云数据迁移

API 参考

文档版本 1.0

发布日期 2025-04-02

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目 录

1 使用前必读

1.1 概述

欢迎使用云数据迁移（Cloud Data Migration，以下简称CDM），该服务提供同构/异构数据源之间批量数据迁移服务，帮助您实现数据自由流动。支持自建和云上的文件系统，关系数据库，数据仓库，NoSQL，大数据云服务，对象存储等数据源。

您可以使用本文档提供的API对云数据迁移服务进行操作，如创建集群、创建迁移任务等，支持的全部操作请参见**API概览**。

在调用云数据迁移服务API之前，请确保已经充分了解云数据迁移服务相关概念，详细信息请参见**产品介绍**。

1.2 终端节点

终端节点（Endpoint）即调用API的**请求地址**，不同服务不同区域的终端节点不同，您可以从**地区和终端节点**中查询所有服务的终端节点。

云数据迁移服务的终端节点如**表1-1**所示，请您根据业务需要选择对应区域的终端节点。

表 1-1 CDM 数据集成 Endpoint

区域名称	区域ID	终端节点（Endpoint）	协议类型
华北-北京一	cn-north-1	cdm.cn-north-1.myhuaweicloud.com	HTTPS
华北-北京二	cn-north-2	cdm.cn-north-2.myhuaweicloud.com	HTTPS
华北-北京四	cn-north-4	cdm.cn-north-4.myhuaweicloud.com	HTTPS
华北-乌兰察布一	cn-north-9	cdm.cn-north-9.myhuaweicloud.com	HTTPS
华东-上海一	cn-east-3	cdm.cn-east-3.myhuaweicloud.com	HTTPS
华东-上海二	cn-east-2	cdm.cn-east-2.myhuaweicloud.com	HTTPS

区域名称	区域ID	终端节点（Endpoint）	协议类型
华南-广州	cn-south-1	cdm.cn-south-1.myhuaweicloud.com	HTTPS
西南-贵阳一	cn-southwest-2	cdm.cn-southwest-2.myhuaweicloud.com	HTTPS

1.3 项目 ID 和账号 ID

获取项目 ID 和账号 ID

项目ID表示租户的资源，账号ID对应当前账号，IAM用户ID对应当前用户。用户可在对应页面下查看不同Region对应的项目ID、账号ID和用户ID。

1. 注册并登录管理控制台。
2. 在用户名的下拉列表中单击“我的凭证”。
3. 在“API凭证”页面，查看账号名和账号ID、IAM用户名和IAM用户ID，在项目列表中查看项目和项目ID。

调用 API 获取项目 ID

项目ID可以通过调用**查询指定条件下的项目信息**API获取，接口为“GET https://{Endpoint}/v3/projects”，其中{Endpoint}为IAM的终端节点，可参考IAM文档获取。

接口的认证鉴权请参见**认证鉴权**。

响应示例如下，其中projects下的“id”即为项目ID。当返回多个id，请依据实际的区域（name）获取。

```
{
    "projects": [
        {
            "domain_id": "65382450e8f64ac0870cd180d14e684b",
            "is_domain": false,
            "parent_id": "65382450e8f64ac0870cd180d14e684b",
            "name": "region-name",
            "description": "",
            "links": {
                "next": null,
                "previous": null,
                "self": "https://www.example.com/v3/projects/a4a5d4098fb4474fa22cd05f897d6b99"
            },
            "id": "a4a5d4098fb4474fa22cd05f897d6b99",
            "enabled": true
        }
    ],
    "links": {
        "next": null,
        "previous": null,
        "self": "https://www.example.com/v3/projects"
    }
}
```

1.4 基本概念

- 账号

  用户注册时的账号，账号对其所拥有的资源及云服务具有完全的访问权限，可以重置用户密码、分配用户权限等。由于账号是付费主体，为了确保账号安全，建议您不要直接使用账号进行日常管理工作，而是创建IAM用户并使用他们进行日常管理工作。

- IAM用户

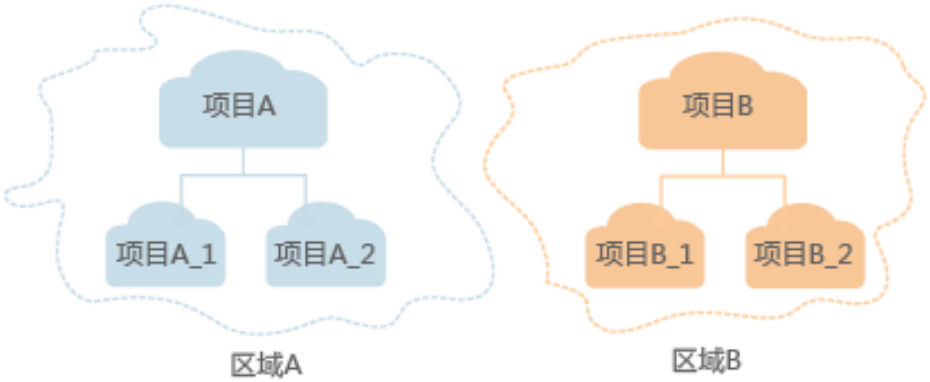
  由账号在IAM中创建的用户，是云服务的使用人员，具有身份凭证（密码和访问密钥）。

  在**我的凭证**下，您可以查看账号ID和用户ID。通常在调用API的鉴权过程中，您需要用到账号、用户和密码等信息。

- 项目

  区域默认对应一个项目，这个项目由系统预置，用来隔离物理区域间的资源（计算资源、存储资源和网络资源），以默认项目为单位进行授权，用户可以访问您账号中该区域的所有资源。如果您希望进行更加精细的权限控制，可以在区域默认的项目中创建子项目，并在子项目中购买资源，然后以子项目为单位进行授权，使得用户仅能访问特定子项目中资源，使得资源的权限控制更加精确。

  图 1-1 项目隔离模型



2 API 概览

CDM所提供的API为自研API。通过配合使用CDM自研API，您可以使用CDM的如下功能。

表 2-1 CDM API 概览

类型	API	说明	用户流量限制（单位时间内的单个用户请求次数上限）
集群管理API	创建集群	创建CDM集群。	5次/min
	查询集群列表	查询并显示集群列表。	120次/min
	查询集群详情	查询集群详情。	120次/min
	重启集群	重启CDM集群。	20次/min
	删除集群	删除指定CDM集群。	20次/min
	停止集群	将指定CDM集群关机。	20次/min
	查询所有可用区	查询并显示所有可用区。	20次/min
	查询支持的版本	查询并显示支持的集群版本。	20次/min
	查询版本规格	查询并显示集群版本规格。	20次/min
	查询规格详情	查询并显示集群规格详情。	20次/min
	查询所有集群的企业项目ID	查询并显示所有集群的企业项目ID。	20次/min

类型	API	说明	用户流量限制（单位时间内的单个用户请求次数上限）
	查询集群的企业项目ID	查询并显示集群的企业项目ID。	20次/min
	查询集群实例信息	查询并显示集群实例信息。	20次/min
	修改集群	修改指定CDM集群。	20次/min
	启动集群	开启指定CDM集群。	20次/min
连接管理API	创建连接	连接指定的数据源。	120次/min
	查询连接	查询连接列表。	120次/min
	修改连接	修改连接的参数。	120次/min
	删除连接	删除指定连接。	120次/min
作业管理API	指定集群创建作业	在指定的CDM集群上创建数据迁移任务，作业不会启动。	1200次/min
	随机集群创建作业并执行	在指定的CDM集群列表中，随机选择一个开机状态的集群，在该集群中创建作业并执行作业。	120次/min
	查询作业	查询并显示作业列表。	120次/min
	修改作业	修改作业的参数。	120次/min
	启动作业	启动数据迁移的任务。	1200次/min
	停止作业	停止运行中的作业。	1200次/min
	查询作业状态	查询并显示作业的运行状态。	120次/min
	查询作业执行历史	查询并显示作业执行的历史状态。	120次/min
	删除作业	删除指定作业。	120次/min

使用 API 注意事项

- CDM的作业数据量太多，会造成数据库压力，建议定时清理不需要的作业。
- 短时间内下发大量作业，可能会导致集群资源耗尽异常等，您调用API时需要注意。
- CDM是批量离线迁移工具，不建议客户创建大量小作业场景。

3 如何调用 API

3.1 构造请求

本节介绍REST API请求的组成，并以调用IAM服务的**获取用户Token**接口来说明如何调用API，该API获取用户的Token，Token可以用于调用其他API时鉴权。

您还可以通过这个视频教程了解如何构造请求调用API：**https://bbs.huaweicloud.com/videos/102987** 。

请求 URI

请求URI由如下部分组成。

{URI-scheme}://{Endpoint}/{resource-path}?{query-string}

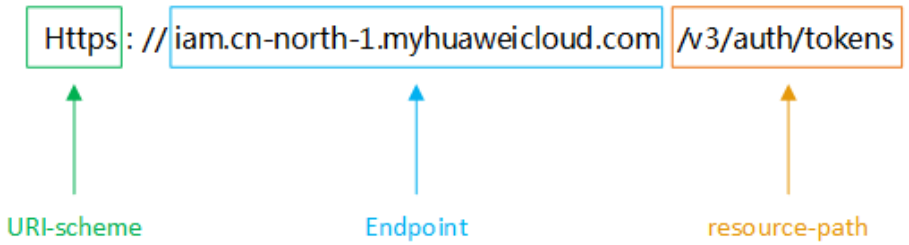
表 3-1 请求 URI

参数	说明
URI-scheme	传输请求的协议，当前所有API均采用**HTTPS**协议。
Endpoint	承载REST服务端点的服务器域名或IP，不同服务在不同区域时，对应Endpoint不同，可以从**终端节点**中获取。 例如IAM服务在“华北-北京一”区域的Endpoint为“iam.cn-north-1.myhuaweicloud.com”。
resource-path	资源路径，即API访问路径。从具体API的URI模块获取，例如“获取用户Token”API的resource-path为“/v3/auth/tokens”。
query-string	查询参数，可选，查询参数前面需要带一个“？”，形式为“参数名=参数取值”，例如“limit=10”，表示查询不超过10条数据。

例如您需要获取“华北-北京一”区域的Token，则需使用“华北-北京一”区域的Endpoint（iam.cn-north-1.myhuaweicloud.com），并在**获取用户Token**的URI部分找到resource-path（**/v3/auth/tokens**），拼接起来如下所示。

```
https://iam.cn-north-1.myhuaweicloud.com/v3/auth/tokens
```

图 3-1 URI 示意图



说明

为查看方便，服务每个具体API的URI，只给出resource-path部分，并将请求方法写在一起。这是因为URI-scheme都是HTTPS，而Endpoint在同一个区域也相同，所以简洁起见将这两部分省略。

请求方法

HTTP请求方法（也称为操作或动词），它告诉服务您正在请求什么类型的操作。

表 3-2 HTTP 方法

方法	说明
GET	请求服务器返回指定资源。
PUT	请求服务器更新指定资源。
POST	请求服务器新增资源或执行特殊操作。
DELETE	请求服务器删除指定资源，如删除对象等。
HEAD	请求服务器资源头部。
PATCH	请求服务器更新资源的部分内容。 当资源不存在的时候，PATCH可能会去创建一个新的资源。

在**获取用户Token**的URI部分，您可以看到其请求方法为“POST”，则其请求为：

```
POST https://iam.cn-north-1.myhuaweicloud.com/v3/auth/tokens
```

请求消息头

附加请求头字段，如指定的URI和HTTP方法所要求的字段。例如定义消息体类型的请求头“Content-Type”，请求鉴权信息等。

需要添加到请求中的公共消息头如**表3-3**所示。

表 3-3 公共请求消息头

名称	描述	是否必选	示例
Content-type	消息体的类型（格式），默认取值为“application/json”。 如果请求消息体中含有中文字符，则还需要通过charset=utf8指定中文字符集。	是	application/json;charset=utf8
Content-Length	请求body长度，单位为Byte。	否	3495
X-Project-Id	project id，用于不同project取token。	否	e9993fc787d94b6c886cbaa340f9c0f4
X-Auth-Token	用户Token，也就是调用**获取用户Token**接口的响应值，该接口是唯一不需要认证的接口。	是	-
X-Language	请求语言。	是	en_us
x-sdk-date	请求的发生时间，格式为(YYYYMMDD'T'HHMMSS'Z')。 取值为当前系统的GMT时间。	否	20190407T101459Z
Host	请求的服务器信息，从服务API的URL中获取，值为“hostname[:port]”。端口缺省时使用默认的端口，https的默认端口为443。	否	code.test.com 或 code.test.com:443

对于**获取用户Token**接口，由于不需要认证，所以只添加“Content-Type”即可，添加消息头后的请求如下所示。

```
POST https://iam.cn-north-1.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

请求消息体

请求消息体通常以结构化格式发出，与请求消息头中Content-type对应，传递除请求消息头之外的内容。若请求消息体中参数支持中文，则中文字符必须为UTF-8编码。

每个接口的请求消息体内容不同，也并不是每个接口都需要有请求消息体（或者说消息体为空），GET、DELETE操作类型的接口就不需要消息体，消息体具体内容需要根据具体接口而定。

对于**获取用户Token**接口，您可以从接口的请求部分看到所需的请求参数及参数说明。将消息体加入后的请求如下所示。

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json

{
    "auth": {
        "identity": {
```

```
        "methods": [
            "password"
        ],
        "password": {
            "user": {
                "name": "username",
                "password": "********",
                "domain": {
                    "name": "domainname"
                }
            }
        }
    },
    "scope": {
        "project": {
            "id": "xxxxxxxxxxxxxxxxxx"
        }
    }
  }
}
```

说明

scope参数定义了Token的作用范围，取值为project或domain，示例中取值为project，表示获取的Token仅能访问指定project下的资源。取值为domain时，表示获取的token可以访问指定账号下所有资源。

scope参数的详细说明，请参见**获取用户Token**接口，

到这里为止这个请求需要的内容就具备齐全了，您可以使用**curl**、**Postman**或直接编写代码等方式发送请求调用API。对于获取用户Token接口，返回的响应消息头中“x-subject-token”的值，就是需要获取的用户Token。有了Token之后，您就可以使用Token认证调用其他API。

3.2 认证鉴权

调用接口有如下两种认证方式，您可以选择其中一种进行认证鉴权。

- Token认证：通过Token认证通用请求。
- AK/SK认证：通过AK（Access Key ID）/SK（Secret Access Key）加密调用请求。

说明

仅当创建IAM用户时的访问方式勾选“编程访问”后，此IAM用户才能通过认证鉴权，从而使用API、SDK等方式访问DataArts Studio。

Token 认证

说明

- Token的有效期为24小时，需要使用一个Token鉴权时，可以先缓存起来，避免频繁调用。
- 使用Token前请确保Token离过期有足够的时间，防止调用API的过程中Token过期导致调用API失败。

Token在计算机系统中代表令牌（临时）的意思，拥有Token就代表拥有某种权限。Token认证就是在调用API的时候将Token加到请求消息头，从而通过身份认证，获得操作API的权限。

Token可通过调用**获取用户Token**接口获取，调用本服务API需要project级别的Token，即调用**获取用户Token**接口时，请求body中auth.scope的取值需要选择project，如下所示。

```
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "id": "xxxxxxxxxxxxxxxxxx"
            }
        }
    }
}
```

获取Token后，再调用其他接口时（以数据开发组件的“查询连接列表”接口为例），您需要在请求消息头中添加“X-Auth-Token”，其值即为Token。例如Token值为“ABCDEFG....”，则调用接口时将“X-Auth-Token: ABCDEFJ....”加到请求消息头即可，如下所示。

```
GET https://iam.cn-north-4.myhuaweicloud.com/v1/{project_id}/connections
Content-Type: application/json
X-Auth-Token: ABCDEFG....
```

您还可以通过这个视频教程了解如何使用Token认证：**https://bbs.huaweicloud.com/videos/101333** 。

AK/SK 认证

说明

- AK/SK签名认证方式仅支持消息体大小12MB以内，12MB以上的请求请使用Token认证。
- AK/SK既可以使用永久访问密钥中的AK/SK，也可以使用临时访问密钥中的AK/SK，但使用临时访问密钥的AK/SK时需要额外携带“X-Security-Token”字段，字段值为临时访问密钥的security_token。

AK/SK认证就是使用AK/SK对请求进行签名，在请求时将签名信息添加到消息头，从而通过身份认证。

- AK(Access Key ID)：访问密钥ID。与私有访问密钥关联的唯一标识符；访问密钥ID和私有访问密钥一起使用，对请求进行加密签名。
- SK(Secret Access Key)：与访问密钥ID结合使用的密钥，对请求进行加密签名，可标识发送方，并防止请求被修改。

您可以通过如下方式获取访问密钥。

1. 登录控制台，在用户名下拉列表中选择“我的凭证”。
2. 进入“我的凭证”页面，选择“访问密钥 > 新增访问密钥”，如**图3-2**所示。

图 3-2 单击新增访问密钥

3. 单击“确定”，根据浏览器提示，保存密钥文件。密钥文件会直接保存到浏览器默认的下载文件夹中。打开名称为“credentials.csv”的文件，即可查看访问密钥（Access Key Id和Secret Access Key）。

说明

- 每个用户仅允许新增两个访问密钥。
- 为保证访问密钥的安全，访问密钥仅在初次生成时自动下载，后续不可再次通过管理控制台界面获取。请在生成后妥善保管。

使用AK/SK认证时，您可以基于签名算法使用AK/SK对请求进行签名，也可以使用专门的签名SDK对请求进行签名。详细的签名方法和SDK使用方法请参见**API签名指南**。

须知

签名SDK只提供签名功能，与服务提供的SDK不同，使用时请注意。

您也可以通过这个视频教程了解AK/SK认证的使用：**https://bbs.huaweicloud.com/videos/100697** 。

3.3 返回结果

请求发送以后，您会收到响应，包含：状态码、响应消息头和响应消息体。

状态码

状态码是一组从1xx到5xx的数字代码，状态码表示了请求响应的状态，完整的状态码列表请参见**状态码**。

对于“获取用户Token”接口，如果调用后返回状态码为“201”，则表示请求成功。

响应消息头

对应请求消息头，响应同样也有消息头，如“Content-type”。

表 3-4 公共响应消息头

消息头名称	描述	是否必选
Content-Type	用于指明发送给接收者的实体正文的媒体类型。 类型：字符串。 默认值：application/json; charset=UTF-8	是

消息头名称	描述	是否必选
X-request-id	此字段携带请求ID号，以便任务跟踪。 类型：字符串。request_id-timestamp-hostname（request_id在服务器端生成UUID，timestamp为当前时间戳，hostname为处理当前接口的服务器名称）。 默认值：无。	否
X-ratelimit	此字段携带总计流控请求数。 类型：整型。 默认值：无。	否
X-ratelimit-used	此字段携带剩下请求数。 类型：整型。 默认值：无。	否
X-ratelimit-window	此字段携带流控单位。 类型：字符串。单位按照分钟、小时、天。 默认值：小时。	否

对于“获取用户Token”接口，返回如图3-3所示的消息头。

其中“x-subject-token”就是需要获取的用户Token。有了Token之后，您就可以使用Token认证调用其他API。

图 3-3 获取用户 Token 响应消息头

connection → keep-alive

content-type → application/json

date → Tue, 12 Feb 2019 06:52:13 GMT

server → Web Server

strict-transport-security → max-age=31536000; includeSubdomains;

transfer-encoding → chunked

via → proxy A

x-content-type-options → nosniff

x-download-options → noopen

x-frame-options → SAMEORIGIN

x-iam-trace-id → 218d45ab-d674-4995-af3a-2d0255ba41b5

x-subject-token
→ MIIYXQYJKoZIhvcNAQcCoIIYTjCCGEoCAQExDTALBglghkgBZQMEAgEwgharBgkqhkiG9w0BBwGgghacBIIWmHsidG9rZW4iOnsiZXhwaXJlc19hdCI6IjIwMTktMDItMTNUMD
fj3KJs6YgKnpVNRbW2eZ5eb78SZOkqjACgklqO1wi4JlGzrpd18LGXK5txldfq4lqHCYb8P4NaY0NYejcAgzJVeFIYtLWT1GSO0zxKZmlQHQj82HBqHdglZO9fuEbL5dMhdavj+33wEI
xHRCE9l87o+k9-
j+CMZSEB7bUGd5Uj6eRASXI1jipPEGA270g1FruooL6jqglFkNPQuFSOU8+uSsttVwRtNfsC+qTp22Rkd5MCqFGQ8LcuUxC3a+9CMBnOintWW7oeRUVhVpxk8pxiX1wTEboX-
RzT6MUbpvGw-oPNFYxJECKnoH3HRozv0vN--n5d6Nbxg==

x-xss-protection → 1; mode=block;

响应消息体

响应消息体通常以结构化格式返回，与响应消息头中Content-type对应，传递除响应消息头之外的内容。

对于“获取用户Token”接口，返回如下消息体。为篇幅起见，这里只展示部分内容。

```
{
    "token": {
        "expires_at": "2019-02-13T06:52:13.855000Z",
        "methods": [
            "password"
        ],
        "catalog": [
            {
                "endpoints": [
                    {
                        "region_id": "cn-north-4",
.....
```

当接口调用出错时，会返回错误码及错误信息说明，错误响应的Body体格式如下所示。

```
{
    "error_msg": "The format of message is error",
    "error_code": "AS.0001"
}
```

其中，error_code表示错误码，error_msg表示错误描述信息，具体请参见**错误码**。

4 应用示例

本节通过cURL调用CDM API，迁移本地MySQL数据库中的数据到云上服务DWS为例，介绍使用CDM API的基本流程。

1. **获取token**

   获取用户的token，因为在后续的请求中需要将token放到请求消息头中作为认证。

2. **创建CDM集群**

   - 如果您已经创建过CDM集群，可以跳过该步骤，直接使用已创建的集群ID。
   - 如果您需要使用新的集群执行迁移任务，调用**创建集群**API创建。

3. **创建连接**

   调用**创建连接**API创建MySQL连接和DWS连接。

4. **创建迁移作业**

   调用**指定集群创建作业**API创建MySQL到DWS的迁移作业。

5. **查看作业结果**

   调用**启动作业**API开始执行作业。

准备数据

在调用API之前，您需要准备如下数据。

表 4-1 准备数据

数据项	名称	说明	样例
云账户信息	项目名	CDM所属的项目名。	Project Name
	项目ID	CDM所属的项目ID。	1551c7f6c808414d8e9f3c514a170f2e
	账号名	用户所属的企业账户名称。	Account Name
	用户名	使用云服务的用户名，该用户需要拥有CDM的操作权限。	Username

数据项	名称	说明	样例
	密码	用户密码。	password
VPC信息	VPC的ID	CDM所属的VPC必须与DWS一致。	6b47302a-bf79-4b20-bf7a-80987408e196
	子网ID	CDM所属的子网必须与DWS一致。	63bdc3cb-a4e7-486f-82ee-d9bf208c8f8c
	安全组ID	CDM所属的安全组必须与DWS一致。	005af77a-cce5-45ac-99c7-2ea50ea8addf
Endpoint	IAM的Endpoint	终端节点（Endpoint）即调用API的**请求地址**，不同服务不同区域的终端节点不同。Endpoint您可以从**终端节点及区域说明**获取。	iam_endpoint
	CDM的Endpoint	终端节点（Endpoint）即调用API的**请求地址**，不同服务不同区域的终端节点不同。本服务的Endpoint您可以从**终端节点Endpoint**获取。	cdm_endpoint
MySQL数据库	IP地址	本地的MySQL数据库的IP地址，且该地址允许CDM通过公网IP访问。	1xx.120.85.24
	端口	MySQL数据库的端口。	3306
	数据库名称	待导出数据的MySQL数据库名称。	DB_name
	用户名	访问MySQL数据库的用户，该用户拥有MySQL数据库的读、写和删除权限。	username
	密码	访问MySQL数据库的用户密码。	DB_password
DWS数据库	IP地址	DWS数据库的IP地址，CDM可通过内网访问该地址。	10.120.85.24
	端口	DWS数据库的端口。	3306
	数据库名称	待写入数据的DWS数据库名称。	DWS

数据项	名称	说明	样例
	用户名	访问DWS数据库的用户，该用户拥有DWS数据库的读、写和删除权限。	user_dws
	密码	访问DWS数据库的用户密码。	dws_password

获取 token

1. 调用其他API前，需要获取token，并设置成环境变量。

```
curl -H "Content-Type:application/json" https://{iam_endpoint}/v3/auth/tokens -X POST -d '
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "Username",
                    "password": "password",
                    "domain": {
                        "name": "Account Name"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "id": "1551c7f6c808414d8e9f3c514a170f2e"
            }
        }
    }
}
' -v -k
```

响应Header中“X-Subject-Token”的值即为Token：

```
X-Subject-Token:MIIDkgYJKoZIhvcNAQcCoIIDgzCCA38CAQExDTALBglghkgBZQMEAgEwgXXXXX...
```

2. 使用如下命令将token设置为环境变量，方便后续事项。

```
export Token = MIIDkgYJKoZIhvcNAQcCoIIDgzCCA38CAQExDTALBglghkgBZQMEAgEwgXXXXX...
```

创建 CDM 集群

1. 调用**创建集群**API创建集群，假设集群详情如下：
   - 集群名称为“cdm-ab82”。
   - 集群规格为“cdm.medium”。
   - VPC、子网、安全组与DWS一致，且自动绑定弹性IP。

   如果返回状态码为200，则说明创建命令执行成功。

```
curl -X POST -H 'Content-Type:application/json;charset=utf-8' -H "X-Auth-Token:$Token" -d '
{
  "cluster": {
    "name": "cdm-ab82",
    "vpcId": "6b47302a-bf79-4b20-bf7a-80987408e196",
    "instances": [{
          "flavorRef": "fb8fe666-6734-4b11-bc6c-43d11db3c745",
          "nics": [{
```

```
            "net-id": "63bdc3cb-a4e7-486f-82ee-d9bf208c8f8c",
            "securityGroupId": "005af77a-cce5-45ac-99c7-2ea50ea8addf"
        }],
        "availability_zone": "Project Name",
        "type": "cdm"
    }],
  "datastore": {
        "version": "1.8.5",
        "type": "cdm"
    },
  "isScheduleBootOff": false,
  "scheduleBootTime": "null",
  "scheduleOffTime": "null",
  "isAutoOff": false,
  "sys_tags": [{
        "key": "_sys_enterprise_project_id",
        "value": "1ce45885-4033-40d2-bdde-d4dbaceb387d"
    }]
  },
 "autoRemind": false,
 "phoneNum": "null",
 "email": "null"
}'
https://{cdm_endpoint}/v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters -v -k
```

2. 调用**查询集群列表**查询集群信息，获取集群的ID，并设置为全局变量。

```
curl -X GET -H 'Content-Type:application/json;charset=utf-8' -H "X-Auth-Token:$Token" https://
{cdm_endpoint}/v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters -k -v
```

获取响应如下所示。

```
{
 "clusters": [{
  "version": "x.x.x",
  "updated": "2018-09-05T08:38:25",
  "name": "cdm-ab82",
  "created": "2018-09-05T08:38:25",
  "id": "bae65496-643e-47ca-84af-948672de7eeb",
  "status": "200",
  "isFrozen": "0",
  "statusDetail": "Normal",
  "actionProgress": {},
  "config_status": "In-Sync"
 }]
}
```

“status”的状态如果为**200**则表示集群创建成功，集群对应的ID为**bae65496-643e-47ca-84af-948672de7eeb**。

3. 使用如下命令将集群对应的ID设置为全局变量，方便后续事项。

```
export ID = bae65496-643e-47ca-84af-948672de7eeb
```

创建连接

1. 调用**创建连接**API创建MySQL连接，连接名称为*mysql_link*。这里假设本地MySQL数据库信息如下：
   - IP地址为*1xx.120.85.24*。
   - 端口为*3306*。
   - 数据库名称为*DB_name*。
   - 登录用户为*username*。
   - 密码为*DB_password*。

   如果返回状态码为200，则说明创建命令执行成功。

```
curl -X POST -H "Content-Type:application/json" -H "X-Auth-Token:$Token" -d '{
 "links": [{
  "enabled": true,
```

```
  "update-user": null,
  "name": "mysql_link",
  "link-config-values": {
    "configs": [
                {
                    "name": "linkConfig",
                    "inputs": [
                      {
                        "name": "linkConfig.databaseType",
                        "value": "MYSQL"
                      },
                      {
                        "name": "linkConfig.host",
                        "value": "1xx.120.85.24"
                      },
                      {
                        "name": "linkConfig.port",
                        "value": "3306"
                      },
                      {
                        "name": "linkConfig.database",
                        "value": "DB_name"
                      },
                      {
                        "name": "linkConfig.username",
                        "value": "username"
                      },
                      {
                        "name": "linkConfig.password",
                        "value": "DB_password"
                      },
                      {
                        "name": "linkConfig.fetchSize",
                        "value": "100000"
                      },
                      {
                        "name": "linkConfig.usingNative",
                        "value": "true"
                      }
                    ]
                }
            ]
  },
  "connector-name": "generic-jdbc-connector",
  "creation-date": 1536654788622,
  "update-date": 1536654788622,
  "creation-user": null
 }]
}'
https://{cdm_endpoint}/v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/
bae65496-643e-47ca-84af-948672de7eeb/cdm/link -k -v
```

2. 调用**创建连接**API创建DWS连接，连接名称为*dws_link*。这里假设DWS数据库信息如下：
   - 数据库的IP地址为*10.120.85.24*。
   - 端口为*3306*。
   - 数据库的名称为*DWS*。
   - 登录用户为*user_dws*。
   - 密码为*dws_password*。

```
curl -X POST -H "Content-Type:application/json" -H "X-Auth-Token:$Token" -d '{
 "links": [{
  "enabled": true,
  "update-user": null,
  "name": "dws_link",
  "link-config-values": {
    "configs": [
```

```
            {
                "name": "linkConfig",
                "inputs": [
                    {
                        "name": "linkConfig.databaseType",
                        "value": "DWS"
                    },
                    {
                        "name": "linkConfig.host",
                        "value": "10.120.85.24"
                    },
                    {
                        "name": "linkConfig.port",
                        "value": "3306"
                    },
                    {
                        "name": "linkConfig.database",
                        "value": "DWS"
                    },
                    {
                        "name": "linkConfig.username",
                        "value": "user_dws"
                    },
                    {
                        "name": "linkConfig.password",
                        "value": "dws_password"
                    },
                    {
                        "name": "linkConfig.fetchSize",
                        "value": "100000"
                    },
                    {
                        "name": "linkConfig.usingNative",
                        "value": "true"
                    }
                ]
            }
        ]
  },
  "connector-name": "generic-jdbc-connector",
  "creation-date": 1536654788622,
  "update-date": 1536654788622,
  "creation-user": null
 }]
}'
https://{cdm_endpoint}/v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/
bae65496-643e-47ca-84af-948672de7eeb/cdm/link -k -v
```

创建迁移作业

1. 连接创建成功后，调用**指定集群创建作业**API创建迁移作业，作业样例如下：
   - 作业名称：mysql2dws。
   - 从MySQL导出数据的数据库名称为*default*，导出的表名称为*mysql_tbl*，依据*id*字段将作业分割为多个任务并发执行。
   - 导入DWS的数据库名称为*public*，表名为*cdm_all_type*，导入前不清空数据。
   - 当DWS数据库里没有本地MySQL数据库中的表时，CDM自动在DWS端创建该表。
   - DWS端加载的字段列表为*id&gid&name*。
   - 作业抽取数据时，并发执行的Extractor数量为3。

   如果返回状态码为200，则说明创建命令执行成功。

```
curl -X POST -H "Content-Type:application/json" -H "X-Cluster-ID:$ID" -H "X-Auth-Token:$Token" -d '{
 "jobs": [{
```

```
  "job_type": "NORMAL_JOB",
  "name": "mysql2dws",
  "from-link-name": "mysql_link",
  "from-connector-name": "generic-jdbc-connector",
  "to-link-name": "dws_link",
  "to-connector-name": "generic-jdbc-connector",
  "from-config-values": {
    "configs": [{
      "name": "fromJobConfig",
      "inputs": [{
        "name": "fromJobConfig.schemaName",
        "value": "default"
      },
      {
        "name": "fromJobConfig.tableName",
        "value": "mysql_tbl"
      },
      {
        "name": "fromJobConfig.partitionColumn",
        "value": "id"
      }]
    }]
  },
"to-config-values": {
          "configs": [
             {
                "inputs": [
                   {
                      "name": "toJobConfig.schemaName",
                      "value": "public"
                   },
                   {
                      "name": "toJobConfig.tablePreparation",
                      "value": "CREATE_WHEN_NOT_EXIST"
                   },
                   {
                      "name": "toJobConfig.tableName",
                      "value": "cdm_all_type"
                   },
                   {
                      "name": "toJobConfig.columnList",
                      "value": "id&gid&name"
                   },
                   {
                      "name": "toJobConfig.shouldClearTable",
                      "value": "false"
                   }
                ],
                "name": "toJobConfig"
             }
          ]
       },
  "driver-config-values": {
    "configs": [{
      "name": "throttlingConfig",
      "inputs": [{
        "name": "throttlingConfig.numExtractors",
        "value": "3"
      }]
    }]
  }
 }]
}' https://{cdm_endpoint}/v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/bae65496-643e-47ca-84af-948672de7eeb/cdm/job -k -v
```

2. 调用启动作业API开始执行作业。

```
curl -X GET -H 'Content-Type:application/json;charset=utf-8' -H "X-Cluster-ID:$ID" -H "X-Auth-Token:$Token" https://{cdm_endpoint}/v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/bae65496-643e-47ca-84af-948672de7eeb/cdm/job/mysql2dws/start -k -v
```

响应如下：

```
{
  "submissions": [{
    "progress": 1,
    "job-name": "mysql2dws",
    "status": "BOOTING",
    "creation-date": 1536654788622,
    "creation-user": "cdm"
  }]
}
```

查看作业结果

1. 调用**查询作业状态**API查询作业状态。

```
curl -X GET -H 'Content-Type:application/json;charset=utf-8' -H "X-Cluster-ID:$ID" -H "X-Auth-Token:$Token" https://{cdm_endpoint}/v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/job/mysql2dws/status -k -v
```

2. 查看作业执行结果，作业执行成功的响应如下：

```
{
  "submissions": [{
    "progress": 0,
    "job-name": "mysql2dws",
    "status": "SUCCEEDED",
    "creation-date": 1536654788622,
    "creation-user": "cdm",
    "isStopingIncrement": "",
    "last-update-date": 1536654888622,
    "is-execute-auto": false,
    "last-udpate-user": "cdm",
    "isDeleteJob": false,
    "isIncrementing": false,
    "external-id": "job_local1127970451_0009",
    "counters": {
      "org.apache.sqoop.submission.counter.SqoopCounters": {
        "BYTES_WRITTEN": -1,
        "TOTAL_FILES": -1,
        "BYTES_READ": -1,
        "FILES_WRITTEN": -1,
        "TOTAL_SIZE": -1,
        "FILES_READ": -1,
        "ROWS_WRITTEN": 80,
        "ROWS_READ": 80
      }
    }
  }]
}
```

说明

- BYTES_WRITTEN：表示写入的字节数。
- BYTES_READ：表示读取的字节数。
- TOTAL_FILES：表示总文件数。
- FILES_WRITTEN：表示写入的文件数。
- FILES_READ：表示读取的文件数。
- ROWS_WRITTEN：表示写入成功的行数。
- ROWS_READ：表示读取成功的行数。

5 API

5.1 集群管理

5.1.1 查询集群详情

功能介绍

查询集群详情接口。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/clusters/{cluster_id}

表 5-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。

请求参数

表 5-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码：200

表 5-3 响应 Body 参数

参数	参数类型	描述
publicEndpoint	String	集群绑定的EIP。
instances	Array of **ClusterDetailInstance** objects	集群的节点信息，请参见instances参数说明。
security_group_id	String	安全组id。
subnet_id	String	子网id。
vpc_id	String	虚拟私有云ID。
customerConfig	**CustomerConfig** object	用户配置信息。
datastore	**Datastore** object	cdm信息。
isAutoOff	Boolean	自动关机。
publicEndpointDomainName	String	集群绑定的EIP域名。
bakExpectedStartTime	String	开始时间。
bakKeepDay	Integer	保留时间。
maintainWindow	**maintainWindow** object	维护窗口。
recentEvent	Integer	事件数。
flavorName	String	规格名称。
azName	String	az名称。

参数	参数类型	描述
endpointDomainName	String	对端域名。
publicEndpointStatus	**publicEndpointStatus** object	EIP状态信息。
isScheduleBootOff	Boolean	选择是否启用定时开关机功能。定时开关机功能和自动关机功能不可同时开启。
namespace	String	命名空间。
eipId	String	弹性ip id。
failedReasons	**FailedReasons** object	失败原因。集群处于正常状态时不返回。
dbuser	String	数据库用户。
links	Array of **ClusterLinks** objects	集群连接信息。
clusterMode	String	集群模式：sharding(分片集群)。
task	**ClusterTask** object	任务信息。
created	String	集群创建时间，格式为ISO8601：YYYY-MM-DDThh:mm:ssZ。
statusDetail	String	集群状态描述：Normal（正常）。
config_status	String	集群配置状态： • In-Sync：配置已同步。 • Applying：配置中。 • Sync-Failure：配置失败。
actionProgress	**ActionProgress** object	集群操作进度，任务信息，由key、value组成。key值为正在进行的任务，value值为正在进行任务的进度。示例如"action_progress": {"SNAPSHOTTING":"16%"}。
name	String	集群名称。
id	String	集群ID。
isFrozen	String	集群是否冻结：0：否，1：是。
actions	Array of strings	集群配置状态：In-Sync：配置已同步。Applying：配置中。Sync-Failure：配置失败。

参数	参数类型	描述
updated	String	集群更新时间，格式为 ISO8601：YYYY-MM-DDThh:mm:ssZ。
status	String	集群状态： • 100：创建中。 • 200：正常。 • 300：失败。 • 303：创建失败。 • 800：冻结。 • 900：已关机。 • 910：正在关机。 • 920：正在开机。

表 5-4 ClusterDetailInstance

参数	参数类型	描述
flavor	**flavor** object	节点的虚拟机规格，请参见flavor参数说明（查询集群列表时返回值为null）。
volume	**volume** object	节点的磁盘信息，请参见volume参数说明（查询集群列表时返回值为null）。
status	String	节点状态： • 100：创建中。 • 200：正常。 • 300：失败。 • 303：创建失败。 • 400：已删除。 • 800：冻结。
actions	Array of strings	节点操作状态列表： • REBOOTING：重启中。 • RESTORING：恢复中。 • REBOOT_FAILURE：重启失败。
type	String	节点类型，只支持一种类型“cdm”。
id	String	节点的虚拟机ID。
name	String	节点的虚拟机名称。
isFrozen	String	节点是否冻结：0：否。1：是。
components	String	组件。

参数	参数类型	描述
config_status	String	节点配置状态（查询集群列表时为null）： • In-Sync：配置已同步。 • Applying：配置中。 • Sync-Failure：配置失败。
role	String	实例角色。
group	String	分组。
links	Array of ClusterLinks objects	链接信息（查询集群列表时返回值为null）。
paramsGroupId	String	组件分组id。
publicIp	String	公网ip。
manageIp	String	管理ip。
trafficIp	String	流量ip。
shard_id	String	分片id。
manage_fix_ip	String	管理修复ip。
private_ip	String	私有ip。
internal_ip	String	内部ip。
resource	Array of Resource objects	资源信息（查询集群列表时返回值为null）。

表 5-5 flavor

参数	参数类型	描述
id	String	节点虚拟机的规格ID。
links	Array of ClusterLinks objects	链接信息。

表 5-6 volume

参数	参数类型	描述
type	String	节点的磁盘类型，只支持本地磁盘。
size	Long	节点磁盘大小，单位G。

表 5-7 Resource

参数	参数类型	描述
resource_id	String	资源id。
resource_type	String	资源类型：server(服务器)。

表 5-8 CustomerConfig

参数	参数类型	描述
failureRemind	String	失败提醒。
clusterName	String	集群类型。
serviceProvider	String	服务提供。
localDisk	String	是否本地磁盘。
ssl	String	是否使用ssl。
createFrom	String	创建来源。
resourceId	String	资源ID。
flavorType	String	规格类型。
workSpaceId	String	工作空间ID。
trial	String	适用。

表 5-9 Datastore

参数	参数类型	描述
type	String	类型，一般为cdm。
version	String	集群版本。

表 5-10 maintainWindow

参数	参数类型	描述
day	String	周几。
startTime	String	开始时间。
endTime	String	结束时间。

表 5-11 publicEndpointStatus

参数	参数类型	描述
status	String	状态。
errorMessage	String	错误信息。

表 5-12 FailedReasons

参数	参数类型	描述
CREATE_FAILED	**CREATE_FAILED** object	集群创建失败原因。

表 5-13 CREATE_FAILED

参数	参数类型	描述
errorCode	String	错误码。
errorMsg	String	失败原因。

表 5-14 ClusterLinks

参数	参数类型	描述
rel	String	关系。
href	String	链接地址。

表 5-15 ClusterTask

参数	参数类型	描述
description	String	任务描述。
id	String	任务id。
name	String	任务名称。

表 5-16 ActionProgress

参数	参数类型	描述
CREATING	String	创建集群进度，例如：29%。

参数	参数类型	描述
GROWING	String	扩容集群进度，例如：29%。
RESTORING	String	恢复集群进度，例如：29%。
SNAPSHOTTING	String	集群快照进度，例如：29%。
REPAIRING	String	修复集群进度，例如：29%。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/bae65496-643e-47ca-84af-948672de7eeb
```

响应示例

状态码：200

OK。

```
{
  "publicEndpoint" : "49.xx.xx.10",
  "instances" : [ {
    "flavor" : {
      "id" : "fb8fe666-6734-4b11-bc6c-43d11db3c745"
    },
    "volume" : {
      "size" : "100",
      "type" : "LOCAL_DISK"
    },
    "name" : "cdm-c018",
    "id" : "635dce67-3df8-4756-b4c7-90e45e687367",
    "isFrozen" : "0",
    "type" : "cdm",
    "actions" : "REBOOTING",
    "config_status" : "In-Sync",
    "status" : "200"
  } ],
  "created" : "2018-09-05T08:38:25",
  "statusDetail" : "Normal",
  "actionProgress" : { },
  "name" : "cdm-c018",
  "id" : "bae65496-643e-47ca-84af-948672de7eeb",
  "isFrozen" : "0",
  "actions" : "REBOOTING",
  "updated" : "2018-09-05T08:38:25",
  "status" : "200"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
```

```
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ShowClusterDetailSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowClusterDetailRequest request = new ShowClusterDetailRequest();
        request.withClusterId("{cluster_id}");
        try {
            ShowClusterDetailResponse response = client.showClusterDetail(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
```

```
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowClusterDetailRequest()
        request.cluster_id = "{cluster_id}"
        response = client.show_cluster_detail(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowClusterDetailRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ShowClusterDetail(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.2 删除集群

功能介绍

删除集群接口。

调用方法

请参见**如何调用API**。

URI

DELETE /v1.1/{project_id}/clusters/{cluster_id}

表 5-17 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。

请求参数

表 5-18 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 5-19 请求 Body 参数

参数	是否必选	参数类型	描述
keep_last_manual_backup	是	Integer	日志备份数，填写为默认填0即可。

响应参数

状态码：202

表 5-20 响应 Body 参数

参数	参数类型	描述
jobId	String	作业ID。

请求示例

```
DELETE /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920

{
  "keep_last_manual_backup" : 0
}
```

响应示例

状态码：202

Accepted。

```
{
  "jobId" : "ff8080815e55125a015e552eddba001a"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class DeleteClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteClusterRequest request = new DeleteClusterRequest();
        request.withClusterId("{cluster_id}");
        CdmDeleteClusterReq body = new CdmDeleteClusterReq();
        body.withKeepLastManualBackup(0);
        request.withBody(body);
        try {
            DeleteClusterResponse response = client.deleteCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
```

```
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteClusterRequest()
        request.cluster_id = "{cluster_id}"
        request.body = CdmDeleteClusterReq(
            keep_last_manual_backup=0
        )
        response = client.delete_cluster(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteClusterRequest{}
    request.ClusterId = "{cluster_id}"
    request.Body = &model.CdmDeleteClusterReq{
        KeepLastManualBackup: int32(0),
    }
    response, err := client.DeleteCluster(request)
```

```
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
202	Accepted。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.3 查询所有可用区

功能介绍

查询CDM集群的所有可用区。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/regions/{region_id}/availability_zones

表 5-21 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。

参数	是否必选	参数类型	描述
region_id	是	String	通过调用IAM服务的“查询区域列表”接口获取响应消息中的区域ID。

请求参数

表 5-22 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token，使用Token认证时必选。通过调用IAM服务的“获取用户Token”接口获取响应消息头中X-Subject-Token的值。

响应参数

状态码：200

表 5-23 响应 Body 参数

参数	参数类型	描述
regionId	String	区域ID。
defaultAZ	String	默认可用区。
availableZones	Array of CdmClusterAvailabilityZone objects	可用区。

表 5-24 CdmClusterAvailabilityZone

参数	参数类型	描述
availableZoneId	String	可用区ID。
availableZoneName	String	可用区名称。
availableZoneCode	String	可用区码。
azStatus	String	可用区状态。
type	String	可用区类型。

参数	参数类型	描述
tags	Object	可用区tag。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/regions/xxx-xxx-xxx/availability_zones
```

响应示例

状态码：200

请求成功。

```
{
  "regionId" : "xxx-xxx-xxx",
  "defaultAZ" : "xxx-xxx-xxx",
  "availableZones" : [ {
    "availableZoneId" : "xxx-xxx-xxx",
    "availableZoneName" : "xxx-xxx-xxx",
    "availableZoneCode" : "xxx-xxx-xxx",
    "azStatus" : "Available",
    "type" : null,
    "tags" : null
  } ]
}
```

状态码

状态码	描述
200	请求成功。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.4 查询支持的版本

功能介绍

查询CDM集群支持的版本。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/datastores

表 5-25 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。

请求参数

表 5-26 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token，使用**Token认证**时必选。通过调用IAM服务的“获取用户Token”接口获取响应消息头中X-Subject-Token的值。

响应参数

状态码：200

表 5-27 响应 Body 参数

参数	参数类型	描述
datastores	Array of **CdmClusterDatastore** objects	数据库列表。

表 5-28 CdmClusterDatastore

参数	参数类型	描述
id	String	服务ID，用于区分不同服务。
name	String	服务名称。
bigclusterEnable	Boolean	是否支持大规格集群。

参数	参数类型	描述
defaultVersion	String	默认版本。
versions	Array of **CdmClusterVersion** objects	版本。

表 5-29 CdmClusterVersion

参数	参数类型	描述
active	String	版本状态。
id	String	版本ID。
image	String	版本镜像。
name	String	版本名称。
packages	String	版本的包。
datastore	String	服务ID，用于区分不同服务。
links	Array of **ClusterLinks** objects	链接信息。

表 5-30 ClusterLinks

参数	参数类型	描述
rel	String	关系。
href	String	链接地址。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/datastores
```

响应示例

状态码：200

请求成功。

```
[ {
  "id" : "736270b9-27c7-4f03-823b-447d8245e1c2",
  "name" : "cdm",
  "bigclusterEnable" : false,
  "defaultVersion" : "2.9.3.300",
  "links" : null,
  "versions" : [ {
    "active" : "1",
```

```
    "id" : "e8a8b8cc-63f8-4fb5-8d4a-24c502317b11",
    "image" : null,
    "links" : [ {
      "rel" : "self",
      "href" : "https://10.63.25.93:443/rds/v1.0/datastores/736270b9-27c7-4f03-823b-447d8245e1c2"
    }, {
      "rel" : "bookmark",
      "href" : "https://10.63.25.93:443/datastores/736270b9-27c7-4f03-823b-447d8245e1c2"
    } ],
    "name" : "2.9.3.300",
    "packages" : "cdm",
    "datastore" : "736270b9-27c7-4f03-823b-447d8245e1c2"
  } ]
} ]
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ShowDatastoresSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowDatastoresRequest request = new ShowDatastoresRequest();
        try {
            ShowDatastoresResponse response = client.showDatastores(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
```

```
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowDatastoresRequest()
        response = client.show_datastores(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()
```

```
    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowDatastoresRequest{}
    response, err := client.ShowDatastores(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求成功。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.5 查询版本规格

功能介绍

按版本ID查询所有兼容规格。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/datastores/{datastore_id}/flavors

表 5-31 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
datastore_id	是	String	版本ID，获取方法请参见**CDM支持的版本**。

请求参数

表 5-32 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token，使用**Token认证**时必选。通过调用IAM服务的“获取用户Token”接口获取响应消息头中X-Subject-Token的值。

响应参数

状态码：200

表 5-33 响应 Body 参数

参数	参数类型	描述
id	String	服务ID，用于区分不同服务。
dbname	String	db名称，一般为cdm。
versions	Array of **CdmClusterDatastoreVersion** objects	版本信息列表。

表 5-34 CdmClusterDatastoreVersion

参数	参数类型	描述
id	String	版本ID。
name	String	版本名称。
flavors	Array of **CdmClusterFlavor** objects	规格信息。

表 5-35 CdmClusterFlavor

参数	参数类型	描述
cpu	Integer	CPU。
ram	Integer	内存。
name	String	规格名称。
region	String	region。
typename	String	类型名称。
clusterMode	String	集群模式。
status	String	规格状态。
str_id	String	规格ID。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/datastores/736270b9-27c7-4f03-823b-447d8245e1c2/flavors
```

响应示例

状态码：200

请求成功。

```
{
  "id" : "736270b9-27c7-4f03-823b-447d8245e1c2",
  "dbname" : "cdm",
  "versions" : [ {
    "id" : "e8a8b8cc-63f8-4fb5-8d4a-24c502317b11",
    "name" : "2.9.3.300",
    "flavors" : [ {
      "cpu" : 4,
      "ram" : 8,
      "name" : "cdm.small",
      "region" : "xxx-xxx-xxx",
      "typename" : "cdm",
      "clusterMode" : "sharding",
      "status" : "abandon",
      "str_id" : "a79fd5ae-1833-448a-88e8-3ea2b913e1f6"
    } ]
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
```

```
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ShowFlavorsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowFlavorsRequest request = new ShowFlavorsRequest();
        request.withDatastoreId("{datastore_id}");
        try {
            ShowFlavorsResponse response = client.showFlavors(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)
```

```
    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowFlavorsRequest()
        request.datastore_id = "{datastore_id}"
        response = client.show_flavors(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowFlavorsRequest{}
    request.DatastoreId = "{datastore_id}"
    response, err := client.ShowFlavors(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求成功。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.6 查询规格详情

功能介绍

查询指定规格ID的规格详情。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/flavors/{flavor_id}

表 5-36 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
flavor_id	是	String	规格ID，获取方法请参见**查询版本规格**。

请求参数

表 5-37 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token，使用**Token认证**时必选。通过调用IAM服务的“获取用户Token”接口获取响应消息头中X-Subject-Token的值。

响应参数

状态码：200

表 5-38 响应 Body 参数

参数	参数类型	描述
name	String	规格名称。
str_id	String	规格ID。
flavor_detail	Array of **flavorAttribute** objects	规格详细列表。

表 5-39 flavorAttribute

参数	参数类型	描述
name	String	规格属性名称，如mem、cpu。
value	String	规格属性值。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/flavors/a79fd5ae-1833-448a-88e8-3ea2b913e1f6
```

响应示例

状态码：200

请求成功。

```
{
  "str_id" : "a79fd5ae-1833-448a-88e8-3ea2b913e1f6",
  "name" : "cdm.large",
  "flavor_detail" : [ {
    "name" : "cpu",
    "value" : 8
  }, {
```

```
    "name" : "mem",
    "value" : 16
  }, {
    "name" : "volumeType",
    "value" : "SATA"
  }, {
    "name" : "flavor",
    "value" : "s6.2xlarge.2"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ShowFlavorDetailSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowFlavorDetailRequest request = new ShowFlavorDetailRequest();
        request.withFlavorId("{flavor_id}");
        try {
            ShowFlavorDetailResponse response = client.showFlavorDetail(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowFlavorDetailRequest()
        request.flavor_id = "{flavor_id}"
        response = client.show_flavor_detail(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
            WithCredential(auth).
            Build())

    request := &model.ShowFlavorDetailRequest{}
    request.FlavorId = "{flavor_id}"
    response, err := client.ShowFlavorDetail(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求成功。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.7 查询所有集群的企业项目 ID

功能介绍

查询当前项目下的所有集群的企业项目ID。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/enterprise-projects

表 5-40 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。

请求参数

表 5-41 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token，使用**Token认证**时必选。通过调用IAM服务的“获取用户Token”接口获取响应消息头中X-Subject-Token的值。

响应参数

状态码：200

表 5-42 响应 Body 参数

参数	参数类型	描述
resources	Array of **CdmClusterEnterpriseProject** objects	集群企业项目列表。

表 5-43 CdmClusterEnterpriseProject

参数	参数类型	描述
cluster_id	String	集群ID。
sys_tags	Array of **sys_tags** objects	企业项目列表。

表 5-44 sys_tags

参数	参数类型	描述
value	String	企业项目ID。

参数	参数类型	描述
key	String	该值目前固定为“_sys_enterprise_project_id”。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/enterprise-projects
```

响应示例

状态码：200

请求成功。

```
{
  "resources" : [ {
    "cluster_id" : "2b2a676f-0b91-4ad5-8c24-ec61be586fd1",
    "sys_tags" : [ {
      "key" : "_sys_enterprise_project_id",
      "value" : "1ce45885-4033-40d2-bdde-d4dbaceb387d"
    } ]
  }, {
    "cluster_id" : "387f8b24-c4b0-4211-97fe-745d0e88dc88",
    "sys_tags" : [ {
      "key" : "_sys_enterprise_project_id",
      "value" : 0
    } ]
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ShowEnterpriseProjectsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
```

```
            .withProjectId(projectId)
            .withAk(ak)
            .withSk(sk);

    CdmClient client = CdmClient.newBuilder()
            .withCredential(auth)
            .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
            .build();
    ShowEnterpriseProjectsRequest request = new ShowEnterpriseProjectsRequest();
    try {
        ShowEnterpriseProjectsResponse response = client.showEnterpriseProjects(request);
        System.out.println(response.toString());
    } catch (ConnectionException e) {
        e.printStackTrace();
    } catch (RequestTimeoutException e) {
        e.printStackTrace();
    } catch (ServiceResponseException e) {
        e.printStackTrace();
        System.out.println(e.getHttpStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
  }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowEnterpriseProjectsRequest()
        response = client.show_enterprise_projects(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
```

```
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowEnterpriseProjectsRequest{}
    response, err := client.ShowEnterpriseProjects(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求成功。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.8 查询集群的企业项目 ID

功能介绍

查询指定集群的企业项目ID。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/clusters/{cluster_id}/enterprise-projects

表 5-45 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	CDM集群ID，获取方法请参见**查询集群列表**。

请求参数

表 5-46 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token，使用**Token认证**时必选。通过调用IAM服务的“获取用户Token”接口获取响应消息头中X-Subject-Token的值。

响应参数

状态码：200

表 5-47 响应 Body 参数

参数	参数类型	描述
sys_tags	Array of **sys_tags** objects	企业项目列表。

表 5-48 sys_tags

参数	参数类型	描述
value	String	企业项目ID。
key	String	该值目前固定为“_sys_enterprise_project_id”。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/2b2a676f-0b91-4ad5-8c24-ec61be586fd1/enterprise-projects
```

响应示例

状态码：200

请求成功。

```
{
  "sys_tags" : [ {
    "key" : "_sys_enterprise_project_id",
    "value" : "1ce45885-4033-40d2-bdde-d4dbaceb387d"
  } ]
}
```

状态码

状态码	描述
200	请求成功。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.9 查询集群实例信息

功能介绍

查询集群实例信息。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/instances/{instance_id}

表 5-49 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
instance_id	是	String	实例ID，获取方法请参见**获取集群列表**。

请求参数

表 5-50 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token，使用**Token认证**时必选。通过调用IAM服务的“获取用户Token”接口获取响应消息头中X-Subject-Token的值。

响应参数

状态码：200

表 5-51 响应 Body 参数

参数	参数类型	描述
instance	**CdmQueryClusterInstanceDetail** object	实例信息。

表 5-52 CdmQueryClusterInstanceDetail

参数	参数类型	描述
configurationStatus	String	节点配置状态： ● In-Sync：配置已同步。 ● Applying：配置中。 ● Sync-Failure：配置失败。
paramsGroupId	String	配置ID。
type	String	配置服务类型，这里为cdm。
role	String	实例模式，这里为Standalone。
subnetid	String	实例的子网ID。
securegroup	String	安全组ID。
vpc	String	实例的VPC ID。
azcode	String	可用区名称。
region	String	局点名称。
created	String	实例创建时间，格式为ISO8601：YYYY-MM-DDThh:mm:ssZ。
updated	String	实例更新时间，格式为ISO8601：YYYY-MM-DDThh:mm:ssZ。
name	String	实例名称。
id	String	实例ID。
flavor	**flavor** object	节点的虚拟机规格，请参见flavor参数说明。
datastore	**Datastore** object	集群信息，请参见datastore参数说明。
dbuser	String	数据库用户，这里为cdm。
payModel	Integer	付费模式： ● 0：按需。 ● 1：包周期。
publicIp	String	集群绑定的公网地址。
trafficIp	String	集群的内网地址。
trafficIpv6	String	集群的内网IPv6地址。
cluster_id	String	集群ID。

表 5-53 flavor

参数	参数类型	描述
id	String	节点虚拟机的规格ID。
links	Array of **ClusterLinks** objects	链接信息。

表 5-54 ClusterLinks

参数	参数类型	描述
rel	String	关系。
href	String	链接地址。

表 5-55 Datastore

参数	参数类型	描述
type	String	类型，一般为cdm。
version	String	集群版本。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/instances/2c529048-ed06-4bcb-a48e-bf1800e1f496
```

响应示例

状态码：200

请求成功。

```
{
  "instance" : {
    "configurationStatus" : "In-Sync",
    "paramsGroupId" : "26084bb9-e74b-47d5-8be6-c0fbee9449d5",
    "type" : "cdm",
    "subnetid" : "9e4049b5-19a6-48fe-b5a2-2857c842fe56",
    "securegroup" : "560a3642-ddb1-4e93-a4bb-e484ae975127",
    "vpc" : "f35aee01-c4a3-47c1-8d92-9df430537de4",
    "azcode" : "xxx-xxx-xxxa",
    "region" : "xxx-xxx-xxx",
    "created" : "2018-09-05T08:38:25",
    "updated" : "2018-09-05T08:38:25",
    "name" : "test-cdm-dn-1-1",
    "id" : "2c529048-ed06-4bcb-a48e-bf1800e1f496",
    "flavor" : {
      "id" : "a79fd5ae-1833-448a-88e8-3ea2b913e1f6",
      "links" : [ ]
    },
    "datastore" : {
      "type" : "cdm",
```

```
    "version" : "2.9.3.300"
  },
  "dbuser" : "cdm",
  "payModel" : 0,
  "publicIp" : "49.xx.xx.10",
  "trafficIp" : "192.168.0.128",
  "trafficIpv6" : null,
  "cluster_id" : "2d9ac57e-3ebf-4557-86d5-89ae750ff61c"
 }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ShowInstanceDetailSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowInstanceDetailRequest request = new ShowInstanceDetailRequest();
        request.withInstanceId("{instance_id}");
        try {
            ShowInstanceDetailResponse response = client.showInstanceDetail(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowInstanceDetailRequest()
        request.instance_id = "{instance_id}"
        response = client.show_instance_detail(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
            WithCredential(auth).
            Build())

    request := &model.ShowInstanceDetailRequest{}
    request.InstanceId = "{instance_id}"
    response, err := client.ShowInstanceDetail(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求成功。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.10 修改集群

功能介绍

修改CDM集群配置。

调用方法

请参见**如何调用API**。

URI

POST /v1.1/{project_id}/cluster/modify/{cluster_id}

表 5-56 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	CDM集群ID，获取方法请参见**查询集群列表**。

请求参数

表 5-57 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式），有Body体的情况下必选，没有Body体无需填写。如果请求消息体中含有中文字符，则需要通过charset=utf8指定中文字符集，例如取值为：application/json;charset=utf8。
X-Auth-Token	是	String	用户Token，使用**Token认证**时必选。通过调用IAM服务的“获取用户Token”接口获取响应消息头中X-Subject-Token的值。
X-Language	是	String	请求语言。

表 5-58 请求 Body 参数

参数	是否必选	参数类型	描述
autoOff	否	Boolean	自动关机。
scheduleBootOff	否	Boolean	定时关机。
scheduleBootTime	否	String	定时开机。
scheduleOffTime	否	String	定时关机时间。
autoRemind	否	Boolean	消息通知。
phoneNum	否	String	手机号码，最多填写20个，以英文逗号分隔。

参数	是否必选	参数类型	描述
email	否	String	邮箱地址，最多填写20个，以英文逗号分隔。

响应参数

无

请求示例

修改集群配置。

```
POST /v1.1/1551c7f6c808414d8e9f3c514a170f2e/cluster/modify/bae65496-643e-47ca-84af-948672de7eeb

{
  "autoOff" : false,
  "scheduleBootOff" : true,
  "scheduleBootTime" : "00:00:00",
  "scheduleOffTime" : "10:00:00",
  "autoRemind" : true,
  "phoneNum" : "xxx",
  "email" : "xxx@xxx.com"
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

修改集群配置。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ModifyClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";
```

```
        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ModifyClusterRequest request = new ModifyClusterRequest();
        request.withClusterId("{cluster_id}");
        CdmModifyClusterReq body = new CdmModifyClusterReq();
        body.withEmail("xxx@xxx.com");
        body.withPhoneNum("xxx");
        body.withAutoRemind(true);
        body.withScheduleOffTime("10:00:00");
        body.withScheduleBootTime("00:00:00");
        body.withScheduleBootOff(true);
        body.withAutoOff(false);
        request.withBody(body);
        try {
            ModifyClusterResponse response = client.modifyCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

修改集群配置。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ModifyClusterRequest()
```

```
        request.cluster_id = "{cluster_id}"
        request.body = CdmModifyClusterReq(
            email="xxx@xxx.com",
            phone_num="xxx",
            auto_remind=True,
            schedule_off_time="10:00:00",
            schedule_boot_time="00:00:00",
            schedule_boot_off=True,
            auto_off=False
        )
        response = client.modify_cluster(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

修改集群配置。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ModifyClusterRequest{}
    request.ClusterId = "{cluster_id}"
    emailCdmModifyClusterReq:= "xxx@xxx.com"
    phoneNumCdmModifyClusterReq:= "xxx"
    autoRemindCdmModifyClusterReq:= true
    scheduleOffTimeCdmModifyClusterReq:= "10:00:00"
    scheduleBootTimeCdmModifyClusterReq:= "00:00:00"
    scheduleBootOffCdmModifyClusterReq:= true
    autoOffCdmModifyClusterReq:= false
    request.Body = &model.CdmModifyClusterReq{
        Email: &emailCdmModifyClusterReq,
        PhoneNum: &phoneNumCdmModifyClusterReq,
        AutoRemind: &autoRemindCdmModifyClusterReq,
        ScheduleOffTime: &scheduleOffTimeCdmModifyClusterReq,
        ScheduleBootTime: &scheduleBootTimeCdmModifyClusterReq,
```

```
        ScheduleBootOff: &scheduleBootOffCdmModifyClusterReq,
        AutoOff: &autoOffCdmModifyClusterReq,
    }
    response, err := client.ModifyCluster(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
202	请求成功。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部接口异常。
503	服务不可用。

错误码

请参见**错误码**。

5.1.11 重启集群

功能介绍

重启集群接口。

调用方法

请参见**如何调用API**。

URI

POST /v1.1/{project_id}/clusters/{cluster_id}/action

表 5-59 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。

请求参数

表 5-60 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 5-61 请求 Body 参数

参数	是否必选	参数类型	描述
restart	是	**restart** object	集群重启操作，定义哪些集群节点需要重启，请参见restart参数说明。

表 5-62 restart

参数	是否必选	参数类型	描述
restartDelayTime	否	Integer	重启时延，单位：秒。
restartMode	否	String	重启类型： • IMMEDIATELY：立即重启。 • FORCELY：强制重启。 • SOFTLY：一般重启。 默认值为“IMMEDIATELY”。 强制重启业务进程会中断，并重启集群的虚拟机。

参数	是否必选	参数类型	描述
restartLevel	否	String	重启级别： ● SERVICE：重启服务。 ● VM：重启虚拟机。 默认值为“SERVICE”。
type	是	String	集群节点类型，只支持“cdm”。
instance	否	String	预留字段，“restartLevel”为“SERVICE”时，“instance”必填，填空字串。
group	否	String	预留字段，“restartLevel”为“SERVICE”时，“group”必填，填空字串。

响应参数

状态码：200

表 5-63 响应 Body 参数

参数	参数类型	描述
jobId	String	作业ID。

请求示例

重启集群。

```
POST /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/bae65496-643e-47ca-84af-948672de7eeb/action

{
  "restart" : {
    "instance" : "",
    "type" : "cdm",
    "group" : ""
  }
}
```

响应示例

状态码：200

OK。

```
{
  "jobId" : "ff8080815e59d92d015e5b27ccb0004d"
}
```

SDK 代码示例

SDK代码示例如下。

Java

重启集群。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class RestartClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        RestartClusterRequest request = new RestartClusterRequest();
        request.withClusterId("{cluster_id}");
        CdmRestartClusterReq body = new CdmRestartClusterReq();
        CdmRestartClusterReqRestart restartbody = new CdmRestartClusterReqRestart();
        restartbody.withType("cdm")
            .withInstance("")
            .withGroup("");
        body.withRestart(restartbody);
        request.withBody(body);
        try {
            RestartClusterResponse response = client.restartCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

重启集群。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = RestartClusterRequest()
        request.cluster_id = "{cluster_id}"
        restartbody = CdmRestartClusterReqRestart(
            type="cdm",
            instance="",
            group=""
        )
        request.body = CdmRestartClusterReq(
            restart=restartbody
        )
        response = client.restart_cluster(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

重启集群。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
```

```
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.RestartClusterRequest{}
    request.ClusterId = "{cluster_id}"
    instanceRestart:= ""
    groupRestart:= ""
    restartbody := &model.CdmRestartClusterReqRestart{
        Type: "cdm",
        Instance: &instanceRestart,
        Group: &groupRestart,
    }
    request.Body = &model.CdmRestartClusterReq{
        Restart: restartbody,
    }
    response, err := client.RestartCluster(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.12 启动集群

功能介绍

启动集群接口。

调用方法

请参见**如何调用API**。

URI

POST /v1.1/{project_id}/clusters/{cluster_id}/action

表 5-64 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。

请求参数

表 5-65 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 5-66 请求 Body 参数

参数	是否必选	参数类型	描述
start	是	Object	集群启动操作，定义集群启动标识，为空对象。

响应参数

状态码：200

表 5-67 响应 Body 参数

参数	参数类型	描述
jobId	Array of strings	作业ID。

请求示例

启动集群。

```
POST /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/bae65496-643e-47ca-84af-948672de7eeb/action

{
  "start" : { }
}
```

响应示例

状态码：200

OK。

```
{
  "jobId" : [ "ff8080815e59d92d015e5b27ccb0004d" ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

启动集群。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class StartClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);
```

```
        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        StartClusterRequest request = new StartClusterRequest();
        request.withClusterId("{cluster_id}");
        CdmStartClusterReq body = new CdmStartClusterReq();
        body.withStart(new Object());
        request.withBody(body);
        try {
            StartClusterResponse response = client.startCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

启动集群。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = StartClusterRequest()
        request.cluster_id = "{cluster_id}"
        request.body = CdmStartClusterReq(
            start={}
        )
        response = client.start_cluster(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

启动集群。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.StartClusterRequest{}
    request.ClusterId = "{cluster_id}"
    var startCdmStartClusterReq interface{} = make(map[string]string)
    request.Body = &model.CdmStartClusterReq{
        Start: &startCdmStartClusterReq,
    }
    response, err := client.StartCluster(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。

状态码	描述
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.13 停止集群（待下线）

功能介绍

停止集群接口。

调用方法

请参见**如何调用API**。

URI

POST /v1.1/{project_id}/clusters/{cluster_id}/action

表 5-68 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。

请求参数

表 5-69 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 5-70 请求 Body 参数

参数	是否必选	参数类型	描述
stop	是	**stop** object	集群停止操作，定义集群停止标识，请参见stop参数说明。

表 5-71 stop

参数	是否必选	参数类型	描述
stopMode	是	String	关机类型： ● IMMEDIATELY：立即关机。 ● GRACEFULLY：优雅关机。
delayTime	否	Integer	关机时延，仅在stopMode为“GRACEFULLY”生效，单位：秒。该值为-1时，表示等待所有作业完成，并停止接受新作业。该值为大于0的任意值表示等待该时长后关机，并停止接受新作业。

响应参数

状态码：200

表 5-72 响应 Body 参数

参数	参数类型	描述
jobId	Array of strings	作业ID。

请求示例

停止集群

```
POST /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/bae65496-643e-47ca-84af-948672de7eeb/action

{
  "stop" : {
    "stopMode" : "GRACEFULLY",
    "delayTime" : -1
  }
}
```

响应示例

状态码：200

OK。

```
{
  "jobId" : [ "ff8080815e59d92d015e5b27ccb0004d" ]
}
```

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.1.14 创建集群

功能介绍

创建集群接口。

调用方法

请参见**如何调用API**。

URI

POST /v1.1/{project_id}/clusters

表 5-73 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。

请求参数

表 5-74 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
X-Language	是	String	请求语言。

表 5-75 请求 Body 参数

参数	是否必选	参数类型	描述
cluster	是	**cluster** object	集群对象，请参见cluster参数说明。
auto_remind	否	Boolean	选择是否开启消息通知。开启后，支持配置20个手机号码或邮箱，作业（目前仅支持表/文件迁移的作业）失败时、EIP异常时会发送短信或邮件通知用。
phone_num	否	String	接收消息通知的手机号码。
email	否	String	接收消息通知的邮箱。

表 5-76 cluster

参数	是否必选	参数类型	描述
scheduleBootTime	否	String	定时开机的时间，CDM集群会在每天这个时间开机。
isScheduleBootOff	否	Boolean	选择是否启用定时开关机功能。定时开关机功能和自动关机功能不可同时开启。
instances	否	Array of **instance** objects	节点列表，请参见instances参数说明。
datastore	否	**Datastore** object	集群信息，请参见datastore参数说明。
extended_properties	否	**ExtendedProperties** object	扩展属性，请参见extended_properties参数说明。

参数	是否必选	参数类型	描述
scheduleOffTime	否	String	定时关机的时间，定时关机时系统不会等待未完成的作业执行完成。
vpcId	否	String	指定虚拟私有云ID，用于集群网络配置。
name	否	String	集群名称。
sys_tags	否	Array of **sys_tags** objects	企业项目信息，请参见sys_tags参数说明。
isAutoOff	否	Boolean	选择是否启用自动关机功能，自动关机功能和定时开关机功能不可同时开启。如果选择自动关机，则当集群中无作业运行且无定时作业时，等待15分钟后集群将自动关机来帮您节约成本。

表 5-77 instance

参数	是否必选	参数类型	描述
availability_zone	是	String	集群的可用分区。可通过https://developer.huaweicloud.com/endpoint获取。
nics	是	Array of **nics** objects	网卡列表，最多两个网卡。请参见nics参数说明。

参数	是否必选	参数类型	描述
flavorRef	是	String	实例规格： ● a79fd5ae-1833-448a-88e8-3ea2b913e1f6：表示cdm.small规格，2核CPU、4G内存的虚拟机。适合PoC验证和开发测试。 ● fb8fe666-6734-4b11-bc6c-43d11db3c745：表示cdm.medium规格，4核CPU、8G内存的虚拟机适合单张表规模<1000万条的场景。 ● 5ddb1071-c5d7-40e0-a874-8a032e81a697：表示cdm.large规格，8核CPU、16G内存的虚拟机。适合单张表规模≥1000万条的场景。 ● 6ddb1072-c5d7-40e0-a874-8a032e81a698：表示cdm.xlarge规格，16核CPU、32G内存的虚拟机。需要10GE高速带宽进行TB以上的数据量迁移时使用。
type	是	String	节点类型，当前只有“cdm”一种类型。

表 5-78 nics

参数	是否必选	参数类型	描述
securityGroupId	是	String	安全组ID。
net-id	是	String	子网ID。

表 5-79 Datastore

参数	是否必选	参数类型	描述
type	否	String	类型，一般为cdm。
version	否	String	集群版本。

表 5-80 ExtendedProperties

参数	是否必选	参数类型	描述
workSpaceId	否	String	工作空间ID。
resourceId	否	String	资源ID。
trial	否	String	是否是试用集群。

表 5-81 sys_tags

参数	是否必选	参数类型	描述
value	是	String	企业项目ID。
key	是	String	该值目前固定为“_sys_enterprise_project_id”。

响应参数

状态码：202

表 5-82 响应 Body 参数

参数	参数类型	描述
name	String	集群名称。
id	String	集群ID。
task	**Task** object	任务信息。
datastore	**Datastore** object	集群信息。
instances	Array of **ClusterInstance** objects	集群的节点信息。

表 5-83 Task

参数	参数类型	描述
id	String	任务id。
name	String	任务名称。

表 5-84 Datastore

参数	参数类型	描述
type	String	类型，一般为cdm。
version	String	集群版本。

表 5-85 ClusterInstance

参数	参数类型	描述
id	String	节点的虚拟机ID。
name	String	节点的虚拟机名称。
type	String	节点类型，只支持一种类型“cdm”。
shard_id	String	分片ID。

请求示例

创建一个1.8.10版本，集群名为cdm-ab82的CDM集群。

```
POST /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters

{
  "cluster" : {
    "scheduleBootTime" : "",
    "isScheduleBootOff" : false,
    "instances" : [ {
      "availability_zone" : "xx-xxx",
      "nics" : [ {
        "securityGroupId" : "c37852d2-2d12-41cb-af47-65c80e995c80",
        "net-id" : "2d120298-6130-44d4-a438-454912fff901"
      } ],
      "flavorRef" : "5ddb1071-c5d7-40e0-a874-8a032e81a697",
      "type" : "cdm"
    } ],
    "datastore" : {
      "type" : "cdm",
      "version" : "1.8.10"
    },
    "scheduleOffTime" : "",
    "vpcId" : "67c06084-2212-4242-bcd4-d2144c2385a9",
    "name" : "cdm-ab82",
    "sys_tags" : [ {
      "value" : "1ce45885-4033-40d2-bdde-d4dbaceb387d",
      "key" : "_sys_enterprise_project_id"
    } ],
    "isAutoOff" : false
  },
  "auto_remind" : false,
  "phone_num" : "",
  "email" : ""
}
```

响应示例

状态码：202

Accepted。

```
{
  "id" : "befc862c-9286-46a0-a1d6-300d98b63aad",
  "name" : "cdm-4ef213",
  "task" : {
    "id" : "2c9080047f1b1185017f1ef6ad0500ac",
    "name" : "rdsCreateBackupJob"
  },
  "datastore" : {
    "type" : "cdm",
    "version" : "2.9.1.100"
  },
  "instances" : [ {
    "id" : "b2672e7d-2faf-423f-96bb-0664cd743cfd",
    "name" : "cdm-4ef213-cdm-dn-1-1",
    "type" : "cdm",
    "shard_id" : "dn-1"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建一个1.8.10版本，集群名为cdm-ab82的CDM集群。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class CreateClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateClusterRequest request = new CreateClusterRequest();
        CdmCreateClusterReq body = new CdmCreateClusterReq();
        List<SysTags> listClusterSysTags = new ArrayList<>();
```

```
        listClusterSysTags.add(
            new SysTags()
                .withValue("1ce45885-4033-40d2-bdde-d4dbaceb387d")
                .withKey("_sys_enterprise_project_id")
        );
        Datastore datastoreCluster = new Datastore();
        datastoreCluster.withType("cdm")
            .withVersion("1.8.10");
        List<Nics> listInstancesNics = new ArrayList<>();
        listInstancesNics.add(
            new Nics()
                .withSecurityGroupId("c37852d2-2d12-41cb-af47-65c80e995c80")
                .withNetId("2d120298-6130-44d4-a438-454912fff901")
        );
        List<Instance> listClusterInstances = new ArrayList<>();
        listClusterInstances.add(
            new Instance()
                .withAvailabilityZone("xx-xxx")
                .withNics(listInstancesNics)
                .withFlavorRef("5ddb1071-c5d7-40e0-a874-8a032e81a697")
                .withType("cdm")
        );
        CdmCreateClusterReqCluster clusterbody = new CdmCreateClusterReqCluster();
        clusterbody.withScheduleBootTime("")
            .withIsScheduleBootOff(false)
            .withInstances(listClusterInstances)
            .withDatastore(datastoreCluster)
            .withScheduleOffTime("")
            .withVpcId("67c06084-2212-4242-bcd4-d2144c2385a9")
            .withName("cdm-ab82")
            .withSysTags(listClusterSysTags)
            .withIsAutoOff(false);
        body.withEmail("");
        body.withPhoneNum("");
        body.withAutoRemind(false);
        body.withCluster(clusterbody);
        request.withBody(body);
        try {
            CreateClusterResponse response = client.createCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建一个1.8.10版本，集群名为cdm-ab82的CDM集群。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
```

```
variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateClusterRequest()
        listSysTagsCluster = [
            SysTags(
                value="1ce45885-4033-40d2-bdde-d4dbaceb387d",
                key="_sys_enterprise_project_id"
            )
        ]
        datastoreCluster = Datastore(
            type="cdm",
            version="1.8.10"
        )
        listNicsInstances = [
            Nics(
                security_group_id="c37852d2-2d12-41cb-af47-65c80e995c80",
                net_id="2d120298-6130-44d4-a438-454912fff901"
            )
        ]
        listInstancesCluster = [
            Instance(
                availability_zone="xx-xxx",
                nics=listNicsInstances,
                flavor_ref="5ddb1071-c5d7-40e0-a874-8a032e81a697",
                type="cdm"
            )
        ]
        clusterbody = CdmCreateClusterReqCluster(
            schedule_boot_time="",
            is_schedule_boot_off=False,
            instances=listInstancesCluster,
            datastore=datastoreCluster,
            schedule_off_time="",
            vpc_id="67c06084-2212-4242-bcd4-d2144c2385a9",
            name="cdm-ab82",
            sys_tags=listSysTagsCluster,
            is_auto_off=False
        )
        request.body = CdmCreateClusterReq(
            email="",
            phone_num="",
            auto_remind=False,
            cluster=clusterbody
        )
        response = client.create_cluster(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建一个1.8.10版本，集群名为cdm-ab82的CDM集群。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateClusterRequest{}
    var listSysTagsCluster = []model.SysTags{
        {
            Value: "1ce45885-4033-40d2-bdde-d4dbaceb387d",
            Key: "_sys_enterprise_project_id",
        },
    }
    typeDatastore:= "cdm"
    versionDatastore:= "1.8.10"
    datastoreCluster := &model.Datastore{
        Type: &typeDatastore,
        Version: &versionDatastore,
    }
    var listNicsInstances = []model.Nics{
        {
            SecurityGroupId: "c37852d2-2d12-41cb-af47-65c80e995c80",
            NetId: "2d120298-6130-44d4-a438-454912fff901",
        },
    }
    var listInstancesCluster = []model.Instance{
        {
            AvailabilityZone: "xx-xxx",
            Nics: listNicsInstances,
            FlavorRef: "5ddb1071-c5d7-40e0-a874-8a032e81a697",
            Type: "cdm",
        },
    }
    scheduleBootTimeCluster:= ""
    isScheduleBootOffCluster:= false
    scheduleOffTimeCluster:= ""
    vpcIdCluster:= "67c06084-2212-4242-bcd4-d2144c2385a9"
    nameCluster:= "cdm-ab82"
    isAutoOffCluster:= false
    clusterbody := &model.CdmCreateClusterReqCluster{
        ScheduleBootTime: &scheduleBootTimeCluster,
        IsScheduleBootOff: &isScheduleBootOffCluster,
        Instances: &listInstancesCluster,
```

```
        Datastore: datastoreCluster,
        ScheduleOffTime: &scheduleOffTimeCluster,
        VpcId: &vpcIdCluster,
        Name: &nameCluster,
        SysTags: &listSysTagsCluster,
        IsAutoOff: &isAutoOffCluster,
    }
    emailCdmCreateClusterReq:= ""
    phoneNumCdmCreateClusterReq:= ""
    autoRemindCdmCreateClusterReq:= false
    request.Body = &model.CdmCreateClusterReq{
        Email: &emailCdmCreateClusterReq,
        PhoneNum: &phoneNumCdmCreateClusterReq,
        AutoRemind: &autoRemindCdmCreateClusterReq,
        Cluster: clusterbody,
    }
    response, err := client.CreateCluster(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
202	Accepted。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部接口异常。
503	服务不可用。

错误码

请参见**错误码**。

5.1.15 查询集群列表

功能介绍

查询集群列表接口。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/clusters

表 5-86 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。

请求参数

表 5-87 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码：200

表 5-88 响应 Body 参数

参数	参数类型	描述
clusters	Array of **clusters** objects	集群列表，请参见clusters参数说明。

表 5-89 clusters

参数	参数类型	描述
customerConfig	**CustomerConfig** object	用户配置信息。
datastore	**Datastore** object	cdm信息。

参数	参数类型	描述
instances	Array of **ClusterDetailInstance** objects	集群的节点信息，请参见instances参数说明。
azName	String	az名称。
dbuser	String	数据库用户。
flavorName	String	规格名称。
recentEvent	Integer	事件数。
isAutoOff	Boolean	自动关机。
isScheduleBootOff	Boolean	选择是否启用定时开关机功能。定时开关机功能和自动关机功能不可同时开启。
clusterMode	String	集群模式：sharding(分片集群)。
namespace	String	命名空间。
task	**ClusterTask** object	任务信息。
publicEndpoint	String	集群绑定的EIP。
actionProgress	**ActionProgress** object	集群操作进度，任务信息，由key、value组成。key值为正在进行的任务，value值为正在进行任务的进度。示例如"action_progress":{"SNAPSHOTTING":"16%"}。
created	String	集群创建时间，格式为ISO8601：YYYY-MM-DDThh:mm:ssZ。
bakExpectedStartTime	String	开始时间。
bakKeepDay	Integer	保留时间。
name	String	集群名称。
statusDetail	String	集群状态描述：Normal（正常）。
id	String	集群ID。
isFrozen	String	集群是否冻结：0：否，1：是。
updated	String	集群更新时间，格式为ISO8601：YYYY-MM-DDThh:mm:ssZ。

参数	参数类型	描述
status	String	集群状态： • 100：创建中。 • 200：正常。 • 300：失败。 • 303：创建失败。 • 500：重启中。 • 800：冻结。 • 900：已关机。 • 910：正在关机。 • 920：正在开机。
failedReasons	FailedReasons object	失败原因。集群处于正常状态时不返回。

表 5-90 CustomerConfig

参数	参数类型	描述
failureRemind	String	失败提醒。
clusterName	String	集群类型。
serviceProvider	String	服务提供。
localDisk	String	是否本地磁盘。
ssl	String	是否使用ssl。
createFrom	String	创建来源。
resourceId	String	资源ID。
flavorType	String	规格类型。
workSpaceId	String	工作空间ID。
trial	String	适用。

表 5-91 Datastore

参数	参数类型	描述
type	String	类型，一般为cdm。
version	String	集群版本。

表 5-92 ClusterDetailInstance

参数	参数类型	描述
flavor	**flavor** object	节点的虚拟机规格，请参见flavor参数说明（查询集群列表时返回值为null）。
volume	**volume** object	节点的磁盘信息，请参见volume参数说明（查询集群列表时返回值为null）。
status	String	节点状态： ● 100：创建中。 ● 200：正常。 ● 300：失败。 ● 303：创建失败。 ● 400：已删除。 ● 800：冻结。
actions	Array of strings	节点操作状态列表： ● REBOOTING：重启中。 ● RESTORING：恢复中。 ● REBOOT_FAILURE：重启失败。
type	String	节点类型，只支持一种类型“cdm”。
id	String	节点的虚拟机ID。
name	String	节点的虚拟机名称。
isFrozen	String	节点是否冻结：0：否。1：是。
components	String	组件。
config_status	String	节点配置状态（查询集群列表时为null）： ● In-Sync：配置已同步。 ● Applying：配置中。 ● Sync-Failure：配置失败。
role	String	实例角色。
group	String	分组。
links	Array of **ClusterLinks** objects	链接信息（查询集群列表时返回值为null）。
paramsGroupId	String	组件分组id。
publicIp	String	公网ip。
manageIp	String	管理ip。

参数	参数类型	描述
trafficIp	String	流量ip。
shard_id	String	分片id。
manage_fix_ip	String	管理修复ip。
private_ip	String	私有ip。
internal_ip	String	内部ip。
resource	Array of **Resource** objects	资源信息（查询集群列表时返回值为null）。

表 5-93 flavor

参数	参数类型	描述
id	String	节点虚拟机的规格ID。
links	Array of **ClusterLinks** objects	链接信息。

表 5-94 volume

参数	参数类型	描述
type	String	节点的磁盘类型，只支持本地磁盘。
size	Long	节点磁盘大小，单位G。

表 5-95 ClusterLinks

参数	参数类型	描述
rel	String	关系。
href	String	链接地址。

表 5-96 Resource

参数	参数类型	描述
resource_id	String	资源id。
resource_type	String	资源类型：server(服务器)。

表 5-97 ClusterTask

参数	参数类型	描述
description	String	任务描述。
id	String	任务id。
name	String	任务名称。

表 5-98 ActionProgress

参数	参数类型	描述
CREATING	String	创建集群进度，例如：29%。
GROWING	String	扩容集群进度，例如：29%。
RESTORING	String	恢复集群进度，例如：29%。
SNAPSHOTTING	String	集群快照进度，例如：29%。
REPAIRING	String	修复集群进度，例如：29%。

表 5-99 FailedReasons

参数	参数类型	描述
CREATE_FAILED	**CREATE_FAILED** object	集群创建失败原因。

表 5-100 CREATE_FAILED

参数	参数类型	描述
errorCode	String	错误码。
errorMsg	String	失败原因。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters
```

响应示例

状态码：200

OK。

```
{
  "clusters" : [ {
```

```
    "publicEndpoint" : "49.xx.xx.10",
    "actionProgress" : { },
    "created" : "2018-09-05T08:38:25",
    "name" : "cdm-c018",
    "statusDetail" : "Normal",
    "id" : "bae65496-643e-47ca-84af-948672de7eeb",
    "isFrozen" : "0",
    "updated" : "2018-09-05T08:38:25",
    "status" : "200"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ListClustersSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ListClustersRequest request = new ListClustersRequest();
        try {
            ListClustersResponse response = client.listClusters(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListClustersRequest()
        response = client.list_clusters(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
```

```
        Build())

    request := &model.ListClustersRequest{}
    response, err := client.ListClusters(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.2 作业管理

5.2.1 查询作业

功能介绍

查询作业接口。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{job_name}

表 5-101 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。
job_name	是	String	查询多个作业用all,查询单个作业输入作业名。

表 5-102 Query 参数

参数	是否必选	参数类型	描述
filter	否	String	当“job_name”为“all”时，此参数用于模糊过滤作业。
page_no	否	Integer	指定作业页号。
page_size	否	Integer	每页作业数，值在10-100之间。
jobType	否	String	作业类型: • jobType=NORMAL_JOB：表示查询表/文件迁移的作业。 • jobType=BATCH_JOB：表示查询整库迁移的作业。 • jobType=SCENARIO_JOB：表示查询场景迁移的作业。 • 不指定该参数时，默认只查询表/文件迁移的作业。

请求参数

表 5-103 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码：200

表 5-104 响应 Body 参数

参数	参数类型	描述
total	Integer	作业数,查询单个作业时为0。
jobs	Array of **Job** objects	作业列表，请参见jobs参数说明。
page_no	Integer	返回指定页号的作业。
page_size	Integer	每页作业数。

表 5-105 Job

参数	参数类型	描述
job_type	String	作业类型： • NORMAL_JOB：表/文件迁移。 • BATCH_JOB：整库迁移。 • SCENARIO_JOB：场景迁移。
from-connector-name	String	源端连接类型，对应的连接参数如下： • generic-jdbc-connector：关系数据库连接。 • obs-connector：OBS连接。 • hdfs-connector：HDFS连接。 • hbase-connector：HBase连接、CloudTable连接。 • hive-connector：Hive连接。 • ftp-connector/sftp-connector：FTP/SFTP连接。 • mongodb-connector：MongoDB连接。 • redis-connector：Redis/DCS连接。 • kafka-connector：Kafka连接。 • dis-connector：DIS连接。 • elasticsearch-connector：Elasticsearch/云搜索服务连接。 • dli-connector：DLI连接。 • http-connector：HTTP/HTTPS连接，该连接暂无连接参数。 • dms-kafka-connector：DMSKafka连接。

参数	参数类型	描述
to-config-values	**ConfigValues** object	目的连接参数配置。根据不同目的端有不同的参数配置，具体可参考**目的端作业参数说明**下相应的目的端参数配置。
to-link-name	String	目的端连接名称，即为通过“创建连接”接口创建的连接对应的连接名。
driver-config-values	**ConfigValues** object	作业任务参数配置。例如配置作业失败重试、抽取并发数，具体可参考**作业任务参数说明**。
from-config-values	**ConfigValues** object	源连接参数配置。根据不同源端有不同的参数配置，具体可参考**源端作业参数说明**下相应的源端参数配置。
to-connector-name	String	目的端连接类型，对应的连接参数如下： • generic-jdbc-connector：关系数据库连接。 • obs-connector：OBS连接。 • hdfs-connector：HDFS连接。 • hbase-connector：HBase连接、CloudTable连接。 • hive-connector：Hive连接。 • ftp-connector/sftp-connector：FTP/SFTP连接。 • mongodb-connector：MongoDB连接。 • redis-connector：Redis/DCS连接。 • kafka-connector：Kafka连接。 • dis-connector：DIS连接。 • elasticsearch-connector：Elasticsearch/云搜索服务连接。 • dli-connector：DLI连接。 • http-connector：HTTP/HTTPS连接，该连接暂无连接参数。 • dms-kafka-connector：DMSKafka连接。
name	String	作业名称，长度在1到240个字符之间。
from-link-name	String	源连接名称，即为通过“创建连接”接口创建的连接对应的连接名。
creation-user	String	创建作业的用户。由系统生成，用户无需填写。

参数	参数类型	描述
creation-date	Long	作业创建的时间，单位：毫秒。由系统生成，用户无需填写。
update-date	Long	作业最后更新的时间，单位：毫秒。由系统生成，用户无需填写。
is_incre_job	Boolean	是否是增量作业。已废弃。
flag	Integer	是否是定时作业标记，如果是定时作业则为1，否则为0。由系统根据定时任务配置生成，用户无需填写。
files_read	Integer	已读文件数。由系统生成，用户无需填写。
update-user	String	最后更新作业的用户。由系统生成，用户无需填写。
external_id	String	具体执行的作业id，如果是本地作业，则一般为"job_local1202051771_0002"形式，如果是DLI作业，则为DLI作业ID，比如"12345"。由系统生成，用户无需填写。
type	String	与job_type一致，作业类型： ● NORMAL_JOB：表/文件迁移。 ● BATCH_JOB：整库迁移。 ● SCENARIO_JOB：场景迁移。
execute_start_date	Long	最近一次执行任务开始时间，单位：毫秒。由系统生成，用户无需填写。
delete_rows	Integer	增量作业删除行数，已废弃。
enabled	Boolean	是否激活连接。由系统生成，用户无需填写。
bytes_written	Long	作业写入的字节。由系统生成，用户无需填写。
id	Integer	作业ID。由系统生成，用户无需填写。
is_use_sql	Boolean	用户是否使用sql。由系统根据源端抽取是否使用sql语句生成，用户无需填写。
update_rows	Integer	增量作业更新行数，已废弃。
group_name	String	组名。
bytes_read	Long	作业读取的字节。由系统生成，用户无需填写。
execute_update_date	Long	最近一次执行任务更新时间，单位：毫秒。由系统生成，用户无需填写。

参数	参数类型	描述
write_rows	Integer	增量作业写入行数，已废弃。
rows_written	Integer	作业写入的行数。由系统生成，用户无需填写。
rows_read	Long	作业读取的行数。由系统生成，用户无需填写。
files_written	Integer	写入文件数。由系统生成，用户无需填写。
is_incrementing	Boolean	是否是增量作业，同is_incre_job，已废弃。
execute_create_date	Long	最近一次执行任务创建时间，单位：毫秒。由系统生成，用户无需填写。
status	String	作业最后的执行状态： • BOOTING：启动中。 • RUNNING：运行中。 • SUCCEEDED：成功。 • FAILED：失败。 • NEW：未被执行。

表 5-106 ConfigValues

参数	参数类型	描述
configs	Array of **configs** objects	源连接参数、目的连接参数和作业任务参数，它们的配置数据结构相同，其中“inputs”里的参数不一样，详细请参见configs数据结构说明。
extended-configs	**extended-configs** object	扩展配置，请参见extended-configs参数说明。扩展配置暂不对外开放，用户无需填写。

表 5-107 configs

参数	参数类型	描述
inputs	Array of **Input** objects	输入参数列表，列表中的每个参数为“name,value”结构，请参考inputs数据结构参数说明。在“from-config-values”数据结构中，不同的源连接类型有不同的“inputs”参数列表，请参见源端作业参数说明下的章节。在“to-config-values”数据结构中，不同的目的连接类型有不同的“inputs”参数列表，请参见目的端作业参数说明下面的子章节。在“driver-config-values”数据结构中，“inputs”具体参数请参见作业任务参数说明。
name	String	配置名称：源端作业的配置名称为“fromJobConfig”。目的端作业的配置名称为“toJobConfig”，连接的配置名称固定为“linkConfig”。
id	Integer	配置ID，由系统生成，用户无需填写。
type	String	配置类型，由系统生成，用户无需填写。值为LINK或者JOB，如果是连接管理API，则为LINK；如果是作业管理API，则为JOB。

表 5-108 Input

参数	参数类型	描述
name	String	参数名: • 如果是连接管理API，则以“linkConfig.”开头，对于不同连接类型有不同的参数，具体可参见**连接参数说明**下相应连接的参数说明。 • 如果是作业管理API，对于源端连接参数，则以“fromJobConfig.”开头，具体可参见**源端作业参数说明**下相应的源端参数说明；对于目的端连接参数，则以“toJobConfig.”开头，具体可参见**目的端作业参数说明**下相应的目的端参数说明；对于作业任务参数，请参见**作业任务参数说明**下相应的任务参数说明。
value	Object	参数值，参数名对应的值，必须填写为字符串。

参数	参数类型	描述
type	String	值类型，如STRING、INTEGER，由系统设定，用户无需填写。

表 5-109 extended-configs

参数	参数类型	描述
name	String	扩展配置名称，暂不对外开放，用户无需填写。
value	String	扩展配置值，暂不对外开放，用户无需填写。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/job/all?jobType=NORMAL_JOB
```

响应示例

状态码：200

OK。

```
{
  "total" : 1,
  "jobs" : [ {
    "job_type" : "NORMAL_JOB",
    "from-connector-name" : "elasticsearch-connector",
    "to-config-values" : {
      "configs" : [ {
        "inputs" : [ {
          "name" : "toJobConfig.streamName",
          "value" : "dis-lkGm"
        }, {
          "name" : "toJobConfig.separator",
          "value" : "|"
        }, {
          "name" : "toJobConfig.columnList",
          "value" : "1&2&3"
        } ],
        "name" : "toJobConfig"
      } ]
    },
    "to-link-name" : "dis",
    "driver-config-values" : {
      "configs" : [ {
        "inputs" : [ {
          "name" : "throttlingConfig.numExtractors",
          "value" : "1"
        }, {
          "name" : "throttlingConfig.submitToCluster",
          "value" : "false"
        }, {
          "name" : "throttlingConfig.numLoaders",
          "value" : "1"
        }, {
          "name" : "throttlingConfig.recordDirtyData",
```

```
        "value" : "false"
      } ],
      "name" : "throttlingConfig"
    }, {
      "inputs" : [ ],
      "name" : "jarConfig"
    }, {
      "inputs" : [ {
        "name" : "schedulerConfig.isSchedulerJob",
        "value" : "false"
      }, {
        "name" : "schedulerConfig.disposableType",
        "value" : "NONE"
      } ],
      "name" : "schedulerConfig"
    }, {
      "inputs" : [ ],
      "name" : "transformConfig"
    }, {
      "inputs" : [ {
        "name" : "retryJobConfig.retryJobType",
        "value" : "NONE"
      } ],
      "name" : "retryJobConfig"
    } ]
  },
  "from-config-values" : {
    "configs" : [ {
      "inputs" : [ {
        "name" : "fromJobConfig.index",
        "value" : "52est"
      }, {
        "name" : "fromJobConfig.type",
        "value" : "est_array"
      }, {
        "name" : "fromJobConfig.columnList",
        "value" : "array_f1_int:long&array_f2_text:string&array_f3_object:nested"
      }, {
        "name" : "fromJobConfig.splitNestedField",
        "value" : "false"
      } ],
      "name" : "fromJobConfig"
    } ]
  },
  "to-connector-name" : "dis-connector",
  "name" : "es_css",
  "from-link-name" : "css"
 } ],
 "page_no" : 1,
 "page_size" : 10
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;
```

```
public class ShowJobsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowJobsRequest request = new ShowJobsRequest();
        request.withClusterId("{cluster_id}");
        request.withJobName("{job_name}");
        try {
            ShowJobsResponse response = client.showJobs(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()
```

```
    try:
        request = ShowJobsRequest()
        request.cluster_id = "{cluster_id}"
        request.job_name = "{job_name}"
        response = client.show_jobs(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowJobsRequest{}
    request.ClusterId = "{cluster_id}"
    request.JobName = "{job_name}"
    response, err := client.ShowJobs(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。

错误码

请参见**错误码**。

5.2.2 删除作业

功能介绍

删除作业接口。

调用方法

请参见**如何调用API**。

URI

DELETE /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{job_name}

表 5-110 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。
job_name	是	String	作业名称。

请求参数

表 5-111 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码：500

表 5-112 响应 Body 参数

参数	参数类型	描述
errCode	String	错误码。
externalMessage	String	错误描述。

请求示例

```
DELETE /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/job/jdbc2hive
```

响应示例

状态码：500

服务内部错误，具体返回错误码请参考错误码。

```
{
  "errCode" : "Cdm.0100",
  "externalMessage" : "Job[jdbc2hive] doesn't exist."
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class DeleteJobSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
```

```
            .withSk(sk);

    CdmClient client = CdmClient.newBuilder()
            .withCredential(auth)
            .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
            .build();
    DeleteJobRequest request = new DeleteJobRequest();
    request.withClusterId("{cluster_id}");
    request.withJobName("{job_name}");
    try {
        DeleteJobResponse response = client.deleteJob(request);
        System.out.println(response.toString());
    } catch (ConnectionException e) {
        e.printStackTrace();
    } catch (RequestTimeoutException e) {
        e.printStackTrace();
    } catch (ServiceResponseException e) {
        e.printStackTrace();
        System.out.println(e.getHttpStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
  }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteJobRequest()
        request.cluster_id = "{cluster_id}"
        request.job_name = "{job_name}"
        response = client.delete_job(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main
```

```
import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteJobRequest{}
    request.ClusterId = "{cluster_id}"
    request.JobName = "{job_name}"
    response, err := client.DeleteJob(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。

错误码

请参见**错误码**。

5.2.3 修改作业

功能介绍

修改作业接口。

调用方法

请参见**如何调用API**。

URI

PUT /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{job_name}

表 5-113 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。
job_name	是	String	作业名称。

请求参数

表 5-114 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 5-115 请求 Body 参数

参数	是否必选	参数类型	描述
jobs	是	Array of **Job** objects	作业列表，请参见jobs数据结构说明。

表 5-116 Job

参数	是否必选	参数类型	描述
job_type	是	String	作业类型： • NORMAL_JOB：表/文件迁移。 • BATCH_JOB：整库迁移。 • SCENARIO_JOB：场景迁移。
from-connector-name	是	String	源端连接类型，对应的连接参数如下： • generic-jdbc-connector：关系数据库连接。 • obs-connector：OBS连接。 • hdfs-connector：HDFS连接。 • hbase-connector：HBase连接、CloudTable连接。 • hive-connector：Hive连接。 • ftp-connector/sftp-connector：FTP/SFTP连接。 • mongodb-connector：MongoDB连接。 • redis-connector：Redis/DCS连接。 • kafka-connector：Kafka连接。 • dis-connector：DIS连接。 • elasticsearch-connector：Elasticsearch/云搜索服务连接。 • dli-connector：DLI连接。 • http-connector：HTTP/HTTPS连接，该连接暂无连接参数。 • dms-kafka-connector：DMSKafka连接。
to-config-values	是	ConfigValues object	目的连接参数配置。根据不同目的端有不同的参数配置，具体可参考目的端作业参数说明下相应的目的端参数配置。

参数	是否必选	参数类型	描述
to-link-name	是	String	目的端连接名称，即为通过“创建连接”接口创建的连接对应的连接名。
driver-config-values	是	**ConfigValues** object	作业任务参数配置。例如配置作业失败重试、抽取并发数，具体可参考**作业任务参数说明**。
from-config-values	是	**ConfigValues** object	源连接参数配置。根据不同源端有不同的参数配置，具体可参考**源端作业参数说明**下相应的源端参数配置。
to-connector-name	是	String	目的端连接类型，对应的连接参数如下： • generic-jdbc-connector：关系数据库连接。 • obs-connector：OBS连接。 • hdfs-connector：HDFS连接。 • hbase-connector：HBase连接、CloudTable连接。 • hive-connector：Hive连接。 • ftp-connector/sftp-connector：FTP/SFTP连接。 • mongodb-connector：MongoDB连接。 • redis-connector：Redis/DCS连接。 • kafka-connector：Kafka连接。 • dis-connector：DIS连接。 • elasticsearch-connector：Elasticsearch/云搜索服务连接。 • dli-connector：DLI连接。 • http-connector：HTTP/HTTPS连接，该连接暂无连接参数。 • dms-kafka-connector：DMSKafka连接。
name	是	String	作业名称，长度在1到240个字符之间。

参数	是否必选	参数类型	描述
from-link-name	是	String	源连接名称，即为通过“创建连接”接口创建的连接对应的连接名。
creation-user	否	String	创建作业的用户。由系统生成，用户无需填写。
creation-date	否	Long	作业创建的时间，单位：毫秒。由系统生成，用户无需填写。
update-date	否	Long	作业最后更新的时间，单位：毫秒。由系统生成，用户无需填写。
is_incre_job	否	Boolean	是否是增量作业。已废弃。
flag	否	Integer	是否是定时作业标记，如果是定时作业则为1，否则为0。由系统根据定时任务配置生成，用户无需填写。
files_read	否	Integer	已读文件数。由系统生成，用户无需填写。
update-user	否	String	最后更新作业的用户。由系统生成，用户无需填写。
external_id	否	String	具体执行的作业id，如果是本地作业，则一般为"job_local1202051771_0002"形式，如果是DLI作业，则为DLI作业ID，比如"12345"。由系统生成，用户无需填写。
type	否	String	与job_type一致，作业类型： ● NORMAL_JOB：表/文件迁移。 ● BATCH_JOB：整库迁移。 ● SCENARIO_JOB：场景迁移。
execute_start_date	否	Long	最近一次执行任务开始时间，单位：毫秒。由系统生成，用户无需填写。
delete_rows	否	Integer	增量作业删除行数，已废弃。
enabled	否	Boolean	是否激活连接。由系统生成，用户无需填写。
bytes_written	否	Long	作业写入的字节。由系统生成，用户无需填写。

参数	是否必选	参数类型	描述
id	否	Integer	作业ID。由系统生成，用户无需填写。
is_use_sql	否	Boolean	用户是否使用sql。由系统根据源端抽取是否使用sql语句生成，用户无需填写。
update_rows	否	Integer	增量作业更新行数，已废弃。
group_name	否	String	组名。
bytes_read	否	Long	作业读取的字节。由系统生成，用户无需填写。
execute_update_date	否	Long	最近一次执行任务更新时间，单位：毫秒。由系统生成，用户无需填写。
write_rows	否	Integer	增量作业写入行数，已废弃。
rows_written	否	Integer	作业写入的行数。由系统生成，用户无需填写。
rows_read	否	Long	作业读取的行数。由系统生成，用户无需填写。
files_written	否	Integer	写入文件数。由系统生成，用户无需填写。
is_incrementing	否	Boolean	是否是增量作业，同is_incre_job，已废弃。
execute_create_date	否	Long	最近一次执行任务创建时间，单位：毫秒。由系统生成，用户无需填写。
status	否	String	作业最后的执行状态： ● BOOTING：启动中。 ● RUNNING：运行中。 ● SUCCEEDED：成功。 ● FAILED：失败。 ● NEW：未被执行。

表 5-117 ConfigValues

参数	是否必选	参数类型	描述
configs	是	Array of **configs** objects	源连接参数、目的连接参数和作业任务参数，它们的配置数据结构相同，其中“inputs”里的参数不一样，详细请参见configs数据结构说明。
extended-configs	否	**extended-configs** object	扩展配置，请参见extended-configs参数说明。扩展配置暂不对外开放，用户无需填写。

表 5-118 configs

参数	是否必选	参数类型	描述
inputs	是	Array of **Input** objects	输入参数列表，列表中的每个参数为“name,value”结构，请参考inputs数据结构参数说明。在“from-config-values”数据结构中，不同的源连接类型有不同的“inputs”参数列表，请参见源端作业参数说明下的章节。在“to-config-values”数据结构中，不同的目的连接类型有不同的“inputs”参数列表，请参见目的端作业参数说明下面的子章节。在“driver-config-values”数据结构中，“inputs”具体参数请参见作业任务参数说明。
name	是	String	配置名称：源端作业的配置名称为“fromJobConfig”。目的端作业的配置名称为“toJobConfig”，连接的配置名称固定为“linkConfig”。
id	否	Integer	配置ID，由系统生成，用户无需填写。
type	否	String	配置类型，由系统生成，用户无需填写。值为LINK或者JOB，如果是连接管理API，则为LINK；如果是作业管理API，则为JOB。

表 5-119 Input

参数	是否必选	参数类型	描述
name	是	String	参数名: • 如果是连接管理API，则以“linkConfig.”开头，对于不同连接类型有不同的参数，具体可参见**连接参数说明**下相应连接的参数说明。 • 如果是作业管理API，对于源端连接参数，则以“fromJobConfig.”开头，具体可参见**源端作业参数说明**下相应的源端参数说明；对于目的端连接参数，则以“toJobConfig.”开头，具体可参见**目的端作业参数说明**下相应的目的端参数说明；对于作业任务参数，请参见**作业任务参数说明**下相应的任务参数说明。
value	是	Object	参数值，参数名对应的值，必须填写为字符串。
type	否	String	值类型，如STRING、INTEGER，由系统设定，用户无需填写。

表 5-120 extended-configs

参数	是否必选	参数类型	描述
name	否	String	扩展配置名称，暂不对外开放，用户无需填写。
value	否	String	扩展配置值，暂不对外开放，用户无需填写。

响应参数

状态码：200

表 5-121 响应 Body 参数

参数	参数类型	描述
validation-result	Array of **JobValidationResult** objects	校验结果：如果修改失败，返回失败原因。如果修改成功，返回空列表。

表 5-122 JobValidationResult

参数	参数类型	描述
message	String	错误描述。
status	String	错误级别，如：ERROR、WARNING。

状态码：400

表 5-123 响应 Body 参数

参数	参数类型	描述
code	String	返回编码。
errCode	String	错误码。
message	String	报错信息。
externalMessage	String	附加信息。

请求示例

修改一个源端为Elasticsearch数据连接，目的端为DIS数据连接，作业名为es_css的表数据迁移作业。

```
PUT /v1.1/1551c7f6c808414d8e9f3c514a170f2e/cluster/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/job/es_css

{
  "jobs" : [ {
    "job_type" : "NORMAL_JOB",
    "from-connector-name" : "elasticsearch-connector",
    "to-config-values" : {
      "configs" : [ {
        "inputs" : [ {
          "name" : "toJobConfig.streamName",
          "value" : "dis-lkGm"
        }, {
          "name" : "toJobConfig.separator",
          "value" : "|"
        }, {
          "name" : "toJobConfig.columnList",
          "value" : "1&2&3"
        } ],
        "name" : "toJobConfig"
```

```
    } ]
  },
  "to-link-name" : "dis",
  "driver-config-values" : {
    "configs" : [ {
      "inputs" : [ {
        "name" : "throttlingConfig.numExtractors",
        "value" : "1"
      }, {
        "name" : "throttlingConfig.submitToCluster",
        "value" : "false"
      }, {
        "name" : "throttlingConfig.numLoaders",
        "value" : "1"
      }, {
        "name" : "throttlingConfig.recordDirtyData",
        "value" : "false"
      } ],
      "name" : "throttlingConfig"
    }, {
      "inputs" : [ ],
      "name" : "jarConfig"
    }, {
      "inputs" : [ {
        "name" : "schedulerConfig.isSchedulerJob",
        "value" : "false"
      }, {
        "name" : "schedulerConfig.disposableType",
        "value" : "NONE"
      } ],
      "name" : "schedulerConfig"
    }, {
      "inputs" : [ ],
      "name" : "transformConfig"
    }, {
      "inputs" : [ {
        "name" : "retryJobConfig.retryJobType",
        "value" : "NONE"
      } ],
      "name" : "retryJobConfig"
    } ]
  },
  "from-config-values" : {
    "configs" : [ {
      "inputs" : [ {
        "name" : "fromJobConfig.index",
        "value" : "52est"
      }, {
        "name" : "fromJobConfig.type",
        "value" : "est_array"
      }, {
        "name" : "fromJobConfig.columnList",
        "value" : "array_f1_int:long&array_f2_text:string&array_f3_object:nested"
      }, {
        "name" : "fromJobConfig.splitNestedField",
        "value" : "false"
      } ],
      "name" : "fromJobConfig"
    } ]
  },
  "to-connector-name" : "dis-connector",
  "name" : "es_css",
  "from-link-name" : "css"
 } ]
}
```

响应示例

状态码：200

OK。

```
{
  "validation-result" : [ { }, { }, { } ]
}
```

状态码：400

报错错误码。

```
{
  "code" : "Cdm.0095",
  "errCode" : "Cdm.00095",
  "message" : "A job with the name obs-obs does not exist.",
  "externalMessage" : "A job with the name obs-obs does not exist."
}
```

SDK 代码示例

SDK代码示例如下。

Java

修改一个源端为Elasticsearch数据连接，目的端为DIS数据连接，作业名为es_css的表数据迁移作业。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class UpdateJobSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateJobRequest request = new UpdateJobRequest();
        request.withClusterId("{cluster_id}");
        request.withJobName("{job_name}");
        CdmUpdateJobJsonReq body = new CdmUpdateJobJsonReq();
        List<Input> listConfigsInputs = new ArrayList<>();
        listConfigsInputs.add(
```

```
        new Input()
            .withName("fromJobConfig.index")
            .withValue("52est")
    );
    listConfigsInputs.add(
        new Input()
            .withName("fromJobConfig.type")
            .withValue("est_array")
    );
    listConfigsInputs.add(
        new Input()
            .withName("fromJobConfig.columnList")
            .withValue("array_f1_int:long&array_f2_text:string&array_f3_object:nested")
    );
    listConfigsInputs.add(
        new Input()
            .withName("fromJobConfig.splitNestedField")
            .withValue("false")
    );
    List<Configs> listFromConfigValuesConfigs = new ArrayList<>();
    listFromConfigValuesConfigs.add(
        new Configs()
            .withInputs(listConfigsInputs)
            .withName("fromJobConfig")
    );
    ConfigValues fromconfigvaluesJobs = new ConfigValues();
    fromconfigvaluesJobs.withConfigs(listFromConfigValuesConfigs);
    List<Input> listConfigsInputs1 = new ArrayList<>();
    listConfigsInputs1.add(
        new Input()
            .withName("retryJobConfig.retryJobType")
            .withValue("NONE")
    );
    List<Input> listConfigsInputs2 = new ArrayList<>();
    listConfigsInputs2.add(
        new Input()
            .withName("schedulerConfig.isSchedulerJob")
            .withValue("false")
    );
    listConfigsInputs2.add(
        new Input()
            .withName("schedulerConfig.disposableType")
            .withValue("NONE")
    );
    List<Input> listConfigsInputs3 = new ArrayList<>();
    listConfigsInputs3.add(
        new Input()
            .withName("throttlingConfig.numExtractors")
            .withValue("1")
    );
    listConfigsInputs3.add(
        new Input()
            .withName("throttlingConfig.submitToCluster")
            .withValue("false")
    );
    listConfigsInputs3.add(
        new Input()
            .withName("throttlingConfig.numLoaders")
            .withValue("1")
    );
    listConfigsInputs3.add(
        new Input()
            .withName("throttlingConfig.recordDirtyData")
            .withValue("false")
    );
    List<Configs> listDriverConfigValuesConfigs = new ArrayList<>();
    listDriverConfigValuesConfigs.add(
        new Configs()
            .withInputs(listConfigsInputs1)
```

```
            .withName("retryJobConfig")
        );
        ConfigValues driverconfigvaluesJobs = new ConfigValues();
        driverconfigvaluesJobs.withConfigs(listDriverConfigValuesConfigs);
        List<Input> listConfigsInputs4 = new ArrayList<>();
        listConfigsInputs4.add(
            new Input()
                .withName("toJobConfig.streamName")
                .withValue("dis-lkGm")
        );
        listConfigsInputs4.add(
            new Input()
                .withName("toJobConfig.separator")
                .withValue("|")
        );
        listConfigsInputs4.add(
            new Input()
                .withName("toJobConfig.columnList")
                .withValue("1&2&3")
        );
        List<Configs> listToConfigValuesConfigs = new ArrayList<>();
        listToConfigValuesConfigs.add(
            new Configs()
                .withInputs(listConfigsInputs4)
                .withName("toJobConfig")
        );
        ConfigValues toconfigvaluesJobs = new ConfigValues();
        toconfigvaluesJobs.withConfigs(listToConfigValuesConfigs);
        List<Job> listbodyJobs = new ArrayList<>();
        listbodyJobs.add(
            new Job()
                .withJobType(Job.JobTypeEnum.fromValue("NORMAL_JOB"))
                .withFromConnectorName("elasticsearch-connector")
                .withToConfigValues(toconfigvaluesJobs)
                .withToLinkName("dis")
                .withDriverConfigValues(driverconfigvaluesJobs)
                .withFromConfigValues(fromconfigvaluesJobs)
                .withToConnectorName("dis-connector")
                .withName("es_css")
                .withFromLinkName("css")
        );
        body.withJobs(listbodyJobs);
        request.withBody(body);
        try {
            UpdateJobResponse response = client.updateJob(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

修改一个源端为Elasticsearch数据连接，目的端为DIS数据连接，作业名为es_css的表数据迁移作业。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
```

```
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateJobRequest()
        request.cluster_id = "{cluster_id}"
        request.job_name = "{job_name}"
        listInputsConfigs = [
            Input(
                name="fromJobConfig.index",
                value="52est"
            ),
            Input(
                name="fromJobConfig.type",
                value="est_array"
            ),
            Input(
                name="fromJobConfig.columnList",
                value="array_f1_int:long&array_f2_text:string&array_f3_object:nested"
            ),
            Input(
                name="fromJobConfig.splitNestedField",
                value="false"
            )
        ]
        listConfigsFromconfigvalues = [
            Configs(
                inputs=listInputsConfigs,
                name="fromJobConfig"
            )
        ]
        fromconfigvaluesJobs = ConfigValues(
            configs=listConfigsFromconfigvalues
        )
        listInputsConfigs1 = [
            Input(
                name="retryJobConfig.retryJobType",
                value="NONE"
            )
        ]
        listInputsConfigs2 = [
            Input(
                name="schedulerConfig.isSchedulerJob",
                value="false"
            ),
            Input(
                name="schedulerConfig.disposableType",
                value="NONE"
            )
        ]
        listInputsConfigs3 = [
```

```
            Input(
                name="throttlingConfig.numExtractors",
                value="1"
            ),
            Input(
                name="throttlingConfig.submitToCluster",
                value="false"
            ),
            Input(
                name="throttlingConfig.numLoaders",
                value="1"
            ),
            Input(
                name="throttlingConfig.recordDirtyData",
                value="false"
            )
        ]
        listConfigsDriverconfigvalues = [
            Configs(
                inputs=listInputsConfigs1,
                name="retryJobConfig"
            )
        ]
        driverconfigvaluesJobs = ConfigValues(
            configs=listConfigsDriverconfigvalues
        )
        listInputsConfigs4 = [
            Input(
                name="toJobConfig.streamName",
                value="dis-lkGm"
            ),
            Input(
                name="toJobConfig.separator",
                value="|"
            ),
            Input(
                name="toJobConfig.columnList",
                value="1&2&3"
            )
        ]
        listConfigsToconfigvalues = [
            Configs(
                inputs=listInputsConfigs4,
                name="toJobConfig"
            )
        ]
        toconfigvaluesJobs = ConfigValues(
            configs=listConfigsToconfigvalues
        )
        listJobsbody = [
            Job(
                job_type="NORMAL_JOB",
                from_connector_name="elasticsearch-connector",
                to_config_values=toconfigvaluesJobs,
                to_link_name="dis",
                driver_config_values=driverconfigvaluesJobs,
                from_config_values=fromconfigvaluesJobs,
                to_connector_name="dis-connector",
                name="es_css",
                from_link_name="css"
            )
        ]
        request.body = CdmUpdateJobJsonReq(
            jobs=listJobsbody
        )
        response = client.update_job(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
```

```
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

修改一个源端为Elasticsearch数据连接，目的端为DIS数据连接，作业名为es_css的表数据迁移作业。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateJobRequest{}
    request.ClusterId = "{cluster_id}"
    request.JobName = "{job_name}"
    valueInputs:= "52est"
    var valueInputsInterface interface{} = valueInputs
    valueInputs1:= "est_array"
    var valueInputsInterface1 interface{} = valueInputs1
    valueInputs2:= "array_f1_int:long&array_f2_text:string&array_f3_object:nested"
    var valueInputsInterface2 interface{} = valueInputs2
    valueInputs3:= "false"
    var valueInputsInterface3 interface{} = valueInputs3
    var listInputsConfigs = []model.Input{
        {
            Name: "fromJobConfig.index",
            Value: &valueInputsInterface,
        },
        {
            Name: "fromJobConfig.type",
            Value: &valueInputsInterface1,
        },
        {
            Name: "fromJobConfig.columnList",
            Value: &valueInputsInterface2,
        },
        {
            Name: "fromJobConfig.splitNestedField",
            Value: &valueInputsInterface3,
        },
```

```
}
var listConfigsFromConfigValues = []model.Configs{
    {
        Inputs: listInputsConfigs,
        Name: "fromJobConfig",
    },
}
fromconfigvaluesJobs := &model.ConfigValues{
    Configs: listConfigsFromConfigValues,
}
valueInputs4:= "NONE"
var valueInputsInterface4 interface{} = valueInputs4
var listInputsConfigs1 = []model.Input{
    {
        Name: "retryJobConfig.retryJobType",
        Value: &valueInputsInterface4,
    },
}
valueInputs5:= "false"
var valueInputsInterface5 interface{} = valueInputs5
valueInputs6:= "NONE"
var valueInputsInterface6 interface{} = valueInputs6
var listInputsConfigs2 = []model.Input{
    {
        Name: "schedulerConfig.isSchedulerJob",
        Value: &valueInputsInterface5,
    },
    {
        Name: "schedulerConfig.disposableType",
        Value: &valueInputsInterface6,
    },
}
valueInputs7:= "1"
var valueInputsInterface7 interface{} = valueInputs7
valueInputs8:= "false"
var valueInputsInterface8 interface{} = valueInputs8
valueInputs9:= "1"
var valueInputsInterface9 interface{} = valueInputs9
valueInputs10:= "false"
var valueInputsInterface10 interface{} = valueInputs10
var listInputsConfigs3 = []model.Input{
    {
        Name: "throttlingConfig.numExtractors",
        Value: &valueInputsInterface7,
    },
    {
        Name: "throttlingConfig.submitToCluster",
        Value: &valueInputsInterface8,
    },
    {
        Name: "throttlingConfig.numLoaders",
        Value: &valueInputsInterface9,
    },
    {
        Name: "throttlingConfig.recordDirtyData",
        Value: &valueInputsInterface10,
    },
}
var listConfigsDriverConfigValues = []model.Configs{
    {
        Inputs: listInputsConfigs1,
        Name: "retryJobConfig",
    },
}
driverconfigvaluesJobs := &model.ConfigValues{
    Configs: listConfigsDriverConfigValues,
}
valueInputs11:= "dis-lkGm"
var valueInputsInterface11 interface{} = valueInputs11
```

```
    valueInputs12:= "|"
    var valueInputsInterface12 interface{} = valueInputs12
    valueInputs13:= "1&2&3"
    var valueInputsInterface13 interface{} = valueInputs13
    var listInputsConfigs4 = []model.Input{
        {
            Name: "toJobConfig.streamName",
            Value: &valueInputsInterface11,
        },
        {
            Name: "toJobConfig.separator",
            Value: &valueInputsInterface12,
        },
        {
            Name: "toJobConfig.columnList",
            Value: &valueInputsInterface13,
        },
    }
    var listConfigsToConfigValues = []model.Configs{
        {
            Inputs: listInputsConfigs4,
            Name: "toJobConfig",
        },
    }
    toconfigvaluesJobs := &model.ConfigValues{
        Configs: listConfigsToConfigValues,
    }
    var listJobsbody = []model.Job{
        {
            JobType: model.GetJobJobTypeEnum().NORMAL_JOB,
            FromConnectorName: "elasticsearch-connector",
            ToConfigValues: toconfigvaluesJobs,
            ToLinkName: "dis",
            DriverConfigValues: driverconfigvaluesJobs,
            FromConfigValues: fromconfigvaluesJobs,
            ToConnectorName: "dis-connector",
            Name: "es_css",
            FromLinkName: "css",
        },
    }
    request.Body = &model.CdmUpdateJobJsonReq{
        Jobs: listJobsbody,
    }
    response, err := client.UpdateJob(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	报错错误码。

错误码

请参见**错误码**。

5.2.4 随机集群创建作业并执行

功能介绍

随机集群创建作业并执行接口。

调用方法

请参见**如何调用API**。

URI

POST /v1.1/{project_id}/clusters/job

表 5-124 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。

请求参数

表 5-125 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
X-Language	是	String	请求语言。

表 5-126 请求 Body 参数

参数	是否必选	参数类型	描述
jobs	是	Array of **Job** objects	作业列表，请参见jobs数据结构说明。
clusters	是	Array of strings	CDM集群ID列表，系统会从里面随机选择一个开机状态的集群，在该集群中创建作业并执行作业。

表 5-127 Job

参数	是否必选	参数类型	描述
job_type	是	String	作业类型： • NORMAL_JOB：表/文件迁移。 • BATCH_JOB：整库迁移。 • SCENARIO_JOB：场景迁移。
from-connector-name	是	String	源端连接类型，对应的连接参数如下： • generic-jdbc-connector：关系数据库连接。 • obs-connector：OBS连接。 • hdfs-connector：HDFS连接。 • hbase-connector：HBase连接、CloudTable连接。 • hive-connector：Hive连接。 • ftp-connector/sftp-connector：FTP/SFTP连接。 • mongodb-connector：MongoDB连接。 • redis-connector：Redis/DCS连接。 • kafka-connector：Kafka连接。 • dis-connector：DIS连接。 • elasticsearch-connector：Elasticsearch/云搜索服务连接。 • dli-connector：DLI连接。 • http-connector：HTTP/HTTPS连接，该连接暂无连接参数。 • dms-kafka-connector：DMSKafka连接。
to-config-values	是	ConfigValues object	目的连接参数配置。根据不同目的端有不同的参数配置，具体可参考目的端作业参数说明下相应的目的端参数配置。

参数	是否必选	参数类型	描述
to-link-name	是	String	目的端连接名称，即为通过“创建连接”接口创建的连接对应的连接名。
driver-config-values	是	**ConfigValues** object	作业任务参数配置。例如配置作业失败重试、抽取并发数，具体可参考**作业任务参数说明**。
from-config-values	是	**ConfigValues** object	源连接参数配置。根据不同源端有不同的参数配置，具体可参考**源端作业参数说明**下相应的源端参数配置。
to-connector-name	是	String	目的端连接类型，对应的连接参数如下： • generic-jdbc-connector：关系数据库连接。 • obs-connector：OBS连接。 • hdfs-connector：HDFS连接。 • hbase-connector：HBase连接、CloudTable连接。 • hive-connector：Hive连接。 • ftp-connector/sftp-connector：FTP/SFTP连接。 • mongodb-connector：MongoDB连接。 • redis-connector：Redis/DCS连接。 • kafka-connector：Kafka连接。 • dis-connector：DIS连接。 • elasticsearch-connector：Elasticsearch/云搜索服务连接。 • dli-connector：DLI连接。 • http-connector：HTTP/HTTPS连接，该连接暂无连接参数。 • dms-kafka-connector：DMSKafka连接。
name	是	String	作业名称，长度在1到240个字符之间。

参数	是否必选	参数类型	描述
from-link-name	是	String	源连接名称，即为通过“创建连接”接口创建的连接对应的连接名。
creation-user	否	String	创建作业的用户。由系统生成，用户无需填写。
creation-date	否	Long	作业创建的时间，单位：毫秒。由系统生成，用户无需填写。
update-date	否	Long	作业最后更新的时间，单位：毫秒。由系统生成，用户无需填写。
is_incre_job	否	Boolean	是否是增量作业。已废弃。
flag	否	Integer	是否是定时作业标记，如果是定时作业则为1，否则为0。由系统根据定时任务配置生成，用户无需填写。
files_read	否	Integer	已读文件数。由系统生成，用户无需填写。
update-user	否	String	最后更新作业的用户。由系统生成，用户无需填写。
external_id	否	String	具体执行的作业id，如果是本地作业，则一般为"job_local1202051771_0002"形式，如果是DLI作业，则为DLI作业ID，比如"12345"。由系统生成，用户无需填写。
type	否	String	与job_type一致，作业类型： ● NORMAL_JOB：表/文件迁移。 ● BATCH_JOB：整库迁移。 ● SCENARIO_JOB：场景迁移。
execute_start_date	否	Long	最近一次执行任务开始时间，单位：毫秒。由系统生成，用户无需填写。
delete_rows	否	Integer	增量作业删除行数，已废弃。
enabled	否	Boolean	是否激活连接。由系统生成，用户无需填写。
bytes_written	否	Long	作业写入的字节。由系统生成，用户无需填写。

参数	是否必选	参数类型	描述
id	否	Integer	作业ID。由系统生成，用户无需填写。
is_use_sql	否	Boolean	用户是否使用sql。由系统根据源端抽取是否使用sql语句生成，用户无需填写。
update_rows	否	Integer	增量作业更新行数，已废弃。
group_name	否	String	组名。
bytes_read	否	Long	作业读取的字节。由系统生成，用户无需填写。
execute_update_date	否	Long	最近一次执行任务更新时间，单位：毫秒。由系统生成，用户无需填写。
write_rows	否	Integer	增量作业写入行数，已废弃。
rows_written	否	Integer	作业写入的行数。由系统生成，用户无需填写。
rows_read	否	Long	作业读取的行数。由系统生成，用户无需填写。
files_written	否	Integer	写入文件数。由系统生成，用户无需填写。
is_incrementing	否	Boolean	是否是增量作业，同is_incre_job，已废弃。
execute_create_date	否	Long	最近一次执行任务创建时间，单位：毫秒。由系统生成，用户无需填写。
status	否	String	作业最后的执行状态： ● BOOTING：启动中。 ● RUNNING：运行中。 ● SUCCEEDED：成功。 ● FAILED：失败。 ● NEW：未被执行。

表 5-128 ConfigValues

参数	是否必选	参数类型	描述
configs	是	Array of **configs** objects	源连接参数、目的连接参数和作业任务参数，它们的配置数据结构相同，其中“inputs”里的参数不一样，详细请参见configs数据结构说明。
extended-configs	否	**extended-configs** object	扩展配置，请参见extended-configs参数说明。扩展配置暂不对外开放，用户无需填写。

表 5-129 configs

参数	是否必选	参数类型	描述
inputs	是	Array of **Input** objects	输入参数列表，列表中的每个参数为“name,value”结构，请参考inputs数据结构参数说明。在“from-config-values”数据结构中，不同的源连接类型有不同的“inputs”参数列表，请参见源端作业参数说明下的章节。在“to-config-values”数据结构中，不同的目的连接类型有不同的“inputs”参数列表，请参见目的端作业参数说明下面的子章节。在“driver-config-values”数据结构中，“inputs”具体参数请参见作业任务参数说明。
name	是	String	配置名称：源端作业的配置名称为“fromJobConfig”。目的端作业的配置名称为“toJobConfig”，连接的配置名称固定为“linkConfig”。
id	否	Integer	配置ID，由系统生成，用户无需填写。
type	否	String	配置类型，由系统生成，用户无需填写。值为LINK或者JOB，如果是连接管理API，则为LINK；如果是作业管理API，则为JOB。

表 5-130 Input

参数	是否必选	参数类型	描述
name	是	String	参数名: • 如果是连接管理API，则以“linkConfig.”开头，对于不同连接类型有不同的参数，具体可参见**连接参数说明**下相应连接的参数说明。 • 如果是作业管理API，对于源端连接参数，则以“fromJobConfig.”开头，具体可参见**源端作业参数说明**下相应的源端参数说明；对于目的端连接参数，则以“toJobConfig.”开头，具体可参见**目的端作业参数说明**下相应的目的端参数说明；对于作业任务参数，请参见**作业任务参数说明**下相应的任务参数说明。
value	是	Object	参数值，参数名对应的值，必须填写为字符串。
type	否	String	值类型，如STRING、INTEGER，由系统设定，用户无需填写。

表 5-131 extended-configs

参数	是否必选	参数类型	描述
name	否	String	扩展配置名称，暂不对外开放，用户无需填写。
value	否	String	扩展配置值，暂不对外开放，用户无需填写。

响应参数

状态码：200

表 5-132 响应 Body 参数

参数	参数类型	描述
submissions	Array of **StartJobSubmission** objects	作业运行信息，请参见submission参数说明。

表 5-133 StartJobSubmission

参数	参数类型	描述
isIncrementing	Boolean	作业是否为增量迁移。
delete_rows	Integer	删除数据行数。
update_rows	Integer	更新数据行数。
write_rows	Integer	写入数据行数。
submission-id	Integer	作业提交id。
job-name	String	作业名称。
creation-user	String	创建用户。
creation-date	Long	创建时间，单位：毫秒。
execute-date	Long	执行时间。
progress	Float	作业进度，失败时为“-1”，其它情况为0～100。
status	String	作业状态： • BOOTING：启动中。 • FAILURE_ON_SUBMIT：提交失败。 • RUNNING：运行中。 • SUCCEEDED：成功。 • FAILED：失败。 • UNKNOWN：未知。 • NEVER_EXECUTED：未被执行。
isStopingIncrement	String	是否停止增量迁移。
is-execute-auto	Boolean	是否定时执行作业。
last-update-date	Long	作业最后更新时间。
last-udpate-user	String	最后更新作业状态的用户。
isDeleteJob	Boolean	作业执行完成后是否删除。

请求示例

在CDM集群ID列表中随机选择一个集群，创建一个源端为elasticsearch，目的端为DIS，作业名为es_css的表迁移作业。

```
POST /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/job

{
  "jobs" : [ {
    "job_type" : "NORMAL_JOB",
    "from-connector-name" : "elasticsearch-connector",
    "to-config-values" : {
      "configs" : [ {
        "inputs" : [ {
          "name" : "toJobConfig.streamName",
          "value" : "dis-lkGm"
        }, {
          "name" : "toJobConfig.separator",
          "value" : "|"
        }, {
          "name" : "toJobConfig.columnList",
          "value" : "1&2&3"
        } ],
        "name" : "toJobConfig"
      } ]
    },
    "to-link-name" : "dis",
    "driver-config-values" : {
      "configs" : [ {
        "inputs" : [ {
          "name" : "throttlingConfig.numExtractors",
          "value" : "1"
        }, {
          "name" : "throttlingConfig.submitToCluster",
          "value" : "false"
        }, {
          "name" : "throttlingConfig.numLoaders",
          "value" : "1"
        }, {
          "name" : "throttlingConfig.recordDirtyData",
          "value" : "false"
        } ],
        "name" : "throttlingConfig"
      }, {
        "inputs" : [ ],
        "name" : "jarConfig"
      }, {
        "inputs" : [ {
          "name" : "schedulerConfig.isSchedulerJob",
          "value" : "false"
        }, {
          "name" : "schedulerConfig.disposableType",
          "value" : "NONE"
        } ],
        "name" : "schedulerConfig"
      }, {
        "inputs" : [ ],
        "name" : "transformConfig"
      }, {
        "inputs" : [ {
          "name" : "retryJobConfig.retryJobType",
          "value" : "NONE"
        } ],
        "name" : "retryJobConfig"
      } ]
    },
    "from-config-values" : {
      "configs" : [ {
        "inputs" : [ {
```

```
        "name" : "fromJobConfig.index",
        "value" : "52est"
      }, {
        "name" : "fromJobConfig.type",
        "value" : "est_array"
      }, {
        "name" : "fromJobConfig.columnList",
        "value" : "array_f1_int:long&array_f2_text:string&array_f3_object:nested"
      }, {
        "name" : "fromJobConfig.splitNestedField",
        "value" : "false"
      } ],
      "name" : "fromJobConfig"
    } ]
  },
  "to-connector-name" : "dis-connector",
  "name" : "es_css",
  "from-link-name" : "css"
 } ],
 "clusters" : [ "b0791496-e111-4e75-b7ca-9277aeab9297", "c2db1191-eb6c-464a-a0d3-b434e6c6df26",
"c2db1191-eb6c-464a-a0d3-b434e6c6df26" ]
}
```

响应示例

状态码：200

OK。

```
{
  "submissions" : [ {
    "isIncrementing" : false,
    "job-name" : "obs2obs-03",
    "submission-id" : 13,
    "isStopingIncrement" : "",
    "last-update-date" : 1635909057030,
    "is-execute-auto" : false,
    "delete_rows" : 0,
    "write_rows" : 0,
    "isDeleteJob" : false,
    "creation-user" : "cdmUser",
    "progress" : 0,
    "creation-date" : 1635909057030,
    "update_rows" : 0,
    "status" : "PENDING",
    "execute-date" : 1635909057030
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

在CDM集群ID列表中随机选择一个集群，创建一个源端为elasticsearch，目的端为DIS，作业名为es_css的表迁移作业。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
```

```
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class CreateAndStartRandomClusterJobSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAndStartRandomClusterJobRequest request = new CreateAndStartRandomClusterJobRequest();
        CdmRandomCreateAndStartJobJsonReq body = new CdmRandomCreateAndStartJobJsonReq();
        List<String> listbodyClusters = new ArrayList<>();
        listbodyClusters.add("b0791496-e111-4e75-b7ca-9277aeab9297");
        listbodyClusters.add("c2db1191-eb6c-464a-a0d3-b434e6c6df26");
        listbodyClusters.add("c2db1191-eb6c-464a-a0d3-b434e6c6df26");
        List<Input> listConfigsInputs = new ArrayList<>();
        listConfigsInputs.add(
            new Input()
                .withName("fromJobConfig.index")
                .withValue("52est")
        );
        listConfigsInputs.add(
            new Input()
                .withName("fromJobConfig.type")
                .withValue("est_array")
        );
        listConfigsInputs.add(
            new Input()
                .withName("fromJobConfig.columnList")
                .withValue("array_f1_int:long&array_f2_text:string&array_f3_object:nested")
        );
        listConfigsInputs.add(
            new Input()
                .withName("fromJobConfig.splitNestedField")
                .withValue("false")
        );
        List<Configs> listFromConfigValuesConfigs = new ArrayList<>();
        listFromConfigValuesConfigs.add(
            new Configs()
                .withInputs(listConfigsInputs)
                .withName("fromJobConfig")
        );
        ConfigValues fromconfigvaluesJobs = new ConfigValues();
        fromconfigvaluesJobs.withConfigs(listFromConfigValuesConfigs);
        List<Input> listConfigsInputs1 = new ArrayList<>();
        listConfigsInputs1.add(
            new Input()
                .withName("retryJobConfig.retryJobType")
                .withValue("NONE")
        );
        List<Input> listConfigsInputs2 = new ArrayList<>();
```

```
listConfigsInputs2.add(
    new Input()
        .withName("schedulerConfig.isSchedulerJob")
        .withValue("false")
);
listConfigsInputs2.add(
    new Input()
        .withName("schedulerConfig.disposableType")
        .withValue("NONE")
);
List<Input> listConfigsInputs3 = new ArrayList<>();
listConfigsInputs3.add(
    new Input()
        .withName("throttlingConfig.numExtractors")
        .withValue("1")
);
listConfigsInputs3.add(
    new Input()
        .withName("throttlingConfig.submitToCluster")
        .withValue("false")
);
listConfigsInputs3.add(
    new Input()
        .withName("throttlingConfig.numLoaders")
        .withValue("1")
);
listConfigsInputs3.add(
    new Input()
        .withName("throttlingConfig.recordDirtyData")
        .withValue("false")
);
List<Configs> listDriverConfigValuesConfigs = new ArrayList<>();
listDriverConfigValuesConfigs.add(
    new Configs()
        .withInputs(listConfigsInputs1)
        .withName("retryJobConfig")
);
ConfigValues driverconfigvaluesJobs = new ConfigValues();
driverconfigvaluesJobs.withConfigs(listDriverConfigValuesConfigs);
List<Input> listConfigsInputs4 = new ArrayList<>();
listConfigsInputs4.add(
    new Input()
        .withName("toJobConfig.streamName")
        .withValue("dis-lkGm")
);
listConfigsInputs4.add(
    new Input()
        .withName("toJobConfig.separator")
        .withValue("|")
);
listConfigsInputs4.add(
    new Input()
        .withName("toJobConfig.columnList")
        .withValue("1&2&3")
);
List<Configs> listToConfigValuesConfigs = new ArrayList<>();
listToConfigValuesConfigs.add(
    new Configs()
        .withInputs(listConfigsInputs4)
        .withName("toJobConfig")
);
ConfigValues toconfigvaluesJobs = new ConfigValues();
toconfigvaluesJobs.withConfigs(listToConfigValuesConfigs);
List<Job> listbodyJobs = new ArrayList<>();
listbodyJobs.add(
    new Job()
        .withJobType(Job.JobTypeEnum.fromValue("NORMAL_JOB"))
        .withFromConnectorName("elasticsearch-connector")
        .withToConfigValues(toconfigvaluesJobs)
```

```
                .withToLinkName("dis")
                .withDriverConfigValues(driverconfigvaluesJobs)
                .withFromConfigValues(fromconfigvaluesJobs)
                .withToConnectorName("dis-connector")
                .withName("es_css")
                .withFromLinkName("css")
        );
        body.withClusters(listbodyClusters);
        body.withJobs(listbodyJobs);
        request.withBody(body);
        try {
            CreateAndStartRandomClusterJobResponse response =
client.createAndStartRandomClusterJob(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

在CDM集群ID列表中随机选择一个集群，创建一个源端为elasticsearch，目的端为DIS，作业名为es_css的表迁移作业。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAndStartRandomClusterJobRequest()
        listClustersbody = [
            "b0791496-e111-4e75-b7ca-9277aeab9297",
            "c2db1191-eb6c-464a-a0d3-b434e6c6df26",
            "c2db1191-eb6c-464a-a0d3-b434e6c6df26"
        ]
        listInputsConfigs = [
            Input(
                name="fromJobConfig.index",
                value="52est"
```

```
    ),
    Input(
        name="fromJobConfig.type",
        value="est_array"
    ),
    Input(
        name="fromJobConfig.columnList",
        value="array_f1_int:long&array_f2_text:string&array_f3_object:nested"
    ),
    Input(
        name="fromJobConfig.splitNestedField",
        value="false"
    )
]
listConfigsFromconfigvalues = [
    Configs(
        inputs=listInputsConfigs,
        name="fromJobConfig"
    )
]
fromconfigvaluesJobs = ConfigValues(
    configs=listConfigsFromconfigvalues
)
listInputsConfigs1 = [
    Input(
        name="retryJobConfig.retryJobType",
        value="NONE"
    )
]
listInputsConfigs2 = [
    Input(
        name="schedulerConfig.isSchedulerJob",
        value="false"
    ),
    Input(
        name="schedulerConfig.disposableType",
        value="NONE"
    )
]
listInputsConfigs3 = [
    Input(
        name="throttlingConfig.numExtractors",
        value="1"
    ),
    Input(
        name="throttlingConfig.submitToCluster",
        value="false"
    ),
    Input(
        name="throttlingConfig.numLoaders",
        value="1"
    ),
    Input(
        name="throttlingConfig.recordDirtyData",
        value="false"
    )
]
listConfigsDriverconfigvalues = [
    Configs(
        inputs=listInputsConfigs1,
        name="retryJobConfig"
    )
]
driverconfigvaluesJobs = ConfigValues(
    configs=listConfigsDriverconfigvalues
)
listInputsConfigs4 = [
    Input(
        name="toJobConfig.streamName",
```

```
            value="dis-lkGm"
        ),
        Input(
            name="toJobConfig.separator",
            value="|"
        ),
        Input(
            name="toJobConfig.columnList",
            value="1&2&3"
        )
    ]
    listConfigsToconfigvalues = [
        Configs(
            inputs=listInputsConfigs4,
            name="toJobConfig"
        )
    ]
    toconfigvaluesJobs = ConfigValues(
        configs=listConfigsToconfigvalues
    )
    listJobsbody = [
        Job(
            job_type="NORMAL_JOB",
            from_connector_name="elasticsearch-connector",
            to_config_values=toconfigvaluesJobs,
            to_link_name="dis",
            driver_config_values=driverconfigvaluesJobs,
            from_config_values=fromconfigvaluesJobs,
            to_connector_name="dis-connector",
            name="es_css",
            from_link_name="css"
        )
    ]
    request.body = CdmRandomCreateAndStartJobJsonReq(
        clusters=listClustersbody,
        jobs=listJobsbody
    )
    response = client.create_and_start_random_cluster_job(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

在CDM集群ID列表中随机选择一个集群，创建一个源端为elasticsearch，目的端为DIS，作业名为es_css的表迁移作业。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"
```

```
auth := basic.NewCredentialsBuilder().
    WithAk(ak).
    WithSk(sk).
    WithProjectId(projectId).
    Build()

client := cdm.NewCdmClient(
    cdm.CdmClientBuilder().
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

request := &model.CreateAndStartRandomClusterJobRequest{}
var listClustersbody = []string{
    "b0791496-e111-4e75-b7ca-9277aeab9297",
    "c2db1191-eb6c-464a-a0d3-b434e6c6df26",
    "c2db1191-eb6c-464a-a0d3-b434e6c6df26",
}
valueInputs:= "52est"
var valueInputsInterface interface{} = valueInputs
valueInputs1:= "est_array"
var valueInputsInterface1 interface{} = valueInputs1
valueInputs2:= "array_f1_int:long&array_f2_text:string&array_f3_object:nested"
var valueInputsInterface2 interface{} = valueInputs2
valueInputs3:= "false"
var valueInputsInterface3 interface{} = valueInputs3
var listInputsConfigs = []model.Input{
    {
        Name: "fromJobConfig.index",
        Value: &valueInputsInterface,
    },
    {
        Name: "fromJobConfig.type",
        Value: &valueInputsInterface1,
    },
    {
        Name: "fromJobConfig.columnList",
        Value: &valueInputsInterface2,
    },
    {
        Name: "fromJobConfig.splitNestedField",
        Value: &valueInputsInterface3,
    },
}
var listConfigsFromConfigValues = []model.Configs{
    {
        Inputs: listInputsConfigs,
        Name: "fromJobConfig",
    },
}
fromconfigvaluesJobs := &model.ConfigValues{
    Configs: listConfigsFromConfigValues,
}
valueInputs4:= "NONE"
var valueInputsInterface4 interface{} = valueInputs4
var listInputsConfigs1 = []model.Input{
    {
        Name: "retryJobConfig.retryJobType",
        Value: &valueInputsInterface4,
    },
}
valueInputs5:= "false"
var valueInputsInterface5 interface{} = valueInputs5
valueInputs6:= "NONE"
var valueInputsInterface6 interface{} = valueInputs6
var listInputsConfigs2 = []model.Input{
    {
        Name: "schedulerConfig.isSchedulerJob",
```

```
        Value: &valueInputsInterface5,
    },
    {
        Name: "schedulerConfig.disposableType",
        Value: &valueInputsInterface6,
    },
}
valueInputs7:= "1"
var valueInputsInterface7 interface{} = valueInputs7
valueInputs8:= "false"
var valueInputsInterface8 interface{} = valueInputs8
valueInputs9:= "1"
var valueInputsInterface9 interface{} = valueInputs9
valueInputs10:= "false"
var valueInputsInterface10 interface{} = valueInputs10
var listInputsConfigs3 = []model.Input{
    {
        Name: "throttlingConfig.numExtractors",
        Value: &valueInputsInterface7,
    },
    {
        Name: "throttlingConfig.submitToCluster",
        Value: &valueInputsInterface8,
    },
    {
        Name: "throttlingConfig.numLoaders",
        Value: &valueInputsInterface9,
    },
    {
        Name: "throttlingConfig.recordDirtyData",
        Value: &valueInputsInterface10,
    },
}
var listConfigsDriverConfigValues = []model.Configs{
    {
        Inputs: listInputsConfigs1,
        Name: "retryJobConfig",
    },
}
driverconfigvaluesJobs := &model.ConfigValues{
    Configs: listConfigsDriverConfigValues,
}
valueInputs11:= "dis-lkGm"
var valueInputsInterface11 interface{} = valueInputs11
valueInputs12:= "|"
var valueInputsInterface12 interface{} = valueInputs12
valueInputs13:= "1&2&3"
var valueInputsInterface13 interface{} = valueInputs13
var listInputsConfigs4 = []model.Input{
    {
        Name: "toJobConfig.streamName",
        Value: &valueInputsInterface11,
    },
    {
        Name: "toJobConfig.separator",
        Value: &valueInputsInterface12,
    },
    {
        Name: "toJobConfig.columnList",
        Value: &valueInputsInterface13,
    },
}
var listConfigsToConfigValues = []model.Configs{
    {
        Inputs: listInputsConfigs4,
        Name: "toJobConfig",
    },
}
toconfigvaluesJobs := &model.ConfigValues{
```

```
        Configs: listConfigsToConfigValues,
    }
    var listJobsbody = []model.Job{
        {
            JobType: model.GetJobJobTypeEnum().NORMAL_JOB,
            FromConnectorName: "elasticsearch-connector",
            ToConfigValues: toconfigvaluesJobs,
            ToLinkName: "dis",
            DriverConfigValues: driverconfigvaluesJobs,
            FromConfigValues: fromconfigvaluesJobs,
            ToConnectorName: "dis-connector",
            Name: "es_css",
            FromLinkName: "css",
        },
    }
    request.Body = &model.CdmRandomCreateAndStartJobJsonReq{
        Clusters: listClustersbody,
        Jobs: listJobsbody,
    }
    response, err := client.CreateAndStartRandomClusterJob(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。

错误码

请参见**错误码**。

5.2.5 停止作业

功能介绍

停止作业接口。

调用方法

请参见**如何调用API**。

URI

PUT /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{job_name}/stop

表 5-134 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。
job_name	是	String	作业名称。

请求参数

表 5-135 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码：200

表 5-136 响应 Body 参数

参数	参数类型	描述
validation-result	Array of **JobValidationResult** objects	校验结构：如果停止作业接失败，返回失败原因，请参见validation-result参数说明。如果停止成功，返回空列表。

表 5-137 JobValidationResult

参数	参数类型	描述
message	String	错误描述。
status	String	错误级别，如：ERROR、WARNING。

请求示例

```
PUT /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/job/jdbc2hive/stop
```

响应示例

状态码：200

OK。

```
{ }
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class StopJobSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        StopJobRequest request = new StopJobRequest();
        request.withClusterId("{cluster_id}");
        request.withJobName("{job_name}");
        try {
            StopJobResponse response = client.stopJob(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = StopJobRequest()
        request.cluster_id = "{cluster_id}"
        request.job_name = "{job_name}"
        response = client.stop_job(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
```

```
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

    request := &model.StopJobRequest{}
    request.ClusterId = "{cluster_id}"
    request.JobName = "{job_name}"
    response, err := client.StopJob(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。

错误码

请参见**错误码**。

5.2.6 指定集群创建作业

功能介绍

指定集群创建作业接口。

调用方法

请参见**如何调用API**。

URI

POST /v1.1/{project_id}/clusters/{cluster_id}/cdm/job

表 5-138 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。

请求参数

表 5-139 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 5-140 请求 Body 参数

参数	是否必选	参数类型	描述
jobs	是	Array of **Job** objects	作业列表，请参见jobs数据结构说明。

表 5-141 Job

参数	是否必选	参数类型	描述
job_type	是	String	作业类型： • NORMAL_JOB：表/文件迁移。 • BATCH_JOB：整库迁移。 • SCENARIO_JOB：场景迁移。

参数	是否必选	参数类型	描述
from-connector-name	是	String	源端连接类型，对应的连接参数如下： • generic-jdbc-connector：关系数据库连接。 • obs-connector：OBS连接。 • hdfs-connector：HDFS连接。 • hbase-connector：HBase连接、CloudTable连接。 • hive-connector：Hive连接。 • ftp-connector/sftp-connector：FTP/SFTP连接。 • mongodb-connector：MongoDB连接。 • redis-connector：Redis/DCS连接。 • kafka-connector：Kafka连接。 • dis-connector：DIS连接。 • elasticsearch-connector：Elasticsearch/云搜索服务连接。 • dli-connector：DLI连接。 • http-connector：HTTP/HTTPS连接，该连接暂无连接参数。 • dms-kafka-connector：DMSKafka连接。
to-config-values	是	**ConfigValues** object	目的连接参数配置。根据不同目的端有不同的参数配置，具体可参考**目的端作业参数说明**下相应的目的端参数配置。
to-link-name	是	String	目的端连接名称，即为通过“创建连接”接口创建的连接对应的连接名。
driver-config-values	是	**ConfigValues** object	作业任务参数配置。例如配置作业失败重试、抽取并发数，具体可参考**作业任务参数说明**。

参数	是否必选	参数类型	描述
from-config-values	是	ConfigValues object	源连接参数配置。根据不同源端有不同的参数配置，具体可参考源端作业参数说明下相应的源端参数配置。
to-connector-name	是	String	目的端连接类型，对应的连接参数如下： • generic-jdbc-connector：关系数据库连接。 • obs-connector：OBS连接。 • hdfs-connector：HDFS连接。 • hbase-connector：HBase连接、CloudTable连接。 • hive-connector：Hive连接。 • ftp-connector/sftp-connector：FTP/SFTP连接。 • mongodb-connector：MongoDB连接。 • redis-connector：Redis/DCS连接。 • kafka-connector：Kafka连接。 • dis-connector：DIS连接。 • elasticsearch-connector：Elasticsearch/云搜索服务连接。 • dli-connector：DLI连接。 • http-connector：HTTP/HTTPS连接，该连接暂无连接参数。 • dms-kafka-connector：DMSKafka连接。
name	是	String	作业名称，长度在1到240个字符之间。
from-link-name	是	String	源连接名称，即为通过“创建连接”接口创建的连接对应的连接名。
creation-user	否	String	创建作业的用户。由系统生成，用户无需填写。
creation-date	否	Long	作业创建的时间，单位：毫秒。由系统生成，用户无需填写。

参数	是否必选	参数类型	描述
update-date	否	Long	作业最后更新的时间，单位：毫秒。由系统生成，用户无需填写。
is_incre_job	否	Boolean	是否是增量作业。已废弃。
flag	否	Integer	是否是定时作业标记，如果是定时作业则为1，否则为0。由系统根据定时任务配置生成，用户无需填写。
files_read	否	Integer	已读文件数。由系统生成，用户无需填写。
update-user	否	String	最后更新作业的用户。由系统生成，用户无需填写。
external_id	否	String	具体执行的作业id，如果是本地作业，则一般为"job_local1202051771_0002"形式，如果是DLI作业，则为DLI作业ID，比如"12345"。由系统生成，用户无需填写。
type	否	String	与job_type一致，作业类型： ● NORMAL_JOB：表/文件迁移。 ● BATCH_JOB：整库迁移。 ● SCENARIO_JOB：场景迁移。
execute_start_date	否	Long	最近一次执行任务开始时间，单位：毫秒。由系统生成，用户无需填写。
delete_rows	否	Integer	增量作业删除行数，已废弃。
enabled	否	Boolean	是否激活连接。由系统生成，用户无需填写。
bytes_written	否	Long	作业写入的字节。由系统生成，用户无需填写。
id	否	Integer	作业ID。由系统生成，用户无需填写。
is_use_sql	否	Boolean	用户是否使用sql。由系统根据源端抽取是否使用sql语句生成，用户无需填写。
update_rows	否	Integer	增量作业更新行数，已废弃。
group_name	否	String	组名。

参数	是否必选	参数类型	描述
bytes_read	否	Long	作业读取的字节。由系统生成，用户无需填写。
execute_update_date	否	Long	最近一次执行任务更新时间，单位：毫秒。由系统生成，用户无需填写。
write_rows	否	Integer	增量作业写入行数，已废弃。
rows_written	否	Integer	作业写入的行数。由系统生成，用户无需填写。
rows_read	否	Long	作业读取的行数。由系统生成，用户无需填写。
files_written	否	Integer	写入文件数。由系统生成，用户无需填写。
is_incrementing	否	Boolean	是否是增量作业，同is_incre_job，已废弃。
execute_create_date	否	Long	最近一次执行任务创建时间，单位：毫秒。由系统生成，用户无需填写。
status	否	String	作业最后的执行状态： • BOOTING：启动中。 • RUNNING：运行中。 • SUCCEEDED：成功。 • FAILED：失败。 • NEW：未被执行。

表 5-142 ConfigValues

参数	是否必选	参数类型	描述
configs	是	Array of **configs** objects	源连接参数、目的连接参数和作业任务参数，它们的配置数据结构相同，其中“inputs”里的参数不一样，详细请参见configs数据结构说明。
extended-configs	否	**extended-configs** object	扩展配置，请参见extended-configs参数说明。扩展配置暂不对外开放，用户无需填写。

表 5-143 configs

参数	是否必选	参数类型	描述
inputs	是	Array of **Input** objects	输入参数列表，列表中的每个参数为“name,value”结构，请参考inputs数据结构参数说明。在“from-config-values”数据结构中，不同的源连接类型有不同的“inputs”参数列表，请参见源端作业参数说明下的章节。在“to-config-values”数据结构中，不同的目的连接类型有不同的“inputs”参数列表，请参见目的端作业参数说明下面的子章节。在“driver-config-values”数据结构中，“inputs”具体参数请参见作业任务参数说明。
name	是	String	配置名称：源端作业的配置名称为“fromJobConfig”。目的端作业的配置名称为“toJobConfig”，连接的配置名称固定为“linkConfig”。
id	否	Integer	配置ID，由系统生成，用户无需填写。
type	否	String	配置类型，由系统生成，用户无需填写。值为LINK或者JOB，如果是连接管理API，则为LINK；如果是作业管理API，则为JOB。

表 5-144 Input

参数	是否必选	参数类型	描述
name	是	String	参数名: • 如果是连接管理API，则以“linkConfig.”开头，对于不同连接类型有不同的参数，具体可参见**连接参数说明**下相应连接的参数说明。 • 如果是作业管理API，对于源端连接参数，则以“fromJobConfig.”开头，具体可参见**源端作业参数说明**下相应的源端参数说明；对于目的端连接参数，则以“toJobConfig.”开头，具体可参见**目的端作业参数说明**下相应的目的端参数说明；对于作业任务参数，请参见**作业任务参数说明**下相应的任务参数说明。
value	是	Object	参数值，参数名对应的值，必须填写为字符串。
type	否	String	值类型，如STRING、INTEGER，由系统设定，用户无需填写。

表 5-145 extended-configs

参数	是否必选	参数类型	描述
name	否	String	扩展配置名称，暂不对外开放，用户无需填写。
value	否	String	扩展配置值，暂不对外开放，用户无需填写。

响应参数

状态码：200

表 5-146 响应 Body 参数

参数	参数类型	描述
name	String	作业名称。

参数	参数类型	描述
validation-result	Array of **JobValidationResult** objects	校验结果： • 如果修改失败，返回失败原因。 • 如果修改成功，返回空列表。

表 5-147 JobValidationResult

参数	参数类型	描述
message	String	错误描述。
status	String	错误级别，如：ERROR、WARNING。

状态码：400

表 5-148 响应 Body 参数

参数	参数类型	描述
code	String	返回编码。
errCode	String	错误码。
message	String	报错信息。
externalMessage	String	附加信息。

请求示例

创建一个源端为Elasticsearch数据连接，目的端为DIS数据连接，作业名为es_css的数据迁移作业。

```
POST /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/job

{
  "jobs" : [ {
    "job_type" : "NORMAL_JOB",
    "from-connector-name" : "elasticsearch-connector",
    "to-config-values" : {
      "configs" : [ {
        "inputs" : [ {
          "name" : "toJobConfig.streamName",
          "value" : "dis-lkGm"
        }, {
          "name" : "toJobConfig.separator",
          "value" : "|"
        }, {
          "name" : "toJobConfig.columnList",
          "value" : "1&2&3"
        } ],
        "name" : "toJobConfig"
      } ]
    },
```

```
  "to-link-name" : "dis",
  "driver-config-values" : {
    "configs" : [ {
      "inputs" : [ {
        "name" : "throttlingConfig.numExtractors",
        "value" : "1"
      }, {
        "name" : "throttlingConfig.submitToCluster",
        "value" : "false"
      }, {
        "name" : "throttlingConfig.numLoaders",
        "value" : "1"
      }, {
        "name" : "throttlingConfig.recordDirtyData",
        "value" : "false"
      } ],
      "name" : "throttlingConfig"
    }, {
      "inputs" : [ ],
      "name" : "jarConfig"
    }, {
      "inputs" : [ {
        "name" : "schedulerConfig.isSchedulerJob",
        "value" : "false"
      }, {
        "name" : "schedulerConfig.disposableType",
        "value" : "NONE"
      } ],
      "name" : "schedulerConfig"
    }, {
      "inputs" : [ ],
      "name" : "transformConfig"
    }, {
      "inputs" : [ {
        "name" : "retryJobConfig.retryJobType",
        "value" : "NONE"
      } ],
      "name" : "retryJobConfig"
    } ]
  },
  "from-config-values" : {
    "configs" : [ {
      "inputs" : [ {
        "name" : "fromJobConfig.index",
        "value" : "52est"
      }, {
        "name" : "fromJobConfig.type",
        "value" : "est_array"
      }, {
        "name" : "fromJobConfig.columnList",
        "value" : "array_f1_int:long&array_f2_text:string&array_f3_object:nested"
      }, {
        "name" : "fromJobConfig.splitNestedField",
        "value" : "false"
      } ],
      "name" : "fromJobConfig"
    } ]
  },
  "to-connector-name" : "dis-connector",
  "name" : "es_css",
  "from-link-name" : "css"
 } ]
}
```


响应示例

状态码：200

OK。

```
{
  "name" : "mysql2hive"
}
```

状态码：400

请求报错。

```
{
  "code" : "Cdm.0104",
  "errCode" : "Cdm.0104",
  "message" : "Job name already exist or created by other.",
  "ternalMessage" : "Job name already exist or created by other."
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建一个源端为Elasticsearch数据连接，目的端为DIS数据连接，作业名为es_css的数据迁移作业。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class CreateJobSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateJobRequest request = new CreateJobRequest();
        request.withClusterId("{cluster_id}");
        CdmCreateJobJsonReq body = new CdmCreateJobJsonReq();
        List<Input> listConfigsInputs = new ArrayList<>();
        listConfigsInputs.add(
            new Input()
                .withName("fromJobConfig.index")
                .withValue("52est")
```

```
);
listConfigsInputs.add(
    new Input()
        .withName("fromJobConfig.type")
        .withValue("est_array")
);
listConfigsInputs.add(
    new Input()
        .withName("fromJobConfig.columnList")
        .withValue("array_f1_int:long&array_f2_text:string&array_f3_object:nested")
);
listConfigsInputs.add(
    new Input()
        .withName("fromJobConfig.splitNestedField")
        .withValue("false")
);
List<Configs> listFromConfigValuesConfigs = new ArrayList<>();
listFromConfigValuesConfigs.add(
    new Configs()
        .withInputs(listConfigsInputs)
        .withName("fromJobConfig")
);
ConfigValues fromconfigvaluesJobs = new ConfigValues();
fromconfigvaluesJobs.withConfigs(listFromConfigValuesConfigs);
List<Input> listConfigsInputs1 = new ArrayList<>();
listConfigsInputs1.add(
    new Input()
        .withName("retryJobConfig.retryJobType")
        .withValue("NONE")
);
List<Input> listConfigsInputs2 = new ArrayList<>();
listConfigsInputs2.add(
    new Input()
        .withName("schedulerConfig.isSchedulerJob")
        .withValue("false")
);
listConfigsInputs2.add(
    new Input()
        .withName("schedulerConfig.disposableType")
        .withValue("NONE")
);
List<Input> listConfigsInputs3 = new ArrayList<>();
listConfigsInputs3.add(
    new Input()
        .withName("throttlingConfig.numExtractors")
        .withValue("1")
);
listConfigsInputs3.add(
    new Input()
        .withName("throttlingConfig.submitToCluster")
        .withValue("false")
);
listConfigsInputs3.add(
    new Input()
        .withName("throttlingConfig.numLoaders")
        .withValue("1")
);
listConfigsInputs3.add(
    new Input()
        .withName("throttlingConfig.recordDirtyData")
        .withValue("false")
);
List<Configs> listDriverConfigValuesConfigs = new ArrayList<>();
listDriverConfigValuesConfigs.add(
    new Configs()
        .withInputs(listConfigsInputs1)
        .withName("retryJobConfig")
);
ConfigValues driverconfigvaluesJobs = new ConfigValues();
```

```
        driverconfigvaluesJobs.withConfigs(listDriverConfigValuesConfigs);
        List<Input> listConfigsInputs4 = new ArrayList<>();
        listConfigsInputs4.add(
            new Input()
                .withName("toJobConfig.streamName")
                .withValue("dis-lkGm")
        );
        listConfigsInputs4.add(
            new Input()
                .withName("toJobConfig.separator")
                .withValue("|")
        );
        listConfigsInputs4.add(
            new Input()
                .withName("toJobConfig.columnList")
                .withValue("1&2&3")
        );
        List<Configs> listToConfigValuesConfigs = new ArrayList<>();
        listToConfigValuesConfigs.add(
            new Configs()
                .withInputs(listConfigsInputs4)
                .withName("toJobConfig")
        );
        ConfigValues toconfigvaluesJobs = new ConfigValues();
        toconfigvaluesJobs.withConfigs(listToConfigValuesConfigs);
        List<Job> listbodyJobs = new ArrayList<>();
        listbodyJobs.add(
            new Job()
                .withJobType(Job.JobTypeEnum.fromValue("NORMAL_JOB"))
                .withFromConnectorName("elasticsearch-connector")
                .withToConfigValues(toconfigvaluesJobs)
                .withToLinkName("dis")
                .withDriverConfigValues(driverconfigvaluesJobs)
                .withFromConfigValues(fromconfigvaluesJobs)
                .withToConnectorName("dis-connector")
                .withName("es_css")
                .withFromLinkName("css")
        );
        body.withJobs(listbodyJobs);
        request.withBody(body);
        try {
            CreateJobResponse response = client.createJob(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建一个源端为Elasticsearch数据连接，目的端为DIS数据连接，作业名为es_css的数据迁移作业。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *
```

```
if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateJobRequest()
        request.cluster_id = "{cluster_id}"
        listInputsConfigs = [
            Input(
                name="fromJobConfig.index",
                value="52est"
            ),
            Input(
                name="fromJobConfig.type",
                value="est_array"
            ),
            Input(
                name="fromJobConfig.columnList",
                value="array_f1_int:long&array_f2_text:string&array_f3_object:nested"
            ),
            Input(
                name="fromJobConfig.splitNestedField",
                value="false"
            )
        ]
        listConfigsFromconfigvalues = [
            Configs(
                inputs=listInputsConfigs,
                name="fromJobConfig"
            )
        ]
        fromconfigvaluesJobs = ConfigValues(
            configs=listConfigsFromconfigvalues
        )
        listInputsConfigs1 = [
            Input(
                name="retryJobConfig.retryJobType",
                value="NONE"
            )
        ]
        listInputsConfigs2 = [
            Input(
                name="schedulerConfig.isSchedulerJob",
                value="false"
            ),
            Input(
                name="schedulerConfig.disposableType",
                value="NONE"
            )
        ]
        listInputsConfigs3 = [
            Input(
                name="throttlingConfig.numExtractors",
                value="1"
            ),
```

```
            Input(
                name="throttlingConfig.submitToCluster",
                value="false"
            ),
            Input(
                name="throttlingConfig.numLoaders",
                value="1"
            ),
            Input(
                name="throttlingConfig.recordDirtyData",
                value="false"
            )
        ]
        listConfigsDriverconfigvalues = [
            Configs(
                inputs=listInputsConfigs1,
                name="retryJobConfig"
            )
        ]
        driverconfigvaluesJobs = ConfigValues(
            configs=listConfigsDriverconfigvalues
        )
        listInputsConfigs4 = [
            Input(
                name="toJobConfig.streamName",
                value="dis-lkGm"
            ),
            Input(
                name="toJobConfig.separator",
                value="|"
            ),
            Input(
                name="toJobConfig.columnList",
                value="1&2&3"
            )
        ]
        listConfigsToconfigvalues = [
            Configs(
                inputs=listInputsConfigs4,
                name="toJobConfig"
            )
        ]
        toconfigvaluesJobs = ConfigValues(
            configs=listConfigsToconfigvalues
        )
        listJobsbody = [
            Job(
                job_type="NORMAL_JOB",
                from_connector_name="elasticsearch-connector",
                to_config_values=toconfigvaluesJobs,
                to_link_name="dis",
                driver_config_values=driverconfigvaluesJobs,
                from_config_values=fromconfigvaluesJobs,
                to_connector_name="dis-connector",
                name="es_css",
                from_link_name="css"
            )
        ]
        request.body = CdmCreateJobJsonReq(
            jobs=listJobsbody
        )
        response = client.create_job(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建一个源端为Elasticsearch数据连接，目的端为DIS数据连接，作业名为es_css的数据迁移作业。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateJobRequest{}
    request.ClusterId = "{cluster_id}"
    valueInputs:= "52est"
    var valueInputsInterface interface{} = valueInputs
    valueInputs1:= "est_array"
    var valueInputsInterface1 interface{} = valueInputs1
    valueInputs2:= "array_f1_int:long&array_f2_text:string&array_f3_object:nested"
    var valueInputsInterface2 interface{} = valueInputs2
    valueInputs3:= "false"
    var valueInputsInterface3 interface{} = valueInputs3
    var listInputsConfigs = []model.Input{
        {
            Name: "fromJobConfig.index",
            Value: &valueInputsInterface,
        },
        {
            Name: "fromJobConfig.type",
            Value: &valueInputsInterface1,
        },
        {
            Name: "fromJobConfig.columnList",
            Value: &valueInputsInterface2,
        },
        {
            Name: "fromJobConfig.splitNestedField",
            Value: &valueInputsInterface3,
        },
    }
    var listConfigsFromConfigValues = []model.Configs{
        {
            Inputs: listInputsConfigs,
            Name: "fromJobConfig",
```

```
    },
}
fromconfigvaluesJobs := &model.ConfigValues{
    Configs: listConfigsFromConfigValues,
}
valueInputs4:= "NONE"
var valueInputsInterface4 interface{} = valueInputs4
var listInputsConfigs1 = []model.Input{
    {
        Name: "retryJobConfig.retryJobType",
        Value: &valueInputsInterface4,
    },
}
valueInputs5:= "false"
var valueInputsInterface5 interface{} = valueInputs5
valueInputs6:= "NONE"
var valueInputsInterface6 interface{} = valueInputs6
var listInputsConfigs2 = []model.Input{
    {
        Name: "schedulerConfig.isSchedulerJob",
        Value: &valueInputsInterface5,
    },
    {
        Name: "schedulerConfig.disposableType",
        Value: &valueInputsInterface6,
    },
}
valueInputs7:= "1"
var valueInputsInterface7 interface{} = valueInputs7
valueInputs8:= "false"
var valueInputsInterface8 interface{} = valueInputs8
valueInputs9:= "1"
var valueInputsInterface9 interface{} = valueInputs9
valueInputs10:= "false"
var valueInputsInterface10 interface{} = valueInputs10
var listInputsConfigs3 = []model.Input{
    {
        Name: "throttlingConfig.numExtractors",
        Value: &valueInputsInterface7,
    },
    {
        Name: "throttlingConfig.submitToCluster",
        Value: &valueInputsInterface8,
    },
    {
        Name: "throttlingConfig.numLoaders",
        Value: &valueInputsInterface9,
    },
    {
        Name: "throttlingConfig.recordDirtyData",
        Value: &valueInputsInterface10,
    },
}
var listConfigsDriverConfigValues = []model.Configs{
    {
        Inputs: listInputsConfigs1,
        Name: "retryJobConfig",
    },
}
driverconfigvaluesJobs := &model.ConfigValues{
    Configs: listConfigsDriverConfigValues,
}
valueInputs11:= "dis-lkGm"
var valueInputsInterface11 interface{} = valueInputs11
valueInputs12:= "|"
var valueInputsInterface12 interface{} = valueInputs12
valueInputs13:= "1&2&3"
var valueInputsInterface13 interface{} = valueInputs13
var listInputsConfigs4 = []model.Input{
```

```
        {
            Name: "toJobConfig.streamName",
            Value: &valueInputsInterface11,
        },
        {
            Name: "toJobConfig.separator",
            Value: &valueInputsInterface12,
        },
        {
            Name: "toJobConfig.columnList",
            Value: &valueInputsInterface13,
        },
    }
    var listConfigsToConfigValues = []model.Configs{
        {
            Inputs: listInputsConfigs4,
            Name: "toJobConfig",
        },
    }
    toconfigvaluesJobs := &model.ConfigValues{
        Configs: listConfigsToConfigValues,
    }
    var listJobsbody = []model.Job{
        {
            JobType: model.GetJobJobTypeEnum().NORMAL_JOB,
            FromConnectorName: "elasticsearch-connector",
            ToConfigValues: toconfigvaluesJobs,
            ToLinkName: "dis",
            DriverConfigValues: driverconfigvaluesJobs,
            FromConfigValues: fromconfigvaluesJobs,
            ToConnectorName: "dis-connector",
            Name: "es_css",
            FromLinkName: "css",
        },
    }
    request.Body = &model.CdmCreateJobJsonReq{
        Jobs: listJobsbody,
    }
    response, err := client.CreateJob(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求报错。

错误码

请参见**错误码**。

5.2.7 启动作业

功能介绍

启动作业接口。

调用方法

请参见**如何调用API**。

URI

PUT /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{job_name}/start

表 5-149 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。
job_name	是	String	作业名称。

请求参数

表 5-150 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 5-151 请求 Body 参数

参数	是否必选	参数类型	描述
variables	否	Map<String,String>	启动作业参数，用于设置变量参数值，Map<String, String>结构。作业配置中无变量时，可以设置为空对象。

响应参数

状态码：200

表 5-152 响应 Body 参数

参数	参数类型	描述
submissions	Array of StartJobSubmission objects	作业运行信息，请参见submission参数说明。

表 5-153 StartJobSubmission

参数	参数类型	描述
isIncrementing	Boolean	作业是否为增量迁移。
delete_rows	Integer	删除数据行数。
update_rows	Integer	更新数据行数。
write_rows	Integer	写入数据行数。
submission-id	Integer	作业提交id。
job-name	String	作业名称。
creation-user	String	创建用户。
creation-date	Long	创建时间，单位：毫秒。
execute-date	Long	执行时间。
progress	Float	作业进度，失败时为“-1”，其它情况为0～100。
status	String	作业状态： • BOOTING：启动中。 • FAILURE_ON_SUBMIT：提交失败。 • RUNNING：运行中。 • SUCCEEDED：成功。 • FAILED：失败。 • UNKNOWN：未知。 • NEVER_EXECUTED：未被执行。
isStopingIncrement	String	是否停止增量迁移。
is-execute-auto	Boolean	是否定时执行作业。
last-update-date	Long	作业最后更新时间。
last-udpate-user	String	最后更新作业状态的用户。
isDeleteJob	Boolean	作业执行完成后是否删除。

请求示例

启动作业，参数为空。

```
PUT /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/job/jdbc2hive/start

{
  "variables" : { }
}
```

响应示例

状态码：200

OK。

```
{
  "submissions" : [ {
    "job-name" : "jdbc2hive",
    "creation-user" : "cdm",
    "creation-date" : "1536905778725",
    "progress" : 1,
    "status" : "BOOTING"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

启动作业，参数为空。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class StartJobSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
```

```
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        StartJobRequest request = new StartJobRequest();
        request.withClusterId("{cluster_id}");
        request.withJobName("{job_name}");
        CdmStartJobReq body = new CdmStartJobReq();
        body.withVariables(new Object());
        request.withBody(body);
        try {
            StartJobResponse response = client.startJob(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

启动作业，参数为空。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = StartJobRequest()
        request.cluster_id = "{cluster_id}"
        request.job_name = "{job_name}"
        request.body = CdmStartJobReq(
            variables={}
        )
        response = client.start_job(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

启动作业，参数为空。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.StartJobRequest{}
    request.ClusterId = "{cluster_id}"
    request.JobName = "{job_name}"
    var variablesCdmStartJobReq interface{} = make(map[string]string)
    request.Body = &model.CdmStartJobReq{
        Variables: &variablesCdmStartJobReq,
    }
    response, err := client.StartJob(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。

错误码

请参见**错误码**。

5.2.8 查询作业状态

功能介绍

查询作业状态接口。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{job_name}/status

表 5-154 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。
job_name	是	String	作业名称。

请求参数

表 5-155 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码：200

表 5-156 响应 Body 参数

参数	参数类型	描述
submissions	Array of **Submission** objects	作业运行信息，详见submissions参数说明。

表 5-157 Submission

参数	参数类型	描述
isIncrementing	Boolean	作业是否为增量迁移。
job-name	String	作业名称。
counters	**counters** object	作业运行结果统计，当“status”为“SUCCEEDED”时才有此字段，请参见counters数据结构参数说明。
isStopingIncrement	String	是否停止增量迁移。
is-execute-auto	Boolean	是否定时执行作业。
last-update-date	Long	作业最后更新时间。
last-udpate-user	String	最后更新作业状态的用户。
isDeleteJob	Boolean	作业执行完成后是否删除。
creation-user	String	创建用户。
creation-date	Long	创建时间。
external-id	String	作业ID。
progress	Float	作业进度，失败时为“-1”，其它情况为0～100。
submission-id	Integer	作业提交id。
delete_rows	Integer	删除数据行数。
update_rows	Integer	更新数据行数。
write_rows	Integer	写入数据行数。
execute-date	Long	执行时间。

参数	参数类型	描述
status	String	作业状态： • BOOTING：启动中。 • FAILURE_ON_SUBMIT：提交失败。 • RUNNING：运行中。 • SUCCEEDED：成功。 • FAILED：失败。 • UNKNOWN：未知。 • NEVER_EXECUTED：未被执行。
error-details	String	错误详情，当“status”为“FAILED”时才有此字段。
error-summary	String	错误总结，当“status”为“FAILED”时才有此字段。

表 5-158 counters

参数	参数类型	描述
org.apache.sqoop.submission.counter.SqoopCounters	**counter** object	作业运行结果统计，请参见统计结果参数说明。

表 5-159 counter

参数	参数类型	描述
BYTES_WRITTEN	Long	写入的字节数。
TOTAL_FILES	Integer	总文件数。
ROWS_READ	Long	读取的行数。
BYTES_READ	Long	读取的字节数。
ROWS_WRITTEN	Long	写入的行数。
FILES_WRITTEN	Integer	写入的文件数。
FILES_READ	Integer	读取的文件数。
TOTAL_SIZE	Long	总字节数。
FILES_SKIPPED	Integer	跳过的文件数。
ROWS_WRITTEN_SKIPPED	Long	跳过的行数。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/job/jdbc2hive/status
```

响应示例

状态码：200

OK。

```
{
  "submissions" : [ {
    "job-name" : "jdbc2hive",
    "creation-user" : "cdm",
    "creation-date" : "1536905778725",
    "progress" : 1,
    "status" : "BOOTING"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ShowJobStatusSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowJobStatusRequest request = new ShowJobStatusRequest();
        request.withClusterId("{cluster_id}");
        request.withJobName("{job_name}");
        try {
            ShowJobStatusResponse response = client.showJobStatus(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
```

```
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowJobStatusRequest()
        request.cluster_id = "{cluster_id}"
        request.job_name = "{job_name}"
        response = client.show_job_status(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
```

```
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowJobStatusRequest{}
    request.ClusterId = "{cluster_id}"
    request.JobName = "{job_name}"
    response, err := client.ShowJobStatus(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。

错误码

请参见**错误码**。

5.2.9 查询作业执行历史

功能介绍

查询作业执行历史接口。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/clusters/{cluster_id}/cdm/submissions

表 5-160 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。

表 5-161 Query 参数

参数	是否必选	参数类型	描述
jname	是	String	作业名称。

请求参数

表 5-162 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码：200

表 5-163 响应 Body 参数

参数	参数类型	描述
submissions	Array of **Submission** objects	作业运行信息，详见submissions参数说明。
total	Integer	查询该作业总的历史记录数。
page_no	Integer	查询作业记录时，分页数。
page_size	Integer	分页查询，每页返回的记录数。默认值：10。

表 5-164 Submission

参数	参数类型	描述
isIncrementing	Boolean	作业是否为增量迁移。
job-name	String	作业名称。
counters	**counters** object	作业运行结果统计，当“status”为“SUCCEEDED”时才有此字段，请参见counters数据结构参数说明。
isStopingIncrement	String	是否停止增量迁移。
is-execute-auto	Boolean	是否定时执行作业。
last-update-date	Long	作业最后更新时间。
last-udpate-user	String	最后更新作业状态的用户。
isDeleteJob	Boolean	作业执行完成后是否删除。
creation-user	String	创建用户。
creation-date	Long	创建时间。
external-id	String	作业ID。
progress	Float	作业进度，失败时为“-1”，其它情况为0～100。
submission-id	Integer	作业提交id。
delete_rows	Integer	删除数据行数。
update_rows	Integer	更新数据行数。
write_rows	Integer	写入数据行数。
execute-date	Long	执行时间。
status	String	作业状态： • BOOTING：启动中。 • FAILURE_ON_SUBMIT：提交失败。 • RUNNING：运行中。 • SUCCEEDED：成功。 • FAILED：失败。 • UNKNOWN：未知。 • NEVER_EXECUTED：未被执行。
error-details	String	错误详情，当“status”为“FAILED”时才有此字段。
error-summary	String	错误总结，当“status”为“FAILED”时才有此字段。

表 5-165 counters

参数	参数类型	描述
org.apache.sqoop.submission.counter.SqoopCounters	**counter** object	作业运行结果统计，请参见统计结果参数说明。

表 5-166 counter

参数	参数类型	描述
BYTES_WRITTEN	Long	写入的字节数。
TOTAL_FILES	Integer	总文件数。
ROWS_READ	Long	读取的行数。
BYTES_READ	Long	读取的字节数。
ROWS_WRITTEN	Long	写入的行数。
FILES_WRITTEN	Integer	写入的文件数。
FILES_READ	Integer	读取的文件数。
TOTAL_SIZE	Long	总字节数。
FILES_SKIPPED	Integer	跳过的文件数。
ROWS_WRITTEN_SKIPPED	Long	跳过的行数。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/submissions?jname=jdbc2hive
```

响应示例

状态码：200

OK。

```
{
  "submissions" : [ {
    "job-name" : "jdbc2hive",
    "creation-user" : "cdm",
    "creation-date" : "1536905778725",
    "progress" : 1,
    "status" : "BOOTING"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ShowSubmissionsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowSubmissionsRequest request = new ShowSubmissionsRequest();
        request.withClusterId("{cluster_id}");
        try {
            ShowSubmissionsResponse response = client.showSubmissions(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *
```

```
if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowSubmissionsRequest()
        request.cluster_id = "{cluster_id}"
        response = client.show_submissions(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowSubmissionsRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ShowSubmissions(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
```

```
  }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。

错误码

请参见**错误码**。

5.3 连接管理

5.3.1 创建连接

功能介绍

创建连接接口。

调用方法

请参见**如何调用API**。

URI

POST /v1.1/{project_id}/clusters/{cluster_id}/cdm/link

表 5-167 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。

表 5-168 Query 参数

参数	是否必选	参数类型	描述
validate	否	String	为“true”时，此API仅校验参数是否正确，不创建连接。

请求参数

表 5-169 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 5-170 请求 Body 参数

参数	是否必选	参数类型	描述
links	是	Array of **links** objects	连接列表，请参见links数据结构说明。

表 5-171 links

参数	是否必选	参数类型	描述
link-config-values	是	**link-config-values** object	连接参数配置，请参见link-config-values参数说明。
creation-user	否	String	创建连接的用户。
name	是	String	连接名称。
id	否	Integer	连接ID。
creation-date	否	Long	创建连接的时间。

参数	是否必选	参数类型	描述
connector-name	是	String	连接器名称，对应的连接参数如下：generic-jdbc-connector：关系数据库连接。obs-connector：OBS连接。hdfs-connector：HDFS连接。hbase-connector：HBase连接、CloudTable连接。hive-connector：Hive连接。ftp-connector/sftp-connector：FTP/SFTP连接。mongodb-connector：MongoDB连接。redis-connector：Redis/DCS连接。kafka-connector：Kafka连接。dis-connector：DIS连接。elasticsearch-connector：Elasticsearch/云搜索服务连接。dli-connector：DLI连接。http-connector：HTTP/HTTPS连接，该连接暂无连接参数。dms-kafka-connector：DMSKafka连接。
update-date	否	Long	更新连接的时间。
enabled	否	Boolean	是否激活连接，默认为“true”。
update-user	否	String	更新连接的用户。

表 5-172 link-config-values

参数	是否必选	参数类型	描述
configs	是	Array of **configs** objects	连接配置参数数据结构，请参见configs参数说明。
extended-configs	否	**extended-configs** object	扩展配置，请参见extended-configs参数说明。
validators	否	Array of strings	校验器。

表 5-173 configs

参数	是否必选	参数类型	描述
inputs	是	Array of **Input** objects	输入参数列表，列表中的每个参数为“name,value”结构，请参考inputs数据结构参数说明。在“from-config-values”数据结构中，不同的源连接类型有不同的“inputs”参数列表，请参见源端作业参数说明下的章节。在“to-config-values”数据结构中，不同的目的连接类型有不同的“inputs”参数列表，请参见目的端作业参数说明下面的子章节。在“driver-config-values”数据结构中，“inputs”具体参数请参见作业任务参数说明。
name	是	String	配置名称：源端作业的配置名称为“fromJobConfig”。目的端作业的配置名称为“toJobConfig”，连接的配置名称固定为“linkConfig”。
id	否	Integer	配置ID，由系统生成，用户无需填写。
type	否	String	配置类型，由系统生成，用户无需填写。值为LINK或者JOB，如果是连接管理API，则为LINK；如果是作业管理API，则为JOB。

表 5-174 Input

参数	是否必选	参数类型	描述
name	是	String	参数名: • 如果是连接管理API，则以“linkConfig.”开头，对于不同连接类型有不同的参数，具体可参见**连接参数说明**下相应连接的参数说明。 • 如果是作业管理API，对于源端连接参数，则以“fromJobConfig.”开头，具体可参见**源端作业参数说明**下相应的源端参数说明；对于目的端连接参数，则以“toJobConfig.”开头，具体可参见**目的端作业参数说明**下相应的目的端参数说明；对于作业任务参数，请参见**作业任务参数说明**下相应的任务参数说明。
value	是	Object	参数值，参数名对应的值，必须填写为字符串。
type	否	String	值类型，如STRING、INTEGER，由系统设定，用户无需填写。

表 5-175 extended-configs

参数	是否必选	参数类型	描述
name	否	String	名称。
value	否	String	值。

响应参数

状态码：200

表 5-176 响应 Body 参数

参数	参数类型	描述
name	String	连接名称。

参数	参数类型	描述
validation-result	Array of **validationResult** objects	校验结构：如果创建连接失败，返回失败原因，请参见validation-result参数说明。如果创建成功，返回空列表。

表 5-177 validationResult

参数	参数类型	描述
linkConfig	Array of **validationLinkConfig** objects	创建或更新连接校验结果，请参见linkConfig参数说明。

表 5-178 validationLinkConfig

参数	参数类型	描述
message	String	错误描述。
status	String	错误级别，如：ERROR、WARNING。

状态码：400

表 5-179 响应 Body 参数

参数	参数类型	描述
code	String	返回编码。
errCode	String	错误码。
message	String	报错信息。
externalMessage	String	附加信息。

状态码：500

表 5-180 响应 Body 参数

参数	参数类型	描述
message	String	错误描述。
status	String	错误级别，如：ERROR、WARNING。

请求示例

创建一个名为mysql_link的数据连接。

```
POST /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/link
```

```
{
  "links" : [ {
    "link-config-values" : {
      "configs" : [ {
        "inputs" : [ {
          "name" : "linkConfig.databaseType",
          "value" : "MYSQL"
        }, {
          "name" : "linkConfig.host",
          "value" : "100.94.8.163"
        }, {
          "name" : "linkConfig.port",
          "value" : "3306"
        }, {
          "name" : "linkConfig.database",
          "value" : "DB_name"
        }, {
          "name" : "linkConfig.username",
          "value" : "username"
        }, {
          "name" : "linkConfig.password",
          "value" : "DB_password"
        }, {
          "name" : "linkConfig.fetchSize",
          "value" : "100000"
        }, {
          "name" : "linkConfig.usingNative",
          "value" : "false"
        } ],
        "name" : "linkConfig"
      } ]
    },
    "name" : "mysql_link",
    "creation-date" : 1496654788622,
    "connector-name" : "generic-jdbc-connector",
    "update-date" : 1496654788622,
    "enabled" : true
  } ]
}
```

响应示例

状态码：200

OK。

```
{
  "name" : "rdb_link",
  "validation-result" : [ { } ]
}
```

状态码：400

请求错误。

```
{
  "code" : "Cdm.0315",
  "errCode" : "Cdm.0315",
  "message" : "Link name [ftp_link] already exist or created by other user.",
  "externalMessage" : "Link name [ftp_link] already exist or created by other user."
}
```

状态码：500

服务内部错误，具体返回错误码请参考错误码。

```
{
  "validation-result" : [ {
    "linkConfig" : [ {
      "message" : "Can't connect to the database with given credentials: The authentication type 12 is not supported. Check that you have configured the pg_hba.conf file to include the client's IP address or subnet, and that it is using an authentication scheme supported by the driver.",
      "status" : "ERROR"
    } ]
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建一个名为mysql_link的数据连接。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class CreateLinkSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateLinkRequest request = new CreateLinkRequest();
        request.withClusterId("{cluster_id}");
        CdmCreateAndUpdateLinkReq body = new CdmCreateAndUpdateLinkReq();
        List<Input> listConfigsInputs = new ArrayList<>();
        listConfigsInputs.add(
            new Input()
                .withName("linkConfig.databaseType")
                .withValue("MYSQL")
        );
```

```
listConfigsInputs.add(
    new Input()
        .withName("linkConfig.host")
        .withValue("100.94.8.163")
);
listConfigsInputs.add(
    new Input()
        .withName("linkConfig.port")
        .withValue("3306")
);
listConfigsInputs.add(
    new Input()
        .withName("linkConfig.database")
        .withValue("DB_name")
);
listConfigsInputs.add(
    new Input()
        .withName("linkConfig.username")
        .withValue("username")
);
listConfigsInputs.add(
    new Input()
        .withName("linkConfig.password")
        .withValue("DB_password")
);
listConfigsInputs.add(
    new Input()
        .withName("linkConfig.fetchSize")
        .withValue("100000")
);
listConfigsInputs.add(
    new Input()
        .withName("linkConfig.usingNative")
        .withValue("false")
);
List<Configs> listLinkConfigValuesConfigs = new ArrayList<>();
listLinkConfigValuesConfigs.add(
    new Configs()
        .withInputs(listConfigsInputs)
        .withName("linkConfig")
);
LinksLinkconfigvalues linkConfigValuesLinks = new LinksLinkconfigvalues();
linkConfigValuesLinks.withConfigs(listLinkConfigValuesConfigs);
List<Links> listbodyLinks = new ArrayList<>();
listbodyLinks.add(
    new Links()
        .withLinkConfigValues(linkConfigValuesLinks)
        .withName("mysql_link")
        .withCreationDate(1496654788622L)
        .withConnectorName("generic-jdbc-connector")
        .withUpdateDate(1496654788622L)
        .withEnabled(true)
);
body.withLinks(listbodyLinks);
request.withBody(body);
try {
    CreateLinkResponse response = client.createLink(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
```

```
    }
}
```

Python

创建一个名为mysql_link的数据连接。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateLinkRequest()
        request.cluster_id = "{cluster_id}"
        listInputsConfigs = [
            Input(
                name="linkConfig.databaseType",
                value="MYSQL"
            ),
            Input(
                name="linkConfig.host",
                value="100.94.8.163"
            ),
            Input(
                name="linkConfig.port",
                value="3306"
            ),
            Input(
                name="linkConfig.database",
                value="DB_name"
            ),
            Input(
                name="linkConfig.username",
                value="username"
            ),
            Input(
                name="linkConfig.password",
                value="DB_password"
            ),
            Input(
                name="linkConfig.fetchSize",
                value="100000"
            ),
            Input(
                name="linkConfig.usingNative",
                value="false"
            )
        ]
```

```
        listConfigsLinkConfigValues = [
            Configs(
                inputs=listInputsConfigs,
                name="linkConfig"
            )
        ]
        linkConfigValuesLinks = LinksLinkconfigvalues(
            configs=listConfigsLinkConfigValues
        )
        listLinksbody = [
            Links(
                link_config_values=linkConfigValuesLinks,
                name="mysql_link",
                creation_date=1496654788622,
                connector_name="generic-jdbc-connector",
                update_date=1496654788622,
                enabled=True
            )
        ]
        request.body = CdmCreateAndUpdateLinkReq(
            links=listLinksbody
        )
        response = client.create_link(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建一个名为mysql_link的数据连接。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateLinkRequest{}
    request.ClusterId = "{cluster_id}"
    valueInputs:= "MYSQL"
```

```
var valueInputsInterface interface{} = valueInputs
valueInputs1:= "100.94.8.163"
var valueInputsInterface1 interface{} = valueInputs1
valueInputs2:= "3306"
var valueInputsInterface2 interface{} = valueInputs2
valueInputs3:= "DB_name"
var valueInputsInterface3 interface{} = valueInputs3
valueInputs4:= "username"
var valueInputsInterface4 interface{} = valueInputs4
valueInputs5:= "DB_password"
var valueInputsInterface5 interface{} = valueInputs5
valueInputs6:= "100000"
var valueInputsInterface6 interface{} = valueInputs6
valueInputs7:= "false"
var valueInputsInterface7 interface{} = valueInputs7
var listInputsConfigs = []model.Input{
    {
        Name: "linkConfig.databaseType",
        Value: &valueInputsInterface,
    },
    {
        Name: "linkConfig.host",
        Value: &valueInputsInterface1,
    },
    {
        Name: "linkConfig.port",
        Value: &valueInputsInterface2,
    },
    {
        Name: "linkConfig.database",
        Value: &valueInputsInterface3,
    },
    {
        Name: "linkConfig.username",
        Value: &valueInputsInterface4,
    },
    {
        Name: "linkConfig.password",
        Value: &valueInputsInterface5,
    },
    {
        Name: "linkConfig.fetchSize",
        Value: &valueInputsInterface6,
    },
    {
        Name: "linkConfig.usingNative",
        Value: &valueInputsInterface7,
    },
}
var listConfigsLinkConfigValues = []model.Configs{
    {
        Inputs: listInputsConfigs,
        Name: "linkConfig",
    },
}
linkConfigValuesLinks := &model.LinksLinkconfigvalues{
    Configs: listConfigsLinkConfigValues,
}
creationDateLinks:= int64(1496654788622)
updateDateLinks:= int64(1496654788622)
enabledLinks:= true
var listLinksbody = []model.Links{
    {
        LinkConfigValues: linkConfigValuesLinks,
        Name: "mysql_link",
        CreationDate: &creationDateLinks,
        ConnectorName: "generic-jdbc-connector",
        UpdateDate: &updateDateLinks,
        Enabled: &enabledLinks,
```

```
        },
    }
    request.Body = &model.CdmCreateAndUpdateLinkReq{
        Links: listLinksbody,
    }
    response, err := client.CreateLink(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.3.2 查询连接

功能介绍

查询连接接口。

调用方法

请参见**如何调用API**。

URI

GET /v1.1/{project_id}/clusters/{cluster_id}/cdm/link/{link_name}

表 5-181 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。
link_name	是	String	连接名称。

请求参数

表 5-182 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码：200

表 5-183 响应 Body 参数

参数	参数类型	描述
links	Array of **links** objects	连接列表，请参见links数据结构说明。
fromTo-unMapping	String	表/文件迁移不支持哪些数据源迁移到哪些数据源。
batchFromTo-mapping	String	整库迁移支持哪些数据源迁移到哪些数据源。

表 5-184 links

参数	参数类型	描述
link-config-values	**link-config-values** object	连接参数配置，请参见link-config-values参数说明。
creation-user	String	创建连接的用户。
name	String	连接名称。

参数	参数类型	描述
id	Integer	连接ID。
creation-date	Long	创建连接的时间。
connector-name	String	连接器名称，对应的连接参数如下：generic-jdbc-connector：关系数据库连接。obs-connector：OBS连接。hdfs-connector：HDFS连接。hbase-connector：HBase连接、CloudTable连接。hive-connector：Hive连接。ftp-connector/sftp-connector：FTP/SFTP连接。mongodb-connector：MongoDB连接。redis-connector：Redis/DCS连接。kafka-connector：Kafka连接。dis-connector：DIS连接。elasticsearch-connector：Elasticsearch/云搜索服务连接。dli-connector：DLI连接。 http-connector：HTTP/HTTPS连接，该连接暂无连接参数。dms-kafka-connector：DMSKafka连接。
update-date	Long	更新连接的时间。
enabled	Boolean	是否激活连接，默认为“true”。
update-user	String	更新连接的用户。

表 5-185 link-config-values

参数	参数类型	描述
configs	Array of **configs** objects	连接配置参数数据结构，请参见configs参数说明。
extended-configs	**extended-configs** object	扩展配置，请参见extended-configs参数说明。
validators	Array of strings	校验器。

表 5-186 configs

参数	参数类型	描述
inputs	Array of **Input** objects	输入参数列表，列表中的每个参数为“name,value”结构，请参考inputs数据结构参数说明。在“from-config-values”数据结构中，不同的源连接类型有不同的“inputs”参数列表，请参见源端作业参数说明下的章节。在“to-config-values”数据结构中，不同的目的连接类型有不同的“inputs”参数列表，请参见目的端作业参数说明下面的子章节。在“driver-config-values”数据结构中，“inputs”具体参数请参见作业任务参数说明。
name	String	配置名称：源端作业的配置名称为“fromJobConfig”。目的端作业的配置名称为“toJobConfig”，连接的配置名称固定为“linkConfig”。
id	Integer	配置ID，由系统生成，用户无需填写。
type	String	配置类型，由系统生成，用户无需填写。值为LINK或者JOB，如果是连接管理API，则为LINK；如果是作业管理API，则为JOB。

表 5-187 Input

参数	参数类型	描述
name	String	参数名: • 如果是连接管理API，则以“linkConfig.”开头，对于不同连接类型有不同的参数，具体可参见**连接参数说明**下相应连接的参数说明。 • 如果是作业管理API，对于源端连接参数，则以“fromJobConfig.”开头，具体可参见**源端作业参数说明**下相应的源端参数说明；对于目的端连接参数，则以“toJobConfig.”开头，具体可参见**目的端作业参数说明**下相应的目的端参数说明；对于作业任务参数，请参见**作业任务参数说明**下相应的任务参数说明。
value	Object	参数值，参数名对应的值，必须填写为字符串。

参数	参数类型	描述
type	String	值类型，如STRING、INTEGER，由系统设定，用户无需填写。

表 5-188 extended-configs

参数	参数类型	描述
name	String	名称。
value	String	值。

请求示例

```
GET /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/link/sftplink
```

响应示例

状态码：200

OK。

```
{
  "links" : [ {
    "link-config-values" : {
      "configs" : [ {
        "inputs" : [ {
          "name" : "linkConfig.server",
          "type" : "STRING",
          "value" : "100.94.8.163"
        }, {
          "name" : "linkConfig.port",
          "type" : "INTEGER",
          "value" : 22
        }, {
          "name" : "linkConfig.username",
          "type" : "STRING",
          "value" : "root"
        }, {
          "name" : "linkConfig.password",
          "type" : "STRING",
          "value" : "Add password here"
        } ],
        "name" : "linkConfig"
      } ]
    },
    "creation-user" : "cdm",
    "name" : "sftp_link",
    "creation-date" : 1516674482640,
    "connector-name" : "sftp-connector",
    "update-date" : 1516674476022,
    "enabled" : true,
    "update-user" : "cdm"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class ShowLinkSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowLinkRequest request = new ShowLinkRequest();
        request.withClusterId("{cluster_id}");
        request.withLinkName("{link_name}");
        try {
            ShowLinkResponse response = client.showLink(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *
```

```
if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowLinkRequest()
        request.cluster_id = "{cluster_id}"
        request.link_name = "{link_name}"
        response = client.show_link(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowLinkRequest{}
    request.ClusterId = "{cluster_id}"
    request.LinkName = "{link_name}"
    response, err := client.ShowLink(request)
    if err == nil {
```

```
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.3.3 删除连接

功能介绍

删除连接接口。

调用方法

请参见**如何调用API**。

URI

DELETE /v1.1/{project_id}/clusters/{cluster_id}/cdm/link/{link_name}

表 5-189 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。

参数	是否必选	参数类型	描述
cluster_id	是	String	集群ID。
link_name	是	String	需要删除的连接名。

请求参数

表 5-190 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码：500

表 5-191 响应 Body 参数

参数	参数类型	描述
errCode	String	错误码。
externalMessage	String	错误描述。

请求示例

```
DELETE /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/link/jdbclink
```

响应示例

状态码：500

服务内部错误，具体返回错误码请参考错误码。

```
{
  "errCode" : "Cdm.0021",
  "externalMessage" : "Given link name is in use"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;


public class DeleteLinkSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteLinkRequest request = new DeleteLinkRequest();
        request.withClusterId("{cluster_id}");
        request.withLinkName("{link_name}");
        try {
            DeleteLinkResponse response = client.deleteLink(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
```

```
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteLinkRequest()
        request.cluster_id = "{cluster_id}"
        request.link_name = "{link_name}"
        response = client.delete_link(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteLinkRequest{}
    request.ClusterId = "{cluster_id}"
    request.LinkName = "{link_name}"
    response, err := client.DeleteLink(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。
503	服务不可用。

错误码

请参见**错误码**。

5.3.4 修改连接

功能介绍

修改连接接口。

调用方法

请参见**如何调用API**。

URI

PUT /v1.1/{project_id}/clusters/{cluster_id}/cdm/link/{link_name}

表 5-192 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参见**项目ID和账号ID**。
cluster_id	是	String	集群ID。
link_name	是	String	连接名称。

请求参数

表 5-193 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 5-194 请求 Body 参数

参数	是否必选	参数类型	描述
links	是	Array of **links** objects	连接列表，请参见links数据结构说明。

表 5-195 links

参数	是否必选	参数类型	描述
link-config-values	是	**link-config-values** object	连接参数配置，请参见link-config-values参数说明。
creation-user	否	String	创建连接的用户。
name	是	String	连接名称。
id	否	Integer	连接ID。
creation-date	否	Long	创建连接的时间。

参数	是否必选	参数类型	描述
connector-name	是	String	连接器名称，对应的连接参数如下：generic-jdbc-connector：关系数据库连接。obs-connector：OBS连接。hdfs-connector：HDFS连接。hbase-connector：HBase连接、CloudTable连接。hive-connector：Hive连接。ftp-connector/sftp-connector：FTP/SFTP连接。mongodb-connector：MongoDB连接。redis-connector：Redis/DCS连接。kafka-connector：Kafka连接。dis-connector：DIS连接。elasticsearch-connector：Elasticsearch/云搜索服务连接。dli-connector：DLI连接。http-connector：HTTP/HTTPS连接，该连接暂无连接参数。dms-kafka-connector：DMSKafka连接。
update-date	否	Long	更新连接的时间。
enabled	否	Boolean	是否激活连接，默认为“true”。
update-user	否	String	更新连接的用户。

表 5-196 link-config-values

参数	是否必选	参数类型	描述
configs	是	Array of **configs** objects	连接配置参数数据结构，请参见configs参数说明。
extended-configs	否	**extended-configs** object	扩展配置，请参见extended-configs参数说明。
validators	否	Array of strings	校验器。

表 5-197 configs

参数	是否必选	参数类型	描述
inputs	是	Array of **Input** objects	输入参数列表，列表中的每个参数为“name,value”结构，请参考inputs数据结构参数说明。在“from-config-values”数据结构中，不同的源连接类型有不同的“inputs”参数列表，请参见源端作业参数说明下的章节。在“to-config-values”数据结构中，不同的目的连接类型有不同的“inputs”参数列表，请参见目的端作业参数说明下面的子章节。在“driver-config-values”数据结构中，“inputs”具体参数请参见作业任务参数说明。
name	是	String	配置名称：源端作业的配置名称为“fromJobConfig”。目的端作业的配置名称为“toJobConfig”，连接的配置名称固定为“linkConfig”。
id	否	Integer	配置ID，由系统生成，用户无需填写。
type	否	String	配置类型，由系统生成，用户无需填写。值为LINK或者JOB，如果是连接管理API，则为LINK；如果是作业管理API，则为JOB。

表 5-198 Input

参数	是否必选	参数类型	描述
name	是	String	参数名: • 如果是连接管理API，则以“linkConfig.”开头，对于不同连接类型有不同的参数，具体可参见**连接参数说明**下相应连接的参数说明。 • 如果是作业管理API，对于源端连接参数，则以“fromJobConfig.”开头，具体可参见**源端作业参数说明**下相应的源端参数说明；对于目的端连接参数，则以“toJobConfig.”开头，具体可参见**目的端作业参数说明**下相应的目的端参数说明；对于作业任务参数，请参见**作业任务参数说明**下相应的任务参数说明。
value	是	Object	参数值，参数名对应的值，必须填写为字符串。
type	否	String	值类型，如STRING、INTEGER，由系统设定，用户无需填写。

表 5-199 extended-configs

参数	是否必选	参数类型	描述
name	否	String	名称。
value	否	String	值。

响应参数

状态码：200

表 5-200 响应 Body 参数

参数	参数类型	描述
validation-result	Array of **validationResult** objects	校验结构：如果创建连接失败，返回失败原因，请参见validation-result参数说明。如果创建成功，返回空列表。

表 5-201 validationResult

参数	参数类型	描述
linkConfig	Array of **validationLinkConfig** objects	创建或更新连接校验结果，请参见linkConfig参数说明。

表 5-202 validationLinkConfig

参数	参数类型	描述
message	String	错误描述。
status	String	错误级别，如：ERROR、WARNING。

状态码：500

表 5-203 响应 Body 参数

参数	参数类型	描述
message	String	错误描述。
status	String	错误级别，如：ERROR、WARNING。

请求示例

修改一个名为mysql_link的数据连接。

```
PUT /v1.1/1551c7f6c808414d8e9f3c514a170f2e/clusters/6ec9a0a4-76be-4262-8697-e7af1fac7920/cdm/link/rdb_link

{
  "links" : [ {
    "link-config-values" : {
      "configs" : [ {
        "inputs" : [ {
          "name" : "linkConfig.databaseType",
          "value" : "MYSQL"
        }, {
          "name" : "linkConfig.host",
          "value" : "100.94.8.163"
        }, {
          "name" : "linkConfig.port",
          "value" : "3306"
        }, {
          "name" : "linkConfig.database",
          "value" : "DB_name"
        }, {
          "name" : "linkConfig.username",
          "value" : "username"
        }, {
          "name" : "linkConfig.password",
          "value" : "DB_password"
        }, {
```

```
        "name" : "linkConfig.fetchSize",
        "value" : "100000"
      }, {
        "name" : "linkConfig.usingNative",
        "value" : "false"
      } ],
      "name" : "linkConfig"
    } ]
  },
  "name" : "mysql_link",
  "creation-date" : 1496654788622,
  "connector-name" : "generic-jdbc-connector",
  "update-date" : 1496654788622,
  "enabled" : true
 } ]
}
```

响应示例

状态码：200

OK。

```
{
 "validation-result" : [ { } ]
}
```

状态码：500

服务内部错误，具体返回错误码请参考错误码。

```
{
 "validation-result" : [ {
  "linkConfig" : [ {
   "message" : "Can't connect to the database with given credentials: The authentication type 12 is not supported. Check that you have configured the pg_hba.conf file to include the client's IP address or subnet, and that it is using an authentication scheme supported by the driver.",
   "status" : "ERROR"
  } ]
 } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

修改一个名为mysql_link的数据连接。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cdm.v1.region.CdmRegion;
import com.huaweicloud.sdk.cdm.v1.*;
import com.huaweicloud.sdk.cdm.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class UpdateLinkSolution {
```

```
    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CdmClient client = CdmClient.newBuilder()
                .withCredential(auth)
                .withRegion(CdmRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateLinkRequest request = new UpdateLinkRequest();
        request.withClusterId("{cluster_id}");
        request.withLinkName("{link_name}");
        CdmCreateAndUpdateLinkReq body = new CdmCreateAndUpdateLinkReq();
        List<Input> listConfigsInputs = new ArrayList<>();
        listConfigsInputs.add(
            new Input()
                .withName("linkConfig.databaseType")
                .withValue("MYSQL")
        );
        listConfigsInputs.add(
            new Input()
                .withName("linkConfig.host")
                .withValue("100.94.8.163")
        );
        listConfigsInputs.add(
            new Input()
                .withName("linkConfig.port")
                .withValue("3306")
        );
        listConfigsInputs.add(
            new Input()
                .withName("linkConfig.database")
                .withValue("DB_name")
        );
        listConfigsInputs.add(
            new Input()
                .withName("linkConfig.username")
                .withValue("username")
        );
        listConfigsInputs.add(
            new Input()
                .withName("linkConfig.password")
                .withValue("DB_password")
        );
        listConfigsInputs.add(
            new Input()
                .withName("linkConfig.fetchSize")
                .withValue("100000")
        );
        listConfigsInputs.add(
            new Input()
                .withName("linkConfig.usingNative")
                .withValue("false")
        );
        List<Configs> listLinkConfigValuesConfigs = new ArrayList<>();
        listLinkConfigValuesConfigs.add(
            new Configs()
                .withInputs(listConfigsInputs)
                .withName("linkConfig")
```

```
        );
        LinksLinkconfigvalues linkConfigValuesLinks = new LinksLinkconfigvalues();
        linkConfigValuesLinks.withConfigs(listLinkConfigValuesConfigs);
        List<Links> listbodyLinks = new ArrayList<>();
        listbodyLinks.add(
            new Links()
                .withLinkConfigValues(linkConfigValuesLinks)
                .withName("mysql_link")
                .withCreationDate(1496654788622L)
                .withConnectorName("generic-jdbc-connector")
                .withUpdateDate(1496654788622L)
                .withEnabled(true)
        );
        body.withLinks(listbodyLinks);
        request.withBody(body);
        try {
            UpdateLinkResponse response = client.updateLink(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

修改一个名为mysql_link的数据连接。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcdm.v1.region.cdm_region import CdmRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcdm.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CdmClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CdmRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateLinkRequest()
        request.cluster_id = "{cluster_id}"
        request.link_name = "{link_name}"
        listInputsConfigs = [
            Input(
                name="linkConfig.databaseType",
                value="MYSQL"
```

```
            ),
            Input(
                name="linkConfig.host",
                value="100.94.8.163"
            ),
            Input(
                name="linkConfig.port",
                value="3306"
            ),
            Input(
                name="linkConfig.database",
                value="DB_name"
            ),
            Input(
                name="linkConfig.username",
                value="username"
            ),
            Input(
                name="linkConfig.password",
                value="DB_password"
            ),
            Input(
                name="linkConfig.fetchSize",
                value="100000"
            ),
            Input(
                name="linkConfig.usingNative",
                value="false"
            )
        ]
        listConfigsLinkConfigValues = [
            Configs(
                inputs=listInputsConfigs,
                name="linkConfig"
            )
        ]
        linkConfigValuesLinks = LinksLinkconfigvalues(
            configs=listConfigsLinkConfigValues
        )
        listLinksbody = [
            Links(
                link_config_values=linkConfigValuesLinks,
                name="mysql_link",
                creation_date=1496654788622,
                connector_name="generic-jdbc-connector",
                update_date=1496654788622,
                enabled=True
            )
        ]
        request.body = CdmCreateAndUpdateLinkReq(
            links=listLinksbody
        )
        response = client.update_link(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

修改一个名为mysql_link的数据连接。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
```

```
    cdm "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cdm/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cdm.NewCdmClient(
        cdm.CdmClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateLinkRequest{}
    request.ClusterId = "{cluster_id}"
    request.LinkName = "{link_name}"
    valueInputs:= "MYSQL"
    var valueInputsInterface interface{} = valueInputs
    valueInputs1:= "100.94.8.163"
    var valueInputsInterface1 interface{} = valueInputs1
    valueInputs2:= "3306"
    var valueInputsInterface2 interface{} = valueInputs2
    valueInputs3:= "DB_name"
    var valueInputsInterface3 interface{} = valueInputs3
    valueInputs4:= "username"
    var valueInputsInterface4 interface{} = valueInputs4
    valueInputs5:= "DB_password"
    var valueInputsInterface5 interface{} = valueInputs5
    valueInputs6:= "100000"
    var valueInputsInterface6 interface{} = valueInputs6
    valueInputs7:= "false"
    var valueInputsInterface7 interface{} = valueInputs7
    var listInputsConfigs = []model.Input{
        {
            Name: "linkConfig.databaseType",
            Value: &valueInputsInterface,
        },
        {
            Name: "linkConfig.host",
            Value: &valueInputsInterface1,
        },
        {
            Name: "linkConfig.port",
            Value: &valueInputsInterface2,
        },
        {
            Name: "linkConfig.database",
            Value: &valueInputsInterface3,
        },
        {
            Name: "linkConfig.username",
            Value: &valueInputsInterface4,
        },
        {
            Name: "linkConfig.password",
```

```
            Value: &valueInputsInterface5,
        },
        {
            Name: "linkConfig.fetchSize",
            Value: &valueInputsInterface6,
        },
        {
            Name: "linkConfig.usingNative",
            Value: &valueInputsInterface7,
        },
    }
    var listConfigsLinkConfigValues = []model.Configs{
        {
            Inputs: listInputsConfigs,
            Name: "linkConfig",
        },
    }
    linkConfigValuesLinks := &model.LinksLinkconfigvalues{
        Configs: listConfigsLinkConfigValues,
    }
    creationDateLinks:= int64(1496654788622)
    updateDateLinks:= int64(1496654788622)
    enabledLinks:= true
    var listLinksbody = []model.Links{
        {
            LinkConfigValues: linkConfigValuesLinks,
            Name: "mysql_link",
            CreationDate: &creationDateLinks,
            ConnectorName: "generic-jdbc-connector",
            UpdateDate: &updateDateLinks,
            Enabled: &enabledLinks,
        },
    }
    request.Body = &model.CdmCreateAndUpdateLinkReq{
        Links: listLinksbody,
    }
    response, err := client.UpdateLink(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK。
400	请求错误。
401	鉴权失败。
403	没有操作权限。
404	找不到资源。
500	服务内部错误，具体返回错误码请参考错误码。

状态码	描述
503	服务不可用。

错误码

请参见**错误码**。

6 公共数据结构

6.1 连接参数说明

6.1.1 关系数据库连接

介绍

通过JDBC连接，可以对以下关系型数据库抽取、加载数据：

- 云数据库 PostgreSQL
- 云数据库 SQL Server
- PostgreSQL
- Microsoft SQL Server

连接样例

```
{
   "links": [
      {
         "link-config-values": {
            "configs": [
               {
                  "inputs": [
                     {
                        "name": "linkConfig.databaseType",
                        "value": "MYSQL"
                     },
                     {
                        "name": "linkConfig.host",
                        "value": "10.120.205.30"
                     },
                     {
                        "name": "linkConfig.port",
                        "value": "3306"
                     },
                     {
                        "name": "linkConfig.database",
                        "value": "DB_name"
                     },
                     {
                        "name": "linkConfig.username",
```

```
                    "value": "username"
                },
                {
                    "name": "linkConfig.password",
                    "value": "Add password here"
                },
                {
                    "name": "linkConfig.fetchSize",
                    "value": "100000"
                },
                {
                    "name": "linkConfig.commitSize",
                    "value": "10000"
                },
                {
                    "name": "linkConfig.usingNative",
                    "value": "false"
                },
                {
                    "name": "linkConfig.useSSL",
                    "value": "false"
                }
            ],
            "name": "linkConfig"
        }
      ]
    },
    "name": "mysql_link",
    "connector-name": "generic-jdbc-connector"
  }
 ]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.databaseType	是	枚举	数据库类型： • ORACLE • MYSQL • SQLSERVER • DB2 • POSTGRESQL • DWS • DDM • SAP HANA
linkConfig.host	是	String	数据库服务器地址。
linkConfig.port	是	String	数据库服务器的端口号。

参数	是否必选	类型	说明
linkConfig.databaseconfig	否	枚举	创建Oracle连接时才有该参数，选择Oracle数据库连接类型： ● SERVICENAME：通过SERVICE_NAME连接Oracle数据库。 ● SID：通过SID连接Oracle数据库。
linkConfig.sidname	否	String	配置Oracle实例ID，用于实例区分各个数据库。创建Oracle连接，且linkConfig.databaseconfig（数据库连接类型）选择为“SID”时才有该参数。
linkConfig.database	否	String	数据库名称。
linkConfig.username	是	String	用户名。
linkConfig.password	是	String	用户密码。
linkConfig.fetchSize	否	String	每次请求获取的数据行数。
linkConfig.commitSize	否	String	每次请求提交的数据行数。

参数	是否必选	类型	说明
linkConfig.usingNative	否	Boolean	是否使用数据库本地API加速。 创建MySQL连接时，使用本地API加速，可以使用MySQL的LOAD DATA功能加快数据导入，提高导入数据到MySQL数据库的性能。 **说明** REPLACE 和 IGNORE 修饰符用于处理与现有行具有相同唯一键值（PRIMARY KEY或UNIQUE索引值）的新输入的行。 ● 约束冲突处理配置为"replace into"或"local"，使用REPLACE，与现有行中的唯一键值具有相同值的新行将替换现有行。 ● 约束冲突处理配置为"insert into"，默认使用IGNORE，与唯一键值上的现有行重复的新行将被丢弃，任务不会终止。因为Mysql服务机制无法在操作过程中停止文件传输，此情况下，CDM界面显示写入记录与实际更新行数会不一致。 Mysql Local模式详情请参见：https://dev.mysql.com/doc/refman/8.0/en/load-data.html
linkConfig.isRds	否	Boolean	是否支持RDS服务，云数据库默认true，其余默认false
linkConfig.useSSL	否	Boolean	是否使用加密传输，支持对RDS服务启用SSL加密传输，仅在创建dws连接时才有此参数。
linkConfig.jdbcProperties	否	Map	连接属性，指定数据源的JDBC连接器的属性，参考对应数据库的JDBC连接器说明文档进行配置。

参数	是否必选	类型	说明
linkConfig.version	否	枚举	创建Oracle连接时才有该参数，根据您Oracle数据库的版本来选择。 • HIGH_VERSION：当您的Oracle数据库版本高于12.1时，选择该值。 • MED_VERSION：当您的Oracle数据库版本为12.1时，选择该值。 • LOW_VERSION：当您的Oracle数据库版本低于12.1时，选择该值。 当出现“java.sql.SQLException: Protocol violation异常”时，可以尝试更换版本号。
dialect.identifierEnclose	否	String	引用符号，连接引用表名或列名时的分隔符号，参考对应数据库的产品文档进行配置。

6.1.2 OBS 连接

介绍

通过OBS连接，可以对对象存储服务（Object Storage Service，简称OBS）抽取或加载文件，支持CSV、JSON和二进制格式。

连接样例

本示例为连接样例消息体。在实际使用中，AK（accessKey）和SK（securityKey）建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。

```
{
   "links": [
      {
         "link-config-values": {
            "configs": [
               {
                  "inputs": [
                     {
                        "name": "linkConfig.storageType",
                        "value": "OBS"
                     },
                     {
                        "name": "linkConfig.obsBucketType",
                        "value": "PFS"
                     },
                     {
                        "name": "linkConfig.server",
                        "value": "10.121.16.183"
                     },
                     {
                        "name": "linkConfig.port",
```

```
                    "value": "443"
                },
                {
                    "name":"linkConfig.accessKey",
                    "value": "<YOUR AK>"
                },
                {
                    "name":"linkConfig.securityKey",
                    "value":"<YOUR SK>"
                }
            ],
            "name": "linkConfig"
        }
    ]
},
"name": "obs_link",
"connector-name": "obs-connector"
}
]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.storageType	是	String	对象存储的类型。
linkConfig.obsBucketType	否	String	OBS桶类型。 ● 对象桶：OB。 ● 并行文件系统：PFS。 **说明** 桶类型分为对象桶和并行文件系统，对象桶过滤文件会比较慢，建议客户使用并行文件桶。
linkConfig.server	是	String	OBS服务器的终端节点（Endpoint）。
linkConfig.port	是	String	数据传输协议端口，https是443，http是80。
linkConfig.accessKey	是	String	访问标识（AK）。建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。
linkConfig.securityKey	是	String	密钥（SK）。建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。

6.1.3 HDFS 连接

介绍

通过HDFS连接，可以对MRS、FusionInsight HD或开源Hadoop的HDFS抽取、加载文件，支持CSV、Parquet和二进制格式。

连接样例

```
{
   "links": [
      {
         "link-config-values": {
            "configs": [
               {
                  "inputs": [
                     {
                        "name": "linkConfig.hadoopType",
                        "value": "FusionInsight HD"
                     },
                     {
                        "name": "linkConfig.host",
                        "value": "10.120.205.143"
                     },
                     {
                        "name": "linkConfig.casPort",
                        "value": "20009"
                     },
                     {
                        "name": "linkConfig.port",
                        "value": "28443"
                     },
                     {
                        "name": "linkConfig.authType",
                        "value": "KERBEROS"
                     },
                     {
                        "name": "linkConfig.user",
                        "value": "admin"
                     },
                     {
                        "name": "linkConfig.password",
                        "value": "Add password here"
                     },
                     {
                        "name": "linkConfig.runMode",
                        "value": "STANDALONE"
                     }
                  ],
                  "name": "linkConfig"
               }
            ]
         },
         "name": "hdfslink",
         "connector-name": "hdfs-connector"
      }
   ]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.hadoopType	是	枚举	Hadoop类型： • MRS：表示连接MRS的HDFS。 • FusionInsight HD：表示连接FusionInsight HD的HDFS。 • Apache Hadoop：表示连接开源Apache Hadoop的HDFS。

参数	是否必选	类型	说明
linkConfig.uri	否	String	连接Apache Hadoop时的Namenode URI地址，格式为“ip:port”。
linkConfig.host	否	String	连接MRS或FusionInsight HD时，需要配置Manager平台的IP地址。
linkConfig.port	否	String	连接FusionInsight HD时，需要配置Manager平台的端口。
linkConfig.casPort	否	String	连接FusionInsight HD时，需要配置与FusionInsight HD对接的CAS Server的端口。
linkConfig.user	否	String	登录Manager平台的用户名，使用集群配置时不用配置
linkConfig.password	否	String	登录Manager平台的密码，使用集群配置时不用配置
linkConfig.authType	否	枚举	认证类型，分为以下两种： ● Simple：非安全模式选择Simple鉴权。 ● Kerberos：安全模式选择Kerberos鉴权。
linkConfig.principal	否	String	Kerveros认证所需的Principal，您也可以联系管理员获取此账号。使用集群配置前需在集群配置管理中配置此参数。
linkConfig.keytab	否	FileContent	Kerveros认证所需的keytab文件的本地绝对路径，您也可以联系管理员获取此文件。使用集群配置前需在集群配置管理中配置此参数。

参数	是否必选	类型	说明
linkConfig.runMode	否	枚举	选择HDFS连接的运行模式： ● EMBEDDED：连接实例与CDM运行在一起，该模式性能较好。 ● STANDALONE：连接实例运行在独立进程。如果CDM需要对接多个Hadoop数据源（MRS、Hadoop或CloudTable），并且既有KERBEROS认证模式又有SIMPLE认证模式，只能使用STANDALONE模式。 选择STANDALONE模式时，CDM支持在多个MRS集群的HDFS之间迁移数据。 若在一个CDM中同时连接两个及以上开启Kerberos认证且realm相同的集群，只能使用EMBEDDED运行模式连接其中一个集群，其余需使用STANDALONE。
linkConfig.properties	否	Map	属性配置，可以添加客户端的配置属性，所添加的每个属性需配置属性名称和值

6.1.4 HBase 连接

介绍

通过HBase连接，可以对MRS、FusionInsight HD、Apache Hadoop的HBase抽取、加载数据。

连接样例

```
{
  "links": [
    {
      "link-config-values": {
        "configs": [
          {
            "inputs": [
              {
                "name": "linkConfig.hbaseType",
                "value": "MRS"
              },
              {
                "name": "linkConfig.host",
                "value": "192.168.0.34"
              },
              {
                "name": "linkConfig.user",
```

```
            "value": "zephyr"
          },
          {
            "name": "linkConfig.password",
            "value": "Add password here."
          },
          {
            "name": "linkConfig.authType",
            "value": "KERBEROS"
          },
          {
            "name": "linkConfig.serviceType",
            "value": "HDFS"
          },
          {
            "name": "linkConfig.hBaseVersion",
            "value": "HBASE_2_X"
          },
          {
            "name": "linkConfig.runMode",
            "value": "EMBEDDED"
          }
        ],
        "name": "linkConfig"
      }
    ],
    "extended-configs": {
      "name": "linkConfig.extendedFields",
      "value":
"eyL1c2VDbHVzdGVyQ29uZmlnIjoiZmFsc2UiJCLjbHVzdGVyQ29uZmlnUHLpbmNpcGFsIjoiemVwaHlyIn0="
    }
  },
  "name": "mrs_hbase_dlf",
  "connector-name": "hbase-connector"
 }
]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.hbaseType	是	枚举	HBase类型： ● CloudTable：表示连接CloudTable服务。 ● MRS：表示连接MRS的HBase。 ● FusionInsight HD：表示连接FusionInsight HD的HBase。 ● Apache Hadoop：表示连接开源Apache Hadoop的HBase。
linkConfig.uri	否	String	连接Apache Hadoop时的Namenode URI地址，格式为“ip:port”。
llinkConfig.host	否	String	连接MRS或FusionInsight HD时，需要配置Manager平台的IP地址。
linkConfig.port	否	String	连接FusionInsight HD时，需要配置Manager平台的端口。

参数	是否必选	类型	说明
linkConfig.casPort	否	String	连接FusionInsight HD时，需要配置与FusionInsight HD对接的CAS Server的端口。
linkConfig.hBaseVersion	是	枚举	HBase版本： ● HBASE_1_X ● HBASE_2_X
linkConfig.user	否	String	登录Manager平台的用户名，使用集群配置时不用配置
linkConfig.password	否	String	登录Manager平台的密码，使用集群配置时不用配置
linkConfig.authType	否	枚举	认证类型，分为以下两种： ● Simple：非安全模式选择Simple鉴权。 ● Kerberos：安全模式选择Kerberos鉴权。
linkConfig.principal	否	String	Kerveros认证所需的Principal，您也可以联系管理员获取此账号。
linkConfig.keytab	否	FileContent	Kerveros认证所需的keytab文件的本地绝对路径，您也可以联系管理员获取此文件。
linkConfig.serviceType	否	String	服务类型。目前支持HDFS和HBase。

参数	是否必选	类型	说明
linkConfig.runMode	否	枚举	“HBase_2_X”版本支持该参数。支持以下模式： ● EMBEDDED：连接实例与CDM运行在一起，该模式性能较好。 ● STANDALONE：连接实例运行在独立进程。如果CDM需要对接多个Hadoop数据源（MRS、Hadoop或CloudTable），并且既有KERBEROS认证模式又有SIMPLE认证模式，只能使用STANDALONE模式。选择STANDALONE模式时，CDM支持在多个MRS集群的HDFS之间迁移数据。 若在一个CDM中同时连接两个及以上开启Kerberos认证且realm相同的集群，只能使用EMBEDDED运行模式连接其中一个集群，其余需使用STANDALONE。
linkConfig.properties	否	Map	属性配置，可以添加客户端的配置属性，所添加的每个属性需配置属性名称和值

6.1.5 CloudTable 连接

介绍

通过CloudTable连接，可以对CloudTable服务抽取、加载数据。

连接样例

本示例为连接样例消息体。在实际使用中，AK（accessKey）和SK（securityKey）建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。

```
{
  "links": [
    {
      "link-config-values": {
        "configs": [
          {
            "inputs": [
              {
                "name": "linkConfig.hbaseType",
                "value": "CloudTable"
              },
              {
                "name": "linkConfig.zookeeperQuorum",
                "value": "cloudtable-pass-zk2-bae54VGN.cloudtable.com:2181,cloudtable-pass-zk1-
Fu828so2.cloudtable.com:2181"
```

```
          },
          {
            "name": "linkConfig.iamAuth",
            "value": "true"
          },
          {
            "name": "linkConfig.cloudtableUser",
            "value": "zane"
          },
          {
            "name": "linkConfig.accessKey",
            "value": "<YOUR AK>"
          },
          {
            "name": "linkConfig.securityKey",
            "value": "<YOUR SK>"
          },
          {
            "name": "linkConfig.runMode",
            "value": "EMBEDDED"
          }
        ],
        "name": "linkConfig"
      }
    ]
  },
  "name": "cloudtablelink",
  "connector-name": "hbase-connector"
 }
]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.hbaseType	是	枚举	HBase类型： ● CloudTable：表示连接CloudTable服务。 ● MRS：表示连接MRS。 ● FusionInsight HD：表示连接FusionInsight HD。 ● Apache Hadoop：表示连接开源Apache Hadoop。
linkConfig.zookeeperQuorum	是	String	连接“CloudTable”时必选，表示CloudTable的ZooKeeper链接地址。
linkConfig.iamAuth	是	Boolean	当选择IAM统一身份认证时，需要输入用户名、AK和SK。

参数	是否必选	类型	说明
linkConfig.runMode	是	枚举	选择HBase连接的运行模式： ● EMBEDDED：连接实例与CDM运行在一起，该模式性能较好。 ● STANDALONE：连接实例运行在独立进程。如果CDM需要对接多个Hadoop数据源（MRS、Hadoop或CloudTable），并且既有KERBEROS认证模式又有SIMPLE认证模式，只能使用STANDALONE模式。
linkConfig.cloudtableUser	是	String	登录CloudTable集群的用户名。
linkConfig.accessKey	是	String	登录CloudTable集群的访问标识。建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。
linkConfig.securityKey	是	String	登录CloudTable集群的密钥。建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。

6.1.6 Hive 连接

介绍

通过Hive连接，可以对MRS的Hive数据源进行抽取、加载数据。

连接样例

本示例为连接样例消息体。在实际使用中，AK（accessKey）和SK（securityKey）建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。

```
{
  "links": [
    {
      "link-config-values": {
        "configs": [
          {
            "inputs": [
              {
                "name": "linkConfig.host",
                "value": "10.120.205.230"
              },
              {
                "name": "linkConfig.authType",
                "value": "KERBEROS"
              },
              {
                "name": "linkConfig.user",
                "value": "cdm"
              },
              {
```

```
            "name": "linkConfig.password",
            "value": "Add password here"
          }
        ],
        "name": "linkConfig"
      }
    ]
  },
  "name": "hive_link",
  "connector-name": "hive-connector"
 }
]
}
```

连接参数

参数	是否必选	类型	说明
llinkConfig.host	是	String	MRS Manager的IP地址。
linkConfig.authType	是	枚举	访问MRS的认证类型： ● SIMPLE：非安全模式选择Simple鉴权。 ● KERBEROS：安全模式选择Kerberos鉴权。
linkConfig.principal	否	String	Kerveros认证所需的Principal，您也可以联系管理员获取此账号。使用集群配置前需在集群配置管理中配置此参数。
linkConfig.keytab	否	FileContent	Kerveros认证所需的keytab文件的本地绝对路径，您也可以联系管理员获取此文件。使用集群配置前需在集群配置管理中配置此参数。
linkConfig.hiveVersion	是	枚举	Hive版本： ● HIVE_1_X ● HIVE_3_X
linkConfig.user	否	String	登录Manager平台的用户名，使用集群配置时不用配置
linkConfig.password	否	String	登录Manager平台的密码，使用集群配置时不用配置
linkConfig.uri	否	String	连接Apache Hadoop时的Namenode URI地址，格式为“ip:port”
linkConfig.hiveMsUris	否	String	连接Apache Hadoop时的Hive元数据地址，参考hive.metastore.uris配置项。例如：thrift://host-192-168-1-212:9083

参数	是否必选	类型	说明
linkConfig.obsSupport	是	Boolean	需服务端支持OBS存储。在创建Hive表时，您可以指定将表存储在OBS中。
linkConfig.runMode	是	枚举	“HIVE_3_X”版本支持该参数。支持以下模式： ● EMBEDDED：连接实例与CDM运行在一起，该模式性能较好。 ● STANDALONE：连接实例运行在独立进程。如果CDM需要对接多个Hadoop数据源（MRS、Hadoop或CloudTable），并且既有KERBEROS认证模式又有SIMPLE认证模式，只能使用STANDALONE模式。 说明：STANDALONE模式主要是用来解决版本冲突问题的运行模式。当同一种数据连接的源端或者目的端连接器的版本不一致时，存在jar包冲突的情况，这时需要将源端或目的端放在STANDALONE进程里，防止冲突导致迁移失败。
linkConfig.accessKey	否	String	访问标识（AK）。obs支持选是时需配置此参数。建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。
linkConfig.securityKey	否	String	密钥（SK）。obs支持选是时需配置此参数。建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。
linkConfig.properties	否	Map	属性配置，可以添加客户端的配置属性，所添加的每个属性需配置属性名称和值

6.1.7 FTP/SFTP 连接

介绍

通过FTP或SFTP连接，可以对FTP或SFTP服务器抽取或加载文件，支持CSV、二进制和JSON格式。

连接样例

```
{
  "links": [
```

```
    {
      "link-config-values": {
        "configs": [
          {
            "inputs": [
              {
                "name": "linkConfig.server",
                "value": "10.120.85.167"
              },
              {
                "name": "linkConfig.port",
                "value": "22"
              },
              {
                "name": "linkConfig.username",
                "value": "username"
              },
              {
                "name": "linkConfig.password",
                "value": "Add password here"
              }
            ],
            "name": "linkConfig"
          }
        ]
      },
      "name": "sftp_link",
      "connector-name": "sftp-connector"
    }
  ]
}
```

连接参数

FTP和SFTP的连接参数相同。

参数	是否必选	类型	说明
linkConfig.server	是	String	FTP或SFTP服务器的地址。
linkConfig.port	是	String	FTP或SFTP服务器端口号。
linkConfig.username	是	String	登录FTP或SFTP服务器的用户名。
linkConfig.password	是	String	登录用户的密码。

6.1.8 MongoDB 连接

介绍

通过MongoDB连接，可以对MongoDB服务器抽取、加载数据。

连接样例

```
{
  "links": [
    {
      "link-config-values": {
        "configs": [
```

```
            {
               "inputs": [
                  {
                     "name": "linkConfig.serverList",
                     "value": "10.120.84.149:27017"
                  },
                  {
                     "name": "linkConfig.database",
                     "value": "DB_name"
                  },
                  {
                     "name": "linkConfig.userName",
                     "value": "username"
                  },
                  {
                     "name": "linkConfig.password",
                     "value": "Add password here"
                  }
               ],
               "name": "linkConfig"
            }
         ]
      },
      "name": "mongo_link",
      "connector-name": "mongodb-connector"
   }
  ]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.serverList	是	String	服务器地址列表，格式如：“host1:port1;host2:port2”。
linkConfig.database	是	String	MongoDB的数据库名称。
linkConfig.userName	是	String	连接MongoDB服务器的用户名。
linkConfig.password	是	String	连接MongoDB服务器的密码。

6.1.9 Redis 连接

介绍

通过Redis连接，可以对Redis服务器抽取或加载数据。

连接样例

```
{
  "links": [
    {
      "link-config-values": {
        "configs": [
          {
            "inputs": [
```

```
                    {
                        "name": "linkConfig.deploymentMode",
                        "value": "Cluster"
                    },
                    {
                        "name": "linkConfig.serverlist",
                        "value": "10.120.84.149:7300"
                    },
                    {
                        "name": "linkConfig.password",
                        "value": "Add password here"
                    },
                    {
                        "name": "linkConfig.dbIndex",
                        "value": "0"
                    }
                ],
                "name": "linkConfig"
            }
        ]
    },
    "name": "redis_link",
    "connector-name": "redis-connector"
  }
 ]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.deploymentMode	是	枚举	Redis部署方式： ● Single：表示单机部署。 ● Cluster：表示集群部署。
linkConfig.serverlist	是	String	服务器地址列表，格式如：“host1:port1;host2:port2”。
linkConfig.password	是	String	连接Redis服务器的密码。
linkConfig.dbIndex	是	String	Redis数据库索引。

6.1.10 Kafka 连接

介绍

通过Kafka连接器可以与开源的Kafka数据源建立连接，并按照用户指定配置将Kafka中的数据迁移到其它数据源。目前仅支持从Kafka导出数据。

连接样例

```
{
"links": [
  {
    "link-config-values": {
      "configs": [
        {
          "inputs": [
```

```
          {
            "name": "linkConfig.hadoopType",
            "value": "MRS"
          },
          {
            "name": "linkConfig.host",
            "value": "192.168.1.147"
          },
          {
            "name": "linkConfig.user",
            "value": "liuhuan1"
          },
          {
            "name": "linkConfig.password",
            "value": "Add password here."
          },
          {
            "name": "linkConfig.authType",
            "value": "KERBEROS"
          }
        ],
        "name": "linkConfig"
      }
    ],
    "extended-configs": {
      "name": "linkConfig.extendedFields",
      "value": "e30="
    }
  },
  "name": "mrs_kafka_link",
  "connector-name": "kafka-connector"
 }
]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.hadoopType	是	枚举	Hadoop类型： ● MRS：表示连接MRS的Kafka。 ● Apache Kafka：表示连接Apache Kafka的Kafka。
linkConfig.brokerList	是	String	Apache Kafka 连接需配置此参数。Kafka broker列表，格式如：“host1:port1,host2:port2”。
llinkConfig.host	是	String	MRS Manager的浮动IP地址，可以单击输入框后的“选择”来选定已创建的MRS集群，CDM会自动填充下面的鉴权参数。
linkConfig.user	是	String	登录MRS Manager平台的用户名。
linkConfig.password	是	String	登录MRS Manager平台的密码。

参数	是否必选	类型	说明
linkConfig.authType	是	枚举	认证类型，分为以下两种： • Simple：非安全模式选择Simple鉴权。 • Kerberos：安全模式选择Kerberos鉴权。
linkConfig.properties	否	Map	属性配置，可以添加客户端的配置属性，所添加的每个属性需配置属性名称和值

6.1.11 DIS 连接

介绍

通过DIS连接可以与DIS建立连接，并按照用户指定配置将DIS中的数据迁移到其他数据源。

连接样例

```
{
   "links": [
      {
         "link-config-values": {
            "configs": [
               {
                  "inputs": [
                     {
                        "name": "linkConfig.region",
                        "value": "Region"
                     },
                     {
                        "name": "linkConfig.endpoint",
                        "value": "https://dis.cn-north-1.myhuaweiclouds.com"
                     },
                     {
                        "name": "linkConfig.ak",
                        "value": "RSO6TTEZMJ6TTFBBAACE"
                     },
                     {
                        "name": "linkConfig.sk",
                        "value": "Add password here"
                     },
                     {
                        "name": "linkConfig.projectId",
                        "value": "11d4d5af17c84660bc90b6631327d7c7"
                     }
                  ],
                  "name": "linkConfig"
               }
            ]
         },
         "name": "dis_link",
         "connector-name": "dis-connector"
      }
   ]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.region	是	String	DIS服务所属地域。
linkConfig.endpoint	是	String	待连接DIS的URL，URL格式为https://Endpoint。
linkConfig.ak	是	String	DIS服务端的AK。
linkConfig.sk	是	String	DIS服务端的SK。
linkConfig.projectId	是	String	项目ID，获取方法请参见**项目ID和账号ID**。

6.1.12 Elasticsearch/云搜索服务(CSS)连接

介绍

通过Elasticsearch连接，可以对Elasticsearch服务器或云搜索服务抽取、加载数据。

连接样例

```
{
  "links": [
    {
      "link-config-values": {
        "configs": [
          {
            "inputs": [
              {
                "name": "linkConfig.host",
                "value": "192.168.0.50:9200;192.168.0.62:9200"
              },
              {
                "name": "linkConfig.safemode",
                "value": "true"
              },
              {
                "name": "linkConfig.user",
                "value": "admin"
              },
              {
                "name": "linkConfig.password",
                "value": "Add password here."
              },
              {
                "name": "linkConfig.linkType",
                "value": "CSS"
              }
            ],
            "name": "linkConfig"
          }
        ],
        "extended-configs": {
          "name": "linkConfig.extendedFields",
          "value": "eyLodHRwc0FjY2VzcyI6InRydWUifQ=="
        }
      },
      "name": "css-cdm-autotest-nodel",
```

```
    "connector-name": "elasticsearch-connector"
  }
 ]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.host	是	String	配置为Elasticsearch服务器的IP地址或域名，包括端口号，格式为“ip:port”，多个地址之间使用分号（；）分隔，例如：192.168.0.1:9200;192.168.0.2:9200。
linkConfig.safemode	否	Boolean	当选择安全模式认证时，需要输入用户名、密码和选择是否https访问。
linkConfig.user	否	String	对于支持用户名密码鉴权的Elasticsearch，需要在创建连接时配置用户名和密码。
linkConfig.password	否	String	登录Elasticsearch的密码。
linkConfig.linkType	是	String	连接类型，用于区分连接的是Elasticsearch或云搜索服务。

6.1.13 DLI 连接

介绍

通过DLI连接，可以导入数据到数据湖探索（DLI）服务，CDM暂不支持从DLI服务导出数据。

连接样例

```
{
  "links": [
    {
      "link-config-values": {
        "configs": [
          {
            "inputs": [
              {
                "name": "linkConfig.ak",
                "value": "GRC2WR0IDC6NGROYLWU2"
              },
              {
                "name": "linkConfig.sk",
                "value": "Add password here"
              },
              {
                "name": "linkConfig.region",
                "value": "cn-north-1"
              },
```

```
                {
                    "name": "linkConfig.projectId",
                    "value": "c48475ce8e174a7a9f775706a3d5ebe2"
                }
            ],
            "name": "linkConfig"
        }
      ]
    },
    "name": "dli",
    "connector-name": "dli-connector"
  }
 ]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.ak	是	String	登录DLI数据库的AK。
linkConfig.sk	是	String	登录DLI数据库的SK。
linkConfig.region	是	String	DLI服务所在的区域。
linkConfig.projectId	是	String	DLI服务的项目ID。

6.1.14 CloudTable OpenTSDB 连接

介绍

通过OpenTSDB连接，可以从CloudTable OpenTSDB导入导出数据。

连接样例

```
{
 "links": [
  {
   "link-config-values": {
    "configs": [
     {
      "inputs": [
       {
        "name": "linkConfig.openTSDBQuorum",
        "value": "opentsdb-sp8afz7bgbps5ur.cloudtable.com:4242"
       },
       {
        "name": "linkConfig.securityMode",
        "value": "UNSAFE"
       }
      ],
      "name": "linkConfig"
     }
    ]
   },
   "name": "opentsdb",
   "connector-name": "opentsdb-connector"
  }
```

```
  ]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.openTSDBQuorum	是	String	OpenTSDB的ZooKeeper链接地址。
linkConfig.securityMode	是	String	选择安全或非安全模式。 选择安全模式时，需要输入项目ID、用户名、AK/SK。
linkConfig.user	否	String	访问CloudTable服务的用户名。
linkConfig.ak	否	String	访问CloudTable服务的AK。
linkConfig.sk	否	String	访问CloudTable服务的SK。
linkConfig.projectId	否	String	CloudTable服务的项目ID。

6.1.15 DMS Kafka 连接

介绍

通过DMS Kafka连接，可以连接DMS Kafka普通队列或者专享版Kafka。目前仅支持从DMS Kafka导出数据到云搜索服务。

连接样例

```
{
  "links": [
    {
      "link-config-values": {
        "configs": [
          {
            "inputs": [
              {
                "name": "linkConfig.kafkaType",
                "value": "Platinum"
              },
              {
                "name": "linkConfig.brokerList",
                "value": "100.85.121.112:9094,100.85.220.134:9094,100.85.127.232:9094"
              },
              {
                "name": "linkConfig.isPlatinumInstance",
                "value": "false"
              }
            ],
            "name": "linkConfig"
          }
        ],
        "extended-configs": {
          "name": "linkConfig.extendedFields",
```

```
        "value": "e30="
      }
    },
    "name": "dms_kafka",
    "connector-name": "dms-kafka-connector"
  }
 ]
}
```

连接参数

参数	是否必选	类型	说明
linkConfig.kafkaType	是	枚举	选择DMS Kafka版本,目前只有专享版。 ● Basic：指DMS Kafka普通队列。 ● Platinum：指DMS Kafka专享版。
linkConfig.brokerList	是	String	DMS Endpoint格式为“host1:port1,host2:port2”。
linkConfig.isPlatinumInstance	是	Boolean	选择是否打开客户端连接Kafka专享版实例时SSL认证的开关。 开启Kafka SASL_SSL，则数据加密传输，安全性更高，但性能会下降。
linkConfig.user	否	String	开启Kafka SASL_SSL时显示该参数，表示连接DMS Kafka的用户名。
linkConfig.password	否	String	开启Kafka SASL_SSL时显示该参数，表示连接DMS Kafka的密码。

6.2 源端作业参数说明

6.2.1 源端为关系数据库

JSON 样例

```
"from-config-values": {
    "configs": [
     {
      "inputs": [
       {
        "name": "fromJobConfig.useSql",
        "value": "false"
       },
       {
```

```
        "name": "fromJobConfig.schemaName",
        "value": "rf_database"
      },
      {
        "name": "fromJobConfig.tableName",
        "value": "rf_from"
      },
      {
        "name": "fromJobConfig.columnList",
        "value": "AA&BB"
      },
      {
        "name": "fromJobConfig.incrMigration",
        "value": "false"
      }
    ],
    "name": "fromJobConfig"
  }
 ]
}
```

参数说明

参数	是否必选	类型	说明
fromJobConfig.useSql	是	Boolean	导出关系型数据库的数据时，可以选择是否使用自定义SQL语句导出。
fromJobConfig.sql	否	String	可以在这里输入自定义的SQL语句，CDM将根据该语句导出数据。
fromJobConfig.schemaName	是	String	数据库模式或表空间，例如：“public”。 **说明** 该参数支持配置通配符（*），实现导出以某一前缀开头或者以某一后缀结尾的所有数据库。例如： ● **SCHEMA***表示导出所有以“SCHEMA”开头的数据库。 ● ***SCHEMA**表示导出所有以“SCHEMA”结尾的数据库。 ● ***SCHEMA***表示数据库名称中只要有“SCHEMA”字符串，就全部导出。

参数	是否必选	类型	说明
fromJobConfig.tableName	是	String	表名，例如：“TBL_EXAMPLE”。 **说明** 表名支持配置通配符（*），实现导出以某一前缀开头或者以某一后缀结尾的所有表（要求表中的字段个数和类型都一样）。例如： • **table***表示导出所有以“table”开头的表。 • ***table**表示导出所有以“table”结尾的表。 • ***table***表示表名中只要有“table”字符串，就全部导出。
fromJobConfig.whereClause	否	String	指定抽取的Where子句，不指定则抽取整表，例如：“age > 18 and age <= 60”。
fromJobConfig.columnList	否	String	需要抽取的字段列表，字段名之间使用“&”分割，例如：“id&gid&name”。
fromJobConfig.partitionColumn	否	String	抽取分区字段，依据此字段将作业分割为多个任务并发执行，例如：“id”。
fromJobConfig.usePartition	否	Boolean	从Oracle导出数据时，支持从分区表的各个分区并行抽取数据。启用该功能时，可以通过下面的“fromJobConfig.partitionList”参数指定具体的Oracle表分区，该功能不支持非分区表。
fromJobConfig.partitionList	否	String	输入需要迁移数据的Oracle表分区，多个分区以&分隔，不填则迁移所有分区。

6.2.2 源端为对象存储

JSON 样例

```
"from-config-values": {
      "configs": [
        {
          "inputs": [
            {
              "name": "fromJobConfig.bucketName",
              "value": "cdm-est"
            },
            {
              "name": "fromJobConfig.inputDirectory",
              "value": "/obsfrom/varchar.txt"
            },
```

```
        {
          "name": "fromJobConfig.inputFormat",
          "value": "CSV_FILE"
        },
        {
          "name": "fromJobConfig.columnList",
          "value": "1&2&3"
        },
        {
          "name": "fromJobConfig.fieldSeparator",
          "value": ","
        },
        {
          "name": "fromJobConfig.quoteChar",
          "value": "false"
        },
        {
          "name": "fromJobConfig.regexSeparator",
          "value": "false"
        },
        {
          "name": "fromJobConfig.firstRowAsHeader",
          "value": "false"
        },
        {
          "name": "fromJobConfig.encodeType",
          "value": "UTF-8"
        },
        {
          "name": "fromJobConfig.fromCompression",
          "value": "NONE"
        },
        {
          "name": "fromJobConfig.splitType",
          "value": "FILE"
        }
      ],
      "name": "fromJobConfig"
    }
  ]
}
```

参数说明

参数	是否必选	类型	说明
fromJobConfig.bucketName	是	String	对象存储的桶名。
fromJobConfig.inputDirectory	是	String	抽取文件的路径。支持输入多个文件路径（最多50个），默认以“|”分隔，也可以自定义文件分隔符，例如：“FROM/example.csv|FROM/b.txt”。

参数	是否必选	类型	说明
fromJobConfig.inputFormat	是	枚举	传输数据时所用的文件格式，目前支持以下文件格式： ● CSV_FILE：CSV格式，用于迁移文件到数据表的场景。 ● JSON_FILE：JSON格式，一般都是用于迁移文件到数据表的场景。 ● BINARY_FILE：二进制格式，不解析文件内容直接传输，不要求文件格式必须为二进制。适用于文件到文件的原样复制。 当选择“BINARY_FILE”时，目的端也必须为文件系统。
fromJobConfig.lineSeparator	否	String	文件中的换行符，默认自动识别"\\n"、"\\r"或"\\r\\n"。手动配置特殊字符，如空格回车需使用URL编码后的值。或通过编辑作业json方式配置，无需URL编码。
fromJobConfig.columnList	否	String	需要抽取的列号，列号之间使用“&”分割，并由小到大排序，例如：“1&3&5”。
fromJobConfig.regexSeparator	否	Boolean	是否使用正则表达式分割字段，当文件格式为“CSV_FILE”时此参数有效。
fromJobConfig.regex	否	String	正则表达式，当选择使用正则表达式分割字段时，此参数有效。
fromJobConfig.fieldSeparator	否	String	字段分隔符，当文件格式为“CSV_FILE”时此参数有效，默认值为：“,”。
fromJobConfig.quoteChar	否	Boolean	是否使用包围符，选择“true”时，包围符内的字段分隔符会被视为字符串值的一部分，目前CDM默认的包围符为："。
fromJobConfig.firstRowAsHeader	否	Boolean	是否默认首行为标题行，当文件格式为“CSV_FILE”时此参数有效。在迁移CSV文件到表时，CDM默认是全部写入，当该参数选择“true”时，CDM会将CSV文件的第一行数据作为标题行，不写入目的端的表。

参数	是否必选	类型	说明
fromJobConfig.fromCompression	否	枚举	压缩格式，当文件格式为“CSV_FILE”或“JSON”时此参数有效。选择对应压缩格式的源文件： ● NONE：表示传输所有格式的文件。 ● GZIP：表示只传输GZIP格式的文件。 ● ZIP：表示只传输ZIP格式的文件。
fromJobConfig.jsonReferenceNode	否	String	记录节点，当文件格式为“JSON_FILE”时此参数有效。对该JSON节点下的数据进行解析，如果该节点对应的数据为JSON数组，那么系统会以同一模式从该数组中提取数据。多层嵌套的JSON节点以字符“.”分割，例如：“data.list”。
fromJobConfig.encodeType	否	String	编码类型，例如：“UTF_8”或“GBK”。
fromJobConfig.useMarkerFile	否	Boolean	选择是否开启作业标识文件的功能。当源端路径下存在启动作业的标识文件时才启动作业，否则会挂起等待一段时间，等待时长在下方“fromJobConfig.waitTime”（等待时间）参数中配置。
fromJobConfig.markerFile	否	String	启动作业的标识文件名。指定文件后，只有在源端路径下存在该文件的情况下才会运行任务，不指定时默认不启用该功能，例如：“ok.txt”。
fromJobConfig.waitTime	否	String	选择开启作业标识文件的功能时，如果源路径下不存在启动作业的标识文件，作业挂机等待的时长，当超时后任务会失败。 等待时间设置为0时，当源端路径下不存在标识文件，任务会立即失败。 单位：秒。

参数	是否必选	类型	说明
fromJobConfig.filterType	否	枚举	选择过滤器类型： ● WILDCARD：输入通配符作为过滤文件，满足过滤条件的路径或文件会被传输。 ● TIME：时间过滤器，当文件的修改时间晚于输入的时间时，该文件才会被传输。
fromJobConfig.pathFilter	否	String	路径过滤器，过滤类型为通配符时配置，用于过滤文件目录，例如：“*input”。
fromJobConfig.fileFilter	否	String	文件过滤器，过滤类型为通配符时配置，用于过滤目录下的文件，支持配置多个文件，中间使用“,”分隔，例如：“*.csv,*.txt”。
fromJobConfig.startTime	否	String	“过滤类型”选择“时间过滤器”时，可以指定一个时间值，当文件的修改时间大于等于该时间才会被传输，输入的时间格式需为“yyyy-MM-dd HH:mm:ss”。 该参数支持配置为时间宏变量，例如**${timestamp(dateformat(yyyy-MM-dd HH:mm:ss,-90,DAY))}**表示：只迁移最近90天内的文件。
fromJobConfig.endTime	否	String	“过滤类型”选择“时间过滤器”时，可以指定一个时间值，当文件的修改时间小于该时间才会被传输，输入的时间格式需为“yyyy-MM-dd HH:mm:ss”。 该参数支持配置为时间宏变量，例如**${timestamp(dateformat(yyyy-MM-dd HH:mm:ss))}**表示：只迁移修改时间为当前时间以前的文件。
fromJobConfig.fileSeparator	否	String	“fromJobConfig.inputDirectory”（抽取文件的路径）参数中如果输入的是多个文件路径，CDM使用这里配置的文件分隔符来区分各个文件，默认为“|”。

参数	是否必选	类型	说明
fromJobConfig.md5FileSuffix	否	String	校验CDM抽取的文件，是否与源文件一致。

6.2.3 源端为 HDFS

JSON 样例

```
"from-config-values": {
      "configs": [
       {
        "inputs": [
         {
          "name": "fromJobConfig.inputDirectory",
          "value": "/hdfsfrom/from_hdfs_est.csv"
         },
         {
          "name": "fromJobConfig.inputFormat",
          "value": "CSV_FILE"
         },
         {
          "name": "fromJobConfig.columnList",
          "value": "1"
         },
         {
          "name": "fromJobConfig.fieldSeparator",
          "value": ","
         },
         {
          "name": "fromJobConfig.quoteChar",
          "value": "false"
         },
         {
          "name": "fromJobConfig.regexSeparator",
          "value": "false"
         },
         {
          "name": "fromJobConfig.firstRowAsHeader",
          "value": "false"
         },
         {
          "name": "fromJobConfig.encodeType",
          "value": "UTF-8"
         },
         {
          "name": "fromJobConfig.fromCompression",
          "value": "NONE"
         },
         {
          "name": "fromJobConfig.compressedFileSuffix",
          "value": "*"
         },
         {
          "name": "fromJobConfig.splitType",
          "value": "FILE"
         },
         {
          "name": "fromJobConfig.useMarkerFile",
          "value": "false"
         },
         {
          "name": "fromJobConfig.fileSeparator",
          "value": "|"
```

```
        },
        {
          "name": "fromJobConfig.filterType",
          "value": "NONE"
        }
      ],
      "name": "fromJobConfig"
    }
  ]
}
```

参数说明

- HDFS作业参数说明

参数	是否必选	类型	说明
fromJobConfig.inputDirectory	是	String	待抽取数据的路径，例如：“/data_dir”。
fromJobConfig.inputFormat	是	枚举	传输数据时所用的文件格式，目前支持以下文件格式： ● CSV_FILE：CSV格式。 ● PARQUET_FILE：PARQUET格式。 ● BINARY_FILE：二进制格式。 当选择“BINARY_FILE”时，目的端也必须为文件系统。
fromJobConfig.columnList	否	String	需要抽取的列号，列号之间使用“&”分割，并由小到大排序，例如：“1&3&5”。
fromJobConfig.lineSeparator	否	String	文件中的换行符，默认自动识别"\\n"、"\\r"或"\\r\\n"。手动配置特殊字符，如空格回车需使用URL编码后的值。或通过编辑作业json方式配置，无需URL编码。
fromJobConfig.fieldSeparator	否	String	字段分隔符，当文件格式为“CSV_FILE”时此参数有效，默认值为：“,”。
fromJobConfig.quoteChar	否	Boolean	是否使用包围符，选择“true”时，包围符内的字段分隔符会被视为字符串值的一部分，目前CDM默认的包围符为："。
fromJobConfig.regexSeparator	否	Boolean	是否使用正则表达式分割字段，当文件格式为“CSV_FILE”时此参数有效。

参数	是否必选	类型	说明
fromJobConfig.encodeType	否	String	编码类型，例如：“UTF_8”或“GBK”。
fromJobConfig.firstRowAsHeader	否	Boolean	是否默认首行为标题行，当文件格式为“CSV_FILE”时此参数有效。在迁移CSV文件到表时，CDM默认是全部写入，当该参数选择“true”时，CDM会将CSV文件的第一行数据作为标题行，不写入目的端的表。
fromJobConfig.fromCompression	否	枚举	压缩格式，表示选择只传输对应压缩格式的源文件。“NONE”表示传输所有格式的文件。
fromJobConfig.compressedFileSuffix	否	String	需要解压缩的文件后缀名。当一批文件中以该值为后缀时，才会执行解压缩操作，否则保持原样传输。当输入*或为空时，所有文件都会被解压。
fromJobConfig.splitType	否	枚举	指定任务分片方式，选择按文件或文件大小进行分割。HDFS上的文件，如果在HDFS上已经分片，则HDFS每个分片视为一个文件。 ● FILE：按文件数量进行分片。例如有10个文件，并在任务参数中指定“throttlingConfig.numExtractors”（抽取并发数）为“5”，则每个分片2个文件。 ● SIZE：按文件大小分割。注意这里不会将文件做切分来实现均衡。例如：有10个文件，9个10M，1个200M，在并发任务数中指定“throttlingConfig.numExtractors”（抽取并发数）为“2”，则会分两个分片，一个处理9个10M的文件，一个处理1个200M的文件。

参数	是否必选	类型	说明
fromJobConfig.useMarkerFile	否	Boolean	选择是否开启作业标识文件的功能。当源端路径下存在启动作业的标识文件时才启动作业，否则会挂起等待一段时间，等待时长在下方“fromJobConfig.waitTime”（等待时间）参数中配置。
fromJobConfig.markerFile	否	String	启动作业的标识文件名。指定文件后，只有在源端路径下存在该文件的情况下才会运行任务，不指定时默认不启用该功能，例如：“ok.txt”。
fromJobConfig.fileSeparator	否	String	“fromJobConfig.inputDirectory”（抽取文件的路径）参数中如果输入的是多个文件路径，CDM使用这里配置的文件分隔符来区分各个文件，默认为“|”。
fromJobConfig.filterType	否	枚举	选择过滤器类型： ● WILDCARD：输入通配符作为过滤文件，满足过滤条件的路径或文件会被传输。 ● TIME：时间过滤器，当文件的修改时间晚于输入的时间时，该文件才会被传输。
fromJobConfig.pathFilter	否	String	路径过滤器，过滤类型为通配符时配置，用于过滤文件目录，例如：“*input”。
fromJobConfig.fileFilter	否	String	文件过滤器，过滤类型为通配符时配置，用于过滤目录下的文件，支持配置多个文件，中间使用“,”分隔，例如：“*.csv,*.txt”。

参数	是否必选	类型	说明
fromJobConfig.startTime	否	String	“过滤类型”选择“时间过滤器”时，可以指定一个时间值，当文件的修改时间大于等于该时间才会被传输，输入的时间格式需为“yyyy-MM-dd HH:mm:ss”。 该参数支持配置为时间宏变量，例如**${timestamp(dateformat(yyyy-MM-dd HH:mm:ss,-90,DAY))}**表示：只迁移最近90天内的文件。
fromJobConfig.endTime	否	String	“过滤类型”选择“时间过滤器”时，可以指定一个时间值，当文件的修改时间小于该时间才会被传输，输入的时间格式需为“yyyy-MM-dd HH:mm:ss”。 该参数支持配置为时间宏变量，例如**${timestamp(dateformat(yyyy-MM-dd HH:mm:ss))}**表示：只迁移修改时间为当前时间以前的文件。
fromJobConfig.createSnapshot	否	Boolean	如果配置为“true”，CDM读取HDFS系统上的文件时，会先对待迁移的源目录创建快照（不允许对单个文件创建快照），然后CDM迁移快照中的数据。 需要HDFS系统的管理员权限才可以创建快照，CDM作业完成后，快照会被删除。
fromJobConfig.formats	否	数据结构	时间格式，当“fromJobConfig.inputFormat”（文件格式）为“CSV_FILE”（CSV格式），并且文件中有时间类型字段时，才需要输入，具体说明请参见**fromJobConfig.formats参数说明**。

参数	是否必选	类型	说明
fromJobConfig.decryption	否	枚举	“fromJobConfig.inputFormat”（文件格式）选择为“BINARY_FILE”（二进制格式）时才有该参数，选择是否对已加密的文件解密后再导出，以及解密方式： ● NONE：不解密，直接导出文件。 ● AES-256-GCM：使用AES-256-GCM（NoPadding）算法解密后再导出文件。
fromJobConfig.dek	否	String	数据解密密钥，密钥由长度64的十六进制数组成，且必须与加密时配置的“toJobConfig.dek”（导入时配置的数据加密密钥）一致。如果不一致系统不会报异常，只是解密出来的数据会错误。
fromJobConfig.iv	否	String	解密需要的初始化向量，初始化向量由长度32的十六进制数组成，且必须与加密时配置的“toJobConfig.iv”（导入时配置的初始化向量）一致。如果不一致系统不会报异常，只是解密出来的数据会错误。

- fromJobConfig.formats参数说明

参数	是否必选	类型	说明
name	是	String	列号，例如：“1”。
value	是	String	时间格式，例如：“yyyy-MM-dd”。

6.2.4 源端为 Hive

JSON 样例

```
"from-config-values": {
      "configs": [
       {
         "inputs": [
           {
             "name": "fromJobConfig.hive",
             "value": "hive"
           },
           {
```

```
            "name": "fromJobConfig.database",
            "value": "rf_database"
          },
          {
            "name": "fromJobConfig.table",
            "value": "rf_from"
          },
          {
            "name": "fromJobConfig.columnList",
            "value": "tiny&small&int&integer&bigint&float&double&timestamp&char&varchar&text"
          }
        ],
        "name": "fromJobConfig"
      }
    ]
  }
```

参数说明

参数	是否必选	类型	说明
fromJobConfig.hive	否	String	待抽取数据的数据源，作业源端为Hive时，这里为“hive”。
fromJobConfig.database	否	String	待抽取数据的数据库，例如“default”。
fromJobConfig.table	是	String	待抽取数据的表名，例如“cdm”。
fromJobConfig.columnList	否	String	需要抽取的列号，列号之间使用“&”分割，并由小到大排序，例如：“1&3&5”。

6.2.5 源端为 HBase/CloudTable

JSON 样例

```
"from-config-values": {
    "configs": [
      {
        "inputs": [
          {
            "name": "fromJobConfig.table",
            "value": "rf_from"
          },
          {
            "name": "fromJobConfig.columnFamilies",
            "value": "rowkey&f"
          },
          {
            "name": "fromJobConfig.columns",
            "value": "rowkey:rowkey&f:_small"
          },
          {
            "name": "fromJobConfig.formats",
            "value": {
              "f:_date": "yyyy-MM-dd",
              "f:_timestamp": "yyyy-MM-dd HH:mm:ss"
            }
          }
        ],
```

```
      "name": "fromJobConfig"
    }
  ]
}
```

参数说明

- HBase/CloudTable作业参数说明

参数	是否必选	类型	说明
fromJobConfig.table	是	String	需要抽取数据的表名，例如“cdm”。
fromJobConfig.columnFamilies	否	String	抽取数据所属的列族。
fromJobConfig.columns	否	String	需要抽取的列，列号之间使用“&”分割，列族与列之间用“:”分隔，例如：“cf1:c1&cf2:c2”。
fromJobConfig.isSplit	否	Boolean	选择是否拆分Rowkey，例如“true”。
fromJobConfig.delimiter	否	String	用于切分Rowkey的分隔符，若不设置则不切分，例如“|”。
fromJobConfig.startTime	否	String	时间区间左边界（包含该值），格式为“yyyy-MM-dd hh:mm:ss”。 表示只抽取该时间及以后的数据。
fromJobConfig.endTime	否	String	时间区间右边界（不包含该值），格式为“yyyy-MM-dd hh:mm:ss”。 表示只抽取该时间以前的数据。
fromJobConfig.formats	否	数据结构	时间格式，请参见**fromJobConfig.formats参数说明**。

- fromJobConfig.formats参数说明

参数	是否必选	类型	说明
name	是	String	列号，例如：“1”。
value	是	String	时间格式，例如：“yyyy-MM-dd”。

6.2.6 源端为 FTP/SFTP

JSON 样例

```
"from-config-values": {
      "configs": [
       {
        "inputs": [
         {
          "name": "fromJobConfig.inputDirectory",
          "value": "/sftpfrom/from_sftp.csv"
         },
         {
          "name": "fromJobConfig.inputFormat",
          "value": "CSV_FILE"
         },
         {
          "name": "fromJobConfig.columnList",
          "value": "1&2&3&4&5&6&7&8&9&10&11&12"
         },
         {
          "name": "fromJobConfig.fieldSeparator",
          "value": ","
         },
         {
          "name": "fromJobConfig.regexSeparator",
          "value": "false"
         },
         {
          "name": "fromJobConfig.firstRowAsHeader",
          "value": "false"
         },
         {
          "name": "fromJobConfig.encodeType",
          "value": "UTF-8"
         },
         {
          "name": "fromJobConfig.fromCompression",
          "value": "NONE"
         },
         {
          "name": "fromJobConfig.splitType",
          "value": "FILE"
         }
        ],
        "name": "fromJobConfig"
       }
      ]
     }
```

参数说明

FTP、SFTP的源端作业参数相同，如表6-1所示。

表 6-1 文件类源端作业参数

参数	是否必选	类型	说明
fromJobConfig.inputDirectory	是	String	抽取文件的路径。支持输入多个文件路径（最多50个），默认以“|”分隔，也可以自定义文件分隔符，例如：“FROM/example.csv|FROM/b.txt”。

参数	是否必选	类型	说明
fromJobConfig.inputFormat	是	枚举	传输数据时所用的文件格式，目前支持以下文件格式： ● CSV_FILE：CSV格式，用于迁移文件到数据表的场景。 ● JSON_FILE：JSON格式，一般都是用于迁移文件到数据表的场景。 ● BINARY_FILE：二进制格式，不解析文件内容直接传输，不要求文件格式必须为二进制。适用于文件到文件的原样复制。 当选择“BINARY_FILE”时，目的端也必须为文件系统。
fromJobConfig.lineSeparator	否	String	文件中的换行符，默认自动识别"\\n"、"\\r"或"\\r\\n"。手动配置特殊字符，如空格回车需使用URL编码后的值。或通过编辑作业json方式配置，无需URL编码。
fromJobConfig.columnList	否	String	需要抽取的列号，列号之间使用“&”分割，并由小到大排序，例如：“1&3&5”。
fromJobConfig.fieldSeparator	否	String	字段分隔符，当文件格式为“CSV_FILE”时此参数有效，默认值为：“,”。
fromJobConfig.quoteChar	否	Boolean	是否使用包围符，选择“true”时，包围符内的字段分隔符会被视为字符串值的一部分，目前CDM默认的包围符为："。
fromJobConfig.regexSeparator	否	Boolean	是否使用正则表达式分割字段，当文件格式为“CSV_FILE”时此参数有效。
fromJobConfig.regex	否	String	正则表达式，当选择使用正则表达式分割字段时，此参数有效。
fromJobConfig.firstRowAsHeader	否	Boolean	是否默认首行为标题行，当文件格式为“CSV_FILE”时此参数有效。在迁移CSV文件到表时，CDM默认是全部写入，当该参数选择“true”时，CDM会将CSV文件的第一行数据作为标题行，不写入目的端的表。

参数	是否必选	类型	说明
fromJobConfig.fromCompression	否	枚举	压缩格式，当文件格式为“CSV_FILE”或“JSON”时此参数有效。选择对应压缩格式的源文件： ● NONE：表示传输所有格式的文件。 ● GZIP：表示只传输GZIP格式的文件。 ● ZIP：表示只传输ZIP格式的文件。
fromJobConfig.splitType	否	枚举	指定任务分片方式，选择按文件或文件大小进行分割。 ● FILE：按文件数量进行分片。例如有10个文件，并在任务参数中指定“throttlingConfig.numExtractors”（抽取并发数）为“5”，则每个分片2个文件。 ● SIZE：按文件大小分割。注意这里不会将文件做切分来实现均衡。例如：有10个文件，9个10M，1个200M，在并发任务数中指定“throttlingConfig.numExtractors”（抽取并发数）为“2”，则会分两个分片，一个处理9个10M的文件，一个处理1个200M的文件。
fromJobConfig.jsonReferenceNode	否	String	记录节点，当文件格式为“JSON_FILE”时此参数有效。对该JSON节点下的数据进行解析，如果该节点对应的数据为JSON数组，那么系统会以同一模式从该数组中提取数据。多层嵌套的JSON节点以字符“.”分割，例如：“data.list”。
fromJobConfig.encodeType	否	String	编码类型，例如：“UTF_8”或“GBK”。
fromJobConfig.useMarkerFile	否	Boolean	选择是否开启作业标识文件的功能。当源端路径下存在启动作业的标识文件时才启动作业，否则会挂起等待一段时间，等待时长在下方“fromJobConfig.waitTime”（等待时间）参数中配置。

参数	是否必选	类型	说明
fromJobConfig.markerFile	否	String	启动作业的标识文件名。指定文件后，只有在源端路径下存在该文件的情况下才会运行任务，不指定时默认不启用该功能，例如：“ok.txt”。
fromJobConfig.waitTime	否	String	选择开启作业标识文件的功能时，如果源路径下不存在启动作业的标识文件，作业挂机等待的时长，当超时后任务会失败。 等待时间设置为0时，当源端路径下不存在标识文件，任务会立即失败。 单位：秒。
fromJobConfig.filterType	否	枚举	选择过滤器类型： ● WILDCARD：输入通配符作为过滤文件，满足过滤条件的路径或文件会被传输。 ● TIME：时间过滤器，当文件的修改时间晚于输入的时间时，该文件才会被传输。
fromJobConfig.pathFilter	否	String	路径过滤器，过滤类型为通配符时配置，用于过滤文件目录，例如：“*input”。
fromJobConfig.fileFilter	否	String	文件过滤器，过滤类型为通配符时配置，用于过滤目录下的文件，支持配置多个文件，中间使用“,”分隔，例如：“*.csv,*.txt”。
fromJobConfig.startTime	否	String	“过滤类型”选择“时间过滤器”时，可以指定一个时间值，当文件的修改时间大于等于该时间才会被传输，输入的时间格式需为“yyyy-MM-dd HH:mm:ss”。 该参数支持配置为时间宏变量，例如**${timestamp(dateformat(yyyy-MM-dd HH:mm:ss,-90,DAY))}**表示：只迁移最近90天内的文件。

参数	是否必选	类型	说明
fromJobConfig.endTime	否	String	“过滤类型”选择“时间过滤器”时，可以指定一个时间值，当文件的修改时间小于该时间才会被传输，输入的时间格式需为“yyyy-MM-dd HH:mm:ss”。 该参数支持配置为时间宏变量，例如**${timestamp(dateformat(yyyy-MM-dd HH:mm:ss))}**表示：只迁移修改时间为当前时间以前的文件。
fromJobConfig.fileSeparator	否	String	“fromJobConfig.inputDirectory”（抽取文件的路径）参数中如果输入的是多个文件路径，CDM使用这里配置的文件分隔符来区分各个文件，默认为“|”。
fromJobConfig.md5FileSuffix	否	String	校验CDM抽取的文件，是否与源文件一致。

6.2.7 源端为 HTTP/HTTPS

JSON 样例

```
"from-config-values": {
      "configs": [
        {
          "inputs": [
            {
              "name": "fromJobConfig.inputDirectory",
              "value": "http://10.114.196.186:8080/httpfrom/symbol.txt"
            },
            {
              "name": "fromJobConfig.inputFormat",
              "value": "BINARY_FILE"
            },
            {
              "name": "fromJobConfig.fromCompression",
              "value": "TARGZ"
            },
            {
              "name": "fromJobConfig.compressedFileSuffix",
              "value": "*"
            },
            {
              "name": "fromJobConfig.fileSeparator",
              "value": "|"
            }
          ],
          "name": "fromJobConfig"
        }
      ]
    }
```

参数说明

参数	是否必选	类型	说明
fromJobConfig.inputDirectory	是	String	待抽取文件的URL。 用于读取一个公网HTTP/HTTPS URL的文件，包括第三方对象存储的公共读取场景和网盘场景。
fromJobConfig.inputFormat	是	枚举	传输数据时所用的文件格式，目前只支持二进制格式。
fromJobConfig.fromCompression	否	枚举	选择对应压缩格式的源文件进行迁移： ● NONE：表示传输所有格式的文件。 ● GZIP：表示只传输GZIP格式的文件。 ● ZIP：表示只传输ZIP格式的文件。 ● TAR.GZ：表示只传输TAR.GZ格式的文件。
fromJobConfig.compressedFileSuffix	否	String	需要解压缩的文件后缀名。当一批文件中以该值为后缀时，才会执行解压缩操作，否则保持原样传输。当输入*或为空时，所有文件都会被解压。
fromJobConfig.fileSeparator	否	String	传输多个文件时，CDM使用这里配置的文件分隔符来区分各个文件，默认为|。
fromJobConfig.useQuery	否	Boolean	● 该参数设置为“true”时，上传到OBS的对象使用的对象名，为去掉query参数后的字符。 ● 该参数设置为“false”时，上传到OBS的对象使用的对象名，包含query参数。
fromJobConfig.md5FileSuffix	否	String	校验CDM抽取的文件，是否与源文件一致。

6.2.8 源端为 MongoDB/DDS

JSON 样例

```
"from-config-values": {
      "configs": [
        {
          "inputs": [
```

```
        {
          "name": "fromJobConfig.database",
          "value": "cdm"
        },
        {
          "name": "fromJobConfig.collectionName",
          "value": "rf_from"
        },
        {
          "name": "fromJobConfig.columnList",
          "value": "TINYTEST&SMALLTEST&INTTEST&INTEGERTEST&BIGINTTEST&FLOATTEST"
        },
        {
          "name": "fromJobConfig.isBatchMigration",
          "value": "false"
        },
        {
          "name": "fromJobConfig.filters",
          "value": "{'last_name': 'Smith'}"
        }
      ],
      "name": "fromJobConfig"
    }
  ]
}
```

参数说明

参数	是否必选	类型	说明
fromJobConfig.database	是	String	MongoDB/DDS的数据库名称。
fromJobConfig.collectionName	是	String	MongoDB/DDS的集合名称。
fromJobConfig.columnList	否	String	需要抽取的字段列表，字段名之间使用“&”分割，例如：“id&gid&name”。
fromJobConfig.isBatchMigration	否	Boolean	是否为整库迁移。
fromJobConfig.filters	否	String	创建用于匹配文档的筛选条件，CDM只迁移符合条件的数据。例如： 1. 按表达式对象筛选：例如{'last_name': 'Smith'}，表示查找所有“last_name”属性值为“Smith”的文档。 2. 按参数选项筛选：例如{ x : "john" }, { z : 1 }，表示查找x=john的所有z字段。 3. 按条件筛选：例如{ "field" : { $gt: 5 } }，表示查找field字段中大于5的值。

6.2.9 源端为 Redis

JSON 样例

```
"from-config-values": {
      "configs": [
       {
        "inputs": [
         {
          "name": "fromJobConfig.isBatchMigration",
          "value": "false"
         },
         {
          "name": "fromJobConfig.keyPrefix",
          "value": "rf_string_from"
         },
         {
          "name": "fromJobConfig.keySeparator",
          "value": ":"
         },
         {
          "name": "fromJobConfig.valueStoreType",
          "value": "STRING"
         },
         {
          "name": "fromJobConfig.valueSeparator",
          "value": ","
         },
         {
          "name": "fromJobConfig.columnList",
          "value": "1&2&3&4&5&6&7&8&9&10&11&12"
         }
        ],
        "name": "fromJobConfig"
       }
      ]
     }
```

参数说明

- Redis源端作业参数说明

参数	是否必选	类型	说明
fromJobConfig.isBatchMigration	否	Boolean	是否为整库迁移。
fromJobConfig.keyPrefix	是	String	key前缀，对应关系表的表名。 Redis和关系表的映射：用关系表的“表名+分隔符”来对应Redis的Key；关系表的一行数据对应Redis的Value。
fromJobConfig.keySeparator	是	String	key分隔符，一般用来分割关系表和主键。

参数	是否必选	类型	说明
fromJobConfig.valueStoreType	是	String	关系表行数据在Redis中的存储方式分为“string”和“hash”两种存储方式。 • STRING：表示用字符串通过分隔符来表示一行数据的各列，可以有效节省存储空间。 • HASH：表示一行数据通过“列名：列值”的方式存储在hash表中。
fromJobConfig.valueSeparator	否	String	值分隔符号，当“valueStoreType”为“STRING”时此参数有效，默认值为：“\tab”。
fromJobConfig.columnList	否	String	需要抽取的字段列表，字段名之间使用“&”分割，例如：“id&gid&name”。
fromJobConfig.formats	否	数据结构	时间格式，请参见**fromJobConfig.formats参数说明**。

- fromJobConfig.formats参数说明

参数	是否必选	类型	说明
name	是	String	列号，例如：“1”。
value	是	String	时间格式，例如：“yyyy-MM-dd”。

6.2.10 源端为 DIS

JSON 样例

```
"from-config-values": {
        "configs": [
          {
            "inputs": [
              {
                "name": "fromJobConfig.streamName",
                "value": "cdm"
              },
              {
                "name": "fromJobConfig.disConsumerStrategy",
                "value": "FROM_LAST_STOP"
              },
              {
                "name": "fromJobConfig.isPermanency",
                "value": "true"
              },
              {
```

```
                "name": "fromJobConfig.maxPollRecords",
                "value": "100"
              },
              {
                "name": "fromJobConfig.shardId",
                "value": "0"
              },
              {
                "name": "fromJobConfig.dataFormat",
                "value": "BINARY"
              },
              {
                "name": "fromJobConfig.separator",
                "value": ","
              }
            ],
            "name": "fromJobConfig"
          }
        ]
    }
```

参数说明

参数	是否必选	类型	说明
fromJobConfig.streamName	是	String	DIS的通道名。
fromJobConfig.disConsumerStrategy	是	枚举	设置从DIS拉取数据时的初始偏移量： ● LATEST：最大偏移量，即最新的数据。 ● FROM_LAST_STOP：从上次停止处继续拉取。 ● EARLIEST：最小偏移量，即最早的数据。
fromJobConfig.isPermanency	是	Boolean	是否永久运行。
fromJobConfig.maxPollRecords	否	String	每次向DIS请求数据限制最大请求记录数。
fromJobConfig.shardId	是	String	DIS分区ID，该参数支持输入多个分区ID，使用“,”分隔。
fromJobConfig.dataFormat	是	枚举	解析数据时使用的格式： ● BINARY：适用于文件迁移场景，不解析数据内容原样传输。 ● CSV：以CSV格式解析源数据。
fromJobConfig.separator	否	String	字段分隔符。
fromJobConfig.appName	否	String	用户数据消费程序的唯一标识符。

6.2.11 源端为 Kafka

JSON 样例

```
"from-config-values": {
        "configs": [
          {
            "inputs": [
              {
              "name": "fromJobConfig.topicsList",
              "value": "est1,est2"
              },
              {
              "name": "fromJobConfig.kafkaConsumerStrategy",
              "value": "EARLIEST"
              },
              {
              "name": "fromJobConfig.isPermanency",
              "value": "true"
              }
            ],
            "name": "fromJobConfig"
          }
        ]
    }
```

参数说明

参数	是否必选	类型	说明
fromJobConfig.topicsList	是	String	Kafka topic列表，可以为多个topic，以“,”作为分隔符。
fromJobConfig.kafkaConsumerStrategy	是	枚举	从Kafka拉取数据时的初始偏移量设置： ● LATEST：最大偏移量，即最新的数据。 ● EARLIEST：最小偏移量，即最老的数据。
fromJobConfig.isPermanency	是	Boolean	是否永久运行。
fromJobConfig.groupId	否	String	用户指定消费组ID。 如果是从DMS Kafka导出数据，专享版请任意输入，标准版请输入有效的消费组ID。
fromJobConfig.dataFormat	是	枚举	解析数据时使用的格式： ● BINARY：适用于文件迁移场景，不解析数据内容原样传输。 ● CSV：以CSV格式解析源数据。

参数	是否必选	类型	说明
fromJobConfig.maxPollRecords	否	String	每次向Kafka请求数据限制最大请求记录数。
fromJobConfig.maxPollInterval	否	String	每次poll之间的最大时间间隔。
fromJobConfig.separator	否	String	字段分隔符。

6.2.12 源端为 Elasticsearch/云搜索服务

JSON 样例

```
"from-config-values": {
            "configs": [
               {
                  "inputs": [
                     {
                        "name": "fromJobConfig.index",
                        "value": "cdm"
                     },
                     {
                        "name": "fromJobConfig.type",
                        "value": "es"
                     },
                     {
                        "name": "fromJobConfig.columnList",
                        "value": "a1:numeric&s1:string"
                     },
                     {
                        "name": "fromJobConfig.splitNestedField",
                        "value": "true"
                     },
                     {
                        "name": "fromJobConfig.queryString",
                        "value": "last_name:Smith"
                     }
                  ],
                  "name": "fromJobConfig"
               }
            ]
         }
```

参数说明

参数	是否必选	类型	说明
fromJobConfig.index	是	String	抽取数据的索引，类似关系数据库中的数据库名称。
fromJobConfig.type	是	String	抽取数据的类型，类似关系数据库中的表名。
fromJobConfig.columnList	否	String	需要抽取的字段列表，字段名之间使用“&”分隔，例如：“id&gid&name”。

参数	是否必选	类型	说明
fromJobConfig.splitNestedField	否	Boolean	选择是否将nested字段的json内容拆分，例如：将“a:{ b:{ c:1, d:{ e:2, f:3 } } }” 拆成三个字段“a.b.c”、“a.b.d.e”、“a.b.d.f”。
fromJobConfig.queryString	否	String	使用Elasticsearch的查询字符串（query string）对源数据进行过滤，CDM只迁移满足过滤条件的数据。

6.3 目的端作业参数说明

6.3.1 目的端为关系数据库

JSON 样例

```
"to-config-values": {
      "configs": [
       {
        "inputs": [
         {
          "name": "toJobConfig.schemaName",
          "value": "cdm"
         },
         {
          "name": "toJobConfig.tablePreparation",
          "value": "DROP_AND_CREATE"
         },
         {
          "name": "toJobConfig.tableName",
          "value": "rf_to"
         },
         {
          "name": "toJobConfig.columnList",
          "value": "id&gid&name"
         },
         {
          "name": "toJobConfig.isCompress",
          "value": "false"
         },
         {
          "name": "toJobConfig.orientation",
          "value": "ROW"
         },
         {
          "name": "toJobConfig.useStageTable",
          "value": "false"
         },
         {
          "name": "toJobConfig.shouldClearTable",
          "value": "false"
         },
         {
          "name": "toJobConfig.extendCharLength",
          "value": "false"
         }
        ],
```

```
      "name": "toJobConfig"
    }
  ]
}
```

参数说明

参数	是否必选	类型	说明
toJobConfig.schemaName	是	String	数据库模式或表空间。
toJobConfig.tablePreparation	是	枚举	只有当源端和目的端都为关系数据库时，才有该参数。表示写入表数据时，用户选择的操作： ● DO_NOTHING：不自动建表。 ● CREATE_WHEN_NOT_EXIST：当目的端的数据库没有“tableName”参数中指定的表时，CDM会自动创建该表。 ● DROP_AND_CREATE：先删除“tableName”参数中指定的表，然后再重新创建该表。
toJobConfig.tableName	是	String	写入数据的表名。
toJobConfig.columnList	否	String	需要加载的字段列表，字段名之间使用“&”分割，例如：“id&gid&name”。
toJobConfig.beforeImportType	否	枚举	导入数据前，选择是否清除目的表的数据： ● none：写入数据前不清除目标表中数据，数据追加写入。 ● shouldClearTable：写入数据前会清除目标表中数据。 ● whereClause：选择根据where条件删除时，需要配置“toJobConfig.whereClause”参数，CDM根据条件选择性删除目标表的数据。
toJobConfig.whereClause	否	String	where条件，导入前根据where条件删除目的表的数据。

参数	是否必选	类型	说明
toJobConfig.orientation	否	枚举	存储方式，此参数只有当数据库类型为DWS时启用，当需要自动创建DWS数据库表，指定表的数据存储方式： ● ROW：表的数据以行式存储。 ● COLUMN：表的数据以列式存储。
toJobConfig.isCompress	否	Boolean	是否压缩，此参数只有当数据库类型为DWS时启用，当需要自动创建DWS数据库表，指定是否对表的数据进行压缩存储。
toJobConfig.useStageTable	否	Boolean	先导入阶段表，如果设置为“true”，数据导入目的表之前会把数据先导入阶段表，如果成功导入阶段表，则再从阶段表导入到目的表，这样避免导入过程失败，在目的表遗留部分成功数据。
toJobConfig.extendCharLength	否	Boolean	扩大字符字段长度，如果设置为“true”，当需要自动创建目的表时，目标表的字符类型字段长度设置为源表相应字段长度的3倍。
toJobConfig.useNullable	否	Boolean	当选择自动创建目的表时，如果选择使用非空约束，则目的表字段的是否非空约束，与原表具有相应非空约束的字段保持一致。

6.3.2 目的端为 OBS

JSON 样例

```
"to-config-values": {
      "configs": [
       {
         "inputs": [
          {
            "name": "toJobConfig.bucketName",
            "value": "cdm"
          },
          {
            "name": "toJobConfig.outputDirectory",
            "value": "/obsfrom/advance/"
          },
          {
            "name": "toJobConfig.outputFormat",
            "value": "CSV_FILE"
          },
```

```
        {
          "name": "toJobConfig.fieldSeparator",
          "value": ","
        },
        {
          "name": "toJobConfig.writeToTempFile",
          "value": "false"
        },
        {
          "name": "toJobConfig.validateMD5",
          "value": "false"
        },
        {
          "name": "toJobConfig.recordMD5Result",
          "value": "false"
        },
        {
          "name": "toJobConfig.encodeType",
          "value": "UTF-8"
        },
        {
          "name": "toJobConfig.markerFile",
          "value": "finish.txt"
        },
        {
          "name": "toJobConfig.duplicateFileOpType",
          "value": "REPLACE"
        },
        {
          "name": "toJobConfig.columnList",
          "value": "1&2"
        },
        {
          "name": "toJobConfig.quoteChar",
          "value": "false"
        },
        {
          "name": "toJobConfig.encryption",
          "value": "NONE"
        },
        {
          "name": "toJobConfig.copyContentType",
          "value": "false"
        }
      ],
      "name": "toJobConfig"
    }
  ]
}
```

参数说明

参数	是否必选	类型	说明
toJobConfig.bucketName	是	String	OBS的桶名，例如“cdm”。
toJobConfig.outputDirectory	是	String	数据写入路径，例如“data_dir”。

参数	是否必选	类型	说明
toJobConfig.outputFormat	是	枚举	写入数据时所用的文件格式（二进制除外），支持以下文件格式： ● CSV_FILE：按照CSV格式写入数据。 ● BINARY_FILE：二进制格式，不解析文件内容直接传输，CDM会原样写入文件，不改变原始文件格式。 当选择“BINARY_FILE”时，源端也必须为文件系统。
toJobConfig.fieldSeparator	否	String	列分割符号，当“toJobConfig.outputFormat”（文件格式）为“CSV_FILE”时此参数有效，默认值为：“,”。
toJobConfig.lineSeparator	否	String	行分割符号，当“toJobConfig.outputFormat”（文件格式）为“CSV_FILE”时此参数有效，默认值为：“\r\n”。
toJobConfig.writeFileSize	否	String	源端为数据库时该参数有效，支持按大小分成多个文件存储，避免导出的文件过大，单位为MB。
toJobConfig.duplicateFileOpType	否	枚举	重复文件处理方式，只有文件名和文件大小都相同才会判定为重复文件。重复文件支持以下处理方式： ● REPLACE：替换重复文件。 ● SKIP：跳过重复文件。 ● ABANDON：发现重复文件停止任务。
toJobConfig.columnList	否	String	需要抽取的字段列表，字段名之间使用“&”分割，例如：“id&gid&name”。

参数	是否必选	类型	说明
toJobConfig.encryption	否	枚举	选择是否对上传的数据进行加密，以及加密方式： ● NONE：不加密，直接写入数据。 ● KMS：使用数据加密服务中的KMS进行加密。如果启用KMS加密则无法进行数据的MD5校验。
toJobConfig.kmsID	否	String	上传时加密使用的密钥。需先在密钥管理服务中创建密钥。
toJobConfig.projectID	否	String	KMS密钥所属的项目ID。
toJobConfig.writeToTempFile	否	Boolean	将二进制文件先写入到临时文件（临时文件以“.tmp”作为后缀），迁移成功后，再进行rename或move操作，在目的端恢复文件。
toJobConfig.validateMD5	否	Boolean	选择是否校验MD5值，不能与KMS加密同时使用。使用二进制格式传输文件时，才能校验MD5值。 计算源文件的MD5值，并与OBS返回的MD5值进行校验。如果源端已经存在MD5文件，则直接读取源端的MD5文件与OBS返回的MD5值进行校验。
toJobConfig.recordMD5Result	否	Boolean	当选择校验MD5值时，这里配置是否记录校验结果。
toJobConfig.recordMD5Link	否	String	可以指定任意一个OBS连接，将MD5校验结果写入该连接的桶。
toJobConfig.recordMD5Bucket	否	String	写入MD5校验结果的OBS桶。
toJobConfig.recordMD5Directory	否	String	写入MD5校验结果的目录。
toJobConfig.encodeType	否	String	编码类型，例如：“UTF_8”或“GBK”。
toJobConfig.markerFile	否	String	当作业执行成功时，会在写入目录下生成一个标识文件，文件名由用户指定，不指定时默认关闭该功能。

参数	是否必选	类型	说明
toJobConfig.copyContentType	否	Boolean	“toJobConfig.outputFormat”（文件格式）为“BINARY_FILE”，且源端、目的端都为对象存储时，才有该参数。 选择“是”后，迁移对象文件时会复制源文件的Content-Type属性，主要用于静态网站的迁移场景。 归档存储的桶不支持设置Content-Type属性，所以如果开启了该参数，目的端选择写入的桶时，必须选择非归档存储的桶。
toJobConfig.quoteChar	否	Boolean	“toJobConfig.outputFormat”（文件格式）为“CSV_FILE”，才有该参数，用于将数据库的表迁移到文件系统的场景。 选择“是”时，如果源端数据表中的某一个字段内容包含字段分隔符或换行符，写入目的端时CDM会使用双引号（"）作为包围符将该字段内容括起来，作为一个整体存储，避免其中的字段分隔符误将一个字段分隔成两个，或者换行符误将字段换行。例如：数据库中某字段为hello,world，使用包围符后，导出到CSV文件的时候数据为"hello,world"。
toJobConfig.firstRowAsHeader	否	Boolean	“toJobConfig.outputFormat”（文件格式）为“CSV_FILE”时才有该参数。在迁移表到CSV文件时，CDM默认是不迁移表的标题行，如果该参数选择“是”，CDM在才会将表的标题行数据写入文件。

参数	是否必选	类型	说明
toJobConfig.filePrefix	否	String	自定义文件名前缀，支持任意自定义名称，也支持表名宏时间宏版本宏。例如：${tableName}_${dateformat(yyyy-MM-dd HH:mm:ss, -1, DAY)}_${version}。 注意文件名格式要符合obs文件路径命名规范。

6.3.3 目的端为 HDFS

JSON 样例

```
"to-config-values": {
      "configs": [
       {
        "inputs": [
         {
          "name": "toJobConfig.outputDirectory",
          "value": "/hdfsto"
         },
         {
          "name": "toJobConfig.outputFormat",
          "value": "BINARY_FILE"
         },
         {
          "name": "toJobConfig.writeToTempFile",
          "value": "false"
         },
         {
          "name": "toJobConfig.duplicateFileOpType",
          "value": "REPLACE"
         },
         {
          "name": "toJobConfig.compression",
          "value": "NONE"
         },
         {
          "name": "toJobConfig.appendMode",
          "value": "true"
         }
        ],
        "name": "toJobConfig"
       }
      ]
     }
```

参数说明

参数	是否必选	类型	说明
toJobConfig.outputDirectory	是	String	数据写入的路径，例如“/data_dir”。

参数	是否必选	类型	说明
toJobConfig.outputFormat	是	枚举	写入数据时所用的文件格式（二进制除外），支持以下文件格式： ● CSV_FILE：按照CSV格式写入数据。 ● BINARY_FILE：二进制格式，不解析文件内容直接传输，CDM会原样写入文件，不改变原始文件格式。 当选择“BINARY_FILE”时，源端也必须为文件系统。
toJobConfig.lineSeparator	否	String	行分割符号，当“toJobConfig.outputFormat”（文件格式）为“CSV_FILE”时此参数有效，默认值为：“\r\n”。
toJobConfig.fieldSeparator	否	String	列分割符号，当“toJobConfig.outputFormat”（文件格式）为“CSV_FILE”时此参数有效，默认值为：“,”。
toJobConfig.writeToTempFile	否	Boolean	将二进制文件先写入到临时文件（临时文件以“.tmp”作为后缀），迁移成功后，再进行rename或move操作，在目的端恢复文件。
toJobConfig.duplicateFileOpType	否	枚举	重复文件处理方式，只有文件名和文件大小都相同才会判定为重复文件。重复文件支持以下处理方式： ● REPLACE：替换重复文件。 ● SKIP：跳过重复文件。 ● ABANDON：发现重复文件停止任务。

参数	是否必选	类型	说明
toJobConfig.compression	否	枚举	写入文件后，选择对文件的压缩格式。支持以下压缩格式： ● NONE：不压缩。 ● DEFLATE：压缩为DEFLATE格式。 ● GZIP：压缩为GZIP格式。 ● BZIP2：压缩为BZIP2格式。 ● LZ4：压缩为LZ4格式。 ● SNAPPY：压缩为SNAPPY格式。
toJobConfig.appendMode	是	Boolean	当加载路径已经存在文件，是否需要写入，默认值为“false”。
toJobConfig.encryption	否	枚举	当“toJobConfig.outputFormat”（文件格式）为“BINARY_FILE”（二进制）时才有该参数，选择是否对导入的数据进行加密，以及加密方式： ● NONE：不加密，直接写入数据。 ● AES-256-GCM：使用长度为256byte的AES对称加密算法，目前加密算法只支持AES-256-GCM（NoPadding）。
toJobConfig.dek	否	String	数据加密密钥，“toJobConfig.encryption”（加密方式）选择“AES-256-GCM”时有该参数，密钥由长度64的十六进制数组成。 请您牢记这里配置的密钥，解密时的密钥与这里配置的必须一致。如果不一致系统不会报异常，只是解密出来的数据会错误。

参数	是否必选	类型	说明
toJobConfig.iv	否	String	初始化向量，“toJobConfig.encryption”（加密方式）选择“AES-256-GCM”时有该参数，初始化向量由长度32的十六进制数组成。 请您牢记这里配置的初始化向量，解密时的初始化向量与这里配置的必须一致。如果不一致系统不会报异常，只是解密出来的数据会错误。
toJobConfig.filePrefix	否	String	自定义文件名前缀，支持时间宏。例如：test_${dateformat(yyyyMMdd, -1, DAY)} 注意文件名格式要符合hdfs文件路径命名规范。

6.3.4 目的端为 Hive

JSON 样例

```
"to-config-values": {
       "configs": [
        {
          "inputs": [
            {
              "name": "toJobConfig.hive",
              "value": "hive"
            },
            {
              "name": "toJobConfig.database",
              "value": "rf_database"
            },
            {
              "name": "toJobConfig.table",
              "value": "rf_to"
            },
            {
              "name": "toJobConfig.tablePreparation",
              "value": "DO_NOTHING"
            },
            {
              "name": "toJobConfig.columnList",
              "value": "aa&bb&cc&dd"
            },
            {
              "name": "toJobConfig.shouldClearTable",
              "value": "true"
            }
          ],
          "name": "toJobConfig"
        }
      ]
    }
```

参数说明

参数	是否必选	类型	说明
toJobConfig.hive	否	String	写入数据的数据源。
toJobConfig.database	否	String	写入数据的数据库名称，例如：“default”。
toJobConfig.table	是	String	写入数据的表名。
toJobConfig.tablePreparation	是	枚举	写入表数据时，用户选择的操作： ● DO_NOTHING：不自动建表。 ● CREATE_WHEN_NOT_EXIST：当目的端的数据库没有“tableName”参数中指定的表时，CDM会自动创建该表。 ● DROP_AND_CREATE：先删除“tableName”参数中指定的表，然后再重新创建该表。
toJobConfig.columnList	否	String	需要加载的字段列表，字段名之间使用“&”分割，例如：“id&gid&name”。
toJobConfig.shouldClearTable	否	Boolean	导入前是否清空目标表的数据，如果设置为true，任务启动前会清除目标表中数据。

6.3.5 目的端为 HBase/CloudTable

JSON 样例

```
"to-config-values": {
      "configs": [
       {
         "inputs": [
          {
            "name": "toJobConfig.table",
            "value": "rf_to"
          },
          {
            "name": "toJobConfig.storageType",
            "value": "PUTLIST"
          },
          {
            "name": "toJobConfig.columns",
            "value": "AA:AA&BB:BB&CC:CC&DD:DD"
          },
          {
```

```
            "name": "toJobConfig.rowKeyColumn",
            "value": "AA:AA"
          },
          {
            "name": "toJobConfig.isOverride",
            "value": "false"
          },
          {
            "name": "toJobConfig.isRowkeyRedundancy",
            "value": "false"
          },
          {
            "name": "toJobConfig.algorithm",
            "value": "NONE"
          },
          {
            "name": "toJobConfig.writeToWAL",
            "value": "true"
          },
          {
            "name": "toJobConfig.transType",
            "value": "false"
          }
        ],
        "name": "toJobConfig"
      }
    ]
  }
```

参数说明

参数	是否必选	类型	说明
toJobConfig.table	是	String	写入数据的表名，例如：“TBL_EXAMPLE”。
toJobConfig.storageType	是	枚举	将数据写入到HBase表中的方法： ● PUTLIST：put list方式写入。
toJobConfig.columns	否	String	需要抽取数据的列，列号之间使用“&”分割，列族与列之间用“:”分隔，例如：“cf1:c1&cf2:c2”。
toJobConfig.rowKeyColumn	是	String	作为rowkey的列，列号之间使用“&”分割，列族与列之间用“:”分隔，例如：“cf1:c1&cf2:c2”。
toJobConfig.isOverride	否	Boolean	使用BULKLOAD方式导入数据时，是否清空数据，例如：“true”。
toJobConfig.delimiter	否	String	当选取多个列做rowkey时，连接多列的分隔符，例如：“|”。

参数	是否必选	类型	说明
toJobConfig.isRowkeyRedundancy	否	Boolean	是否将选做Rowkey的数据同时写入HBase的列。
toJobConfig.algorithm	否	枚举	创建新HBase表时采用的压缩算法，支持SNAPPY和GZ算法，默认为“NONE”。
toJobConfig.writeToWAL	否	Boolean	选择是否开启HBase的预写日志机制（WAL，Write Ahead Log）。 ● 是：开启后如果出现HBase服务器宕机，则可以从WAL中回放执行之前没有完成的操作。 ● 否：关闭时能提升写入性能，但如果HBase服务器宕机可能会造成数据丢失。
toJobConfig.transType	否	Boolean	● true：源端数据库中的Short、Int、Long、Float、Double、Decimal类型列的数据，会转换为Byte[]数组（二进制）写入HBase，其他类型的按字符串写入。如果这几种类型中，有合并做rowkey的，就依然当字符串写入。 该功能作用是：降低存储占用空间，存储更高效；特定场景下rowkey分布更均匀。 ● false：源端数据库中所有类型的数据，都会按照字符串写入HBase。

6.3.6 目的端为 DDS

JSON 样例

```
"to-config-values": {
       "configs": [
        {
          "inputs": [
            {
              "name": "toJobConfig.database",
              "value": "demo"
            },
            {
              "name": "toJobConfig.collectionName",
              "value": "cdmbase"
            },
            {
```

```
            "name": "toJobConfig.columnList",
            "value": "_char&_varchar"
          },
          {
            "name": "toJobConfig.isBatchMigration",
            "value": "false"
          }
        ],
        "name": "toJobConfig"
      }
    ]
  }
```

参数说明

参数	是否必选	类型	说明
toJobConfig.database	是	String	MongoDB/DDS的数据库名称
toJobConfig.collectionName	是	String	MongoDB/DDS的集合名称。
toJobConfig.columnList	否	String	需要抽取的字段列表，字段名之间使用“&”分割，例如：“id&gid&name”。
toJobConfig.isBatchMigration	否	Boolean	是否为整库迁移。

6.3.7 目的端为 Elasticsearch/云搜索服务

JSON 样例

```
"to-config-values": {
        "configs": [
          {
            "inputs": [
              {
                "name": "toJobConfig.index",
                "value": "cdm"
              },
              {
                "name": "toJobConfig.type",
                "value": "type1"
              },
              {
                "name": "toJobConfig.shouldClearType",
                "value": "false"
              },
              {
                "name": "toJobConfig.pipeLine",
                "value": "es_03"
              }
            ],
            "name": "toJobConfig"
          }
        ]
      }
```

参数说明

参数	是否必选	类型	说明
toJobConfig.index	是	String	写入数据的索引，类似关系数据库中的数据库名称。
toJobConfig.type	是	String	写入数据的类型，类似关系数据库中的表名。
toJobConfig.shouldClearType	否	Boolean	导入前是否清除数据。
toJobConfig.primaryKey	否	String	主键或唯一索引。
toJobConfig.columnList	否	String	需要写入的字段列表，字段名之间使用“&”分隔，例如：“id&gid&name”。
toJobConfig.pipeLine	否	String	需要先在kibana中创建管道ID，这里才可以选择，该参数用于数据传到云搜索服务/Elasticsearch后，通过Elasticsearch的数据转换pipeline进行数据格式变换。

参数	是否必选	类型	说明
toJobConfig.createIndexStrategy	否	枚举	对于持续写入数据到Elasticsearch的流式作业，CDM支持在Elasticsearch中定时创建新索引并写入数据，方便用户后期删除过期的数据。支持按以下周期创建新索引： ● EveryHour：每小时整点创建新索引，新索引的命名格式为“索引名+年+月+日+小时”，例如“index2018121709”。 ● EveryDay：每天零点零分创建新索引，新索引的命名格式为“索引名+年+月+日”，例如“index20181217”。 ● EveryWeek：每周周一的零点零分创建新索引，新索引的命名格式为“索引名+年+周”，例如“index201842”。 ● EveryMonth：每月一号零点零分创建新索引，新索引的命名格式为“索引名+年+月”，例如“index201812”。 从文件类抽取数据时，必须配置单个抽取（“抽取并发数”参数配置为1），否则该参数无效。

6.3.8 目的端为 DLI

JSON 样例

```
"to-config-values": {
            "configs": [
               {
                  "inputs": [
                     {
                        "name": "toJobConfig.queue",
                        "value": "cdm"
                     },
                     {
                        "name": "toJobConfig.database",
                        "value": "sqoop"
                     },
                     {
                        "name": "toJobConfig.table",
                        "value": "est1"
                     },
                     {
                        "name": "toJobConfig.columnList",
                        "value":
```

```
"string_&int_&date_&double_&boolean_&short_&timestamp_&long_&smallint_&bigint_"
                    },
                    {
                        "name": "toJobConfig.shouldClearTable",
                        "value": "false"
                    }
                ],
                "name": "toJobConfig"
            }
        ]
    }
```

参数说明

参数	是否必选	类型	说明
toJobConfig.queue	是	String	写入数据的资源队列。
toJobConfig.database	是	String	写入数据到数据湖探索（DLI）的哪个数据库。
toJobConfig.table	是	String	写入数据的表名。
toJobConfig.columnList	否	String	需要加载的字段列表，字段名之间使用“&”分割，例如：“id&gid&name”。
toJobConfig.shouldClearTable	否	Boolean	导入前是否清空资源队列的数据。

6.3.9 目的端为 DIS

JSON 样例

```
"to-config-values": {
    "configs": [
      {
        "inputs": [
          {
            "name": "toJobConfig.streamName",
            "value": "cdm"
          },
          {
            "name": "toJobConfig.separator",
            "value": ","
          },
          {
            "name": "toJobConfig.identifierEnclose",
            "value": "'"
          }
        ],
        "name": "toJobConfig"
      }
    ]
  }
```

参数说明

参数	是否必选	类型	说明
toJobConfig.streamName	是	String	DIS的通道名。
toJobConfig.separator	否	String	字段分隔符，默认为空格。
toJobConfig.identifierEnclose	否	String	连接引用表名或列名时的分隔符号，默认为空。

6.4 作业任务参数说明

在**指定集群创建作业**或者**随机集群创建作业并执行**时，由“driver-config-values”参数指定作业任务配置，包含如下功能：

- 作业失败重试：如果作业执行失败，可选择是否自动重新启动作业。
- 作业分组：CDM支持对作业进行分组，分组后的作业可以按组显示、按组批量删除作业、按组批量启动作业、按组导出等。
- 是否定时执行：可选择作业是否定时自动启动。
- 抽取并发数：可设置同时执行的抽取任务数。
- 是否写入脏数据：如果需要将作业执行过程中处理失败的数据、或者被清洗过滤掉的数据写入OBS中，以便后面查看，可通过该参数配置，写入脏数据前需要先配置好OBS连接。
- 作业运行完是否删除：可选择是否自动删除作业。

JSON 样例

```
"driver-config-values": {
      "configs": [
        {
          "inputs": [
            {
              "name": "throttlingConfig.numExtractors",
              "value": "1"
            },
            {
              "name": "throttlingConfig.numLoaders",
              "value": "1"
            },
            {
              "name": "throttlingConfig.recordDirtyData",
              "value": "false"
            }
          ],
          "name": "throttlingConfig"
        },
        {
          "inputs": [],
          "name": "jarConfig"
        },
        {
          "inputs": [
            {
              "name": "schedulerConfig.isSchedulerJob",
```

```
            "value": "false"
          },
          {
            "name": "schedulerConfig.disposableType",
            "value": "NONE"
          }
        ],
        "name": "schedulerConfig"
      },
      {
        "inputs": [],
        "name": "transformConfig"
      },
      {
        "inputs": [
          {
            "name": "retryJobConfig.retryJobType",
            "value": "NONE"
          }
        ],
        "name": "retryJobConfig"
      }
    ]
  }
```

参数说明

参数	是否必选	类型	说明
throttlingConfig.numExtractors	否	Integer	最大抽取任务并发数，例如：“20”。
groupJobConfig.groupName	否	枚举	选择作业所属的分组，默认分组为“DEFAULT”。
throttlingConfig.numLoaders	否	Integer	仅当HBase或Hive作为目的数据源时该参数才有效。 最大加载任务数，例如：“5”。
throttlingConfig.recordDirtyData	否	Boolean	是否写入脏数据，例如：“true”。
throttlingConfig.writeToLink	否	String	脏数据要写入的连接，目前只支持写入到OBS连接或HDFS连接。例如：“obslink”。
throttlingConfig.obsBucket	否	String	写入脏数据的OBS桶的名称，只有当脏数据要写入OBS连接的时候，此参数才生效。例如：“dirtyData”。
throttlingConfig.dirtyDataDirectory	否	String	写入脏数据的目录： ● 如果选择写入到HDFS，此参数即为HDFS目录。 ● 如果选择写入到OBS，此参数表示相应的OBS桶下的目录，例如：“/data/dirtydata/”。

参数	是否必选	类型	说明
throttlingConfig.maxErrorRecords	否	String	单个分片的最大错误记录数。单个map的错误记录超过设置的最大错误记录数时，任务自动结束，已经导入的数据不回退。
schedulerConfig.isSchedulerJob	否	Boolean	是否开启定时任务，例如：“true”。
schedulerConfig.cycleType	否	String	定时任务的周期类型，目前支持五种周期类型： ● minute：分钟 ● hour：小时 ● day：天 ● week：周 ● month：月
schedulerConfig.cycle	否	Integer	定时任务的周期，如果周期类型选择了“minute”，“cycle”输入“10”，就表示该定时任务每10分钟执行一次。
schedulerConfig.runAt	否	String	定时任务在周期内的触发时间，当周期为“hour”、“week”或“month”时，该参数有效。 ● 如果周期类型为“month”，设定周期为“1”，“runAt”输入“15”，就表示每个月的15号执行该定时任务。并且该参数支持输入多个，以英文的逗号“,”分隔开。 例如上述场景中“runAt”输入“1,2,3,4,5”，就表示每个月的1日、2日、3日、4日和5日执行该定时任务。 ● 周期类型为“week”时，“runAt”输入“mon,tue,wed,thu,fri”，就表示周一到周五执行该定时任务。 ● 周期类型为“hour”时，“runAt”输入“27,57”，表示周期内的27分和57分执行该定时任务。
schedulerConfig.startDate	否	String	定时任务的开始时间，例如：“2018-01-24 19:56:19”。

参数	是否必选	类型	说明
schedulerConfig.stopDate	否	String	定时任务的结束日期，例如：“2018-01-27 23:59:00”。 如果不输入结束时间，则表示定时任务将一直执行，永远不会结束。
schedulerConfig.disposableType	否	枚举	作业运行完是否删除： ● NONE：作业执行完不删除。 ● DELETE_AFTER_SUCCEED：仅作业执行成功时删除该作业，适合海量一次性作业。 ● DELETE：作业执行完删除该作业，执行成功或失败都会删除。
retryJobConfig.retryJobType	否	枚举	如果作业执行失败，选择是否自动重试： ● NONE：不重试。 ● RETRY_TRIPLE：重试三次。

7 权限及授权项说明

如果您需要对您所拥有的CDM服务进行精细的权限管理，您可以使用统一身份认证服务（Identity and Access Management，简称IAM）。如果云账号已经能满足您的要求，不需要创建独立的IAM用户，您可以跳过本章节，不影响您使用CDM服务。

默认情况下，新建的IAM用户没有任何权限，您需要将其加入用户组，并给用户组授予策略或角色，才能使用户组中的用户获得相应的权限，这一过程称为授权。授权后，用户就可以基于已有权限对云服务进行操作。

权限根据授权的精细程度，分为**角色**和**策略**。角色以服务为粒度，是IAM最初提供的一种根据用户的工作职能定义权限的粗粒度授权机制。策略以API接口为粒度进行权限拆分，授权更加精细，可以精确到某个操作、资源和条件，能够满足企业对权限最小化的安全管控要求。

说明

如果您要允许或是禁止某个接口的操作权限，请使用策略。

账号具备所有接口的调用权限，如果使用账号下的IAM用户发起API请求时，该IAM用户必须具备调用该接口所需的权限，否则，API请求将调用失败。每个接口所需要的权限，与各个接口所对应的授权项相对应，只有发起请求的用户被授予授权项所对应的策略，该用户才能成功调用该接口。例如，用户要调用接口来查询集群列表，那么这个IAM用户被授予的策略中必须包含允许“cdm:cluster:list”的授权项，该接口才能调用成功。

支持的授权项

策略包含系统策略和自定义策略，如果系统策略不满足授权要求，管理员可以创建自定义策略，并通过给用户组授予自定义策略来进行精细的访问控制。策略支持的操作与API相对应，授权项列表说明如下：

- 权限：允许或拒绝某项操作。
- 对应API接口：自定义策略实际调用的API接口。
- 授权项：自定义策略中支持的Action，在自定义策略中的Action中写入授权项，可以实现授权项对应的权限功能。
- IAM项目(Project)/企业项目(Enterprise Project)：自定义策略的授权范围，包括IAM项目与企业项目。授权范围如果同时支持IAM项目和企业项目，表示此授权项对应的自定义策略，可以在IAM和企业管理两个服务中给用户组授权并生效。如

果仅支持IAM项目，不支持企业项目，表示仅能在IAM中给用户组授权并生效，如果在企业管理中授权，则该自定义策略不生效。

说明

“√”表示支持，“x”表示暂不支持。

CDM的支持自定义策略授权项如表7-1所示，表中的授权项作用域支持项目（Project）和企业项目（Enterprise Project）。

表 7-1 API 授权项列表

权限	对应API接口	授权项（Action）	IAM项目(Project)	企业项目(Enterprise Project)
创建集群	POST /v1.1/{project_id}/clusters	cdm:cluster:create	√	×
查询集群列表	GET /v1.1/{project_id}/clusters	cdm:cluster:list	√	×
查询集群详情	GET /v1.1/{project_id}/clusters/{cluster_id}	cdm:cluster:get	√	×
重启集群	POST /v1.1/{project_id}/clusters/{cluster_id}/action	cdm:cluster:operate	√	×
修改集群配置	POST /v1.1/{project_id}/cluster/modify/{cluster_id}	cdm:cluster:modify	√	×
删除集群	DELETE /v1.1/{project_id}/clusters/{cluster_id}	cdm:cluster:delete	√	×
创建连接	POST /v1.1/{project_id}/clusters/{cluster_id}/cdm/link	cdm:link:operate	√	×

权限	对应API接口	授权项（Action）	IAM项目(Project)	企业项目(Enterprise Project)
查询连接	GET /v1.1/{project_id}/clusters/{cluster_id}/cdm/link/{linkName}	cdm:cluster:get	√	×
修改连接	PUT /v1.1/{project_id}/clusters/{cluster_id}/cdm/link/{link_name}	cdm:link:operate	√	×
删除连接	DELETE /v1.1/{project_id}/clusters/{cluster_id}/cdm/link/{linkName}	cdm:link:operate	√	×
指定集群创建作业	POST /v1.1/{project_id}/clusters/{cluster_id}/cdm/job	cdm:job:operate	√	×
查询作业	GET /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{jobName}	cdm:cluster:get	√	×
修改作业	PUT /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{jobName}	cdm:job:operate	√	×
启动作业	PUT /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{jobName}/start	cdm:job:operate	√	×
停止作业	PUT /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{jobName}/stop	cdm:job:operate	√	×

权限	对应API接口	授权项（Action）	IAM项目(Project)	企业项目(Enterprise Project)
查询作业状态	GET /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{jobName}/status	cdm:cluster:get	√	×
查询作业执行历史	GET /v1.1/{project_id}/clusters/{cluster_id}/cdm/submissions	cdm:cluster:get	√	×
删除作业	DELETE /v1.1/{project_id}/clusters/{cluster_id}/cdm/job/{jobName}	cdm:job:operate	√	×

8 附录

8.1 状态码

状态码如表8-1所示。

表 8-1 状态码

状态码	编码	状态说明
100	Continue	继续请求。 这个临时响应用来通知客户端，它的部分请求已经被服务器接收，且仍未被拒绝。
101	Switching Protocols	切换协议。只能切换到更高级的协议。 例如，切换到HTTP的新版本协议。
201	Created	创建类的请求完全成功。
202	Accepted	已经接受请求，但未处理完成。
203	Non-Authoritative Information	非授权信息，请求成功。
204	NoContent	请求完全成功，同时HTTP响应不包含响应体。 在响应OPTIONS方法的HTTP请求时返回此状态码。
205	Reset Content	重置内容，服务器处理成功。
206	Partial Content	服务器成功处理了部分GET请求。
300	Multiple Choices	多种选择。请求的资源可包括多个位置，相应可返回一个资源特征与地址的列表用于用户终端（例如：浏览器）选择。
301	Moved Permanently	永久移动，请求的资源已被永久的移动到新的URI，返回信息会包括新的URI。

状态码	编码	状态说明
302	Found	资源被临时移动。
303	See Other	查看其它地址。 使用GET和POST请求查看。
304	Not Modified	所请求的资源未修改，服务器返回此状态码时，不会返回任何资源。
305	Use Proxy	所请求的资源必须通过代理访问。
306	Unused	已经被废弃的HTTP状态码。
400	BadRequest	非法请求。 建议直接修改该请求，不要重试该请求。
401	Unauthorized	在客户端提供认证信息后，返回该状态码，表明服务端指出客户端所提供的认证信息不正确或非法。
402	Payment Required	保留请求。
403	Forbidden	请求被拒绝访问。 返回该状态码，表明请求能够到达服务端，且服务端能够理解用户请求，但是拒绝做更多的事情，因为该请求被设置为拒绝访问，建议直接修改该请求，不要重试该请求。
404	NotFound	所请求的资源不存在。 建议直接修改该请求，不要重试该请求。
405	MethodNotAllowed	请求中带有该资源不支持的方法。 建议直接修改该请求，不要重试该请求。
406	Not Acceptable	服务器无法根据客户端请求的内容特性完成请求。
407	Proxy Authentication Required	请求要求代理的身份认证，与401类似，但请求者应当使用代理进行授权。
408	Request Time-out	服务器等候请求时发生超时。 客户端可以随时再次提交该请求而无需进行任何更改。
409	Conflict	服务器在完成请求时发生冲突。 返回该状态码，表明客户端尝试创建的资源已经存在，或者由于冲突请求的更新操作不能被完成。
410	Gone	客户端请求的资源已经不存在。 返回该状态码，表明请求的资源已被永久删除。

状态码	编码	状态说明
411	Length Required	服务器无法处理客户端发送的不带Content-Length的请求信息。
412	Precondition Failed	未满足前提条件，服务器未满足请求者在请求中设置的其中一个前提条件。
413	Request Entity Too Large	由于请求的实体过大，服务器无法处理，因此拒绝请求。为防止客户端的连续请求，服务器可能会关闭连接。如果只是服务器暂时无法处理，则会包含一个Retry-After的响应信息。
414	Request-URI Too Large	请求的URI过长（URI通常为网址），服务器无法处理。
415	Unsupported Media Type	服务器无法处理请求附带的媒体格式。
416	Requested range not satisfiable	客户端请求的范围无效。
417	Expectation Failed	服务器无法满足Expect的请求头信息。
422	UnprocessableEntity	请求格式正确，但是由于含有语义错误，无法响应。
429	TooManyRequests	表明请求超出了客户端访问频率的限制或者服务端接收到多于它能处理的请求。建议客户端读取相应的Retry-After首部，然后等待该首部指出的时间后再重试。
500	InternalServerError	表明服务端能被请求访问到，但是不能理解用户的请求。
501	Not Implemented	服务器不支持请求的功能，无法完成请求。
502	Bad Gateway	充当网关或代理的服务器，从远端服务器接收到了一个无效的请求。
503	ServiceUnavailable	被请求的服务无效。 建议直接修改该请求，不要重试该请求。
504	ServerTimeout	请求在给定的时间内无法完成。客户端仅在为请求指定超时（Timeout）参数时会得到该响应。
505	HTTP Version not supported	服务器不支持请求的HTTP协议的版本，无法完成处理。

8.2 错误码

调用API出错后，将不会返回结果数据。调用方可根据每个API对应的错误码来定位错误原因。当调用出错时，HTTP请求返回一个4xx或5xx的HTTP状态码。返回的消息体

中是具体的错误代码及错误信息。在调用方找不到错误原因时，可以联系客服，并提供错误码，以便尽快帮您解决问题。

- 异常响应样例

```
{
  "errCode": "Cdm.0100",
  "externalMessage": "Job[jdbc2hive] doesn't exist."
}
```

- 参数说明

参数	是否必选	类型	说明
errCode	否	String	错误码。
externalMessage	否	String	错误消息。

- 错误码说明

  以下错误信息中的*%s*为变量，实际返回信息时会替换为具体的参数名、表名、作业名、连接名等。

  当您调用API时，如果遇到“APIGW”开头的错误码，请参见**API网关错误码**进行处理。

错误码	状态码	错误信息	描述	处理措施
Cdm.0000	400	系统错误。	系统错误。	请联系客服或技术支持人员协助解决。
Cdm.0001	400	资源不存在或不合法。	请求的资源不存在或无访问权限。	请联系客服或技术支持人员协助解决。
Cdm.0004	400	无效的参数类型	输入参数和类型不匹配	请根据错误提示将参数修改正确后请重试。
Cdm.0009	400	*%s*不是整型数字或超出整型数的取值范围[0～2147483647]。	输入参数不是整型数字或超出整型数的取值范围。	请根据错误提示将参数修改正确后请重试。
Cdm.0010	400	整数必须在区间[*%s*]。	校验程序的参数缺失或长度为0。	请根据错误提示将参数修改正确后请重试。
Cdm.0011	400	输入超过取值范围。	参数格式不正确或超过取值范围，无法解析。	请根据返回的详细错误信息，确认参数值是否合法，修改正确后请重试。

错误码	状态码	错误信息	描述	处理措施
Cdm.0012	400	没有匹配的数据库JDBC驱动。	没有匹配的数据库JDBC驱动。	请联系客服或技术支持人员协助解决。
Cdm.0014	400	非法参数。	参数不合法。	请确认参数值是否合法，修改正确后请重试。
Cdm.0015	400	解析文件内容出错。	解析文件内容失败。	请确认上传的文件内容或格式是否正确，修改正确后请重试。
Cdm.0016	400	上传文件不能为空。	上传的文件为空。	请确认上传的文件是否为空，修改正确后请重试。
Cdm.0017	400	无法将输入值保存到存储库	无法将输入值保存到存储库。	请联系客服或技术支持人员协助解决。
Cdm.0018	400	作业和连接内容不合法。	作业和连接内容非法。	请联系客服或技术支持人员协助解决。
Cdm.0019	400	无法删除存储库中的链接	删除存储库中的链接失败。	请稍后重试，或联系或客服或技术支持人员协助解决。
Cdm.0020	400	必须包含子字符串：*%s*。	被校验参数为空或不包含指定子字符串。	请根据错误提示将参数修改正确后，再重试。
Cdm.0021	400	不能连接服务器：*%s*。	连接服务器失败。	请联系客服或技术支持人员协助解决。
Cdm.0024	400	[*%s*]必须在区间[*%s*]。	被校验参数不在指定区间范围内。	请根据错误提示将参数修改正确后，再重试。
Cdm.0031	400	无法创建新的提交数据	无法创建新的提交数据	请联系客服或技术支持人员协助解决。

错误码	状态码	错误信息	描述	处理措施
Cdm.0032	400	当前用户没有操作权限，请通过IAM检查账户权限！	用户权限不足	在CDM控制台操作时，请参考**CDM权限管理**为该用户授予足够的操作权限。 在DataArts Studio控制台操作时，请参考**DataArts Studio权限管理**为该用户授予足够的操作权限。
Cdm.0036	400	内部错误。	内部错误。 该报错由内部错误引起，显示的报错信息可能不唯一，请根据真实报错（如“内部错误：空指针异常”）联系客服或技术支持人员协助解决。	请联系客服或技术支持人员协助解决。
Cdm.0037	400	无法提交作业。	无法提交作业。	请联系客服或技术支持人员协助解决。
Cdm.0051	400	无效的提交引擎：*%s*。	作业引擎名称非法。	请指定正确的作业引擎后再重试。
Cdm.0052	400	作业*%s*正在运行。	作业正在运行。	作业正在运行，无法执行当前操作，请等待作业运行结束后再重试。
Cdm.0053	400	作业*%s*未运行。	作业未运行。	请运行作业后再重试。
Cdm.0054	400	作业*%s*不存在。	作业不存在。	请确认作业是否存在。
Cdm.0055	400	作业类型不支持。	作业类型不支持。	请参考官网，配置支持的作业类型。
Cdm.0056	400	不能提交作业。原因：*%s*。	作业提交失败。	请根据返回的详细错误信息，定位原因，修改正确后请重试。
Cdm.0057	400	无效的作业执行引擎：*%s*。	作业引擎无效。	请指定正确的作业引擎后再重试。

错误码	状态码	错误信息	描述	处理措施
Cdm.0058	400	提交和执行引擎组合不合法。	提交和执行引擎组合不合法。	请指定正确的作业引擎后再重试。
Cdm.0059	400	作业*%s*已被禁用。不能提交作业。	作业已被禁用，无法提交。	当前作业无法提交，建议重新创建一个作业后再重试。或者，请联系客服或技术支持人员协助解决。
Cdm.0060	400	作业使用的连接*%s*已被禁用。不能提交作业。	作业使用的连接已被禁用。	请改为其他连接后，再重新提交作业。
Cdm.0061	400	连接器*%s*不支持此方向。不能提交作业。	该连接器不能作为作业的源端或目的端。	该连接器不能作为作业的源端或目的端，请改为其他连接后，再重新提交作业。
Cdm.0062	400	二进制文件仅适合SFTP/FTP/HDFS/OBS连接器。	连接器不在指定的范围内。	请指定正确的连接器后再重试。
Cdm.0063	400	创建表格错误。原因：*%s*。	创建表格失败。	请根据返回的详细错误信息定位原因，修改正确后请重试。
Cdm.0064	400	数据格式不匹配。	数据格式不匹配。	请根据返回的详细错误信息，确认数据格式是否正确，修改正确后请重试。
Cdm.0065	400	定时器启动失败，原因*%s*。	定时器启动失败。	请联系客服或技术支持人员协助解决。
Cdm.0066	400	获取样值失败，原因：*%s*。	获取样值失败。	请联系客服或技术支持人员协助解决。
Cdm.0067	400	获取Schema失败，原因：*%s*。	获取Schema失败。	请联系客服或技术支持人员协助解决。

错误码	状态码	错误信息	描述	处理措施
Cdm.0085	400	%s 超过最大值 %s。	参数超过最大值。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0089	400	配置项 [%s] 不存在。	配置项不存在。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0100	400	作业[*%s*]不存在。	作业不存在。	请指定正确的作业后再重试。
Cdm.0101	400	连接[*%s*]不存在。	连接不存在。	请指定正确的连接后再重试。
Cdm.0102	400	连接器[*%s*]不存在。	连接器不存在。	请指定正确的连接器后再重试。
Cdm.0104	400	作业名已存在。	作业名已存在。	作业名已存在，请重新命名后，再重试。
Cdm.0201	400	获取实例失败。	获取实例失败。	请联系客服或技术支持人员协助解决。
Cdm.0202	400	作业状态未知。	作业状态未知。	请稍后重试，或请联系客服或技术支持人员协助解决。
Cdm.0204	400	没有已创建的MRS连接。	没有已创建的MRS连接。	当前没有MRS连接，您需要先前往集群的“连接管理”页面创建一个MRS连接，然后再重新执行当前的操作。
Cdm.0230	400	不能加载该类：*%s*。	类加载失败。	请联系客服或技术支持人员协助解决。
Cdm.0231	400	不能初始化该类：*%s*。	类初始化失败。	请联系客服或技术支持人员协助解决。
Cdm.0232	400	数据写入失败。原因：*%s*。	数据写入失败。	请联系客服或技术支持人员协助解决。

错误码	状态码	错误信息	描述	处理措施
Cdm.0233	400	提取数据过程异常。原因：*%s*。	提取数据过程异常。	请联系客服或技术支持人员协助解决。
Cdm.0234	400	载入数据过程异常。原因：*%s*。	载入数据过程异常。	请联系客服或技术支持人员协助解决。
Cdm.0235	400	数据已全部消费完毕。原因： *%s*。	数据已全部消费完毕。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0236	400	从分区程序中检索到无效分区数。	从分区程序中检索到无效分区数。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0238	400	*%s*不能为空。	参数不合法。	请根据错误提示将参数修改正确后再重试。
Cdm.0240	400	获取文件*%s*状态失败。	获取文件状态失败。	请联系客服或技术支持人员协助解决。
Cdm.0241	400	获取文件*%s*类型失败。	获取文件类型失败。	请联系客服或技术支持人员协助解决。
Cdm.0242	400	文件检查异常： *%s*。	文件检查异常。	请联系客服或技术支持人员协助解决。
Cdm.0243	400	重命名*%s*为*%s*失败。	重命名失败。	可能是名称已存在，请重新命名后再重试。
Cdm.0244	400	创建文件*%s*失败。	创建文件失败。	请确认是否具有创建权限，或稍后重试。若无法解决，请联系客服或技术支持人员协助解决。

错误码	状态码	错误信息	描述	处理措施
Cdm.0245	400	删除文件*%s*失败。	删除文件失败。	请确认是否具有删除权限，或稍后重试。若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0246	400	创建目录*%s*失败。	创建目录失败。	请确认是否具有创建权限，或稍后重试。若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0247	400	操作HBase失败。原因：*%s*。	操作HBase失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0248	400	清空*%s*数据失败。原因：*%s*。	清空数据失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0249	400	文件名*%s*无效。	文件名无效。	请将文件名修改正确后，再重试。
Cdm.0250	400	不能操作该路径：*%s*。	不能操作该路径。	请确认是否具有该路径的操作权限，或稍后重试。若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0251	400	向HBase加载数据失败。原因：*%s*。	向HBase加载数据失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0307	400	无法获得请求事务的链接租约，原因：*%s*。	无法获得请求事务的链接租约。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0315	400	连接名*%s*已存在。	该连接已存在。	请指定其他连接名后再重试。

错误码	状态码	错误信息	描述	处理措施
Cdm.0316	400	无法更新不存在的连接。	无法更新不存在的连接。	请指定正确的连接后再重试。
Cdm.0317	400	连接*%s*无效。	连接无效。	请指定正确的连接后再重试。
Cdm.0318	400	作业已存在，无法重复创建。	作业已存在。	请指定其他作业名再重试。
Cdm.0319	400	无法更新不存在的作业。	无法更新不存在的作业。	请确认待更新的作业是否存在，作业名修改正确后再重试。
Cdm.0320	400	作业*%s*无效。	作业无效。	请联系客服或技术支持人员协助解决。
Cdm.0321	400	连接*%s*已被使用。	连接已被使用。	连接已被使用，无法执行当前的操作，请将连接释放后再重试。
Cdm.0322	400	作业*%s*已被使用。	作业已被使用。	请联系客服或技术支持人员协助解决。
Cdm.0323	400	该提交已存在，无法重复创建。	该提交已存在。	您已提交过相同操作的请求，请稍后再重试。
Cdm.0327	400	无效的连接或作业： *%s*。	无效的连接或作业。	请指定正确的连接或作业再重试。
Cdm.0411	400	连接到文件服务器时出错。	连接到文件服务器时出错。	请联系客服或技术支持人员协助解决。
Cdm.0413	400	向文件服务器传输数据时出错。	向文件服务器传输数据时出错。	请联系客服或技术支持人员协助解决。
Cdm.0415	400	从文件服务器下载文件出错。	从文件服务器下载文件出错。	请联系客服或技术支持人员协助解决。
Cdm.0416	400	抽取数据时出错。	抽取数据时出错。	请联系客服或技术支持人员协助解决。

错误码	状态码	错误信息	描述	处理措施
Cdm.0420	400	源文件或源目录不存在。	源文件或源目录不存在。	请确认源文件或源目录是否存在，修改正确后再重试。
Cdm.0423	400	目的路径存在重复文件。	目的路径存在重复文件。	请在目的路径中删除重复文件后再重试。
Cdm.0467	400	连接超时。	连接超时。	请检查IP、主机名、端口填写是否正确，检查网络安全组和防火墙配置是否正确。
Cdm.0501	400	无效的URI[*%s*]。	无效的URI。	请指定正确的URI后，再重试。
Cdm.0518	400	连接HDFS失败。原因：*%s*。	连接HDFS失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0600	400	无法连接FTP服务器。	无法连接FTP服务器。	可能是由于网络不通、安全组或防火墙规则未放行、FTP主机名无法解析、FTP用户名密码错误等原因。若排除上述原因后仍无法解决，请联系客服或技术支持人员协助解决。
Cdm.0700	400	无法连接SFTP服务器。	无法连接SFTP服务器。	可能是由于网络不通、安全组或防火墙规则未放行、SFTP主机名无法解析、SFTP用户名密码错误等原因。若排除上述原因后仍无法解决，请联系客服或技术支持人员协助解决。

错误码	状态码	错误信息	描述	处理措施
Cdm.0800	400	无法连接OBS服务器。	无法连接OBS服务器。	可能是由于OBS终端节点与当前区域不一致、AK/SK错误、AK/SK不是当前用户的AK/SK、安全组或防火墙规则未放行等原因。若排除上述原因后仍无法解决，请联系客服或技术支持人员协助解决。
Cdm.0801	400	OBS桶[*%s*]不存在。	OBS桶不存在。	指定的OBS桶可能不存在或不在当前区域，请指定正确的OBS桶后再重试。
Cdm.0831	400	无法连接到KODO服务器。原因: *%s*。	无法连接到KODO服务器。	请联系客服或技术支持人员协助解决。
Cdm.0900	400	表[*%s*]不存在。	表不存在。	请指定正确的表名后再重试。
Cdm.0901	400	无法连接数据库服务器。原因： *%s*。	无法连接数据库服务器。	请联系客服或技术支持人员协助解决。
Cdm.0902	400	SQL语句无法执行。原因*%s*。	SQL语句无法执行。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0903	400	元数据获取失败。原因: *%s*。	元数据获取失败。	请确认在集群的“连接管理”页面创建连接时引用符号是否正确或查看数据库表是否存在。若仍无法解决，请联系客服或技术支持人员协助解决。
Cdm.0904	400	从结果中检索数据时发生错误。原因: *%s*。	从结果中检索数据时发生错误。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。

错误码	状态码	错误信息	描述	处理措施
Cdm.0913	400	Schema和SQL不可以同时为空。	Schema和SQL需指定其中一项。	请确认Schema和SQL是否同时为空，请指定其中一项后，再重试。
Cdm.0916	400	增量读取情况下必须指定上次的值。	增量读取时未指定上次的值。	请指定上次的值后再重试。
Cdm.0917	400	缺少字段检查将无法获得上次的值。	缺少字段。	请联系客服或技术支持人员协助解决。
Cdm.0921	400	不支持类型*%s*。	类型不合法。	请指定正确的类型后再重试。
Cdm.0925	400	分区字段含有不支持的值。	分区字段含有不支持的值。	请确认分区字段是否含有不支持的值，修改正确后再重试。
Cdm.0926	400	取不到Schema。原因：*%s*。	获取Schema失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.0927	400	中转表不为空。	中转表不为空。	请指定一个空的中转表后再重试。
Cdm.0928	400	中转表到目的表进行数据迁移时发生错误。	中转表到目的表进行数据迁移时发生错误。	请联系客服或技术支持人员协助解决。
Cdm.0931	400	Schema字段大小[*%s*]与结果集的字段大小[*%s*]不匹配。	Schema字段大小与结果集的字段大小不匹配。	请将Schema字段大小和结果集中的字段大小改为一致后再重试。
Cdm.0932	400	找不到字段最大值。	找不到字段最大值。	请联系客服或技术支持人员协助解决。
Cdm.0934	400	不同Schema/Catalog下有重名表。	不同Schema/Catalog下有重名表。	请联系客服或技术支持人员协助解决。
Cdm.0936	400	错误脏数据条数达到上限。	错误脏数据条数达到上限。	您可以编辑作业，在作业的任务配置中将错误脏数据条数增大。

错误码	状态码	错误信息	描述	处理措施
Cdm.0940	400	表名准确匹配失败。	表名准确匹配失败。	匹配不到表名，请指定正确的表名后再重试。
Cdm.0941	400	无法连接服务器。原因:[*%s*]	无法连接服务器。	请检查IP、主机名、端口填写是否正确，检查网络安全组和防火墙配置是否正确，参考数据库返回消息进行定位。若仍无法解决，请联系客服或技术支持人员协助解决。
Cdm.0950	400	当前认证信息无法连接到数据库。	当前认证信息无法连接到数据库。	认证信息错误，请修改正确后再重试。
Cdm.0962	400	必须指定主机IP。	未指定主机IP。	未指定主机IP，请指定主机IP后，再重试。
Cdm.0963	400	必须指定主机端口。	未指定主机端口。	未指定主机端口，请指定主机端口后，再重试。
Cdm.0964	400	必须指定数据库。	未指定数据库。	未指定数据库，请指定数据库后，再重试。
Cdm.1000	400	Hive表[*%s*]不存在。	Hive表不存在。	请输入正确的Hive表名后，再重试。
Cdm.1010	400	无效的URI *%s*。URI必须为null或有效的URI。	无效的URI。	请输入正确的URI后，再重试。下面是一些URI示例: • hdfs://example.com:8020/ • hdfs://example.com/ • file:/// • file:///tmp • file://localhost/tmp

错误码	状态码	错误信息	描述	处理措施
Cdm.1011	400	连接Hive失败，原因：*%s*。	连接Hive失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1100	400	表[*%s*]不存在。	表不存在。	请确认表是否存在，输入正确的表名后再重试。
Cdm.1101	400	获取连接失败，原因：*%s*。	获取连接失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1102	400	创表失败，原因：*%s*。	创建表失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1103	400	未设置Rowkey。	未设置Rowkey。	请设置Rowkey后再重试。
Cdm.1104	400	打开表格失败。原因：*%s*。	打开表格失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1105	400	作业初始化失败。原因*%s*。	作业初始化失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1111	400	表名不能为空。	表名为空。	请输入正确的表名后，再重试。
Cdm.1114	400	Rowkey为空，请在字段映射步骤重新设置。	Rowkey为空。	请按照错误提示进行处理。
Cdm.1115	400	Columns为空，请在字段映射步骤重新设置。	Columns为空。	请按照错误提示进行处理。

错误码	状态码	错误信息	描述	处理措施
Cdm.1116	400	列名重复，请在字段映射步骤重新设置。	列名重复。	请按照错误提示进行处理。
Cdm.1117	400	判断表格是否存在失败，原因：*%s*。	判断表格是否存在失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1118	400	表*%s*不包含列族*%s*。	表中不包含列族。	请指定列族后再重试。
Cdm.1120	400	表中有数据，请清空表数据或重新设置导入前是否清空表数据配置项。	表中有数据，请清空表数据或重新设置导入前是否清空表数据配置项。	请按照错误提示进行处理。
Cdm.1121	400	关闭连接已失败。原因：*%s*。	关闭连接失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1201	400	不能连接到Redis服务器，原因：*%s*。	无法连接到Redis服务器。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1203	400	从Redis服务器抽取数据失败，原因：*%s*。	从Redis服务器获取数据失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1205	400	Redis值前缀不能为空白符。	Redis值前缀不能为空白符。	请去除Redis前缀前的空白符，然后再重试。
Cdm.1206	400	Redis值存储类型必须指定为“string”或“hash”。	Redis值存储类型必须指定为“string”或“hash”。	请按照错误提示进行处理。
Cdm.1207	400	当值存储类型为“string”时，必须指定值分隔符。	值存储类型为“string”，未指定值分隔符。	请指定分隔符后再重试。

错误码	状态码	错误信息	描述	处理措施
Cdm.1208	400	Redis存储字段列表必须指定。	Redis存储字段列表未指定。	请指定Redis存储字段列表后再重试。
Cdm.1209	400	Redis键分隔符不能为空白符。	Redis键分隔符不能为空白符。	请输入正确的分隔符后，再重试。
Cdm.1210	400	必须指定Redis主键字段列表。	Redis主键字段列表未指定。	请指定Redis主键字段列表后再重试。
Cdm.1211	400	Redis主键字段列表必须在字段列表中存在。	Redis主键字段列表不在字段列表中。	请指定Redis主键字段列表后再重试。
Cdm.1213	400	必须指定Redis服务器列表。	未指定Redis服务器列表。	请指定Redis服务器列表后再重试。
Cdm.1301	400	不能连接到MongoDB服务器，原因：*%s*。	连接到MongoDB服务器失败	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1302	400	从MongoDB服务器抽取数据失败，原因：*%s*。	从MongoDB服务器抽取数据失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1304	400	必须指定MongoDB服务器的集合。	未指定MongoDB服务器的集合。	未指定MongoDB服务器的集合，请指定后，再重试。
Cdm.1305	400	必须指定MongoDB服务列表。	未指定MongoDB服务列表。	未指定MongoDB服务列表，请指定后，再重试。
Cdm.1306	400	必须指定MongoDB服务的数据库名称。	未指定MongoDB服务的数据库名称。	未指定MongoDB服务的数据库名称，请指定数据库后，再重试。
Cdm.1307	400	必须指定MongoDB服务的字段列表。	未指定MongoDB服务的字段列表。	未指定MongoDB服务的字段列表，请指定字段列表后，再重试。

错误码	状态码	错误信息	描述	处理措施
Cdm.1501	400	不能连接到Elasticsearch服务器，原因：*%s*。	无法连接到Elasticsearch服务器	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1502	400	向Elasticsearch服务器写入数据失败，原因：*%s*。	向Elasticsearch服务器写入数据失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1503	400	关闭Elasticsearch连接失败，原因：*%s*。	关闭Elasticsearch连接失败。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1504	400	获取Elasticsearch索引错误，原因：*%s*。	获取Elasticsearch索引错误	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1505	400	获取Elasticsearch类型错误，原因：*%s*。	获取Elasticsearch类型错误。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1506	400	获取Elasticsearch文档字段错误，原因：*%s*。	获取Elasticsearch文档字段错误。	请根据错误提示进行定位，若无法解决，请联系客服或技术支持人员协助解决。
Cdm.1508	400	必须指定Elasticsearch服务器主机名或IP地址。	未指定Elasticsearch服务器主机名或IP地址。	未指定Elasticsearch服务器主机名或IP地址，请指定后，再重试。
Cdm.1510	400	必须指定Elasticsearch索引。	未指定Elasticsearch索引。	当前未指定Elasticsearch索引，请指定后再重试。
Cdm.1511	400	必须指定Elasticsearch类型。	未指定Elasticsearch类型。	当前未指定Elasticsearch类型，请指定后再重试。

错误码	状态码	错误信息	描述	处理措施
Cdm.1513	400	字段列表中必须包含字段类型定义。	字段列表中未包含字段类型定义。	请确认字段列表中是否包含字段类型定义，修改正确后再重试。
Cdm.1514	400	字段列表中必须包含主键字段。	未设置主键字段。	当前未设置主键字段，请设置主键字段后再重试。
Cdm.1516	400	非法列名*%s*。	列名不合法。	请确认列名是否合法，输入正确的列名后再重试。
Cdm.1517	400	获取文档数量产生错误。	获取文档数量产生错误。	请联系客服或技术支持人员协助解决。
Cdm.1519	400	抽取数据错误。	抽取数据错误。	请联系客服或技术支持人员协助解决。
Cdm.1601	400	连接服务器失败。	连接服务器失败。	请联系客服或技术支持人员协助解决。
Cdm.1603	400	获取topic *%s*的样值失败。	获取topic样值失败。	请联系客服或技术支持人员协助解决。
Cdm.1604	400	topic *%s*没有数据。	该topic中无数据。	该topic中无数据，请排查无数据的原因。或者，请改为其他topic后再重试。