node_templates:
  ecs:
    properties:
      availabilityZone:
        get_input: ecs_availabilityZone
      flavor:
        get_input: ecs_flavor
      imageId:
        get_input: ecs_imageId
      instances: 1
      name: jkhlh
      nics:
        - subnetId:
            get_input: ecs_nics_0_subnetId
      publicIP:
        eip:
          bandwidth:
            shareType: PER
          ipType: 5_bgp
      rootVolume:
        size: 40
        volumeType: SATA
      sshKeyName:
        get_input: ecs-key
      userData:
        base64_encode:
          replace:
            - |
              #!/bin/bash -x
              useradd ${user_name}
              echo '${user_name}:${user_pwd}' | chpasswd
            - user_name:
                get_input: user-name
              user_pwd:
                get_input: password
      vpcId:
        get_input: ecs_vpcId
    type: Cloud.ECS.CloudServer
```

5.1.9.8 concat

During template compilation, the **concat** function is often used. For example, you may obtain an IP address from the VM result and a listening port number from the APP result, and then print the final and intuitive HTTP access address in the output of the template.

The **concat** function is a built-in function and used to convert descriptions into strings and concatenate them. It can be embedded with the **get_attribute** and **get_input** functions.

In the current version, **concat** can only be defined in the **outputs** section. It cannot be defined in the **node_templates** section or embedded with the **get_attribute** function.

Syntax

```
concat: [args, {get_attribute:[...]}, {get_input: [...]} ]
```

Parameters

Table 5-17 Parameters

Parameter	Description
args	Any user-defined field. The value can be an integer, Boolean value, or string. Example: concat: ["string example", 100, -10, true, false], {get_attribute: [...]}, {get_input: [...]} There is no sequence requirement for the preceding three parameters.

Return Value

Strings that are successfully concatenated are returned.

Examples

```
properties:
  package:
    image: {get_input: magento-image}
    imagePullPolicy: {get_input: imagePullPolicy}
      env:
        - name: MYSQL_HOST # Specifies where MySQL is located.
          value:
            concat:
              - {get_input: mysql-name}
              - .default.svc.cluster.local # Actual address of MySQL, which is an internal domain name of Kubernetes.
        - name: MYSQL_USER
          value: {get_input: mysql-user}
        - name: MYSQL_PASSWORD
          value: {get_input: mysql-password}
        - name: MYSQL_DATABASE
          value: {get_input: mysql-database}
```

```
    - name: ACCESS_URL
      value:
        concat:
          - "http://"
          - {get_input: magento-EIP}
          - ":"
          - {get_input: magento-EPORT-s}
```

5.1.9.9 split

Generally, the **split** function is used together with the **select** or **get_list_length** function. The **split** function is mainly used in the following scenarios:

- A string is split into a group of strings so that specific elements can be easily obtained from the result string list.
- A result string array is directly used.

Syntax

```
split: [delimiter, sourceString ]
```

Parameters

Table 5-18 Parameters

Parameter	Description
delimiter	Separator, which can be a string, single character, or variable obtained using the **get_input** function.
sourceString	Original string, which can be a variable obtained using the **get_input** function. Original strings are grouped by separators.

Return Values

Split string arrays are returned.

Examples

The following describes how to use the **split** function to group strings:

```
inputs:
  source:
    default: "a,b,c,d,e,f,g"
node_templates:
  test:
    type: Cloud.AOS.Stack
    properties:
      templateId: "abcdf-fdeee"
      inputs:
        aaa: {select: [0, {split: [",", {get_input: source}]}]}              # The value is a.
```

5.1.9.10 select

The **select** function can be used to obtain the object with a specified subscript from an array structure. Generally, this function is used together with the **split** function.

Syntax

```
select: [index, list]
```

Parameters

Table 5-19 Parameters

Parameter	Description
index	Subscript, which is used to obtain the specified elements in an array. If the subscript is not in the range supported by the array, an error is reported.
list	Array structure, which cannot be empty.

Return Values

Objects in the corresponding positions in an array are returned.

Examples

The following describes how to use the select function to obtain the specified object:

```
inputs:
  source:
    default: "a,b,c,d,e,f,g"
node_templates:
  test:
    type: Cloud.AOS.Stack
    properties:
      templateId: "abcdf-fdeee"
      inputs:
        aaa: {select: [0, {split: [",", {get_input: source}]}]}              # The value is a.
        bbb: {select: [1, ["alpha", "beta", "gamma"]]}          # The value is beta.
```

5.1.9.11 get_list_length

The **get_list_length** function can be used to calculate the number of elements in an array structure. Generally, this function is used together with the **split** function.

Syntax

```
get_list_length: list
```

Parameter Description

Table 5-20 Parameter description

Parameter	Description
list	Array structure

Return Value

The length of an array is returned.

Examples

The following describes how to use the **get_list_length** function to obtain the length of an array:

```
inputs:
  source:
    default: "a,b,c,d,e,f,g"
node_templates:
  testStack:
    type: Cloud.AOS.Stack
    properties:
      templateId: "abcdf-fdeee"
      inputs:
        aaa: {select: [0, {split: [",", {get_input: source}]}]}                # The value is a.
        bbb: {select: [1, ["alpha", "beta", "gamma"]]}            # The value is beta.
        cc_length: {get_list_length: {split: [",", {get_input: source}]}}       # The value is 7.
        bbb_length: {get_list_length: ["alpha", "beta", "gamma"]}  # The value is 3.
```

5.2 List of Elements

5.2.1 Resource Indexes

Service	Element	Description
AOS	AOS.Stack	The **AOS.Stack** element is used to create stack resources of AOS so that AOS can orchestrate various resources. Corresponding to solutions in real scenarios, this element can deploy a solution in a few clicks. After a model is defined, batch replication can be achieved and services can be migrated to the cloud quickly.
CCE	CCE.Addon.AutoScaler	The **CCE.Addon.AutoScaler** element is a plug-in for node auto-scaling in a Kubernetes cluster.

Service	Element	Description
	CCE.Cluster	The **CCE.Cluster** element is used to deploy Kubernetes cluster resources at the PaaS layer. A master node can be created based on this element to manage and create worker nodes. This element provides users with the application orchestration function.
	CCE.HelmRelease	Helm is a type of Kubernetes-based package specifications provided by CCE. The **CCE.HelmRelease** element is a deployment instance of the Helm package.
	CCE.NodePool	The **CCE.NodePool** element is used to deploy Kubernetes node resources at the PaaS layer. It can be used to orchestrate cloud resources on nodes, providing more powerful functions.
	CCE.Pod	The **CCE.Pod** element is used to create a pod in the Kubernetes cluster.
	CCE.Storage.OBS	The **CCE.Storage.OBS** element corresponds to an OBS bucket under CCE storage management. This type of resources must be used together with CCE clusters.
	CCE.Storage.SFS	The **CCE.Storage.SFS** element corresponds to an SFS file system under CCE storage management. This type of resources must be used together with CCE clusters.
DCS	DCS.Redis	Distributed Cache Service (DCS) is an online, distributed, in-memory cache service. It is reliable, scalable, usable out of the box, and easy to manage. Compatible with Redis and Memcached, DCS supports three instance types: single-node, master/standby, and cluster. It can meet your requirements for high read/write performance and fast data access.
ECS	ECS.CloudServer	The **ECS.CloudServer** element is used to deploy an ECS at the IaaS layer. The ECS is a computing server that consists of the CPU, memory, image, and EVS disk, and allows on-demand allocation and auto scaling.
	ECS.KeyPair	The **ECS.KeyPair** element is used to create a key pair for remote login authentication. For security purposes, you are advised to use the key authentication mode when logging in to an ECS.

Service	Element	Description
NAT Gateway	NAT.Instance	The **NAT.Instance** element is used to create a NAT gateway instance.
	NAT.SNatRule	The **NAT.SNatRule** element is used to create a source NAT rule, which specifies the network segment for accessing the external network.
OBS	OBS.Bucket	The **OBS.Bucket** element is used to deploy an OBS bucket. OBS provides secure, reliable, and cost-effective data storage capabilities, and uses buckets to store objects.
RDS	RDS.MySQL	RDS is a cloud-based web service that is reliable, scalable, easy to manage, and ready to use out-of-the-box.
SFS	SFS.FileSystem	SFS provides high-performance file system storage and supports on-demand scaling. It can be shared by multiple ECSs.
Shared load balancers	ULB.Healthmonitor	The **ULB.Healthmonitor** element is a health check component of a shared load balancer. One pool corresponds to one health monitor. One health monitor can manage multiple ECSs. You can add or delete health monitors as required.
	ULB.Listener	The **ULB.Listener** element indicates a listener of a shared load balancer. One load balancer corresponds to multiple listeners. You can add or delete listeners as required.
	ULB.LoadBalancer	The **ULB.LoadBalancer** element is used to deploy a shared load balancer at the PaaS layer. By creating such a shared load balancer, you can provide a unified entry for a group of containerized applications with the same functions, and distribute requests to backend containerized applications in load balancing mode. Shared load balancers are applicable to web services with high access traffic. They forward requests based on domain names or URLs, making request routing more flexible. Compared with classic load balancers, shared load balancers provide better HTTP and HTTPS forwarding performance and stability.
	ULB.Member	The **ULB.Member** element indicates an ECS. One pool corresponds to multiple ECSs. You can add or delete ECSs as required.

Service	Element	Description
	ULB.Pool	The **ULB.Pool** element indicates an ECS group. A listener corresponds to multiple ECS groups. You can add or delete ECS groups as required. An ECS group consists of multiple ECSs.
VPC	VPC.EIP	The **VPC.EIP** element is used to create a public elastic IP address. A public elastic IP address is a static IP address. You can bind or unbind an elastic IP address to an ECS in a subnet. An ECS in a VPC can access the Internet through a fixed public IP address.
	VPC.SecurityGroup	The **VPC.SecurityGroup** element indicates a logical group. It provides access control rules for ECSs which have the same security protection requirements and are mutually trusted in a VPC.
	VPC.SecurityGroupRule	The **VPC.SecurityGroupRule** element indicates an access policy added for an ECS to implement access control.
	VPC.Subnet	The **VPC.Subnet** element is used to create a VPC subnet for cloud products.
	VPC.VPC	The **VPC.VPC** element is used to create a VPC for cloud products.

5.2.2 AOS.Stack

Element Description

The **AOS.Stack** element is used to create stack resources of AOS so that AOS can orchestrate various resources. The **AOS.Stack** element corresponds to the solution in real scenarios. It can implement one-click deployment of the solution. After being defined, the element can be replicated in batches, helping services to be quickly deployed on the cloud.

Element Properties

Table 5-21 Property Description

Property	Mandatory	Descripiton
inputs	Yes	Input information required by the nested stack **Type:** dict **Value Description:** a customized structure **Default:** {} **Value Constraint:** A maximum of 60 inputs properties can be defined in a template.
description	No	Stack description **Type:** string **Value Description:** Customize the value. **Default:** "" **Value Constraint:** The value must be a text string and contain a maximum of 1,024 characters.
failureStrategy	No	Failure strategy **Type:** string **Value Description:** The options are **DoNothing** and **Rollback**. **Default:** DoNothing
deploy	No	Whether to deploy the application **Type:** boolean **Value Description:** The options are **true** and **false**. If this parameter is set to **false**, the application (including software components contained in the application and host resources required by the application) will not be deployed. **Default:** true

Property	Mandatory	Descripiton
clusterId	No	ID of the cluster which is associated with the storage system **Type:** Cloud.CCE.Cluster.Id **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Enter the cluster ID. Specifically, log in to the CCE console, and choose **Resource Management** > **Clusters**. Click the target cluster, and you can obtain its cluster ID. 2. Connect to the cluster object and use the **get_reference** function to automatically obtain the value.
templateId	Yes	ID of the template that the created stack depends on **Type:** string **Value Description:** Enter an ID of an existing template. **Value Constraint:** The value must be a text string and contain a maximum of 64 characters.

Relationships Between Elements

Table 5-22 Relationship description

Description	Target
Dependency	**VPC.EIP**
Dependency	**CCE.Addon.AutoScaler**
Dependency	**CCE.Cluster**
Dependency	**SFS.FileSystem**
Dependency	**AOS.Stack**
Dependency	**NAT.Instance**
Dependency	**OBS.Bucket**

Description	Target
Dependency	**CCE.Storage.SFS**
Dependency	**CCE.HelmRelease**
Dependency	**CCE.NodePool**
Dependency	**ECS.KeyPair**
Dependency	**CCE.Pod**
Dependency	**DCS.Redis**
Dependency	**VPC.VPC**
Dependency	**ECS.CloudServer**
Dependency	**VPC.Subnet**
Dependency	**CCE.Storage.OBS**
Dependency	**RDS.MySQL**
Inclusion	**CCE.Cluster**

Return Value

Property	Type	Description
refName	string	Solution stack name
refID	string	Solution stack ID

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  delpoy1:
   default: false
   type: boolean
  delpoy2:
   default: true
```

```
    type: boolean
   delpoy3:
    default: true
    type: boolean
   description:
    default: nginx stack
    type: string
   template-id1:
    default: 370f60c6-afc2-e08a-d1c4-fd33bd58b785
    type: string
   template-id2:
    default: 753c30cf-3b3b-cd63-f7f0-1550d058eaac
    type: string
   template-id3:
    default: 2fdd9e05-1406-15d4-7b35-1274a036bcfb
    type: string
   images:
    default: 192.168.0.249:20202/op_svc_servicestage_88b899/nginx:latest
    type: string
node_templates:
  stackone:
    type: Cloud.AOS.Stack
    properties:
      deploy: {get_input: delpoy1}
      description: {get_input: description}
      templateId: {get_input: template-id1}
      inputs:
        images: {get_input: images}
    requirements:
      - dependency:
          node: stacktwo
  stacktwo:
    type: Cloud.AOS.Stack
    properties:
      deploy: {get_input: delpoy2}
      description: {get_input: description}
      templateId: {get_input: template-id2}
      inputs:
        images: {get_input: images}
        myport: {get_attribute: [stackthree,nginx-NodePort]}
    requirements:
      - dependency:
          node: stackthree
  stackthree:
    type: Cloud.AOS.Stack
    properties:
      deploy: {get_input: delpoy3}
      description: {get_input: description}
      templateId: {get_input: template-id3}
      inputs:
        image: {get_input: images}
```

5.2.3 CCE.Addon.AutoScaler

Element Description

CCE.Addon.AutoScaler is a plug-in for node auto-scaling in a K8S cluster.

Element Properties

Table 5-23 Property description

Property	Mandatory	Descripiton
scaleDownUtilizationThreshold	No	Node resource usage ratio **Type:** float **Value Description:** The value ranges from 0 to 1. **Default:** 0.4 **Value Constraint:** The value ranges from 0 to 1. **Suggestion:** Set this parameter based on the live environment.
clusterId	Yes	ID of the cluster to which the resource belongs **Type:** Cloud.CCE.Cluster.Id **Value Description:** Indicates the ID of an existing or new container cluster. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Enter the cluster ID. Specifically, log in to the CCE console, and choose **Resource Management** > **Clusters**. Click the target cluster, and you can then obtain its cluster ID. 2. Connect to the cluster object and use the **get_reference** function to obtain the cluster ID.
scaleDownEnabled	Yes	Scale down function switch **Type:** boolean **Default:** False
publicKey	No	Public key. This parameter is mandatory if the stack is billed in the yearly/monthly mode. **Type:** Cloud.ECS.KeyPair.PublicKey
nodePasswd	No	Password of the scaled node **root** user **Type:** password
nodes	Yes	AZs, specifications, OSs, and taints of the scaled nodes **Type:** **CCE.Addon.AutoScaler.Node** array **Suggestion:** In node scaling, taints are an array consisting of key, value, and effect. The effect can be set to **NoSchedule**, **PreferNoSchedule**, or **NoExecute**.
sshKeyName	No	Node key pair **Type:** Cloud.ECS.KeyPair.Name

Property	Mandatory	Descripiton
scaleDownUnneededTime	No	When a node remains idle for this specified time duration (in minutes), scaling down will be performed. **Type:** integer **Value Description:** The value ranges from 1 to 1000. **Default:** 10 **Value Constraint:** The value ranges from 1 to 1000. **Suggestion:** Set this parameter based on the live environment.

Relationships Between Elements

Table 5-24 Relationship description

Description	Target
Dependency	**CCE.NodePool**
Inclusion	**CCE.Cluster**

Return Value

Property	Type	Description
clusterId	string	ID of the cluster which is associated with the AutoScaler
refName	string	Name of the AutoScaler
refID	string	UID of the AutoScaler

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  clusterId:
    default: "e0f98d46-9716-11e8-a25f-0255ac106314"
    description: cluster ID.
  nodePasswd:
    default: "******"
    description: node root user password
  scaleDownEnabled:
    default: true
    description: scale down enabled.
  scaleDownUnneededTime:
    default: 10
```

```
    description: sale down unneeded time
  scaleDownUtilizationThreshold:
    default: 0.5
    description: scale down utilization threshold
  avaliableZone:
    default: az1.dc1
    description: avaliableZone.
  nodeFlavor:
    default: s1.xlarge
    description: node flavor.
  nodeOS:
    default: EulerOS 2.2
    description: node OS.
node_templates:
  autoscaler:
    type: Cloud.CCE.Addon.AutoScaler
    properties:
      clusterId:
        get_input: clusterId
      nodePasswd:
        get_input: nodePasswd
      scaleDownEnabled:
        get_input: scaleDownEnabled
      scaleDownUnneededTime:
        get_input: scaleDownUnneededTime
      scaleDownUtilizationThreshold:
        get_input: scaleDownUtilizationThreshold
      nodes:
        - az:
            get_input: avaliableZone
          flavor:
            get_input: nodeFlavor
          os:
            get_input: nodeOS
outputs:
  autoscaler_id:
    value: {get_attribute: [autoscaler, refID]}
```

5.2.4 CCE.Cluster

Element Description

The **CCE.Cluster** element is used to deploy Kubernetes cluster resources at the PaaS layer. A master node can be created based on this element to manage and create worker nodes. This element provides the application orchestration function for users.

Element Properties

Table 5-25 Property Description

Property	Mandatory	Description
multiAZ	No	Multi-AZ cluster **Type:** boolean **Default:** False **Value Constraint:** Only when HA clusters are used, for example, clusters of cce.s2 specifications, can you set this parameter to true. **Suggestion:** If **multiAZ** is set to **true**, the cluster flavor must support multi-AZ cluster creation, for example, flavors of cce.s2 specifications.
vpcId	Yes	VPC ID **Type:** Cloud.VPC.VPC.Id **Value Constraint:** An existing or new VPC ID can be used. To use a new VPC ID, you need to define the VPC object in the template and establish the dependency. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.VPC** element created by the stack. 3. Obtain the ID of the created VPC from the VPC console.
networkMode	No	Container network type **Type:** string **Default:** overlay_l2 **Value Constraint:** Currently, **overlay_l2**, **underlay_ipvlan**, and **vpc-router** are supported. If you set it to **vpc-router**, the selected VPC can contain only one subnet. **Suggestion:** Use the default value.
description	No	Cluster description **Type:** string **Suggestion:** Customize the value.
name	No	Cluster name **Type:** string **Value Constraint:** Enter 4 to 128 characters, starting with a letter and ending with a letter or digit. Only lowercase letters, digits, and hyphens (-) are allowed. The value must meet regular expression **(^$)\|(^[a-z]([-a-z0-9]*[a-z0-9])?$)**. **Suggestion:** Customize the value.

Property	Mandatory	Description
kubeProxyMode	No	Service forwarding mode **Type:** string **Default:** iptables **Value Constraint:** Currently, only **iptables** and **ipvs** are supported. **Suggestion:** You are advised to use the default value **iptables** for cluster 1.7, and **ipvs** for cluster 1.9 and later to achieve better performance.
highwaySubnetId	No	High-speed subnet ID **Type:** Cloud.VPC.Subnet.Id **Value Constraint:** An existing or new subnet ID can be used. To use a new subnet ID, you need to define the subnet object in the template and establish the dependency. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.Subnet** element created by the stack. 3. Obtain the ID of the created subnet from the VPC console.
containerNetworkCIDR	No	Container network segment **Type:** string **Default:** "" **Value Constraint:** Set this parameter based on the live environment. The available network segments are **172.16.0.0/16-172.31.0.0/16**, **10.0.0.0/16-10.255.0.0.0/16**, and **192.168.0.0/16**. **Suggestion:** Use the default value.
version	No	Cluster version **Type:** string **Value Constraint:** Currently, v1.15, v1.13 and v1.11 are supported. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.
namespaces	No	Namespace created during cluster creation **Type:** string array **Default:** [] **Value Constraint:** array

Property	Mandatory	Description
subnetId	Yes	Subnet ID **Type:** Cloud.VPC.Subnet.Id **Value Constraint:** An existing or new subnet ID can be used. To use a new subnet ID, you need to define the subnet object in the template and establish the dependency. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.Subnet** element created by the stack. 3. Obtain the ID of the created subnet from the VPC console.
flavor	Yes	Cluster specifications **Type:** Cloud.CCE.Cluster.Flavor.Name **Value Constraint:** The value must comply with CCE specifications definitions. You can view supported specifications on the CCE console. **Suggestion:** You can view the names of the available cluster specifications on the cluster creation page of the CCE console.
type	No	Cluster type **Type:** Cloud.CCE.Cluster.Type **Default:** VirtualMachine **Value Constraint:** Currently, **VirtualMachine**, **BareMetal**, **Windows**, and **ARM64** are supported. **Suggestion:** Use the default value.
nodes	No	User node configuration during yearly/monthly-billed cluster creation **Type:** **CCE.NodePool** **Default:** {u'dataVolumes': [], u'availabilityZone': u'unset', u'instances': 1, u'rootVolume': {u'volumeType': u'unset', u'size': 40}, u'flavor': u'unset', u'sshKeyName': u'unset'} **Value Constraint:** The value of this parameter must comply with the description and constraint of **Cloud.CCE.NodePool**. **Suggestion:** Set the value based on the live environment.

Property	Mandatory	Description
availabilityZone	No	AZ. For clusters billed in the yearly/monthly mode, this field is mandatory. **Type:** Cloud.ECS.AvailabilityZone.Name **Value Constraint:** The value varies depending on regions. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.

Relationships Between Elements

Table 5-26 Relationship description

Description	Target
Connected	**VPC.Subnet**
Inclusion	**VPC.VPC**

Return Value

Property	Type	Description
refName	string	Cluster name
refID	string	Cluster ID

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  availabilityZone:
    default: az1.dc1
  vpcId:
    default: ba6e4347-99d2-4649-b114-85c28d3d71b0
  subnetId:
    default: 3be61f68-9bfc-41bf-8f5e-66c57122f270
  clusterFlavor:
    default: cce.s1.small
node_templates:
  cluster:
    type: Cloud.CCE.Cluster
    properties:
      availabilityZone: {get_input: availabilityZone}
      vpcId: {get_input: vpcId}
      subnetId: {get_input: subnetId}
      flavor: {get_input: clusterFlavor}
```

```
outputs:
  cluster_id:
    value: {get_attribute: [cluster, clusterId]}
```

5.2.5 CCE.HelmRelease

Element Description

Helm is a type of Kubernetes-based package specifications provided by CCE. The **CCE.HelmRelease** element is a deployment instance of the Helm package.

Element Properties

Table 5-27 Property Description

Property	Mandatory	Descripiton
name	Yes	Name of the created **CCE.HelmRelease** **Type:** string **Value Description:** Customize the value, for example, my_release. **Value Constraint:** The value must start with a letter. Only lowercase letters, digits, and hyphens (-) are allowed.
clusterId	No	ID of the cluster to which the resource belongs **Type:** Cloud.CCE.Cluster.Id **Value Description:** ID of an existing or new container cluster. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** This parameter is optional. You can set this parameter when creating a stack.
namespace	No	Namespace where the resource is located **Type:** string **Value Description:** namespace of a cluster **Default:** default **Value Constraint:** The value must start with a letter. Only lowercase letters, digits, and hyphens (-) are allowed. **Suggestion:** This parameter is optional. You can set this parameter when creating a stack.

Property	Mandatory	Descripiton
chart	Yes	Chart information about the Helm application **Type:** CCE.HelmChart **Value Description:** Information includes the chart package name and version number, which can be obtained from **Charts** in the navigation pane on the CCE console. **Default:** {u'version': u'', u'name': u''} **Suggestion:** Set the value based on the helm application to be orchestrated.
values	Yes	Input value of the Helm application **Type:** dict **Value Description:** Customize the value. **Default:** {} **Value Constraint:** Composite structure, which is similar to {"key": "value"}, where value can be nested. **Suggestion:** For your own applications, enter the corresponding value.

Relationships Between Elements

Table 5-28 Relationship description

Description	Target
Connected	CCE.Pod
Connected	CCE.Storage.SFS
Connected	CCE.HelmRelease
Connected	CCE.Storage.OBS
Inclusion	CCE.Cluster

Return Value

Property	Type	Description
clusterId	string	Cluster ID
refName	string	Release name

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  release_name:
    default: "release"
  cluster_id:
    default: "25f511bc-00f7-11e8-958d-0255ac101a5a"
  namespace:
    default: "default"
  chart_name:
    default: "redis"
  chart_version:
    default: "1.0.0"
  app_image:
    default: "10.125.5.235:20202/test/redis:3.2.8"
  config_image:
    default: "10.125.5.235:20202/test/redis-conf:3.2.8"
  service_port:
    type: integer
    default: 6379
node_templates:
  redis-helm:
    type: Cloud.CCE.HelmRelease
    properties:
      name: {get_input: release_name}
      chart:
        name: {get_input: chart_name}
        version: {get_input: chart_version}
      clusterId: {get_input: cluster_id}
      namespace: {get_input: namespace}
      values:
        chartimage:
          app_image: {get_input: app_image}
          config_image: {get_input: config_image}
        format1:
          redis_master_replicas: 1
          redis_sentinel_replicas: 1
          redis_slave_replicas: 1
        format2:
          redis_master_replicas: 1
          redis_sentinel_replicas: 1
          redis_slave_replicas: 2
        highavailable:
          redis_replication_enabled: true
          redis_sentinel_replicas: 1
          redis_slave_replicas: 1
        servicestorage:
          service:
            instance: "127.0.0.1"
            service_port: {get_input: service_port}
            type: "ClusterIP"
          storage:
            enabled: false
            kind: "sas"
            size: "10Gi"
```

5.2.6 CCE.NodePool

Element Description

The **CCE.NodePool** element is used to deploy Kubernetes node resources at the PaaS layer. You can use it to orchestrate cloud resources on nodes.

Element Properties

Table 5-29 Property Description

Property	Mandatory	Description
dataVolumes	Yes	Data disk of a created node **Type:** CCE.DataVolume array **Value Description:** Customize the value, for example, [{"volumeType":"SATA","size":100}]. **Value Constraint:** Array format. Currently, only one object is supported. **Suggestion:** Customize the value.
availabilityZone	Yes	AZ where a node is located **Type:** Cloud.ECS.AvailabilityZone.Name **Value Description:** AZ where the to-be-created ECS is located. The name of the AZ needs to be specified, for example, ae-ad-1a. For details, see the Regions and Endpoints page. **Value Constraint:** The value varies depending on regions. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.
name	Yes	Name of the created node **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value contains 4 to 32 characters and must start with a lowercase letter. Only lowercase letters, digits, and underscores (_) are allowed. **Suggestion:** Customize the value. Generally, the stack name is used as the node name.

Property	Mandatory	Description
publicKey	No	Public key of the key pair. If the node pool is billed in the yearly/month mode, this parameter is mandatory. **Type:** Cloud.ECS.KeyPair.PublicKey **Value Description:** Selects an existing public key. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console based on the value of **sshKeyName**.
postInstall	No	Node post-installation script **Type:** string **Value Description:** Customize the value. **Value Constraint:** The script you specify here will be executed after K8S software is installed. **Suggestion:** The script is usually used to modify Docker parameters.
labels	No	Node label **Type:** **CCE.Labels** array **Value Description:** Customize the value, for example, {"app": "aos"}. **Suggestion**: You can enter multiple key-value pairs.
clusterId	No	ID of the cluster to which the resource belongs **Type:** Cloud.CCE.Cluster.Id **Value Description:** ID of an existing or new container cluster. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Enter the cluster ID. Specifically, log in to the CCE console, and choose **Resource Management** > **Clusters**. Click the target cluster, and you can then obtain its cluster ID. 2. Connect to the cluster object and use the **get_reference** function to obtain the cluster ID.
preInstall	No	Node pre-installation script **Type:** string **Value Description:** Customize the value. **Value Constraint:** The script you specify here will be executed before K8S software is installed. Note that if the script is incorrect, K8S software may not be installed successfully. **Suggestion:** The script is usually used to format data disks.

Property	Mandatory	Description
publicIp	No	Virtual IP address of the created node **Type:** CCE.PublicIP **Value Description:** Customize the value, for example, {"eip":{"bandwidth:{"shareType":PER}, 5_sbgp"}}. **Default:** {} **Value Constraint:** Only one elastic IP address can be defined for each node. **Suggestion:** Customize the value.
instances	Yes	Number of created nodes **Type:** integer **Value Description:** Customize the value. The value ranges from 1 to 50. **Default:** 1 **Value Constraint:** {u'in_range': [1, 50]} **Suggestion:** Set the value based on the live environment.
rootVolume	Yes	System disk of the created node **Type:** ECS.RootVolume **Value Description:** Customize the value, for example, {"volumeType":"SATA","size":40}. **Default:** {u'volumeType': u'unset', u'size': 40} **Suggestion:** Customize the value.
os	No	Node OS **Type:** string **Value Description:** ["EulerOS 2.2", "CentOS 7.4"] **Default:** EulerOS 2.2 **Value Constraint:** {u'valid_values': [u'CentOS 7.4', u'EulerOS 2.2']}

Property	Mandatory	Description
nodePasswd	No	Password of the node **root** user **Type:** password **Value Description:** Customize the value. **Value Constraint:** 1. The parameter must be written into inputs and set using the **get_input** function. 2. The value must not be a weak password. Enter 8 to 26 characters. Only uppercase and lowercase letters, digits, and special characters !@$%^-_=+[{}]:,./? are allowed. The value must contain at least two types of characters. **Suggestion:** You are advised to use the get_input function to obtain the value and avoid plaintext passwords to ensure security.
flavor	Yes	Container node specification **Type:** Cloud.CCE.Node.Flavor.Name **Value Description:** ID of the system flavor of the cloud server to be created. For details about the available flavors, see *Elastic Cloud Server Service Overview*. It is advised to use the **get_input** function to pass this parameter. **Suggestion:** Select the node specification during node creation on the CCE console. In the node template, you can set the inputs to specify the node specification.
sshKeyName	No	Key pair used for logging in to a node, which needs to be kept properly **Type:** Cloud.ECS.KeyPair.Name **Value Description:** It must be created on the ECS console in advance. **Suggestion:** 1. You are advised to use the **get_input** function to set the parameter so that you can select a value when using the template. 2. Search for the information on the ECS console, and enter the information accordingly.
annotations	No	Node annotations **Type:** dict **Value Description:** Customize the value, for example, {"app": "aos"}. **Suggestion**: You can enter multiple key-value pairs.

Relationships Between Elements

Table 5-30 Relationship description

Description	Target
Inclusion	**CCE.Cluster**
Connected	**ECS.KeyPair**
Connected	**CCE.Storage.OBS**
Connected	**CCE.Pod**
Connected	**CCE.NodePool**
Connected	**CCE.Storage.SFS**

Return Value

Property	Type	Description
floatingIpId	string	ID of an elastic IP address
clusterId	string	Cluster ID
refName	string	Node name
privateIp	Array	List of private elastic IP addresses
publicIp	Array	List of public elastic IP addresses
refID	string	Node ID

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
node_templates:
  ccenp1ep:
    type: Cloud.CCE.NodePool
    properties:
      dataVolumes:
        - volumeType: SATA
          size: 100
      name: ''
      instances: 1
      rootVolume:
        volumeType: SATA
        size: 40
```

```
      flavor:
        get_input: ccenp1ep_flavor
      sshKeyName:
        get_input: ccenp1ep_sshKeyName
inputs:
  ccenp1ep_flavor:
    description: Container node specifications
    label: ''
  ccenp1ep_sshKeyName:
    description: Key pair used for logging in to a node. Keep the key pair properly.
    label: ''
```

5.2.7 CCE.Pod

Element Description

The **CCE.Pod** element is used to create a pod in the Kubernetes cluster on the CCE.

Element Properties

Table 5-31 Property Description

Property	Mandatory	Descripiton
k8sManifest	Yes	Native YAML file content of a Kubernetes object **Type:** dict **Value Description:** Customize the value. You are advised to use a public image, which is uploaded to the image repository and whose type is set to public, and not to change the name under the metadata during an update. **Value Constraint:** This field cannot be left blank.
name	No	Pod name **Type:** string **Value Description:** Customize the value, for example, my-pod. **Value Constraint:** The value supports a maximum of 63 characters and must start with a letter. Only lowercase letters, digits, and hyphens (-) are allowed.

Property	Mandatory	Descripiton
clusterId	No	ID of the cluster to which the resource belongs **Type:** Cloud.CCE.Cluster.Id **Value Description:** ID of an existing or new container cluster. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Enter the cluster ID. Specifically, log in to the CCE console, and choose **Resource Management** > **Clusters**. Click the target cluster, and you can then obtain its cluster ID. 2. Connect to the cluster object and use the **get_reference** function to obtain the cluster ID.
namespace	No	Namespace where the resource is located **Type:** string **Value Description:** It must be a valid namespace in the cluster, for example, default. **Value Constraint:** The value must start with a letter. Only lowercase letters, digits, and hyphens (-) are allowed. **Suggestion:** Log in to the CCE console, and choose **Resource Management** > **Namespaces**. View and select the target namespace.

Relationships Between Elements

Table 5-32 Relationship description

Description	Target
Dependency	**DCS.Redis**
Dependency	**RDS.MySQL**
Dependency	**OBS.Bucket**
Dependency	**CCE.Storage.SFS**
Dependency	**CCE.Storage.OBS**

Description	Target
Dependency	**CCE.NodePool**
Dependency	**CCE.Pod**
Inclusion	**CCE.Cluster**

Return Value

Property	Type	Description
refName	string	Pod name

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
node_templates:
  ccepxbto:
    type: Cloud.CCE.Pod
    properties:
      k8sManifest:
        kind: Pod
        spec:
          containers:
            - image:
                get_input: ccepxbto_k8sManifest_spec_containers_0_image
              imagePullSecrets:
                - name: default-secret
              name: test
              restartPolicy: Always
              imagePullPolicy: Always
        apiVersion: v1
        metadata:
          labels:
            name: pod-test
          name: pod-test
      name:
        get_input: ccepxbto_name
      clusterId:
        get_input: ccepxbto_clusterId
      namespace:
        get_input: ccepxbto_namespace
inputs:
  ccepxbto_k8sManifest_spec_containers_0_image:
    description: Container image
    label: Pod
  ccepxbto_name:
    description: Pod name
    label: Pod
  ccepxbto_clusterId:
    description: ID of the cluster where the resource is located
    label: Pod
  ccepxbto_namespace:
    description: Namespace where the resource is located
    label: Pod
outputs:
```

```
name:
  value:
    get_attribute:
      - ccepxbto
      - refName
  description: pod name
```

5.2.8 CCE.Storage.OBS

Element Description

The **CCE.Storage.OBS** element corresponds to object storage volumes in the CCE storage management function. This type of resources must be used together with CCE clusters.

Element Properties

Table 5-33 Property Description

Property	Mandatory	Descripiton
k8sManifest	No	K8s-native manifest object of the OBS, which can be used to replace other configuration items in OBS resource creation **Type:** dict **Value Constraint:** The value must meet the Kubernetes specifications. **Suggestion:** For details, see the sample or CCE documentation.
name	No	PVC name **Type:** string **Value Description:** Customize the value. **Value Constraint:** Each PVC name must be unique in a namespace. The value must contain 1 to 24 characters and meet regular expression **(^$)\|(^[a-z]([-a-z0-9]*[a-z0-9])?$)**. **Suggestion:** Customize the value.

Property	Mandatory	Descripiton
clusterId	No	ID of the cluster to which the resource belongs **Type:** Cloud.CCE.Cluster.Id **Value Description:** ID of an existing or new container cluster. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Enter the cluster ID. Specifically, log in to the CCE console, and choose **Resource Management** > **Clusters**. Click the target cluster, and you can then obtain its cluster ID. 2. Connect to the cluster object and use the **get_reference** function to obtain the cluster ID. 3. Leave it blank, and specify the ID on the AOS console when creating a stack.
namespace	No	Namespace to which the resource belongs **Type:** string **Value Description:** It must be a valid namespace in the cluster, for example, default. **Value Constraint:** The value must start with a letter. Only lowercase letters, digits, and hyphens (-) are allowed. **Suggestion:** Customize the value based on the existing cluster or the cluster to be created.
volumeId	No	Cloud storage volume ID when you import a volume **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** None
deleteVolume	No	Whether to delete the cloud storage when you import a volume and delete a PVC **Type:** boolean **Default:** False **Value Constraint:** The options are **true** and **false**. **Suggestion:** None

Relationships Between Elements

Table 5-34 Relationship description

Description	Target
Dependency	**DCS.Redis**
Dependency	**RDS.MySQL**
Dependency	**OBS.Bucket**
Dependency	**CCE.Storage.SFS**
Dependency	**CCE.Storage.OBS**
Dependency	**CCE.NodePool**
Dependency	**CCE.Pod**
Inclusion	**CCE.Cluster**

Return Value

Property	Type	Description
clusterId	string	ID of the cluster which is associated with the OBS file system
refID	string	UID of the OBS file system
refName	string	Name of the OBS file system

Blueprint Example

Example 1:

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  storage-name:
    default: my-evc-storage
node_templates:
  my-storage:
    type: Cloud.CCE.Storage.OBS
    properties:
      name: {get_input: storage-name}
```

Example 2: Custom K8s Manifest for Orchestration

- For clusters of version 1.13 or earlier, the example configuration of the YAML file is as follows:

```
tosca_definitions_version: cloud_tosca_version_1_0
node_templates:
  my-storage:
    type: Cloud.CCE.Storage.OBS
    properties:
      apiVersion: v1
      kind: PersistentVolumeClaim
      metadata:
        annotations:
          volume.beta.kubernetes.io/storage-class: obs-standard
          volume.beta.kubernetes.io/storage-provisioner: flexvolume.com/obs
        name: cce-obs-k7yhr36u-iuu9
        namespace: default
      spec:
        accessModes:
        - ReadWriteMany
        resources:
          requests:
            storage: 10Gi
```

5.2.9 CCE.Storage.SFS

Element Description

The **CCE.Storage.SFS** element corresponds to file storage volumes in the CCE storage management function. This type of resources must be used together with CCE clusters.

Element Properties

Table 5-35 Property Description

Property	Mandatory	Descripiton
size	No	Storage space size, in GB. The default value is **80**. **Type:** integer **Default:** 10 **Value Constraint:** The value ranges from 1 to 511800. **Suggestion:** Set the value based on the live environment.
k8sManifest	No	K8s-native manifest object of the SFS, which can be used to replace other configuration items in SFS resource creation **Type:** dict **Value Constraint:** The value must meet the Kubernetes specifications. **Suggestion:** For details, see the sample or CCE documentation.

Property	Mandatory	Descripiton
name	No	Name of the CCE SFS file system, which is mounted to the container **Type:** string **Value Constraint:** Enter 1 to 24 characters, starting with a letter and ending with a letter or digit. Only lowercase letters, digits, and hyphens (-) are allowed. The value must meet regular expression **(^$)\|(^[a-z]([-a-z0-9]*[a-z0-9])?$)**. **Suggestion:** Customize the value.
clusterId	No	ID of the cluster which is associated with the storage system **Type:** Cloud.CCE.Cluster.Id **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Enter the cluster ID. Specifically, log in to the CCE console, and choose **Resource Management** > **Clusters**. Click the target cluster, and you can then obtain its cluster ID. 2. Connect to the cluster object and use the **get_reference** function to obtain the cluster ID.
volumeId	No	Cloud storage volume ID when you import a volume **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** None
deleteVolume	No	Whether to delete the cloud storage when you import a volume and delete a PVC **Type:** boolean **Default:** False **Value Constraint:** The options are **true** and **false**. **Suggestion:** None
namespace	No	Namespace where the resource is located **Type:** string **Value Constraint:** The value must start with a letter. Only lowercase letters, digits, and hyphens (-) are allowed. **Suggestion:** Log in to the CCE console, and choose **Resource Management** > **Namespaces**. View and select the target namespace.

Relationships Between Elements

Table 5-36 Relationship description

Description	Target
Dependency	**DCS.Redis**
Dependency	**RDS.MySQL**
Dependency	**OBS.Bucket**
Dependency	**CCE.Storage.SFS**
Dependency	**CCE.Storage.OBS**
Dependency	**CCE.NodePool**
Dependency	**CCE.Pod**
Inclusion	**CCE.Cluster**

Return Value

Property	Type	Description
status	string	Status of the SFS file system
clusterId	string	ID of the cluster which is associated with the SFS file system
refID	string	UID of the SFS file system
refName	string	Name of the SFS file system

Blueprint Example

Example 1:

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  storage-name:
    default: my-evc-storage
node_templates:
  my-storage:
    type: Cloud.CCE.Storage.SFS
    properties:
```

```
    name:
      get_input: storage-name
```

Example 2: Custom K8s Manifest for Orchestration

- For clusters of version 1.13 or earlier, the example configuration of the YAML file is as follows:

```
tosca_definitions_version: cloud_tosca_version_1_0
node_templates:
  my-storage:
    type: Cloud.CCE.Storage.SFS
    properties:
      apiVersion: v1
      kind: PersistentVolumeClaim
      metadata:
        annotations:
          volume.beta.kubernetes.io/storage-class: nfs-rw
          volume.beta.kubernetes.io/storage-provisioner: flexvolume.com/fs
        name: cce-sfs-k7yimkqa-p66e
        namespace: default
      spec:
        accessModes:
        - ReadWriteMany
        resources:
          requests:
            storage: 10Gi
```

5.2.10 DCS.Redis

Element Description

Distributed Cache Service (DCS) is an online, distributed, in-memory cache service. It is reliable, scalable, usable out of the box, and easy to manage. Compatible with Redis and Memcached, DCS supports three instance types: single-node, master/standby, and cluster. It can meet your requirements for high read/write performance and fast data access.

Element Properties

Table 5-37 Property Description

Property	Mandatory	Descripiton
vpcId	Yes	ID of the VPC to which the DCS instance belongs **Type:** Cloud.VPC.VPC.Id **Value Description:** Use the ID of an existing VPC or a new VPC. To use a new VPC ID, you need to define the VPC object in the template and establish the dependency. You are advised to drag the object to the VPC to automatically establish the dependency. **Value Constraint:** The value must satisfy the UUID generation rule. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.VPC** element created by the stack. 3. Obtain the ID of the created VPC from the VPC console.
capacity	Yes	Capacity of the DCS instance **Type:** integer **Value Description:** Customize the value. **Default:** 2 **Value Constraint:** Currently, the value can only be **2**, **4**, **8**, **16**, **32**, **64**, **128**, **256**, **512**, or **1024**. **Suggestion:** Use the default value.
description	No	Description of the DCS instance **Type:** string **Value Description:** Customize the value. **Value Constraint:** {u'max_length': 1024}
name	No	Name of the DCS instance **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value must start with a letter. Only letters, digits, underscores (_), and hyphens (-) are allowed.

Property	Mandatory	Descripiton
securityGroupId	Yes	ID of the security group used by the DCS instance **Type:** Cloud.VPC.SecurityGroup.Id **Value Description:** Obtains the security group ID from the VPC service or connects to the **VPC.SecurityGroup** to automatically generate a security group ID. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.SecurityGroup** element created by the stack. 3. Obtain the ID of the created security group from the VPC console.
availablityZone1	No	AZ1 to which the DCS instance belongs **Type:** Cloud.ECS.AvailabilityZone.Name **Value Description:** AZ1 to which the DCS instance belongs. The AZ can be automatically selected on the AOS console. You need to specify the AZ name, for example, ae-ad-1a. For details, see the Regions and Endpoints page. **Value Constraint:** The value varies depending on regions. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.
instanceMode	Yes	Type of the DCS instance **Type:** string **Default:** single **Value Constraint:** Currently, the value can only be **single**, **HA**, or **cluster**. **Suggestion:** Use the default value.
availablityZone2	No	AZ2 to which the DCS instance belongs. AZ2 is required for creating master/standby DCS instances. **Type:** Cloud.ECS.AvailabilityZone.Name **Value Description:** AZ2 to which the DCS instance belongs. The AZ can be automatically selected on the AOS console and must be different from AZ1. You need to specify the AZ name. For details, see the Regions and Endpoints page. **Value Constraint:** The value varies depending on regions. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.

Property	Mandatory	Descripiton
instanceBackupPolicy	No	Backup plan of the DCS instance **Type:** DCS.InstanceBackupPolicy **Value Description:** Customize the value. **Default:** {u'extendParam': {u'backupAt': [], u'beginAt': u'00', u'periodType': u'weekly'}, u'backupType': u'auto', u'saveDays': 1} **Suggestion:** Use the default value.
maintainBegin	No	Start time of the maintenance time window **Type:** string **Default:** 02:00:00 **Value Constraint:** Currently, the value can only be **02:00**, **06:00**, **10:00**, **14:00**, **18:00**, or **22:00**. **Suggestion:** Use the default value.
subnetId	Yes	Subnet ID of the DCS instance **Type:** Cloud.VPC.Subnet.Id **Value Description:** Use the ID of an existing subnet or a new subnet. To use a new subnet ID, you need to define the subnet object in the template and establish the dependency. You are advised to connect the **VPC.Subnet** to automatically establish the dependency. **Value Constraint:** The subnet must correspond to the VPC. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.Subnet** element created by the stack. 3. Obtain the ID of the created subnet from the VPC console.
maintainEnd	No	End time of the maintenance time window **Type:** string **Default:** 06:00:00 **Value Constraint:** Currently, the value can only be **06:00**, **10:00**, **14:00**, **18:00**, **22:00**, or **02:00**. **Suggestion:** Use the default value.

Property	Mandatory	Descripiton
password	Yes	Login password of the DCS instance **Type:** password **Value Description:** Customize the value. **Value Constraint:** 1. The parameter must be written into inputs and set using the **get_input** function. 2. It must not be a weak password. Enter 6 to 32 characters. Only uppercase and lowercase letters, digits, and special characters `~!@#$%^&*()-_=+|[{}]:'",<.>/? are allowed. The password must contain at least two types of characters. **Suggestion:** You are advised to use the **get_input** function to obtain the value and avoid plaintext passwords to ensure security.

Relationships Between Elements

Table 5-38 Relationship description

Description	Target
Connected	**VPC.Subnet**
Connected	**VPC.SecurityGroup**
Inclusion	**VPC.VPC**

Return Value

Property	Type	Description
refIP	string	Access IP address of the DCS instance
refPort	integer	Access port of the DCS instance
refName	string	Name of the DCS instance
refID	string	ID of the DCS instance
chargeMode	string	Billing mode of the DCS instance

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  dcs-name:
    default: my-dcsinstance
  dcs-description:
    default: dcs service
  dcs-capacity:
    default: 2
  dcs-vpcId:
    default: fdcd13cf-579e-41d6-b2b5-01cda2f37719
  dcs-securityGroupId:
    default: 07f01d47-11fc-4b9b-bce3-f0f47350ad7a
  dcs-subnetId:
    default: 85786d98-06ed-4d33-a85c-572238649029
  dcs-password:
    default: "******"
  dcs-instanceMode:
    default: "single"
node_templates:
  my-dcs:
    type: Cloud.DCS.Redis
    properties:
      name: {get_input: dcs-name}
      description: {get_input: dcs-description}
      capacity: {get_input: dcs-capacity}
      vpcId: {get_input: dcs-vpcId}
      securityGroupId: {get_input: dcs-securityGroupId}
      subnetId: {get_input: dcs-subnetId}
      password: {get_input: dcs-password}
      instanceMode: {get_input: dcs-instanceMode}
```

5.2.11 ECS.CloudServer

Element Description

The **ECS.CloudServer** element is used to deploy an ECS at the IaaS layer. It consists of CPUs, memory, images, and EVS disks.

Element Properties

Table 5-39 Property Description

Property	Mandatory	Descripiton
vpcId	Yes	ID of the VPC to which the ECS belongs **Type:** Cloud.VPC.VPC.Id **Value Description:** Use the ID of an existing VPC or a new VPC. To use a new VPC ID, you need to define the VPC object in the template and establish the dependency. You are advised to drag the object to the VPC to automatically establish the dependency. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.VPC** element created by the stack. 3. Obtain the created VPC ID on the VPC console.
mountedVolumes	No	A shared disk can be attached to multiple ECSs, but a non-shared disk can be attached to only one ECS. **Type:** **ECS.MountedVolumes** Array **Value Description:** ECS.MountedVolumes array **Value Constraint:** ECS.MountedVolumes
imageId	Yes	ID of the image used by the ECS **Type:** Cloud.ECS.Image.Id **Value Description:** Indicates the system image of the to-be-created ECS. The ID of the created image must be specified. The ID format is UUID. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. You are advised to use the **get_input** function to assign values so that you can select a value when using the template. Search for the information in the ECS service documentation.
serverTags	No	Tags of an ECS **Type:** **ECS.ServerTags** array **Value Constraint:** One ECS can have up to 10 tags. The key of a tag can contain only uppercase and lowercase letters, digits, underscores (_), and hyphens (-). The value of a tag can contain only uppercase and lowercase letters, digits, underscores (_), hyphens (-), and periods (.).

Property	Mandatory	Descripiton
instances	Yes	Number of created ECSs **Type:** integer **Value Description:** The value ranges from 1 to 500. **Default:** 1 **Value Constraint:** The value ranges from 1 to 500. **Suggestion:** Set the value based on the live environment.
securityGroups	No	Array of the security group ID used by the cloud server **Type:** **ECS.SecurityGroup** array **Value Description:** ECS.SecurityGroup type array. **Value Constraint:** The value must meet the definition of the **ECS.SecurityGroup** type.
flavor	Yes	ECS specifications **Type:** Cloud.ECS.Flavor.Name **Value Description:** ID of the system specifications of the cloud server to be created **Value Constraint:** The definition of the flavor format is met. **Suggestion:** You are advised to use the **get_input** function to set values so that you can select a value when using the template. Alternatively, you can obtain the value through ECS documentations.
serverGroupId	No	ID of the cloud server group to which the host belongs **Type:** Cloud.ECS.ServerGroup.Id **Value Description:** Existing cloud server group ID of the current account **Value Constraint:** Existing cloud server group ID of the current account **Suggestion:** If you are adding this server to an existing cloud server group, specify the server group ID. If you are adding this server to a cloud server group created together with this server in the same template, use the **get_reference** function to automatically obtain the value.
nics	Yes	Information about the NIC of the ECS **Type:** **ECS.NICS** array **Value Description:** **ECS.NICS** type array **Value Constraint:** The value must comply with the definition of the **ECS.NICS** type. The minimum length of the array is 1 and the maximum is 12.

Property	Mandatory	Descripiton
rootVolume	Yes	System disk configuration of the ECS **Type:** **ECS.RootVolume** **Value Description: ECS.RootVolume** type **Default:** {u'volumeType': u'unset', u'size': 40} **Value Constraint:** The value must meet the definition of the **ECS.RootVolume** type.
userData	No	User data to be injected during ECS creation. Texts, text files, or GZIP files are supported. **Type:** string **Value Description:** Customize the value. **Value Constraint:** The content to be injected must be encoded using base64. The maximum size of the content to be injected before encoding is 32 KB. If **key_name** is not specified, the data injected by **user_data** is the password of the root user for logging in to the ECS by default. This parameter is mandatory when you create a Linux ECS using the password authentication mode. Its value is the initial password of the **root** user. **Suggestion:** Set this parameter based on live network. For more information about user data to be injected, see section "Injecting User Data into ECSs" of *Elastic Cloud Server User Guide*.
availabilityZone	Yes	AZ to which the ECS belongs **Type:** Cloud.ECS.AvailabilityZone.Name **Value Description:** AZ where the to-be-created ECS is located. The name of the AZ needs to be specified. **Value Constraint:** The value varies depending on regions. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.
dataVolumes	No	Data disk configuration of the ECS **Type:** **ECS.DataVolume** array **Value Description: ECS.DataVolume** type array **Value Constraint:** The value must meet the definition of the **ECS.DataVolume** type.

Property	Mandatory	Descripiton
name	Yes	ECS name **Type:** string **Value Description:** Customize the value, for example, myvm. **Value Constraint:** The value contains 1 to 64 characters. This value is unique under an account, and must meet regular expression **{"regex":"^[a-zA-Z][0-9a-zA-Z-_]*$","min_length":1,"max_length":64}**. **Suggestion:** Customize the value.
publicIP	No	Elastic IP address of the ECS **Type: ECS.PublicIP** **Value Description: ECS.PublicIP** type **Default:** {} **Value Constraint:** The value must meet the definition of the **ECS.PublicIP** type.
adminPwd	No	Initial login password of the administrator account for logging in to an ECS using password authentication. **Type:** password **Value Description:** The Linux administrator is **root**. The Windows administrator is **Administrator**. Set either a login key or login password. **Value Constraint:** 1. Enter 8 to 26 characters. The password must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters !@$%^-_=+[{}]:,./?. 2. The password cannot contain the username or the username in reverse. 3. The Windows ECS password cannot contain the username, the username in reverse, or more than two consecutive characters in the username. **Suggestion:** It is advised to set this parameter using the **get_input** function.

Property	Mandatory	Descripiton
sshKeyName	No	SSH key pair for login **Type:** Cloud.ECS.KeyPair.Name **Value Description:** It must be created in advance on the ECS console. Set either a login key or login password. **Value Constraint:** The value contains 1 to 64 characters. This value is unique under an account, and must meet regular expression **{"regex":"^[a-zA-Z][0-9a-zA-Z-_]*$","min_length":1,"max_length":64}**. **Suggestion:** 1. You are advised to use the **get_input** function to assign values so that you can select a value when using the template. 2. Obtain the information on the ECS console and then enter the information accordingly.

Relationships Between Elements

Table 5-40 Relationship description

Description	Target
Inclusion	**VPC.VPC**
Connected	**VPC.SecurityGroup**
Dependency	**SFS.FileSystem**
Connected	**VPC.Subnet**
Connected	**ECS.KeyPair**
Connected	**VPC.EIP**

Return Value

Property	Type	Description
publicIps	string	Elastic IP address array of an ECS instance
privateIps	string	Private IP address array of an ECS instance

Property	Type	Description
floatingIp Ids	string	Elastic IP address ID array of an ECS instance
refID	Array	List of all ECS instance IDs
refName	Array	List of all ECS instance names

Blueprint Example

The following uses the CloudServer resource orchestration blueprint as an example:

- Creating a subnet under the existing VPCs and subnets.

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  ecs-name:
    default: "my-cloudserver"
  ecs-image:
    default: "327946b5-e954-42c3-949a-3312688c9269"
  ecs-flavor:
    default: "c2.large"
  vpc-id:
    default: "ba6e4347-99d2-4649-b114-85c28d3d71b0"
  az:
    default: "az1.dc1"
  subnet-vpcid:
    default: "3be61f68-9bfc-41bf-8f5e-66c57122f270"
  ecs-volumetype:
    default: "SATA"
  ecs-sshKeyName:
    default: "KeyPair-magento"
node_templates:
  my-ecs:
    type: Cloud.ECS.CloudServer
    properties:
      name: {get_input: ecs-name}
      instances: 2
      imageId: {get_input: ecs-image}
      flavor: {get_input: ecs-flavor}
      vpcId: {get_input: vpc-id}
      availabilityZone: {get_input: az}
      nics:
        - subnetId: {get_input: subnet-vpcid}
      rootVolume:
        volumeType: {get_input: ecs-volumetype}
      dataVolumes:
        - volumeType: SATA
          size: 100
      sshKeyName: {get_input: ecs-sshKeyName}
```

- Associating with a VPC and subnet. CloudServer is automatically created under the newly created VPC and subnet.

  If you have not created a VPC or subnet, or you do not need to use an existing VPC or subnet, you can create a blueprint file and create a VPC, subnet, and CloudServer at the same time. When you create CloudServer, it can be automatically associated with the created VPC and subnet. The following is an example:

- Add the dependency requirements to the desired subnet. In this manner, the objects on which the subnet depends will be created during blueprint execution.
- For the **vpcId** property and its value on a subnet, use the **get_attribute** function to obtain the response attribute **refID** of the created VPC (my-vpc).
- For the **subnetId** property and its value in **vpcId** and **nics** on CloudServer, use the **get_attribute** function to obtain the response attribute **refID** of the created subnet (my-subnet).

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  ecs-name:
    default: "my-cloudserver"
  ecs-image:
    default: "327946b5-e954-42c3-949a-3312688c9269"
  ecs-flavor:
    default: "c2.large"
  ecs-volumetype:
    default: "SATA"
  az:
    default: "az1.dc1"
  subnet-name:
    default: "my-ecs-subnet2"
  subnet-cidr:
    default: "192.168.1.0/24"
  subnet-gateway:
    default: "192.168.1.1"
  vpc-name:
    default: "my-ecs-vpc2"
  vpc-cidr:
    default: "192.168.0.0/16"
node_templates:
  my-ecs:
    type: Cloud.ECS.CloudServer
    properties:
      name: {get_input: ecs-name}
      instances: 1
      imageId: {get_input: ecs-image}
      flavor: {get_input: ecs-flavor}
      vpcId: {get_attribute: [my-vpc, refID]}
      availabilityZone: {get_input: az}
      nics:
        - subnetId: {get_attribute: [my-subnet, refID]}
      rootVolume:
        volumeType: {get_input: ecs-volumetype}
      dataVolumes:
        - volumeType: SATA
          size: 100
    requirements:
      - nics.subnetId:
          node: my-subnet
    my-subnet:
    type: Cloud.VPC.Subnet
    properties:
      name: {get_input: subnet-name}
      cidr: {get_input: subnet-cidr}
      gateway: {get_input: subnet-gateway}
      dnsList: [114.114.114.115,114.114.114.114]
      vpcId: {get_attribute: [my-vpc, refID]}
      availabilityZone: {get_input: az}
    requirements:
      - vpcId:
          node: my-vpc
  my-vpc:
    type: Cloud.VPC.VPC
    properties:
```

```
name: {get_input: vpc-name}
cidr: {get_input: vpc-cidr}
```

5.2.12 ECS.KeyPair

Element Description

ECS.KeyPair is used to create a key pair for remote login authentication. To ensure security, you are advised to use the key authentication mode when logging in to an ECS.

Element Properties

Table 5-41 Property Description

Property	Mandatory	Descripiton
bucketName	Yes	Bucket name **Type:** string **Value Description:** Customize the value, for example, my-bucket. **Value Constraint:** {u'regex': u'^[a-z]([-a-z0-9]*[a-z0-9])?$', u'min_length': 3, u'max_length': 63} **Suggestion:** Customize the value.
name	Yes	Key pair name **Type:** string **Value Description:** Customize the value, for example, my-key. **Value Constraint:** {u'regex': u'^[-_a-zA-Z0-9]*$', u'min_length': 1, u'max_length': 63} **Suggestion:** Customize the value.

Relationships Between Elements

Table 5-42 Relationship description

Description	Target
Connected	**OBS.Bucket**

Return Value

Property	Type	Description
refName	string	Key pair name

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
node_templates:
  obsbozli:
    type: Cloud.OBS.Bucket
    properties:
      acl: private
  ecskp4ep:
    type: Cloud.ECS.KeyPair
    properties:
      name:
        get_input: ecskp4ep_name
      bucketName:
        get_reference: obsbozli
    requirements:
      - bucketName:
          node: obsbozli
inputs:
  ecskp4ep_name:
    description: keypair name
    label: ''
```

5.2.13 NAT.Instance

Element Description

The **NAT.Instance** element is used to create a NAT gateway instance.

Element Properties

Table 5-43 Property Description

Property	Mandatory	Descripiton
subnetId	Yes	ID of the subnet to which the NAT gateway belongs **Type:** Cloud.VPC.Subnet.Id **Value Description:** Obtains the subnet ID from the VPC service or connects to the **NAT.Subnet** to automatically generate a subnet ID. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Connect to a subnet and use the **get_reference** function to obtain the subnet ID.

Property	Mandatory	Descripiton
flavor	Yes	NAT gateway specifications **Type:** string **Default:** small **Value Constraint:** The value of this parameter must comply with the definition of NAT gateway specifications. **Suggestion:** The options are **small**, **middle**, **large**, and **xlarge**.
vpcId	Yes	ID of the VPC to which the NAT gateway belongs **Type:** Cloud.VPC.VPC.Id **Value Description:** Use the ID of an existing VPC or a new VPC. To use a new VPC ID, you need to define the VPC object in the template and establish the dependency. You are advised to drag the object to the VPC to automatically establish the dependency. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.VPC** element created by the stack. 3. Obtain the ID of the created VPC from the VPC console.
description	No	Description of the NAT gateway instance **Type:** string **Value Description:** Customize the value. **Value Constraint:** Enter a maximum of 255 characters. Only letters and digits are supported. **Suggestion:** Customize the value.
name	Yes	NAT name **Type:** string **Value Description:** Customize the value, for example, my-nat. **Value Constraint:** Enter 1 to 64 characters. The value must meet regular expression **[-_a-zA-Z0-9]*$**. **Suggestion:** Customize the value.

Relationships Between Elements

Table 5-44 Relationship description

Description	Target
Connected	VPC.Subnet
Inclusion	VPC.VPC

Return Value

Property	Type	Description
refName	string	NAT instance name
refID	string	NAT instance ID

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
node_templates:
  nat:
    type: Cloud.NAT.Instance
    properties:
      subnetId:
        get_input: nat_subnetId
      flavor: small
      vpcId:
        get_input: nat_vpcId
      name:
        get_input: nat_name
  snatrule:
    type: Cloud.NAT.SNatRule
    properties:
      subnetId:
        get_input: snatrule_subnetId
      floatingIpId:
        get_input: snatrule_floatingIpId
      natGatewayId:
        get_reference: nat
    requirements:
      - natGatewayId:
          node: nat
inputs:
  nat_subnetId:
    description: ID of the subnet to which the NAT gateway belongs
    label: ''
  nat_vpcId:
    description: ID of the VPC to which the NAT gateway belongs
    label: ''
  nat_name:
    description: NAT name
    label: ''
  snatrule_subnetId:
    description: ID of the subnet to which the source NAT rule belongs
    label: ''
  snatrule_floatingIpId:
```

```
description: ID of the user's elastic IP address
label: ''
```

5.2.14 NAT.SNatRule

Element Description

The **NAT.SNatRule** element is used to create a source NAT rule, which specifies the network segment for accessing the external network.

Element Properties

Table 5-45 Property Description

Property	Mandatory	Descripiton
subnetId	Yes	ID of the subnet to which the **SNatRule** belongs **Type:** Cloud.VPC.Subnet.Id **Description:** You can obtain a subnet ID from the VPC console or connect to the **SNatRule.Subnet** to automatically generate one. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Connect to a subnet and use the **get_reference** function to obtain the subnet ID.
floatingIpId	Yes	User EIP ID **Type:** Cloud.VPC.EIP.Id **Value Description:** Use the ID of an existing or new public elastic IP address. **Suggestion:** 1. Use the **get_attribute** function to obtain the ID of the elastic public IP address created by the template. 2. On the public elastic IP address page, obtain the ID of the created IP address.
natGatewayId	Yes	ID of the NAT gateway **Type:** string **Value Description:** Obtain the NAT gateway ID from the NAT service or put **SNatRule** in **NatGateWay** to obtain the NAT gateway ID. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Put **SNatRule** in **NatGateWay** and use the **get_reference** function to obtain the NAT gateway ID.

Relationships Between Elements

Table 5-46 Relationship description

Description	Target
Connected	VPC.Subnet
Inclusion	NAT.Instance
Connected	VPC.EIP

Return Value

Property	Type	Description
refID	string	NAT sNatRule ID

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
node_templates:
  nat:
    type: Cloud.NAT.Instance
    properties:
      subnetId:
        get_input: nat_subnetId
      flavor: small
      vpcId:
        get_input: nat_vpcId
      name:
        get_input: nat_name
  snatrule:
    type: Cloud.NAT.SNatRule
    properties:
      subnetId:
        get_input: snatrule_subnetId
      floatingIpId:
        get_input: snatrule_floatingIpId
      natGatewayId:
        get_reference: nat
    requirements:
      - natGatewayId:
          node: nat
inputs:
  nat_subnetId:
    description: ID of the subnet to which the NAT gateway belongs
    label: ''
  nat_vpcId:
    description: ID of the VPC to which the NAT gateway belongs
    label: ''
  nat_name:
    description: NAT name
```

```
  label: ''
snatrule_subnetId:
  description: ID of the subnet to which the source NAT rule belongs
  label: ''
snatrule_floatingIpId:
  description: ID of the user's elastic IP address
  label: ''
```

5.2.15 OBS.Bucket

Element Description

The **OBS.Bucket** element is used to deploy an OBS bucket. OBS provides massive, secure, highly reliable, and low-cost data storage capabilities. OBS buckets are used to store objects.

Element Properties

Table 5-47 Property Description

Property	Mandatory	Description
location	No	Region where the OBS bucket is located **Type:** string **Value Description:** You can set it to **ae-ad-1**. **Suggestion:** You are not advised to set the value. The system automatically allocates the value to the current region.
name	No	OBS bucket name **Type:** string **Value Description:** Customize the value, for example, my-bucket. **Value Constraint:** The value must be globally unique. It contains 3 to 63 characters and must meet regular expression **^[a-z]([-a-z0-9]*[a-z0-9])?$**. **Suggestion:** Customize the value.
acl	Yes	Permission control policy of an OBS bucket **Type:** string **Value Description:** The options are **private**, **public-read**, and **public-read-write**. **Default:** private **Value Constraint:** The options are **private**, **public-read**, and **public-read-write**.

Table 5-48 Description of pre-defined permission control policies in OBS

Pre-defined Permission Control Policy	Description
private	Owner of a bucket or object has the **FULL_CONTROL** permission for the bucket or object. Other users have no permission to access the bucket or object.
public-read	Owner of a bucket or object has the **FULL_CONTROL** permission for the bucket or object. Other users including anonymous users have the **READ** permission.
public-read-write	Owner of a bucket or object has the **FULL_CONTROL** permission for the bucket or object. Other users including anonymous users have the **READ** and **WRITE** permissions.

Relationships Between Elements

None.

Return Value

Property	Type	Description
refName	string	Bucket name

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  bucket-name:
    default: my-first-bucket
  bucket-acl:
    default: public-read
  bucket-location:
    default: ae-ad-1
node_templates:
  my-bucket:
    type: Cloud.OBS.Bucket
    properties:
      name: {get_input: bucket-name}
      acl: {get_input: bucket-acl}
      location: {get_input: bucket-location}
```

5.2.16 RDS.MySQL

Element Description

Relational Database Service (RDS) is a cloud-based web service that is reliable, scalable, easy to manage, and ready to use out-of-the-box.

RDS provides an optimized performance monitoring system, multiple security protection measures, and a professional database management platform, helping you easily configure, operate, and expand the relational database. On the RDS console, you can execute all necessary tasks without programming, which simplifies the operation process and reduces routine O&M workload. Therefore, you can focus on application development and service development.

Element Properties

Table 5-49 Property Description

Property	Mandatory	Descripiton
dbPort	No	Port for accessing the instance **Type:** integer **Value Description:** The value ranges from 2100 to 9500. Currently, this field is invalid. **Default:** 3306 **Constraint:** {u'in_range': [2100, 9500]} **Suggestion:** Set the value within the port range based on requirements.
availabilityZone	Yes	AZ where the instance is located **Type:** Cloud.ECS.AvailabilityZone.Name **Value Description:** AZ where the to-be-created ECS is located. The name of the AZ needs to be specified, for example, ae-ad-1a. **Value Constraint:** The value varies depending on regions. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.
name	No	Instance name **Type:** string **Value Description:** Customize the value. **Default:** "" **Value Constraint:** Enter 4 to 64 characters, starting with a letter. Only letters (case-insensitive), digits, hyphens (-), and underscores (_) are allowed. 2. The name of an instance of the same type must be unique under the same account. **Suggestion:** Customize the value.

Property	Mandatory	Descripiton
dataBase	No	Configuration of the database of the instance **Type:** MySQL.DataBase **Default:** {u'characterSet': u'utf8', u'name': u'unset', u'collate': u'utf8_general_ci'} **Suggestion:** Select the **dataBase** field in the component part, and then fill in the field based on prompts.
paramsGroupId	No	Parameters group ID of an instance **Type:** Cloud.RDS.ParamsGroup.Id **Suggestion:** It is advised to set it to a **get_input** function and select it from a drop-down list. You can also fill in a default parameter group ID which needs to be obtained from the RDS console.
securityGroupId	Yes	ID of the security group to which the instance belongs **Type:** Cloud.VPC.SecurityGroup.Id **Value Description:** Obtain the security group ID from the VPC service or connects to the **VPC.SecurityGroup** to automatically generate a security group ID. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.SecurityGroup** element created by the stack. 3. Obtain the ID of the created security group from the VPC console.
dbUser	No	Configuration of the user of the instance **Type:** MySQL.DBUser **Default:** {u'userPassword': u'unset', u'name': u'unset'} **Suggestion:** Select the **dbUser** field in the component part, and then fill in the field based on prompts.

Property	Mandatory	Descripiton
dbRootPassword	Yes	Password of the **root** user of the instance. This parameter cannot be left blank. The password must not be a weak password. Enter 8 to 32 characters. Only letters, digits, and special characters ~!@#%^*-_=+? are allowed. **Type:** password **Value Description:** Customize the value. **Value Constraint:** 1. The parameter must be written into inputs and set using the **get_input** function. 2. This parameter cannot be left blank. The password must not be a weak password. Enter 8 to 32 characters. Only letters, digits, and special characters ~!@#%^*-_=+? are allowed. **Suggestion:** You are advised to use the **get_input** function to obtain the value and avoid plaintext passwords to ensure security.
volume	Yes	Information about the data disk used by the instance **Type:** RDS.Volume **Default:** {u'volumetype': u'COMMON', u'size': 100} **Suggestion:** Select the **volume** field in the component part, and then fill in the field based on prompts.
timeZone	No	Time zone where the instance locates. This parameter is supported only in stacks billed in the yearly/monthly mode. Stacks billed in pay-per-use mode do not support setting this parameter. **Type:** string **Value Description**: If this parameter is specified, the value ranges from UTC-12:00 to UTC+12:00 at the full hour. For example, the value can be UTC+08:00 rather than UTC+08:30. **Value Constraint:** {u'valid_values': [u'UTC-12:00', u'UTC-11:00', u'UTC-10:00', u'UTC-09:00', u'UTC-08:00', u'UTC-07:00', u'UTC-06:00', u'UTC-05:00', u'UTC-04:00', u'UTC-03:00', u'UTC-02:00', u'UTC-01:00', u'UTC', u'UTC+01:00', u'UTC+02:00', u'UTC+03:00', u'UTC+04:00', u'UTC+05:00', u'UTC+06:00', u'UTC+07:00', u'UTC+08:00', u'UTC+09:00', u'UTC+10:00', u'UTC+11:00', u'UTC+12:00']} **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.

Property	Mandatory	Descripiton
backupStrategy	Yes	Backup policy of the instance **Type:** RDS.BackupStrategy **Default:** {u'keepDays': 0, u'endTime': u'02:00', u'startTime': u'01:00'} **Value Constraint:** Set the value based on specifications.
subnetId	Yes	ID of the subnet to which the instance belongs **Type:** Cloud.VPC.Subnet.Id **Value Description:** Use the ID of an existing subnet or a new subnet. To use a new subnet ID, you need to define the subnet object in the template and establish the dependency. You are advised to connect the **VPC.Subnet** to automatically establish the dependency. **Value Constraint:** The subnet must correspond to the VPC. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.Subnet** element created by the stack. 3. Obtain the ID of the created subnet from the VPC console.
slaveAvailabilityZone	No	AZ where the standby HA instance is located **Type:** Cloud.ECS.AvailabilityZone.Name **Value Description:** AZ where the to-be-created ECS is located. The name of the AZ needs to be specified, for example, ae-ad-1a. **Value Constraint:** The value varies depending on regions. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.
dataStore	Yes	Database information **Type:** MySQL.DataStore **Default:** {u'dbtype': u'MySQL', u'version': u'5.7'} **Suggestion:** Select the **dataStore** field in the component part, and then fill in the field based on prompts.

Property	Mandatory	Descripiton
HA	Yes	HA configuration of the instance **Type:** RDS.HA.Mysql **Default:** {u'replicationMode': u'semisync', u'enable': u'unset'} **Suggestion:** Select the **HA** field in the component part, and then fill in the field based on prompts.
vpcId	Yes	ID of the VPC to which the instance belongs **Type:** Cloud.VPC.VPC.Id **Value Description:** Use the ID of an existing VPC or a new VPC. To use a new VPC ID, you need to define the VPC object in the template and establish the dependency. You are advised to drag the object to the **VPC** to automatically establish the dependency. **Value Constraint:** The value must satisfy the UUID generation rule. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Use the **get_reference** function to obtain the **VPC.VPC** element created by the stack. 3. Obtain the ID of the created VPC from the VPC console.
flavor	Yes	Instance specifications **Type:** Cloud.RDS.Flavor.Id **Value Description:** Flavor ID of the to-be-created database instance, which is generated based on the instance specifications and user project. **Value Constraint:** Flavor IDs vary depending on project. The property must match the database type and version. In a resource specification code, for example, **rds.mysql.m1.xlarge**, **rds** indicates the RDS service. **mysql** indicates the database engine. **m1.xlarge** indicates high memory, a performance specification. If a code contains **rr**, it indicates that this is a read-only instance. If a code does not contain **rr**, it indicates that this is a single or HA database instance. **Suggestion:** You are advised to obtain the value by using the RDS APIs.

Relationships Between Elements

Table 5-50 Relationship description

Description	Target
Connected	VPC.Subnet
Connected	VPC.SecurityGroup
Inclusion	VPC.VPC

Return Value

Property	Type	Description
refIP	string	Access IP address of the RDS MySQL instance
refPort	integer	Access port of the RDS MySQL instance
refName	string	Name of the RDS MySQL instance
refID	string	ID of the RDS MySQL instance
chargeMode	string	Billing mode of the RDS MySQL instance

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
node_templates:
  rdsms528:
    type: Cloud.RDS.MySQL
    properties:
      dataStore:
        dbtype: MySQL
        version: '5.7'
      dbPort: 3306
      vpcId:
        get_input: rdsms528_vpcId
      securityGroupId:
        get_input: rdsms528_securityGroupId
      availabilityZone:
        get_input: rdsms528_availabilityZone
      dbRootPassword:
        get_input: rdsms528_dbRootPassword
      volume:
        volumetype: COMMON
        size: 100
      backupStrategy:
        keepDays: 0
        endTime: '02:00'
        startTime: '01:00'
      subnetId:
        get_input: rdsms528_subnetId
```

```
      flavor:
        get_input: rdsms528_flavor
      HA:
        replicationMode: semisync
        enable:
          get_input: rdsms528_HA_enable
inputs:
  rdsms528_vpcId:
    description: ID of the VPC to which the instance belongs
    label: ''
  rdsms528_securityGroupId:
    description: ID of the security group to which the instance belongs
    label: ''
  rdsms528_availabilityZone:
    description: AZ to which the instance belongs
    label: ''
  rdsms528_dbRootPassword:
    description: 'Password of the root user of the instance. The password must be 8 to 32 characters long
and cannot be a weak password. Only letters, digits, and special characters ~!@#%^*-_=+? are allowed.'
    label: ''
  rdsms528_subnetId:
    description: ID of the subnet to which the instance belongs
    label: ''
  rdsms528_flavor:
    description: Instance specifications
    label: ''
  rdsms528_HA_enable:
    description: Whether HA is supported
    label: ''
```

5.2.17 SFS.FileSystem

Element Description

SFS provides high-performance file storage which supports on-demand scaling. It can be shared by multiple ECSs.

Element Properties

Table 5-51 Property Description

Property	Mandatory	Descripiton
size	Yes	Storage space size (unit: GB). The minimum value is 1 and the maximum value is 511800. **Type:** integer **Value Description:** The value ranges from 1 to 511800. **Default:** 1 **Value Constraint:** [1, 511800]

Property	Mandatory	Descripiton
vpcId	Yes	ID of the belonged VPC. Only ECSs in the VPC can access the SFS file system. **Type:** Cloud.VPC.VPC.Id **Value Description:** Use the ID of an existing VPC or a new VPC. **Value Constraint:** The value must satisfy the UUID generation rule. **Suggestion:** 1. Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console. 2. Obtain the ID of the created VPC from the VPC console.
description	Yes	Shared description **Type:** string **Value Description:** Customize the value. **Default:** "" **Value Constraint:** [0, 255]
name	Yes	SFS instance name **Type:** string **Value Description:** Customize the value. **Default:** "" **Value Constraint:** [0, 255]
availabilityZone	Yes	AZ to which the file system belongs **Type:** Cloud.ECS.AvailabilityZone.Name **Value Description:** AZ where the to-be-shared file system is located. The name of the AZ needs to be specified. **Value Constraint:** The value varies depending on regions.
accessLevel	Yes	Permission level of the shared access **Type:** string **Value Description:** Customize the value. **Default:** rw **Value Constraint:** The options are **ro** and **rw**. **ro** indicates read-only, and **rw** indicates read and write.

Relationships Between Elements

Table 5-52 Relationship description

Description	Target
Inclusion	VPC.VPC

Return Value

Property	Type	Description
ShareAccessId	string	UUID of a share access rule
export_location	string	Sharing path
refID	string	SFS ID
export_locations	string	Sharing path

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  name:
    default: my-sfs
  availabilityZone:
    type: Cloud.ECS.AvailabilityZone.Name
  vpcId:
    type: Cloud.VPC.VPC.Id
  accessLevel:
    default: "ro"
  size:
    default: 10
node_templates:
  my-sfs:
    type: Cloud.SFS.FileSystem
    properties:
      name: {get_input: name}
      size: {get_input: size}
      availabilityZone: {get_input: availabilityZone}
      accessLevel: {get_input: accessLevel}
      vpcId: {get_input: vpcId}
```

5.2.18 ULB.Healthmonitor

Element Description

The **ULB.Healthmonitor** element is a health check component of a shared load balancer. One pool corresponds to one health monitor. One health monitor can manage multiple ECSs. You can add or delete health monitors as required.

Element Properties

Table 5-53 Property Description

Property	Mandatory	Descripiton
monitorPort	No	Health check port **Type:** integer **Value Description:** Customize the value. The value is an integer between 1 and 65535, for example, 8089. If this parameter is left blank, the backend port of the ECS is used by default. **Value Constraint:** The value ranges from 1 to 65535. **Suggestion:** Set the value based on the live environment.
name	No	Name of a health check job **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value supports a maximum of 64 characters and can only contain digits, letters, underscores (_), and hyphens (-). **Suggestion:** Customize the value.
urlPath	No	URI for health check. This parameter is valid when **type** is set to **HTTP**. You are advised to perform check on the static page. **Type:** string **Value Description:** Customize the value, for example, / or /index.html. **Value Constraint:** The value contains 1 to 80 characters and must start with a slash (/). Only letters, digits, and special characters -/.%?#&_= are allowed. It must meet regular expression **^/[0-9a-zA-Z-_?/.?#&=]***. **Suggestion:** Set the value based on the live environment.

Property	Mandatory	Descripiton
delay	Yes	Interval for health check (unit: s) **Type:** integer **Value Description:** Customize the value. The value is an integer ranging from 0 to 2147483647, for example, 5. **Default:** 5 **Value Constraint:** The value is an integer ranging from 0 to 2147483647. **Suggestion:** Set the value based on the live environment.
httpMethod	No	HTTP method for health check. This parameter is valid when **type** is set to **HTTP**. **Type:** string **Value Description:** The options are **GET**, **HEAD**, **POST**, **PUT**, **DELETE**, **TRACE**, **OPTIONS**, **CONNECT**, and **PATCH**. **Value Constraint:** The options are **GET**, **HEAD**, **POST**, **PUT**, **DELETE**, **TRACE**, **OPTIONS**, **CONNECT**, and **PATCH**. **Suggestion:** Set the value based on the live environment.
timeout	Yes	Maximum timeout duration for health check (unit: s) **Type:** integer **Value Description:** Customize the value. The value is an integer ranging from 0 to 2147483647, for example, 10. **Default:** 10 **Value Constraint:** The value is an integer ranging from 0 to 2147483647. **Suggestion:** Set the value based on the live environment.
poolId	Yes	ECS group ID **Type:** string **Value Description:** ECS group ID **Suggestion:** Use the **get_reference** function to automatically generate the value.

Property	Mandatory	Descripiton
maxRetries	Yes	The number of consecutive times a cloud server fails or passes health checks, after which the health check status of the backend cloud server changes from **success** to **fail** or from **fail** to **success**, respectively. **Type:** integer **Value Description:** Customize the value. The value is an integer between 1 and 10, for example, 3. **Default:** 3 **Value Constraint:** The value is an integer between 1 and 10. **Suggestion:** Set the value based on the live environment.
expectedCode	No	HTTP status code used to determine the health status of a backend ECS. This parameter is valid when **type** is set to **HTTP**. **Type:** string **Value Description:** Customize the value, for example, 200. **Value Constraint:** The value ranges from 1 to 250. **Suggestion:** Set the value based on the live environment.
type	Yes	Health check protocol **Type:** string **Value Description:** The options are **HTTP**, **TCP**, **HTTPS**, **PING**, and **TLS-HELLO**. **Value Constraint:** The options are **HTTP**, **TCP**, **HTTPS**, **PING**, and **TLS-HELLO**. **Suggestion:** Set the value based on the live environment.

Relationships Between Elements

Table 5-54 Relationship description

Description	Target
Inclusion	**ULB.Pool**

Return Value

Property	Type	Description
refID	string	ID of a health check instance

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  pool_protocol:
    description: 'Cloud server group protocol. It must be the same as the listener protocol.'
  pool_listenerId:
    description: ID of the listener to which it belongs
  pool_lbAlgorithm:
    description: Allocation policy type
  delay:
    description: Interval for health check (unit: s)
  timeout:
    description: Maximum timeout duration for health check (unit: s)
  max_retries:
    description: The number of consecutive times a cloud server fails or passes health checks, after which the
health check status of the backend cloud server changes from success to fail or from fail to success,
respectively.
  type:
    description: Health check protocol
node_templates:
  pool:
    type: Cloud.ULB.Pool
    properties:
      protocol:
        get_input: pool_protocol
      listenerId:
        get_input: pool_listenerId
      lbAlgorithm:
        get_input: pool_lbAlgorithm
  health-monitor:
    type: Cloud.ULB.Healthmonitor
    properties:
      delay:
        get_input: delay
      timeout:
        get_input: timeout
      maxRetries:
        get_input: max_retries
      type:
        get_input: type
      poolId:
        get_reference: pool
    requirements:
      - poolId:
          node: pool
```

5.2.19 ULB.Listener

Element Description

The **ULB.Listener** element indicates the listener under a shared load balancer. One shared load balancer corresponds to multiple listeners. You can add or delete listeners as required.

Element Properties

Table 5-55 Property Description

Property	Mandatory	Descripiton
protocol	Yes	Listening protocol **Type:** string **Value Description:** This value can be **TCP** or **HTTP**. **Value Constraint:** This value can be **TCP** or **HTTP**. **Suggestion:** Set the value based on the live environment.
description	No	Description **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value supports a maximum of 255 characters. **Suggestion:** Customize the value.
connectionLimit	No	Maximum number of connections of the listener **Type:** integer **Value Description:** If the number of connections is **-1**, there is no constraints. **Value Constraint:** The value ranges from -1 to 2147483647. **Suggestion:** Set the value based on the live environment.
loadBalancerId	Yes	ID of the belonged ULB **Type:** string **Value Description:** ID generated after a ULB instance is created, for example, 8abbd7a9-c1f8-440d-96ff-376ee7382082. **Value Constraint:** The ID must be the ID of an existing ULB instance. **Suggestion:** You are advised to drag the object to the **ULB.LoadBalancer** and use the **get_reference** function to automatically generate the value. Alternatively, obtain the ULB instance ID on the ULB console and enter it accordingly.

Property	Mandatory	Descripiton
port	Yes	Listening port **Type:** integer **Value Description:** The value ranges from 1 to 65535. **Value Constraint:** The value ranges from 1 to 65535. **Suggestion:** Set the value based on the live environment.
name	No	Listener name **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value supports a maximum of 64 characters and can only contain digits, letters, underscores (_), and hyphens (-). **Suggestion:** Customize the value.

Relationships Between Elements

Table 5-56 Relationship description

Description	Target
Inclusion	**ULB.LoadBalancer**

Return Value

Property	Type	Description
refName	string	Listener instance name
refID	string	Listener instance ID

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  listener_protocol:
    description: Listening protocol
    label: ''
  listener_port:
    description: Listening port
    label: ''
  listener_loadBalancerId:
    description: ID of the belonged ULB
    label: ''
node_templates:
```

```
listener:
  type: Cloud.ULB.Listener
  properties:
    protocol:
      get_input: listener_protocol
    port:
      get_input: listener_port
    loadBalancerId:
      get_input: listener_loadBalancerId
```

5.2.20 ULB.LoadBalancer

Element Description

The **ULB.LoadBalancer** element can be used to deploy a shared load balancer resource object at the PaaS layer. By creating such an object, you can provide a unified entry address for a group of containerized applications with the same functions, and distribute requests in load balancing mode to backend container applications. Shared load balancers are applicable to web services with high access traffic. They forward requests based on domain names or URLs, making request routing more flexible. Compared with classic load balancers, shared load balancers provide stronger HTTP and HTTPS forwarding capabilities, and better forwarding performance and stability.

Element Properties

Table 5-57 Property Description

Property	Mandatory	Descripiton
vipAddress	No	IP address of the VPC where the shared load balancer is located **Type:** ip **Value Description:** IP address that is not used in the selected subnet **Value Constraint:** The value must be an IP address.
description	No	Description **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value supports a maximum of 255 characters. **Suggestion:** Customize the value.

Property	Mandatory	Descripiton
publicIpId	No	ID of the elastic IP address that can be bound to the shared load balancer **Type:** string **Value Description:** ID of the elastic IP address that can be bound to the VPC. **Suggestion:** Query the binding status and ID of the elastic IP address on the elastic IP address page of the VPC service.
subnetId	Yes	ID of the subnet that allocates VIP addresses to the shared load balancer **Type:** Cloud.VPC.Subnet.All.Id **Value Description:** ID of the subnet of the VPC **Value Constraint:** You can view the subnet ID in the VPC details page. **Suggestion:** Drag the object to **VPC.Subnet** and use **{get_attribute: [*element name*, *neutron_subnet_id*]}** to automatically generate the value.
name	No	Name of the shared load balancer **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value supports a maximum of 64 characters and can only contain digits, letters, underscores (_), and hyphens (-). **Suggestion:** Customize the value.

Relationships Between Elements

Table 5-58 Relationship description

Description	Target
Connected	**VPC.Subnet**
Connected	**VPC.EIP**

Return Value

Property	Type	Description
vip_port_id	string	**PORT_ID** of the VPC where the shared load balancer is located
refName	string	Name of the shared load balancer
refID	string	ID of the shared load balancer
vip_address	string	IP address of the VPC where the shared load balancer is located

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  subnetId:
    description: ID of the subnet that allocates VIP addresses to the shared load balancer. It is the subnet ID
rather than the subnet network ID.
node_templates:
  ulb:
    properties:
      description: ulb load balancer
      subnetId:
        get_input: subnetId
    type: Cloud.ULB.LoadBalancer
```

5.2.21 ULB.Member

Element Description

For ECSs under a shared load balancer, one pool corresponds to multiple ECSs. You can add or delete ECSs as required.

Element Properties

Table 5-59 Property Description

Property	Mandatory	Descripiton
weight	No	Weight of an ECS, which determines the proportion of requests to be forwarded compared with other members in the same ECS group **Type:** integer **Value Description:** Customize the value. The value is an integer between 1 and 256, for example, 3. **Value Constraint:** {u'in_range': [0, 256]} **Suggestion:** Set the value based on the live environment.

Property	Mandatory	Descripiton
address	No	Private IP address of the backend ECS added to the listener **Type:** ip array **Value Description:** private network IP address generated after an ECS is created, for example, 192.168.0.45 **Value Constraint:** The IP address must be the private network IP address of the existing ECS instance. The ECS and listener must be in the same subnet. Set either **address** or **serverId**. **Suggestion:** You are advised to drag the object to the **ECS.CloudServer** and use **{get_attribute: [*ECS element*, privateIps]}** to automatically generate the value. Alternatively, obtain the private network IP address on the ECS console and enter it accordingly.
poolId	Yes	ID of the ECS group to which the ECS is to be added **Type:** string **Value Description:** ID of the ECS group to which the ECS is to be added **Suggestion:** Use the **get_reference** function to automatically generate the value.
subnetId	Yes	ID of the subnet where the ECS and listener are located **Type:** Cloud.VPC.Subnet.All.Id **Value Description:** ID of the subnet of the VPC **Value Constraint:** The subnet ID must be the same as that in the listener. **Suggestion:** Drag the object to **VPC.Subnet** and use **{get_attribute: [*element name*, neutron_subnet_id]}** to automatically generate the value. Alternatively, obtain the subnet ID on the VPC details page.
serverId	No	ID of the backend ECS added to the listener **Type:** string array **Value Description:** ID generated after an ECS is created, for example, b7a65ad3-c031-43cc-93ac-ac6dbdbd2295. **Value Constraint:** The ID must be the ID of an existing ECS instance. The ECS and listener must be in the same subnet. Set either **address** or **serverId**. **Suggestion:** You are advised to drag the object into **ECS.CloudServer** and use **{get_attribute: [*ECS element*, *refID*]}** to automatically generate the value. Alternatively, search for the ID on the ECS console and enter it accordingly.

Property	Mandatory	Descripiton
port	Yes	Backend port of the ECS **Type:** integer **Value Description:** Customize the value. The value is an integer between 1 and 65535, for example, 8089. **Value Constraint:** {u'in_range': [1, 65535]} **Suggestion:** Set the value based on the live environment.

Relationships Between Elements

Table 5-60 Relationship description

Description	Target
Connected	**VPC.Subnet**
Connected	**ECS.CloudServer**
Inclusion	**ULB.Pool**

Return Value

Property	Type	Description
refID	string	Backend ECS instance ID
poolId	string	ID of the ECS group to which the backend ECS belongs

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  pool_protocol:
    description: 'ECS group protocol, which must be consistent with the listener protocol'
  pool_listenerId:
    description: Belonged listener ID
  pool_lbAlgorithm:
    description: Allocation policy type
  delay:
    description: Interval for health check (unit: s)
  timeout:
    description: Maximum timeout duration for health check (unit: s)
  max_retries:
    description: The number of consecutive times a cloud server fails or passes health checks, after which the
health check status of the backend cloud server changes from success to fail or from fail to success,
```

```
respectively.
  type:
    description: Health check protocol
  subnetId:
    description: ID of the subnet to which the ECS and listener belong. It is the subnet ID rather than the
subnet network ID.
  address:
    description: Private IP address of the backend ECS added to the listener
  port:
    description: Backend port of the ECS
node_templates:
  pool:
    type: Cloud.ULB.Pool
    properties:
      protocol:
        get_input: pool_protocol
      listenerId:
        get_input: pool_listenerId
      lbAlgorithm:
        get_input: pool_lbAlgorithm
  health-monitor:
    type: Cloud.ULB.Healthmonitor
    properties:
      delay:
        get_input: delay
      timeout:
        get_input: timeout
      maxRetries:
        get_input: max_retries
      type:
        get_input: type
      poolId:
        get_reference: pool
    requirements:
      - poolId:
          node: pool
  member:
    type: Cloud.ULB.Member
    properties:
      subnetId:
        get_input: subnetId
      address:
        - get_input: address
      port:
        get_input: port
      poolId:
        get_reference: pool
    requirements:
      - poolId:
          node: pool
```

5.2.22 ULB.Pool

Element Description

For ECS groups under a shared load balancer, one listener corresponds to multiple ECS groups. You can add or delete ECS groups as required. An ECS group consists of multiple ECSs.

Element Properties

Table 5-61 Property Description

Property	Mandatory	Descripiton
sessionPersistence	No	Session persistence setting **Type:** ULB.StickySession **Value Description:** If this option is selected, the session persistence function is enabled by default. **Default:** {u'type': u'SOURCE_IP'} **Suggestion:** Set the value based on the live environment.
protocol	Yes	ECS group protocol **Type:** string **Value Description:** The options are **HTTP** and **TCP**. **Value Constraint:** The value must be consistent with the listener protocol. **Suggestion:** Set the value based on the live environment.
name	No	ECS group name **Type:** string **Value Description:** Customize the value. **Value Constraint:** The value supports a maximum of 64 characters and can only contain digits, letters, underscores (_), and hyphens (-). **Suggestion:** Customize the value.
lbAlgorithm	Yes	Allocation policy type **Type:** string **Value Description: ROUND_ROBIN**: indicates the weighted round robin algorithm. **LEAST_CONNECTIONS**: indicates the weighted least connection. **SOURCE_IP**: indicates the source IP algorithm. **Default:** ROUND_ROBIN **Value Constraint:** The options are **ROUND_ROBIN**, **LEAST_CONNECTIONS**, and **SOURCE_IP**. **Suggestion:** Set the value based on the live environment.

Property	Mandatory	Descripiton
listenerId	Yes	ID of the belonged listener **Type:** string **Value Description:** ID generated after a ULB instance is created, for example, 8abbd7a9-c1f8-440d-96ff-376ee7382082. **Value Constraint:** The ID must be the listener ID of an existing ULB instance. **Suggestion:** You are advised to drag the object to the **ULB.Listener** and use the **get_reference** function to automatically generate the value. Alternatively, obtain the ULB listener ID on the ULB console and enter it accordingly.

Relationships Between Elements

Table 5-62 Relationship description

Description	Target
Inclusion	**ULB.Listener**

Return Value

Property	Type	Description
refID	string	ECS group instance ID

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  pool_protocol:
    description: ECS group protocol, which must be consistent with the listener protocol
  pool_listenerId:
    description: Belonged listener ID
  pool_lbAlgorithm:
    description: Allocation policy type
node_templates:
  pool:
    type: Cloud.ULB.Pool
    properties:
      protocol:
        get_input: pool_protocol
      listenerId:
        get_input: pool_listenerId
      lbAlgorithm:
        get_input: pool_lbAlgorithm
```

5.2.23 VPC.EIP

Element Description

VPC.EIP is used to create a public elastic IP address. A public elastic IP address is a static IP address. You can bind or unbind an elastic IP address to an Elastic Cloud Server (ECS) in a subnet. An ECS in a Virtual Private Cloud (VPC) can access the Internet through a fixed public IP address.

Element Properties

Table 5-63 Property description

Property	Mandatory	Description
publicIP	Yes	Elastic IP address object **Type**: VPC.PublicIP **Default value:** {u'type': u'unset'}
instances	No	Number of elastic IP addresses **Type:** integer **Value Description:** Customize the value. The default value is **1**. **Default Value:** 1 **Constraint:** {u'greater_or_equal': 1}
bandwidth	Yes	Bandwidth object **Type**: VPC.BandWidth **Default value**: {u'shareType': u'PER'}

Relationships Between Elements

Table 5-64 Relationship description

Description	Target
Dependency	CCE.NodePool
Dependency	ECS.CloudServer

Return Value

Property	Type	Description
refIP	string	Elastic IP address created
idList	string	ID of the elastic IP addresses created in batch
refID	string	ID of the elastic IP address.
ipList	string	Elastic IP addresses created in batch

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  publicip-type:
    default: 5_bgp
    description: Public IP type.
  size:
    default: 1
    description: bandwidth size
node_templates:
  eip:
    properties:
      bandwidth:
        name: test-eip
        shareType: PER
        size:
          get_input: size
      publicIP:
        type:
          get_input: publicip-type
    type: Cloud.VPC.EIP
```

5.2.24 VPC.SecurityGroup

Element Description

A security group (a logical group) is a collection of access control policies for ECSs that have the same security protection requirements and are mutually trusted in a VPC.

Element Properties

Table 5-65 Property Description

Property	Mandatory	Descripiton
name	No	Security group name **Type:** string **Value Description:** Customize the value, for example, my-securitygroup. **Value Constraint:** Enter 1 to 64 characters. Only digits, letters, underscores (_), hyphens (-), and periods (.) are allowed. **Suggestion:** Customize the value.

Relationships Between Elements

Table 5-66 Relationship description

Description	Target
Inclusion	VPC.VPC

Return Value

Property	Type	Description
refID	string	Security group instance ID
refName	string	Security group instance name

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  sg-name:
    default: my-security-group
node_templates:
  my-sg:
    type: Cloud.VPC.SecurityGroup
    properties:
      name:
        get_input: sg-name
outputs:
  sg-id:
    value:
      get_attribute: [my-sg, refID]
```

5.2.25 VPC.SecurityGroupRule

Element Description

A security group rule is an access policy added for an ECS to implement access control.

Element Properties

Table 5-67 Property Description

Property	Mandatory	Descripiton
direction	Yes	Ingress or egress control direction (that is, ingress or egress) **Type:** string **Value Description:** The options are **egress** and **ingress**. **Default:** ingress **Value Constraint:** {u'valid_values': [u'egress', u'ingress']}
protocol	No	Protocol type **Type:** string **Value Description:** The options are **ICMP**, **TCP**, and **UDP**. If this property is left blank, all protocols are supported. **Value constraint:** {u'valid_values': [u'ICMP', u'TCP', u'UDP']}
remoteSecurityGroupId	No	Peer security group ID **Type:** Cloud.VPC.SecurityGroup.Id **Value Description:** Obtain the security group ID from the VPC service or automatically generate it through **VPC.SecurityGroup**. **Value Constraint:** The value of this parameter and the value of **remoteIpPrefix** are mutually exclusive. **Suggestion:** It is advised to obtain the ID of a **SecurityGroup** object using **get_input** or **get_reference**.
ethertype	No	Protocol type of the IP address **Type:** string **Value Description:** Set it to **IPv4**. **Default:** IPv4 **Value constraint:** {u'valid_values': [u'IPv4']}

Property	Mandatory	Descripiton
securityGroupId	Yes	ID of the security group the resource belongs **Type:** Cloud.VPC.SecurityGroup.Id **Value Description:** Obtain the security group ID from the VPC service or connects to the **VPC.SecurityGroup** to automatically generate a security group ID. **Value Constraint:** The value must meet the UUID generation rule and be the ID of an existing security group of the tenant. **Suggestion:** You are advised to use the **get_input** function to obtain the value, or connect the **SecurityGroup** object and use the **get_reference** function to automatically generate the value.
remoteIpPrefix	No	Remote IP address **Type:** string **Value Description:** When the direction is egress, it is the address of the terminal that accesses the VM. When the direction is ingress, it is the address of the to-be-accessed VM. **Value Constraint:** The value can be in the CIDR format or an IP address. The value of this parameter and the value of **remoteSecurityGroup** are mutually exclusive.
maxPort	No	Destination port number **Type:** integer **Value Description:** Customize the value. The value ranges from 1 to 65535. **Value Constraint:** {u'in_range': [1, 65535]} **Suggestion:** If the protocol is not ICMP, the value cannot be smaller than the value of **minPort**. When **minPort** and **maxPort** are left blank, all port numbers are supported.
minPort	No	Start port number **Type:** integer **Value Description:** Customize the value. The value ranges from 1 to 65535. **Value Constraint:** {u'in_range': [1, 65535]} **Suggestion:** The value cannot be greater than the value of **maxPort**. When **minPort** and **maxPort** are left blank, all port numbers are supported.

Relationships Between Elements

Table 5-68 Relationship description

Description	Target
Inclusion	VPC.SecurityGroup

Return Value

Property	Type	Description
refName	string	Security group rule name
refID	string	Security group rule ID

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  sg-id:
    type: Cloud.VPC.SecurityGroup.Id
  direction:
    default: ingress
    type: string
  ethertype:
    default: IPv4
    type: string
  protocol:
    default: TCP
    type: string
  minPort:
    default: 80
    type: integer
  maxPort:
    default: 80
    type: integer
  remoteSecurityGroup:
    type: Cloud.VPC.SecurityGroup.Id
node_templates:
  my-rule:
    type: Cloud.VPC.SecurityGroupRule
    properties:
      securityGroupId: {get_input: sg-id}
      direction: {get_input: direction}
      ethertype: {get_input: ethertype}
      protocol: {get_input: protocol}
      minPort: {get_input: minPort}
      maxPort: {get_input: maxPort}
      remoteSecurityGroup: {get_input: remoteSecurityGroup}
outputs:
  rule-id:
    value:
      get_attribute: [my-rule, refID]
```

5.2.26 VPC.Subnet

Element Description

The **VPC.Subnet** element is used to create a subnet on a VPC.

Element Properties

Table 5-69 Property Description

Property	Mandatory	Descripiton
dnsList	No	IP address set of the DNS server on the subnet. Set this field if you want to use more than two DNS servers. **Type:** ip array **Value Description:** It must be an IP address array, for example, ["8.8.8.8", "4.4.4.4", "6.6.6.6"]. **Value Constraint:** The value must be an IP address array and contain the values of **primaryDns** and **secondaryDns**. **Suggestion:** If a DNS server is needed in a subnet, set at least one of the **primaryDns** and **dnsLists** parameters. If **primaryDns**, **secondaryDns** and **dnsLists** are not set, the created subnet will not have a DNS server.
vpcId	Yes	ID of the VPC to which the subnet belongs **Type:** Cloud.VPC.VPC.Id **Value Description:** Use the ID of an existing VPC or a new VPC. To use a new VPC ID, you need to define the VPC object in the template and establish the dependency. You are advised to drag the object to the VPC to automatically establish the dependency. **Value Constraint:** The value must be in the CIDR format, for example, 192.168.0.0/16. **Suggestion:** 1. You are advised to use the **get_input** function to set values so that you can select an existing VPC when creating a stack. 2. To obtain the VPC information about this template, you are advised to use the **get_reference** function. 3. Obtain the ID of the created VPC from the VPC console.

Property	Mandatory	Descripiton
name	Yes	Subnet name **Type:** string **Value Description:** Customize the value, for example, musubnet. **Default:** "" **Value Constraint:** The value contains 1 to 64 characters. The value is unique in a VPC. Only letters, digits, underscores (_), and hyphens (-) are allowed. **Suggestion:** Customize the value. If this parameter is left blank, the system automatically assigns a name.
secondaryDns	No	IP address 2 of the DNS server on the subnet **Type:** ip **Value Description:** It must be in the IP address format, for example, 4.4.4.4. **Value Constraint:** The value must be an IP address.
gateway	Yes	Subnet gateway **Type:** ip **Value Description:** gateway address within the subnet CIDR address range **Default:** 192.168.1.1 **Value Constraint:** The value must be an IP address and comply with the gateway IP address rule, for example, 192.168.1.1. **Suggestion:** Customize the value based on the IP address range as required.
availabilityZone	No	AZ to which the subnet belongs **Type:** Cloud.ECS.AvailabilityZone.Name **Value Description:** AZ where the to-be-created ECS is located. The name of the AZ needs to be specified. **Value Constraint:** The value varies depending on regions. **Suggestion:** You are advised to use the **get_input** function to set values so that you can select a value from the list when creating a stack.

Property	Mandatory	Descripiton
primaryDns	No	IP address 1 of the DNS server on the subnet **Type:** ip **Value Description:** It must be in the IP address format, for example, 8.8.8.8. **Value Constraint:** The value must be an IP address. **Suggestion:** If a DNS server is needed in a subnet, set at least one of the **primaryDns** and **dnsLists** parameters. If **primaryDns**, **secondaryDns** and **dnsLists** are not set, the created subnet will not have a DNS server.
dhcpEnable	Yes	Whether to enable DHCP for the VPC subnet **Type:** boolean **Value Description: true**: Enabling the DHCP function. After an ECS using the VPC starts, the ECS automatically obtains an IP address using the DHCP protocol. **false**: Disabling the DHCP function. After an ECS using this VPC starts, the ECS cannot automatically obtain an IP address. You must manually assign an IP address to the ECS. **Default:** True **Value Constraint:** The value can be **true** or **false**. **Suggestion:** Set the value based on requirements. You are advised to enable the function.
cidr	Yes	Range of available addresses in a subnet **Type:** string **Value Description:** Range: 10.0.0.0/8-10.255.255.0/24, 172.16.0.0/12-172.31.255.0/24, or 192.168.0.0/16-192.168.255.0/24. **Default:** 192.168.1.0/24 **Value Constraint:** The value must be in the CIDR format, for example, 192.168.0.0/16. The value must be within the VPC CIDR block. **Suggestion:** Customize the value based on the IP address range as required.

Relationships Between Elements

Table 5-70 Relationship description

Description	Target
Contained In	VPC.VPC

Return Value

Property	Type	Description
neutron_network_id	string	OpenStack network ID
vpcId	string	ID of the VPC to which the subnet belongs
neutron_subnet_id	string	OpenStack subnet ID
refName	string	Subnet name
refID	string	Subnet ID

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  vpc-name:
    default: vpc
    type: string
  vpc-cidr:
    default: 192.168.0.0/16
    type: string
  subnet-name:
    type: string
    default: subnet
  subnet-cidr:
    default: 192.168.0.0/24
    type: string
  subnet-gateway:
    type: ip
    default: 192.168.0.1
  dhcenable:
    type: boolean
    default: true
  availabilityZone:
    description: Name of az
    label: ''
node_templates:
  my-vpc:
    type: Cloud.VPC.VPC
    properties:
      name:
        get_input: vpc-name
      cidr:
        get_input: vpc-cidr
```

```
my-subnet:
  type: Cloud.VPC.Subnet
  properties:
    name:
      get_input: subnet-name
    cidr:
      get_input: subnet-cidr
    gateway:
      get_input: subnet-gateway
    dhcpEnable:
      get_input: dhcenable
    dnsList: [114.114.114.115,114.114.114.114]
    vpcId:
      get_attribute: [my-vpc,refID]
    availabilityZone:
      get_input: availabilityZone
  requirements:
    - vpcId:
        node: my-vpc
```

5.2.27 VPC.VPC

Element Description

The **VPC.VPC** element is used to create a VPC network.

Element Properties

Table 5-71 Property Description

Property	Mandatory	Descripiton
cidr	Yes	Range of available subnets in the VPC **Type:** string **Value Description:** Range: 10.0.0.0/8-10.255.255.0/24, 172.16.0.0/12-172.31.255.0/24, or 192.168.0.0/16-192.168.255.0/24. **Default Value:** 192.168.0.0/16 **Value Constraint:** The value must be in the CIDR format, for example, 192.168.0.0/16. **Suggestion:** Customize the value based on the IP address range as required.

Property	Mandatory	Descripiton
name	Yes	VPC name **Type:** string **Value Description:** Customize the value, for example, myvpc. **Default:** "" **Value Constraint:** The value contains a maximum of 64 characters. Only letters, digits, underscores (_), hyphens (-), and periods (.) are allowed. When you specify a VPC name, ensure that it is unique within the account. **Suggestion:** Customize the value. If this parameter is left blank, the system automatically assigns a name.

Relationships Between Elements

None.

Return Value

Property	Type	Description
refID	string	VPC ID
refName	string	VPC name

Blueprint Example

```
tosca_definitions_version: cloud_tosca_version_1_0
inputs:
  vpc-name:
    default: vpc
    type: string
  vpc-cidr:
    default: 192.168.0.0/16
    type: string
node_templates:
  my-vpc:
    type: Cloud.VPC.VPC
    properties:
      name:
        get_input: vpc-name
      cidr:
        get_input: vpc-cidr
```

5.3 Data Structure

5.3.1 AOS.BatchItem

Property Description

Table 5-72 Property description

Property	Mandatory	Type	Description
values	No	dict	Variable defined in the batch template. Ensure that each key in the internal structure complies with the following requirement: "^[a-zA-Z_][a-zA-Z0-9_]*$".
properties	Yes	string	Attribute template of the Batch element. The template format is jinja. Based on the basic template, you can reconstruct a template to the YAML format (character strings) and define variables as required (that is, using the {{}} format). The built-in variables include {{item}}, {{limit}}, and {{offset}}.
element	Yes	string	Basic object of the Batch element **Value Constraint:** The value must be true and complete and match the item relationship.

5.3.2 Basic.KeyValuePair

Property Description

Table 5-73 Property description

Property	Required	Type	Description
key	Yes	string	Key of KeyValuePair
value	Yes	string	Value of KeyValuePair

5.3.3 Basic.Label

Property Description

Table 5-74 Property description

Property	Required	Type	Description
value	Yes	string	Value of Label
key	Yes	string	Key of Label

5.3.4 Basic.LabelSelector

Property Description

Table 5-75 Property description

Property	Required	Type	Description
values	Yes	string	Values of labelSelector
key	Yes	string	Key of LabelSelector
op	Yes	string	Op of the labelSelector, Supports "In", "NotIn", "Exists", "DoesNotExist", "Gt", "Lt"

5.3.5 Basic.NameAndSecretValue

Property Description

Table 5-76 Property description

Property	Required	Type	Description
name	Yes	string	Name of NameAndSecretValue
value	Yes	secret	Value of NameAndSecretValue

5.3.6 Basic.NameKeyPair

Property Description

Table 5-77 Property description

Property	Required	Type	Description
name	Yes	string	Name of NameKeyPair
key	Yes	string	Key of NameKeyPair

5.3.7 Basic.NameValuePair

Property Description

Table 5-78 Property description

Property	Required	Type	Description
name	Yes	string	Name of NameValuePair
value	Yes	string	Value of NameValuePair

5.3.8 CCE.Addon.AutoScaler.Node

Property Description

Table 5-79 Property description

Property	Mandatory	Type	Description
flavor	Yes	Cloud.CCE.Node.Flavor.Name	Node flavor
az	Yes	Cloud.ECS.AvailabilityZone.Name	Node AZ
os	Yes	String	Node OS
taints	No	CCE.Addon.AutoScaler.Taints	Node taints

5.3.9 CCE.DataVolume

Property Description

Table 5-80 Property description

Property	Mandatory	Type	Description
multiAttach	No	boolean	Information about the shared disk **Value Description: true**: shared EVS disk. **false**: common EVS disk. **Value Constraint:** The value can be **true** or **false**. **Suggestion:** Set the value based on the live environment.
volumeType	Yes	Cloud.EVS.Volume.Type.Name	Data disk type corresponding to the ECS. The data disk type must match the disk type provided by the system. **Value Description:** Data disk type corresponding to the ECS. The data disk type must match the disk type provided by the system. **Constraints:** The options are **SATA**, **SAS**, **SSD**, **co-p1**, and **uh-l1**. **SATA**: common I/O disk type; **SAS**: high I/O disk type; **SSD**: ultra-high I/O disk type; **co-p1**: high I/O (performance-optimized I); **uh-l1**: ultra-high I/O (latency-optimized) **Suggestion:** Set the value based on the live environment.
hw:passthrough	No	string	Data disk type **Value Description: true**: SCSI type. If this field does not exist, the VBD type is used by default. **Value Constraint:** This parameter can be set only to **true** or left blank. **Suggestion:** Set the value based on the live environment.
size	Yes	integer	Data disk size **Value Description:** data disk size (unit: GB) **Value Constraint:** [10, 32768] **Suggestion:** Set the value based on the live environment.

5.3.10 CCE.HelmChart

Property Description

Table 5-81 Property description

Property	Required	Type	Description
version	Yes	string	The version of the chart, default value is empty string.
name	Yes	string	The name of the chart, default value is empty string.

5.3.11 CCE.Labels

Property Description

Table 5-82 Property description

Property	Mandatory	Type	Description
scope	No	integer	Number of the nodes labeled **Value Description**: Enter an integer that is not greater than the number of nodes. If the value is obtained through the **get_input** function, set its type to integer, for example, type: integer. **Suggestion:** Customize the value.
key	No	string	Key of the label **Value Description:** Customize the value. **Suggestion:** Customize the value.
value	No	string	Value of the label **Value Description:** Customize the value. **Suggestion:** Customize the value.

5.3.12 CCE.NodePool

Property Description

Table 5-83 Property description

Property	Mandatory	Type	Description
dataVolumes	Yes	CCE.DataVolume	Data disk of the created node **Value Description:** Customize the value, for example, [{"volumeType":"SATA","size":100}]. **Value Constraint:** Array format. Currently, only one object is supported. **Suggestion:** Customize the value.
availabilityZone	Yes	Cloud.ECS.AvailabilityZone.Name	AZ where the node is located **Value Description:** AZ where the to-be-created ECS is located. The name of the AZ needs to be specified, for example, ae-ad-1a. For details, see the Regions and Endpoints page. **Value Constraint:** Select a value based on the region. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.
name	No	string	Name of the created node **Value Description:** Customize the value. **Value Constraint:** The value contains 4 to 32 characters and must start with a lowercase letter. Only lowercase letters, digits, and underscores (_) are allowed. **Suggestion:** Customize the value. Generally, the stack name is used as the node name.
publicKey	No	Cloud.ECS.KeyPair.PublicKey	Public key of the key pair when the stack is billed in the yearly/monthly mode **Value Description:** Selects an existing public key. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console based on the value of **sshKeyName**.

Property	Mandatory	Type	Description
postInstall	No	string	Node post-installation script **Value Description:** Customize the value. **Value Constraint:** The script you specify here will be executed after K8S software is installed **Suggestion:** The script is usually used to modify Docker parameters.
labels	No	dict	labels of the created node **Value Description:** Customize the value, for example, {"key": "aos","value": "app","scope":[1,2]}. **Suggestion:** Enter the key, value, and scope as required.
preInstall	No	string	Node pre-installation script **Value Description:** Customize the value. **Value Constraint:** The script you specify here will be executed before K8S software is installed. Note that if the script is incorrect, K8S software may not be installed successfully. **Suggestion:** The script is usually used to format data disks.
publicIp	No	**CCE.PublicIP**	Virtual IP address of the created node **Value Description:** Customize the value, for example, {"eip":{"bandwidth:{" shareType":PER}, 5_sbgp"}}. **Value Constraint:** Only one elastic IP address can be defined for each node. **Suggestion:** Customize the value.
instances	Yes	integer	Number of created nodes **Value Description:** Customize the value. The value ranges from 1 to 50. **Value Constraint:** {u'in_range': [1, 50]} **Suggestion:** Set the value based on specifications and requirements.
rootVolume	Yes	**ECS.RootVolume**	System disk of the created node **Value Description:** Customize the value, for example, {"volumeType":"SATA","size": 40}. **Suggestion:** Customize the value.

Property	Mandatory	Type	Description
os	No	string	os of the created node **Value Description:** ["EulerOS 2.2", "CentOS 7.4"] **Value Constraint:** Customize the value. If this parameter is left blank, EulerOS 2.2 is used by default. **Suggestion:** The options are **EulerOS 2.2** and **CentOS 7.4**.
nodePasswd	No	password	Password of the node **root** user **Value Description:** Customize the value. **Value Constraint:** 1. The parameter must be written into inputs and set using the **get_input** function. 2. The value must not be left blank or is a weak password. Enter 8 to 26 characters. Only uppercase and lowercase letters, digits, and special characters !@$%^-_=+[{}]:,./? are allowed. The value must contain at least two types of characters. 3. **sshkeyName** and **nodePasswd** cannot be used at the same time. **Suggestion:** You are advised to use the **get_input** function to obtain the value and avoid plaintext passwords to ensure security.
flavor	Yes	Cloud.CCE.Node.Flavor.Name	Container node specification **Value Description:** ID of the system flavor of the ECS to be created. You are advised to set this parameter in get_input mode. **Suggestion:** Select the node specification during node creation on the CCE console. In the node template, you can set the inputs to specify the node specification.

Property	Mandatory	Type	Description
sshKeyName	Yes	Cloud.ECS.KeyPair.Name	Key pair used for logging in to a node, which needs to be kept properly **Value Description:** It must be created on the ECS console in advance. **Value Constraint: sshkeyName** and **nodePasswd** cannot be used at the same time. **Suggestion:** 1. You are advised to use the **get_input** function to assign values so that you can select a value when using the template. 2. Obtain the information on the ECS console and then enter the information accordingly.
annotations	No	dict	Annotations of Node **Value Description:** Customize the value, for example, {"app": "aos"}. **Suggestion**: You can enter multiple key-value pairs.

5.3.13 CCE.PublicIP

Property Description

Table 5-84 Property description

Property	Mandatory	Type	Description
eip	No	CCE.EIP	Configuration parameter for creating an elastic IP address that will be automatically assigned to the ECS **Value Description:** CCE.EIP type **Value Constraint:** The value must meet the CCE.EIP type.
ids	No	string	ID of the existing elastic IP address list assigned to the to-be-created cluster node **Value Description:** It must be in the UUID format. Only elastic IP addresses in the DOWN state can be assigned. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters.

5.3.14 DCS.InstanceBackupPolicy

Property Description

Table 5-85 Property description

Property	Mandatory	Type	Description
extendParam	Yes	DCS.PeriodicalBackupPlan	Extended parameter of a DCS instance backup policy
backupType	Yes	string	Backup type **Value Description:** Customize the value. **Value Constraint:** The value can be **auto** or **manual**. **auto**: automatic backup. **manual**: manual backup. **Suggestion:** Use the default value.
saveDays	Yes	integer	Backup retention days **Value Description:** Customize the value. **Value Constraint:** The value ranges from 1 to 7. **Suggestion:** Use the default value.

5.3.15 DCS.PeriodicalBackupPlan

Property Description

Table 5-86 Property description

Property	Required	Type	Description
backupAt	Yes	string	Day in a week when backup starts **Value Description:** Supports customization. **Value Constraint:** The value ranges from 1 to 7. The value 1 indicates Monday and the value 7 indicates Sunday. **Suggestion:** You are advised to enter 1.

Property	Required	Type	Description
beginAt	Yes	string	Backup execution time. For example, 00 indicates 24:00, and 08 indicates 08:00. **Value Description:** Supports customization. **Value Constraint:** Currently, the value can only be 00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, or 23. **Suggestion:** Use the default value.
periodType	Yes	string	Backup period type **Value Description:** Supports customization. **Value Constraint:** Currently, the value can only be weekly. **Suggestion:** Use the default value.

5.3.16 ECS.DataVolume

Property Description

Table 5-87 Property description

Property	Mandatory	Type	Description
multiAttach	No	boolean	Information about the shared disk **Value Description: true**: shared EVS disk. **false**: common EVS disk. **Value Constraint:** The value can be **true** or **false**. **Suggestion:** Set the value based on the live environment.

Property	Mandatory	Type	Description
volumeType	Yes	Cloud.EVS.Volume.Type.Name	Data disk type corresponding to the ECS. The data disk type must match the disk type provided by the system. **Value Description:** Type of the ECS data disk. The data disk type must match the disk type provided by the system. The value can be **SATA** (common I/O), **SAS** (high I/O), **SSD** (ultra-high I/O), **co-p1** (high I/O; performance-optimized I), or **uh-l1** (ultra-high I/O; latency-optimized). **Constraints:** The options are **SATA**, **SAS**, **SSD**, **co-p1**, and **uh-l1**. **SATA**: common I/O disk type; **SAS**: high I/O disk type; **SSD**: ultra-high I/O disk type; **co-p1**: high I/O (performance-optimized I); **uh-l1**: ultra-high I/O (latency-optimized) **Suggestion:** Set the value based on the live environment.
hw:passthrough	No	string	Data disk type **Value Description: true**: SCSI type. If this field does not exist, the VBD type is used by default. **Value Constraint:** This parameter can be set only to **true** or left blank. **Suggestion:** Set the value based on the live environment.
size	Yes	integer	Data disk size **Value Description:** data disk size (unit: GB) **Value Constraint:** [10, 32768] **Suggestion:** Set the value based on the live environment.

5.3.17 ECS.EIP

Property Description

Table 5-88 Property description

Property	Mandatory	Type	Description
bandwidth	Yes	VPC.BandWidth	IP address bandwidth **Value Description:** VPC.BandWidth type **Value Constraint:** The value must meet the definition of the VPC.BandWidth type.
ipProductId	No	string	Product ID corresponding to the IP address **Value Description:** ID of the elastic IP address assigned to the ECS to be created. The value is in UUID format. **Value Constraint:** Only elastic IP addresses in the down state can be assigned.
ipType	Yes	Cloud.VPC.EIP.Spec.Name	Type of the virtual IP address

5.3.18 ECS.ExtendParam

Property Description

Table 5-89 Property description

Property	Required	Type	Description
CB_CSBS_BACKUP	No	string	Back up information **Value Description:** Customize the value. **Suggestion:** None
imageproductid	No	string	Image product ID **Value Description:** Customize the value. **Suggestion:** None
productId	No	string	product ID **Value Description:** Customize the value. **Suggestion:** None

5.3.19 ECS.MountedVolumes

Property Description

Table 5-90 Property description

Property	Mandatory	Type	Description
mountPath	Yes	string	Disk mount path, for example, /dev/sdb, /dev/sdc, or /dev/sdd. New disk mount paths cannot be the same as existing disk mount paths. **Value Constraint:** {u'regex': u'(/dev/[a-z]d[a-z]$)\|(^/dev/[a-z]da[a-z]$)|(^/dev/[a-z]db[a-h]$)'}
volumeId	Yes	string	ID of the disk to be mounted

5.3.20 ECS.NICS

Property Description

Table 5-91 Property description

Property	Mandatory	Type	Description
subnetId	Yes	Cloud.VPC.Subnet.Id	Information about the NIC of the to-be-created ECS **Value Description:** Obtain the subnet ID from the VPC service or connect to the **ECS.Subnet** to automatically establish the dependency. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** 1. Use the **get_input** function to import the **subnetId** field. The value of this field can be automatically selected on the AOS console. 2. Connect to a subnet and use the **get_reference** function to obtain the subnet ID.
allowedAddressPairs	No	ECS.AddressPair	Address pairs **Value Description:** Set this parameter based on the live network. **Suggestion:** None

Property	Mandatory	Type	Description
ipAddress	No	ip	IP address of the NIC of the to-be-created ECS **Value Description:** Indicates an IP address. If this field is left blank or is set to an empty string, an IP address will be automatically assigned. **Value Constraint:** IPv4 format. If this parameter is left blank or is an empty string, an unused IP address in the subnet of this network is automatically assigned as the IP address of the NIC. If an IP address is specified, the IP address must be in the network segment of the subnet corresponding to the network and is not in use.
ipCheck	No	boolean	Whether to perform IP check **Value Description:** Set this parameter based on the live network. **Suggestion:** None
portSecurityEnabled	No	boolean	Whether to enable **portSecurity** **Value Description:** Set this parameter based on the live network. **Suggestion:** None

5.3.21 ECS.Personality

Property Description

Table 5-92 Property description

Property	Required	Type	Description
path	Yes	string	path
contents	Yes	string	contents

5.3.22 ECS.PublicIP

Property Description

Table 5-93 Property description

Property	Mandatory	Type	Description
eip	No	ECS.EIP	Configuration parameter for creating an elastic IP address that will be automatically assigned to the ECS **Value Description:** ECS.EIP type. **Value Constraint:** The value must meet the ECS.EIP type.
id	No	string	ID of the existing elastic IP address assigned to the to-be-created ECS **Value Description:** It must be in the UUID format. Only elastic IP addresses in the DOWN state can be assigned. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters.

NOTE

You can configure either but not both of **id** and **eip** in the **publicip** field.

5.3.23 ECS.RootVolume

Property Description

Table 5-94 Property description

Property	Mandatory	Type	Description
volumeType	Yes	Cloud.EVS.Volume.Type.Name	System disk type **Value Description:** System disk type. **SATA**: common I/O disk. **SAS**: high I/O disk. **SSD**: ultra-high I/O disk. **co-p1**: high I/O (performance-optimized I) disk. **uh-l1**: ultra-high I/O (latency-optimized). **Constraints:** The options are **SATA**, **SAS**, **SSD**, **co-p1**, and **uh-l1**. **SATA**: common I/O disk type; **SAS**: high I/O disk type; **SSD**: ultra-high I/O disk type; **co-p1**: high I/O (performance-optimized I); **uh-l1**: ultra-high I/O (latency-optimized) **Suggestion:** Set the value based on the live environment.
size	Yes	integer	System disk type **Value Description:** system disk size (unit: GB). **Value Constraint:** [40,1024] **Suggestion:** Set the value based on the live environment.

5.3.24 ECS.SecurityGroup

Property Description

Table 5-95 Property description

Property	Mandatory	Type	Description
id	Yes	Cloud.VPC.SecurityGroup.Id	ID of the security group corresponding to the ECS. This ID takes effect for the NIC configured on the ECS. **Value Description:** ID of an existing security group. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Suggestion:** Use the **get_input** function to set this field, and then the value can be automatically selected on the AOS console.

5.3.25 ECS.ServerTags

Property Description

Table 5-96 Property description

Property	Required	Type	Description
value	Yes	string	Specifies the tag value. **Value Constraint:** One ECS supports up to 10 tags. The value can contain a maximum of 43 Unicode characters and can be left blank. It cannot contain ASCII (0-31) or the following characters: =*<>\,|/
key	Yes	string	Specifies the tag key. **Value Constraint:** One ECS supports up to 10 tags. The key contains a maximum of 36 Unicode characters. This field cannot be left blank. It cannot contain ASCII (0-31) or the following characters: "=*<>\,|/".

5.3.26 ECS.VolumeExtendParam

Property Description

Table 5-97 Property description

Property	Required	Type	Description
resourceType	No	string	resource Type **Value Description:** Customize the value. **Suggestion:** None
resourceSpecCode	No	string	Specifies the code of the disk specifications **Value Description:** Customize the value. **Suggestion:** None
productId	No	string	product ID **Value Description:** Customize the value. **Suggestion:** None

5.3.27 K8S.PodSecurityContext

Property Description

Table 5-98 Property description

Property	Mandatory	Type	Description
runAsUser	No	integer	User **Value Constraint:** {u'in_range': [0, 2147483647]}
supplementalGroups	No	integer	Supplement group **Value Constraint:** {u'in_range': [0, 2147483647]}
fsGroup	No	integer	File system **Value Constraint:** {u'in_range': [0, 2147483647]}
hostNetwork	No	boolean	Using the host network
runAsNonRoot	No	boolean	Non-root user

Property	Mandatory	Type	Description
seLinuxOptions	No	**K8S.SecurityContext.SeLinuxOptions**	Option
hostIPC	No	boolean	Using the host IPC
hostPID	No	boolean	Using the host PID

5.3.28 K8S.SecurityContext.SeLinuxOptions

Property Description

Table 5-99 Property description

Property	Required	Type	Description
type	No	string	Type of Selinux
role	No	string	Role of Selinux
user	No	string	User of Selinux
level	No	string	Level of Selinux

5.3.29 MySQL.DBUser

Property Description

Table 5-100 Property description

Property	Required	Type	Description
userPassword	Yes	password	Password for logging in to the database. The value is not empty. It contains 8-32 characters and is not a weak password. It contains letters, digits, and the following special characters: ~!@#%^*-_=+? **Value Description:** Supports customization. **Value Constraint:** 1. The parameter must be written into inputs and imported using the get_input function. 2. The value contains 8-32 characters and is not a weak password. It contains letters, digits, and the following special characters: ~!@#%^*-_=+? suggestion: 'You are advised to use the get_input function to obtain the value and avoid plaintext passwords to ensure security.'
name	Yes	string	Username **Value Description:** Cannot be the following fields: root, rdsadmin, rdsbackup, or rdsrepl. If this parameter is left blank, no user is created. **Value Constraint:** The value must meet MySQL user name requirements. **Suggestion:** Customize the value.

5.3.30 MySQL.DataBase

Property Description

Table 5-101 Property description

Property	Mandatory	Type	Description
characterSet	Yes	string	Character set of the database **Value Description:** Set this parameter based on the character sets supported by RDS, for example, **utf8** or **gbk**. **Suggestion:** You can view the attribute of the **character_set_database** field on the parameter group management page of the RDS console.
name	Yes	string	Database name **Value Description:** The following values are not supported: **mysql**, **information_schema**, and **performance_schema**. If this parameter is left blank, no database is created. **Value Constraint:** The value must meet MySQL database name requirements. **Suggestion:** Customize the value.
collate	Yes	string	Encoding format of the database **Value Description:** Set this parameter based on the formats supported by RDS, for example, **utf8_general_ci**, **utf8_bin**, **utf8_unicode_ci**, or **gbk_bin**. **Suggestion:** You can view the attribute of the **collation_server** field on the parameter group management page of the RDS console.

5.3.31 MySQL.DataStore

Property Description

Table 5-102 Property description

Property	Required	Type	Description
dbtype	Yes	string	Database type **Value Description:** MySQL **Value Constraint:** The value can only be MySQL. **Suggestion:** Set the value based on specifications and requirements.
version	Yes	string	Database version **Value Description:** MySQL engine supports versions 5.6 and 5.7 and 8.0. Examples of values:5.7. **Suggestion:** Set the value based on specifications and requirements.

5.3.32 RDS.BackupStrategy

Property Description

Table 5-103 Property description

Property	Mandatory	Type	Description
keepDays	Yes	integer	Backup retention period, which specifies the number of days for which backup files can be stored **Value Description:** The value ranges from 0 to 35. Unit: day. If this parameter is not specified or set to 0, the automatic backup policy is disabled. **Value Constraint:** {u'in_range': [0, 35]} **Suggestion:** Set the value based on the live environment.

Property	Mandatory	Type	Description
endTime	Yes	string	The latest time when a backup task is executed **Value Description:** Customize the value, for example, **23:30**. **Value Constraint:** The value cannot be blank and must be in the **HH:MM** format. The current time is the UTC time. **Suggestion:** Set the value based on the live environment.
startTime	Yes	string	Earliest time when a backup task is executed. Automatic backup will be triggered after the earliest time expires. **Value Description:** Customize the value, for example, **22:30**. **Value Constraint:** The value cannot be blank and must be in the **hh:mm** format. The current time is the UTC time. The value of **HH** must be 1 greater than the value of **hh**. The values of **mm** and **MM** must be the same and must be **00**, **15**, **30**, or **45**. **Suggestion:** Set the value based on the live environment.

5.3.33 RDS.HA.Mysql

Property Description

Table 5-104 Property description

Property	Mandatory	Type	Description
replicationMode	Yes	string	Synchronization parameter of the standby node **Value Description:** The options are **async** and **semisync**. **async** indicates the asynchronous mode. **semisync** indicates the semi-synchronous mode. **Value Constraint:** {u'valid_values': [u'async', u'semisync']} **Suggestion:** Set the value based on the live environment.

Property	Mandatory	Type	Description
enable	Yes	string	Whether HA is supported **Value Description:** The options are **true** and **false**. **Value Constraint:** 1. The HA parameter must be consistent with the specification parameter. 2. Note that the parameter must be in the character string format. When a YAML template is used, quotation marks ('' '') must be added because the values **true** and **false** are considered as Boolean values in the YAML template. **Suggestion:** If the instance specifications name contains the HA parameter, set this parameter to **true**. Otherwise, set this parameter to **false**.

5.3.34 RDS.Volume

Property Description

Table 5-105 Property description

Property	Mandatory	Type	Description
volumetype	Yes	Cloud.RDS.Volume.Type.Name	Disk type **Value Description:** The options are **COMMON** (SATA), **HIGH** (SAS), and **ULTRAHIGH** (SSD). These values are case sensitive. **Value Constraint:** Set the value based on requirements. **Suggestion:** Set the value based on specifications.
size	Yes	integer	Disk size **Value Description:** The value ranges from 40 to 4000. Unit: GB. **Value Constraint:** The value ranges from 40 to 4000. The value must be an integer multiple of 10. **Suggestion:** Set the value based on specifications.

5.3.35 ULB.StickySession

Property Description

Table 5-106 Property description

Property	Required	Type	Description
type	Yes	string	Session persistence type **Value Description:** Indicates the source IP address. **Value Constraint:** ["SOURCE_IP"] **Suggestion:** Use the default value.

5.3.36 VPC.BandWidth

Property Description

Table 5-107 Property description

Property	Mandatory	Type	Description
name	No	string	Name of the created bandwidth **Value Description:** Enter a maximum of 64 characters. Only hyphens (-), underscores (_), uppercase and lowercase letters, and digits are allowed. **Value Constraint:** The value contains 1 to 64 characters. This value is unique under an account, and must meet regular expression **{"regex":"^[a-zA-Z][0-9a-zA-Z-_]*$","min_length":1,"max_length":64}**.
shareType	Yes	string	Bandwidth type **Value Description:** The options are **PER** and **WHOLE**. **PER**: exclusive. **WHOLE**: shared. **Value Constraint:** The options are **PER** and **WHOLE**.
chargeMode	No	string	Billing mode **Value Description:** The options are **bandwidth** and **traffic**. **Value Constraint:** The options are **bandwidth** and **traffic**.

Property	Mandatory	Type	Description
productId	No	string	Product ID **Value Description:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters.
id	No	string	Existing bandwidth ID **Value Description:** The value must satisfy the UUID rule and contain a maximum of 64 characters. **Value Constraint:** The value must satisfy the UUID rule and contain a maximum of 64 characters.
size	No	integer	Bandwidth **Value Description:** bandwidth (Mbit/s). The value ranges from 1 to 300. This parameter is mandatory when **share_type** is set to **PER**. This parameter is ignored when **share_type** is set to **WHOLE** with an ID specified. **Value Constraint:** The value must range from 1 to 300.

5.3.37 VPC.PublicIP

Property Description

Table 5-108 Property description

Property	Mandatory	Type	Description
type	Yes	Cloud.VPC.EIP.Spec.Name	Type of the public elastic IP address
ipAddress	No	string	Public elastic IP address to be applied. If no address is specified, the system automatically assigns one. **Value Constraint:** The value must be an IP address and in the available address range.

5.4 Appendix

5.4.1 YAML Syntax

YAML is a simple and powerful language. It is designed to make the language easy to read.

Basic Syntax Rules

- Characters are case-sensitive.
- Indentation is used to represent hierarchical relationships.
- Only spaces can be used for indentation.
- The number of spaces used for indentation does not matter. Elements of the same level must be aligned on the left side.
- Lines that start with a number sign (#) are comments.

YAML supports three types of data structures.

- Object: A set of key-value pairs, which is also called mapping, hashes, or dictionary.
- Array: A group of values arranged in sequence, which is also called a sequence or list.
- Scalar: A single and irreducible value, which is the minimum data unit.

Object

An object is a group of key-value pairs. For key: value, the colon (:) must be followed by a space or newline character. The valid expression is as follows:

```
animal: pets
plant:
  tree
```

You can also write multiple key-value pairs into an inline object.

```
hash: {name: Steve, foo: bar}
```

However, an error occurs in the following scenario:

```
foo: somebody said I should put a colon here: so I did
windows_drive: c:
```

To resolve the issue, you can use single quotation marks (' '), as shown in the following:

```
foo: 'somebody said I should put a colon here: so I did'
windows_drive: 'c:'
```

Array

An array is represented by a hyphen (-) and space. The valid expression is as follows:

```
animal:
- Cat
```

```
- Dog
- Goldfish
```

You can also use the inline representation.

```
animal: [Cat, Dog, Goldfish]
```

Objects and arrays can be used in combination to form a composite structure.

```
languages:
 - Ruby
 - Perl
 - Python
websites:
 YAML: yaml.org
 Ruby: ruby-lang.org
 Python: python.org
 Perl: use.perl.org
```

Scalar

Scalar data types include string, Boolean value, integer, floating-point number, null, time, and date.

- String:

  By default, a string is not enclosed in quotation marks.

  ```
  str: This_is_a_line
  ```

  If a string contains spaces or special characters, the string needs to be enclosed in quotation marks.

  ```
  str: 'content: a string'
  ```

  Both single and double quotation marks can be used. The difference between them is that the former can identify escape characters while the latter cannot convert special characters.

  ```
  s1: 'content:\n a string'
  s2: "content:\n a string"
  ```

  If there is a single quotation mark between two single quotation marks, ensure that two single quotation marks are used consecutively to achieve conversion.

  ```
  str: 'labor''s day'
  ```

  Strings can be written into multiple lines. The lines except the first line must be indented with one space. The newline character will be converted to a space.

  ```
  str: This_is
   a_multi_line
  ```

- Integer:

  ```
  int_value: 314
  ```

- Floating-point number:

  ```
  float_value: 3.14
  ```

- Null:

  ```
  parent: ~
  ```

- Time:

  The time is in the ISO8601 format.

  ```
  iso8601: 2018-12-14t21:59:43.10-05:00
  ```

- Date:

  The date is in the compound ISO8601 format: year-month-day.

```
date: 1976-07-31
```

Special Symbols

- ---: indicates the start of a YAML file. ...: indicates the end of a YAML file.

```
---
# A list of delicious fruits
- Apple
- Strawberry
- Mango
...
```

- You can use two exclamation marks (!) to forcibly convert an integer, a floating-point number, or a Boolean value.

```
strbool: !!str true
strint: !!str 10
```

- For a string in multiple lines, you can use a literal block scalar (|) to start new lines or folded block scalar (>) to fold new lines. The two symbols are often used in the character strings of YAML files.

```
this: |
 Foo
 Bar
that: >
 Foo
 Bar
```

The corresponding objects are as follows:

```
{ this: 'Foo\nBar\n', that: 'Foo Bar\n' }
```

It is advised to use block scalars (|) in most scenarios.

Comment

YAML supports comments. This is an advantage of YAML compared with JSON.

The comment of YAML starts with a number sign (#), as shown in the following:

```
languages:
 - Ruby              # Indicates the Ruby language.
 - Go               # Indicates the Go language.
   -- PythonPy            # Indicates the Python language.
```

Reference Documents

- **YAML 1.2 Specifications**
- **Ansible YAML Syntax**

6 FAQs

6.1 What Is AOS?

Application Orchestration Service (AOS) allows you to deploy your applications in a few clicks, simplifying cloud service management. Using templates to describe and orchestrate applications and related cloud services, AOS facilitates automatic application deployment, cloud service creation, and E2E application lifecycle management.

6.2 What Is a Stack?

A stack is a collection of applications and cloud service resources. The applications or cloud services in a stack are treated as a unit when they are being created, upgraded, or deleted.

With AOS, you can create stacks to deploy your applications with ease and uniformly manage the cloud service resources on which the applications depend.

6.3 What Is a Template?

A template is a text file that complies with AOS syntax. It defines application attributes, cloud service configurations, and dependencies between applications and cloud services. Like code, templates can be managed by users. Code management tools such as Git and SVN can also be used to manage templates of different versions. Using templates to manage applications and cloud services simplifies the design of deploying application systems on the cloud and facilitates replicating and setting up development, testing, and production environments. This makes application systems configurable, evolvable, and traceable.

6.4 What Is a TOSCA Template?

Topology and Orchestration Specification for Cloud Application (TOSCA) is an OASIS open standard that defines the interoperable description of services and applications hosted in the cloud, thereby facilitating lifecycle management across

cloud service providers. **Figure 6-1** shows the application topology model used by AOS.

Figure 6-1 Application topology model

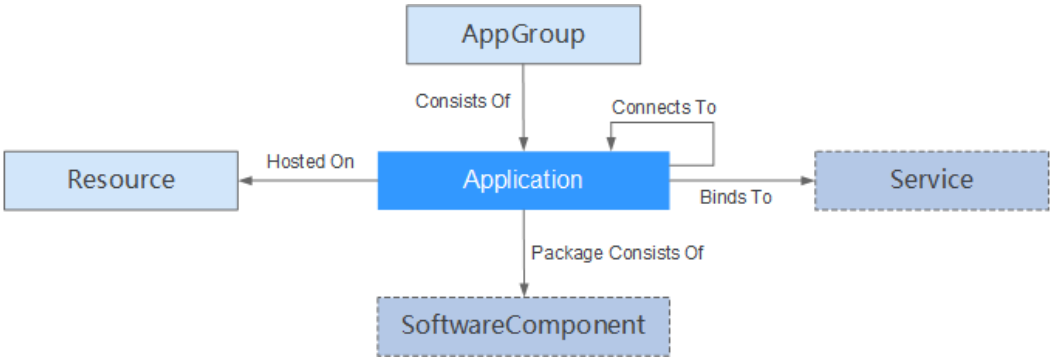


Table 6-1 Elements and relationships in the application topology model

Element	Description
Resource	Resources, including virtual machines (VMs) and containers.
AppGroup	A group of one or more cloud applications. You can perform lifecycle management (such as deployment and upgrade) for the entire group rather than for individual cloud application. A cloud application group can be a customer product, service system, or subsystem.
Application	Cloud application running on resources. It is the smallest deployable object in the application topology model. Note that a microservice is a type of application.
SoftwareComponent	Software component of a cloud application. Each component is packed into a software package. This element is optional and can also be an attribute of the **Application** element.
Service	Service on which an application depends. A service is a set of function objects.
DependsOn	Dependencies between elements, which determine the sequence in which elements will be created.
HostedOn	Relationship between applications and the resources they depend on.
ConsistsOf	Relationship of composition. For example, AppGroup consists of applications.
ConnectsTo	Relationship between two elements. For example, applications or resources are connected with each other.

Element	Description
PackageConsistsOf	Composition relationship between applications and their software components.

6.5 How Do I Upgrade a Stack?

AOS does not support stack upgrade. You can update a template or upload a new template to create an up-to-date stack.

A Change History

Table A-1 Change history

Date	Description
2020-11-05	This issue is the first official release.